ABSTRACT

Title of Dissertation: LEARNING LOW-DIMENSIONAL STRUCTURES IN HIGH-DIMENSIONAL SCIENTIFIC DATA

Shashank Sule
Doctor of Philosophy, 2026

Dissertation Directed by: Professors Maria K. Cameron and Wojciech K. Czaja
Department of Mathematics

This dissertation develops mathematical foundations for data-driven algorithms in three scientific domains: molecular dynamics, magnetic resonance relaxometry, and neural network weight spaces. A unifying theme is that high-dimensional scientific data possess hidden low-dimensional structure, and that principled algorithmic design requires rigorous understanding and exploitation of this structure.

Chapter 2 derives sharp pointwise consistency estimates for the convergence of graph Laplacians in the target-measure diffusion map (TMDmap) algorithm and applies them to the setting of rare events in overdamped Langevin dynamics arising in molecular dynamics (MD) applications. The derived estimates provide bounds on the convergence of solutions to partial differential equations on manifolds and show that this convergence is, in particular, *faster* for the committor problem arising in transition path theory.

Chapter 3 further explores the context of MD simulations by addressing the problem of collective variable discovery, with particular emphasis on dimensionality reduction that preserves dynamics. In

particular, drawing from the quantitative coarse-graining theory of Legoll & Leliévre, we propose an algorithm for devising collective variables (CVs) using the so-called *orthogonality condition*. We validate our algorithm on reproducing transition rates for the n-butane molecule and on coarse-graining the alanine dipeptide. In this algorithm, we develop several new techniques for combining manifold learning algorithms such as diffusion maps, diffusion nets, and conformal autoencoders, and show how group-invariant featurization and functional independence can be crucial for separating timescales and resolving topological artifacts in diffusion map embeddings.

Chapter 4 switches to the context of magnetic resonance imaging, where high-dimensional signals are characterized by a small number of parameters but are highly sensitive to measurement noise. Using *input layer regularization* proposed by Rozowski et al. (2022), we combine the empirical Bayes method with deep learning to provide an input preprocessing technique that optimally informs deep learning using a classical estimator. Our hybrid method improves on both pure deep learning and pure classical estimation and introduces a novel way to combine the generalized cross-validation estimator with deep learning for myelin water fraction estimation in in vivo brain data.

Finally, in Chapter 5, we study the weights of fully connected trained neural networks, which exhibit low-dimensional structure after training. We provide convergence guarantees for *neural network descrambling*, an explainability algorithm that extracts this low-dimensional structure, showing that the singular value decomposition of network weights emerges as an interpretable factorization for deep learning. We show how singular vectors can be used to interpret networks in a variety of settings in mathematical signal processing, including deep electron-electron resonance, magnetic resonance, and phase retrieval.

LEARNING LOW-DIMENSIONAL STRUCTURES IN HIGH-DIMENSIONAL SCIENTIFIC DATA

by

Shashank Sule

Dissertation submitted to the Faculty of the Graduate School of the
University of Maryland, College Park in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
2026

Advisory Committee:
Professor Maria K. Cameron, Chair/Advisor
Professor Wojciech K. Czaja, Co-Advisor
Professor Radu Balan
Professor Haizhao Yang
Professor Pratyush Tiwary

© Copyright by
Shashank Sule
2026

Dedication

To the friends, family, and the versions of myself I lost along the way — and to those I found at the end.

Acknowledgments

The work that appears in this dissertation would not have been possible without a number of individuals who held me up during my PhD. First, I want to acknowledge and thank my co-advisors, Profs. Maria Cameron (Masha) and Wojciech Czaja (Wojtek) for guiding the projects in this dissertation and for contributing enormously to my development as a mathematician. They taught me my first classes in graduate school, and quickly helped me find my footing in the problems that would guide my research program over the next six years. Masha and Wojtek served as my most ardent supporters, and their unwavering belief in me was instrumental in keeping my chin up during the hardest periods of graduate school. In college, I gravitated from the sciences towards mathematics, but in graduate school, I returned somewhat to the sciences as a mathematician participating in interdisciplinary science. I credit Masha and Wojtek for bringing forth this interesting irony in my intellectual journey.

In addition to my co-advisors, I thank Profs. Radu Balan, Haizhao Yang, and Pratyush Tiwary for serving as examiners on this dissertation, and for a cast of collaborators from the National Institutes of Health, including Dr. Richard Spencer and Mirage Modi, whose work was instrumental in Chapter 4. I also want to acknowledge my colleagues at Maryland and elsewhere–Dustin Mixon, Matthias Wellershoff, Fushuai Jiang, Margot Yuan, Brandon Kolstoe and Sanghoon Na. Special thanks go to Luke Evans for his mentorship and collaboration during my first paper in graduate school.

The above academic and intellectual contributions only tell a part of the story. The rest of the story is about how my friends and family helped me keep myself sane during my PhD. To this end, I want to thank my friends at the Maryland math department–Meenakshi Krishnan, Max Springer, Liam Fowl, Maeve Wildes, Michael Rozowski, Stavros Papathanasiou, Spiros Lentas, Stratos Tsoukanis, Luke Evans, and Lora Adams for their humour, support, and validation that made the journey far more enjoyable than the destination. I started at a dark time–in the deepest throes of the COVID

pandemic–and it could've gone very wrong when I moved into a house with three strangers. But over the following year, those three strangers, Noah Chrein, Jermain McDermott, and Ishfaaq Imtiyas, became a big part of the reason I felt settled in College Park. As society reopened, I reconnected with friends living in DC from my past life who would become indispensable parts of my time here–Annabel Yau, Spandan Marasini, Mrugank Bhusari, Annika List, Nasiha Salwati, John Michael, Lumi Youm. Shyam Pradheep, Nirvaan Khera, Soham Patki, Sharvari Karandikar, Parth Nirgudkar and Tanvi Patil supported me despite being oceans apart. I thank them all, from the bottom of my heart. I also owe an immense amount of gratitude to Lizzie and Nadige, who helped me keep myself mentally and physically healthy during my time at Maryland.

I am thankful to my family–my brother Shantanu, my sister in law Abhidnya and my nephew Ruhan, for their unconditional love–they make me believe that I will never be alone. My aunt and uncle Rajani Nadkarni and Vivek Kamath also deserve a special mention for always being available at the drop of a hat, their love expressed in equal parts through words and acts of service. I am grateful to my parents, Sandhya Sule and Virendra Sule. I continue to learn from the example they set, and their love continues to help me strive to be the best person I can be. Finally, I want to thank my significant other, Gismarie Bermudez Mendez, for being my biggest supporter and for showing me how to live with generosity, without fear of the unknown, and how to love one's own life for what it *is* instead of what it could've been.

At last, I would like to acknowledge the one person without whom this dissertation would have literally not existed, and here I borrow a famous quote [1]: "I want to thank *me*. I want to thank me for believing in me, I want to thank me for doing all this hard work...I want to thank me for being *me* at all times."

[1] Dogg, 2018.

Table of Contents

Dedication ii

Acknowledgements iii

Table of Contents v

List of Tables viii

List of Figures ix

List of Symbols x

Introduction xxiii

Chapter 1: Background 1
- 1.1 Relevant differential geometry. . . . 1
  - 1.1.1 Riemannian Metric and Geodesics . . . 2
  - 1.1.2 Exponential Map and Normal Coordinates . . . 6
  - 1.1.3 Maps Between Manifolds . . . 9
  - 1.1.4 Intrinsic Differential Operators . . . 13
- 1.2 Stochastic Differential Equations and Diffusion Processes . . . 15
  - 1.2.1 Itô SDEs in $\mathbb{R}^d$ . . . 15
  - 1.2.2 The Martingale Problem . . . 16
  - 1.2.3 Diffusions on Riemannian Manifolds . . . 17
  - 1.2.4 Overdamped Langevin Dynamics . . . 19
- 1.3 Analysis of rare events via Transition Path Theory . . . 20
  - 1.3.1 The Committor Function . . . 21
  - 1.3.2 Quantities of Interest . . . 22
- 1.4 Data-driven algorithms . . . 24
  - 1.4.1 Diffusion maps . . . 24
  - 1.4.2 Neural Networks . . . 28

Chapter 2: Learning diffusion operators on manifolds 36
- 2.1 Introduction . . . 36
  - 2.1.1 Rare events, the committor problem, and the TMDmap . . . 38
  - 2.1.2 The goal of this chapter and summary of main results . . . 41
  - 2.1.3 Related work . . . 45
- 2.2 Results . . . 47
  - 2.2.1 Problem setup . . . 47
  - 2.2.2 Bias Error . . . 48
  - 2.2.3 Reducing bias error . . . 49
  - 2.2.4 Variance Error . . . 50

2.2.5 Typical error in the kernel density estimate . . . 58
2.2.6 Kernel density estimators for a regular triangular grid . . . 60
2.2.7 Total error model . . . 63
2.2.8 Numerical solution of Dirichlet BVPs with TMD maps . . . 64
2.3 Results: proofs . . . 66
2.3.1 Bias Error: proofs . . . 66
2.3.2 Variance error: proofs . . . 73
2.3.3 Numerical solution of BVPs: proofs . . . 94
2.4 Test problems . . . 101
2.4.1 Illustrating error model . . . 101
2.4.2 Calculating the committor function . . . 107
2.5 Conclusion . . . 119

Chapter 3: Learning collective variables 122
3.1 Introduction . . . 122
3.1.1 Aims and contributions . . . 126
3.1.2 Related work . . . 129
3.1.3 Organization . . . 131
3.2 A brief overview of quantitative coarse-graining . . . 132
3.2.1 Collective Variables . . . 132
3.2.2 Measuring the quality of effective dynamics . . . 135
3.3 Proposed methodology for learning CVs . . . 147
3.3.1 Conditions (OC) and (POC) are equivalent . . . 148
3.3.2 The diffusion tensor may be rank-deficient but must not vanish . . . 150
3.3.3 Proposed algorithm . . . 154
3.4 Learning the residence manifold . . . 156
3.4.1 Data post-processing matters . . . 157
3.4.2 Diffusion maps . . . 159
3.4.3 Independent eigencoordinate selection (IES) . . . 160
3.4.4 Data generation and results on selecting diffusion coordinates . . . 165
3.4.5 Diffusion net . . . 168
3.4.6 Undoing spurious topological features . . . 169
3.4.7 Computational details . . . 171
3.5 Learning CVs from the residence manifold . . . 172
3.5.1 Learning the confining potential . . . 172
3.5.2 Learning collective variable . . . 174
3.5.3 Computational details . . . 176
3.5.4 Reproducing transition rates . . . 178
3.5.5 Attributing differences in rates to scale separation . . . 180
3.5.6 Interpreting the collective variables and embeddings . . . 181
3.6 Application to alanine dipeptide . . . 188
3.7 Discussion . . . 189
3.8 Conclusion . . . 195
3.9 Acknowledgments . . . 195

Chapter 4: Learning the low-dimensional structure of magnetic resonance relaxometry measurements in the human brain 196
4.1 Introduction . . . 196
4.1.1 Contributions . . . 198
4.1.2 Related work . . . 200
4.1.3 Chapter organization . . . 202
4.2 Mathematical formalism . . . 203
4.2.1 Biexponential signal analysis . . . 203
4.2.2 Parameter Estimation . . . 205
4.2.3 Regularization . . . 206
4.2.4 Regularization Parameter Selection . . . 207
4.2.5 Neural networks for parameter estimation . . . 208
4.2.6 Input Layer Regularization . . . 209
4.3 Methods . . . 213
4.3.1 Selecting the hyperparameter with a neural network followed by parameter estimation . . . 214
4.3.2 Selecting the hyperparameter with GCV followed by parameter estimation . . . 215
4.3.3 Computational details . . . 216
4.4 Results . . . 218
4.4.1 Parameter Estimation . . . 218
4.4.2 Myelin water fraction estimation . . . 226
4.5 Discussion . . . 232

Chapter 5: Emergence of the singular value decomposition in explaining neural networks 245
5.1 Introduction . . . 245
5.1.1 Neural Network Descrambling . . . 245
5.1.2 Goals of this chapter and summary of results . . . 248
5.1.3 Related work . . . 250
5.2 Results . . . 252
5.2.1 Neural network descrambling: general results . . . 255
5.2.2 Variations in input distribution and network architecture . . . 258
5.2.3 NN descrambling in the presence of training . . . 261
5.2.4 Descrambling as rearrangement of neurons . . . 265
5.3 Numerical results . . . 266
5.3.1 Non-uniqueness of descramblers . . . 267
5.3.2 Assessing the validity of interpretation . . . 269
5.3.3 Discovering motifs within singular vectors of higher layers . . . 271
5.3.4 Limitations of NN descrambling . . . 273
5.4 Explaining input layer regularization . . . 274
5.4.1 Improving training with explainability . . . 276
5.5 Understanding neural STFT phase retrieval . . . 279

Conclusion 285

Bibliography 289

List of Tables

1.1 Summary of the Dmap and TMDmap algorithms. The kernel k_ϵ is given by (1.60). . . . 29

2.1 Measuring the bias error prefactor in a 1-D example 107

3.1 Anti-gauche transition rates for collective variables 153
3.2 Architectures, optimizers, and hyperparameters for the neural networks trained in the paper. 177
3.3 Transition rates for different CVs 181
3.4 Regression results for different feature maps and diffusion coordinates 184

4.1 Glossary of abbreviations used in the chapter 204
4.2 Parameters defined for dataset creation 236
4.3 Earth mover's distances between learned distribution and true distribution of λ_{oracle} . . 237
4.4 RMSE in MWF for synthetic data 237
4.5 RMSE for T_{21} performance on synthetic data 238
4.6 RMSE for T_{22} performance on synthetic data 238
4.7 RMSE performance in MWF estimation on in-vivo data 242
4.8 Percentage improvement of the best performing method (ND, REG)$_{\text{GCV}}$ over the respective alternative method. 243

5.1 Measuring the relative strength of bandpass filters discovered by various methods for computing the descramblers 269
5.2 Explainable training for two-parameter biexponential recovery problem with ILR . . . 284

List of Figures

1 Map of the dissertation . . . xxx

1.1 Spherical coordinates on the sphere . . . 2
1.2 The tangent space to $p \in \mathbb{S}^2$ . . . 8
1.3 Illustrating the differential . . . 11
1.4 Transition path theory . . . 21
1.5 Committor function for Müller's potential . . . 22
1.6 Reactive current for Müller's potential . . . 23

2.1 Plotting the scaling law between ϵ and n . . . 58
2.2 Scaling of optimal ϵ on hexagonal grid . . . 62
2.3 Visualizing the 1-D potential . . . 103
2.4 Comparing uniform and non-uniform sampling densities on the unit circle . . . 105
2.5 Two choices of functions on the unit circle . . . 105
2.6 Measuring the bias prefactor in a 1 dimensional example . . . 108
2.7 Müller's potential landscape . . . 114
2.8 RMSE vs ϵ for Müller's potential, different spatial uniformities . . . 115
2.9 RMSE vs ϵ for Müller's potential, different sampling densities . . . 115
2.10 Two-well potential landscape . . . 116
2.11 RMSE vs ϵ for the two-well potential, different spatial uniformities . . . 117
2.12 RMSE vs ϵ for the two-well potential, different sampling densities . . . 118

3.1 Illustrating the orthogonality condition . . . 126
3.2 Training phase of main algorithm . . . 130
3.3 Inference phase of main algorithm . . . 131
3.4 A chain-like butane molecule . . . 151
3.5 Coarse graining butane trajectories with the dihedral angle . . . 152
3.6 Comparing the free energies and diffusion tensors for CVs based on the dihedral angle 153
3.7 Illustrating BondAlign(1,2) . . . 159
3.8 Illustrating BondAlign(2,3) . . . 159
3.9 Illustrating PlaneAlign . . . 160
3.10 Plotting HyperSurface$(D+1)$ against $D+1$ for different feature maps . . . 165
3.11 Comparing different diffusion map embeddings . . . 167
3.12 Visualizing the first two selected Laplacian eigenvectors for PlaneAlign and GramMatrixCarbon . . . 168
3.13 Visualizing the topological defects in the BondAlign(2,3) embedding . . . 171
3.14 Gradients of surrogate potential . . . 174
3.15 Gradients of learned CVs . . . 175
3.16 The learned CV correlates with the dihedral angle . . . 176
3.17 Visualizing the metastable states in BondAlign(2,3) . . . 178
3.18 Free energy and diffusion tensor for BondAlign(2,3) . . . 179
3.19 Free energy and diffusion tensor for PlaneAlign. . . . 180

3.20 Measuring the spectral gap in the generators in collective variables 182
3.21 Schematic of the alanine dipeptide molecule 190
3.22 Diffusion map embedding on PlaneAlign featurization of alanine dipeptide 191
3.23 Diffusion map embedding on GramMatrix featurizations of alanine dipeptide 192

4.1 (ND, REG) network architecture 212
4.2 Comparing three approaches to parameter estimation 213
4.3 λ_{NN} architecture 217
4.4 Lambda estimates for three different estimation strategies 219
4.5 Distributions of the estimated hyperparameter 220
4.6 Differences in relative RMSE for (ND, REG)$_{\text{GCV}}$ vs (ND, ND) mapped over T_2 space 224
4.7 Differences in relative RMSE for (ND, REG)$_{\text{NN}}$ vs (ND, ND) mapped over T_2 space . 225
4.8 Visualizing the 5-parameter model estimates of the MWF with pixel-wise filtering via AIC 239
4.9 Comparison of c_1 estimation according to the two methods of λ selection and the MI-ML benchmarking model for 41 year old male. 240
4.10 Comparison of c_1 estimation according to the two methods of λ selection and the MI-ML benchmarking model for 79 year old male 241
4.11 SNRs of brain voxels 244

5.1 A graphical illustration of Theorem 5.2.4 251
5.2 Comparing descramblers computed via four different strategies 269
5.3 Comparing descrambled weights computed via four different strategies 270
5.4 Validating the interpretation of the first layer 271
5.5 Emergence of chebyshev polynomials 272
5.6 Comparing descrambling before and after the nonlinearity 274
5.7 Comparing the right singular vectors of the first layer weights of (ND, ND) and (ND, REG) networks 277
5.8 Singular vectors of first layer weights for $m = d, d/2$. 282
5.9 Singular vectors of the singular matrices formed by rearranging the SVD of the first layer weights from Figure 5.8 283

List of Symbols

Symbol	Meaning	First Appearance
$\mathcal{M}$	Compact d-dimensional Riemannian manifold	—
$\mathbb{R}^m$	Ambient Euclidean space containing $\mathcal{M}$	—
$\mathbb{R}^d$	d-dimensional Euclidean space	—
$\mathbb{R}$	Real numbers	—
$\mathbb{N}$	Natural numbers	—
m, N	Ambient dimension of the data	—
$T_x\mathcal{M}$	Tangent space of $\mathcal{M}$ at x	—
$\mathcal{X}(n)$	Point cloud of n points sampled from ρ	—
$\{x_j\}_{j=1}^n$	Enumerated points of the point cloud	—
$A,\ B$	Disjoint reactant and product sets in $\mathcal{M}$	(1.54)
$\mathcal{M}_{AB}$	$\mathcal{M}\backslash(A \cup B)$, the transition region	—
Σ_{AB}	Dividing surface separating A and B	(1.57)
Ω	Closed subset of $\mathcal{M}$ with C^1 boundary	(2.8)
X_t	Stochastic process following overdamped Langevin dynamics	(1.51)
W_t	Brownian motion	(1.51)
$\mathcal{L}$	Backward Kolmogorov operator (infinitesimal generator)	(1.52)

Continued on next page

Symbol	Meaning	First Appearance
$\mathcal{L}_{\epsilon,\mu}$	Continuous integral generator (LLN limit of $L^{(n)}_{\epsilon,\mu}$)	(1.74)
$L^{(n)}_{\epsilon,\mu}$	Discrete TMDmap generator (with discrete KDE)	(1.72)
$L_{\epsilon,\mu}$	Discrete generator with continuous KDE	(2.34)
$L^{\mathcal{II}}$, $L^{\mathcal{ID}}$, $L^{\mathcal{DI}}$, $L^{\mathcal{DD}}$	Block submatrices of the generator L	(2.7)
$P^{(n)}_{\epsilon,\mu}$	Discrete Markov operator	(1.72)
$\mathcal{P}_{\epsilon,\mu}$	Continuous Markov operator (LLN limit of $P^{(n)}_{\epsilon,\mu}$)	(1.73)
$\mathcal{K}_{\epsilon,\mu}$	Continuous normalized integral operator	(1.73)
$\mathcal{G}_{\epsilon}$	Unnormalized integral operator	(2.21)
$\mathcal{Q}$	Fourth-order differential operator on $\mathcal{M}$	(2.21)
Δ, $\Delta_{\mathcal{M}}$	Laplace–Beltrami operator on $\mathcal{M}$	(1.38)
∇, $\nabla_{\mathcal{M}}$	Gradient operator on $\mathcal{M}$	(1.37)
∂^i_i, ∂^j_j, ∂^{ii}_{ii}	Partial derivative operators in index notation (flat case)	(2.22)
$K^{(n)}_{\epsilon}$	Symmetric kernel matrix, $[K^{(n)}_{\epsilon}]_{ij} = k_\epsilon(x_i, x_j)$	(1.60)
$K^{(n)}_{\epsilon,\mu}$	Normalized kernel matrix	(1.68)
$D^{(n)}_{\epsilon}$	Diagonal KDE matrix	(1.61)
$D^{(n)}_{\epsilon,\mu}$	Diagonal row-sum matrix for $K^{(n)}_{\epsilon,\mu}$	(1.72)
$M^{(n)}_{\mu}$	Diagonal matrix with entries $\mu^{1/2}(x_i)$	(1.68)
ρ	Sampling density	—
μ	Target Gibbs measure/density, $\mu \propto \exp(-\beta V)$	—

Continued on next page

Symbol	Meaning	First Appearance
$\exp_x$	Exponential map at x	—
$k_\epsilon(x, y)$	Gaussian kernel function	(1.60)
$k_{\epsilon,\mu}(x, y)$	Normalized kernel, $k_\epsilon(x, y)\mu^{1/2}(y)/\rho_\epsilon(y)$	(1.70)
$\rho_\epsilon(x)$	Continuous kernel density estimate (KDE)	(1.62)
$\rho_\epsilon^{(n)}(x)$	Discrete kernel density estimate (KDE)	(1.61)
q	Exact committor function	(1.54)
$q_{n,\epsilon}$	TMDmap numerical solution to the committor problem	(2.3)
u	Exact solution to the BVP	(2.8)
$\mathbb{I}_\Omega$	Indicator function of Ω	—
$\mathcal{B}_1[f, \mu](x)$	First bias error prefactor	(2.14)
$\mathcal{B}_2[f, \mu, \rho](x)$	Second bias error prefactor	(2.15)
$\mathcal{B}_3[f, \mu, \rho](x)$	Third bias error prefactor	(2.16)
ϵ	Kernel bandwidth parameter	(1.60)
β	Inverse temperature	(1.51)
n	Number of points in the point cloud	—
d	Intrinsic dimension of the manifold	—
n_r	Grid resolution parameter for hexagonal grid	(2.60)
$\rho_{\min}$, $\rho_{\max}$	Lower and upper bounds on sampling density ρ	—
ν_{AB}	Transition rate from A to B	(1.57)

Continued on next page

Symbol	Meaning	First Appearance
k_{AB}	Escape rate from A to B	(1.59)
ρ_A	Probability that trajectory last hit A rather than B	(1.58)
N_{AB}	Number of transitions from A to B in time T	(1.57)
T_A	Total time trajectory last hit A in interval of length T	(1.59)
ξ	Collective variable (CV), $\xi : \mathbb{R}^{3N} \to \mathbb{R}^d$	—
Y_t	Projected dynamics, $Y_t = \xi(X_t)$	(3.1)
Z_t	Effective dynamics process in CV space	(3.3)
$\widehat{Z}_t$	Free energy dynamics, equivalent to Z_t under Assumption 3.2.2	(3.41)
$b(z)$	Effective drift of Z_t	(3.4)
$M(z)$	Effective diffusion tensor of Z_t	(3.5)
$D\xi$	Jacobian of ξ	(3.1)
ε	Scale separation constant	(3.6)
V_0	Driving potential in scale-separated decomposition	(3.6)
V_1	Confining term in scale-separated decomposition	(3.6)
Ω_z	Level set $\{\xi = z\}$	(3.11)
μ^ξ	Pushforward measure of μ under ξ	(3.15)
μ_z	Conditional measure of μ on level set Ω_z	(3.16)
N_z	Normalization constant on Ω_z	(3.14)

Continued on next page

Symbol	Meaning	First Appearance
$f_\mu(z)$	Free energy in CV space	(3.17)
$F_z(x)$	Local mean force	(3.19)
Π	Projection operator induced by ξ	(3.32)
Φ	Matrix $D\xi D\xi^\top$ used in projection operator	(3.32)
$\mathcal{L}_0$, $\mathcal{L}_1$	Decomposition of generator along and orthogonal to level sets	(3.32)
κ	Lipschitz constant of local mean force	(3.25)
λ_ξ	Variation constant of $D\xi D\xi^\top$ on level sets	Assumption 3.2.8
ρ_{LSI}	Log-Sobolev constant	(3.22)
κ_1, κ_2	Lipschitz constants in drift and diffusion for pathwise estimates	(3.32)
d_z	Geodesic distance on level set Ω_z	Assumption 3.2.7
δ_ξ	Lower bound on singular values of $D\xi$	Assumption 3.2.2
$\tilde{A}$, $\tilde{B}$	Metastable sets in CV space	after (3.42)
$\tilde{q}$	Committor function in CV space	(3.42)
$\tilde{\nu}_{AB}$	Transition rate estimated in CV space	(3.43)
θ	Dihedral angle in butane	Section 3.3.2.1
$\mathcal{F}$	Group-invariant feature map, $\mathcal{F} : \mathbb{R}^{3N} \to \mathbb{R}^{d_\mathcal{F}}$	Section 3.4.1
$d_\mathcal{F}$	Dimension of feature space	Section 3.4.1
$\mathcal{M}_\mathcal{F}$	Featurized manifold $\mathcal{F}(\mathcal{M})$	Section 3.4

Continued on next page

Symbol	Meaning	First Appearance
$\mathcal{X}_{\mathcal{F}}$	Featurized point cloud $\{\mathcal{F}(x_i)\}$	Section 3.4
p_i	Featurized data point $p_i = \mathcal{F}(x_i)$	Section 3.4
Ψ	Diffusion net parameterization $\mathcal{M} \to \widehat{\mathcal{M}} \subseteq \mathbb{R}^{D+1}$	Algorithm 3
$\widehat{\mathcal{M}}$	Surrogate manifold $\Psi(\mathcal{M})$ in $\mathbb{R}^{D+1}$	Algorithm 3
D	Intrinsic dimension of surrogate manifold (embedding dimension $D+1$)	Algorithm 3
$\widehat{\Phi}$	Surrogate potential vanishing on $\widehat{\mathcal{M}}$	Algorithm 3
$\widehat{\xi}$	Collective variable in surrogate space $\mathbb{R}^{D+1}$	Algorithm 3
S^*	Optimal subset of eigencoordinate indices from IES	Algorithm 5
S	Final hypersurface coordinate index set, $S = S^* \cup \{k\}$	Algorithm 6
$R_\zeta(S')$	IES volume score for coordinate subset S'	(3.65)
ζ	Regularization parameter for eigenfrequency penalty in IES	(3.65)
$\mathsf{HyperSurface}(D+1)$	Score measuring quality of codimension-1 embedding	(3.66)
$\widehat{n}(y)$, $\widehat{n}_i$	Unit normal vector to $\widehat{\mathcal{M}}$ at y (or at y_i)	Algorithm 8
$\widehat{\mathcal{X}}$	Point cloud on surrogate manifold, $\{y_i\} = \{\Psi(x_i)\}$	Section 3.5
y_i	Point on surrogate manifold, $y_i = \Psi(x_i)$	Section 3.5
$\mathcal{E}(\Psi)$	Conformal autoencoder energy for functional independence	(3.70)

Continued on next page

Symbol	Meaning	First Appearance
L_{DNet}	Diffusion net training loss	(3.68)
L_{eikonal}	Eikonal loss for learning surrogate potential	(3.73)
$L_{\text{potential}}$	Full loss for learning $\widehat{\Phi}$ including normal alignment	(3.74)
$L_{\text{alignment}}$	Loss for learning CV $\widehat{\xi}$ via orthogonality	(3.76)
L_{LAPCAE}	LAPCAE autoencoder training loss	(3.72)
$\Theta_1, \Theta_2, \Theta_3$	Neural network parameters for Ψ, $\widehat{\Phi}$, $\widehat{\xi}$ respectively	Algorithm 3
α_{DNet}, α_{OC}, α_{LAPCAE}	Loss weighting hyperparameters	(3.68) (3.76) (3.72)
T_2	Transverse relaxation time	Section 4.1
$p(T_2)$	Probability density of T_2 times	Section 4.1
$T_{2,1}$, $T_{2,2}$	Short and long T_2 decay constants (myelin and non-myelin water)	(4.2)
c_1, c_2	Fractional amplitudes of the two water components; $c_2 = 1 - c_1$	(4.2)
$s(t)$	Continuous biexponential signal at echo time t	Section 4.1
$\mathbf{p}$	Parameter vector $(c_1, T_{2,1}, T_{2,2})$	(4.3)
$\mathbf{G}(\mathbf{p})$	Noiseless biexponential model vector evaluated at $\mathbf{p}$, $\mathbf{G}(\mathbf{p}) \in \mathbb{R}^m$	(4.3)
$\{t_i\}_{i=1}^m$	Discrete echo times (acquisition times); m denotes the ambient (signal) dimension	(4.3)
η_1, η_2	Independent zero-mean Gaussian random variables forming the Rician noise model	(4.4)

Continued on next page

Symbol	Meaning	First Appearance
$\mathbf{s}$	Observed noisy signal vector, $\mathbf{s} \in \mathbb{R}^m$	(4.4)
$\mathbf{p}_{true}$	True underlying parameter vector generating $\mathbf{G}$	Section 4.2.1
$\mathbf{p}^*$	NLLS parameter estimate	(4.5)
λ	Tikhonov regularisation hyperparameter, $\lambda \geq 0$	(4.7)
B	Diagonal scaling matrix $\mathrm{diag}(100, 1, 1)$ equalising parameter scales	(4.8)
$\mathbf{p}^*_\lambda(\mathbf{s})$	TR-NLLS parameter estimate at regularisation level λ	(4.7)
$\lambda_{\text{oracle}}(\mathbf{s})$	Optimal (oracle) regularisation parameter	(4.9)
$\lambda_{\text{NN}}(\mathbf{s})$	Neural-network estimate of λ_{oracle}	Section 4.2.4
$\lambda_{\text{GCV}}(\mathbf{s})$	GCV-based estimate of λ_{oracle}	Section 4.2.4
L_λ	L^1 loss function for training λ_{NN}	(4.22)
L_{ILR}	Weighted MSE loss function for training the (ND, REG) parameter-estimation NN	(4.23)
$\|\cdot\|_B^2$	Weighted squared norm, $\|x\|_B^2 = x^\top B x$	(4.23)
$\mathbf{x}$	Concatenated ILR input vector $[\mathbf{s},\ \mathbf{G}(\mathbf{p}^*_{\lambda(\mathbf{s})}(\mathbf{s}))]$	(4.12)
$J(\lambda)$	Jacobian matrix $D\mathbf{G}(\mathbf{p}^*_\lambda(\mathbf{s}))$, used in the GCV criterion	(4.24)
$\varphi(\mathbf{s}, \lambda(\mathbf{s}))$	Empirical Bayes point estimator of $\mathbf{p}$; instantiated by the (ND, REG) network	Section 4.2.6.1
$\mathcal{W}_1$	Wasserstein-1 (earth mover's) distance between distributions of λ estimates	Table 4.3

Continued on next page

Symbol	Meaning	First Appearance
$\mathbf{W}_k$	Weight matrix of the kth layer of the NN under analysis	(5.1)
n_0, n_L	Input and output dimensions of the network f	(5.1)
n_k	Number of neurons in the kth layer	(5.5)
$\mathbf{X}$	Input data matrix, $\mathbf{X} \in \mathbb{R}^{n_0 \times n}$	(5.2)
$\mathbf{Y}$	Output data matrix, $\mathbf{Y} \in \mathbb{R}^{n_L \times n}$	Section 5.1.1
n	Number of data columns / training samples	Section 5.1.1
$h_k(x)$	Post-activation hidden-layer output at layer k	(5.20)
$f_k(\mathbf{X})$	Preactivation matrix at layer k	(5.2)
$\mathbf{F}$	Network output matrix after identity insertion, $\mathbf{F} = \mathbf{W}_L \cdots \phi \mathbf{P}^{-1}\mathbf{P} f_k(\mathbf{X})$	(5.2)
$\mathbf{P}$	Descrambler matrix	(5.2)
$\widehat{\mathbf{P}}$	Optimal descrambler	(5.3)
$\widehat{\mathbf{P}}_{SC}$	Smoothness-criterion descrambler	(5.5)
$\tilde{\mathbf{P}}_{SC}$	Post-nonlinearity smoothness descrambler	(5.45)
η	Generic explainability loss functional	(5.3)
η_{SC}	Smoothness criterion loss	(5.4)
G	Matrix group	(5.3)
$O(n_k)$	Orthogonal group of $n_k \times n_k$ real matrices	(5.5)
$\mathbf{D}$	Finite-difference or spectral differentiation matrix	(5.4)

Continued on next page

Symbol	Meaning	First Appearance
$\widehat{\mathcal{W}_k}$	Limiting descrambled weight matrix, $\widehat{\mathcal{W}_k}x$	Theorem 5.2.6
$T_m(l)$	(m, l) entry of the Hartley transform matrix	(5.21)
$\mathbf{\Sigma}$	Population autocorrelation matrix of the preactivations	Theorem 5.2.4
$\mathbf{U}$	Eigenvector matrix of $\mathbf{\Sigma}$	Theorem 5.2.4
$\mathbf{\Lambda}$	Diagonal matrix of eigenvalues of $\mathbf{\Sigma}$	Theorem 5.2.4
$\mathbf{U}_n$	Eigenvector matrix of $\mathbf{\Sigma}_n$	Theorem 5.2.4
u^j	jth eigenvector of $\mathbf{U}$	Theorem 5.2.4
u_n^j	jth eigenvector of $\mathbf{U}_n$	Theorem 5.2.4
$\mathbf{U}_k$, $\mathbf{\Lambda}_k$, $\mathbf{V}_k$	Left singular vectors, singular values, and right singular vectors of $\mathbf{W}_k$ respectively	Theorem 5.2.6
v_i^k	ith right singular vector of $\mathbf{W}_k$	(5.22)
λ_i^k	ith singular value of $\mathbf{W}_k$.	(5.22)
α_{SNR}	SNR scaling parameter	Theorem 5.2.9
z	Latent signal random variable	Theorem 5.2.9
$s(z)$	Signal function	Theorem 5.2.9
ς	Zero-mean isotropic noise vector	Theorem 5.2.9
$\mathbf{\Sigma}_\alpha$	Autocorrelation matrix of the rescaled hidden data αf_k	Theorem 5.2.9
$\mathbf{\Sigma}_0$	Limiting autocorrelation as $\alpha \to 0$, $\mathbf{\Sigma}_0 = \mathbf{W}_k\mathbf{W}_k^\top$	Theorem 5.2.9
$\mathbf{U}_\alpha$, $\mathbf{U}_0$	Eigenvector matrices of $\mathbf{\Sigma}_\alpha$ and $\mathbf{\Sigma}_0$ respectively	Theorem 5.2.9

Continued on next page

Symbol	Meaning	First Appearance
$\mathcal{L}_{\text{MSE}}$	MSE training loss	(5.25)
τ	Learning rate	(5.26)
$\mathbf{\Sigma}_n^{xx}$	Sample input–input autocorrelation matrix	(5.28)
$\mathbf{\Sigma}_n^{xy}$	Sample input–output cross-correlation matrix	(5.28)
$\mathbf{V}_n^x$, $\mathbf{\Lambda}_n^x$	Eigenvectors and eigenvalues of $\mathbf{\Sigma}_n^{xx}$	(5.28)
$\mathbf{U}_n^{xy}$, $\mathbf{\Lambda}_n^{xy}$	Left singular vectors and singular values of $\mathbf{\Sigma}_n^{xy}$	(5.28)
$\mathbf{R}$	Arbitrary fixed orthogonal matrix used at initialization	(5.29)
$\mathbf{\Lambda}_1(t)$, $\mathbf{\Lambda}_2(t)$	Diagonal matrix-valued functions describing the training dynamics of $\mathbf{W}_1(t)$ and $\mathbf{W}_2(t)$	(5.30)
$\mathbf{\Lambda}_1'(t)$, $\mathbf{\Lambda}_2'(t)$	Diagonal matrix-valued functions in the descrambled weights	(5.31)
$\mathbf{K}$	Discretised integral kernel	(5.35)
ϑ	Scalar variance parameter	(5.35)
$\mathbf{\Sigma}^{xy}$, $\mathbf{\Sigma}^{xx}$	Population cross- and auto-correlation matrices;	(5.35)
$\mathbf{U}_K$, $\mathbf{\Lambda}_K$	Left singular vectors and singular values of $\mathbf{K}$	Section 5.3.2
$\mathbf{U}_1$, $\mathbf{\Lambda}_1$, $\mathbf{V}_1$	SVD factors of $\mathbf{W}_1$ used in the DEERNet numerical validation	Section 5.3.2
$\mathbf{\Pi}$	Permutation matrix	Proposition 5.2.12
w	Convolution filter kernel; $\mathbf{W}_k x = w * x$	Proposition 5.2.12
$\mathbf{\Sigma}_n^h$	Autocorrelation matrix of the hidden data $h_k(\mathbf{X})$	Proposition 5.2.12

Continued on next page

Symbol	Meaning	First Appearance
$S_d^-(\mathbb{R})$	$d \times d$ skew-symmetric matrices	(5.39)
$\mathbf{Q}$	Skew-symmetric matrix	(5.39)
$\mathbf{P}(\mathbf{Q})$	Cayley transform parameterisation of $O(d)$	(5.39)
$\widehat{\mathbf{Q}}$	Minimiser of the Cayley-reparameterised smoothness problem (5.40)	(5.40)
$G(\mathbf{Q})$	Cayley-reparameterised smoothness objective	(5.40)
$\mathrm{STFT}_{a,M}^{w}(x)$	Spectrogram of signal x with window w, hop length a and number of frequency bins M	(5.47)
L_{PR}	Loss function for phase retrieval networks	(5.48)

Introduction

Perhaps the most significant story in computational mathematics in the 21st century thus far has been the emergence of *data-driven* methods for scientific discovery. Unlike *classical* numerical methods, data driven methods seek to numerically solve the problems of continuous mathematics like optimization, approximation, and solution of differential equations by inferring or *learning* patterns directly from the experimental data that arises in the natural sciences. Due to the stunning leaps in the engineering aspects of machine learning, data driven methods are now not only tractable on modest computational resources but also *desirable*–in that they match classical methods in accuracy, outperform them in terms of speed, and apply to high dimensional problems, where the underlying domain of the data is a Euclidean space of dimension much greater than 3. In this high-dimensional context, data-driven methods are not always faster by design *per se*. Instead their speedup draws from the fact that several high-dimensional problems are intrinsically *low-dimensional*–this perspective is known as the *manifold hypothesis* or the *geometric prior*. Data driven methods either implicitly or explicitly sense this geometric prior and exploit the hidden low dimensional structure of data. However, the theoretical foundations of such methods often lag behind their empirical successes: it is frequently unclear why a given algorithm works, when it will fail, and how it should be designed to exploit the structure of a particular problem. To this end, this dissertation develops mathematical perspectives on the analysis and design of data-driven algorithms in the context of three scientific applications: molecular dynamics, magnetic resonance relaxometry, and neural network weight spaces. While these applications may be disparate, below we describe how they may be united from the perspective of studying computational problems involving the geometric prior.

Molecular Dynamics. Understanding the long-time behavior of molecular systems—how proteins fold, how chemical reactions proceed, how conformational changes occur—is one of the central challenges of computational chemistry and biophysics. All-atom molecular dynamics simulations nominally evolve in a space of tens of thousands of degrees of freedom, yet the dynamics of interest are governed by a small number of slow, collective motions: transitions between metastable states separated by large free energy barriers. These rare events occur on timescales orders of magnitude longer than what direct simulation can reach, making naive simulations intractable. The central task is therefore twofold: to accurately infer transition rates and mechanisms from available (often biased) simulation data, and to identify low-dimensional *collective variables* (CVs) that capture the slow dynamics and enable efficient enhanced sampling. Both tasks are fundamentally problems of learning structure from high-dimensional point cloud data, and graph Laplacian and manifold learning methods have emerged as powerful tools for this purpose. However, the mathematical properties of these methods—their accuracy, stability, and sensitivity to sampling bias—are incompletely understood, limiting principled algorithmic design.

Magnetic Resonance Relaxometry. Quantitative magnetic resonance relaxometry is a powerful non-invasive technique for characterizing tissue microstructure, with applications ranging from the diagnosis of demyelinating diseases such as multiple sclerosis to the assessment of liver fibrosis and cartilage degradation. The underlying biophysical model relates the measured multi-echo signal to a distribution of relaxation times reflecting distinct water compartments within tissue; recovering this distribution is the central computational task. The difficulty is that this inverse problem is severely ill-posed: the forward map from parameters to measurements is poorly conditioned, so small amounts of noise—unavoidable in clinical MRI acquisitions—can produce large errors in estimated parameters.

Classical regularization approaches offer stability at the cost of resolution and systematic bias, while purely data-driven methods can overfit or fail to generalize across scanners and acquisition protocols. The fundamental challenge is therefore to combine the structural knowledge encoded in the biophysical signal model with the flexibility of modern machine learning, in a way that is both theoretically grounded and practically effective.

Neural Network Weight Spaces. As neural networks are increasingly deployed for scientific inverse problems—replacing or augmenting classical solvers in applications from medical imaging to seismology—understanding *what* these networks have learned and *why* they generalize becomes essential. Empirically, hidden-layer weights in trained networks spontaneously organize along smooth, low-dimensional subspaces reflecting both the structure of the training data and the inductive biases of the architecture. Extracting and interpreting these motifs is a prerequisite for auditing model behavior, diagnosing failure modes, and designing architectures with provably good inductive biases for a given problem class. Yet most existing interpretability methods are heuristic and lack theoretical grounding, making it difficult to distinguish genuine learned structure from artifacts of the explanation method itself.

Contributions

This dissertation addresses the challenges above from the perspective of *computational and applied mathematics*. In particular, we ask the questions of numerical analysis—convergence rates, stability, approximation quality, the role of inductive biases—of data driven methods. Each chapter identifies a precise mathematical question, develops theory to answer it, and uses that theory to motivate or improve a concrete algorithm. The contributions proceed chapter by chapter as follows.

In Chapter 2, we ask how accurately graph Laplacian methods approximate the generator of a diffusion process from a point cloud, and what governs the error under the biased sampling that arises in enhanced sampling simulations. We derive sharp, pointwise consistency estimates for the target measure diffusion map (TMDmap) generator, with explicit analytical formulas for bias and variance prefactors. We prove that consistency implies stability for the committor boundary value problem—the key quantity governing transition rates—and identify prefactor cancellations that make TMDmap particularly accurate in the rare event setting. The analysis is extended to quasi-uniform sampling via δ-nets. This chapter is based on [SEC25]:

> Sule, S., Evans, L., & Cameron, M. (2025). Sharp error estimates for target measure diffusion maps with applications to the committor problem. *Applied and Computational Harmonic Analysis*, Vol. 79, 101803.

In Chapter 3, we ask how one should design collective variables that faithfully represent the slow dynamics of a molecular system. We prove that an orthogonality condition on collective variables simultaneously controls the relative entropy and pathwise distances between the projected dynamics and the effective dynamics on the CV space, providing a precise mathematical rationale for orthogonality-based CV design. Motivated by this theory, we propose a three-step algorithm integrating group-invariant featurization, residence manifold learning, and neural-network-based CV parameterization, and introduce the Laplacian Conformal Autoencoder (LAPCAE) to resolve topological artifacts in manifold learning embeddings. The method is validated on n-butane and alanine dipeptide. This chapter is based on [SMC]:

> Sule, S., Mehta, A., & Cameron, M. K. (2025). Learning collective variables that preserve transition rates. To appear in: *SIAM Multiscale Modeling and Simulation*.

In Chapter 4, we ask how domain knowledge about the signal model should be integrated with a learned estimator for myelin water estimation from multi-echo MRI data. We show that input layer regularization (ILR)—a technique that prepends a fixed map to a neural network—can be rationalized as an empirical Bayes estimator, where the fixed map computes an optimally regularized signal and can be considered as a denoised version of the input. This perspective motivates learning the regularization parameter via bilevel optimization rather than setting it by hand, and the resulting method achieves nearly two-fold improvements over classical Tikhonov Regularized Nonlinear Least Squares (TRNLLS) estimators and 10–15% improvements over standard deep learning approaches on clinical brain MRI data. This chapter is based on [MSP+25]:

> Modi, M., Sule, S., Palumbo, J., Rozowski, M., Bouhrara, M., Czaja, W., & Spencer, R. G. (2025). Input layer regularization and automated regularization hyperparameter tuning for myelin water estimation using deep learning. To appear in *NMR in Biomedicine*.

In Chapter 5, we ask what the geometry of trained network weights can reveal about learned computation, providing rigorous foundations for the neural network descrambling algorithm. We characterize the large-sample limit of smoothness descramblers as a composition of PCA projection of hidden-layer activations with expansion in the eigenbasis of a differentiation stencil, explaining the empirical observation that descrambled weights appear sparse in the Fourier domain. This singular vector framework is applied to explain ILR networks and neural STFT phase retrieval networks, and we demonstrate that incorporating signal-model-informed singular vectors into network training improves generalization by at least 40%. This chapter is based on [SSC25]:

> Sule, S., Spencer, R. G., & Czaja, W. (2025). On the limits of neural network explainability via descrambling. *Applied and Computational Harmonic Analysis*, Vol. 79, 101793.

Organization of this dissertation

In Chapter 1, we begin with the mathematical preliminaries used throughout the dissertation. The following four technical chapters are then organized by *application domain*: Chapters 2 and 3 address molecular dynamics, Chapter 4 addresses magnetic resonance relaxometry, and Chapter 5 addresses explainable artificial intelligence (AI). Within this application-first organization, however, each chapter is structured around the interplay of three ingredients: a scientific *application* that motivates the problem, a *machine learning algorithm* that addresses it, and a *mathematical subject* that provides the theoretical tools for analysis and design. The specific combination varies by chapter:

- Chapter 2 combines **molecular dynamics** (application), **diffusion maps** (ML algorithm), and **differential geometry and stochastic differential equations** (mathematics).
- Chapter 3 also addresses **molecular dynamics** (application)–in addition to the subjects in Chapter 2, it proposes an algorithm that heavily utilizes **neural networks** (algorithm).
- Chapter 4 switches to the context of **MRI**, and uses **Estimation Theory** to motivate a new type of data pre-processing for **neural networks** (application).
- Chapter 5 features the analysis of an technique used in **Explainable AI** for explaining **neural networks** using techniques from **matrix analysis**.

Figure 1 illustrates these relationships across the four chapters.

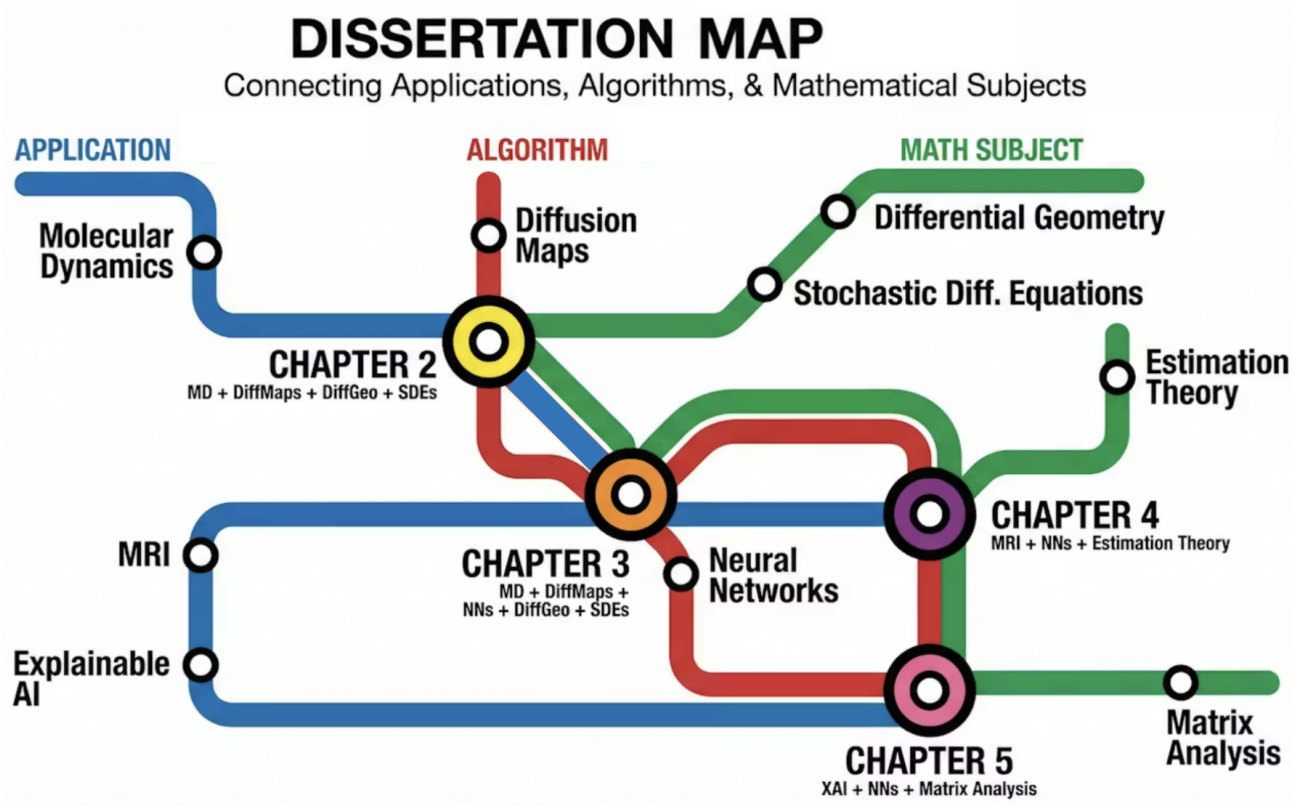


Figure 1: A subway map-style map for reading this dissertation. Each chapter (represented as a junction station) lies at the intersection of an application area (blue), a machine learning algorithm (red), and mathematical subjects (green).

Chapter 1: Background

In this chapter,, we develop precise definitions for the geometric and random phenomena that occur when studying data with an underlying nonlinear low-dimensional structure. These definitions will be especially pertinent in Chapters 2 & 3, where the algorithm we use, diffusion maps, makes heavy use of randomness and geometry. Moreover, we also discuss neural networks–the key data-driven method utilized in Chapters 4 & 5.

1.1 Relevant differential geometry.

In Chapters 2 & 3 we study the situation where the data lies on a Riemannian manifold, a natural generalization of standard Euclidean space that includes the notions of length, distance, curvature. For instance, consider the unit sphere

$$\mathbb{S}^2 = \big\{(x, y, z) \in \mathbb{R}^3 \mid x^2 + y^2 + z^2 = 1\big\}. \tag{1.1}$$

Using spherical coordinates, each point $p \in \mathbb{S}^2$ can be written in terms of its latitude θ and longitude φ as $p = (\sin\theta\cos\varphi,\ \sin\theta\sin\varphi,\ \cos\theta)$. Thus the map $X : \mathbb{S}^2 \to [0, \pi] \times [0, 2\pi)$ that takes

$$X(\sin\theta\cos\varphi,\ \sin\theta\sin\varphi,\ \cos\theta) = (\theta, \varphi) \tag{1.2}$$

is a *locally* Euclidean description of $\mathbb{S}^2$. In fact when restricted away from the longitudes $\varphi = 0$, X is a homeomorphism, whereupon it is termed a *chart*. As such, the power of differential geometry lies in conceptualizing manifolds as abstract objects independent from their manifestation in Euclidean space

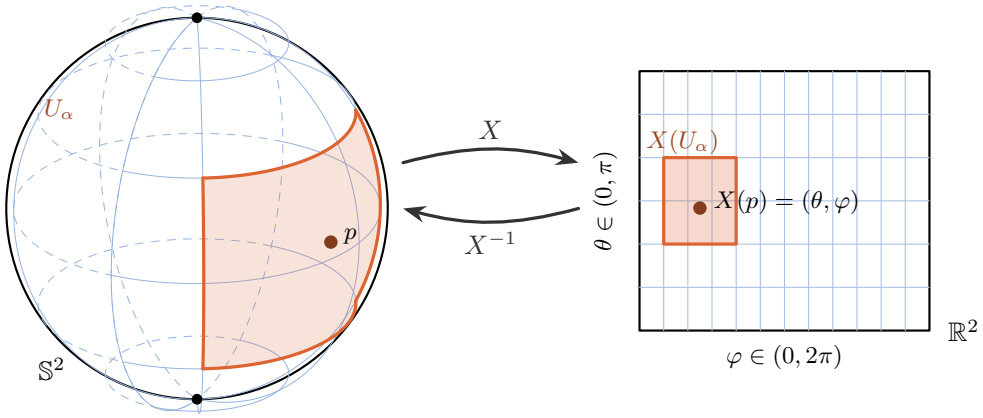


Figure 1.1: Latitudes and longitudes give spherical coordinates, i.e a chart on the sphere.

(akin to how one views $\mathbb{S}^2$ as set in $\mathbb{R}^3$). Charts then provide a (local) bridge between the abstract geometry of a manifold and the intuitive notions of Euclidean space. Using this unit sphere and the chart X as the running example, we now summarize the key notions from differential geometry needed for this dissertation. All of this material is standard and can be found in [GQ20].

1.1.1 Riemannian Metric and Geodesics

Definition 1.1.1. *Fixing a chart (U_α, X_α) on $\mathcal{M}$, the* tangent space *at $x \in \mathcal{M}$ is*

$$T_x\mathcal{M} = \{v \in \mathbb{R}^d \mid v = (X_\alpha^{-1} \circ \gamma)'(0) \text{ for some smooth curve } \gamma : (-\varepsilon, \varepsilon) \to \mathcal{M},\ \gamma(0) = x\}. \quad (1.3)$$

The tangent space provides a linear approximation to $\mathcal{M}$ at the chosen point p.

Example 1.1.2 (Tangent space on $\mathbb{S}^2$)**.** *We work in standard spherical coordinates (θ, φ). At a point*

$p = (\theta_0, \varphi_0)$*, the coordinate curves*

$$\gamma_\theta(t) = (\sin(\theta_0 + t)\cos\varphi_0,\ \sin(\theta_0 + t)\sin\varphi_0,\ \cos(\theta_0 + t)), \tag{1.4}$$

$$\gamma_\varphi(t) = (\sin\theta_0\cos(\varphi_0 + t),\ \sin\theta_0\sin(\varphi_0 + t),\ \cos\theta_0). \tag{1.5}$$

yield the coordinate basis vectors

$$\partial_\theta\big|_p = \dot\gamma_\theta(0) = (\cos\theta_0\cos\varphi_0,\ \cos\theta_0\sin\varphi_0,\ -\sin\theta_0),$$

$$\partial_\varphi\big|_p = \dot\gamma_\varphi(0) = (-\sin\theta_0\sin\varphi_0,\ \sin\theta_0\cos\varphi_0,\ 0).$$

Both vectors are tangent to $\mathbb{S}^2$ *(orthogonal to the position vector* p*) and together span*

$$T_p\mathbb{S}^2 = p + \big\{v \in \mathbb{R}^3 \mid v \cdot p = 0\big\}, \tag{1.6}$$

the plane through p*, perpendicular to* p*.*

The key leap in Riemannian geometry is to generalize the notion of angles in Euclidean space to tangent vectors with a point-dependent function called the *metric*. We require the metric to be positive definite at each point, just like the inner product in Euclidean space:

Definition 1.1.3 (Riemannian metric)**.** *A* Riemannian metric g *is a family of symmetric positive semi-definite* 2*-tensors* $\{g_x\}_{x\in\mathcal{M}}$ *satisfying, for every chart* $X_\alpha : U_\alpha \to \mathbb{R}^d$*,*

$$g(x)[v, w] = g_{ij}(x)\, v^i w^j \quad \forall\, v, w \in T_x\mathcal{M}, \tag{1.7}$$

where the functions $g_{ij} \circ X_\alpha^{-1}$ are smooth. Here and below, the Einstein summation convention is in force, where $g_{ij}(x)v^i w^j := \sum_{i,j} g_{ij}(x)v_i w_j$.

Example 1.1.4 (Round metric on $\mathbb{S}^2$)**.** *The metric on $\mathbb{S}^2$ is induced by the Euclidean inner product on $\mathbb{R}^3$. Computing $g_{ij} = \partial_i \cdot \partial_j$ for the coordinate vectors above gives*

$$g_{\theta\theta} = \partial_\theta \cdot \partial_\theta = 1, \qquad g_{\theta\varphi} = \partial_\theta \cdot \partial_\varphi = 0, \qquad g_{\varphi\varphi} = \partial_\varphi \cdot \partial_\varphi = \sin^2\theta. \tag{1.8}$$

Thus the metric tensor (in matrix form) and its inverse are

$$g = \begin{pmatrix} 1 & 0 \\ 0 & \sin^2\theta \end{pmatrix}, \qquad g^{-1} = \begin{pmatrix} 1 & 0 \\ 0 & \csc^2\theta \end{pmatrix}. \tag{1.9}$$

For two tangent vectors $v = v_\theta \partial_\theta + v_\varphi \partial_\varphi$ and $w = w^\theta \partial_\theta + w^\varphi \partial_\varphi$ at a point p,

$$g_p(v, w) = v_\theta w_\theta + \sin^2\theta_0 \; v_\varphi w_\varphi. \tag{1.10}$$

The metric substitutes for the notion of *angles* and can now be used to define the notion of lengths and distances. Returning to the analogy on Euclidean space, the shortest path between two points is a line–this is because it is a minimizer of a length of a curve γ, given by $\int_0^1 |\dot\gamma(t)|\, dt$. Using the metric, we can now define shortest paths between two points on a manifold as the minimizers of a similar notion of length.

Definition 1.1.5 (Geodesic)**.** *The* energy *of a differentiable curve* $\tilde{\gamma} : [a, b] \to \mathcal{M}$ *is*

$$E[\tilde{\gamma}] = \int_a^b g(\tilde{\gamma}(t))\big(\dot{\tilde{\gamma}}(t),\, \dot{\tilde{\gamma}}(t)\big)\, dt. \tag{1.11}$$

A geodesic *between* $a, b \in \mathcal{M}$ *is an energy-minimizing curve:*

$$\gamma := \underset{\substack{\tilde{\gamma}(0)=a \\ \tilde{\gamma}(1)=b}}{\arg\min}\, E[\tilde{\gamma}]. \tag{1.12}$$

Example 1.1.6 (Geodesics on $\mathbb{S}^2$)**.** *The geodesics of* $\mathbb{S}^2$ *are* great circles*, i.e. intersections of* $\mathbb{S}^2$ *with planes through the origin. To see this, note that the energy of a curve* $\tilde{\gamma} : [0, 1] \to \mathbb{S}^2$ *expressed in spherical coordinates is*

$$E[\tilde{\gamma}] = \int_0^1 \left(\dot{\theta}^2 + \sin^2\theta\, \dot{\varphi}^2\right) dt. \tag{1.13}$$

A calculus of variations argument shows that the above energy is minimized by constant-speed traversal $\theta(t) = \theta_a + (\theta_b - \theta_a)t$*. The geodesic distance between two points* $p, q \in \mathbb{S}^2$ *is then the* central angle

$$d(p, q) = \arccos(p \cdot q), \tag{1.14}$$

which equals the length of the shorter great-circle arc connecting them.

1.1.2 Exponential Map and Normal Coordinates

An interesting aspect of the above example is that linear movements in the latitude θ exactly sweep out the geodesic between the north pole and any other point on $\mathbb{S}^2$, showing that the chart we picked was a rather convenient choice for mapping out geodesics. Somewhat surprisingly, this situation is much more general than just $\mathbb{S}^2$: every Riemannian manifold admits a chart of this type, where unit length movements in the geodesic map to unit length movements in the range of the chart. To construct such a chart we formalize what it would mean for lines to map to geodesics. Specifically, for every $p \in \mathcal{M}$ and tangent vector $v \in T_p\mathcal{M}$, ODE theory guarantees a unique geodesic γ_v with $\gamma_v(0) = p$ and $\dot{\gamma}_v(0) = v$. This yields the *exponential map*.

Definition 1.1.7 (Exponential map)**.** *For $x \in \mathcal{M}$ there exists $r(x) > 0$ such that*

$$\exp_x : B_{r(x)}(0) \subset T_x\mathcal{M} \longrightarrow \mathcal{M}, \qquad \exp_x(v) := \gamma_v(1),$$

is well-defined. The largest such $r(x)$ for which $\exp_x$ is a diffeomorphism is the injectivity radius at x*, and*

$$\operatorname{inj}(\mathcal{M}) = \inf_{x\in\mathcal{M}} \sup\big\{r \mid \exp_x : B_r(0) \to \mathcal{M} \text{ is a diffeomorphism}\big\}. \tag{1.15}$$

Example 1.1.8 (Exponential map on $\mathbb{S}^2$)**.** *Fix the north pole $p = (0, 0, 1)$, so $T_p\mathbb{S}^2 = \{(v_1, v_2, 0) \mid v_1, v_2 \in \mathbb{R}\} = \{(r\cos\varphi, r\sin\varphi, 0), r \geq 0, \varphi \in \mathbb{R}\}$. For $v = (v_1, v_2, 0) \in T_p\mathbb{S}^2$ with $\|v\| = r > 0$, the unique unit-speed geodesic through p in direction $\hat{v} = v/r$ is a great circle–thus the direction will be*

φ and the geodesic swept out will be parameterized by rt:

$$\gamma_v(t) = \cos(rt)\, p + \sin(rt)\, \hat{v} \tag{1.16}$$

$$= \left(\tfrac{v_1}{r}\sin(rt),\ \tfrac{v_2}{r}\sin(rt),\ \cos(rt)\right) \tag{1.17}$$

$$= \big(\cos(\varphi)\sin(rt), \sin(\varphi)\sin(rt), \cos(rt)\big) \tag{1.18}$$

Evaluating at $t = 1$ and relabeling r as θ gives the exponential map explicitly:

$$\exp_p(v) := \big(\cos(\varphi)\sin(\theta), \sin(\varphi)\sin(\theta), \cos(\theta)\big) = X^{-1}(\theta, \varphi). \tag{1.19}$$

This is a diffeomorphism precisely when $\|v\| < \pi$; at $\|v\| = \pi$ all directions map to the south pole $-p$. Hence $\mathrm{inj}(\mathbb{S}^2) = \pi$.

For compact $\mathcal{M}$, $0 < \mathrm{inj}(\mathcal{M}) < \infty$. Setting $U = B_{\mathrm{inj}(\mathcal{M})}(0)$ and $V_x = \exp_x(U)$, the pair $(V_x, \exp_x^{-1})$ defines a chart called *normal coordinates* at x. The above example shows that the latitude-longitude pair is exactly the normal coordinate chart for the sphere.

Integration on manifolds. Normal coordinates are especially useful because they provide a general chart for any Riemannian manifold $\mathcal{M}$ at hand and can therefore be used to tackle mathematical problems where additional information on $\mathcal{M}$ is unknown. For instance, as we shall see in Chapter 2, we will encounter integration on manifolds. In this case, normal coordinates are a tool to convert

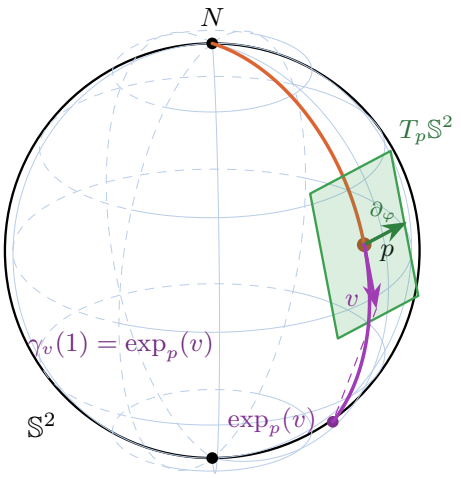


Figure 1.2: The tangent space to $p \in \mathbb{S}^2$ is the best linear approximation to the sphere at p and, in spherical coordinates, is spanned by the vectors ∂_φ and ∂_θ. The inner products between these vectors give the metric. A unit speed line in the tangent space is mapped by the exponential map to a geodesic lying on the sphere. This provides a system of coordinates for mapping any point on the tangent plane to a point on $\mathbb{S}^2$.

this integration to Euclidean space via pullback:

$$\int_{\mathcal{M}} f(y)\, d\text{vol}(y) \approx \int_{\exp(B_r(p))} f(y) d\text{vol}(y) \tag{1.20}$$

$$= \int_{B_r(p)} f(y(x)) \sqrt{\det g(x)} J(x)\, dx. \tag{1.21}$$

Here $B_r(p)$ is a ball of radius r with center p in the tangent space and $J(x)$ is the Jacobian that appears as a result of change of variables. The idea will be to pick the right ball such that the above approximation is valid. In this way, we reduce a difficult problem (integration on $\mathcal{M}$) to an easier problem (integration on $\mathbb{R}^d$) using normal coordinates.

1.1.3 Maps Between Manifolds

The most straightforward task when grappling with high-dimensional data is to map it to low dimensions so that it is easier to visualize, understand, and compute with. For this purpose we need to understand the basics of maps between manifolds, which we summarize below.

Definition 1.1.9 (Differential/pushforward)**.** *Let $F : \mathcal{M} \to \mathcal{N}$ be a smooth map between smooth manifolds. The* differential *(or* pushforward*) of F at $p \in \mathcal{M}$ is the linear map*

$$dF_p : T_p\mathcal{M} \longrightarrow T_{F(p)}\mathcal{N} \tag{1.22}$$

defined by its action on tangent vectors $v \in T_p\mathcal{M}$:

$$(dF_p\, v)(f) = v(f \circ F) \quad \textit{for all } f \in C^\infty(\mathcal{N}). \tag{1.23}$$

Example 1.1.10 (Symmetries of $\mathbb{S}^2$)**.** *The unit sphere is preserved by $O(3)$, making every element of $O(3)$ an automorphism of $\mathbb{S}^2$. These automorphisms are in fact isometries: for tangent vectors $v, w \in T_x\mathbb{S}^2$, the pushforward of O at x is $dO_x = O$ (since O is linear), so*

$$g_{Ox}(dO_x\, v,\, dO_x\, w) = (Ov)^\top (Ow) = v^\top O^\top O w = v^\top w = g_x(v, w).$$

The round metric is thus preserved by every $O \in O(3)$, so $O(3) \subseteq \mathrm{Isom}(\mathbb{S}^2)$.

For the remaining definitions in this section, we use as a running example the *spherical parameteriza-*

tion map

$$X^{-1} : (0,\pi)\times(0,2\pi) \subset \mathbb{R}^2 \longrightarrow \mathbb{S}^2 \subset \mathbb{R}^3, \qquad F(\theta,\varphi) = (\sin\theta\cos\varphi,\ \sin\theta\sin\varphi,\ \cos\theta). \tag{1.24}$$

Example 1.1.11 (Differential of X^{-1})**.** *The domain $\mathcal{M} = (0,\pi)\times(0,2\pi)$ is an open subset of $\mathbb{R}^2$, so $T_{(\theta,\varphi)}\mathcal{M} = \mathbb{R}^2$ with basis $\{e_\theta, e_\varphi\}$. Differentiating column-wise:*

$$dX^{-1}_{(\theta,\varphi)} = \begin{pmatrix} \cos\theta\cos\varphi & -\sin\theta\sin\varphi \\ \cos\theta\sin\varphi & \sin\theta\cos\varphi \\ -\sin\theta & 0 \end{pmatrix}. \tag{1.25}$$

For a tangent vector $v = (v_\theta, v_\varphi) \in T_{(\theta,\varphi)}\mathbb{R}^2$, the pushforward is

$$dX^{-1}(\theta,\varphi)(v) = v_\theta\,\partial_\theta X^{-1} + v_\varphi\,\partial_\varphi X^{-1} \ \in\ T_{X^{-1}(\theta,\varphi)}\mathbb{S}^2 \subset \mathbb{R}^3. \tag{1.26}$$

One checks directly that $X^{-1}(\theta,\varphi)\cdot dX^{-1}_{(\theta,\varphi)}(v) = 0$ for all v, confirming that the image of dX^{-1} lies in $T_{F(\theta,\varphi)}\mathbb{S}^2$.

Definition 1.1.12 (Immersion and embedding)**.** *A smooth map $F : \mathcal{M} \to \mathcal{N}$ is an* immersion *at $p \in \mathcal{M}$ if its differential is injective:*

$$\ker(dF_p) = \{0\}. \tag{1.27}$$

If this holds for all $p \in \mathcal{M}$, then F is called an immersion*. If additionally F is a homeomorphism onto its image, then F is an* embedding *and $\mathcal{M}$ is said to be* embedded *in $\mathcal{N}$.*

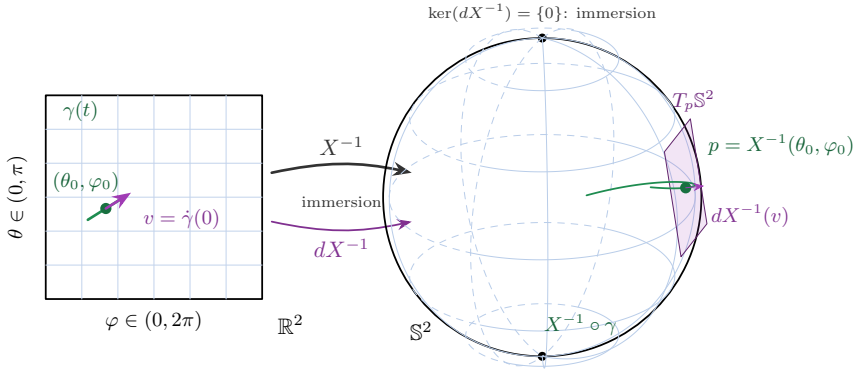


Figure 1.3: Differentials take tangent vectors to tangent vectors, thus serving as the analogue for the Jacobian of a map. When the differential is injective (as it is for the purple curve above) the map is an *immersion*.

Example 1.1.13 (F is an immersion but not a global embedding)**.** *linearly independent at every interior point, so* $\ker(dX^{-1}) = \{0\}$ *and* X^{-1} *is an immersion on* $(0, \pi) \times (0, 2\pi)$*. However,* X^{-1} *is not a global embedding of* $\mathbb{R}^2$ *into* $\mathbb{S}^2$ *because it is not surjective.*

Definition 1.1.14 (Hypersurface)**.** *Let* $\mathcal{N}^n$ *be an* n*-dimensional smooth manifold. A* hypersurface *in* $\mathcal{N}$ *is a smooth manifold* $\mathcal{M}^{n-1}$ *together with an embedding* $F : \mathcal{M}^{n-1} \hookrightarrow \mathcal{N}^n$ *whose image has codimension* 1*:*

$$\dim \mathcal{N} - \dim F(\mathcal{M}) = 1. \tag{1.28}$$

Equivalently, a hypersurface is locally the regular level set of a smooth function: there exist an open set $V \subset \mathcal{N}$ *and a smooth* $\varphi : V \to \mathbb{R}$ *such that*

$$\mathcal{M} \cap V = \{p \in \mathcal{N} : \varphi(p) = 0\}, \quad d\varphi_p \neq 0 \text{ on } \mathcal{M} \cap V. \tag{1.29}$$

Example 1.1.15 ($\mathbb{S}^2$ as a hypersurface in $\mathbb{R}^3$)**.** *Take* $\mathcal{N} = \mathbb{R}^3$ *(*$n = 3$*) and* $\varphi(x, y, z) = x^2 + y^2 + z^2 - 1$.

Then $\mathbb{S}^2 = \varphi^{-1}(0)$*. The differential is*

$$d\varphi_{(x,y,z)} = (2x,\ 2y,\ 2z),$$

which is nonzero for all $(x, y, z) \in \mathbb{S}^2$ *(since* $x^2 + y^2 + z^2 = 1 > 0$*). Hence* 0 *is a regular value of* φ*, so* $\mathbb{S}^2$ *is a smooth hypersurface in* $\mathbb{R}^3$ *of dimension* $3 - 1 = 2$*.*

Definition 1.1.16 (Normal vector)**.** *Let* $F : \mathcal{M}^{n-1} \hookrightarrow \mathcal{N}^n$ *be a smooth embedded hypersurface, and let* g *denote the Riemannian metric on* $\mathcal{N}$*. A* normal vector *at* $p \in \mathcal{M}$ *is a nonzero vector* $\nu_p \in T_{F(p)}\mathcal{N}$ *satisfying*

$$g_{F(p)}(\nu_p,\ dF_p(X)) = 0 \quad \textit{for all } v \in T_p\mathcal{M}. \tag{1.30}$$

Since $\mathcal{M}$ *has codimension* 1*, the space of normal vectors is one-dimensional and any two normal vectors differ by a scalar multiple.*

Example 1.1.17 (Normal vector to $\mathbb{S}^2$)**.** *At a point* $q = F(\theta, \varphi) \in \mathbb{S}^2$*, the normal vector must be orthogonal to both* $\partial_\theta F$ *and* $\partial_\varphi F$ *in the Euclidean metric on* $\mathbb{R}^3$*. Since* $\mathbb{S}^2 = \varphi^{-1}(0)$*, the gradient of* φ *is normal to the level set:*

$$\nabla\varphi\big|_q = 2q = 2(\sin\theta\cos\varphi,\ \sin\theta\sin\varphi,\ \cos\theta).$$

One verifies directly that $\nabla\varphi|_q \cdot \partial_\theta F = 0$ *and* $\nabla\varphi|_q \cdot \partial_\varphi F = 0$ *at every point. The* outward unit normal *is therefore*

$$\nu_q = q = (\sin\theta\cos\varphi,\ \sin\theta\sin\varphi,\ \cos\theta),$$

i.e. the position vector itself, which points radially outward. The inward normal is $-q$.

1.1.4 Intrinsic Differential Operators

Finally, we study *differentiation* of functions on manifolds. In general, this requires the language of *connections* but here we can define the relevant differentiation operations directly in local co-ordinates.

Definition 1.1.18 (Manifold gradient)**.** *Let $(\mathcal{M}, g)$ be a Riemannian manifold and $f \in C^\infty(\mathcal{M})$. The* manifold gradient $\nabla_{\mathcal{M}} f$ *is the unique vector field on $\mathcal{M}$ satisfying*

$$g_x(\nabla_{\mathcal{M}} f(x),\, v) = df_x(v) \quad \textit{for all } v \in T_x\mathcal{M}. \tag{1.31}$$

In local coordinates $(x_1, \ldots, x_d)$, this reads

$$(\nabla_{\mathcal{M}} f)_i = (g^{-1})^{ij}\, \partial_j f, \tag{1.32}$$

Example 1.1.19 (Manifold gradient on $\mathbb{S}^2$)**.** *Take $f(\theta, \varphi) = \cos\theta$. Using the inverse metric $g^{-1} = \operatorname{diag}(1, \csc^2\theta)$:*

$$(\nabla_{\mathbb{S}^2} f)_\theta = (g^{-1})^{\theta\theta}\, \partial_\theta f = 1 \cdot (-\sin\theta) = -\sin\theta, \tag{1.33}$$

$$(\nabla_{\mathbb{S}^2} f)_\varphi = (g^{-1})^{\varphi\varphi}\, \partial_\varphi f = \csc^2\theta \cdot 0 = 0. \tag{1.34}$$

In ambient $\mathbb{R}^3$ coordinates this is

$$\nabla_{\mathbb{S}^2}(\cos\theta)\big|_q = -\sin\theta\ \partial_\theta F = -\sin\theta\,(\cos\theta\cos\varphi,\ \cos\theta\sin\varphi,\ -\sin\theta) \tag{1.35}$$

$$= (-\sin\theta\cos\theta\cos\varphi,\ -\sin\theta\cos\theta\sin\varphi,\ \sin^2\theta). \tag{1.36}$$

Definition 1.1.20 (Laplace–Beltrami operator)**.** *Let $(\mathcal{M}, g)$ be a Riemannian manifold. The* Laplace–Beltrami operator *$\Delta_{\mathcal{M}}$ is the intrinsic generalization of the Euclidean Laplacian, defined for $f \in C^\infty(\mathcal{M})$ by*

$$\Delta_{\mathcal{M}} f := \operatorname{div}_{\mathcal{M}}(\nabla_{\mathcal{M}} f)\,. \tag{1.37}$$

In local coordinates $(x_1, \ldots, x_d)$ this takes the explicit form

$$\Delta_{\mathcal{M}} f = \frac{1}{\sqrt{|g|}}\,\partial_i\Big(\sqrt{|g|}\ g^{ij}\,\partial_j f\Big)\,, \tag{1.38}$$

where $|g| = \det(g_{ij})$. In normal coordinates at x, the metric satisfies $g_{ij}(x) = \delta_{ij}$, so the operator coincides with the Euclidean Laplacian at that point:

$$\Delta_{\mathcal{M}} f(x) = \Delta_{\mathbb{R}^d}(f \circ \exp_x)(0). \tag{1.39}$$

Example 1.1.21 (Laplace–Beltrami operator on $\mathbb{S}^2$)**.** *For $\mathbb{S}^2$ in spherical coordinates, $|g| = \det(g_{ij}) =$*

$\sin^2\theta$, *so* $\sqrt{|g|} = \sin\theta$, *and the formula* (1.38) *gives*

$$\begin{aligned}\Delta_{\mathbb{S}^2} f &= \frac{1}{\sin\theta}\left[\partial_\theta(\sin\theta\,\partial_\theta f) + \partial_\varphi(\csc\theta\,\partial_\varphi f)\right] \\ &= \frac{1}{\sin\theta}\,\partial_\theta(\sin\theta\,\partial_\theta f) + \frac{1}{\sin^2\theta}\,\partial_\varphi^2 f. \end{aligned} \tag{1.40}$$

1.2 Stochastic Differential Equations and Diffusion Processes

1.2.1 Itô SDEs in $\mathbb{R}^d$

We begin with the classical theory of stochastic differential equations in Euclidean space before passing to the intrinsic, coordinate-free setting needed for dynamics on Riemannian manifolds. Let $(\Omega, \mathcal{F}, \{\mathcal{F}_t\}_{t\geq 0}, \mathbb{P})$ be a filtered probability space. We first begin with the definition of a martingale, which will follow along for the rest of this subsection.

Definition 1.2.1. *Let* $(\Omega, \mathcal{F}, \{\mathcal{F}_t\}_{t\geq 0}, \mathbb{P})$ *be a filtered probability space where* $\mathcal{F}_s \subseteq \mathcal{F}_t$ *for ever* $s \leq t$ *and* X_t *an* $\mathcal{F}_t$*-measurable stochastic process.* X_t *is a martingale if for every* $t, s \geq 0$*,*

$$\mathbb{E}[X_{t+s} \mid \mathcal{F}_s] = X_t. \tag{1.41}$$

One of the standard martingales with respect to W_t is a standard d-dimensional Brownian motion adapted to $\{\mathcal{F}_t\}$. An *Itô stochastic differential equation* on $\mathbb{R}^d$ takes the form

$$dX_t = b(X_t)\,dt + \sigma(X_t)\,dW_t, \tag{1.42}$$

where $b : \mathbb{R}^d \to \mathbb{R}^d$ and $\sigma : \mathbb{R}^d \to \mathbb{R}^{d\times d}$ are the drift and diffusion fields respectively. Under standard

Lipschitz and linear-growth conditions on b and σ, there exists a unique strong solution to (1.42) for any $\mathcal{F}_0$-measurable initial condition X_0 [KS07]. The solution $\{X_t\}_{t\geq 0}$ is a (strong) Markov process. The central analytical tool for working with Itô processes is *Itô's formula*: for any $f \in C^2(\mathbb{R}^d)$,

$$df(X_t) = \nabla f(X_t) \cdot dX_t + \frac{1}{2} \sum_{i,j=1}^{d} a_{ij}(X_t)\, \partial_i \partial_j f(X_t)\, dt, \tag{1.43}$$

where $a(x) = \sigma(x)\sigma(x)^\top$ is the *diffusion tensor*. Writing this out in full and separating the finite-variation and martingale parts,

$$f(X_t) = f(X_0) + \int_0^t \mathcal{L} f(X_s)\, ds + M_t^f, \tag{1.44}$$

where M_t^f is a martingale and the *infinitesimal generator* $\mathcal{L}$ acts on $f \in C^2(\mathbb{R}^d)$ by

$$\mathcal{L} f(x) = \sum_{i=1}^{d} b_i(x)\, \partial_i f(x) + \frac{1}{2} \sum_{i,j=1}^{d} a_{ij}(x)\, \partial_i \partial_j f(x). \tag{1.45}$$

The generator completely encodes the infinitesimal statistics of the process: it determines transition probabilities, ergodic properties, and the long-time behavior of observables.

1.2.2 The Martingale Problem

Stroock and Varadhan [SV07] gave an equivalent characterization of diffusion processes that does not appeal to any particular coordinate system or driving noise. Given a linear operator $\mathcal{L}$ acting on a suitable domain $\mathcal{D}(\mathcal{L}) \subset C_b(\mathbb{R}^d)$, we say that a probability measure $\mathbb{P}$ on path space $C([0,\infty);\mathbb{R}^d)$ solves the *martingale problem* for $(\mathcal{L}, \nu)$ if:

1. $\mathbb{P}(X_0 \in \cdot) = \nu$, and

2. for every $f \in \mathcal{D}(\mathcal{L})$, the process

$$M_t^f := f(X_t) - f(X_0) - \int_0^t \mathcal{L}f(X_s)\, ds \tag{1.46}$$

is a $\mathbb{P}$-martingale with respect to the canonical filtration.

Stroock and Varadhan proved that, under ellipticity and continuity conditions on the coefficients of $\mathcal{L}$, the martingale problem is *well-posed*: there exists a unique solution $\mathbb{P}$, and the canonical process under $\mathbb{P}$ is a strong Markov process whose generator is precisely $\mathcal{L}$ [SV07]. In the Euclidean setting, well-posedness of the martingale problem for $\mathcal{L}$ of the form (1.45) is equivalent to the existence and uniqueness (in law) of the SDE (1.42) [KS07].

The martingale problem formulation is particularly valuable because it is *intrinsic*: the characterization (1.46) requires only the operator $\mathcal{L}$ and a topology on the state space, with no reference to a driving Brownian motion, a stochastic integral, or local coordinates.

1.2.3 Diffusions on Riemannian Manifolds

When the state space is a smooth Riemannian manifold $(\mathcal{M}, g)$—as arises naturally when modelling molecular systems with internal degrees of freedom, constrained dynamics, or configuration spaces with nontrivial topology—the Itô formulation (1.42) becomes coordinate-dependent and therefore geometrically ambiguous. The standard Euclidean Itô correction acquires additional *Christoffel-symbol* terms under a change of coordinates, and there is no canonical notion of "adding" a tangent vector to a point on a manifold along a stochastic trajectory [Hsu02].

To overcome this problem, several technical constructions are available, but for the purposes of this dissertation the most natural and direct approach is to bypass the stochastic integral entirely and *define* the diffusion process on $\mathcal{M}$ through the martingale problem. Let $(\mathcal{M}, g)$ be a smooth, connected Riemannian manifold without boundary, and let $\mathcal{L}$ be a second-order elliptic differential operator on $\mathcal{M}$ with smooth coefficients and no zeroth-order term. A continuous $\mathcal{M}$-valued Markov process $\{X_t\}_{t\geq 0}$ is said to be the *diffusion with generator* $\mathcal{L}$ if, for every $f \in C^\infty(\mathcal{M})$, the process

$$M_t^f := f(X_t) - f(X_0) - \int_0^t \mathcal{L}f(X_s)\, ds \tag{1.47}$$

is a local martingale. Equivalently, the law μ_t of X_t is the unique weak solution to the Fokker–Planck equation

$$\partial_t \mu_t = \mathcal{L}^* \mu_t, \tag{1.48}$$

where $\mathcal{L}^*$ is the $L^2(\mathrm{vol}_g)$-formal adjoint of $\mathcal{L}$, computed with respect to the Riemannian volume form $\mathrm{vol}_g = \sqrt{\det g}\, dx_1 \wedge \cdots \wedge dx_d$.

In local coordinates $(x_1, \ldots, x_d)$ on $\mathcal{M}$, the most general second-order elliptic operator without zeroth-order term takes the form

$$\mathcal{L}f = b^i(x)\, \partial_i f + a^{ij}(x)\, \partial_i \partial_j f, \tag{1.49}$$

where $a^{ij}(x)$ is a symmetric positive-definite tensor field (the diffusion tensor) and b^i encodes the drift. The formal adjoint with respect to vol_g is

$$\mathcal{L}^*\mu = -\frac{1}{\sqrt{|g|}}\partial_i\Big(\sqrt{|g|}\, b^i \mu\Big) + \frac{1}{\sqrt{|g|}}\partial_i\partial_j\big(\sqrt{g}\, a^{ij}\mu\big)\,, \tag{1.50}$$

with $|g| = \det(g_{ij})$. The coordinate expressions (1.49)–(1.50) are consistent under changes of chart, since $\mathcal{L}$ is an intrinsically defined operator. Well-posedness of the martingale problem (1.47) on a compact manifold follows from the classical theory of Stroock–Varadhan together with the ellipticity of a_{ij}.

1.2.4 Overdamped Langevin Dynamics

In Chapters 2 & 3, we are concerned with systems of interest that evolve according to the *overdamped Langevin equation*,

$$dX_t = -\nabla V(X_t)\,dt + \sqrt{2\beta^{-1}}\,dW_t, \tag{1.51}$$

where $V : \mathcal{M} \to \mathbb{R}$ is a smooth potential energy function, $\beta = (k_B T)^{-1}$ is the inverse temperature, and W_t is a standard Brownian motion on $\mathcal{M} \subseteq \mathbb{R}^m$. When $\mathcal{M}$ is a Riemannian manifold, equation (1.51) is understood in the intrinsic sense of Section 1.2.3: it denotes the unique Markov process whose generator acts on smooth $f : \mathcal{M} \to \mathbb{R}$ by

$$\mathcal{L}f(x) = -\nabla V(x) \cdot \nabla f(x) + \beta^{-1}\Delta f(x), \tag{1.52}$$

where, whenever appropriate, ∇ and Δ are the Riemannian gradient and Laplace–Beltrami operator on $\mathcal{M}$ respectively. Concretely, $\{X_t\}$ is the unique Markov process for which (1.47) holds with $\mathcal{L}$ given by (1.52), and whose law μ_t satisfies the Fokker-Planck equation $\partial_t \mu_t = \mathcal{L}^* \mu_t$. The dynamics (1.51)

are time-reversible with respect to the *Gibbs* (Boltzmann) measure

$$d\mu(x) \propto \exp(-\beta V(x))\, d\mathrm{vol}(x). \tag{1.53}$$

To verify this, note that with $\mu \propto e^{-\beta V}$ one computes $\mathcal{L}^*\mu = 0$, confirming stationarity.

Many biophysical systems of interest—such as protein folding or conformational changes in macromolecules modeled by (1.51)—are *metastable*: the process spends long periods near local minima of V and transitions rarely between them [VWD22]. This metastability makes direct i.i.d. sampling from μ impractical and motivates the enhanced sampling techniques and data-driven methods described in the following sections.

1.3 Analysis of rare events via Transition Path Theory

The transitions between metastable states A and B in the overdamped Langevin dynamics (1.51) provide a highly flexible model for rare transitions in stochastic systems. Under this model, we may study not only the frequency of these transitions but also the more important question regarding their *mechanism*. Specifically, we are interested in *how* transitions from A to B occur, on average. To study this question, we adopt the framework of transition path theory, where given a trajectory X_t, we consider the ensemble of *reactive trajectories*, i.e pieces of X_t that last exited A and are due to enter B before A (see Figure 1.4). Remarkably, [VEE06] showed that the statistical quantification of such trajectories boils down to the committor function q defined below. Before going into the details, we summarize here the essential point of TPT. Most trajectories are *not* reactive owing to the fact that transitions from A to B are rare. However, despite being rare events, the transitions between A and B happen in the same, consistent way–this is because the other ways of transitioning are *even rarer*.

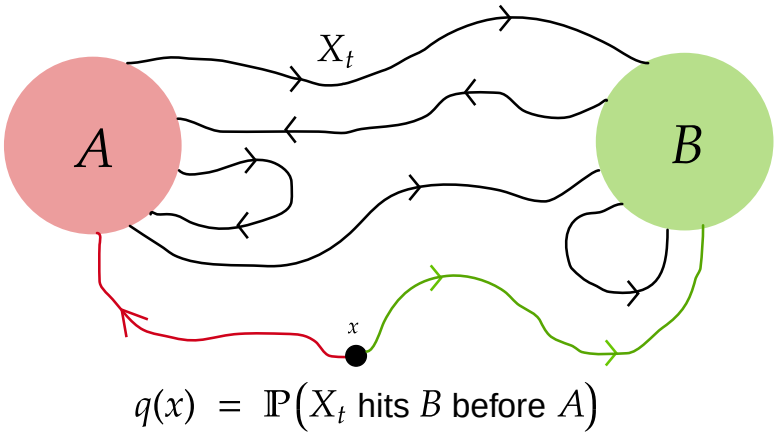


Figure 1.4: Transition path theory studies rare events via the ensemble of trajectories X_t. After determining metastable states A and B as the regions of interest, the preferential hit time of B over A, the committor function, provides a way to study the statistics of transitions from A to B.

1.3.1 The Committor Function

The central object of TPT is the *forward committor* $q(x)$, defined as the probability that a trajectory of (1.51) started at $x \in \mathcal{M}_{AB}$ reaches B before A. By the Feynman–Kac formula, q solves the boundary value problem,

$$\mathcal{L}q(x) = 0, \quad x \in \mathcal{M}_{AB}, \qquad q\big|_{\partial A} = 0, \qquad q\big|_{\partial B} = 1. \tag{1.54}$$

Because (1.51) is time-reversible, the *backward committor*—the probability that a trajectory arriving at x last visited A rather than B—equals $1 - q(x)$. Time-reversibility also yields a variational characterization of q:

$$q = \min_{\substack{u \in C^1(\mathcal{M}_{AB}) \\ u|_{\partial A}=0,\, u|_{\partial B}=1}} \int_{\mathcal{M}_{AB}} \|\nabla u(x)\|_2^2 \, e^{-\beta V(x)} \, d\mathrm{vol}(x). \tag{1.55}$$

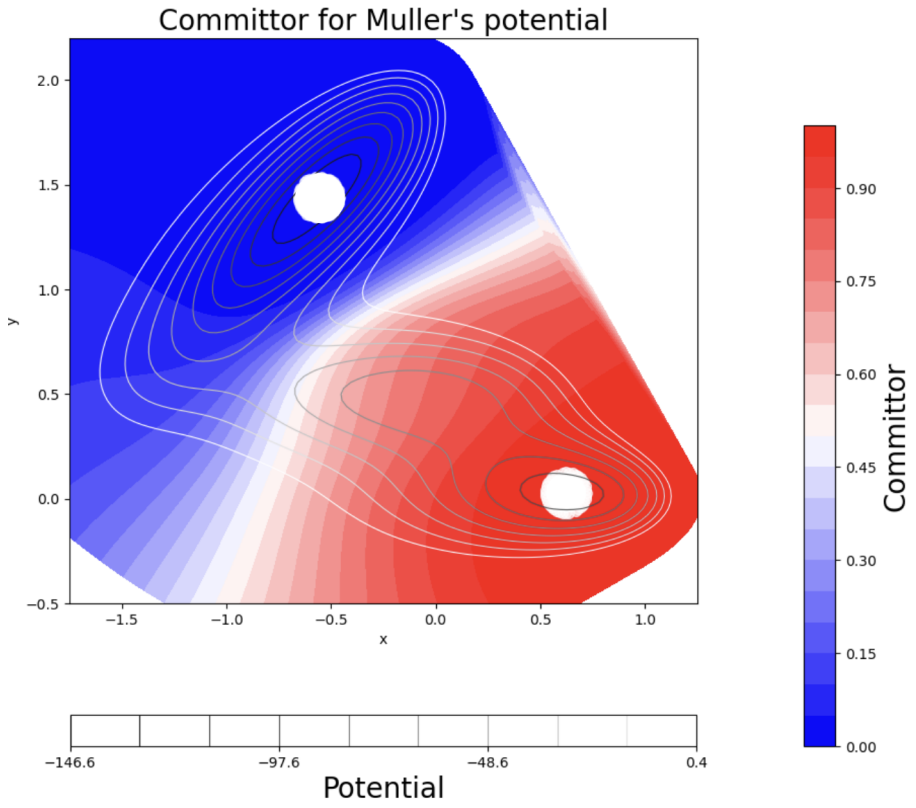


Figure 1.5: Committor function for Müller's potential defined in detail in Chapter 2.

1.3.2 Quantities of Interest

The committor q gives access to the following key statistical quantities.

1. **Reactive current.** The vector field

$$J(x) = \beta^{-1}\mu(x)\nabla q(x) \tag{1.56}$$

is the *reactive current*. The magnitude of its streamlines delineate the dominant transition channels from A to B.

2. **Transition rate.** Define N_{AB} as the number of $A \to B$ transitions observed over a long time

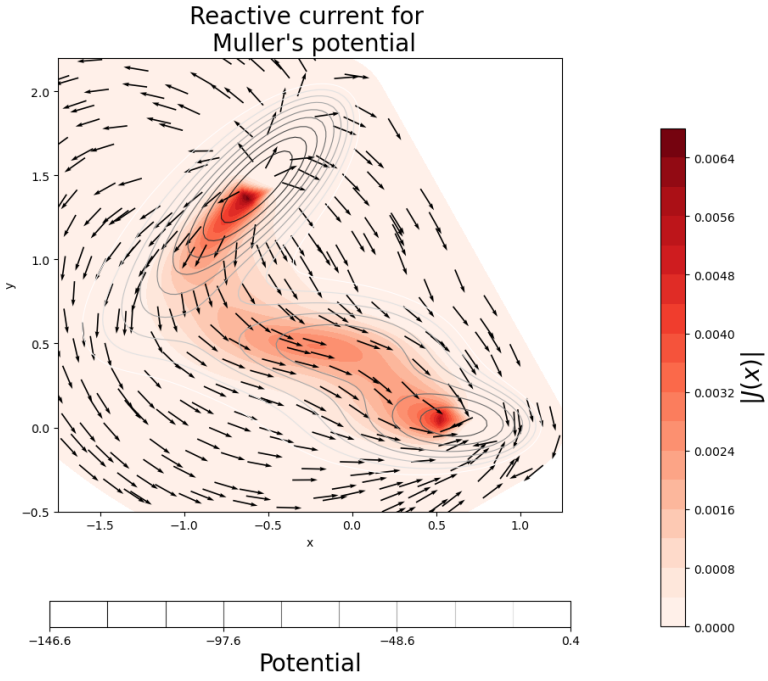


Figure 1.6: Reactive current for Müller's potential. The regions of large magnitude of the current (in red) mark the transition channels between A and B.

interval $[0, T]$. The *transition rate* is

$$\nu_{AB} = \lim_{T\to\infty} \frac{N_{AB}}{T} = \beta^{-1} \int_{\mathcal{M}_{AB}} \|\nabla q(x)\|_2^2 \, \mu(x) \, d\mathrm{vol}(x). \tag{1.57}$$

3. **Probability of last hitting A.** The probability that an infinitely long trajectory last hit A rather than B at a randomly chosen time is

$$\rho_A = \int_{\mathcal{M}\setminus B} (1 - q(x)) \, \mu(x) \, dx. \tag{1.58}$$

4. **Escape rate.** Let T_A denote the total time the trajectory spends having most recently visited A. The *escape rate* from A is

$$k_{AB} = \lim_{T\to\infty} \frac{N_{AB}}{T_A} = \frac{\nu_{AB}}{\rho_A}. \tag{1.59}$$

1.4 Data-driven algorithms

Having described the statistical and geometric notions required to understand the data, we now turn to the algorithms used in this dissertation. As a starting point, we introduce the *dimensionality reduction* problem. In particular, let $\mathcal{X}(n) = \{x_i\}_{i=1}^n \subset \mathcal{M}$ be a set of points on a compact Riemannian manifold $\mathcal{M} \subseteq \mathbb{R}^m$ of unknown dimension. The dimensionality reduction problem amounts to finding a map $\psi : \mathcal{M} \to \mathbb{R}^d$ such that $\{\psi(x_i)\}$ represents $\mathcal{X}(n)$ as closely as possible, with the notion of "close" being up to the user's discretion and dependent on the specific characteristics of the task at hand. We now delve into diffusion maps and neural networks as two algorithms that can be used to compute such maps ψ.

1.4.1 Diffusion maps

The Diffusion map algorithm (Dmap) [CL06, CKL+08] is widely known as a nonlinear dimensionality reduction algorithm for mapping high-dimensional data into low dimensions. However, at its heart, it is a kernel-based algorithm for approximating certain elliptic operators (like $\mathcal{L}$) from finitely many samples supported on a manifold–akin to how finite difference or finite element methods discretize differential operators on Euclidean domains. In this dissertation, we will also investigate in detail an extension of Dmap for diffusion processes, the so-called *Target measure diffusion map* (TMDmap) [BTB+20]. Dmap and TMDmap are essentially the same algorithm but differ in one step. To this end, we detail both algorithms below, starting with the shared part.

1.4.1.1 Shared Construction

Let $\mathcal{X}(n) = \{x_i\}_{i=1}^n \subset \mathcal{M}$ be a dataset sampled from a density ρ. The Gaussian kernel with bandwidth $\epsilon > 0$ is

$$k_\epsilon(x, y) := \exp\left(-\frac{\|x - y\|^2}{\epsilon}\right). \tag{1.60}$$

From this kernel one forms the symmetric *kernel matrix* $[K_\epsilon^{(n)}]_{ij} = k_\epsilon(x_i, x_j)$ and the diagonal *kernel density estimate* (KDE)

$$[D_\epsilon^{(n)}]_{ii} = \rho_\epsilon^{(n)}(x_i) := \frac{1}{n}\sum_{j=1}^{n}[K_\epsilon^{(n)}]_{ij}. \tag{1.61}$$

The continuous counterpart of the KDE, upon taking the law of large numbers (LLN) limit $n \to \infty$, is

$$\rho_\epsilon(x) := \int_{\mathcal{M}} k_\epsilon(x, y)\, \rho(y)\, dy. \tag{1.62}$$

After an algorithm-specific normalization (described below), the kernel matrix is row-normalized to a Markov (stochastic) matrix $P^{(n)}$ and the *discrete generator* is defined by

$$L_\epsilon^{(n)} := \frac{P^{(n)} - I}{\epsilon}. \tag{1.63}$$

1.4.1.2 Standard Diffusion Maps

In Dmap with density-removal parameter $\alpha \in [0, 1]$, the normalized kernel is

$$K_{\epsilon,\alpha}^{(n)} = K_\epsilon^{(n)} \left(D_\epsilon^{(n)}\right)^{-\alpha}. \tag{1.64}$$

Setting $\alpha = 1$ removes the effect of the sampling density ρ entirely. The Markov matrix is

$$D_{\epsilon,\alpha}^{(n)} = \operatorname{diag}\left(K_{\epsilon,\alpha}^{(n)}\mathbf{1}\right), \qquad P_{\epsilon,\alpha}^{(n)} = \left(D_{\epsilon,\alpha}^{(n)}\right)^{-1} K_{\epsilon,\alpha}^{(n)}. \tag{1.65}$$

With $\alpha = 1$, the generator $L_{\epsilon,1}^{(n)}$ converges as $n \to \infty$, $\epsilon \to 0$ to the Laplace–Beltrami operator $\Delta_{\mathcal{M}}$ [CL06]. This makes diffusion maps a meshless discretization of $\Delta_{\mathcal{M}}$, and its eigenvectors provide a low-dimensional description of $\mathcal{M}$. More precisely, let $\tilde{\psi}_i$ be eigenfunctions of $-\Delta_{\mathcal{M}_\mathcal{F}}$ with eigenvalues λ_i. There exists $d \in \mathbb{N}$ such that

$$\tilde{\psi}(x) := \left(\lambda_1\tilde{\psi}_1(x), \ldots, \lambda_d\tilde{\psi}_d(x)\right) \in \mathbb{R}^d \tag{1.66}$$

is an *embedding* [Bat14]. The diffusion map $\tilde{\psi}^{n,\epsilon}$ approximates this Laplacian embedding using the eigenvectors $\tilde{\psi}_j^{n,\epsilon}$ of $L_{\epsilon,1}^{(n)}$:

$$\tilde{\psi}^{n,\epsilon}(p_i) := \left(\lambda_1^{n,\epsilon}\tilde{\psi}_1^{n,\epsilon}(p_i), \ldots, \lambda_d^{n,\epsilon}\tilde{\psi}_d^{n,\epsilon}(p_i)\right). \tag{1.67}$$

1.4.1.3 TMDmap

TMDmap [BTB+20] is designed for the setting in which the data $\mathcal{X}(n)$ are sampled from a *biased* density $\rho \neq \mu$ via enhanced sampling methods such as metadynamics or temperature acceleration [LP02a, BBP11]. The target density $\mu(x) \propto \exp(-\beta V(x))$ is assumed known up to a proportionality constant. The goal is to approximate the action of the generator $\mathcal{L}$ (1.52) from biased data.

TMDmap replaces the Dmap normalization by

$$K_{\epsilon,\mu}^{(n)} = K_{\epsilon}^{(n)} \left(D_{\epsilon}^{(n)}\right)^{-1} M_{\mu}^{(n)}, \qquad [M_{\mu}^{(n)}]_{ii} = \mu^{1/2}(x_i). \tag{1.68}$$

This can be understood as Dmap with $\alpha = 1$ (full density removal) followed by a reweighting by $\mu^{1/2}$, effectively performing *importance sampling* toward the target measure μ. For any fixed $x_i \in \mathcal{X}(n)$, the LLN limit of the normalized kernel matrix is

$$\lim_{n\to\infty} \frac{1}{n} \sum_{j=1}^{n} \left[K_{\epsilon,\mu}^{(n)}\right]_{ij} f(x_j) = \mathcal{K}_{\epsilon,\mu} f(x_i) := \int_{\mathcal{M}} k_{\epsilon,\mu}(x_i, y)\, f(y)\, \rho(y)\, dy, \tag{1.69}$$

where the normalized kernel is

$$k_{\epsilon,\mu}(x, y) := \frac{k_{\epsilon}(x, y)\, \mu^{1/2}(y)}{\rho_{\epsilon}(y)}. \tag{1.70}$$

An equivalent symmetric normalization is

$$K_{\epsilon,\mu}^{(n)} = \left(D_{\epsilon}^{(n)}\right)^{-1} \left[M_{\mu}^{(n)}\right]^{1/2} K_{\epsilon}^{(n)} \left(D_{\epsilon}^{(n)}\right)^{-1} \left[M_{\mu}^{(n)}\right]^{1/2}, \tag{1.71}$$

which produces the same Markov matrix $P_{\epsilon,\mu}^{(n)}$. The Markov matrix and generator are formed by

$$D_{\epsilon,\mu}^{(n)} = \mathrm{diag}\left(K_{\epsilon,\mu}^{(n)} \mathbf{1}_n\right), \quad P_{\epsilon,\mu}^{(n)} = \left(D_{\epsilon,\mu}^{(n)}\right)^{-1} K_{\epsilon,\mu}^{(n)}, \quad L_{\epsilon,\mu}^{(n)} = \frac{P_{\epsilon,\mu}^{(n)} - I}{\epsilon}. \tag{1.72}$$

The LLN limit of the Markov operator $P_{\epsilon,\mu}^{(n)}$ is its continuous counterpart

$$\mathcal{P}_{\epsilon,\mu} f(x) := \frac{\mathcal{K}_{\epsilon,\mu} f(x)}{\mathcal{K}_{\epsilon,\mu} e(x)}, \tag{1.73}$$

where $e(x) = 1$ for all $x \in \mathcal{M}$, and correspondingly the LLN limit of the generator $L_{\epsilon,\mu}^{(n)}$ is

$$\mathcal{L}_{\epsilon,\mu} f(x) := \frac{\mathcal{P}_{\epsilon,\mu} f(x) - f(x)}{\epsilon}. \tag{1.74}$$

The convergence of $L_{\epsilon,\mu}^{(n)}$ to $\mathcal{L}_{\epsilon,\mu}$ is characterized by the following theorem.

Theorem 1.4.1 (Theorem 3.1 of [BTB+20]). *Let $f : \mathcal{M} \to \mathbb{R}$ be smooth. Then as $n \to \infty$ and $\epsilon \to 0$, for every $x_i \in \mathcal{X}(n)$,*

$$\mathcal{L}_{\epsilon,\mu} f(x) \to \frac{1}{4}\big(\Delta f + \nabla \log \mu \cdot \nabla f\big) \equiv \frac{\beta}{4} \mathcal{L} f(x). \tag{1.75}$$

Thus TMDmap recovers the physical generator $\mathcal{L}$ (up to the constant $\beta/4$) even when the data are sampled from a density $\rho \neq \mu$. In Chapter 2, we shall give more advanced asymptotics for this theorem.

1.4.2 Neural Networks

Neural networks (NNs) are parametric function approximators used throughout this dissertation for tasks ranging from parameter selection to operator learning. A fully connected L-layer NN defines a map $f_\Theta : \mathbb{R}^{n_0} \to \mathbb{R}^{n_L}$ via the alternating composition of affine maps and nonlinearities:

$$f_\Theta(x) = W_L\big(\phi(W_{L-1}(\cdots \phi(W_1 x + b_1) \cdots) + b_{L-1})\big) + b_L, \tag{1.76}$$

	Dmap	TMDmap
Input	Dataset $\mathcal{X}(n) \subset \mathcal{M}$, Parameter $\alpha \in [0,1]$, Bandwidth $\epsilon > 0$	Dataset $\mathcal{X}(n) \subset \mathcal{M}$, Target density $\mu : \mathbb{R}^d \to [0,1]$, Bandwidth $\epsilon > 0$
Kernel matrix	$[K_\epsilon]_{ij} = k_\epsilon(x_i, x_j)$	$[K_\epsilon]_{ij} = k_\epsilon(x_i, x_j)$
Density estimation	$D_\epsilon^{(n)} = \mathrm{diag}(K_\epsilon \mathbf{1})$	$D_\epsilon^{(n)} = \mathrm{diag}(K_\epsilon \mathbf{1})$, $M = \mathrm{diag}([\mu(x_i)]_{x_i \in \mathcal{X}})$
Normalization	$K_{\epsilon,\alpha}^{(n)} = K_\epsilon D_\epsilon^{-\alpha}$	$K_{\epsilon,\mu}^{(n)} = K_\epsilon D_\epsilon^{-1} M^{1/2}$
Markov renormalization	$D_{\epsilon,\alpha}^{(n)} = \mathrm{diag}(K_{\epsilon,\alpha}^{(n)} \mathbf{1})$, $P_{\epsilon,\alpha}^{(n)} = (D_{\epsilon,\alpha}^{(n)})^{-1} K_{\epsilon,\alpha}^{(n)}$	$D_{\epsilon,\mu}^{(n)} = \mathrm{diag}(K_{\epsilon,\mu}^{(n)} \mathbf{1})$, $P_{\epsilon,\mu}^{(n)} = (D_{\epsilon,\mu}^{(n)})^{-1} K_{\epsilon,\mu}^{(n)}$
Generator	$L_{\epsilon,\alpha}^{(n)} = \epsilon^{-1}(P_{\epsilon,\alpha}^{(n)} - I)$	$L_{\epsilon,\mu}^{(n)} = \epsilon^{-1}(P_{\epsilon,\mu}^{(n)} - I)$

Table 1.1: Summary of the Dmap and TMDmap algorithms. The kernel k_ϵ is given by (1.60).

where $\phi : \mathbb{R} \to \mathbb{R}$ is a fixed activation function applied element-wise, W_i are weight matrices, b_i are bias vectors, and $\Theta = \{W_i, b_i\}_{i=1}^L$ collects all trainable parameters.

1.4.2.1 Neural networks as function approximators

In Section 1.4.1 we showed that the DMaps $\psi^{n,\epsilon}$ constructed from the data approximate the Laplace-Beltrami eigenfunctions on $\mathcal{M}$. We can therefore recast the dimensionality reduction problem as one of *function approximation*: given the existence of some true underlying map $\Psi : \mathcal{M} \to \mathbb{R}^d$

(such as the Laplace-Beltrami eigenfunctions), we would like to construct some $\widehat{f}$ to approximate this map as closely as possible. Neural networks are especially useful from this perspective as the repeated compositions of nonlinearities and affine functions make them highly efficient function approximators. In fact, several "classical" approximation methods can be recast as neural network approximations. For instance, consider an even function $f \in C^1[-1/2, 1/2]$. Then the *d-degree Fourier series* approximation of f is given by

$$\mathcal{C}_d[f](x) := \sum_{k=-d}^{d} \widehat{f}(k) \cos(2\pi k x). \tag{1.77}$$

Here $\widehat{f}(k)$ are the ith Fourier series coefficients of f. We can recast this as a neural network approximation: let the activation function ϕ be $\phi(\cdot) := \cos(2\pi\cdot)$.. Then $\mathcal{C}_d[f] = W_2(\phi(W_1 x) + b_1) + b_2$ where $W_1 \in \mathbb{R}^{(2d+1)\times 1}$, $[W_1]_k = k$, $[W_2]_k = \widehat{f}(k)$, and $b_1 = b_2 = 0$. Since the Fourier series of any continuously differentiable even function converges uniformly [Jac12], for any even periodic continuously differentiable f and a given accuracy in the sup norm, we can find a neural network (i.e the cosine series of f) that approximates it to that accuracy. This is an example of a *universal approximation theorem*. When it comes to general neural networks, quantitative universal approximation theorems that state the precise depth, width, and number of parameters involved in the network are possible, especially when assuming additional conditions on the function class being approximated. For instance, we have the following theorem:

Theorem 1.4.2 (Corollary 1 in [LSYZ21]). *For any s times continuously differentiable $f \in C^s[0,1]^d$ there exists a fully connected neural network ψ with ReLU activation, depth N and width L such that*

$$\|f - \psi\|_{C^s[0,1]} \leq C_1 \|f\|_{C^s[0,1]} \left(\frac{C_2 \log(8N+8)}{\log N}\right)^{2s/d} \left(\frac{C_3 \log(4L+4)}{L-2d}\right)^{2s/d}, \tag{1.78}$$

with

$$N \gtrsim ds^{d+1}3^{d+2}, L \gtrsim s^2 + 2d, \tag{1.79}$$

and C_1, C_2, C_3 are constants that depend on s, d.

The interpretation of theorem 1.4.2 is as follows: given any target function of suitable regularity, we can always find a neural network that approximates it to any arbitrary degree, as long as it is of sufficient width and sufficient depth. In fact, the smoother the function is, the faster such approximations converge, at the cost of the exponential blow up of the prefactors C_1, C_2, C_3. Results such as Theorem 1.4.2 quantifying the approximation power of neural networks continue to be discovered, leading to a rich approximation theory of neural networks [EPGB21].

1.4.2.2 Training neural networks

In this dissertation, we are not concerned with proving approximation results like Theorem 1.4.2. Instead, we will use neural networks as computational tools for solving empirical problems. However, in this setting, universal approximation theorems (UATs) carry a fatal flaw: the constructions proposed in UATs require knowledge of the target function f. In practice, however, we simply have access to a *training set* of input-output pairs $\{(x_i, y_i)\}_{i=1}^n$ where the y_i's are the putative evaluations of the underlying function, and the network parameters are determined by minimizing a loss function ℓ that measures the differences between predictions and targets:

$$\Theta^\star = \operatorname*{argmin}_{\Theta} \frac{1}{n} \sum_{i=1}^{n} \ell(y_i,\ f_\Theta(x_i))\,. \tag{1.80}$$

Common choices for ℓ include the mean squared error $\ell(y, \hat{y}) = \|y - \hat{y}\|_2^2$ for regression tasks and cross-entropy for classification tasks. In practice, (1.80) is solved with stochastic gradient descent variants such as Adam [KB14]. In Chapter 4 we will use this paradigm of training extensively, and propose a data pre-processing technique that improves trained neural networks in the context of a supervised learning problem.

1.4.2.3 Autoencoders

We now elaborate on how neural networks may be trained to perform dimensionality reduction, using the architecture known as the *autoencoder*.

Definition 1.4.3. *An* autoencoder *is a neural network $f : \mathbb{R}^m \to \mathbb{R}^m$ written as a composition of an encoder $f_{Enc} : \mathbb{R}^m \to \mathbb{R}^d$ and a decoder $f_{Dec} : \mathbb{R}^d \to \mathbb{R}^m$ such that $f = f_{Dec} \circ f_{Enc}$, $d \ll m$, and $f(x) \approx x$ in some sense.*

Essentially, an autoencoder is a neural network that approximates the identity function but is *bottlenecked* in the middle. This bottleneck causes the network to compress or *encode* the essential information of the data into $\mathbb{R}^d$ via the encoder, and then reconstruct or *decode* the data back into $\mathbb{R}^m$ via the decoder. In practice, autoencoders are a standard thoroughfare in deep learning and arise in contexts and forms that are too numerous to note here. We refer the interested reader to [Bal12] for a more detailed survey.

Once an autoencoder has been trained with the relevant loss function, the encoder f_{Enc} constructs a low-dimensional representation of the data. Autoencoders are thus a powerful framework for dimensionality reduction, and in fact, given the right choice of architecture and activation function, can *subsume* other well-known dimensionality reduction algorithms.

Example 1.4.4 (Two layer linear autoencoders perform principal component analysis)**.** *Consider a two-layer autoencoder with the identity activation function and a bottleneck dimension of 1 trained on data* $\{x_i\} \subseteq \mathbb{R}^m$ *with the mean squared error loss. Therefore, we have*

$$f_{Enc} = w^\top x + c,\ f_{Dec} = vx + c,\ w, v, x, c \in \mathbb{R}, \tag{1.81}$$

$$\ell(v, w, b, c) = \frac{1}{n}\sum_{i=1}^{n} \|x_i - vw^\top x_i - bv - c\|_2^2. \tag{1.82}$$

Let μ *be the mean of* x_i *and* $\overline{x}_i = x_i - \mu$*. We can break the loss down above in terms of* μ*:*

$$\ell(v, w, b, c) = \frac{1}{n}\sum_{i=1}^{n} \|(x_i - \mu) - vw^\top(x_i - \mu) - bv - c + \mu + vw^\top\mu\|_2^2 \tag{1.83}$$

$$= \frac{1}{n}\sum_{i=1}^{n} \|(x_i - \mu) - vw^\top(x_i - \mu)\|_2^2 + \frac{1}{n}\sum_{i=1}^{n} \|\mu + vw^\top\mu - bv - c\|_2^2 \tag{1.84}$$

$$+ \underbrace{\frac{1}{n}\sum_{i=1}^{n} \langle \mu + vw^\top\mu - bv - c, (x_i - \mu) - vw^\top(x_i - \mu)\rangle}_{0} \tag{1.85}$$

$$= \frac{1}{n}\sum_{i=1}^{n} \|\overline{x}_i - vw^\top\overline{x}_i\|_2^2 + \frac{1}{n}\sum_{i=1}^{n} \|\mu + vw^\top\mu - bv - c\|_2^2 \tag{1.86}$$

$$\geq \frac{1}{n}\sum_{i=1}^{n} \|\overline{x}_i - vw^\top\overline{x}_i\|_2^2 := \ell^{'}(v, w). \tag{1.87}$$

Here (1.87) *follows from* (1.86) *since a square is non-negative. Now we lower bound* ℓ'*:*

$$\ell'(v,w) = \frac{1}{n}\sum_{i=1}^{n} \|\overline{x}_i - \langle w, \overline{x}_i\rangle v\|_2^2 = \frac{1}{n}\sum_{i=1}^{n} \|\overline{x}_i\|_2^2 - 2\langle w, \overline{x}_i\rangle\langle v, \overline{x}_i\rangle + \langle w, \overline{x}_i\rangle^2\|v\|_2^2 \tag{1.88}$$

$$= \frac{1}{n}\sum_{i=1}^{n} \|\overline{x}_i\|_2^2 - 2\langle w, \overline{x}_i\rangle\langle v, \overline{x}_i\rangle + \langle w, \overline{x}_i\rangle^2\|v\|_2^2 + \langle \frac{v}{\|v\|_2}, x_i\rangle^2 - \langle \frac{v}{\|v\|_2}, x_i\rangle^2 \tag{1.89}$$

$$= \frac{1}{n}\sum_{i=1}^{n} \|\overline{x}_i\|_2^2 + \left(\langle w, \overline{x}_i\rangle\|v\|_2 - \langle \frac{v}{\|v\|_2}, \overline{x}_i\rangle\right)^2 - \langle \frac{v}{\|v\|_2}, \overline{x}_i\rangle \tag{1.90}$$

$$\geq \left[\frac{1}{n}\sum_{i=1}^{n} \|x_i\|_2^2\right] - \left(\frac{v}{\|v\|_2}\right)^\top \left[\frac{1}{n}\sum_{i=1}^{n} \overline{x}_i\overline{x}_i^\top\right] \left(\frac{v}{\|v\|_2}\right) \tag{1.91}$$

$$\geq \left[\frac{1}{n}\sum_{i=1}^{n} \|x_i\|_2^2\right] - \lambda_1. \tag{1.92}$$

Here λ_1 *is the largest eigenvalue of the positive definite covariance matrix* $n^{-1}\sum_{i=1}^{n} \overline{x}_i\overline{x}_i^\top$*. Once again,* (1.91) *follows from* (1.90) *since squares are non-negative, and* (1.92) *follows from* (1.91) *since the Rayleigh quotient can be bounded above by* λ_1*, and the minus sign converts such an upper bound into a lower bound. We thus have*

$$\ell(v,w,b,c) \geq \left[\frac{1}{n}\sum_{i=1}^{n} \|x_i\|_2^2\right] - \lambda_1. \tag{1.93}$$

The inequality from the (1.91) *to* (1.92) *is tight if and only if we select* $v = v_1$*, the leading eigenvector of the covariance matrix. Moreover, the inequality from* (1.90) *to* (1.91) *is tight if and only if* $w = v = v_1$*. Finally, the only other inequality is* (1.86) *to* (1.87)*–this is tight if and only if* $c = \mu$*,* $b = w^\top\mu$ *and* $w = v$*, which is compatible with the choice needed to make* (1.91) *tight. Thus,* $v = w = v_1$*,* $b = v_1^\top\mu$ *and* $c = \mu$ *are global minimizers of the loss function and the minimizing encoder and decoder are given by* $f_{Enc} = v_1^\top x + v_1^\top\mu$*,* $f_{Dec} = v_1 x + \mu$*. The encoder projects the recentered input onto the direction of largest variance–this is exactly PCA.*

We note why Example 1.4.4 is particularly illuminating: primarily, it shows that autoencoders can serve as a suitable generalization of well-known linear dimensionality reduction techniques. It additionally demonstrates that while neural network loss functions can admit global minimizers, these minimizers may not be *unique*–for instance, the choice of $v = -v_1$ also suffices as a global minimizer. Finally, the network considered in Example 1.4.4 is an example of a *deep linear network (DLN)*. DLNs are idealizations of neural networks, and despite not containing nonlinearities, it has been shown [BDFS22, SMG19] that they nonetheless lend insights into the training dynamics of deep neural networks. We will consider these training dynamics in Chapter 5.

Chapter 2: Learning diffusion operators on manifolds

2.1 Introduction

The diffusion map algorithm (Table 1.1) [CL06] is a popular tool for the analysis of high-dimensional data arising from stochastic dynamical systems. For example, applications of Dmap have guided important discoveries in chemical physics [NLCK06], structural biology [KDKD15], and ocean mixing [BK17]. Dmap is specifically amenable to such applications because it has a means for modulating the sampling distribution of the data. In particular, it is assumed that the experimental data $\mathcal{X} \subseteq \mathbb{R}^m$ is modeled by a stochastic process X_t following the overdamped Langevin dynamics on a compact d-dimensional Riemannian manifold described by (1.51) with a smooth potential $V : \mathcal{M} \to \mathbb{R}$. We recall from Chapter 1 that the dynamics X_t are generated by the backward Kolmogorov operator

$$\mathcal{L} := \beta^{-1}\Delta - \langle \nabla V, \nabla \rangle. \tag{1.52}$$

Here Δ, ∇ and $\langle \cdot, \rangle$ are the Laplace-Beltrami operator, manifold gradient, and tangent space inner product introduced in Chapter 1. The generator $\mathcal{L}$ holds immense information about the SDE (1.51). For instance, the eigenvectors of $\mathcal{L}$ form reaction coordinates for the system (1.51) and statistics of (1.51) such as mean first passage times, reaction rates, or exit times can be studied through solutions to PDEs involving $\mathcal{L}$. Additionally, the spectral information of $\mathcal{L}$ reflects the implicit timescales or the transitions between metastable states in the cases when V has finitely many attracting basins [BDH16].

The advantage of Dmap over its predecessor Laplacian eigenmap [BN03] is that Dmap has a

means for controlling the effect of sampling density. With a properly chosen normalization corresponding to the choice of $\alpha = 1/2$ in the renormalization step (Table 1.1), Dmap produces the matrix $L^{(n)}_{\epsilon,1/2}$, discretizing $\mathcal{L}$ on a finite set of points sampled through the invariant density of the Langevin dynamics (1.51). Remarkably, a priori knowledge of the manifold $\mathcal{M}$ is not required. In particular, as the number of points n tends to infinity and the kernel bandwidth ϵ tends to zero, $L^{(n)}_{\epsilon,1/2} \to \mathcal{L}$ in a pointwise sense when applied to a suitable test function. Thus $L^{(n)}_{\epsilon,1/2}$ is a consistent estimator of $\mathcal{L}$; in fact it has also been established that the spectral information of $L^{(n)}_{\epsilon,1/2}$ is a consistent estimator of the spectral information of $\mathcal{L}$ [BN06, CW22c, WR21].

These approximation properties of Dmap and its variants tailored to more general Ito diffusions have motivated numerous applications to unsupervised learning problems especially in natural sciences. However, the construction of $L^{(n)}_{\epsilon,1/2}$ has an important limitation. The discrete operator $L^{(n)}_{\epsilon,1/2}$ approximates $\mathcal{L}$ correctly *only if the input data is sampled i.i.d.*[1] *from the invariant Gibbs density* $\mu \propto \exp(-\beta V)$. This is quite a restrictive requirement. For instance, Refs. [LL18, BTB+20] point out that sampling from μ can be undesirable or unfeasible because μ concentrates large mass near the minima of V, the so-called metastable states. The same issue stymies the standard Langevin Monte Carlo scheme of simulating the SDE (1.51), as the process X_t resides for long times in metastable states and mixes slowly to μ. Due to this lack of sampling or slow mixing, statistical estimators like $L^{(n)}_{\epsilon,1/2}$ can accumulate large variance errors in transition regions away from metastable states. To fix this problem, Banisch et al. [BTB+20] introduced the *Target Measure Diffusion Map* (TMDmap), the subject of this chapter.

[1] The abbreviation i.i.d. stands for *independent, identically distributed*.

2.1.1 Rare events, the committor problem, and the TMDmap

In TMDmap, we seek to approximate the operator $\mathcal{L}$ generating the dynamics (1.51) with a known *target* density μ proportional to $\exp(-\beta V(x))$. Remarkably, unlike DMap, the input data to TMDmap can come from *any* sampling density ρ. As long as μ is absolutely continuous with respect to ρ, i.e the support of μ is contained within the support of ρ, the data kernel can be appropriately normalized to produce a random walk on the data. The generator $L^{(n)}_{\epsilon,\mu}$ of this random walk converges as $n \to \infty$ to an $O(\epsilon)$ perturbation of the action of $\mathcal{L}$ on a suitable test function f (see Theorem 1.4.1):

$$L^{(n)}_{\epsilon,\mu} f(x) \to \mathcal{L} f(x) + O(\epsilon). \tag{2.1}$$

Hence, the estimator $L^{(n)}_{\epsilon,\mu}$ has the same approximation property as $L^{(n)}_{\epsilon,1/2}$ but can be computed from an arbitrary sampling density supported on $\mathcal{M}$. This augments diffusion maps with importance sampling techniques, particularly with enhanced sampling and data post-processing strategies from molecular dynamics. These enhanced sampling techniques allow the exploration of $\mathcal{M}$ beyond the minima of V where numerical simulations of (1.51) tend to cluster due to metastability.

Given finitely many i.i.d samples from a density ρ supported on $\mathcal{M}$, $L^{(n)}_{\epsilon,\mu}$ is a meshless discretization of $\mathcal{L}$ on $\mathcal{X}$ akin to how a central difference stencil discretizes a Laplacian on a planar grid. This in turn allows us to numerically solve boundary-value problems (BVPs) posed on $\mathcal{M}$ involving the generator $\mathcal{L}$, therefore enabling TMDmap as a tool for solving BVPs on manifolds. One such BVP is the *committor problem*, the pivotal problem in *transition path theory* (TPT) [VEE06], introduced in Chapter 1. We recall that the committor function q is defined as follows: given two disjoint regions A and B in $\mathcal{M}$, $q(x)$ is the probability that the process (1.51) started at x visits B before A. Equivalently,

satisfies the following Dirichlet BVP known as the *committor problem* [VEE06]:

$$\mathcal{L}q(x) = 0,\ x \in \mathcal{M} \setminus (A \cup B), \quad q(\partial A) = 0, \quad q(\partial B) = 1. \tag{1.54}$$

If we have data $\mathcal{X}(n) = \{x_i\}_{i=1}^n$ sampled through (1.51) (for instance, through an Euler-Maruyama scheme) then we can use it as input to the Dmap algorithm and obtain a discretization to problem (1.54):

$$L_{\epsilon,1/2}^{(n)} q^{\mathsf{Dmap}}(x_i) = 0,\ x_i \in \mathcal{M} \setminus (A \cup B), \quad q^{\mathsf{Dmap}}(x_i \in \partial A) = 0, \quad q^{\mathsf{Dmap}}(x_i \in \partial B) = 1. \tag{2.2}$$

Sufficiently sampled data $\mathcal{X}(n)$ from (1.51) will resemble samples of the invariant density μ. But in metastable systems, μ undersamples the transition regions between A and B, leading to poor accuracy of q^{Dmap} in precisely the regions where the committor probabilities are needed. To resolve this undersampling problem, molecular dynamics (MD) practitioners have proposed enhanced sampling algorithms (such as metadynamics [LP02a], summarized in Section 2.4.2.4) which increase sampling in transition states. However, these strategies result in data *not sampled* through the invariant measure μ, thus rendering DMap ineffective. Consequently, it is advantageous to treat the data from an MD simulation as sampled through some unknown arbitrary density ρ and discretize (1.54) through TMDmap (instead of DMap) to get a numerical solution $q_{n,\epsilon}$:

$$L_{\epsilon,\mu}^{(n)} q_{n,\epsilon}(x_i) = 0,\ x_i \in \mathcal{M} \setminus (A \cup B), \quad q_{n,\epsilon}(x_i \in \partial A) = 0, \quad q_{n,\epsilon}(x_i \in \partial B) = 1. \tag{2.3}$$

Specifically, this scheme is implemented as follows. Let $\mathcal{I} = \{i \mid x_i \in \mathcal{M}_{AB}\}$ and $\mathcal{D} = \{i \mid x_i \in A \cup B\}$. Reindexing the set $\mathcal{X}(n)$ so that $\mathcal{I} = \{1, \dots j\}$ and $\mathcal{D} := \{j+1, \dots, n\}$ gives a convenient

block form of the numerical committor

$$q_{n,\epsilon} = \begin{bmatrix} \\ q^{\mathcal{I}}_{n,\epsilon} \\ \\ \\ q^{\mathcal{D}}_{n,\epsilon} \\ \\ \end{bmatrix} \quad \text{where} \quad [q^{\mathcal{D}}_{n,\epsilon}]_i = \begin{cases} 0, & x_i \in A \\ \\ 1, & x_i \in B \end{cases}, \tag{2.4}$$

and the generator matrix L

$$L := \begin{bmatrix} \\ L^{\mathcal{I}\mathcal{I}} & L^{\mathcal{I}\mathcal{D}} \\ \\ \\ L^{\mathcal{D}\mathcal{I}} & L^{\mathcal{D}\mathcal{D}} \\ \\ \end{bmatrix}. \tag{2.5}$$

The discrete version of the committor problem (1.54) is

$$\begin{bmatrix} L^{\mathcal{I}\mathcal{I}} & L^{\mathcal{I}\mathcal{D}} \end{bmatrix} \begin{bmatrix} \\ q^{\mathcal{I}}_{n,\epsilon} \\ \\ \\ q^{\mathcal{D}}_{n,\epsilon} \\ \\ \end{bmatrix} = \mathbf{0}, \tag{2.6}$$

or, equivalently,

$$L^{\mathcal{I}\mathcal{I}} q^{\mathcal{I}}_{n,\epsilon} = -L^{\mathcal{I}\mathcal{D}} q^{\mathcal{D}}_{n,\epsilon}. \tag{2.7}$$

A more general Dirichlet BVP on $\mathcal{M}$,

$$\mathcal{L}u(x) = f, \quad x \in \mathcal{M} \setminus \Omega,$$
$$u = g, \quad x \in \partial\Omega, \tag{2.8}$$

can be solved by TMDmap using a similar linear system. The TMDmap numerical solution to (2.8) is the vector $u_{n,\epsilon}$ that satisfies

$$4\beta^{-1}L^{\mathcal{I}\mathcal{I}}u^{\mathcal{I}}_{n,\epsilon} = f^{\mathcal{I}} - 4\beta^{-1}L^{\mathcal{I}\mathcal{D}}g, \quad u^{\mathcal{D}}_{n,\epsilon} = g. \tag{2.9}$$

Note that $L^{\mathcal{I}\mathcal{I}}$ is the submatrix of L obtained by choosing the rows and columns with indices from $\mathcal{I}$. $L^{\mathcal{I}\mathcal{I}}$ is the Dirichlet Laplacian in spectral graph theory and is known to be invertible (due to, for instance, the Gershgorin disk theorem). Thus the above linear system is well-posed.

2.1.2 The goal of this chapter and summary of main results

The committor problem was solved numerically using a variant of TMDmap with Mahalanobis kernel in [ECT22]. It was observed that subsampling data to make it spatially quasi-uniform using δ-*nets* [CM17] improved the accuracy of the TMDmap-based committor $q^{(n)}_{n,\epsilon}$ [ECT22]. Here, δ-nets are defined as maximal subsets of the point cloud $\mathcal{X}(n)$ where any two points are distance δ apart. The utility of such uniform subsampling raises the question of *what an optimal sampling density should be*. Motivated by this question, the goal of this chapter is to quantify the error of the TMDmap algorithm in terms of its parameters. The specific objectives are the following.

- The first objective is to derive an error formula for the TMDmap generator as a function of the

sampling density ρ, the bandwidth parameter ϵ, the number of sample points n, and the target measure μ. To emphasize its dependence on these parameters, we relabel $L^{(n)}_{\epsilon,\mu}$ as $L^{(n)}_{\epsilon,\mu}$ for the remainder of this chapter. The error has two components, the *bias error* that decays as the bandwidth ϵ tends to zero, and the *variance error* that decays as the number n of samples tends to infinity but blows up as $\epsilon \to 0$ unless $n \to \infty$ fast enough.

- The second objective is to propose a way to reduce the error of TMDmap via simple post-processing of the input data and test it on benchmark problems.

Our results are summarized as follows:

1. We derive a sharp error bound and establish a relationship between n and ϵ necessary for the convergence the TMDmap generator.

   Theorem 2.1.1 (The total error bound)**.** *Let $\mathcal{M}$ be a compact d-dimensional manifold without boundary. Let $\mathcal{X}(n) \subset \mathcal{M}$ be a point cloud sampled i.i.d. with density ρ, $0 < \rho_{\min} \leq \rho(x) \leq \rho_{\max} < \infty$. Let $x \in \mathcal{M}$ be an arbitrary point. Let $f \in C^2(\mathcal{M})$ be an arbitrary function. Furthermore, let ϵ be the kernel bandwidth and μ be the target density used for constructing the TMDmap generator $L^{(n)}_{\epsilon,\mu}$. Then as $n \to \infty$ and $\epsilon \to 0$ so that*[2]

$$\lim_{\substack{n\to\infty \\ \epsilon\to 0}} \frac{n\epsilon^{2+d/2}}{\log n} = \infty, \tag{2.10}$$

[2]The notation $\log$ with no subscript is used for the natural logarithm throughout this chapter.

with probability greater than $1 - 2n^{-3}$*, we have:*

$$|4\beta^{-1}L^{(n)}_{\epsilon,\mu}f(x) - \mathcal{L}f(x)| \leq \underbrace{\frac{\alpha\epsilon}{\rho^{1/2}(x_i)}\left(2\|\nabla f(x)\|\epsilon^{1/2} + 11|f(x)|\right)}_{\textit{variance error}} \tag{2.11}$$

$$+ \underbrace{\epsilon\,|\mathcal{B}_1[f,\mu] + \mathcal{B}_2[f,\mu,\rho] + \mathcal{B}_3[f,\mu,\rho]| + O(\epsilon^2)}_{\textit{bias error}}. \tag{2.12}$$

The expressions for $\alpha, \mathcal{B}_1, \mathcal{B}_2$*, and* $\mathcal{B}_3$ *are given by:*

$$\alpha = \frac{1}{(2\pi)^{d/4}}\sqrt{\frac{\log n}{n\epsilon^{4+d/2}}}, \tag{2.13}$$

$$\mathcal{B}_1[f,\mu] := \frac{1}{4}\left[\mathcal{Q}\left(f\mu^{1/2}\right) - f\mathcal{Q}(\mu^{1/2})\right] + \frac{1}{16}\left(2\nabla f\cdot\nabla\left(\mu^{1/2}\omega\right) + (\mu^{1/2}\omega)\Delta f\right), \tag{2.14}$$

$$\mathcal{B}_2[f,\mu,\rho] := -\frac{1}{16}\left(2\nabla f\cdot\nabla\left(\mu^{1/2}\frac{\Delta\rho}{\rho}\right) + \left(\mu^{1/2}\frac{\Delta\rho}{\rho}\right)f\right), \tag{2.15}$$

$$\mathcal{B}_3[f,\mu,\rho] := \frac{1}{16}\left[\frac{\Delta(\mu^{1/2})}{\mu^{1/2}} - \left(\frac{\Delta\rho}{\rho} - \omega\right)\right]\left[f\frac{\Delta(\mu^{1/2})}{\mu^{1/2}} - \frac{\Delta(\mu^{1/2}f)}{\mu^{1/2}}\right]. \tag{2.16}$$

Here, $\mathcal{Q}$ *is a non-linear differential operator and* ω *is a smooth function on* $\mathcal{M}$*.*

The precise forms of ω and $\mathcal{Q}$ are further detailed in Section 2.2. The proof of this theorem is done using techniques similar to those in [HAL07, Sin06, BN06].

Our key contribution is the explicit formulas for the prefactors in the leading-order terms of the bias and variance errors. To our knowledge, this is the first consistency result that enumerates the prefactor from the bias error.

Remark 2.1.2. *The assumption that the manifold* $\mathcal{M}$ *is compact without boundary is not essential. It is adopted to keep the statement of Theorem 2.1.1 more concise. Often, this assumption does not hold in applications. A common case is when the manifold* $\mathcal{M}$ *is not compact, but*

the data points are located within a bounded region $\Omega \subset \mathcal{M}$. In this case, the error bound in Theorem 2.1.1 remains valid for all points $x \in \Omega$ lying far enough from the boundary of Ω, i.e., at a distance greater than $m\sqrt{\epsilon/2}$ from $\partial\Omega$ where m is a positive number such that $\exp\left(-\frac{m^2}{2}\right) < \epsilon^2$.

2. In Theorem 2.2.15 we use the *method of comparisons* [MM05a] to prove an error estimate for solutions to Dirichlet BVPs using TMDmap (e.g. (2.3)), showing that making the *consistency error* on the left-hand side of (2.11)–(2.12) smaller reduces the *solution error* between the numerical solution and the true solution. Thus improvements to the bias and variance error transfer over as improvements in the solution error.

3. The decay of the bias error may be sped up by canceling some of the prefactors. For instance, such cancellations occur when (i) the manifold has zero curvature, (ii) the sampling density is quasi-uniform, or (iii) when the test function satisfies $\mathcal{L}f = 0$. Additionally, when the manifold has constant but non-zero curvature, a prefactor may still be eliminated if $\mathcal{L}f = 0$. Importantly, conditions (i), (ii), and (iii) are satisfied in a large class of problems of interest. Condition (i) occurs naturally in slow-fast dynamics where the process concentrates around a subspace (for more details see [LZ19b, Eq. 13]). Condition (i) also holds in angular systems such as butane or alanine dipeptide where $\mathcal{M}$ is given by Cartesian products of flat tori $\mathbb{R} \setminus \mathbb{Z}$. Condition (ii) can be obtained when $\mathcal{M}$ is sampled through well-tempered metadynamics [BBP11]. Such samples can be made additionally spatially uniform using δ-nets. Most importantly, condition (iii) is precisely the committor equation (1.54). A consequence of these observations is that the error bound for the committor, $|q^{\mathsf{TMD}} - q|$, is smaller than the general error bound implied by Theorem 2.2.15 (see Corollary 2.2.16) implying that *TMDmap is particularly suited to solving*

the committor problem especially using quasi-uniform sampling densities.

4. We show that these speedups in bias error are realizable in practice for a variety of systems including a 1D periodic system embedded in 2D, Mueller's potential in 2D, and a double well potential 2D. Combined with similar results obtained in [ECT22, Fig. 9] for the alanine-dipeptide system, this chapter adds to the growing body of evidence that improvements in the approximation theory of diffusion maps are of practical importance.

2.1.3 Related work

2.1.3.1 Diffusion maps: History of theoretical developments

The consistency of kernel-based estimators L of backward Kolmogorov operators on manifolds is an active topic in unsupervised learning. The approximation theory of L splits into independent analyses of the bias and variance error. Early work on these errors concerned pointwise consistency, i.e. the pointwise convergence of the estimator L applied to a test function f. The seminal works on pointwise consistency include [BN03, VLBB08, Laf04, HAL05, HAL07]. Building on this literature, [Sin06] improved the bias error rate in [HAL05] from $O(\sqrt{\epsilon})$ to $O(\epsilon)$ and the variance error rate for Laplacian eigenmaps. An important development to this subject was introduced in [BN06] where spectral convergence–the convergence of the discrete eigenvectors to the continuous counterparts–was proved for eigenvectors of the normalized graph Laplacians. Further work on spectral convergence includes [Sin06, CW22a, CW22c, Wan15]. A recent exciting development in the subject has been the integration of optimal transport-based techniques for variance error estimation [GTGHS20, WR21].

2.1.3.2 Applications of diffusion maps

Coifman and Lafon in [CL06] initiated an expansive program of adapting diffusion maps to data modeled by vector bundles [SW12], group invariant manifolds [HK23], and diffusion processes [NLCK06] for applications such as coarse-graining [CKL$^+$08, SEKC09], image segmentation [WDD$^+$08], data fusion [KTLW19], data representation [LMR17, ACM12, BCMS06, Clo14], PDEs on manifolds [LJHY21, ABH21, JH23], and rare event quantification [WKR15, ECT23]. One line of research has concerned algorithmic improvements to the Dmap algorithm such as the use of self-tuning, variable bandwidth, or k-nearest neighbors (kNN) sparsified kernels [CW22b, BH16, BS16a]. Another line of work has centered on improving the diffusion map embedding through its stability [LF19, KCM21] or its statistical properties [LLCC20]. In particular, the setting of Dmap was adapted to isotropic diffusions from invariant densities to anisotropic diffusions from general densities. Key works on this subject include the extension to diffusions obtained from collective variables through Mahalanobis diffusion maps (mmap) [SC08], the subsequent generalization to symmetric positive definite diffusion tensors [ECT23], and Ref. [BTB$^+$20] proposing the TMDmap. In [ECT22], the TMDmap was upgraded with the Mahalanobis kernel for solving the committor problem for time-reversible processes with variable and anisotropic diffusion tensors arising in chemical physics applications.

In the context of these works, the present chapter can be viewed as an extension of the approximation theory of graph Laplacians to TMDmap. In addition, it leverages this approximation theory to gain concrete insights for developing a more robust practical application to rare event quantification.

2.2 Results

In this section, we present our theorems quantifying the convergence of $L^{(n)}_{\epsilon,\mu} f$ to $\mathcal{L} f$ as $\epsilon \to 0$ and $n \to \infty$. Their proofs utilize techniques from the theory of the convergence of Graph Laplacians to the differential operators on a manifold $\mathcal{M}$ [BN06, Sin06, HAL07]. For the sake of easier navigation through our theorems, we place all long proofs in Section 2.3 devoted to proofs.

2.2.1 Problem setup

To set up the main problem in this chapter, we make the following assumptions.

Assumption 2.2.1. *The point cloud $\mathcal{X}(n)$ is sampled i.i.d. from the density ρ supported on $\mathcal{M}$.*

Assumption 2.2.2. *Let $x \in \mathcal{M}$ be an arbitrary point. It can but does not have to belong to the point cloud $\mathcal{X}(n)$.*

Assumption 2.2.3. *We fix $f \in C^\infty(\mathcal{M})$ to be an arbitrary function. Abusing notation, we will denote both the smooth function f and the vector $[f(x_i)]_{i=1}^n$ by the same symbol f.*

Assumption 2.2.4. *Let $V : \mathcal{M} \to \mathbb{R}$ be a smooth potential function such that SDE 1.51 admits the invariant Gibbs density $\mu \propto \exp(-\beta V)$. The measure μ is chosen as the target measure. The sampling density ρ is such that μ is absolutely continuous with respect to ρ.*

Let $L^{(n)}_{\epsilon,\mu}$, $\mathcal{L}_{\epsilon,\mu}$, and $\mathcal{L}$ be the generators defined by (1.63), (1.74), and (1.52) respectively. The law of large numbers implies the convergence $L^{(n)}_{\epsilon,\mu} f(x) \longrightarrow \mathcal{L}_{\epsilon,\mu} f(x)$ as $n \to \infty$, while Theorem 1.4.1 [BTB+20] guarantees the convergence $\mathcal{L}_{\epsilon,\mu} f(x) \longrightarrow \frac{\beta}{4} \mathcal{L} f(x)$ as $\epsilon \to 0$.

The bias and the variance errors are defined by:

$$\text{Bias error:} \qquad |4\beta^{-1}\mathcal{L}_{\epsilon,\mu}f(x) - \mathcal{L}f(x)| \tag{2.17}$$

$$\text{Variance error:} \qquad |L^{(n)}_{\epsilon,\mu}f(x) - \mathcal{L}_{\epsilon,\mu}f(x)| \tag{2.18}$$

2.2.2 Bias Error

The bias error formula including the prefactors for the $O(\epsilon)$ term is (1.75) is established in the following theorem.

Theorem 2.2.5 (Bias Error)**.** *Under Assumptions 2.2.1–2.2.4, the bias error is given by*

$$\begin{aligned}
4\beta^{-1}\mathcal{L}_{\epsilon,\mu} - \mathcal{L} = {} & \frac{\epsilon}{4}\left[\mathcal{Q}\left(f\mu^{1/2}\right) - f\mathcal{Q}(\mu^{1/2})\right] + \frac{\epsilon}{16}\left(2\nabla f\cdot\nabla\left(\mu^{1/2}\omega\right) + (\mu^{1/2}\omega)\Delta f\right) \\
& - \frac{\epsilon}{16}\left(2\nabla f\cdot\nabla\left(\mu^{1/2}\frac{\Delta\rho}{\rho}\right) + \left(\mu^{1/2}\frac{\Delta\rho}{\rho}\right) f\right) \\
& + \frac{\epsilon}{16}\left[\frac{\Delta(\mu^{1/2})}{\mu^{1/2}} - \left(\frac{\Delta\rho}{\rho} - \omega\right)\right]\left[f\frac{\Delta(\mu^{1/2})}{\mu^{1/2}} - \frac{\Delta(\mu^{1/2}f)}{\mu^{1/2}}\right] + O(\epsilon^2) \\
=: {} & \epsilon\left(\mathcal{B}_1[f,\mu](x) + \mathcal{B}_2[f,\mu,\rho](x) + \mathcal{B}_3[f,\mu,\rho](x)\right) + O(\epsilon^2).
\end{aligned} \tag{2.19}$$

A detailed proof of Theorem 2.2.5 is found in Section 2.3.1. Its pivotal component is the second-order kernel expansion formula presented in Lemma 2.2.6 below whose proof is also postponed until Section 2.3.1. Lemma 2.2.6 is a second-order upgrade of the first-order kernel expansion formula from [CL06, HAL05] used in several previous works [BH16, Sin06, CW22a]. The first-order kernel expansion formula from [CL06, HAL05] is insufficient for our goal of establishing how the bias error depends on ρ, μ, and f along with ϵ because it leads to hiding these dependencies into an $O(\epsilon)$ term in the expansion of the generator $\mathcal{L}_{(\cdot)}$.

Lemma 2.2.6 (Second-Order Kernel Expansion Formula)**.** *Let $f \in C^\infty(\mathcal{M})$ and $\mathcal{G}_\epsilon$ be the integral operator defined by*

$$\mathcal{G}_\epsilon f(x) = \int_{\mathcal{M}} k_\epsilon(x,y) f(y)\, d\mathrm{vol}(y). \tag{2.20}$$

Then, for small enough ϵ, $\mathcal{G}_\epsilon f$ admits the following expansion at x:

$$(\pi\epsilon)^{-d/2}\mathcal{G}_\epsilon f(x) = f(x) + \frac{\epsilon}{4}(\Delta f(x) - \omega(x) f(x)) + \frac{\epsilon^2}{4}\mathcal{Q} f(x) + O(\epsilon^3), \tag{2.21}$$

where $\mathcal{Q}$ is a fourth-order differential operator on $\mathcal{M}$ and ω arises from the curvature of $\mathcal{M}$. In particular, if $\mathcal{M}$ is isometric to $\mathbb{R}^d$ in a neighborhood of x (i.e $\mathcal{M}$ is locally flat) then

$$\mathcal{Q} = \partial_i^i \partial_j^j + 2\partial_{ii}^{ii}. \tag{2.22}$$

2.2.3 Reducing bias error

Formula (2.19), or, equivalently, (2.14), (2.15), and (2.16), allow us to identify the situations in which some of the prefactors of the bias error are zero. This, in turn, *possibly* speeds up the pointwise convergence of the TMD map estimator $L^{(n)}_{\epsilon,\mu} f(x)$. We say *possibly* because the prefactors $\mathcal{B}_1, \mathcal{B}_2$, and $\mathcal{B}_3$ may have opposing signs and produce favorable cancellations. Nonetheless, several bias error terms zero out in the following important cases.

- If the sampling density ρ is uniform, the term $\mathcal{B}_2[f,\mu,\rho]$ in (2.19) is zero.
- If f is such that $\mathcal{L}f = 0$, i.e., f is the committor, then the term $\mathcal{B}_3[f,\mu,\rho]$ in (2.19) is zero. This follows from the observation that the second factor in $\mathcal{B}_3[f,\mu,\rho]$ is $\mathcal{L}f$.

- The $O(\epsilon)$ term in the kernel expansion formula (2.21) contains the term ω which arises due to the curvature of $\mathcal{M}$ and correcting the ambient Euclidean (i.e extrinsic) distance in terms of the geodesic (i.e intrinsic) distance. If the manifold $\mathcal{M}$ is flat, e.g. $\mathcal{M}$ is a flat d-dimensional torus, then there is no such distortion between extrinsic and intrinsic distances, due to which $\omega = 0$– see [ECT23, Lemma C.2]. Then the second summand of $\mathcal{B}_1[f, \mu]$ in (2.19) is zero as $\omega = 0$.

Corollary 2.2.7. *If $\mathcal{L}f = 0$, $\mathcal{M}$ is flat, i.e., $\omega = 0$, and the sampling density ρ is uniform, i.e., $\rho(x) = const$, then the bias error reduces to*

$$4\beta^{-1}\mathcal{L}_{\epsilon,\mu}f - \mathcal{L}f = \frac{\epsilon}{4}\left[\mathcal{Q}\left(f\mu^{1/2}\right) - f\mathcal{Q}(\mu^{1/2})\right] + O(\epsilon^2), \tag{2.23}$$

where $\mathcal{Q}$ is the fourth-order differential operator given by (2.22).

2.2.4 Variance Error

The variance error depends on the point cloud, a set of n independent identically distributed (i.i.d.) points sampled from the density ρ. Therefore, it is natural to estimate it by obtaining a concentration inequality of the form

$$\mathbb{P}\left(|L^{(n)}_{\epsilon,\mu}f(x) - \mathcal{L}_{\epsilon,\mu}f(x)| > \alpha\right) \leq \mathcal{E}(n, \alpha, \epsilon, f, \mu). \tag{2.24}$$

An important general-purpose concentration inequality is Bernstein's inequality, which states that if X_i, $i = 1, \ldots, n$, are independent random variables with zero mean and the absolute values

bounded by $|X_i| \leq M$, $1 \leq i \leq n$, then

$$\mathbb{P}\left(\sum_{i=1}^{n} X_i \geq t\right) \leq \exp\left[-\frac{t^2}{2\sum_{i=1}^{n}\mathbb{E}[X_i^2] + \frac{2}{3}Mt}\right] \quad \forall t > 0. \tag{2.25}$$

Inequality (2.25) allows one to derive bounds for sums of random variables that hold with high probability. In the following subsections of Section 2.2.4, we will construct such a bound for $|L_{\epsilon,\mu}^{(n)} f(x) - \mathcal{L}_{\epsilon,\mu} f(x)|$ in a sequence of steps starting from bounding the discrepancy between the discrete, $\rho_\epsilon^{(n)}$, and continuous, ρ_ϵ, kernel density estimates (KDEs).

It is worth noting that in a commonly used strategy for derivation of the variance errors [Sin06, BS16a, BH16], the point x_i at which the error estimate is computed is removed from the point cloud. Here we show that this is not necessary. Our argument is somewhat more involved but it results in a sharp error estimate.

2.2.4.1 Bounds for the error in the kernel density estimate

Let $x \in \mathcal{M}$ be any point on the manifold. Our first goal is to bound the error in the KDE $\rho_\epsilon(x)$. Lemma 2.2.6 implies that

$$\rho_\epsilon(x) := \int_{\mathcal{M}} k_\epsilon(x,y)\rho(y)d\mathrm{vol}(y) = (\pi\epsilon)^{d/2}\left(\rho(x) + O(\epsilon)\right). \tag{2.26}$$

Its discrete estimate $\rho_\epsilon^{(n)}(x)$ on the point cloud $\mathcal{X}(n)$ is

$$\rho_\epsilon^{(n)}(x) := \frac{1}{n}\sum_{j=1}^{n} k_\epsilon(x, x_j). \tag{2.27}$$

The discrepancy between $\rho_\epsilon(x)$ and $\rho_\epsilon^{(n)}(x)$ is quantified by the following theorem.

Theorem 2.2.8 (Discrete kernel density estimate)**.** *Let $\mathcal{X}(n) = \{x_j\}_{j=1}^n$ be a point cloud sampled from the density $\rho(x)$. Let $x \in \mathcal{M}$ be either any fixed point on the manifold $\mathcal{M}$ or any point selected from the point cloud $\mathcal{X}(n)$. Then for $\epsilon \to 0$ and $n \to \infty$ so that*

$$\lim_{\substack{n\to\infty \\ \epsilon\to 0}} \frac{n\epsilon^{2+d/2}}{\log n} = \infty, \tag{2.28}$$

the discrepancy between $\rho_\epsilon(x)$ and $\rho_\epsilon^{(n)}(x)$ is bounded, with probability at least $1 - n^{-4}$, by

$$\left|\rho_\epsilon^{(n)}(x) - \rho_\epsilon(x)\right| < 5(\pi\epsilon)^{d/2}\rho^{1/2}(x)\epsilon^2\alpha \tag{2.29}$$

where

$$\alpha = \frac{1}{(2\pi)^{d/4}}\sqrt{\frac{\log n}{n\epsilon^{4+d/2}}}. \tag{2.30}$$

The proof of Theorem 2.2.8 is detailed in Section 2.3.2.1.

2.2.4.2 Bound for the discrepancy $L_{\epsilon,\mu}^{(n)} f$ and $L_{\epsilon,\mu} f$

Our next goal is to estimate the discrepancy between the discrete generators, $L_{\epsilon,\mu}^{(n)} f$ and $L_{\epsilon,\mu} f$ obtained, respectively, using the discrete, $\rho_\epsilon^{(n)}$, and continuous, ρ_ϵ, KDEs. We have already constructed $L_{\epsilon,\mu}^{(n)}$ in (1.63); we now describe the construction of $L_{\epsilon,\mu}$. To this end, let $\mathcal{X}(n) = \{x_j\}_{j=1}^n$ be a point cloud sampled from the density ρ. Let $x \in \mathcal{M}$ be either an arbitrary fixed point on the manifold $\mathcal{M}$ or any point of $\mathcal{X}(n)$. To facilitate our calculations, we introduce random variables $F_j^{(n)}$, F_j, $G_j^{(n)}$, and

G_j defined as

$$F_j^{(n)} = \frac{k_\epsilon(x, x_j)\mu^{1/2}(x_j)f(x_j)}{\rho_\epsilon^{(n)}(x_j)}, \qquad F_j = \frac{k_\epsilon(x, x_j)\mu^{1/2}(x_j)f(x_j)}{\rho_\epsilon(x_j)}, \tag{2.31}$$

$$G_j^{(n)} = \frac{k_\epsilon(x, x_j)\mu^{1/2}(x_j)}{\rho_\epsilon^{(n)}(x_j)}, \qquad G_j = \frac{k_\epsilon(x, x_j)\mu^{1/2}(x_j)}{\rho_\epsilon(x_j)}. \tag{2.32}$$

The discrete Markov operators $P_{\epsilon,\mu}^{(n)}$ and $P_{\epsilon,\mu}$ acted on the function f and evaluated at x are, respectively,

$$\left[P_{\epsilon,\mu}^{(n)} f\right](x) = \frac{\sum_{j=1}^n F_j^{(n)}}{\sum_{j=1}^n G_j^{(n)}} \quad \text{and} \quad \left[P_{\epsilon,\mu} f\right](x) = \frac{\sum_{j=1}^n F_j}{\sum_{j=1}^n G_j}. \tag{2.33}$$

The discrete generators $L_{\epsilon,\mu}^{(n)} f$ and $L_{\epsilon,\mu} f$ applied to $f(x)$ are, respectively,

$$\left[L_{\epsilon,\mu}^{(n)} f\right](x) = \frac{1}{\epsilon}\left(\left[P_{\epsilon,\mu}^{(n)} f\right](x) - f(x)\right) \quad \text{and} \quad \left[L_{\epsilon,\mu} f\right](x) = \frac{1}{\epsilon}\left(\left[P_{\epsilon,\mu} f\right](x) - f(x)\right). \tag{2.34}$$

$L_{\epsilon,\mu}^{(n)}$ refers to the discrete generator with the discrete KDE while $L_{\epsilon,\mu}$ refers to the discrete generator with the continuous KDE – see formulas (2.31) and (2.32).

Theorem 2.2.9 (Discrepancy between the discrete TMDmap generators with discrete and continuous KDEs)**.** *Under the settings of Theorem 2.2.8, the discrepancy between* $\left[L_{\epsilon,\mu}^{(n)} f\right](x)$ *and* $\left[L_{\epsilon,\mu} f\right](x)$ *defined by* (2.34) *is bounded, with probability at least* $1 - n^{-3}$*, by*

$$\left|\left[L_{\epsilon,\mu}^{(n)} f\right](x) - \left[L_{\epsilon,\mu} f\right](x)\right| < C\epsilon\alpha(1 + o(1)) \tag{2.35}$$

where the constants α *and* C *are given, respectively, by*

$$\alpha = \frac{1}{(2\pi)^{d/4}}\sqrt{\frac{\log n}{n\epsilon^{4+d/2}}} \tag{2.36}$$

and

$$C = \frac{10|f(x)|}{\rho^{1/2}(x)}. \tag{2.37}$$

The proof of Theorem 2.2.9 is found in Section 2.3.2.2.

2.2.4.3 Bound for the discrepancy between $L_{\epsilon,\mu}$ and $\mathcal{L}_{\epsilon,\mu}$

Let $x_i \in \mathcal{X}(n)$ be any selected point in the point cloud. Our next goal is to estimate the discrepancy between $[L_{\epsilon,\mu}f](x_i)$ and $[\mathcal{L}_{\epsilon,\mu}f](x_i)$ defined, respectively, by

$$[L_{\epsilon,\mu}f](x_i) = \frac{1}{\epsilon}\left(\frac{\sum_{j=1}^{n}\frac{k_\epsilon(x_i,x_j)\mu^{1/2}(x_j)f(x_j)}{\rho_\epsilon(x_j)}}{\sum_{j=1}^{n}\frac{k_\epsilon(x_i,x_j)\mu^{1/2}(x_j)}{\rho_\epsilon(x_j)}} - f(x_i)\right) \equiv \frac{1}{\epsilon}\left(\frac{\sum_{j=1}^{n}F_j}{\sum_{j=1}^{n}G_j} - f(x_i)\right) \tag{2.38}$$

and

$$[\mathcal{L}_{\epsilon,\mu}f](x_i) = \frac{1}{\epsilon}\left(\frac{\int_{\mathcal{M}}\frac{k_\epsilon(x_i,y)\mu^{1/2}(y)f(y)}{\rho_\epsilon(y)}dy}{\int_{\mathcal{M}}\frac{k_\epsilon(x_i,y)\mu^{1/2}(y)}{\rho_\epsilon(y)}dy} - f(x_i)\right) \equiv \frac{1}{\epsilon}\left(\frac{\mathbb{E}[F]}{\mathbb{E}[G]} - f(x_i)\right). \tag{2.39}$$

Equations (2.38) and (2.39) show that

$$[L_{\epsilon,\mu}f](x) - [\mathcal{L}_{\epsilon,\mu}f](x) = \frac{1}{\epsilon}\left(\frac{\sum_{j=1}^{n}F_j}{\sum_{j=1}^{n}G_j} - \frac{m_F}{m_G}\right), \tag{2.40}$$

where, for brevity, $\mathbb{E}[F]$ and $\mathbb{E}[G]$ are denoted by m_F and m_G respectively.

Theorem 2.2.10 (Discrepancy between the discrete and continuous TMDmap generators with continuous KDE)**.** *Let $\{x_j\}_{j=1}^{n} \equiv \mathcal{X}(n)$ be a point cloud of n points sampled independently from the density ρ. Let $x_i \in \mathcal{X}(n)$ be any selected point. Let $\epsilon \to 0$ and $n \to \infty$ so that*

$$\lim_{\substack{n\to\infty \\ \epsilon\to 0}} n\epsilon^{1+d/2}\log n = \infty. \tag{2.41}$$

Let the function f be smooth and bounded. Then the discrepancy between $[L_{\epsilon,\mu}f](x_i)$ and $[\mathcal{L}_{\epsilon,\mu}f](x_i)$ defined by (2.38) and (2.39) respectively is bounded, with probability at least $1-n^{-3}$, by

$$|[L_{\epsilon,\mu}f](x_i) - [\mathcal{L}_{\epsilon,\mu}f](x_i)| < C\alpha\epsilon^{3/2} \tag{2.42}$$

where the constants α and C are given, respectively, by

$$\alpha = \frac{1}{(2\pi)^{d/4}}\sqrt{\frac{\log n}{n\epsilon^{4+d/2}}} \tag{2.43}$$

and

$$C = \frac{2\|\nabla f(x_i)\|}{\rho^{1/2}(x_i)}. \tag{2.44}$$

The proof of Theorem 2.2.10 is detailed in Section 2.3.2.3.

2.2.4.4 Bound for the variance error $L^{(n)}_{\epsilon,\mu}f - \mathcal{L}_{\epsilon,\mu}f$

Theorems 2.2.9 and 2.2.10 allow us to estimate the variance error for TMDmap. The result is presented in Theorem 2.2.11.

Theorem 2.2.11. *Let $\{x_j\}_{j=1}^n \equiv \mathcal{X}(n)$ be a point cloud of n points sampled independently from the density ρ. Let $x_i \in \mathcal{X}(n)$ be any selected point. Let $\epsilon \to 0$ and $n \to \infty$ so that*

$$\lim_{\substack{n\to\infty \\ \epsilon\to 0}} \frac{n\epsilon^{2+d/2}}{\log n} = \infty. \tag{2.45}$$

Let the density ρ be bounded from above and away from zero, $0 < \rho_{\min} \le \rho \le \rho_{\max}$, and the function f be smooth and bounded. Then the discrepancy $\left[L^{(n)}_{\epsilon,\mu}f\right](x_i) - [\mathcal{L}_{\epsilon,\mu}f](x_i)$ between the discrete

TMDmap generator applied to f and its continuous counterpart is bounded, with probability at least $1 - 2n^{-3}$, by

$$\left|\left[L_{\epsilon,\mu}^{(n)} f\right](x_i) - \left[\mathcal{L}_{\epsilon,\mu} f\right](x_i)\right| < \frac{\alpha\epsilon}{\rho^{1/2}(x_i)} \left(2\|\nabla f(x_i)\|\epsilon^{1/2} + 11|f(x_i)|\right) \tag{2.46}$$

for large enough n and small enough ϵ, where the constant α is given by

$$\alpha = \frac{1}{(2\pi)^{d/4}} \sqrt{\frac{\log n}{n\epsilon^{4+d/2}}}. \tag{2.47}$$

Proof. The result readily follows from Theorems 2.2.9 and 2.2.10. The definitions of α in these Theorems match. Condition (2.28) on n and ϵ from Theorem 2.2.9 implies condition (2.41) on n and ϵ from Theorem 2.2.10. Condition (2.10) on n and ϵ coincides with (2.28). Therefore, the conditions of both Theorems 2.2.9 and 2.2.10 hold. Furthermore, for small enough ϵ and large enough n, the term $10(1 + o(1))$ in Theorem (2.2.9) is less than 11. Inequalities (2.35) and (2.42) fail to hold with probabilities at most n^{-3}. Hence, with probability at least $1 - 2n^{-3}$, we have

$$\begin{aligned}
&\left|\left[L_{\epsilon,\mu}^{(n)} f\right](x_i) - \left[\mathcal{L}_{\epsilon,\mu} f\right](x_i)\right| \\
\leq &\left|\left[L_{\epsilon,\mu}^{(n)} f\right](x_i) - \left[L_{\epsilon,\mu} f\right](x_i)\right| + \left|\left[L_{\epsilon,\mu} f\right](x_i) - \left[\mathcal{L}_{\epsilon,\mu} f\right](x_i)\right| \\
< &\frac{\alpha\epsilon}{\rho^{1/2}(x_i)} \left(\|\nabla f(x_i)\|\epsilon^{1/2} + 11|f(x_i)|\right)
\end{aligned} \tag{2.48}$$

as desired. □

Theorem 2.2.11 allows us to make several observations.

- The bound of the pointwise variance error (2.46) is inversely proportional to $\rho^{1/2}$. Therefore, to

minimize this prefactor, one needs to choose the uniform sampling density ρ.

- If the function f is the committor, it is smooth and assumes values between 0 and 1. However, if the temperature β^{-1} is small, the committor may have a large gradient in the transition region, in a neighborhood of the saddle point of V separating A and B. Therefore, at a finite bandwidth parameter ϵ, either term in the parentheses in the right-hand side of (2.46) can be larger.

- The KDE established in Theorem 2.2.8 is valid for any sampling density ρ. In contrast, the error bound for the variance error in the generator $\mathcal{L}$ applied to a smooth function f is derived under the assumption that the sampling density ρ is bounded away from zero: $0 < \rho_{\min} \leq \rho \leq \rho_{\max}$. While the upper bound is always finite in real-life applications, the lower bound $\rho_{\min}$ is often zero. Enhanced sampling algorithms such as metadynamics or temperature acceleration generate point clouds supported within a compact region in many applications of interest. This issue can be readily resolved as follows. We use all sampled points to obtain the KDE $\rho_\epsilon^{(n)}$. Furthermore, we construct the discrete generator $L_{\epsilon,\mu}^{(n)}$ using all points. Then, we select a compact region $\Omega \subset \mathcal{M}$ in which the sampling density is above a desired threshold $\rho_{\min}$ and use the computed solution only in this region. The error bound obtained in Theorem 2.2.11 applies in this region.

- In order for α to be $O(1)$, the number of points in the point cloud, n, must scale so that

$$\frac{(2\pi)^{d/2} n}{\log n} = \epsilon^{-4-d/2}. \tag{2.49}$$

This means that choosing larger ϵ can significantly reduce the required size of the point cloud to guarantee the same bound for the variance error. For example, if one chooses $n = 10^4$ points, a value of ϵ predicted by (2.49) is about 0.17 – see Fig. 2.1. In applications, ϵ can be an order of

magnitude smaller [ECT23].

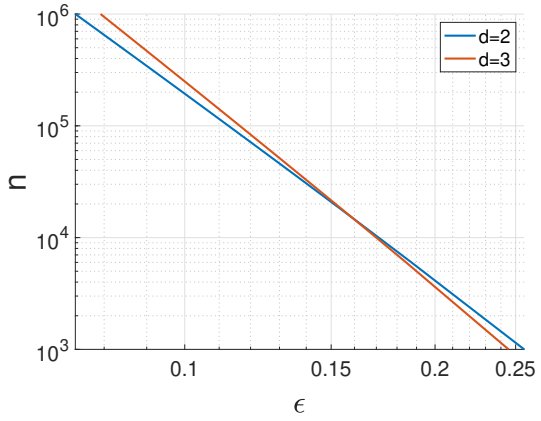


Figure 2.1: The relationship between the number of sample points n on the d-dimensional manifold $\mathcal{M}$ and the bandwidth parameter ϵ at which the parameter α in Theorems 2.2.9, 2.2.10, and 2.2.11 is 1.

2.2.5 Typical error in the kernel density estimate

The error bound obtained in Theorem 2.2.11 is based on Bernstein's general inequality. It is significantly larger than the typical error observed in our numerical experiments. In this section, we find the typical variance error in estimates by computing means and standard deviations of appropriate random variables. Unfortunately, the scaling for n and ϵ for the typical error derived in Theorem 2.2.12 is similar to the one in (2.49). Therefore, the reason for a good performance of TMDmap with uniform ρ achieved via δ-net is different. Some insights on this issue will be given in Section 2.2.6.

Theorem 2.2.12. *Let $\mathcal{X}(n) = \{x_j\}_{j=1}^n$ be a point cloud sampled from the density ρ. Let $x_i \in \mathcal{X}(n)$ be an arbitrarily selected point. The bias and the variance of the error in the kernel density estimate at x_i are*

$$\mathbb{E}\left[\rho_\epsilon^{(n)}(x_i) - \rho_\epsilon(x_i)\right] = \frac{1}{n}\left(1 - (\pi\epsilon)^{d/2}\left[\rho(x_i) + O(\epsilon)\right]\right), \tag{2.50}$$

$$\mathsf{Var}\left(\rho_\epsilon^{(n)}(x_i) - \rho_\epsilon(x_i)\right) = \frac{1}{n}\left(\frac{\pi\epsilon}{2}\right)^{d/2}\left(\rho(x_i) + O(\epsilon^{\min\{1,d/2\}}) + O(n^{-1})\right). \tag{2.51}$$

Proof. We will use expressions obtained in the proofs of Lemmas 2.3.1 and 2.3.2 in Section 2.3.2 below. Recall that the error in the KDE is given by

$$\rho_\epsilon^{(n)}(x_i) - \rho_\epsilon(x_i) = \frac{1}{n}\sum_{j=1}^{n}\left(k_\epsilon(x_i, x_j) - \rho_\epsilon(x_i)\right) =: X_i + \sum_{j\neq i} X_j, \tag{2.52}$$

where X_j, $j \neq i$, are independent random variables with mean zero and variance

$$\mathsf{Var}(X_j) = \frac{1}{n^2}\left[\left(\frac{\pi\epsilon}{2}\right)^{d/2}\left(\rho(x_i) + O(\epsilon^{\min\{1,d/2\}})\right)\right]. \tag{2.53}$$

In contrast, X_i is a deterministic number

$$X_i = \frac{1}{n}\left(k_\epsilon(x_i, x_i) - \rho_\epsilon(x_i)\right) = \frac{1}{n}\left(1 - (\pi\epsilon)^{d/2}\left[\rho(x_i) + O(\epsilon)\right]\right). \tag{2.54}$$

Therefore,

$$\mathbb{E}\left[\rho_\epsilon^{(n)}(x_i) - \rho_\epsilon(x_i)\right] = X_i + (n-1)\mathbb{E}[X_j] = \frac{1}{n}\left(1 - (\pi\epsilon)^{d/2}\left[\rho(x_i) + O(\epsilon)\right]\right), \tag{2.55}$$

and

$$\begin{aligned}\mathsf{Var}\left(\rho_\epsilon^{(n)}(x_i) - \rho_\epsilon(x_i)\right) &= (n-1)\mathsf{Var}(X_j)\\ &= \frac{1}{n}\left(\frac{\pi\epsilon}{2}\right)^{d/2}\left(\rho(x_i) + O(\epsilon^{\min\{1,d/2\}}) + O(n^{-1})\right).\end{aligned} \tag{2.56}$$

Note that estimates (2.50) and (2.51) do not assume any relationship between the rates at which $\epsilon \to 0$

and $n \to \infty$. □

The standard deviation of the error of the KDE (2.51) is

$$\sqrt{\mathsf{Var}\left(\rho_\epsilon^{(n)}(x_i) - \rho_\epsilon(x_i)\right)} = \frac{1}{n^{1/2}} \left(\frac{\pi\epsilon}{2}\right)^{d/4} \left(\rho^{1/2}(x_i) + o(1)\right). \tag{2.57}$$

Equating it with the bound (2.29) for the KDE in Theorem 2.2.8 and setting $\alpha = 1$ we obtain

$$\frac{1}{n^{1/2}} \left(\frac{\pi\epsilon}{2}\right)^{d/4} \left(\rho^{1/2}(x_i) + o(1)\right) = 3(\pi\epsilon)^{d/2}\rho^{1/2}(x_i)\epsilon^2. \tag{2.58}$$

Therefore, to make the typical error as small as the bound in Theorem 2.2.8 one needs to choose

$$n = \frac{(\pi\epsilon)^{d/2}\epsilon^4}{3 \cdot 2^{d/2}}. \tag{2.59}$$

This scaling for n is just a minor improvement over (2.49).

2.2.6 Kernel density estimators for a regular triangular grid

It is evident from the proof of Theorem 2.2.12 that the inclusion of the point x_i in the kernel density estimator at x_i makes it biased. Therefore, it is reasonable to ask, should we remove the point x_i from the kernel density estimator at x_i? On one hand, the removal of x_i makes the estimator unbiased. On the other hand, thinking of δ-nets, the removal of x_i creates a "hole" around it making the point cloud uniform to a lesser extent in the neighborhood of x_i.

To probe this issue quantitatively, we consider the following example. Let $\mathcal{X}(n)$ be a subset of an equilateral triangular grid with step $\delta = n_r^{-1}$ shaped as a right hexagon of radius 1 shown in Fig.

2.2 (Left). The relationship between n and δ is

$$n = 1 + 6\sum_{k=1}^{n_r} k = 3n_r^2 + 3n_r + 1 = 3\delta^{-2} + 3\delta^{-1} + 1. \tag{2.60}$$

This hexagon is an instance of delta-net with parameter δ, an ideal delta-net. The area of the hexagon is $\frac{1}{2}3\sqrt{3}$ and the density is uniform. Hence, by (2.26),

$$\rho_\epsilon = (\pi\epsilon)^{d/2}\frac{2}{3\sqrt{3}} \tag{2.61}$$

if the integration were done over $\mathbb{R}^2$. Let us calculate the KDE at a point $x_i \in \mathcal{X}(n)$ lying at a distance greater or equal to $3\sqrt{\epsilon}$ from the boundary of the hexagon with and without the removal of x_i from the estimator. The minimal distance of $3\sqrt{\epsilon}$ from the boundary is motivated by the commonly used sparsification in diffusion map algorithms. The KDE $\rho_\epsilon^{(n)}(x_i)$ is

$$\rho_\epsilon^{(n)}(x_i) = \begin{cases} \frac{1}{n}\left(1 + 6\sum_{k=1}^{\lfloor\frac{3\sqrt{\epsilon}}{\delta}\rfloor}\exp\left(-\frac{k^2\delta^2}{\epsilon}\right)\right), & \text{with } x_i \\ \frac{6}{n}\sum_{k=1}^{\lfloor\frac{3\sqrt{\epsilon}}{\delta}\rfloor}\exp\left(-\frac{k^2\delta^2}{\epsilon}\right), & \text{without } x_i. \end{cases} \tag{2.62}$$

The plots of the relative error in the KDE given by (2.62) with respect to ρ_ϵ given by (2.61),

$$E_{rel} = \frac{\left|\rho_\epsilon^{(n)} - \rho_\epsilon\right|}{\rho_\epsilon} \tag{2.63}$$

for the biased (with x_i, solid curves) and unbiased (without x_i, dashed curves) estimators for $n_r = 50$, 150, 250, 350, and 450 are displayed in Fig. 2.2 (Middle). The corresponding values of n are given by (2.60) and are shown in the legend.

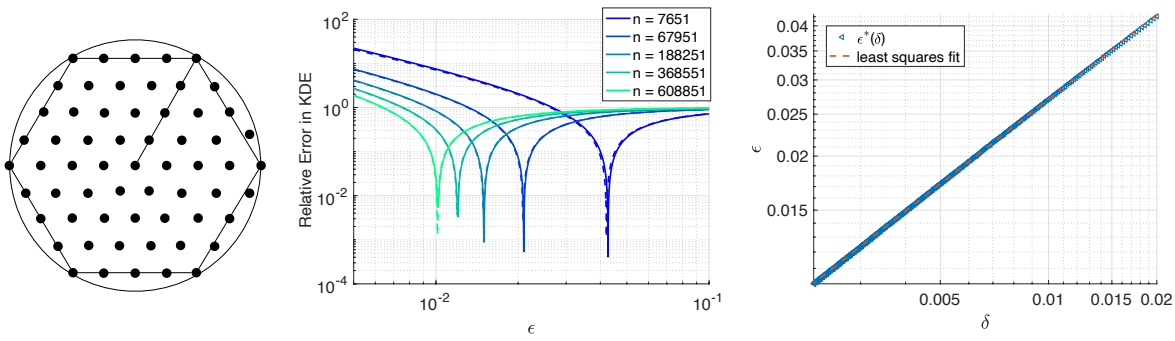


Figure 2.2: Left: A point cloud is a hexagon-shaped regular equilateral triangular grid. The radius of the circumcircle is one. Middle: Plots of biased (solid) and unbiased (dashed) kernel density estimates (KDEs) (2.62) for $n_r = 50$, 150, 250, 350, and 450. The corresponding values of n calculated using (2.60) are shown in the legend. Right: The graph of the optimal value of ϵ that minimizes the relative error in ρ_ϵ versus $\delta = n_r$ and the least squares fit for it.

Inspection of these graphs allows us to make the following observations.

- First, the relative errors for the biased (solid) and unbiased (dashed) are close to each other. A suboptimal choice of ϵ affects the relative error much stronger than the choice of the biased versus unbiased estimator. Unsurprisingly, at larger ϵ, the biased estimator gives a slightly smaller error. On the other hand, at smaller ϵ, the unbiased one is slightly more accurate. Therefore, in practice, the choice of a biased or unbiased estimator is not essential.

- These graphs show that the minimum of the relative error is achieved at a certain relationship between $\delta = n_r^{-1}$ and ϵ. The plot in Fig. 2.2 (Right) shows the optimal ϵ for each δ. The least squares fit to a power law (the red dashed line in Fig. 2.2 (Right)) yields the following relationship between δ and ϵ:

$$\epsilon^* = 0.5425 \cdot \delta^{0.6509}. \tag{2.64}$$

Relationship (2.64) can be used as a guideline for choosing ϵ given the parameter δ of a δ-net.

- Equations (2.64) and (2.60) imply the following scaling for ϵ and n:

$$n = 0.8829\epsilon^{-1.3018}. \tag{2.65}$$

This scaling is much better than the one in (2.49).

Remark 2.2.13. *In practice, δ-nets are obtained by subsampling point clouds and do not have points at a distance closer than δ. Such a subsampling breaks the i.i.d. hypothesis on which the KDE error in Theorem 2.2.8 relies. The lack of the i.i.d. hypothesis requires the development of a different approach for the derivation of the variance error. This problem will be addressed in future work.*

2.2.7 Total error model

Theorems 2.2.5 and 2.2.11 imply the total error bound given in Theorem 2.1.1 placed in Section 2.1. Theorem 2.1.1 implies the following modes of convergence given the limits of n and ϵ:

1. To obtain almost sure convergence using the Borel-Cantelli lemma as $n \to \infty$ and $\epsilon \to 0$ it suffices to take

$$\lim_{\substack{n\to\infty \\ \epsilon\to 0}} \frac{n\epsilon^{2+d/2}}{\log n} = \infty. \tag{2.66}$$

This would leave the variance error converging at $o(1)$ rate which is slower than $O(\epsilon)$.

2. To make the variance and bias errors both $O(\epsilon)$, we must require the stronger limit

$$\lim_{\substack{n\to\infty \\ \epsilon\to 0}} \frac{n\epsilon^{4+d/2}}{\log n} = \infty. \tag{2.67}$$

The additional deceleration of ϵ is needed in (2.67) to ensure the discrete kernel density estimate $\rho_\epsilon^{(n)}$ is within $O(\epsilon^2)$ of the continuous kernel density estimate ρ_ϵ for each point x_i as $n \to \infty$ and $\epsilon \to 0$, and then the division by ϵ would get us to within $O(\epsilon)$ with summably decaying probability.

2.2.8 Numerical solution of Dirichlet BVPs with TMD maps

We now focus on the numerical solution to the Dirichlet boundary value problem (BVP) (2.8). Here we make the following assumption about the domain Ω:

Assumption 2.2.14. *$\Omega \subseteq \mathcal{M}$ is a closed subset of $\mathcal{M}$ with C^1 boundary.*

Obtaining numerical solutions of BVP (2.8) on manifolds is an important application of diffusion maps-based estimators of diffusion operators such as $\mathcal{L}$ [BS16a, JH23, ABH21]. An instance of this BVP is the *committor problem* (1.54). We will solve the BVP (2.8) numerically using the TMDmap algorithm. The resulting solution will be denoted by $v \equiv v_{n,\epsilon}$. The exact solution to BVP (2.8) will be denoted by u. Using a maximum principle argument, we can transfer the consistency estimate from Theorem 2.1.1 to an error estimate for quantifying the accuracy of the numerical solution $v_{n,\epsilon}$.

Theorem 2.2.15 (Error bound for the numerical solution)**.** *Consider the point cloud $\mathcal{X}(n)$ with n i.i.d. samples of ρ. Let $v_{n,\epsilon}$ be a TMDmap numerical solution to BVP (2.8), and let u be the exact solution to (2.8). Furthermore, let $n \to \infty$ and $\epsilon \to 0$ so that*

$$\lim_{\substack{n\to\infty \\ \epsilon\to 0}} \frac{n\epsilon^{2+d/2}}{\log n} = \infty. \tag{2.68}$$

Let $\mathbb{I}_\Omega$ be the indicator of Ω. There exist constants $\epsilon_0 > 0$ and $n_0 \in \mathbb{N}$ such that for all $\epsilon \leq \epsilon_0$ and

$n \geq n_0$ with probability greater or equal to $1 - 4n^{-2} - \exp\left(-n\mathbb{E}[\mathbb{I}_\Omega]\right)$

$$|v_{n,\epsilon}(x_i) - u(x_i)| \leq \epsilon \left[\max_{x\in\mathcal{M}} C_{u,\mu,\rho}(x) + 1\right] |\phi(x_i)| \tag{2.69}$$

where ϕ is the exact solution to $\mathcal{L}\phi = 1$, $x \in \mathcal{M}\backslash\Omega$ and $\phi = 0$, $x \in \partial\Omega$, and

$$\begin{aligned} C_{u,\mu,\rho}(x) := \max_{\substack{\epsilon\leq\epsilon_0 \\ n\geq n_0}} \Bigg[& \frac{\alpha\left(2\|\nabla f(x)\|\epsilon^{1/2} + 11|f(x)|\right)}{\rho^{1/2}(x)} \\ & + |\mathcal{B}_1[u,\mu](x) + \mathcal{B}_2[u,\mu,\rho](x) + \mathcal{B}_3[u,\mu,\rho](x)| + O(\epsilon)\Bigg], \end{aligned} \tag{2.70}$$

where the functions $\mathcal{B}_j[\cdot](\cdot)$, $j = 1, 2, 3$, are defined in Theorem 2.1.1.

The proof of Theorem 2.2.15 is found in Section 2.3.3. Moreover, Theorem 2.2.15 combined with Corollary 2.2.7 straightaway implies the following error estimate for the solution to the committor problem.

Corollary 2.2.16. *Under Assumptions 2.2.1–2.2.14, let $q_{n,\epsilon}$ be the TMDmap numerical solution to the committor problem (2.7). Let $n \to \infty$ and $\epsilon \to 0$ so that (2.68) holds. Suppose that $\mathcal{M}$ is flat and $\rho = \mathrm{vol}(\mathcal{M})^{-1}$ is the uniform density on $\mathcal{M}$. Then there exists a constant $C_q(x) > 0$ given by*

$$C_q(x) := \max_{\substack{\epsilon\leq\epsilon_0 \\ n\geq n_0}} \left[\alpha\,\mathrm{vol}(\mathcal{M})^{1/2}\left(2\|\nabla q(x)\|\epsilon^{1/2} + 11q(x)\right)\right. \tag{2.71}$$

$$\left. + \left|\frac{1}{4}\left[\mathcal{Q}\left(q\mu^{1/2}\right) - q\mathcal{Q}(\mu^{1/2})\right]\right| + O(\epsilon)\right], \tag{2.72}$$

such that

$$|q_{n,\epsilon}(x_i) - q(x_i)| \leq \epsilon \left[\max_{x\in\mathcal{M}} C_q(x) + 1\right] |\phi(x_i)| + O(\epsilon^2) \tag{2.73}$$

with high probability.

2.3 Results: proofs

2.3.1 Bias Error: proofs

We begin with Lemma 2.2.6 that gives the second-order kernel expansion formula (2.21).

Proof. The strategy of proving (2.21) is standard in manifold learning, see e.g. S. Lafon's dissertation [Laf04]. We present it here nonetheless, specifically to prove that the $O(\epsilon^2)$ prefactor is a fourth-order differential operator and that it is given by (2.22) in the case if $\mathcal{M}$ is locally flat.

The computation of the Taylor expansion of the integral

$$\int_{\mathcal{M}} k_\epsilon(x,y) f(y)\, d\mathrm{vol}(y) \tag{2.74}$$

in terms of ϵ is done in three steps.

1. **Reduce to a ball:** We first note that the integral over the manifold in (2.74) can be replaced with an integral over any arbitrarily small open set around $U \ni x$, incurring an error that decays faster than any polynomial in ϵ as $\epsilon \to 0$. This is formalized in [BN08, Lemma 4.1] as follows:

$$\int_{\mathcal{M}\setminus U} k_\epsilon(x,y) f(y)\, d\mathrm{vol}(y) = o(\epsilon^l) \tag{2.75}$$

for any $l > 0$. This error will turn out to be dominated by the $O(\epsilon)$ error of approximating the Laplacian. This approach allows us to focus on computing the integral over U, which can now be expressed as the image of the exponential map on a small ball in the tangent space.

2. **Use exponential coordinates:** To compute the integral

$$\int_U k_\epsilon(x, y) f(y)\, d\mathrm{vol}(y), \tag{2.76}$$

we write $U = \exp_x (B_r(0))$ where $\exp_x$ is the exponential map from $T_x\mathcal{M}$ to $\mathcal{M}$ taking $0 \to x$ and $r = \epsilon^l$ is smaller than $B_{\text{inj}(\mathcal{M})}$ where $0 < l < 1/2$. Note that the Euclidean coordinates $(u_1, \ldots, u_d)$ are mapped by $\exp_x$ to a geodesic: $tu_i \to \gamma_i(t)$ and $s(u)$ is the respective system of normal coordinates $U \to \mathcal{M}$ where $s(0) = x$. Since $y = s(u)$ is a diffeomorphism from $B_r(0)$ onto its image we can use a change of variables:

$$\int_{\exp(B_r(0))} k_\epsilon(x, y) f(y)\, d\mathrm{vol}(y) = \int_{B_r(0)} k_\epsilon(y(0), y(u)) f(y(u)) dy(u)\, du. \tag{2.77}$$

Here $dy(u)$ is the Jacobian of the parametrization of y in terms of the Euclidean coordinates u. To pull the integral back onto the tangent space, we need to expand the *extrinsic* distance on $\mathcal{M}$ in terms of the distance on the tangent plane. Furthermore, we will write the Jacobian as a power series in terms of u. All of these expansions will have the following correction terms [SWW07, Corollary 3, Prop. 6]:

$$\|s(u) - s(0)\|_2^2 = \|u\|_2^2 + \mathcal{C}_x^1(u), \quad \mathcal{C}_x^1(u) = O(\|u\|_2^4) \tag{2.78}$$

$$dy(u) = 1 + \mathcal{C}_x^2(u), \quad \mathcal{C}_x^2(u) = O(\|u\|_2^2) \tag{2.79}$$

3. **Analysis on Euclidean space:** After expressing the integral in (2.77) over a subset of Euclidean space, we compute these integrals for up to the fourth order power series expansions of $f \circ y$ in terms of u. For notational convenience, denote $F : U \to \mathbb{R}$ as $F := f \circ s$.

$$\begin{aligned}
(\pi\epsilon)^{-d/2}\mathcal{G}_\epsilon f(x) &= \int_{B_r(0)} k_\epsilon(y(0), y(u)) f(y(u)) dy(u)\, du + o(\epsilon^l) \\
&= \int_{B_r(0)} k_\epsilon(y(0), y(u)) f(y(u)) dy(u)\, du + o(\epsilon^l) \\
&= \int_{B_r(0)} \exp\left(-\frac{\|u\|_2^2}{\epsilon} - \mathcal{C}_x^1(u)\right) T_4 F(u)(1 + \mathcal{C}_x^2(u))\, du + o(\epsilon^l) \\
&= \int_{\mathbb{R}^d} \exp\left(-\frac{\|u\|_2^2}{\epsilon} - \mathcal{C}_x^1(u)\right) T_4 F(u)(1 + \mathcal{C}_x^2(u))\, du + o(\epsilon^l) \\
&= f(x) + \frac{\epsilon}{4}\left(\Delta f(x) - \omega(x) f(x)\right) + \frac{\epsilon^2}{4}\mathcal{Q} f(x) + O(\epsilon^3). \qquad (2.80)
\end{aligned}$$

The integral above can be distributed and solved using Gaussian integral formulas. All odd powers in u will cancel, resulting in the appearance of leading order terms $O(1), O(\epsilon)$, and $O(\epsilon^2)$, with the prefactors depending on x, f, and $\mathcal{M}$. The function ω arises in the $O(\epsilon)$ term due to the correction factor $\mathcal{C}_x^1(u)$; the analytical formula for ω can be found in [Laf04, Prop. 7]. Additionally, the $O(\epsilon^2)$ terms arise as the integration of fourth-order monomials against the Gaussian kernel. These monomials will arise multiplied with *at most* the fourth derivatives of f. Consequently, the prefactor $\mathcal{Q}$ on the $O(\epsilon^2)$ term is a fourth-order (non-linear) differential operator.

Additionally, if $g \equiv I$ for a small enough neighborhood around x, then the correction terms in (2.78)

and (2.79) are zero. As a consequence, (2.80) simplifies to:

$$
\begin{aligned}
(\pi\epsilon)^{-d/2}\mathcal{G}_\epsilon f(x) &= \int_{\mathbb{R}^d} \exp\left(-\frac{\|u\|_2^2}{\epsilon} - \mathcal{C}_x^1(u)\right) T_4F(u)(1+\mathcal{C}_x^2(u))\,du + o(\epsilon^l) \\
&= \int_{\mathbb{R}^d} \exp\left(-\frac{\|u\|_2^2}{\epsilon}\right) T_4F(u)\,du + o(\epsilon^l) \\
&= \int_{\mathbb{R}^d} \exp\left(-\frac{\|u\|_2^2}{\epsilon}\right)\left(f(x) + \partial_i f(x)u^i + \frac{1}{2}\partial_{ij}f(x)u^iu^j\right. \\
&\left. + \frac{1}{6}\partial_{ijk}f(x)u^iu^ju^k + \frac{1}{24}\partial_{ijkl}f(x)u^iu^ju^ku^l\right)du + o(\epsilon^l) \\
&= f(x) + \frac{\epsilon}{4}\Delta f(x) + \frac{\epsilon^2}{4}\left(\frac{1}{6}\left[\partial_i^i\partial_j^j f + 2\partial_{ii}^{ii}f(x)\right]\right) + O(\epsilon^3).
\end{aligned}
$$

The above argument also shows that under the condition $g \equiv I$ the term $\omega(x) = 0$. □

Next, we prove Theorem 2.2.5 that gives the bias error (2.19).

Proof. (Theorem 2.2.5) Lemma 2.2.6 implies that

$$
\begin{aligned}
\rho_\epsilon(x) &= \int_{\mathcal{M}} k_\epsilon(x,y)\rho(y)dy \\
&= (\pi\epsilon)^{d/2}\left[\rho(x) + \frac{\epsilon}{4}\left(\Delta\rho(x) - \omega(x)\rho(x)\right) + \frac{\epsilon^2}{4}\mathcal{Q}(\rho) + O(\epsilon^3)\right].
\end{aligned} \tag{2.81}
$$

Factoring out ρ and using a geometric series expansion we obtain

$$
\begin{aligned}
\rho_\epsilon^{-1}(x) &= (\pi\epsilon)^{-d/2}\rho^{-1}(x)\left[1 + \left(-\frac{\epsilon}{4}\left(\frac{\Delta\rho}{\rho} - \omega\right) - \frac{\epsilon^2}{4}\frac{\mathcal{Q}(\rho)}{\rho} - O(\epsilon^3)\right)\right. \\
&\left. + \left(-\frac{\epsilon}{4}\frac{\Delta\rho}{\rho} - \frac{\epsilon^2}{4}\frac{\mathcal{Q}(\rho)}{\rho} - O(\epsilon^3)\right)^2 + O(\epsilon^3)\right] \\
&= (\pi\epsilon)^{-d/2}\rho^{-1}(x)\left[1 - \frac{\epsilon}{4}\left(\frac{\Delta\rho}{\rho} - \omega\right) - \frac{\epsilon^2}{4}\frac{\mathcal{Q}(\rho)}{\rho} + \frac{\epsilon^2}{16}\left(\frac{\Delta\rho}{\rho} - \omega\right)^2 + O(\epsilon^3)\right]
\end{aligned} \tag{2.82}
$$

Now we compute the right-normalized kernel operator that factors in the effect of the target density μ:

$$K_{\epsilon,\mu}f(x) = \int_{\mathcal{M}} k_\epsilon(x,y)\mu^{1/2}(y)\rho_\epsilon^{-1}(y)f(y)\rho(y)d\text{vol}(y) \tag{2.83}$$

Plugging (2.82) into (2.83) we get:

$$\begin{aligned}
K_{\epsilon,\mu}f(x) &= \int_{\mathcal{M}} k_\epsilon(x,y)\frac{\mu^{1/2}f\rho}{(\pi\epsilon)^{d/2}\rho}\left[1-\frac{\epsilon}{4}\left(\frac{\Delta\rho}{\rho}-\omega\right)-\frac{\epsilon^2}{4}\frac{\mathcal{Q}(\rho)}{\rho}\right. \\
&\left.+\frac{\epsilon^2}{16}\left(\frac{\Delta\rho}{\rho}-\omega\right)^2+O(\epsilon^3)\right]d\text{vol}(y) \\
&= (\pi\epsilon)^{-d/2}\left[\int_M k_\epsilon(x,y)\mu^{1/2}f(y)\,d\text{vol}(y)-\int_{\mathcal{M}} k_\epsilon(x,y)\mu^{1/2}f\frac{\epsilon}{4}\left(\frac{\Delta\rho}{\rho}-\omega\right)d\text{vol}(y)\right. \\
&-\int_M k_\epsilon(x,y)\mu^{1/2}f\frac{\epsilon^2}{4}\frac{\mathcal{Q}(\rho)}{\rho}\,d\text{vol}(y) \\
&\left.+\int_{\mathcal{M}} k_\epsilon(x,y)\mu^{1/2}f\frac{\epsilon^2}{16}\left(\frac{\Delta\rho}{\rho}-\omega\right)^2 d\text{vol}(y)+O(\epsilon^3)\right].
\end{aligned} \tag{2.84}$$

Computing the integrals in (2.84) using the second-order kernel expansion formula (2.21) we obtain

$$\begin{aligned}
K_{\epsilon,\mu}f(x) &= \mu^{1/2}f+\frac{\epsilon}{4}\Delta(\mu^{1/2}f)+\frac{\epsilon^2}{4}\mathcal{Q}(\mu^{1/2}f)-\mu^{1/2}f\frac{\epsilon}{4}\left(\frac{\Delta\rho}{\rho}-\omega\right) \\
&-\frac{\epsilon^2}{16}\Delta\left(\mu^{1/2}f\left[\frac{\Delta\rho}{\rho}-\omega\right]\right)-\frac{\epsilon^2}{4}\mu^{1/2}f\frac{\mathcal{Q}(\rho)}{\rho}+\frac{\epsilon^2}{16}\mu^{1/2}f\left(\frac{\Delta\rho}{\rho}-\omega\right)^2+O(\epsilon^3).
\end{aligned} \tag{2.85}$$

Finally, we group the terms in (2.85) according to the order of ϵ:

$$\begin{aligned}\mathcal{K}_{\epsilon,\mu}f = f\mu^{1/2} + \frac{\epsilon}{4}\left[\Delta(\mu^{1/2}f) - f\mu^{1/2}\left(\frac{\Delta\rho}{\rho} - \omega\right)\right] + \\ + \frac{\epsilon^2}{4}\left[\mathcal{Q}\left(f\mu^{1/2}\right) - \frac{1}{4}\Delta\left(f\mu^{1/2}\left(\frac{\Delta\rho}{\rho} - \omega\right)\right) + f\mu^{1/2}\left(\frac{1}{4}\left(\frac{\Delta\rho}{\rho} - \omega\right)^2 - \frac{\mathcal{Q}(\rho)}{\rho}\right)\right] \\ + O(\epsilon^3). \end{aligned} \tag{2.86}$$

To make the expression (2.86) more compact, we denote the operator in the last square brackets by

$$\mathcal{R}_{\mu,\rho}f := \mathcal{Q}\left(f\mu^{1/2}\right) - \frac{1}{4}\Delta\left(f\mu^{1/2}\left(\frac{\Delta\rho}{\rho} - \omega\right)\right) + f\mu^{1/2}\left(\frac{1}{4}\left(\frac{\Delta\rho}{\rho} - \omega\right)^2 - \frac{\mathcal{Q}(\rho)}{\rho}\right). \tag{2.87}$$

Then $\mathcal{K}_{\epsilon,\mu}f$ in (2.86) can be written as

$$\mathcal{K}_{\epsilon,\mu}f = f\mu^{1/2} + \frac{\epsilon}{4}\left[\Delta(\mu^{1/2}f) - f\mu^{1/2}\left(\frac{\Delta\rho}{\rho} - \omega\right)\right] + \frac{\epsilon^2}{4}\mathcal{R}_{\mu,\rho}f + O(\epsilon^3). \tag{2.88}$$

Now we are ready to calculate the Markov operator $\mathcal{P}_{\epsilon,\mu}f$:

$$\mathcal{P}_{\epsilon,\mu}f = \frac{\mathcal{K}_{\epsilon,\mu}f}{\mathcal{K}_{\epsilon,\mu}1} = \frac{f\mu^{1/2} + \frac{\epsilon}{4}\left[\Delta(\mu^{1/2}f) - f\mu^{1/2}\left(\frac{\Delta\rho}{\rho} - \omega\right)\right] + \frac{\epsilon^2}{4}\mathcal{R}_{\mu,\rho}f + O(\epsilon^3)}{\mu^{1/2} + \frac{\epsilon}{4}\left[\Delta(\mu^{1/2}) - \mu^{1/2}\left(\frac{\Delta\rho}{\rho} - \omega\right)\right] + \frac{\epsilon^2}{4}\mathcal{R}_{\mu,\rho}1 + O(\epsilon^3)}. \tag{2.89}$$

Dividing the numerator and the denominator by $\mu^{1/2}$ and applying Taylor expansion we get:

$$\begin{aligned}
\mathcal{P}_{\epsilon,\mu} f &= \left(f + \frac{\epsilon}{4}\left[\frac{\Delta(\mu^{1/2} f)}{\mu^{1/2}} - f\left(\frac{\Delta\rho}{\rho} - \omega\right)\right] + \frac{\epsilon^2}{4}\frac{\mathcal{R}_{\mu,\rho} f}{\mu^{1/2}} + O(\epsilon^3)\right) \\
&\times \left(1 - \frac{\epsilon}{4}\left[\frac{\Delta(\mu^{1/2})}{\mu^{1/2}} - \left(\frac{\Delta\rho}{\rho} - \omega\right)\right] - \frac{\epsilon^2}{4}\frac{\mathcal{R}_{\mu,\rho} 1}{\mu^{1/2}} + \frac{\epsilon^2}{16}\left[\frac{\Delta(\mu^{1/2})}{\mu^{1/2}} - \left(\frac{\Delta\rho}{\rho} - \omega\right)\right]^2 + O(\epsilon^3)\right) \\
&= f + \frac{\epsilon}{4}\left[\frac{\Delta(\mu^{1/2} f)}{\mu^{1/2}} - f\frac{\Delta(\mu^{1/2})}{\mu^{1/2}}\right] + \frac{\epsilon^2}{4}\left[\frac{\mathcal{R}_{\mu,\rho} f}{\mu^{1/2}} - f\frac{\mathcal{R}_{\mu,\rho} 1}{\mu^{1/2}}\right] \\
&+ \frac{\epsilon^2}{16}\left(f\left[\frac{\Delta(\mu^{1/2})}{\mu^{1/2}} - \left(\frac{\Delta\rho}{\rho} - \omega\right)\right]^2\right. \\
&\left. - \left[\frac{\Delta(\mu^{1/2} f)}{\mu^{1/2}} - f\left(\frac{\Delta\rho}{\rho} - \omega\right)\right]\left[\frac{\Delta(\mu^{1/2})}{\mu^{1/2}} - \left(\frac{\Delta\rho}{\rho} - \omega\right)\right]\right) + O(\epsilon^3).
\end{aligned} \tag{2.90}$$

The last term in (2.90) can be simplified by noting that

$$\begin{aligned}
&f\left[\frac{\Delta(\mu^{1/2})}{\mu^{1/2}} - \left(\frac{\Delta\rho}{\rho} - \omega\right)\right]^2 - \left[\frac{\Delta(\mu^{1/2} f)}{\mu^{1/2}} - f\left(\frac{\Delta\rho}{\rho} - \omega\right)\right]\left[\frac{\Delta(\mu^{1/2})}{\mu^{1/2}} - \left(\frac{\Delta\rho}{\rho} - \omega\right)\right] \\
&= \left[\frac{\Delta(\mu^{1/2})}{\mu^{1/2}} - \left(\frac{\Delta\rho}{\rho} - \omega\right)\right]\left[f\frac{\Delta(\mu^{1/2})}{\mu^{1/2}} - \frac{\Delta\mu^{1/2} f}{\mu^{1/2}}\right].
\end{aligned} \tag{2.91}$$

Taking (2.91) into account, we write out the generator$\mathcal{L}_{\epsilon,\mu} f$:

$$\begin{aligned}
\mathcal{L}_{\epsilon,\mu} f = \frac{\mathcal{P}_{\epsilon,\mu} f - f}{\epsilon} &= \frac{1}{4}\left[\frac{\Delta(\mu^{1/2} f)}{\mu^{1/2}} - f\frac{\Delta(\mu^{1/2})}{\mu^{1/2}}\right] + \frac{\epsilon}{4}\left[\frac{\mathcal{R}_{\mu,\rho} f}{\mu^{1/2}} - f\frac{\mathcal{R}_{\mu,\rho} 1}{\mu^{1/2}}\right] \\
&+ \frac{\epsilon}{16}\left[\frac{\Delta(\mu^{1/2})}{\mu^{1/2}} - \left(\frac{\Delta\rho}{\rho} - \omega\right)\right]\left[f\frac{\Delta(\mu^{1/2})}{\mu^{1/2}} - \frac{\Delta\mu^{1/2} f}{\mu^{1/2}}\right] + O(\epsilon^2).
\end{aligned} \tag{2.92}$$

It is straightforward to check that the limit $\epsilon \to 0$ is

$$\lim_{\epsilon\to 0} \mathcal{L}_{\epsilon,\mu} = \frac{1}{4}\left[\frac{\Delta(\mu^{1/2} f)}{\mu^{1/2}} - f\frac{\Delta(\mu^{1/2})}{\mu^{1/2}}\right] = \frac{1}{4}\left[\Delta f + \nabla \log\mu \cdot \nabla f\right] \equiv \frac{\beta}{4}\mathcal{L}. \tag{2.93}$$

The term $\epsilon\mathcal{B}_3[f,\mu,\rho](x)$ in (2.19) is the first term in the second line of (2.92). The terms $\epsilon\mathcal{B}_1[f,\mu](x)$ and $\epsilon\mathcal{B}_2[f,\mu,\rho](x)$ in (2.19) are recovered by computing the term with the operators $\mathcal{R}_{\mu,\rho}$ in (2.92). □

2.3.2 Variance error: proofs

2.3.2.1 Bounds for the error in the kernel density estimate

To prove Theorem 2.2.8 establishing error bound for the discrete kernel density estimate (KDE), we first derive concentration inequalities for the discrepancy between $\rho_\epsilon(x)$ and $\rho_\epsilon^{(n)}(x)$ in Lemmas 2.3.1 and 2.3.2 below. There are two cases requiring somewhat different reasoning. In Lemma 2.3.1, the point x is chosen independently of the point cloud $\mathcal{X}(n)$, while in Lemma 2.3.2, the point x belongs to $\mathcal{X}(n)$.

Lemma 2.3.1. *Let $\mathcal{X}(n) := \{x_j\}_{j=1}^n \subset \mathcal{M}$ be a point cloud sampled according to the density $\rho(x)$ and let $x \in \mathcal{M}$ be an arbitrary point selected independently of $\mathcal{X}(n)$. Let $\rho_\epsilon(x)$ and $\rho_\epsilon^{(n)}(x)$ be the KDEs defined by (2.26) and (2.27) respectively. Then for any ϵ sufficiently small and any constant $\alpha = o(\epsilon^{-1})$, the following relationship holds:*

$$\mathbb{P}\Big(\big|\rho_\epsilon^{(n)}(x) - \rho_\epsilon(x)\big| \geq (\pi\epsilon)^{d/2}\rho(x)\epsilon^2\alpha\Big) \leq 2\exp\left[-\frac{n(2\pi)^{d/2}\rho(x)\epsilon^{4+d/2}\alpha^2}{2+O(\epsilon^{\min\{1,d/2\}})}\right]. \tag{2.94}$$

Proof. The difference in the left-hand side of (2.94) can be written as

$$\rho_\epsilon^{(n)}(x) - \rho_\epsilon(x) = \frac{1}{n}\sum_{j=1}^n \left(k_\epsilon(x,x_j) - \rho_\epsilon(x)\right). \tag{2.95}$$

The random variables

$$X_j := \frac{1}{n}\left(k_\epsilon(x, x_j) - \rho_\epsilon(x)\right), \quad 1 \le j \le n, \tag{2.96}$$

have zero mean and are bounded by $|X_j| \le n^{-1}$, $1 \le j \le n$, as $\rho_\epsilon(x) \in (0,1)$ and $\rho_\epsilon^{(n)}(x) \in (0,1]$. The variance of X_j can be calculated as

$$\mathsf{Var}(X_j) = \frac{1}{n^2}\left(\mathbb{E}\left[k_\epsilon(x, x_j)^2\right] - \mathbb{E}^2\left[k_\epsilon(x, x_j)\right]\right). \tag{2.97}$$

The expectations in (2.97) are calculated with the aid of Lemma 2.2.6 and the observation that $k_\epsilon(\cdot,\cdot)^2 = k_{\epsilon/2}(\cdot,\cdot)$:

$$\begin{aligned}\mathbb{E}[k_\epsilon(x,\cdot)^2] &= \int_{\mathcal{M}} (k_\epsilon(x,y))^2 \rho(y)\, d\mathrm{vol}(y) \\ &= \int_{\mathcal{M}} k_{\epsilon/2}(x,y)\rho(y)\, d\mathrm{vol}(y) = \left(\frac{\pi\epsilon}{2}\right)^{d/2}\left(\rho(x) + O(\epsilon)\right);\end{aligned} \tag{2.98}$$

$$\mathbb{E}^2[k_\epsilon(x,\cdot)] = \left(\int_{\mathcal{M}} k_\epsilon(x,y)\rho(y)\, d\mathrm{vol}(y)\right)^2 = (\pi\epsilon)^d\left(\rho^2(x) + O(\epsilon)\right). \tag{2.99}$$

Therefore,

$$\mathsf{Var}(X_j) = \frac{1}{n^2}\left[\left(\frac{\pi\epsilon}{2}\right)^{d/2}\left(\rho(x) + O(\epsilon)\right) + O(\epsilon^d)\right] = \frac{1}{n^2}\left[\left(\frac{\pi\epsilon}{2}\right)^{d/2}\left(\rho(x) + O(\epsilon^{\min\{1,d/2\}})\right)\right]. \tag{2.100}$$

The application of Bernstein's inequality (2.25) to the sum of X_js gives

$$\mathbb{P}\left(\frac{1}{n}\sum_{j=1}^{n}\left[k_\epsilon(x,x_j) - \rho_\epsilon(x)\right] \ge t\right) \le \exp\left[-\frac{t^2}{\frac{1}{n}\left(\frac{\pi\epsilon}{2}\right)^{d/2}\left(\rho(x) + O(\epsilon^{\min\{1,d/2\}})\right) + \frac{2}{3}\frac{1}{n}t}\right]. \tag{2.101}$$

We want the relative error

$$\frac{\left|\rho_\epsilon^{(n)}(x) - \rho_\epsilon(x)\right|}{\rho_\epsilon(x)} \tag{2.102}$$

to be small compared to ϵ. Motivated by the formula (2.26) for $\rho_\epsilon(x)$, we choose

$$t = (\pi\epsilon)^{d/2}\rho(x)\epsilon^2\alpha, \tag{2.103}$$

where α is a constant satisfying $\alpha = o(\epsilon^{-1})$. In this case, the term $\frac{2}{3n}t$ in the denominator of (2.101) is $\frac{2}{3n}(\pi\epsilon)^{d/2}o(\epsilon)$. Plugging (2.95) and (2.103) into (2.101) we obtain

$$\mathbb{P}\left(\rho_\epsilon^{(n)}(x) - \rho_\epsilon(x) \geq (\pi\epsilon)^{d/2}\rho(x)\epsilon^2\alpha\right) \leq \exp\left[-\frac{n(2\pi)^{d/2}\rho(x)\epsilon^{4+d/2}\alpha^2}{2 + O(\epsilon^{\min\{1,d/2\}})}\right]. \tag{2.104}$$

Redefining X_j as $-X_j$, $1 \leq j \leq n$ and repeating the argument above, we obtain the same upper bound for the probability $\mathbb{P}\left(\rho_\epsilon^{(n)}(x) - \rho_\epsilon(x) \leq -(\pi\epsilon)^{d/2}\rho(x)\epsilon^2\alpha\right)$. Then the desired result (2.94) readily follows. □

Unfortunately, the estimate obtained in Lemma 2.3.1 is not suitable for the settings of any standard diffusion map algorithm including the TMD map as all quantities of interest including ρ_ϵ are evaluated at the points of the given point cloud. Therefore, x is one of the points of the point cloud $\mathcal{X}(n)$, say, $x \equiv x_i$, which means that x and $\mathcal{X}(n)$ are not sampled independently. The discrepancy between $\rho_\epsilon^{(n)}(x_i)$ and $\rho_\epsilon(x_i)$ is estimated in the following lemma.

Lemma 2.3.2. *Let $\mathcal{X}(n) := \{x_j\}_{j=1}^n \subset \mathcal{M}$ be a point cloud sampled according to the density $\rho(x)$ and let $x_i \in \mathcal{X}(n)$ be any selected point of this point cloud. Let $\rho_\epsilon(x_i)$ and $\rho_\epsilon^{(n)}(x_i)$ be the KDEs defined by (2.26) and (2.27) respectively with $x \equiv x_i$. Then for ϵ small enough, $\alpha = o(\epsilon^{-1})$ and n large enough*

so that $n(\pi\epsilon)^{d/2}\rho(x_i)\epsilon^2\alpha \gg 1$*, the following relationship holds:*

$$\mathbb{P}\Big(\left|\rho_\epsilon^{(n)}(x_i) - \rho_\epsilon(x_i)\right| \geq (\pi\epsilon)^{d/2}\rho(x_i)\epsilon^2\alpha \mid x_i\Big) \leq 2\exp\left[-\frac{n(2\pi)^{d/2}\rho(x_i)\epsilon^{4+d/2}\alpha^2}{2+O(\epsilon^{\min\{1,d/2\}})}\right]. \tag{2.105}$$

Proof. We will use subscripts to denote the conditional probability:

$$\mathbb{P}_{x_i}(\cdot) := \mathbb{P}(\cdot|x_i) \tag{2.106}$$

Under the distribution conditioned on x_i, for $1 \leq j \leq n$, we define X_j as in (2.96). For any $j \neq i$, X_j is a random variable, while for $j = i$, X_i is a deterministic number. Moreover, for ϵ small enough,

$$X_i = \frac{1}{n}\left(k_\epsilon(x_i,x_i) - \rho_\epsilon(x_i)\right) = \frac{1}{n}\left(1 - (\pi\epsilon)^{d/2}\left[\rho(x_i) + O(\epsilon)\right]\right) \lessapprox \frac{1}{n}. \tag{2.107}$$

In particular, $X_i > 0$ for ϵ small enough. Therefore, we get

$$\begin{aligned}\mathbb{P}_{x_i}\left(\rho_\epsilon^{(n)}(x_i) - \rho_\epsilon(x_i) \geq t\right) &= \mathbb{P}_{x_i}\left(\sum_{j\neq i} X_j \geq t - X_i\right) < \mathbb{P}_{x_i}\left(\sum_{j\neq i} X_j \geq t\right) \\ &\leq \exp\left[-\frac{t^2}{2(n-1)\mathsf{Var}(X_j) + \frac{2}{3n}t}\right].\end{aligned} \tag{2.108}$$

We choose t of the form (2.103) so that the relative error (2.102) is small in comparison with ϵ. Plugging (2.103) and (2.100) into (2.108) we obtain

$$\mathbb{P}_{x_i}\left(\rho_\epsilon^{(n)}(x_i) - \rho_\epsilon(x_i) \geq (\pi\epsilon)^{d/2}\rho(x_i)\epsilon^2\alpha\right) \leq \exp\left[-\frac{n(2\pi)^{d/2}\rho(x)\epsilon^{4+d/2}\alpha^2}{2\frac{n-1}{n}\left[1+O(\epsilon^{\min\{1,d/2\}})\right]}\right]. \tag{2.109}$$

The denominator in the fraction in the exponential in (2.109) can be increased by replacing the factor

of $\frac{n-1}{n}$ with 1. Then the argument of the exponential becomes slightly less negative and hence the exponential increases amplifying the inequality. Therefore, we get

$$\mathbb{P}_{x_i}\left(\rho_\epsilon^{(n)}(x_i) - \rho_\epsilon(x_i) \geq (\pi\epsilon)^{d/2}\rho(x_i)\epsilon^2\alpha\right) \leq \exp\left[-\frac{n(2\pi)^{d/2}\rho(x_i)\epsilon^{4+d/2}\alpha^2}{2 + O(\epsilon^{\min\{1,d/2\}})}\right]. \tag{2.110}$$

Next, as in the proof of Lemma 2.3.1, we redefine X_j as $-X_j$, $1 \leq j \leq n$. Then $X_i \in (-n^{-1}, 0)$ and $X_i \gtrapprox -n^{-1}$. Therefore,

$$\begin{aligned}\mathbb{P}_{x_i}\left(\rho_\epsilon^{(n)}(x_i) - \rho_\epsilon(x_i) \leq -t\right) &= \mathbb{P}_{x_i}\left(\sum_{j\neq i} X_j \geq t - X_i\right) < \mathbb{P}_{x_i}\left(\sum_{j\neq i} X_j \geq t + \frac{1}{n}\right) \\ &\leq \exp\left[-\frac{(t+n^{-1})^2}{2(n-1)\mathsf{Var}(X_j) + \frac{2}{3n}(t+n^{-1})}\right]. \end{aligned} \tag{2.111}$$

We observe that the assumption $n(\pi\epsilon)^{d/2}\rho(x_i)\epsilon^2\alpha \gg 1$ implies that for t of the form (2.103) $t \gg n^{-1}$. Therefore, using similar reasoning as in the proof of Lemma 2.3.1 we calculate:

$$\begin{aligned}&\mathbb{P}_{x_i}\left(\rho_\epsilon^{(n)}(x_i) - \rho_\epsilon(x_i) < -(\pi\epsilon)^{d/2}\rho(x_i)\epsilon^2\alpha\right) \leq \\ &\exp\left[-\frac{\left[(\pi\epsilon)^{d/2}\rho(x_i)\epsilon^2\alpha + \frac{1}{n}\right]^2}{\frac{2(n-1)}{n^2}\left[\left(\frac{\pi\epsilon}{2}\right)^{d/2}\left(\rho(x_i) + O(\epsilon^{\min\{1,d/2\}})\right)\right] + \frac{2}{3n}\left[(\pi\epsilon)^{d/2}\rho(x_i)\epsilon^2\alpha + \frac{1}{n}\right]}\right]. \end{aligned} \tag{2.112}$$

The right-hand side of inequality (2.112) can be simplified by making use of the following observations. First, the last term in square brackets in the denominator is $(\pi\epsilon)^{d/2}o(\epsilon)$ and hence it can be incorporated into $O(\epsilon^{\min\{1,d/2\}})$. Second, $\exp(-(\cdot))$ is a decreasing function of $(\cdot)$. Hence, if we remove $\frac{1}{n}$ from the enumerator we increase the exponent and amplify inequality (2.112). Finally, we can

amplify the inequality by replacing removing the factor of $\frac{n-1}{n}$. As a result, we obtain:

$$\mathbb{P}_{x_i}\left(\rho_\epsilon^{(n)}(x_i) - \rho_\epsilon(x_i) < -(\pi\epsilon)^{d/2}\rho(x_i)\epsilon^2\alpha\right) \leq \exp\left[-\frac{n(2\pi)^{d/2}\rho(x_i)\epsilon^{4+d/2}\alpha^2}{2+O(\epsilon^{\min\{1,d/2\}})}\right]. \tag{2.113}$$

The right-hand sides of inequalities (2.110) and (2.113) are identical. Together, they imply the final result (2.105). □

Now we are ready to prove Theorem 2.2.8.

Proof. Let $\mathcal{X}(n)$ be a point cloud sampled according to the density ρ. By Lemmas 2.3.1 and 2.3.2, for any fixed point $x \in \mathcal{M}$ or any point $x \in \mathcal{X}(n)$, for any ϵ small enough, any $\alpha = o(\epsilon^{-1})$, and any n large enough so that $\pi^{d/2}n\epsilon^{2+d/2}\alpha \gg 1$, inequality (2.94) holds. This inequality can be amplified by replacing $\rho(x)$ with ρ^- since this makes the argument of the exponential less negative and hence increases the exponential. As a result, the following uniform bound for all $x \in \mathcal{M}$ holds:

$$\mathbb{P}_x\left(\left|\rho_\epsilon^{(n)}(x) - \rho_\epsilon(x)\right| \geq (\pi\epsilon)^{d/2}\rho(x)\epsilon^2\alpha\right) \leq 2\exp\left[-\frac{n(2\pi)^{d/2}\rho(x)\epsilon^{4+d/2}\alpha^2}{2+O(\epsilon^{\min\{1,d/2\}})}\right]. \tag{2.114}$$

Here

$$\mathbb{P}_x := \mathbb{P}(\cdot \mid x) \tag{2.115}$$

whether $x \in \mathcal{M}$ or $x := x_i \in \mathcal{X}(n)$. If $x \in \mathcal{M}, x \notin \mathcal{X}(n)$ then $\mathbb{P}_x(\cdot) = \mathbb{P}(\cdot)$. Now we will find α such that the right-hand side of inequality (2.114) is less or equal to n^{-4}. We set

$$2\exp\left[-\frac{n(2\pi)^{d/2}\rho(x)\epsilon^{4+d/2}\alpha^2}{2+O(\epsilon^{\min\{1,d/2\}})}\right] \leq n^{-4}. \tag{2.116}$$

Taking logarithms of both sides of (2.116) and rearranging terms gives the following bound for α:

$$\alpha \geq \sqrt{\frac{(4\log n + \log 2)\,(2 + O(\epsilon^{\min\{1,d/2\}}))}{n(2\pi)^{d/2}\rho(x)\epsilon^{4+d/2}}}. \tag{2.117}$$

Bound (2.117) shows that for ϵ small enough and n large enough,

$$\alpha = \frac{5}{(2\pi)^{d/4}}\sqrt{\frac{\log n}{n\rho(x)\epsilon^{4+d/2}}} \tag{2.118}$$

satisfies inequality (2.117).

Now we observe that

$$(\pi\epsilon)^{d/2}\rho(x)\epsilon^2\alpha = (\pi\epsilon)^{d/2}\rho^{1/2}(x)\epsilon^2\frac{5}{(2\pi)^{d/4}}\sqrt{\frac{\log n}{n\epsilon^{4+d/2}}}.$$

We redefine α as

$$\alpha = \frac{1}{(2\pi)^{d/4}}\sqrt{\frac{\log n}{n\epsilon^{4+d/2}}} \tag{2.119}$$

and obtain the desired inequality (2.29) for each point x_i. To move from conditional probabilities to overall probabilities, we let $E(x_i)$ be the event that (2.29) holds. Moreover, let $\mathbb{I}_{E(x_i)}$ be the indicator function of this event. Then using the tower of expectations,

$$\mathbb{P}(E(x_i)) = \mathbb{E}[\mathbb{I}_{E(x_i)}] = \mathbb{E}_{x_i}[\mathbb{E}[\mathbb{I}_{E(x_i)} \mid x_i]] \leq \mathbb{E}_{x_i}[n^{-4}] = n^{-4}. \tag{2.120}$$

Let

$$E = \bigcap_{i=1}^{n} E(x_i)^c. \tag{2.121}$$

Then

$$\begin{aligned}\mathbb{P}(E) &= \mathbb{P}\left(\bigcap_{i=1}^{n} E(x_i)^c\right) = 1 - \mathbb{P}\left(\bigcup_{i=1}^{n} E(x_i)\right) \\ &\geq 1 - \sum_{i=1}^{n} \mathbb{P}(E(x_i)) = 1 - n \cdot n^{-4} = 1 - n^{-3}.\end{aligned} \tag{2.122}$$

In this event E, (2.29) holds for every $x_i \in \mathcal{X}(n)$ or $x \in \mathcal{M}$ as desired. It remains to check that α given by (2.119) is $o(\epsilon^{-1})$, i.e.,

$$\lim_{\substack{\epsilon \to 0 \\ n \to \infty}} \alpha\epsilon = 0. \tag{2.123}$$

This is so due to the condition (2.28). Indeed, by (2.28),

$$\alpha^2\epsilon^2 \propto \frac{\log n}{n\epsilon^{2+d/2}} \to 0. \tag{2.124}$$

Hence $\alpha\epsilon \to 0$ as desired.

The claim of Theorem 2.2.8 readily follows. □

2.3.2.2 Bound for the discrepancy $L^{(n)}_{\epsilon,\mu} f$ and $L_{\epsilon,\mu} f$

Theorem 2.2.9 bounds the discrepancy between the discrete TMDmap generators with discrete and continuous KDEs. Its proof is given below.

Proof. Equation (2.35) implies that

$$\left|\left[L^{(n)}_{\epsilon,\mu}f\right](x) - \left[L_{\epsilon,\mu}f\right](x)\right| = \frac{1}{\epsilon}\left|\left[P^{(n)}_{\epsilon,\mu}f\right](x) - \left[P_{\epsilon,\mu}f\right](x)\right|. \tag{2.125}$$

Therefore, we proceed to calculate the difference between the discrete Markov operators:

$$\begin{aligned}
&\left[P_{\epsilon,\mu}f\right](x) - \left[P^{(n)}_{\epsilon,\mu}f\right](x) = \frac{\sum_{j=1}^n F_j}{\sum_{j=1}^n G_j} - \frac{\sum_{j=1}^n F^{(n)}_j}{\sum_{j=1}^n G^{(n)}_j} = \\
&\frac{\left(\sum_{j=1}^n F_j\right)\left(\sum_{l=1}^n G^{(n)}_l\right) - \left(\sum_{j=1}^n F^{(n)}_j\right)\left(\sum_{l=1}^n G_l\right)}{\left(\sum_{j=1}^n G_j\right)\left(\sum_{l=1}^n G^{(n)}_l\right)} = \\
&\frac{\sum_{j,l}\left(F_j G^{(n)}_l - F_j G_l + F_j G_l - F^{(n)}_j G_l\right)}{\sum_{j,l} G_j G^{(n)}_l} = \\
&\frac{\sum_{j,l}\left(F_j\left(G^{(n)}_l - G_l\right) + G_l\left(F_j - F^{(n)}_j\right)\right)}{\sum_{j,l} G_j G^{(n)}_l}.
\end{aligned} \tag{2.126}$$

The differences $G^{(n)}_l - G_l$ and $F_j - F^{(n)}_j$ in (2.126) can be bounded using Theorem 2.2.8. We start with

$$\left|G^{(n)}_l - G_l\right| = \left|\frac{k_\epsilon(x,x_l)\mu^{1/2}(x_l)}{\rho^{(n)}_\epsilon(x_l)} - \frac{k_\epsilon(x,x_l)\mu^{1/2}(x_l)}{\rho_\epsilon(x_l)}\right| = k_\epsilon(x,x_l)\mu^{1/2}(x_l)\frac{\left|\rho_\epsilon(x_l) - \rho^{(n)}_\epsilon(x_l)\right|}{\rho_\epsilon(x_l)\rho^{(n)}_\epsilon(x_l)}. \tag{2.127}$$

By Theorem 2.2.8, the probability of the event E that the absolute value of the difference in the numerator in (2.127) for any $x_l \in \mathcal{X}(n)$ is less than $5(\pi\epsilon)^{d/2}\rho^{1/2}(x_l)\epsilon^2\alpha$ is greater or equal to $1 - n^{-3}$. In particular, since α given by (2.30) is $o(\epsilon^{-1})$, in the case of event E, this difference is $(\pi\epsilon)^{d/2}o(\epsilon)$. Recalling that

$$\rho_\epsilon(x_l) = (\pi\epsilon)^{d/2}\left(\rho(x_l) + O(\epsilon)\right),$$

the difference (2.127) can be further bounded as follows:

$$\left|G_l^{(n)} - G_l\right| < k_\epsilon(x, x_l)\mu^{1/2}(x_l)\frac{5(\pi\epsilon)^{d/2}\rho^{1/2}(x_l)\epsilon^2\alpha}{(\pi\epsilon)^d\rho^2(x_l)(1+O(\epsilon))} = \frac{5k_\epsilon(x, x_l)\mu^{1/2}(x_l)\epsilon^2\alpha}{(\pi\epsilon)^{d/2}\rho^{3/2}(x_l)}(1+O(\epsilon)). \quad (2.128)$$

Similarly, the difference $F_j - F_j^{(n)}$ is bounded by

$$\left|F_j - F_j^{(n)}\right| < \frac{5k_\epsilon(x, x_l)\mu^{1/2}(x_j)|f(x_j)|\epsilon^2\alpha}{(\pi\epsilon)^{d/2}\rho^{3/2}(x_j)}(1+O(\epsilon)). \quad (2.129)$$

Therefore we have:

$$\left|\left[P_{\epsilon,\mu}f\right](x) - \left[P_{\epsilon,\mu}^{(n)}f\right](x)\right| \le \frac{\sum_{j,l}\left(|F_j|\left|G_l^{(n)} - G_l\right| + G_l\left|F_j - F_j^{(n)}\right|\right)}{\sum_{j,l}G_jG_l^{(n)}} <$$
$$\frac{\sum_{j,l}5\left(\frac{k_\epsilon(x,x_j)\mu^{1/2}(x_j)|f(x_j)|}{(\pi\epsilon)^{d/2}\rho(x_j)}\frac{k_\epsilon(x,x_l)\mu^{1/2}(x_l)\epsilon^2\alpha}{(\pi\epsilon)^{d/2}\rho^{3/2}(x_l)} + \frac{k_\epsilon(x,x_l)\mu^{1/2}(x_l)}{(\pi\epsilon)^{d/2}\rho(x_l)}\frac{k_\epsilon(x,x_j)\mu^{1/2}(x_j)|f(x_j)|\epsilon^2\alpha}{(\pi\epsilon)^{d/2}\rho^{3/2}(x_j)}\right)}{\sum_{j,l}\frac{k_\epsilon(x,x_l)k_\epsilon(x,x_j)\mu^{1/2}(x_l)\mu^{1/2}(x_j)}{(\pi\epsilon)^d\rho(x_l)\rho(x_j)}}(1+O(\epsilon)) =$$
$$\frac{\sum_{j,l}5\frac{k_\epsilon(x,x_j)k_\epsilon(x,x_l)\mu^{1/2}(x_j)\mu^{1/2}(x_l)|f(x_j)|}{\rho(x_j)\rho(x_l)}\left(\rho^{-1/2}(x_l) + \rho^{-1/2}(x_j)\right)}{\sum_{j,l}\frac{k_\epsilon(x,x_l)k_\epsilon(x,x_j)\mu^{1/2}(x_l)\mu^{1/2}(x_j)}{\rho(x_l)\rho(x_j)}}\epsilon^2\alpha\,(1+O(\epsilon)) \quad (2.130)$$

Equations (2.130) and (2.125) imply that

$$\left|\left[L_{\epsilon,\mu}^{(n)}f\right](x) - \left[L_{\epsilon,\mu}f\right](x)\right| < \tilde{C}\epsilon\alpha(1+O(\epsilon)),$$

where the constant $\tilde{C}$ is the prefactor in (2.130). Now it remains to estimate $\tilde{C}$. Approximating the

sums in the numerator and the denominator of $\tilde{C}$ we obtain

$$\begin{aligned}\tilde{C} =& 5\left(\left[\int_{\mathcal{M}} k_\epsilon(x,y)\mu^{1/2}(y)|f(y)|d\text{vol}(y)\right]\left[\int_{\mathcal{M}} k_\epsilon(x,y)\frac{\mu^{1/2}(y)}{\rho^{1/2}(y)}d\text{vol}(y)\right]+\right.\\ &\left.\left[\int_{\mathcal{M}} k_\epsilon(x,y)\frac{\mu^{1/2}(y)}{\rho^{1/2}(y)}|f(y)|d\text{vol}(y)\right]\left[\int_{\mathcal{M}} k_\epsilon(x,y)\mu^{1/2}(y)d\text{vol}(y)\right]\right)/\\ &\left(\left[\int_{\mathcal{M}} k_\epsilon(x,y)\mu^{1/2}(y)d\text{vol}(y)\right]^2\right)+O(n^{-1/2}).\end{aligned} \tag{2.131}$$

Further, the integrals in (2.131) can be approximated using Expansion Lemma 2.2.6 resulting at

$$\begin{aligned}\tilde{C} &= 5\frac{\left(\frac{\mu(x)|f(x)|}{\rho^{1/2}(x)}+O(\epsilon)\right)+\left(\frac{\mu(x)|f(x)|}{\rho^{1/2}(x)}+O(\epsilon)\right)}{\left(\mu^{1/2}(x)+O(\epsilon)\right)^2}+(n^{-1/2})\\ &= 10|f(x)|\rho^{-1/2}(x)+O(\epsilon)+O(n^{-1/2}).\end{aligned} \tag{2.132}$$

At the points where $|f(x)|$ is nonsmooth, bound (2.132) is obtained using any mollification of $|f(x)|$ that is greater or equal than $|f(x)|$. The difference between $|f(x)|$ and its mollification is incorporated in $O(\epsilon)$ (see Appendix C.4 in [Eva10]). Denoting $10|f(x)|\rho^{-1/2}(x)$ by C and noting that $\epsilon \to 0$ and $n \to \infty$ we complete the proof. □

2.3.2.3 Bound for the discrepancy between $L_{\epsilon,\mu}$ and $\mathcal{L}_{\epsilon,\mu}$

Theorem 2.2.10 bounds the discrepancy between the discrete and continuous TMDmap generators with continuous KDE. The proof of Theorem 2.2.10 will make use of several lemmas. The first two lemmas propose random variables Y_j and Z_j that facilitate the proof of Theorem 2.2.10 via the application of Bernstein's inequality (2.25).

Lemma 2.3.3. *Let $a > 0$ be any positive real number and Y_j be defined by*

$$Y_j = F_j m_G - (m_F + a\epsilon m_G)G_j + a\epsilon m_G^2. \tag{2.133}$$

Let $n \to \infty$ and $\epsilon \to 0$ so that the condition (2.41) holds. Then the following are true:

1.

$$\mathbb{E}[Y_j] = 0 \quad \forall 1 \le j \le n,\ j \ne i, \tag{2.134}$$

2.

$$Y_i = \frac{\mu^{1/2}(x_i)}{\rho(x_i)} O(\epsilon^{1-d/2}) \quad \text{and} \quad \frac{\log n}{n\epsilon^3} Y_i = o(1), \tag{2.135}$$

3.

$$\mathbb{P}\left([L_{\epsilon,\mu} f](x_i) - [\mathcal{L}_{\epsilon,\mu} f](x_i) \ge a\right) = \mathbb{P}\left(\sum_{j \ne i} Y_j \ge n\epsilon a m_G^2 - Y_i\right). \tag{2.136}$$

Proof. 1. To prove item 1 of Lemma 2.3.3 we take the expectation of Y_j and readily find that $\mathbb{E}[Y_j] = 0$ for $j \ne i$.

2. Let us calculate Y_i. Noting that $k_\epsilon(x_i, x_i) = 1$ we write

$$Y_i = \frac{\mu^{1/2}(x_i) f(x_i)}{\rho_\epsilon(x_i)} m_G - \frac{\mu^{1/2}(x_i)}{\rho_\epsilon(x_i)}(m_F + a\epsilon m_G) + a\epsilon m_G^2, \tag{2.137}$$

where

$$\rho_\epsilon(x_i) = (\pi\epsilon)^{d/2} \left(\rho(x_i) + O(\epsilon)\right) \tag{2.138}$$

and, according to (2.88), m_F and m_G are

$$m_F = f\mu^{1/2} + \frac{\epsilon}{4}\left[\Delta\left[f\mu^{1/2}\right] - \left[f\mu^{1/2}\right]\left(\frac{\Delta\rho}{\rho} - \omega\right)\right] + O(\epsilon^2) \quad \text{and} \tag{2.139}$$

$$m_G = \mu^{1/2} + \frac{\epsilon}{4}\left[\Delta\mu^{1/2} - \mu^{1/2}\left(\frac{\Delta\rho}{\rho} - \omega\right)\right] + O(\epsilon^2). \tag{2.140}$$

All functions in (2.139) and (2.140) are evaluated at x_i. Plugging (2.139) and (2.140) into (2.137) and taking into account that f and its derivatives are $O(1)$ we obtain

$$\begin{aligned} Y_i &= \frac{\mu^{1/2} f}{\rho_\epsilon}\left(\mu^{1/2} + O(\epsilon)\right) - \frac{\mu^{1/2}}{\rho_\epsilon}\left(f\mu^{1/2} + O(\epsilon)\right) + O(\epsilon) \\ &= \frac{\mu^{1/2}}{\rho_\epsilon} O(\epsilon) + O(\epsilon) = \frac{\mu^{1/2}}{\rho} O(\epsilon^{1-d/2}). \end{aligned} \tag{2.141}$$

Since $n \to \infty$ and $\epsilon \to 0$ in such a manner that $n\epsilon^{2+d/2} \to \infty$ by assumption, we conclude that $n^{-1}\epsilon^{-2-d/2}\log^{-1} n \to 0$. This means that $n^{-1}\epsilon^{-3}(\log n)Y_i = o(1)$ as desired.

3. We will use identity (2.40) to prove item 3. First, we will show how the random variable Y

comes about. We start with several rearrangements:

$$\begin{aligned}
&\mathbb{P}\left(\left[L_{\epsilon,\mu}f\right](x_i) - \left[\mathcal{L}_{\epsilon,\mu}f\right](x_i) \geq a\right) = \mathbb{P}\left(\frac{\sum_{j=1}^n F_j}{\sum_{j=1}^n G_j} - \frac{m_F}{m_G} > \epsilon a\right) \\
&= \mathbb{P}\left(\frac{\sum_{j=1}^n \left(F(x_j)m_G - m_F G(x_j)\right)}{\sum_{j=1}^n G(x_j)m_G} \geq a\epsilon\right) \\
&= \mathbb{P}\left(\sum_{j=1}^n \left(F(x_j)m_G - (m_F + a\epsilon m_G)G(x_j)\right) \geq 0\right) \\
&= \mathbb{P}\left(\sum_{j=1}^n \left(F(x_j)m_G - (m_F + a\epsilon m_G)G(x_j) - a\epsilon m_G^2 + a\epsilon m_G^2\right) \geq 0\right) \\
&= \mathbb{P}\left(\sum_{j=1}^n \left(F(x_j)m_G - (m_F + a\epsilon m_G)G(x_j) + a\epsilon m_G^2\right) \geq na\epsilon m_G^2\right). \qquad (2.142)
\end{aligned}$$

The terms in the left-hand side of inequality (2.142) are Y_j, $1 \leq j \leq n$. All of them are random variables given x_i except for Y_i. We transfer it to the right-hand side and take item 2 into account:

$$\begin{aligned}
&\mathbb{P}\left(\left[L_{\epsilon,\mu}f\right](x_i) - \left[\mathcal{L}_{\epsilon,\mu}f\right](x_i) \geq a\right) \\
&= \mathbb{P}\left(\sum_{j\neq i} \left(F(x_j)m_G - (m_F + a\epsilon m_G)G(x_j) + a\epsilon m_G^2\right) \geq na\epsilon m_G^2 - Y_i\right). \qquad (2.143)
\end{aligned}$$

□

Lemma 2.3.4. *Let $a > 0$ be any positive real number and Z_j be defined by*

$$Z_j = F_j m_G - (m_F - a\epsilon m_G)G_j - a\epsilon m_G^2. \qquad (2.144)$$

Let $n \to \infty$ and $\epsilon \to 0$ so that the condition (2.41) holds. Then the following are true:

1.

$$\mathbb{E}[Z_j] = 0 \quad \forall 1 \leq j \leq n,\ j \neq i; \tag{2.145}$$

2.

$$Z_i = \frac{\mu^{1/2}(x_i)}{\rho(x_i)} O(\epsilon^{1-d/2}) \quad \text{and} \quad \frac{\log n}{n\epsilon^3} Z_i = o(1); \tag{2.146}$$

3.

$$\mathbb{P}\left([L_{\epsilon,\mu} f](x_i) - [\mathcal{L}_{\epsilon,\mu} f](x_i) \leq -a\right) = \mathbb{P}\left(\sum_{j \neq i} Z_j \leq -n\epsilon a m_G^2 - Z_i\right). \tag{2.147}$$

Proof. The first two items of Lemma 2.3.4 are proven similar to those of Lemma 2.3.3. Let us work out the proof of item 3:

$$\begin{aligned}
&\mathbb{P}\left([L_{\epsilon,\mu} f](x_i) - [\mathcal{L}_{\epsilon,\mu} f](x_i) \leq -a\right) = \mathbb{P}\left(\frac{\sum_{j=1}^n F_j}{\sum_{j=1}^n G_j} - \frac{m_F}{m_G} \leq -\epsilon a\right) \\
=&\mathbb{P}\left(\frac{\sum_{j=1}^n \left(F(x_j) m_G - m_F G(x_j)\right)}{\sum_{j=1}^n G(x_j) m_G} \leq -a\epsilon\right) \\
=&\mathbb{P}\left(\sum_{j=1}^n \left(F(x_j) m_G - (m_F - a\epsilon m_G) G(x_j)\right) \leq 0\right) \\
=&\mathbb{P}\left(\sum_{j=1}^n \left(F(x_j) m_G - (m_F - a\epsilon m_G) G(x_j) - a\epsilon m_G^2\right) \leq -na\epsilon m_G^2\right).
\end{aligned} \tag{2.148}$$

The terms in the left-hand side of inequality (2.148) are Z_j, $1 \leq j \leq n$. All of them are random variables given x_i except for Z_i. As in the proof of Lemma 2.3.3, we transfer it to the right-hand side,

take item 2 into account, and obtain the desired result:

$$\begin{aligned}&\mathbb{P}\left([L_{\epsilon,\mu}f](x_i) - [\mathcal{L}_{\epsilon,\mu}f](x_i) \leq -a\right)\\ =&\mathbb{P}\left(\sum_{j\neq i}\left(F(x_j)m_G - (m_F - a\epsilon m_G)G(x_j) - a\epsilon m_G^2\right) \leq -na\epsilon m_G^2 - Z_i\right).\end{aligned} \tag{2.149}$$

□

The next two lemmas give estimates of the variance of random variables Y and Z respectively.

Lemma 2.3.5. *Let $x \in \mathcal{M}$ be any selected point on the manifold $\mathcal{M}$. Let $y \in \mathcal{M}$ be any point sampled according to the density ρ. Let Y and Z be defined, respectively, by (2.133) and (2.144) where x_j is replaced with y. Then*

$$\mathsf{Var}(Y) = \mathsf{Var}(Z) = \frac{\epsilon}{4}(2\pi\epsilon)^{-d/2}\left(\frac{\mu^2}{\rho}\|\nabla f\|^2 + O(\epsilon)\right). \tag{2.150}$$

Proof. We will calculate the variance of Y. The variance of Z is calculated likewise.

Let F and G be defined by (2.31) and (2.32) respectively where x_j is replaced with y:

$$F := \frac{k_\epsilon(x,y)\mu^{1/2}(y)f(y)}{\rho_\epsilon(y)} \quad \text{and} \quad G := \frac{k_\epsilon(x,y)\mu^{1/2}(y)}{\rho_\epsilon(y)}. \tag{2.151}$$

First, we find Y^2:

$$\begin{aligned}Y^2 &= \left[m_G F - (m_F + a\epsilon m_G)G + a\epsilon m_G^2\right]^2\\ &= m_G^2F^2 + (m_F^2 + 2a\epsilon m_G m_F + (a\epsilon)^2 m_G^2)G^2 + (a\epsilon)^2 m_G^4\\ &\quad - 2m_G(m_F + a\epsilon m_G)GF - 2a\epsilon(m_F + a\epsilon m_G)m_G^2 G + 2a\epsilon m_G^3 F.\end{aligned}$$

Then, taking into account that $\mathbb{E}[Y] = 0$, we get

$$\begin{aligned}
\mathsf{Var}(Y) = \mathbb{E}[Y^2] &= m_G^2\mathbb{E}[F^2] + (m_F^2 + 2a\epsilon m_G m_F + (a\epsilon)^2 m_G^2)\mathbb{E}[G^2] + (a\epsilon)^2 m_G^4 \\
&- 2m_G(m_F + (a\epsilon)m_G)\mathbb{E}[GF] - 2a\epsilon(m_F + a\epsilon m_G)m_G^3 + 2a\epsilon m_G^3 m_F \\
&= m_G^2\mathbb{E}[F^2] - 2m_G m_F\mathbb{E}[GF] + m_F^2\mathbb{E}[G^2] \\
&+ 2a\epsilon m_G\left[m_F\mathbb{E}[G^2] - m_G\mathbb{E}[GF]\right] \\
&+ (a\epsilon)^2 m_G^2\left[\mathbb{E}[G^2] - m_G^2\right]. \qquad (2.152)
\end{aligned}$$

Hence, to compute $\mathsf{Var}(Y)$, we need to calculate expectations of F^2, G^2, and FG. As in the proof of Lemma 2.3.1, we will use the fact that $k_\epsilon^2(x,y) = k_{\epsilon/2}(x,y)$. We also will need the expansion of ρ_ϵ^{-2}:

$$\rho_\epsilon^{-2}(x) := (\pi\epsilon)^{-d}\rho^{-2}(x)\left[1 - \frac{\epsilon}{2}\left(\frac{\Delta\rho(x)}{\rho(x)} - \omega(x)\right) + O(\epsilon^2)\right]. \qquad (2.153)$$

We start with $\mathbb{E}[G^2]$:

$$\begin{aligned}
\mathbb{E}[G^2] &= \int_{\mathcal{M}} k_{\epsilon/2}(x,y)\frac{\mu(y)}{(\pi\epsilon)^d\rho^2(y)}\left[1 - \frac{\epsilon}{2}\left(\frac{\Delta\rho(y)}{\rho(y)} - \omega(y)\right) + O(\epsilon^2)\right]\rho(y)dy \\
&= (\pi\epsilon)^{-d}\int_{\mathcal{M}} k_{\epsilon/2}(x,y)\frac{\mu(y)}{\rho(y)}\left[1 - \frac{\epsilon}{2}\left(\frac{\Delta\rho(y)}{\rho(y)} - \omega(y)\right) + O(\epsilon^2)\right]dy \\
&= (\pi\epsilon)^{-d}\left(\frac{\pi\epsilon}{2}\right)^{d/2}\left[\frac{\mu}{\rho} + \frac{\epsilon}{8}\left(\Delta\left[\frac{\mu}{\rho}\right] - \omega\frac{\mu}{\rho}\right) - \frac{\epsilon}{2}\frac{\mu}{\rho}\left(\frac{\Delta\rho}{\rho} - \omega\right) + O(\epsilon^2)\right] \\
&= (2\pi\epsilon)^{-d/2}\left[\frac{\mu}{\rho} + \frac{\epsilon}{8}\left(\Delta\left[\frac{\mu}{\rho}\right] - 4\frac{\mu}{\rho}\frac{\Delta\rho}{\rho} + 3\omega\frac{\mu}{\rho}\right) + O(\epsilon^2)\right]. \qquad (2.154)
\end{aligned}$$

The expectations of F^2 and FG are computed likewise. The expressions for these expectations can be

written compactly as

$$\mathbb{E}[G^2] = (2\pi\epsilon)^{-d/2}\left[\frac{\mu}{\rho} + \frac{\epsilon}{8}\mathcal{S}\left(\frac{\mu}{\rho}, \rho\right) + O(\epsilon^2)\right], \tag{2.155}$$

$$\mathbb{E}[F^2] = (2\pi\epsilon)^{-d/2}\left[\frac{f^2\mu}{\rho} + \frac{\epsilon}{8}\mathcal{S}\left(\frac{f^2\mu}{\rho}, \rho\right) + O(\epsilon^2)\right], \tag{2.156}$$

$$\mathbb{E}[FG] = (2\pi\epsilon)^{-d/2}\left[\frac{f\mu}{\rho} + \frac{\epsilon}{8}\mathcal{S}\left(\frac{f\mu}{\rho}, \rho\right) + O(\epsilon^2)\right], \tag{2.157}$$

where the operator $\mathcal{S}(\cdot,\cdot)$ is defined by

$$\mathcal{S}(u, v) := \Delta u - 4u\frac{\Delta v}{v} + 3\omega u. \tag{2.158}$$

Further, $\mathsf{Var}(Y^2)$ given by (2.150) contains m_F and m_G. The expressions for m_F and m_G given by (2.139) and (2.140) respectively can be compactified via the introduction of the operator $\mathcal{P}(\cdot,\cdot)$ defined by

$$\mathcal{P}(u, v) := \Delta u - u\frac{\Delta v}{v} + \omega u. \tag{2.159}$$

Then

$$m_F = f\mu^{1/2} + \frac{\epsilon}{4}\mathcal{P}(f\mu^{1/2}, \rho) + O(\epsilon^2), \tag{2.160}$$

$$m_G = \mu^{1/2} + \frac{\epsilon}{4}\mathcal{P}(\mu^{1/2}, \rho) + O(\epsilon^2). \tag{2.161}$$

Plugging (2.156), (2.155), (2.157), (2.160), and (2.161) into (2.150), doing some algebra, and tracking

only the highest-order terms in ϵ we obtain

$$\mathsf{Var}(Y) = \frac{\epsilon}{8}(2\pi\epsilon)^{-d/2}\left\{\mu\mathcal{S}\left(\frac{f^2\mu}{\rho},\rho\right) - 2f\mu\mathcal{S}\left(\frac{f\mu}{\rho},\rho\right) + f^2\mu\mathcal{S}\left(\frac{\mu}{\rho},\rho\right) + O(\epsilon)\right\}. \tag{2.162}$$

Remarkably, all operators $\mathcal{P}$ cancel out. Next, we decode the operator $\mathcal{S}$ in (2.162) using (2.158) and get the desired result:

$$\begin{aligned}
\mathsf{Var}(Y) &= \frac{\epsilon}{8}(2\pi\epsilon)^{-d/2}\left\{\mu\Delta\left(\frac{f^2\mu}{\rho}\right) - 2f\mu\Delta\left(\frac{f\mu}{\rho}\right) + f^2\mu\Delta\left(\frac{\mu}{\rho}\right) + O(\epsilon)\right\} \\
&= \frac{\epsilon}{8}(2\pi\epsilon)^{-d/2}\left\{\mu(\Delta f^2)\frac{\mu}{\rho} + 2\mu\nabla f^2\cdot\nabla\frac{\mu}{\rho} + \mu f^2\Delta\frac{\mu}{\rho}\right. \\
&\left.-2f\mu(\Delta f)\frac{\mu}{\rho} - 4f\mu\nabla f\cdot\nabla\frac{\mu}{\rho} - 2\mu f^2\Delta\frac{\mu}{\rho} + f^2\mu\Delta\frac{\mu}{\rho} + O(\epsilon)\right\} \\
&= \frac{\epsilon}{8}(2\pi\epsilon)^{-d/2}\left\{\frac{\mu^2}{\rho}\left(2\|\nabla f\|^2 + 2f\Delta f\right) + 4\mu f\nabla f\cdot\nabla\frac{\mu}{\rho}\right. \\
&\left.-2f\mu(\Delta f)\frac{\mu}{\rho} - 4f\mu\nabla f\cdot\nabla\frac{\mu}{\rho} + O(\epsilon)\right\} \\
&= \frac{\epsilon}{4}(2\pi\epsilon)^{-d/2}\left(\frac{\mu^2}{\rho}\|\nabla f\|^2 + O(\epsilon)\right).
\end{aligned} \tag{2.163}$$

□

Now we proceed to prove Theorem 2.2.10.

Proof. By assumption, ρ and f are bounded. Therefore, the random variables Y and Z are bounded as well. After conditioning on the point x_i we drop into the regime of Lemmas 2.3.3, 2.3.4, and 2.3.5. Let $|Y_j| \le M_Y$ and $|Z_j| \le M_Z$. Then, according to Bernstein's inequality (2.25) we have:

$$\mathbb{P}_{x_i}\left(\sum_{j\neq i} Y_j \ge na\epsilon m_G^2 - Y_i\right) \le \exp\left(-\frac{(na\epsilon m_G^2 - Y_i)^2}{2(n-1)\mathsf{Var}(Y) + 2M_Y(na\epsilon m_G^2 - Y_i)}\right). \tag{2.164}$$

Let us simplify the argument of the exponent. According to 2.3.3, $Y_i = O(\epsilon^{1-d/2})$. Therefore, the expression $na\epsilon m_G^2 - Y_i$ can be estimated as

$$na\epsilon m_G^2 - Y_i = na\epsilon m_G^2 \left(1 - \frac{O(\epsilon^{1-d/2})}{na\epsilon m_G^2}\right) = na\epsilon m_G^2 \left(1 + o(\epsilon)\right) \tag{2.165}$$

provided that $na\epsilon^{d/2} \to \infty$ as $\epsilon \to 0$ and $n \to \infty$. We will verify this condition later. Plugging (2.150) and (2.165) into (2.164) and canceling n we obtain

$$\begin{aligned}
&\mathbb{P}_{x_i}\left(\sum_{j\neq i} Y_j \geq na\epsilon m_G^2 - Y_i\right) \leq \\
&\exp\left(-\frac{na^2\epsilon^2 m_G^4 \left(1 + o(\epsilon)\right)}{2\frac{n-1}{n}\frac{\epsilon}{4}(2\pi\epsilon)^{-d/2}\left(\frac{\mu^2}{\rho}\|\nabla f\|^2 + O(\epsilon)\right) + 2M_Y a\epsilon m_G^2 \left(1 + o(\epsilon)\right)}\right) = \\
&\exp\left(-\frac{2(2\pi)^{d/2} na^2\epsilon^{1+d/2} m_G^4 \left(1 + o(\epsilon)\right)}{\frac{\mu^2}{\rho}\|\nabla f\|^2 + O(\epsilon) + o(1)}\right) = \\
&\exp\left(-\frac{2(2\pi)^{d/2} na^2\epsilon^{1+d/2} m_G^4}{\frac{\mu^2}{\rho}\|\nabla f\|^2}\left(1 + o(1)\right)\right).
\end{aligned} \tag{2.166}$$

Similarly, the following bound for the sum of Z_js is obtained:

$$\mathbb{P}_{x_i}\left(\sum_{j\neq i} Z_j \leq -na\epsilon m_G^2 - Z_i\right) \leq \exp\left(-\frac{2(2\pi)^{d/2} na^2\epsilon^{1+d/2} m_G^4}{\frac{\mu^2}{\rho}\|\nabla f\|^2}\left(1 + o(1)\right)\right). \tag{2.167}$$

Lemmas 2.3.3 and 2.3.4 and inequalities (2.166) and (2.167) imply that

$$\mathbb{P}_{x_i}\left(\left|\left[L_{\epsilon,\mu} f\right](x_i) - \left[\mathcal{L}_{\epsilon,\mu} f\right](x_i)\right| \geq a\right) \leq 2\exp\left(-\frac{2(2\pi)^{d/2} na^2\epsilon^{1+d/2} m_G^4}{\frac{\mu^2}{\rho}\|\nabla f\|^2}\left(1 + o(1)\right)\right). \tag{2.168}$$

The next step is to find α such that the right-hand side of inequality (2.168) is less or equal that n^{-3} as

it was done in the proofs of Theorems 2.2.8 and 2.2.9. Hence, we set

$$2\exp\left(-\frac{2(2\pi)^{d/2}na^2\epsilon^{1+d/2}m_G^4}{\frac{\mu^2}{\rho}\|\nabla f\|^2}\,(1+o(1))\right)\leq n^{-3}. \tag{2.169}$$

Taking logarithms of both sides, recalling that $m_G = \mu^{1/2}(x_i) + O(\epsilon)$, and doing some algebra we obtain

$$a \geq \frac{\|\nabla f\|}{\rho^{1/2}(x_i)(2\pi)^{d/4}}\sqrt{\frac{(3\log n+\log 2)}{2n\epsilon^{1+d/2}}}(1+o(1)). \tag{2.170}$$

Hence, for large enough n and small enough ϵ we can choose

$$a = \frac{2\|\nabla f\|}{\rho^{1/2}(x_i)(2\pi)^{d/4}}\sqrt{\frac{\log n}{n\epsilon^{1+d/2}}}. \tag{2.171}$$

Let us check that the term in (2.165) is indeed $o(\epsilon)$. To do so, we check that $\epsilon na\epsilon^{d/2} \to \infty$ as $\epsilon \to 0$ and $n \to \infty$. Indeed, with the use of by assumption (2.41) we get

$$\epsilon na\epsilon^{d/2} \propto \sqrt{\frac{n^2\epsilon^{2+d}\log n}{n\epsilon^{1+d/2}}} = \sqrt{n\epsilon^{1+d/2}\log n} \to \infty. \tag{2.172}$$

Finally, we observe that a given by (2.171) can be expressed as

$$a = \frac{2\|\nabla f\|}{\rho^{1/2}(x_i)}\epsilon^{3/2}\alpha, \tag{2.173}$$

where α is given by (2.43). This gives the desired result and completes the proof. □

2.3.3 Numerical solution of BVPs: proofs

Theorem 2.2.15 establishes an error bound for the numerical solution by the TMDmap algorithm to the general boundary-value problem (BVP) of the form (2.8). The proof presented below utilizes a *maximum principle*-based argument, specifically, the *method of comparison functions* commonly used in stability estimates for finite difference methods [MM05a]. A closely related argument was proposed in [JH23, p. 35] for *Ghost Point Diffusion Maps*.

Under Assumptions 1-5, we consider the BVP (2.8). For cleaner notation, we set

$$L := L_{\epsilon,\mu}^{(n)}, \tag{2.174}$$

$$\mathcal{I} := \{i \mid x_i \in \mathcal{M} \setminus \Omega\}, \tag{2.175}$$

$$\mathcal{D} := \{i \mid x_i \in \Omega\}. \tag{2.176}$$

To prove Theorem 2.2.15 we will need several lemmas.

Lemma 2.3.6 (Discrete maximum principle)**.** *Let $\mathcal{D} \neq \emptyset$ and $v = [v^{\mathcal{I}}\ v^{\mathcal{D}}]^\top \in \mathbb{R}^n$.*

1. *If $[Lv]_i \geq 0$ for all $i \in \mathcal{I}$ then v_i reaches its maximum on $\mathcal{D}$ i.e.,*

$$\max_{i \in \mathcal{I} \cup \mathcal{D}} v_i = \max_{i \in \mathcal{D}} v_i. \tag{2.177}$$

2. *If $[Lv]_i \leq 0$ for all $i \in \mathcal{I}$ then v_i reaches its minimum on $\mathcal{D}$, i.e.,*

$$\min_{i \in \mathcal{I} \cup \mathcal{D}} v_i = \min_{i \in \mathcal{D}} v_i. \tag{2.178}$$

Proof. We will prove item 1. The proof of item 2 readily follows by redefining v as $-v$.

By construction, $L = \epsilon^{-1}(P - I)$ where P is a Markov matrix. All entries of any Markov matrix are nonnegative and row sums are 1. In our case, $P = P^{(n)}_{\epsilon,\mu}$. Its off-diagonal entries are positive. Hence, all off-diagonal entries of L are positive and row sums of L are zero. This means that $\sum_{j=1}^{n} L_{ij} = 0$ for all $i \in \mathcal{I} \cup \mathcal{D}$, and diagonal entries of L are negative. To construct a contradiction, we assume that there is $i^* \in \mathcal{I}$ such that

$$v_{i^*} = \max_{i \in \mathcal{I} \cup \mathcal{D}} v_i > \max_{i \in \mathcal{D}} v_i.$$

Since $i^* \in \mathcal{I}$, all entries $L_{i^* j} > 0$ for $j \in \mathcal{D}$ and for all $j \in \mathcal{I} \backslash \{i^*\}$. Therefore,

$$0 \leq \sum_{j \in \mathcal{I} \cup \mathcal{D}} L_{i^* j} v_j = L_{i^* i^*} v_{i^*} + \sum_{j \neq i^*} L_{i^* j} v_j < v_{i^*} \sum_{j \in \mathcal{I} \cup \mathcal{D}} L_{i^* j} = 0. \tag{2.179}$$

Hence, the maximum of v must be reached of $\mathcal{D}$. This implies (2.177). □

Remark 2.3.7. *In practice, the matrix L is often sparsified for computational efficiency. In this case, the discrete maximum principle in Lemma 2.3.6 still holds. Its proof requires a slightly longer argument involving the irreducibility of L as in the proof of Lemma 6.1 in [MM05a].*

Lemma 2.3.8 (The maximum principle)**.** *1. Let $\mathcal{L}\phi \geq 0$ in $\mathcal{M} \backslash \Omega$ where $\mathcal{L}$ is the generator given by (1.52). Then the maximum of ϕ is reached on the boundary $\partial\Omega$, i.e.,*

$$\max_{x \in \mathcal{M} \backslash \Omega} \phi = \max_{\partial\Omega} \phi. \tag{2.180}$$

2. Let $\mathcal{L}\phi \leq 0$ in $\mathcal{M} \backslash \Omega$ where $\mathcal{L}$ is the generator given by (1.52). Then the minimum of ϕ is reached

on the boundary $\partial\Omega$, i.e.,

$$\min_{x\in\mathcal{M}\setminus\Omega}\phi=\min_{\partial\Omega}\phi. \tag{2.181}$$

This is a standard result for elliptic operators (see Theorem 1 in Section 6.4.1 in [Eva10]).

Lemma 2.3.9. *Let $\epsilon\to 0$ and $n\to\infty$ so that the condition (2.10) holds. Then the probability that for every $x_i\in\mathcal{X}(n)$*

$$\begin{aligned}|4\beta^{-1}\left[L^{(n)}_{\epsilon,\mu}u\right](x_i)-\mathcal{L}u(x_i)|\leq\frac{\alpha\epsilon}{\rho^{1/2}(x_i)}\left(2\|\nabla u(x_i)\|\epsilon^{1/2}+11|u(x_i)|\right)\\ +\epsilon\left|\mathcal{B}_1[u,\mu]+\mathcal{B}_2[u,\mu,\rho]+\mathcal{B}_3[u,\mu,\rho]\right|+O(\epsilon^2)\end{aligned} \tag{2.182}$$

is greater or equal to $1-2n^{-2}$.

Proof. Let $E(x_i)$ be the event that where inequality (2.182) holds for x_i. According to Theorem 2.1.1, $\mathbb{P}(E(x_i))>1-2n^{-3}$. Hence

$$\begin{aligned}\mathbb{P}\left(\bigcap_{i=1}^n E(x_i)\right)&=1-\mathbb{P}\left(\bigcup_{i=1}^n E(x_i)^c\right)\geq 1-\sum_{i=1}^n\mathbb{P}(E(x_i)^c)\\ &\geq 1-\sum_{i=1}^n 2n^{-3}=1-2n^{-2}.\end{aligned} \tag{2.183}$$

□

Lemma 2.3.10. *Let $\mathcal{X}(n)$ be a point cloud on $\mathcal{M}$ of n points sampled i.i.d. with density ρ. The probability of $\mathcal{D}$ being non-empty tends to 1 exponentially:*

$$\mathbb{P}(\mathcal{D}\neq\emptyset)\geq 1-\exp\left(-n\mathbb{E}_\rho[\mathbb{I}_\Omega]\right). \tag{2.184}$$

Proof. Obviously,

$$\mathbb{P}(\mathcal{D} \neq \emptyset) = 1 - \mathbb{P}(\mathcal{D} = \emptyset) = 1 - \mathcal{P}\left(\bigcap_i x_i \notin \Omega\right).$$

Since x_i, $1 \leq i \leq n$, are i.i.d. samples, the probability that none of x_i lies in Ω is the product of probabilities that each x_i does not lie in Ω, i.e.,

$$\mathcal{P}\left(\bigcap_i x_i \notin \Omega\right) = \prod_i \mathcal{P}(x_i \notin \Omega).$$

In turn, $\mathcal{P}(x_i \notin \Omega) = 1 - \mathbb{E}_\rho[\mathbb{I}_\Omega]$. Therefore,

$$\mathbb{P}(\mathcal{D} \neq \emptyset) = 1 - \prod_i (1 - \mathbb{E}_\rho[\mathbb{I}_\Omega]) = 1 - (1 - \mathbb{E}_\rho[\mathbb{I}_\Omega])^n \geq 1 - \exp(-n\mathbb{E}_\rho[\mathbb{I}_\Omega]).$$

The last inequality follows from the fact that $1 - x \leq e^{-x}$. □

Now we are ready to prove Theorem 2.2.15.

Proof. Let u be the exact solution to (2.8). Let v be the numerical solution, i.e., v satisfies (2.9). For brevity, we will denote $L^{(n)}_{\epsilon,\mu}$ by L. This argument uses the approach from [MM05b, p.195-196]. We have

$$\begin{cases} \mathcal{L}u = f, & x \in \mathcal{M}\backslash\Omega \\ u = g, & x \in \Omega \end{cases} \quad \text{and} \quad \begin{cases} [4\beta^{-1}Lv]_i = f_i, & x_i \in \mathcal{M}\backslash\Omega \\ v(x_i) = g(x_i), & x_i \in \Omega \end{cases}. \tag{2.185}$$

By Lemmas 2.3.9 and 2.3.10, with high probability greater or equal to $1 - 2n^{-2} - \exp(-n\mathbb{E}[I_\Omega])$, for all $x_i \in \mathcal{X}(n)$, all n large enough and ϵ small enough, there is a positive function $C_u(x)$ that depends

only on x, u, μ, and ρ but not on ϵ and n such that

$$|4\beta^{-1}[Lu](x_i) - \mathcal{L}u(x_i)| \leq C_u(x_i)\epsilon. \tag{2.186}$$

For brevity, we will use the subscript i to denote the value at x_i. Inequality (2.186) implies that

$$f_i - \epsilon C_{u,i} = [\mathcal{L}u]_i - \epsilon C_{u,i} \leq [4\beta^{-1}Lu]_i \leq [\mathcal{L}u]_i + \epsilon C_{u,i} = f_i + \epsilon C_{u,i}. \tag{2.187}$$

To construct comparison functions to which we will apply the discrete maximum principle, we will need an auxiliary function ϕ. It is convenient to choose ϕ to be the solution to the boundary-value problem

$$\begin{cases} \mathcal{L}\phi = 1, & x \in \mathcal{M}\backslash\Omega \\ \\ \phi = 0, & x \in \partial\Omega \end{cases}. \tag{2.188}$$

According to the maximum principle, precisely item 1 of Lemma 2.3.8, the maximum of ϕ is reached on the boundary. Since ϕ is zero on the boundary, $\phi(x) \leq 0$ for $x \in \mathcal{M}\backslash\Omega$. Moreover, according to Lemmas 2.3.9 and 2.3.10, for ϵ small enough and n large enough, there exists a positive function $C_\phi(x)$ that depends on x, u, μ, and ρ but not on ϵ and n such that

$$1 - \epsilon C_{\phi,i} = [\mathcal{L}\phi]_i - \epsilon C_{\phi,i} \leq [4\beta^{-1}L\phi]_i \leq [\mathcal{L}\phi]_i + \epsilon C_{\phi,i} = 1 + \epsilon C_{\phi,i}. \tag{2.189}$$

with probability greater of equal to $1 - 2n^{-2} - \exp\left(-n\mathbb{E}[I_\Omega]\right)$.

Now we define the first comparison function

$$\psi = u - v - \epsilon \left(\max_{1 \le i \le n} C_{u,i} + 1 \right) \phi \equiv u - v - \epsilon (C_{\max} + 1) \phi. \tag{2.190}$$

Equations (2.187) and (2.189) imply that, with high probability, $[L\psi]_i \le 0$ for all $i \in \mathcal{I}$ if ϵ is small enough. Indeed, with probability greater or equal to $1 - 4n^{-2} - \exp(-n\mathbb{E}[I_\Omega])$,

$$\begin{aligned}
[4\beta^{-1} L\psi]_i &= [4\beta^{-1} Lu]_i - [4\beta^{-1} Lv]_i - \epsilon (C_{\max} + 1)[4\beta^{-1} L\phi]_i \\
&\le f_i + \epsilon C_{u,i} - f_i - \epsilon (C_{\max} + 1)(1 - C_{\phi,i}\epsilon) \\
&\le -\epsilon \left(1 - C_{\phi,i}\epsilon (1 + C_{\max})\right) \le 0.
\end{aligned}$$

We did not multiply the exponent by 2 in the probability estimate $1 - 4n^{-2} - \exp(-n\mathbb{E}[I_\Omega])$ because the same set of points is used in the estimates for u and ϕ.

Hence, by the discrete maximum principle in Lemma 2.3.6, the minimum of ψ is achieved on $\mathcal{D}$. Since $u_i = v_i = g_i$ for all $i \in \mathcal{D}$ and $\phi_i = 0$ for $i \in \mathcal{D}$, it follows that $\psi_i = 0$ for all $i \in \mathcal{D}$. Therefore, $\psi_i \ge 0$ for all $1 \le i \le n$, i.e.,

$$u_i - v_i \ge \epsilon (C_{\max} + 1)\phi_i \equiv -\epsilon (C_{\max} + 1)|\phi_i| \quad \forall 1 \le i \le n. \tag{2.191}$$

Next, we define the second comparison function

$$\psi = u - v + \epsilon \left(\max_{1 \le i \le n} C_{u,i} + 1 \right) \phi \equiv u - v + \epsilon (C_{\max} + 1) \phi \tag{2.192}$$

and argue that $[L\psi]_i \ge 0$ for all $i \in \mathcal{I}$ with high probability. Indeed, (2.187) and (2.189) imply that

with probability greater or equal to $1 - 4n^{-2} - \exp(-n\mathbb{E}[I_\Omega])$,

$$\begin{aligned}
[4\beta^{-1}L\psi]_i &= [4\beta^{-1}Lu]_i - [4\beta^{-1}Lv]_i + \epsilon(C_{\max}+1)[4\beta^{-1}L\phi]_i \\
&\geq f_i - \epsilon C_{u,i} - f_i + \epsilon(C_{\max}+1)(1 - C_{\phi,i}\epsilon) \\
&\geq \epsilon\left(1 - C_{\phi,i}\epsilon(1 + C_{\max})\right) \geq 0
\end{aligned}$$

for ϵ small enough. By Lemma 2.3.6, the maximum of ψ_i is achieved on $\mathcal{D}$. Since $u_i = v_i = g_i$ and $\phi_i = 0$ for $i \in \mathcal{D}$, $\psi_i \leq 0$ for all $i \in \mathcal{D}$. Hence $\phi_i \leq 0$ for all $1 \leq i \leq n$, i.e.,

$$u_i - v_i \leq -\epsilon(C_{\max}+1)\phi_i \equiv \epsilon(C_{\max}+1)|\phi_i| \quad \forall 1 \leq i \leq n. \tag{2.193}$$

Inequalities (2.191) and (2.193) imply that, for ϵ small enough, with probability greater or equal to $1 - 4n^{-2} - \exp(-n\mathbb{E}[I_\Omega])$,

$$|u(x_i) - v(x_i)| \leq \epsilon(C_{\max}+1)|\phi(x_i)| \quad \forall 1 \leq i \leq n \tag{2.194}$$

where ϕ is the solution to (2.188). Finally, we note that

$$C_{\max} = \max_{1 \leq i \leq n} C_{u,i} \leq \max_{x \in \mathcal{M}} C_{u,\mu,\rho}(x)$$

where $C_{u,\mu,\rho}(x)$ is defined be (2.70) in the statement of Theorem 2.2.15. □

2.4 Test problems

2.4.1 Illustrating error model

In Corollary 2.2.7 it was shown that bias error prefactors $\mathcal{B}_2$ and $\mathcal{B}_3$ can be eliminated when ρ is uniform on $\mathcal{M}$ and when the test function f is the committor. One can expect that these cancellations reduce the magnitude of the overall bias error prefactor $\mathcal{B}_1 + \mathcal{B}_2 + \mathcal{B}_3$, though this might not be necessary. The example presented in this section demonstrates that these cancellations indeed reduce the magnitude of the prefactor, thus showing that using uniform densities and approximating the committor function is a viable strategy for improving the accuracy of TMDmap.

2.4.1.1 Theoretical considerations

The leading order terms in the error model in Theorem 2.1.1 can be briefly stated as follows:

$$|4\beta^{-1}L^{(n)}_{\epsilon,\mu}f(x) - \mathcal{L}f(x)| \leq \sqrt{\frac{\log n}{n\epsilon^{2+d/2}}}|\mathcal{V}[f,\mu,\rho](x)| + \epsilon|\mathcal{B}[f,\mu,\rho](x)|. \tag{2.195}$$

Here $\mathcal{V}$ and $\mathcal{B}$ are variance and bias prefactors respectively given by

$$\mathcal{V} := \frac{1}{\rho(x)^{1/2}}\left(2\|\nabla f(x)\|\epsilon^{1/2} + 11|f(x)|\right), \tag{2.196}$$

$$\mathcal{B} := \mathcal{B}_1 + \mathcal{B}_2 + \mathcal{B}_3. \tag{2.197}$$

$\mathcal{V}$ and $\mathcal{B}$ are both $O(1)$ in n and ϵ. While (2.195) is an upper bound on the absolute error $|4\beta^{-1}L^{(n)}_{\epsilon,\mu}f(x) - \mathcal{L}f(x)|$ due to the presence of the *variance error*, the *bias error* estimate from Theorem 2.2.5 is asymptotically sharp and does not involve absolute values. Thus if ϵ and n are scaled such that the variance

error is fixed, then with high probability the error behaves approximately as a linear function in ϵ, i.e.

$$4\beta^{-1}L_{\epsilon,\mu}^{(n(\epsilon))}f(x) - \mathcal{L}f(x) \sim a + \epsilon b. \tag{2.198}$$

To fix the variance error, the number of data points n must depend on ϵ such that

$$\frac{\log n(\epsilon)}{n(\epsilon)\epsilon^{2+d/2}} \sim O(1). \tag{2.199}$$

After fixing the variance error, the slope of the linear function (2.198) yields an estimate for the prefactor of the bias error. Using this method, we will obtain a prefactor estimate for a 1D test system embedded in 2D.

2.4.1.2 1D test system

Consider a system on the unit circle $\mathbb{S}^1$ parameterized by arc length (Figure 2.3). Let it evolve according to the overdamped Langevin dynamics (1.51) with

$$V(\theta) = \left(4\cos^2\left(\frac{\theta}{2}\right) - \frac{3}{2}\right)^2. \tag{2.200}$$

In this case, the committor can be derived analytically. Let $0 < \theta_1 < \theta_2 < 2\pi$ be the minima of V where

$$\theta_1 = 2\cos^{-1}\left(-\frac{\sqrt{3}}{2\sqrt{2}}\right) \approx 1.8235, \quad \theta_2 = 2\cos^{-1}\left(\frac{\sqrt{3}}{2\sqrt{2}}\right) \approx 4.4597. \tag{2.201}$$

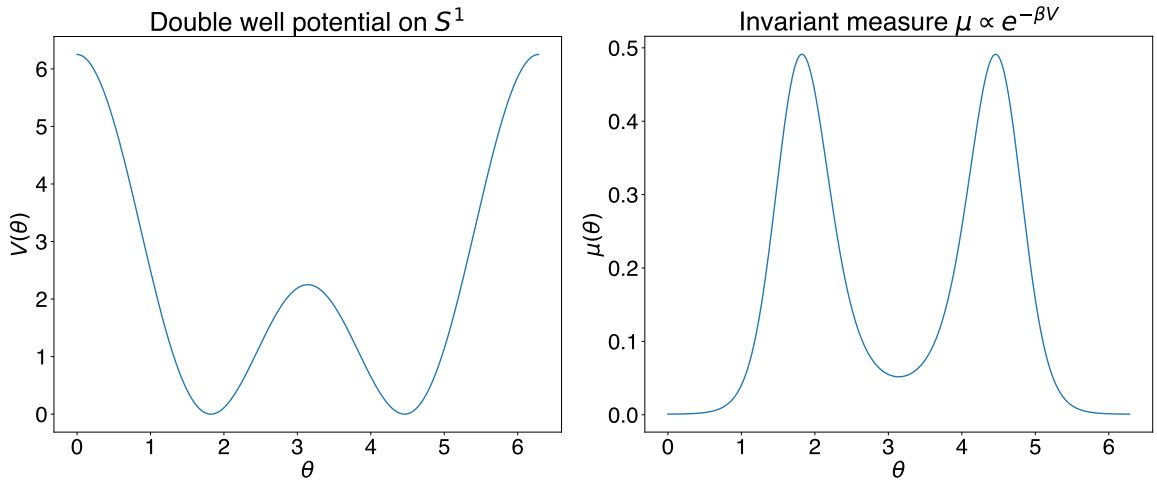


Figure 2.3: The potential $V(\theta)$, $0 \leq \theta < 2\pi$, given by (2.200) and the invariant measure μ for a 1D system on $\mathbb{S}^1$ following the overdamped Langevin dynamics (1.51) with the drift ∇V.

Set $A = [\theta_1 - r, \theta_1 + r]$ and $B = [\theta_2 - r, \theta_2 + r]$ where $r = 0.1$. Moreover, fix $\beta = 1$. Then the committor is

$$
\begin{aligned}
q(\theta) &= \frac{\int_{\theta_1+r}^{\theta} \exp\left(\beta V(\theta')\right) d\theta'}{\int_{\theta_1+r}^{\theta_2-r} \exp\left(\beta V(\theta')\right) d\theta'}, && \theta \in [\theta_1 + r, \theta_2 - r], \\
q(\theta) &= \frac{\int_{\theta}^{2\pi} \exp\left(\beta V(\theta')\right) d\theta' + \int_{0}^{\theta_1-r} \exp\left(\beta V(\theta')\right) d\theta'}{\int_{\theta_2+r}^{2\pi} \exp\left(\beta V(\theta')\right) d\theta' + \int_{0}^{\theta_1-r} \exp\left(\beta V(\theta')\right) d\theta'}, && \theta \in [\theta_2 + r, 2\pi), \\
q(\theta) &= \frac{\int_{\theta}^{\theta_1-r} \exp\left(\beta V(\theta')\right) d\theta'}{\int_{\theta_2+r}^{2\pi} \exp\left(\beta V(\theta')\right) d\theta' + \int_{0}^{\theta_1-r} \exp\left(\beta V(\theta')\right) d\theta'}, && \theta \in [0, \theta_1 - r].
\end{aligned}
\tag{2.202}
$$

2.4.1.3 The experimental setup

To demonstrate the error model in Theorem 2.1.1 and an improvement in the bias error prefactor (2.12), the following experiment is conducted.

1. For each ϵ, the number $n(\epsilon)$ is chosen such that the error behaves according to (2.199):

$$\frac{n(\epsilon)}{\log(n(\epsilon))} = 0.25\epsilon^{-5/2}. \tag{2.203}$$

The constant prefactor of 0.25 is chosen as the $O(1)$ term in (2.199) to make the resulting number of points $n(\epsilon)$ fall into the interval $[10^4, 3 \cdot 10^4]$ for $\epsilon \in [0.023, 0.033]$. This range of values of ϵ was chosen so that a dense kernel matrix could be stored in memory and eigendecomposed on our computational resources without requiring any sparsification of the kernel matrix or randomization of the eigensolver.

2. To measure the effect of changing the sampling density, for each choice of ϵ, $n(\epsilon)$ points are drawn i.i.d. on $\mathbb{S}^1$ using two sampling densities: the uniform density ρ^u and a non-uniform density $\rho^{n.u}$ specified in (2.205). The resulting datasets are defined as follows:

$$\mathcal{X}^u = \psi\left(\{\theta_i^u\}_{i=1}^n\right), \qquad \theta_i^u \sim \rho^u := \mathcal{U}[0, 2\pi], \tag{2.204}$$

$$\mathcal{X}^{n.u} = \psi\left(\{\theta_i^{n.u}\}_{i=1}^n\right), \qquad \theta_i^{n.u} \sim \rho^{n.u} := \pi + 0.1 + \sigma\pi\{\mathcal{N}(0,1)\}. \tag{2.205}$$

The function $\psi(\theta) = [\cos\theta, \sin\theta]^\top$ is the embedding map $\psi : \mathbb{S}^1 \to \mathbb{R}^2$ and $\{\mathcal{N}(0,1)\}$ denotes the fractional part of a normally distributed random variable. The fractional normal distribution is shifted such that the resulting distribution $\rho^{n.u}$ oversamples around the point $\theta = \pi + 0.1$ lying in the transition region and undersamples around the sets A and B. The parameter σ modulates the variance of the fractional normal distribution. It was set to $\sigma = 0.2$. The empirical densities of the data are visualized in Figure 2.4.

3. To measure the effect of changing the test function, $L_{\epsilon,\mu}^{(n(\epsilon))} f(x)$ was computed for two choices

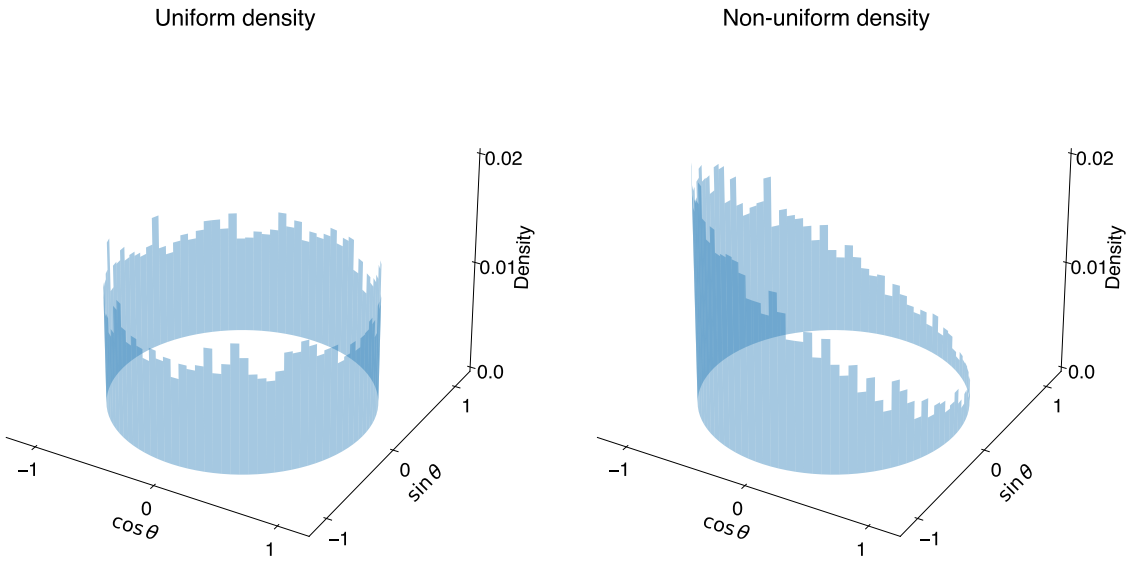


Figure 2.4: The empirical densities of $\mathcal{X}^u$ and $\mathcal{X}^{n.u}$. Note that the nonuniform points $\mathcal{X}^{n.u}$ are clustered in the region around $\theta = \pi + 0.1$.

of test function: the committor q given by formula (2.202) and $t(\theta) = \sin(\theta)$ (Figure 2.5). The evaluation point was fixed to be $\theta = \pi$.

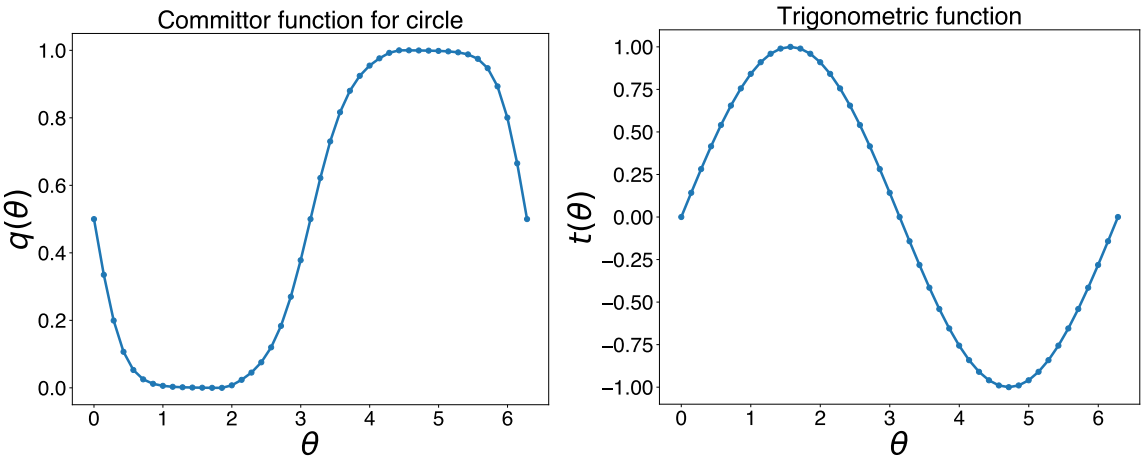


Figure 2.5: The two choices of f for evaluating the consistency error as in (2.198): the committor q (left) and the trigonometric polynomial $t(\theta) = \sin\theta$ (right).

4. For a given combination of sampling density and test function, the error in the left-hand side of (2.198) was computed. This error is a function of ϵ only.

The above experiment was conducted for 10 equispaced values of ϵ in the range $[0.023, 0.033]$. Moreover, since $4\beta^{-1}L_{\epsilon,\mu}^{(n(\epsilon))}f(x) - \mathcal{L}f(x)$ is random due to the i.i.d. draws from ρ, the experiment was repeated 50 times for every value of ϵ to mitigate the statistical variation. Then a linear fit to (2.198) was obtained to this data as a function of ϵ and the slope of this fit was taken as the estimate of the prefactor of the bias error for the given combination of f and ρ. A total of four linear fits were produced. The results are presented in Figure 2.6.

2.4.1.4 Summary of numerical results for reduction in bias error

We computed the error $4\beta^{-1}L_{\epsilon,\mu}^{n(\epsilon)}f(x) - \mathcal{L}f(x)$ for $f = q, t$ and $\rho = \rho^u, \rho^{n.u}$ as a function of ϵ ranging in $[0.023, 0.033]$. In Figure 2.6, a linear fit is obtained for $4\beta^{-1}L_{\epsilon,\mu}^{n(\epsilon)}f(x) - \mathcal{L}f(x)$ for each choice of f and ρ, where the slope of the linear regression is then an approximation of the bias error prefactor involved. Note that the bias error prefactor in general may be negative since the bias error formula in Theorem 2.2.5 does not require taking absolute values. Here, it is of interest to quantify the prefactor of lowest *magnitude* to measure which combination of sampling density and test function gives faster convergence of the bias error. The estimates of the prefactor magnitude are presented in Table 2.1. We find that the bias error prefactor of the lowest magnitude occurs when using a uniform density as input to TMDmap and applying the resulting generator to the committor function, presumably due to the uniform sampling and the committor function canceling the bias prefactors $\mathcal{B}_2$ and $\mathcal{B}_3$ at the given point x.

	Sampling density, Test function			
	$\rho^{n.u}$,$t(\theta)$.	ρ^{u}, $t(\theta)$	$\rho^{n.u}$, q	ρ^{u}, q
Abs. bias error prefactor	1.024	0.778	1.148	**0.398**

Table 2.1: The bias error prefactor b in (2.198) is estimated for two choices of ρ and two choices of f for the 1D system described in Section 2.4.1.2. The overall prefactor of lowest magnitude is obtained when using a uniform sampling density and applying the TMDmap generator to the committor function. This fact can be explained by the cancellations of the prefactors $\mathcal{B}_2$ and $\mathcal{B}_3$ in Theorem 2.2.5.

2.4.2 Calculating the committor function

The goal of this section is to compute the committor as the solution to (2.3) using TMDmap. The main focus will be on studying the discrepancy between the numerical solution q^{TMD} and the true solution q as a function of the sampling density ρ and kernel bandwidth ϵ.

2.4.2.1 Theoretical considerations

Corollary 2.2.16 suggests that solving the committor problem numerically with a uniform sampling density ρ will likely produce improvements in the accuracy of the numerical solution. Furthermore, bandwidth ϵ needs to be commensurate with the fixed number of points n to ensure that the error formula in Theorem 2.1.1 holds.

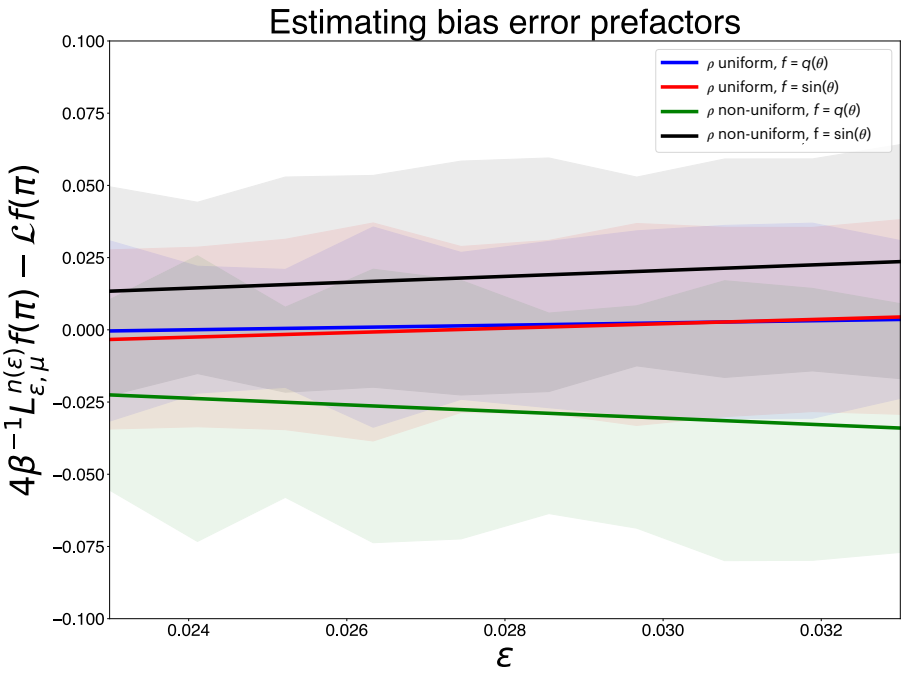


Figure 2.6: The consistency error for the committor, $f = q(\theta)$, and for $f = \sin(\theta)$ at $\theta = \pi$ as a function of ϵ. The number of points $n(\epsilon)$ for each ϵ is chosen according to (2.203) to keep the variance error constant. The datasets we sampled according to the uniform and the nonuniform densities in (2.204) and (2.205) respectively.

2.4.2.2 Optimal bandwidth

Given fixed n, ρ, μ, and x, the right-hand side of (2.195) may be minimized for ϵ to obtain an optimal bandwidth ϵ^* given by

$$\epsilon^* \sim (1 + d/4)^{\frac{1}{2+d/4}} \left(\frac{\log n}{n}\right)^{\frac{1}{4+d/2}} \left|\frac{V}{\mathcal{B}_1 + \mathcal{B}_2 + \mathcal{B}_3}\right|^{\frac{1}{2+d/4}}. \tag{2.206}$$

A similar version of the optimal bandwidth formula was derived in [Sin06] for Laplacian eigenmaps. That work, however, stopped short of expanding further on this formula due to the complicated dependence of the optimal bandwidth on manifold-related parameters. In contrast, this chapter offers a full enumeration of these parameters given by $\mathcal{V}$, $\mathcal{B}_1$, $\mathcal{B}_2$ and $\mathcal{B}_3$. Moreover, the pointwise error in the

left-hand side of (2.195) splits into two regimes locally around ϵ^*.

1. If $\epsilon \leq \epsilon^*$, the error is dominated by the divergence of the variance error term $\epsilon^{-(1+d/4)}$.

2. If $\epsilon \geq \epsilon^*$, the error is dominated by the linear bias error $\epsilon\mathcal{B}$. A reduction in the magnitude of $\mathcal{B}$ will make the error diverge more slowly from the minimum at ϵ^*. Consequently, a smaller bias error prefactor will make the error more stable to small perturbations around ϵ^*.

It is difficult to explicitly compute prefactors in Theorem 2.1.1. Hence it is not advisable to select the bandwidth using (2.206). The common practice is to use a heuristic estimate for ϵ^* instead. We use the *kernel sum* or the *Ksum* test [SEKC09, BH16, ECT22]:

$$\epsilon^*_{\mathsf{Ksum}} = \operatorname*{argmax}_{\epsilon} \frac{\partial \log S(\epsilon)}{\partial \log \epsilon}, \quad S(\epsilon) := \sum_{ij} [K_\epsilon^{(n)}]_{ij}. \tag{2.207}$$

To implement the Ksum test numerically, the sum $S(\epsilon)$ of all entries of the kernel matrix $[K_\epsilon^{(n)}]_{ij} = \exp(-\|x_i - x_j\|^2/\epsilon)$ is computed at a range of ϵ values and its logarithm is plotted against $\log \epsilon$. The value of ϵ for which the slope of this plot is the largest is taken to be $\epsilon^*_{\mathsf{Ksum}}$. This choice is justified by the fact that as $n \to \infty$, $\log S(\epsilon)$ is an affine function of $\log \epsilon$, which is the desired intermediate asymptotic. We refer the interested reader to [ECT22, Section III.C] for a detailed explanation. Although this test is not exact, it gives a good initial guess for a more exhaustive search for the optimal ϵ. We highlight $\epsilon^*_{\mathsf{Ksum}}$ in Figures 2.9, 2.8, 2.12, and 2.11. The RMS error is flatter around ϵ^* and is nearly optimal at $\epsilon^*_{\mathsf{Ksum}}$ in all our test problems when we use quasi-uniform sampling densities.

2.4.2.3 Optimal sampling

The preceding discussion on choosing optimal ϵ reveals that the sensitivity of the error to small perturbations around ϵ^* can be ameliorated through reductions in the bias error prefactors. A quasi-uniform sampling density can be used to reduce $\mathcal{B}_2$ and $\mathcal{B}_1$, thus necessitating increased sampling of rare high-energy configurations lying in the transition region away from the metastable states. There are numerous ways of such rare event sampling: some examples relevant to molecular simulation are temperature acceleration [MVE06], importance sampling [BLT13], umbrella sampling [Käs11], splitting methods [WAS20], as well as recent work utilizing generative models such as normalizing flows [FCR+22], Generative Adversarial Networks (GANs) or diffusion models [SDWMG15, GPAM+20]. See [HLS+22] for a recent comprehensive survey on enhanced sampling in molecular dynamics. In the present case, the situation is complicated because the low-dimensional manifold $\mathcal{M}$ where ρ is supported is unknown and can only be accessed through sampling. Consequently, in this setting, metadynamics [LP02b] emerges as a cheap and robust option to generate samples on $\mathcal{M}$. These samples can then be postprocessed to δ-nets to enhance spatial uniformity.

Metadynamics can be described as a modification of the Euler-Maruyama scheme of discretizing overdamped Langevin dynamics (1.51). The goal of metadynamics is to bias the system towards exploring regions of the energy landscape that are not sufficiently sampled under the original potential $V(x)$. To do so, the metadynamics algorithm introduces a bias potential $W(x,t)$ that is added to the original potential $V(x)$ at each step of the Euler-Maruyama algorithm. The bias potential is constructed as a sum of Gaussian potentials deposited during the simulation time t:

$$W(x,t) = \sum_{t_j<t} w_0 \exp\left(-\frac{(\theta(x)-\theta(x(t_j)))^2}{2\sigma^2}\right), \tag{2.208}$$

where $x(t_j)$ is the set of atomic positions at time t_j, w_0 is the height of each deposited Gaussian potential, σ controls the width of the Gaussians, and $\theta(\cdot)$ is a vector of collective variables. The collective variables represent a coarse-grained model of a complex stochastic system, such as proteins. In our simulations, we will simply use $\theta(x) = x$. The bias potential $W(x,t)$ modifies the original potential energy, leading to an effective potential energy $V_{\text{eff}}(x,t) = V(x) + W(x,t)$. Thus, the metadynamics fills up the free energy wells with Gaussian bumps enabling the system to escape from metastable states and explore the configurational space. It is summarized in Algorithm 1.

Algorithm 1 Metadynamics

Input : Gaussian height w_0, Gaussian width σ_0, deposition stride s, maximum number of steps n, timestep Δt.
Output: Visited states trajectory $\mathcal{X}(n) = \{x_1, \dots x_n\}$
Initialize the visited states trajectory: $\mathcal{X}(n) \leftarrow [\,]$ Initialize the history of visited states: $H \leftarrow \emptyset$ Initialize the potential: $V_{\text{eff}} \leftarrow V$ **for** $i \leftarrow 1$ **to** N_{steps} **do**
  Sample a new state: $x_i \leftarrow$ Euler-Maruyama step of size Δt with potential V_{eff} Append the current state to the trajectory: $\mathcal{X}(n) \leftarrow \mathcal{X}(n) \cup \{x_i\}$
  if $i \bmod s = 0$ **then**
    Deposit Gaussian: $V_{\text{eff}} \leftarrow V_{\text{eff}} + w_0 \exp\left(-\frac{(\theta(\cdot)-\theta(x_i))^2}{2\sigma^2}\right)$
  end
end

The δ-net algorithm outlined in Algorithm 2 is a greedy procedure used to select a sample of points from the dataset $\mathcal{X}(n)$ such that any pair of points is at least a distance of δ apart. The output point cloud tends to be spatially uniform and approximates samples from the uniform density on $\mathcal{M}$.

Algorithm 2 δ-net Algorithm

Input : Set of points P, threshold distance δ
Output: δ-net $D \subseteq P$
Initialize an empty set $D \leftarrow \emptyset$
foreach $p \in P$ **do**
  if *no point in* D *is within distance* δ *from* p **then**
    Add p to the δ-net: $D \leftarrow D \cup \{p\}$
  end
end
return D

2.4.2.4 Experimental setup.

To understand the effect of sampling density ρ and the bandwidth parameter ϵ on the quality of the numerical committor q^{TMD} we will examine the root mean squared error (RMSE):

$$\text{RMSE}\,|q - q^{\mathsf{TMD}}| = \left(\sum_{i=1}^{n} |q(x_i) - q^{\mathsf{TMD}}(x_i)|^2 \right)^{1/2}. \tag{2.209}$$

To investigate the effect of ρ, the underlying point cloud $\mathcal{X}(n)$ for computing q^{TMD} via the TMDmap algorithm will be drawn from three different sampling densities via the following sampling algorithms.

1. The **Euler-Maruyama** sampling (see e.g. [KPKP92]) leads to the data being sampled essentially through the invariant **Gibbs density** $\mu \propto \exp(-\beta V(x))$.

2. **Metadynamics** outlined in Algorithm 1 results in a more uniform distribution of data on $\mathcal{M}$.

3. **Metadynamics with δ-net** first generates data through Algorithm 1 and then post-processes them through Algorithm 2. We will illustrate that such configurations tend to be spatially quasi-uniform on $\mathcal{M}$.

For each choice of sampling density, we will draw a fixed number of points and feed the resulting point cloud as input to the TMDmap algorithm. To study the effect of ϵ, we will then vary ϵ and compute the RMSE as a function of ϵ for two test systems governed by the overdamped Langevin dynamics (1.51) with Müller's potential and a two-well potential in $\mathbb{R}^2$. Since both of these examples are 2D, the function $\theta(\cdot)$ in (2.208) is chosen to be the identity map.

2.4.2.5 Müller's potential

We consider a system governed by (1.51) with V being Müller's potential [MB79]:

$$V(x_1, x_2) = \sum_{i=1}^{4} D_i \exp\left\{a_i(x_1 - X_i)^2 + b_i(x_1 - X_i)(x_2 - Y_i) + c_i(x_2 - Y_i)^2\right\} \tag{2.210}$$

where

$$\begin{aligned}
[a_1, a_2, a_3, a_4] &= [-1, -1, -6.5, 0.7], \\
[b_1, b_2, b_3, b_4] &= [0, 0, 11, 0.6], \\
[c_1, c_2, c_3, c_4] &= [-10, -10, -6.5, 0.7], \\
[D_1, D_2, D_3, D_4] &= [-200, -100, -170, 15], \\
[X_1, X_2, X_3, X_4] &= [1, 0, -0.5, -1], \\
[Y_1, Y_2, Y_3, Y_4] &= [0, 0.5, 1.5, 1].
\end{aligned}$$

We computed q^{TMD} with the three types of input density specified in Section 2.4.2.4. The timestep for the Euler-Maruyama sampling was $\Delta t = 10^{-4}$, and the trajectory length was 10^6 timesteps. The trajectory was subsampled to keep every 100th point resulting in a point cloud of size 10^4. In the metadynamics approach, a Gaussian bump with $\sigma = 0.1$ and $w_0 = 0.5$ was deposited at every 100th timestep. The dataset was then post-processed to a δ-net. The reactant set A and product set B were chosen to be the balls of radius 0.1 centered at $[-0.558, 1.441]$ and $[0.623, 0.028]$ respectively. To validate the results, the numerical solution to the committor problem was computed using the finite element method (FEM). The computational domain for the FEM was $\{x \in \mathbb{R}^2 \mid V(x) \leq 10\}$. The

homogeneous Neumann boundary condition was imposed on the outer boundary where $V = 10$ (Figure 2.7). This FEM solution was used as a surrogate for the true solution in computing the RMSE in (2.209).

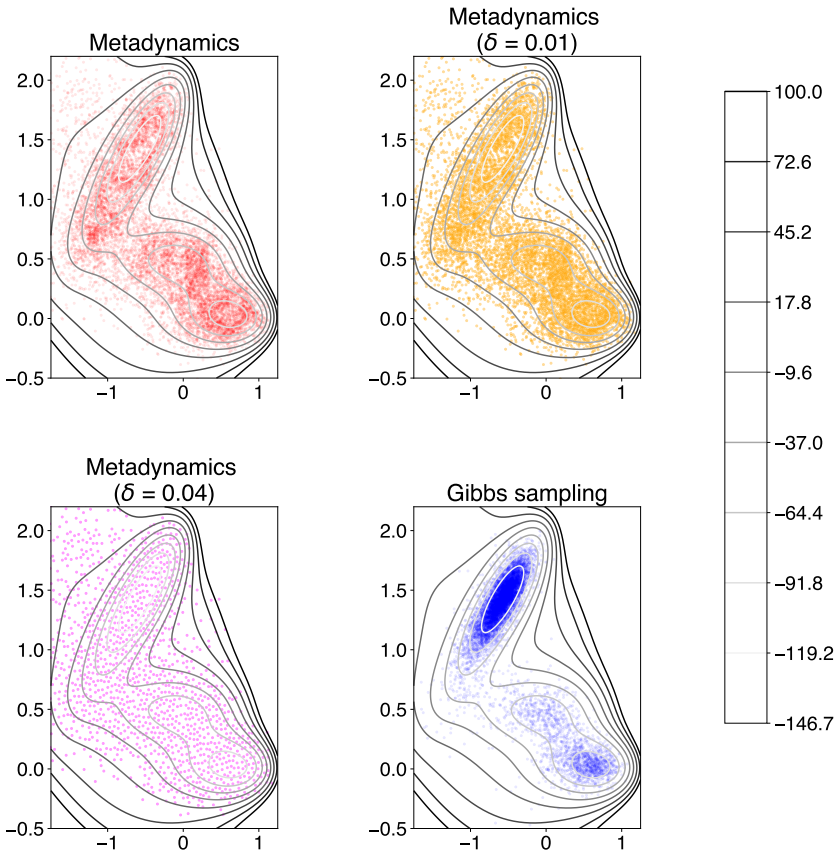


Figure 2.7: Landscape of Müller's potential with different sampling densities

Since it is of interest to study the effect of uniformization on the RMSE, we modulate δ in the δ-net algorithm used for post-processing the metadynamics dataset containing 10^4 points. For each δ, the RMSE over a varying ϵ is computed and presented in Figure 2.8. The δ-net postprocessing tends to result in a flatter error curve around the minimum hence making the numerical solution more robust with respect to the choice of the bandwidth.

The TMDmap solutions obtained using datasets generated with the three sampling types and as well as the FEM mesh consisting of nearly equilateral triangles of nearly equal sizes are compared

in Figure 2.9. The error curve for the metadynamics plus δ-net dataset has a flat region around its minimum and achieves almost as small a minimum as the one for the FEM mesh point cloud.

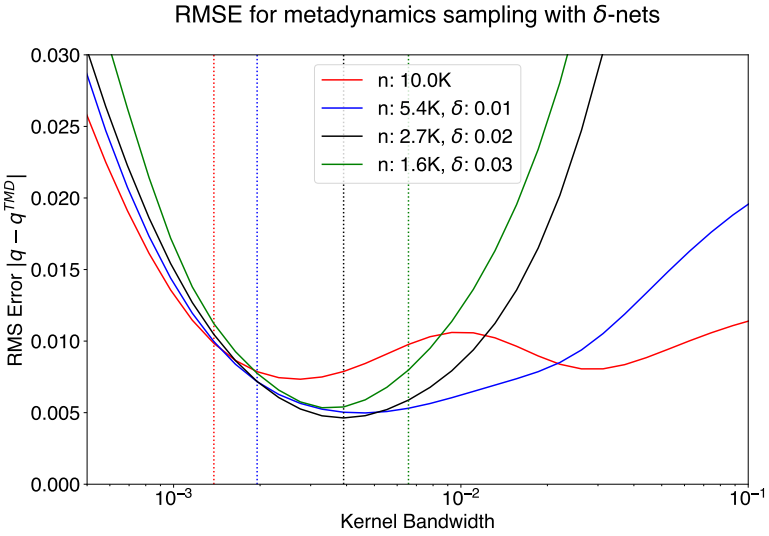


Figure 2.8: We track the RMSE for q^{TMD} computed for Müller's potential as a function of ϵ for different degrees of spatial uniformity. The dotted line marks the ϵ^*_{Ksum} computed using (2.207).

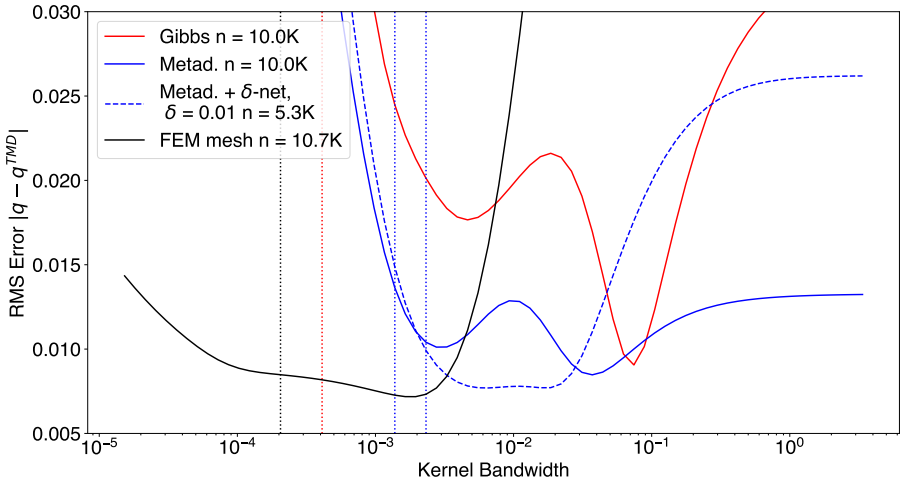


Figure 2.9: Tracking the RMSE for q^{TMD} in Müller's potential for different sampling densities as a function of ϵ. Note that using the mesh from the FEM algorithm gives the least error.

2.4.2.6 Two-well potential in 2D

A 2D test system governed by (1.51) with a two-well potential

$$V(x_1, x_2) = (x_1^2 - 1)^2 \tag{2.211}$$

at $\beta = 1.0$ magnifies the advantage of using point clouds generated by metadynamics plus δ-net even stronger. The sets A and B are balls of radius 0.1 centered at $[-1, 0]$ and $[1, 0]$ respectively. The datasets are sampled from the Gibbs density, metadynamics, and metadynamics with δ-net with $\delta = 0.1, 0.02, 0.03$, and 0.04. Four of these datasets are displayed in Figure 2.10.

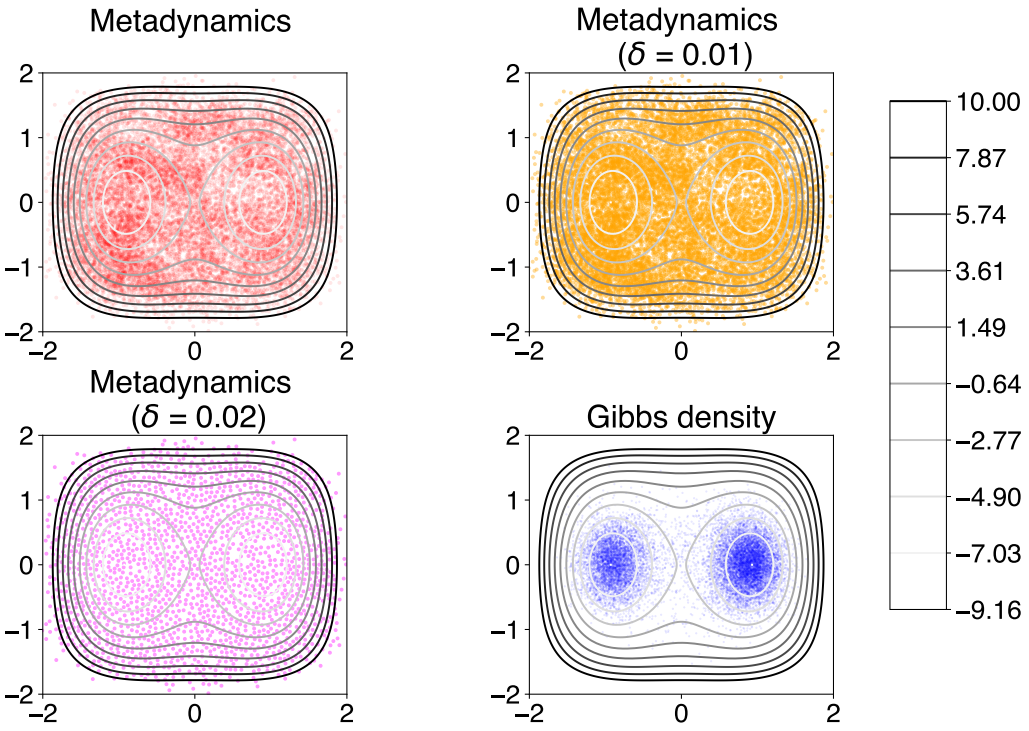


Figure 2.10: The two-well potential landscape with different sampling densities

In Figure 2.11, the effect of δ-net is quantified by tracking the RMSE as a function of ϵ for each value of δ. We find a similar pattern as for the test problem with Müller's potential as in Figure 2.8

that moderate levels of uniformization (at $\delta = 0.02$ in the case of V given by (2.211)) improves and stabilizes the RMSE at the optimal ϵ.

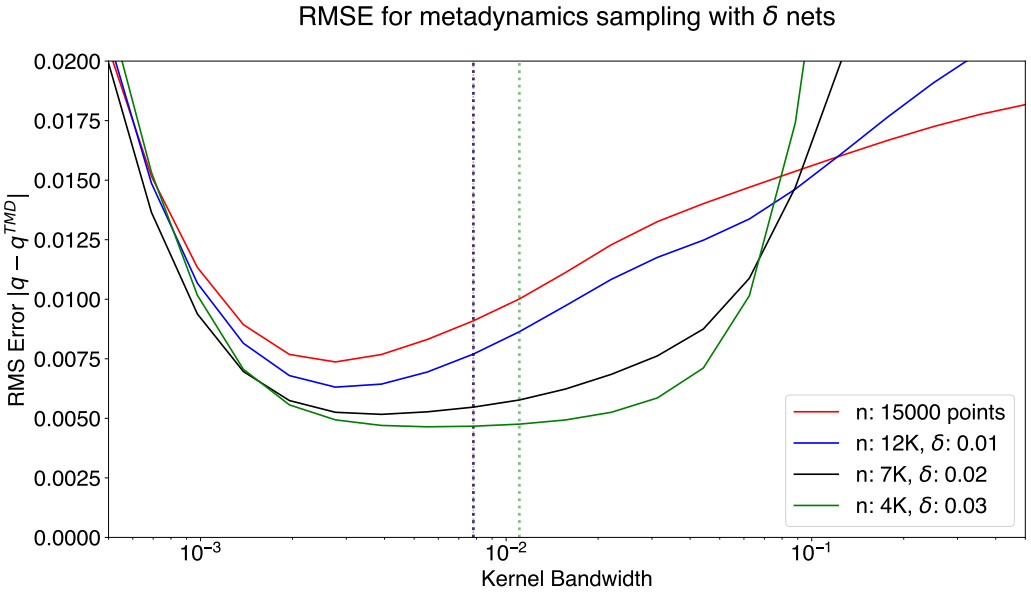


Figure 2.11: The RMSE for q^{TMD} the test system with the two-well potential (2.211) obtained using datasets generated via metadynamics and metadynamics plus δ-net with $\delta = 0.01$, 0.02, and 0.03.

In Figure 2.12, the three choices of sampling density are compared. Adding uniformization using δ-nets produces the same pronounced effect as for Müller's potential in Figure 2.9. Additionally, since this is a 2D example, it is also possible to provide to TMDmap the set of points from the nearly regular FEM mesh used for producing the reference solution. The TMDmap committor q^{TMD} obtained using the FEM mesh dataset has a notably smaller error for a broad range of ϵ values than it is for the other point clouds used here. This reinforces the hypothesis that uniform sets are error-optimal for TMDmaps. The caveat, however, is that, in higher dimensions, such uniform meshing is infeasible to generate and thus practitioners must resort to randomly sampled data on $\mathcal{M}$.

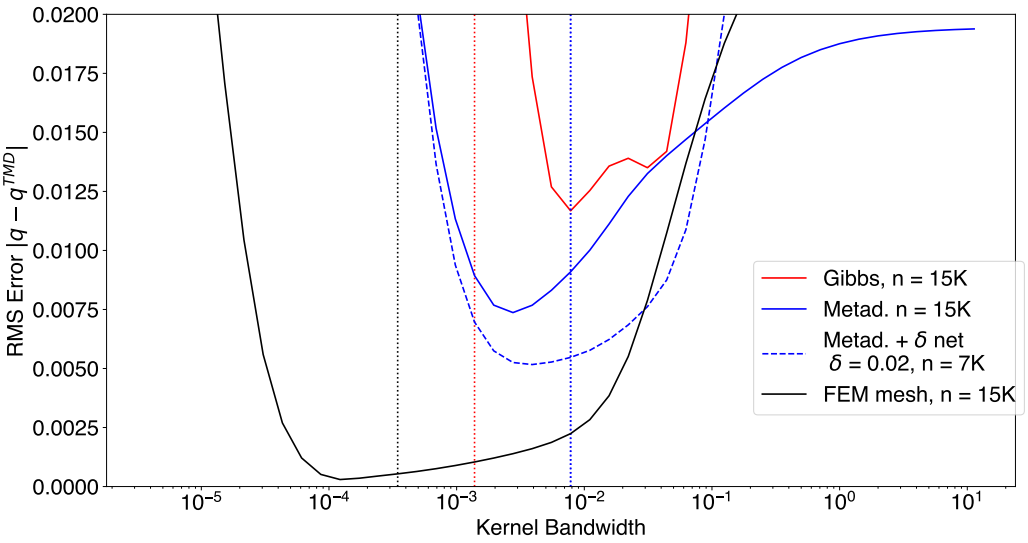


Figure 2.12: Tracking the RMSE for different sampling densities as a function of ϵ. Note that using the mesh from the FEM algorithm gives the least error.

2.4.2.7 Summary of numerical results

Our numerical experiments in Figures 2.8, 2.9, 2.11, and 2.12 confirm that:

1. Spatial uniformization improves not only the accuracy of the RMSE $|q - q^{\mathsf{TMD}}|$ at ϵ^* but also the *stability*. Our error estimate suggests that this results from the faster bias error convergence due to quasi-uniform ρ.

2. Metadynamics with δ-net is a viable strategy to realize the faster bias error convergence rate.

3. If the original data $\mathcal{X}$ has good coverage of the manifold $\mathcal{M}$ then *deleting* some points can *improve* the RMSE as long as those deletions improve the spatial uniformity of the data. In other words, reducing the number of data points can enhance the accuracy of the solution!

We also note that the increase of δ for Muller's potential in Figure 2.8 and the two-well potential in Figure 2.11 seems to have an opposite effect on the sensitivity of the error in the committor to the bandwidth parameter ϵ. In general, if we have a dense enough set of sample points, the increase in δ–to a certain extent–results in more uniform configurations of points, which reduces the error in the committor. This is what we observe in the case of the two-well potential in Figure 2.11. Muller's potential is more complex than the two-well potential and the initial pointcloud may not be dense enough. Consequently, the δ-net with the largest value of δ became too rarefied, resulting in a higher sensitivity of the committor to ϵ.

2.5 Conclusion

In this chapter, we have derived sharp error estimates revealing the precise error rates for the target measure diffusion map. Our results extend the consistency theory for graph Laplacians to a

manifold learning setting involving density reweighting. The main advantage of incorporating the TMDmap density reweighting is the free choice of the sampling density, enabling the use of arbitrary enhanced sampling algorithms for generating the input data.

We have provided a principled approach for tuning the sampling density ρ and kernel bandwidth parameter ϵ to improve the accuracy of the TMDmap algorithm because our results are *asymptotically sharp*. The obtained error formula contains a complete quantification of all prefactors involved in the bias and variance errors. This allows us to find strategies for reducing the approximation error when discretizing the backward Kolmogorov operator $\mathcal{L}$ or when solving a Dirichlet boundary-value problem (BVP) with this discretization. These formulas illuminate that (a) solving a homogeneous BVP such as the committor problem and/or (b) sampling $\mathcal{M}$ with a uniform density are the settings in which bias and variance errors are attenuated. Importantly, we have demonstrated that these improvements in accuracy are attainable in practice.

A significant aspect of our work has also been the exploration of uniform subsampling via δ-nets as a robust and simple technique for reducing the error in the TMDmap approximation to the committor. The consistency formula from Theorem 2.1.1 shows that *a priori* quasi-uniform densities may yield pointwise speedups in the convergence of both bias and variance. Here we obtain a quasi-uniform point cloud with expansive coverage of $\mathcal{M}$ via a simple greedy procedure that subsamples the input dataset into a δ-net. This substantially improves the stability and accuracy of TMDmap. Such distance-based quasi-uniform sampling has previously been used for model reduction [CM17, SC25] and in molecular dynamics applications [RCV23].

Additionally, we have probed the interplay between δ and ϵ numerically and illustrated that, for low-dimensional cases, the δ–net tends to yield more favorable scaling between the optimal bandwidth and number of points beyond the optimal scaling for general densities in Theorem 2.1.1. Although our

justification for the use of uniform densities is grounded in the i.i.d. assumption, we find that the scaling between n and ϵ error is also improved with δ-nets where the i.i.d. assumption no longer holds. We aim to address this interplay of n, ϵ, and δ in future work.

Chapter 3: Learning collective variables

3.1 Introduction

Collective variable (CV) discovery is a fundamental problem in computational statistical physics involving the design of coarse-grained low-dimensional representations of high-dimensional stochastic processes. In practice, CV discovery is particularly important for the design of novel materials [NTM+22, ZNP+19, KIRP21] or drug discovery [CPGM15, ABG06] and for the understanding of rare events in molecular dynamics (MD) simulations of biomolecules and chemical reactions [YSZ+19, PNV19, VTP16]. A standard model for such MD simulations is the *overdamped Langevin dynamics* in Chapter 1, which describes the motion of N coordinates $X_t \in \mathbb{R}^N$ in a potential energy landscape driven by the potential V with random fluctuations driven by a temperature parameter β^{-1}.

The wells of the potential V encode metastable states—regions where the system remains trapped for long periods before transitioning due to thermal fluctuations. Simulating and analyzing these transitions is computationally expensive due to the wide range of timescales involved. Therefore, instead of working directly with the high-dimensional process X_t, we seek a collective variable $\xi : \mathbb{R}^N \to \mathbb{R}^d$ that maps X_t to a low-dimensional process $Y_t = \xi(X_t)$. Typically, CVs are designed with the following objectives:

1. Mapping by ξ should separate metastable states, enabling a clear description of transitions.

2. The CV should enable enhanced sampling to accelerate rare-event simulations.

3. The mapped process $Y_t = \xi(X_t)$ should reproduce the statistical properties of the original dynamics.

Thus, while traditional feature learning techniques in artificial intelligence optimize for expressivity or discriminative power, CVs must respect the dynamical and geometric properties of X_t, making their construction significantly more challenging. There is nonetheless a rich line of work for designing metastable state-separating CVs and CVs for enhanced sampling using data-driven methods, thus satisfying objectives (1) and (2). However, a key hindrance to achieving the objective (3) is that the projected process Y_t is generally not closed because its evolution depends not only on its current state but also on latent high-dimensional variables. This lack of closure makes direct simulation and statistical reproduction of Y_t infeasible. This is seen by applying Ito's lemma to $\xi(X_t)$:

$$dY_t = d\xi(X_t) = \mathcal{L}\xi(X_t)\,dt + \sqrt{2\beta^{-1}}\mathsf{det}(D\xi(X_t)D\xi(X_t)^\top)^{1/2}\,dB_t. \tag{3.1}$$

Here B_t is a rescaled Brownian motion in $\mathbb{R}^d$ and $\mathcal{L}$ is the infinitesimal generator of the overdamped Langevin dynamics (1.51):

$$B_t = \frac{D\xi}{\det\left(D\xi D\xi^\top\right)^{1/2}} W_t, \tag{3.2}$$

$$\mathcal{L}f := \beta^{-1}\Delta f - \nabla V \cdot \nabla f. \tag{1.52}$$

One approach to constructing a closed effective process is the *Zwanzig closure*, i.e conditioning with respect to the law of Y_t, which ensures that the time-marginals of the reduced process match those of the full dynamics [GKS04]. However, due to the time-dependence of the drift and diffusion coefficients, such a reduced process is difficult to simulate. As a remedy, Legoll and Lelièvre [LL10] proposed an alternative *effective dynamics* Z_t, where instead of conditioning on the full history of X_t,

the evolution of Y_t is conditioned on the invariant measure $\mu = Z^{-1} \exp(-\beta V)$ associated with X_t:

$$dZ_t = b(Z_t)\, dt + \sqrt{2\beta^{-1}} M^{1/2}(Z_t)\, dB_t. \tag{3.3}$$

The drift b, the matrix M and the process B_t are described as follows:

$$b(z) = \mathbb{E}_\mu[\mathcal{L}\xi(X) \mid \xi(X) = z], \tag{3.4}$$

$$M(z) = \mathbb{E}_\mu[D\xi(X) D\xi(X)^\top \mid \xi(X) = z]. \tag{3.5}$$

Thus, the effective dynamics $Z_t \in \mathbb{R}^d$ are constructed by conditioning the drift and diffusion of the projected dynamics (3.1) Y_t by fixing ξ under the measure μ. Z_t is intrinsically low-dimensional and can be simulated using established numerical methods [WVE04, MFVEC06]. The downside of the effective dynamics is that the laws of Z_t and Y_t do not coincide, so Z_t may obfuscate the statistics of Y_t. Finding error estimates for guaranteeing the proximity of Z_t and Y_t is therefore a highly active topic of research with a number of works [LL10, LLS19, LZ19b, DLP+18, Sha17] exploring the cases arising from different combinations of assumptions on the underlying dynamics X_t, the CV ξ, and ways to measure distances between stochastic processes. In this chapter we focus on the particular case of *scale separation* where the potential $V := V_\varepsilon$ can be decomposed as

$$V = V_0 + \varepsilon^{-1} V_1^2, \quad \varepsilon \ll 1. \tag{3.6}$$

The comparatively large stiffness on the V_1^2 term created by the small parameter ε, causes the Langevin dynamics (1.51) to be confined to lie around the zero-level set of V_1, henceforth termed the *residence*

manifold:

$$\mathcal{M} := \{V_1 = 0\}. \tag{3.7}$$

Such a process X_t driven by V_ε undergoes small but fast fluctuations around $\mathcal{M}$ and slow fluctuations along $\mathcal{M}$, driven by the *driving potential* V_0. Potentials of the form (3.6) arise routinely in MD simulations where the inter-atomic energy is given by a sum of strong covalent bonds (corresponding to the stiff $\varepsilon^{-1}V_1^2$ term) and weak torsion energies (corresponding to the term V_0) and the slow motions created on the residence manifold are of interest in understanding biophysical phenomena like folding or nucleation. Notably, it was shown in [LL10] that the distance between Z_t (the effective dynamics (3.3)) and Y_t (the projected dynamics (3.1)) as measured by relative entropy can be made to scale with ε as $C + O(\varepsilon)$ if the following equation, the *orthogonality condition*, is satisfied:

$$D\xi \nabla V_1 = 0. \tag{OC}$$

The geometric meaning of (OC) is as follows: if $\mathcal{M}$ has co-dimension 1, then ∇V_1 is the outward pointing normal vector to $\mathcal{M}$–therefore, if $D\xi \perp \nabla V_1$ then the gradients of ξ must be perpendicular to this normal vector and lie in the tangent space of $\mathcal{M}$ at each point (see Figure 3.1). In a toy two-dimensional example [LL10] demonstrated that designing CVs which satisfied (OC) was instrumental in reproducing residence times via the effective dynamics Z_t. However, most realistic MD simulations are high-dimensional and consist of not one, but several confining potentials at different scales, resulting in complicated residence manifolds of intrinsic dimension significantly smaller than N. Nonetheless, (OC) provides a general geometric principle that can be extended beyond potentials

of the form (3.6) to encompass the broader case when X_t resides around a low-dimensional manifold $\mathcal{M}$ in $\mathbb{R}^N$ by suggesting the design of CVs whose gradients flow orthogonally to the normal directions of $\mathcal{M}$.

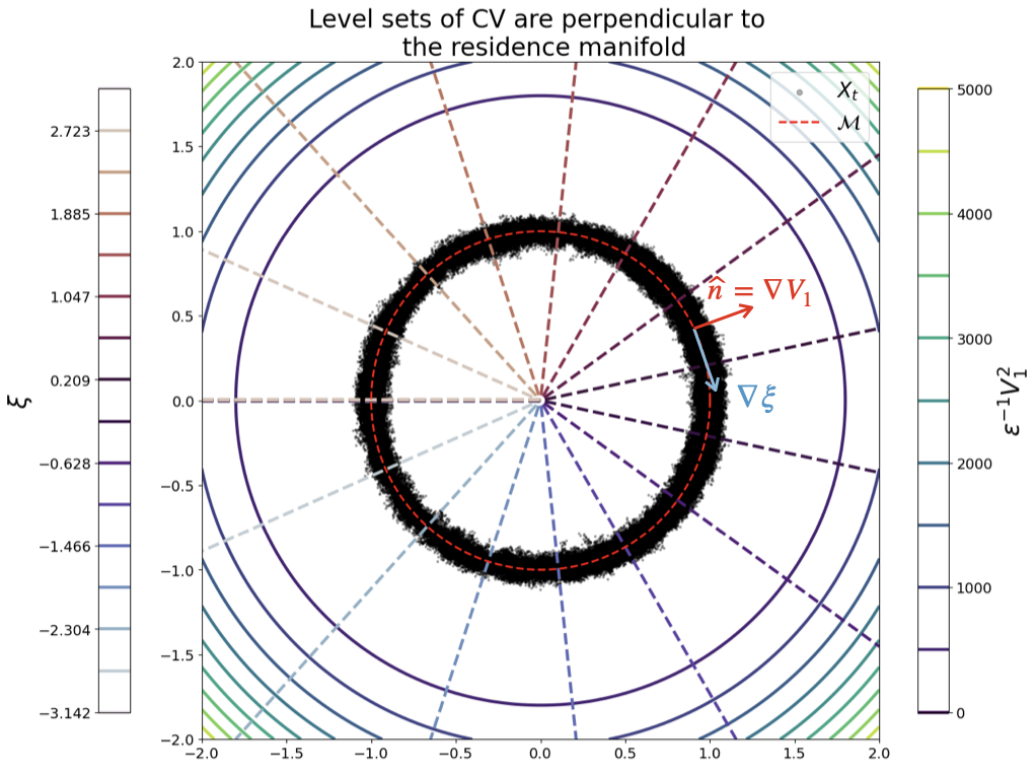


Figure 3.1: Illustrating the orthogonality condition: Here we simulate the dynamics (1.51) with the potential $V(x, y) = 100(x^2 + y^2 - 1)^2 + (x^2 - 1)^2$. Due to the large stiffness $\varepsilon^{-1} = 100$ on $V_1^2 = (x^2 + y^2 - 1)^2$, the process X_t (small black dots) lies around the level set $\mathcal{M} = \{V_1 = 0\}$, i.e the residence manifold. Here clearly the ideal collective variable, which follows the slow dynamics of the process around the circle, is the polar angle $\xi = \arctan(y/x)$. Notably, the level sets of ξ are perpendicular to $\mathcal{M}$ and its gradients lie in $T_p\mathcal{M}$ at each point.

3.1.1 Aims and contributions

In this chapter, we aim to apply the theory of quantitative coarse graining as outlined in [LL10, DLP+18, LZ19b] towards a numerical method for obtaining CVs.

1. We review the theory of quantitative coarse graining from [LL10, DLP+18, LZ19b] and demon-

strate how geometric conditions like (OC) emerge as sufficient conditions for reducing the discrepancy between the projected dynamics Y_t (3.1) and the effective dynamics Z_t. Moreover, we evaluate the analytical assumptions on CVs required by the aforementioned quantitative coarse graining theory in terms of reproduction of reaction rates in molecular dynamics simulations and distances between Z_t and Y_t. In particular, we prove that under the decomposition (3.6), the (OC) is sufficient for reducing both the relative entropy and pathwise distances between the effective and projected dynamics (Proposition 3.3.1). Moreover, we provide evidence that the reaction rate can be reproduced even if the diffusion tensor $M(z)$ is rank-deficient. We illustrate this fact for the anti-gauche transition rate in the *normal*-butane molecule C_4H_{10} (henceforth simply referred to as *butane*) (Table 3.3).

2. We propose an algorithm for designing CVs based on (OC) (Algorithm 3), where the CV is learned as an autoencoder whose encoder satisfies the (OC). We show how to integrate several tools from manifold learning such as diffusion maps [CKL$^+$08], diffusion nets [MSCC19], group invariance, and independent eigencoordinate selection [CSF19] to learn a neural network-based representation of $\mathcal{M}$ as *hypersurface* (Algorithm 7). We remark that this hypersurface $\widehat{\mathcal{M}}$, henceforth termed the *surrogate manifold* can contain self-intersections like knots despite the use of functionally independent coordinates (see top right panel of Figure 3.18). As a solution, we propose a novel manifold learning technique, Laplacian Conformal Autoencoder (LAPCAE) which can *undo* these self-intersections (Figure 3.13).

3. We underline the instrumental role played by group invariance in the post-processing of MD data. In particular, we propose mapping, or *featurizing* data sampled from overdamped Langevin dynamics X_t (1.51) through a feature map $\mathcal{F}$ with invariances to rotations, translations, or re-

flections. We show (Figure 3.11) that different feature maps can provide markedly different embeddings of the same data. Moreover, we demonstrate for butane that the positions of the hydrogen atoms are *necessary* for learning accurate representations of the dynamics. This point is of particular interest to the development of MD workflows, where currently it is standard practice to post-process simulation data by discarding coordinates of light atoms.

4. We conduct a detailed investigation of our algorithm on butane as a case study. Although butane is a toy model for studying conformational dynamics, it is nonetheless the simplest MD example that exhibits the characteristics of MD data, such as rotation and translation invariance, low intrinsic dimensionality, metastability, and slow and fast time scales. Moreover, the existence of a clear CV, the dihedral angle, makes it easier to evaluate and benchmark our proposed three-step algorithm. As a lookahead to our results, the CV learned by our algorithm for the feature map PlaneAlign reproduces the anti-gauche transition rate in butane with a relative error of nearly 10% while the dihedral angle yields a relative error of 24% (Table 3.3). The code to reproduce our experiments and train our models has been provided at `https://github.com/ShashankSule/CV_learning_butane`.

5. We demonstrate the applicability of our algorithm beyond butane by applying the residue manifold learning procedure to alanine dipeptide, a minipeptide of two amino acid residues, in vacuum. We show that featurization via PlaneAlign (applied to the α-Carbon and its neighbouring Carbon and Nitrogen atoms) yields a diffusion map embedding that is a 2D surface in 3D whose angular direction is parameterized by the dihedral angle θ_2 (Figure 3.22), demonstrating the utility of PlaneAlign as a featurization beyond the butane example. Furthermore, consistent with our findings for butane, we show that including hydrogen atoms is *necessary* for separating the

metastable states C7ax and C7eq in the diffusion map embedding.

3.1.1.1 Broad overview of algorithm

In this chapter, we propose Algorithm 3 where we learn a collective variable based on the orthogonality condition. We briefly summarize this algorithm here. In particular, the CV designed will be the encoder part of an autoencoder trained to satisfy the (OC). Since (OC) stipulates that the gradients of the CV be orthogonal to the normal vector of the residence manifold, we recover the residence manifold in low-dimensions via a neural-network parameterization. Then we train an encoder which passes through this parameterization with a training loss that penalizes the (OC). After training this collective variable, we estimate the transition rate in CV space via the transition path theory (TPT) approach further described in Section 3.5.4. We summarize these training and inference phases In Figures 3.2 and 3.3.

3.1.2 Related work

CV discovery is a thriving interdisciplinary subject at the intersection of probability theory, numerical analysis, biophysics, and, more recently, machine learning. On the theoretical front, since the proposal of the effective dynamics in [LL10], there have been several subsequent theoretical works that have extended the conditioning approach to multidimensional CVs, non-reversible processes, and non-isotropic diffusions. Moreover, these works have addressed distances between Z_t and Y_t as measured by relative entropy, Wasserstein distance, and the pathwise distance [DLP+18, LZ19a, HNS20, HNS20, LLS19]. On the computational side, there is a vast literature on designing CVs for MD simulations, starting with human-designed CVs [HWK+20, OA23, CSP+07, HK20, AON+08, SP13], data-

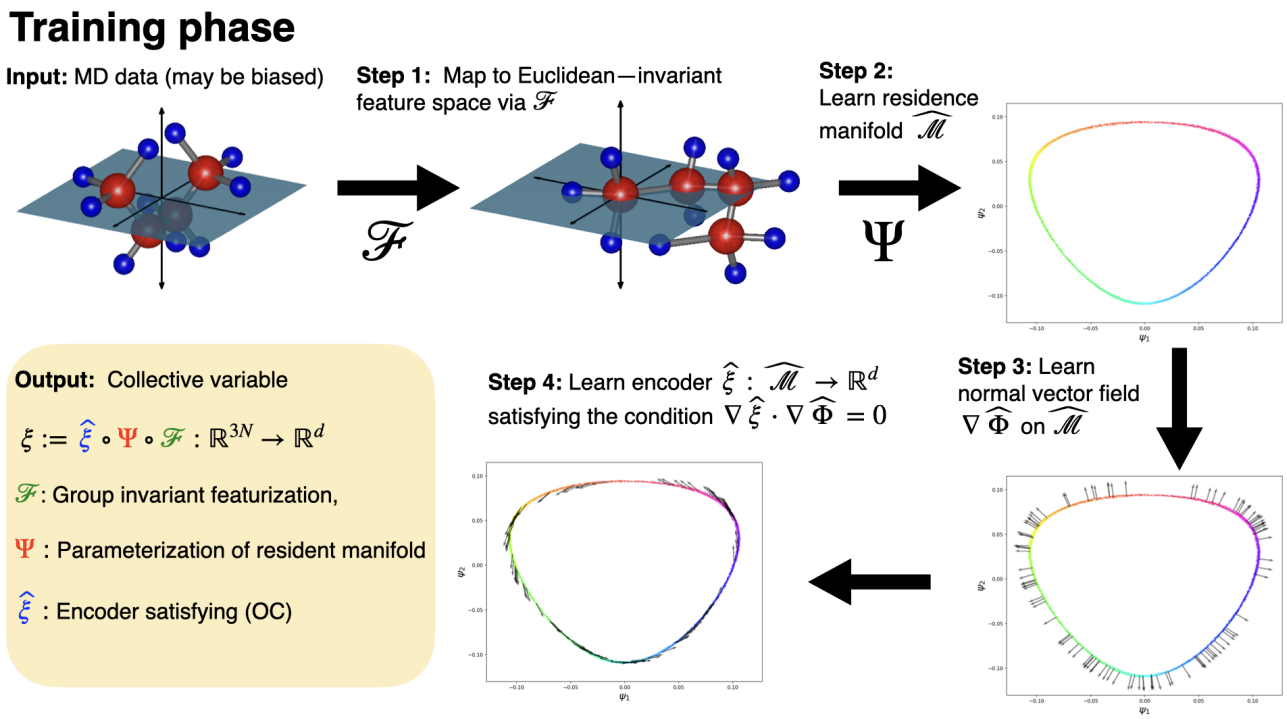


Figure 3.2: Algorithm 3 illustrated on butane: The raw MD simulation data is first featurized to remove Euclidean invariances. From this featurized data, a parameterization is learned to a low-dimensional space where the data is represented as the surrogate manifold. We learn a normal vector to this manifold and training an autoencoder whose encoder gradients are perpendicular to this learned normal vector field.

driven CVs using classical unsupervised learning methods such as diffusion maps [RZMC11, ZRC13, PC14], PCA (and variants) [RA07, GZ07], LDA [SMH23], IsoMap [SK11, ZVR+13], and tICA [MSP17, SP15, SG21] or its generalization, the variational approach for conformational (VAC) dynamics [PHPG+13, NN13, NKPH+14]. With the burgeoning ability to train highly nonlinear deep models, deep learning methods for generalizing the supervised learning methods mentioned above such as VAMPnets [CM19, MPWN18, BPP21] and informational bottlenecked techniques such as time-lagged autoencoders [WN18, ZS23, CF18] or variational autoencoders [WT21, WQB+24, RBWT18, HWSS+18] have emerged. The present chapter sits at the intersection of theory and application in the sense that it realizes the quantitative coarse-graining theory into an algorithm applied to an MD system. A closely related work focused on Lennard-Jones clusters, i.e., systems with permutational

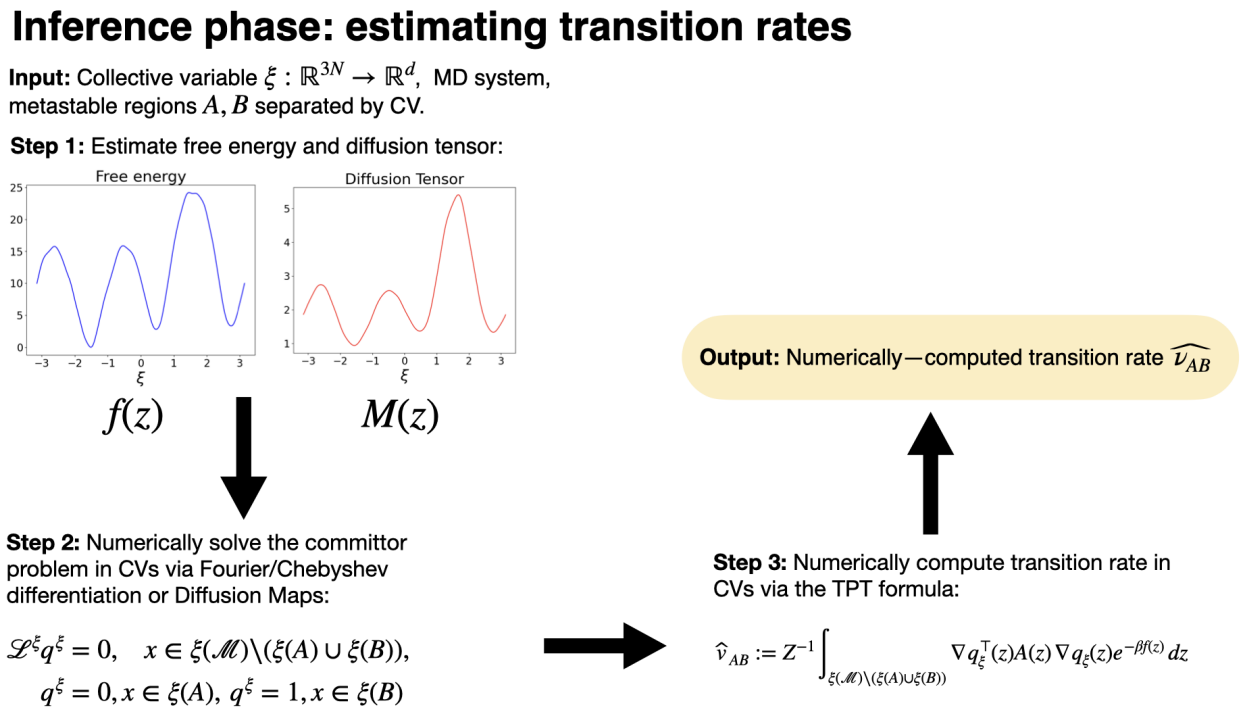


Figure 3.3: At inference time, we estimate the free energy surface (3.17) of the CV and the diffusion coefficients (3.5). Using these, we discretize the committor problem (3.42) and estimate the transition rate using the TPT formulas in [EVE06].

symmetry, is the recent paper [ZS24].

3.1.3 Organization

The rest of this chapter is organized as follows. In Section 3.2, we review the theory of quantitative coarse-graining from [LL10, DLP+18, LZ19b] where we state relative entropy and pathwise distance estimates for the discrepancy between the projected dynamics and the effective dynamics. In Section 3.3, we study two conditions emerging from this theory, the orthogonality condition (OC) and the uniform positive definiteness of $D\xi$ (Assumption 3.2.2). We also propose a numerical method for designing neural network-based CVs. The following two sections are more experimental and concern the application of our algorithm to butane. In particular, we describe how to learn the surrogate mani-

fold $\widehat{\mathcal{M}}$ in Section 3.4 and we learn the CV and present results for transition rate estimation in Section 3.5.

3.2 A brief overview of quantitative coarse-graining

3.2.1 Collective Variables

In this section, we review results from [LL10, DLP+18, LZ19a] to describe how (OC) emerges as a means to optimize coarse-graining error between Z_t (3.3) and Y_t (3.1). To begin, we state some general regularity conditions which ensure the existence of the invariant measure μ and unique solutions to the all-atom dynamics (1.51) and the effective dynamics (3.3):

Assumption 3.2.1. *We assume the potential V satisfies:*

$$V \in C^\infty(\mathbb{R}^N) \text{ almost everywhere}, \quad \exp(-\beta V) \in L^1(\mathbb{R}^N). \tag{3.8}$$

Moreover, we assume that the effective drift (3.4) *and the effective diffusion* (3.5) *are Lipschitz:*

$$\exists\, L_M, L_b > 0 \text{ such that } |b(z) - b(z')| \leq L_b|z - z'|, \quad \|M(z) - M(z')\|_F \leq L_M|z - z'|. \tag{3.9}$$

Conditions (3.8) and (3.9) ensure that both X_t and Z_t exist and are unique. We note here that [DLP+18] gives a more technical assumption for the CV ξ (see [DLP+18, Eq. (C3)]). However, insofar as we are concerned with relative entropy estimates, this assumption is only used for proving that the effective dynamics have Lipschitz coefficients. Therefore, by assuming the above Lipschitzness, we can remove this more technical assumption appearing in [DLP+18]. A more consequential condition

for studying the effective dynamics is the boundedness of the singular values of $D\xi D\xi^\top$:

Assumption 3.2.2. *We assume that $\xi \in C^3(\mathbb{R}^N)$. Moreover, there exists $\delta_\xi > 0$ such that the collective variable ξ satisfies:*

$$D\xi D\xi^\top \geq \delta_\xi^2 I_d, \tag{3.10}$$

where I_d is a $d \times d$ identity matrix.

Given Assumption 3.2.2, the conditioning approach may be further clarified through the use of an integral formulation. First, we introduce the level set $\Omega_z := \{\xi = z\}$. Since Assumption 3.2.2 ensures that z is a regular point of ξ, by the implicit function theorem, Ω_z is a smooth manifold of codimension d whose cotangent space is spanned by the rows of the Jacobian $D\xi$. In particular, the level sets $(\Omega_z)_{z\in\mathbb{R}^d}$ *foliate* $\mathbb{R}^N$ via:

$$\mathbb{R}^N = \bigsqcup_{z\in\mathbb{R}^d} \Omega_z. \tag{3.11}$$

Here $\sqcup$ denotes a disjoint union. Consequently, the *co-area formula* [EG92] can be used to break up the integral on $\mathbb{R}^N$ into integrals on Ω_z. For instance, if $\varphi : \mathbb{R}^N \to \mathbb{R}$ is bounded and continuous then

$$\int_{\mathbb{R}^N} \varphi(x)\mu(x)\,dx = \int_{\mathbb{R}^d}\int_{\Omega_z} \frac{\varphi(x)\mu(x)}{\det(D\xi D\xi^\top)} d\sigma_{\Omega_z}(x)\,dz. \tag{3.12}$$

Note that if $X \sim \mu$ is a random variable, then so is $Z = \xi(X)$. Then, (3.12) is a geometric description

of the tower law

$$\mathbb{E}_\mu[\varphi(X)] = \mathbb{E}\left[\mathbb{E}\left[\varphi(X) \mid Z\right]\right]. \tag{3.13}$$

The manifold Ω_z can then be used to describe the probability densities of random variables after conditioning on the values of ξ. In particular, if X is distributed by the invariant measure $\mu = Z^{-1}\exp(-\beta V(x))$ then $\xi(X) \in \mathbb{R}^d$ is distributed according to the measure μ^ξ given by

$$\mu^\xi(z)dz = Z^{-1}\int_{\Omega_z} \frac{\exp\left(-\beta V(x)\right)}{\sqrt{\det(D\xi D\xi^\top(x))}}\, d\sigma_{\Omega_z}(x)\, dz. \tag{3.14}$$

Therefore, we can define a new measure on the surface Ω_z given by μ_z:

$$\mu_z(x)\, dx := N_z^{-1}\frac{Z^{-1}\exp\left(-\beta V(x)\right)}{\sqrt{\det(D\xi D\xi^\top(x))}}\, d\sigma_{\Omega_z}(x), \tag{3.15}$$

$$N_z = Z^{-1}\int_{\Omega_z} \frac{\exp\left(-\beta V(x)\right)}{\sqrt{\det(D\xi D\xi^\top(x))}}\, d\sigma_{\Omega_z}(x). \tag{3.16}$$

The normalization constant N_z is used to define the *free energy*:

$$f_\mu(z) = -\beta^{-1}\log N_z. \tag{3.17}$$

The connection between between Equations (3.14) and (3.17) is $\mu^\xi(z) = N_z = \exp\left(-\beta f(z)\right)$. By expressing the free energy as an integral we can also write the derivative of the free energy as an integral. This is formulated in [LL10, Lemma 2.2] and [DLP+18, Lemma 2.4]:

Lemma 3.2.3. *The derivative of the free energy $\nabla_z f_\mu(z)$, termed the* mean force *is given by*

$$\nabla_z f_\mu(z) = \int_{\Omega_z} F_z(x)\, dx, \tag{3.18}$$

$$F_z(x) = (D\xi D\xi)^{-1} D\xi \nabla V - \beta^{-1} \nabla \cdot ((D\xi D\xi^\top)^{-1} D\xi). \tag{3.19}$$

Here $\nabla\cdot$ is the divergence operator and is applied row-wise when applied to a matrix.

Lemma 3.2.3 reveals that the mean force can itself be represented as an integral where the integrand F is the *local mean force*. As we shall see, the local mean force will play a crucial role in designing a strategy for choosing a collective variable.

3.2.2 Measuring the quality of effective dynamics

Many questions around the quality of the CV ξ boil down to measuring how close the effective dynamics process Z_t is to the coarse-grained process Y_t. As such, there are many ways to quantify the proximity between two stochastic processes, but two particularly fruitful criteria, *relative entropy* and *pathwise convergence*, were given in [LL10, DLP+18, LZ19b]. We first discuss the relative entropy criterion.

3.2.2.1 Relative entropy

Definition 3.2.4 (Relative entropy)**.** *Let η', η be two probability measures on $\mathbb{R}^d$ which are absolutely continuous with respect to the Lebesgue measure. Then the relative entropy $D_{KL}(\eta', \eta)$ is given as*

follows:

$$D_{KL}(\eta', \eta) := \begin{cases} \int_{\mathbb{R}^d} \ln\left(\frac{d\eta'}{d\eta}\right) d\eta', & \text{if} \quad \eta' \ll \eta, \\ +\infty & \text{otherwise}. \end{cases} \tag{3.20}$$

The relative entropy is related to measuring distances between distributions due to the Csiszar-Kullback inequality:

Proposition 3.2.5 (Cziszar-Kullback inequality)**.** *Let* $\|\eta' - \eta\|_{TV}$ *be the* total variation *distance between the probability measures* η' *and* η *on the* σ*-algebra* $\mathcal{G}$*. Then*

$$\|\eta' - \eta\|_{TV} := \sup_{A \in \mathcal{G}} |\eta(A) - \eta'(A)| \leq \sqrt{2 D_{KL}(\eta', \eta')}. \tag{3.21}$$

The Cziszar-Kullback inequality shows that reducing relative entropy between distributions is a bona fide way of making two distributions close to each other in the TV norm. This was the content of the main result [LL10] which showed how to reduce the relative entropy between the $\mathcal{P}(Y_t)$ and $\mathcal{P}(Z_t)$ distributions on $\mathbb{R}^d$ induced by Y_t and Z_t. In [DLP$^+$18], this result was generalized for a d-dimensional collective variable where [DLP$^+$18] imposed a few additional assumptions:

Assumption 3.2.6. *There exists* $\rho_{LSI} > 0$ *such that for every probability measure* ν *satisfying* $\nu \ll \mu_z$

$$D_{KL}(\nu, \mu_z) \leq \frac{1}{2\rho_{LSI}} \int_{\Omega_z} \left\| \nabla \ln\left(\frac{d\nu}{d\mu_z}\right) \right\|^2 d\nu. \tag{3.22}$$

The above inequality is the log-Sobolev *inequality and* ρ_{LSI} *is the log-Sobolev constant.*

The log-Sobolev constant determines how quickly a Markov chain mixes to its invariant distribution. In particular, if $\partial_t \mu_t = \mathcal{L}^\dagger \mu_t$ is the Fokker-Planck equation and $\mathcal{L}^\dagger \mu = 0$ is the stationary

solution, then by Boltzmann's H-theorem (see e.g. [VW19, Lemma 2]), we have

$$D_{\mathrm{KL}}(\mu_t, \mu) \leq \exp(-2\rho_{\mathrm{LSI}} t) D_{\mathrm{KL}}(\mu_0, \mu). \tag{3.23}$$

If ρ_{LSI} is large, then the associated diffusion process on μ_z mixes quickly. Since Assumption 3.2.6 stipulates that every $\mu_z \in \mathbb{R}^d$ shares a single ρ_{LSI}, this implies that processes on Ω_z mix *uniformly quickly*. Next, we turn to an assumption regarding the local mean force F_z in (3.3). We can endow Ω_z with a geodesic distance:

$$d_z(x_1, x_2) := \inf \left\{ \int_0^1 \|\gamma'(t)\| \, dt \mid \gamma(0) = x_1, \gamma(1) = x_2, \gamma(t) \in \Omega_z \right\}. \tag{3.24}$$

Assumption 3.2.7. *We assume that the local mean force is κ-Lipschitz from Ω_z to $\mathbb{R}^d$ as measured by the domain metric d_z and the range metric $\|x - y\|_M^2 = (x-y)^\top M (x-y)$:*

$$\kappa := \sup_{z \in \mathbb{R}^d} \sup_{x_1, x_2 \in \Omega_z} \frac{\|F_z(x_1) - F_z(x_2)\|_{M(z)}}{d_z(x_1, x_2)} < +\infty, \tag{3.25}$$

where F_z is the local mean force given by (3.19).

The last assumption relates to the variation of $D\xi D\xi^\top$ from its mean $M(z)$ (3.5) on each level set Ω_z.

Assumption 3.2.8. *Since $D\xi$ is smooth, its restriction to Ω_z is well-defined and is measurable. Let $\lambda_z : \Omega_z \to \mathbb{R}$ be given by:*

$$\lambda_z(x) = \|M(z)^{-1/2}(M(z) - D\xi(x)D\xi(x)^\top)(D\xi(x)D\xi(x)^\top)\|_{2\to 2} \tag{3.26}$$

Here $\|\cdot\|_{2\to 2}$ is the $L^2 \to L^2$ matrix norm of the given matrix; for a symmetric matrix this is the largest squared eigenvalue. Now assume that

$$\lambda_\xi := \| \|\lambda_z(x)\|_{L^\infty(\Omega_z)} \|_{L^\infty(\mathbb{R}^d)} < +\infty. \tag{3.27}$$

The constant λ_ξ, in some sense, measures how far ξ is from being an affine function. If, for instance, $\xi = Ax + b$ then $\lambda_\xi = 0$ because $D\xi = A$ and $M(z) = AA^\top$.

[LL10] proved the following result which was later generalized by [DLP+18]:

Theorem 3.2.9. *Let $\mathcal{P}(Z_t), \mathcal{P}(Z_t)$ denote the distributions of Z_t, Y_t and suppose that Z_0 and Y_0 are identically distributed. Under the Assumptions 3.2.1, 3.2.2, 3.2.6, 3.2.7, and 3.2.8,*

$$D_{KL}(\mathcal{P}(Z_t), \mathcal{P}(Y_t)) \leq \frac{1}{4}\left(\lambda_\xi + \frac{\kappa^2\beta^2}{\rho_{LSI}^2}\right)\left(D_{KL}(\mathcal{P}(X_0), \mu) - D_{KL}(\mathcal{P}(X_t), \mu)\right). \tag{3.28}$$

If $D\xi$ is constant on the level set Ω_z then $\lambda_\xi = 0$.

The estimate (3.28) was referred to as an "intermediate time" estimate in [LL10]. The reason is as follows: as $t \to \infty$, it was shown that the effective and projected dynamics have the same invariant measure given by $\xi_\# \mu$, the pushforward of μ by ξ. Thus, the left-hand side of (3.28) decays to zero while the right-hand side does not, so the bound is not sharp in the $t \to \infty$ limit. Moreover, if $t = 0$ then both sides of (3.28) vanish. Thus, the estimate (3.28) prescribes control on intermediate $t \in (0, \infty)$.

3.2.2.2 Reducing the relative entropy and the orthogonality condition

Suppose V satisfies the scale separation condition:

$$V = V_0 + \varepsilon^{-1} V_1^2. \tag{3.6}$$

In this case, [LL10] computed the local mean force in (3.19) explicitly [LL10, p. 2150]:

$$F = \frac{2}{\varepsilon} V_1 \frac{D\xi \nabla V_1}{\|D\xi\|_2^2} + \frac{D\xi \nabla V_0}{\|D\xi\|_2^2} - \beta^{-1} \nabla \cdot \left(\frac{D\xi^\top}{\|D\xi\|_2^2} \right). \tag{3.29}$$

From now on, we will omit the subscript z from the local mean force for brevity. Since $\varepsilon \ll 1$, for a general CV $F = O(1/\varepsilon)$ and therefore κ (3.25) will be $O(1/\varepsilon)$. However, if (OC):

$$D\xi \nabla V_1 = 0$$

is satisfied then $F = O(1)$. Moreover, in this case, ρ_{LSI} (3.22) becomes $O(1/\varepsilon)$ and therefore, the relative entropy estimate reduces to (see [LL10, p. 2150] or [DLP+18, Eq. (4.3)]):

$$D_{\mathrm{KL}}(\mathcal{P}(Z_t), \mathcal{P}(Y_t)) \leq \frac{1}{4} \left(\lambda_\xi + O(\varepsilon) \right) \left(D_{\mathrm{KL}}(\mathcal{P}(X_0), \mu) - D_{\mathrm{KL}}(\mathcal{P}(X_t), \mu) \right). \tag{3.30}$$

The above calculation shows how (OC) emerges as a way to make the error estimate (3.28) small and imparts the geometric requirement to the CV by stipulating that its level sets Ω_z must be orthogonal to those of the confining term V_1 as visualized in Figure 3.1. A similar numerical example was proposed in [LL10, Table 1] where it was shown that imposing the (OC) on a CV led to marked

improvements in the accuracy of residence times computed via the effective dynamics, suggesting that (OC) improves CV design.

Notably, the approach of using (OC) does not touch the constant λ_ξ appearing in the reduced error estimate (3.30). In fact, λ_ξ is seemingly the more significant contributor to the error since it makes the error estimate (3.30) behave as $O(1)$ irrespective of (OC). It has been noted in [LL10] and [DLP+18] that $\lambda_\xi = 0$ if (i) ξ is affine or, more generally, (ii) $D\xi$ is constant on Ω_z for every z. Both conditions (i) and (ii) are rather restrictive. Therefore, there seems to be a geometric trade-off in terms of reducing λ_ξ and κ simultaneously. In this chapter, guided by [LL10]'s example, we will focus on (OC) exclusively, leaving this fascinating trade-off as the subject of future investigation.

3.2.2.3 Pathwise estimates

Next, we discuss another criterion for measuring the distance between two stochastic processes: the *pathwise distance* defined as follows:

Definition 3.2.10. *Let A_t, B_t be two $\mathbb{R}^d$-valued stochastic processes defined on $t \in [0,T]$ and adapted to the same filtration $(\mathcal{F}_t)_{0\le t\le T}$. Then the pathwise distance between A_t and B_t is given by*

$$d(A_t, B_t) := \mathbb{E}\left[\sup_{t\in[0,T]} |A_t - B_t|\right]. \tag{3.31}$$

Note that two processes with zero pathwise distance induce, for a fixed t, the same distribution on $\mathbb{R}^d$ and therefore have zero relative entropy. The works [LL10, LLO17, LZ19b] describe the conditions under which the projected process (3.1) and effective dynamics (3.3) may be brought close in terms of this much stronger distance between stochastic processes. To set up this result, we define the projection

operator Π induced by the collective variable:

$$\Pi = I - \sum_{i,j} (\Phi^{-1})_{ij} \nabla \xi_i \nabla \xi_j^\top \tag{3.32}$$

where $\Phi = D\xi D\xi^\top$. With the projection operator Π, the infinitesimal generator (1.52) can be decomposed as

$$\mathcal{L} = \mathcal{L}_0 + \mathcal{L}_1 \tag{3.33}$$

where

$$\mathcal{L}_0 = \frac{e^{\beta V}}{\beta} \sum_{i,j} \delta_{ij} \frac{\partial}{\partial x_i} \left(e^{-\beta V} \Pi_{ij} \frac{\partial}{\partial x_j} \right), \tag{3.34}$$

$$\mathcal{L}_1 = \frac{e^{\beta V}}{\beta} \sum_{i,j} \delta_{ij} \frac{\partial}{\partial x_i} \left(e^{-\beta V} (I - \Pi)_{ij} \frac{\partial}{\partial x_j} \right). \tag{3.35}$$

Furthermore, we assume

$$\begin{aligned} \kappa_1^2 &:= \sum_i \int (\Pi \nabla \mathcal{L} \xi_i) \cdot (\Pi \nabla \mathcal{L} \xi_i) \, d\mu < \infty, \\ \kappa_2^2 &:= \sum_i \int (\Pi \nabla M_{ij}) \cdot (\Pi \nabla M_{ij}) d\mu < \infty. \end{aligned} \tag{3.36}$$

Next, we state the result of the pathwise distance between the effective dynamics the coarse-grained process below in Theorem 3.2.11:

Theorem 3.2.11. *[LZ19b, Theorem 1] There exists a constant $\gamma(V, \xi) > 0$ such that:*

$$d(Y_t, Z_t) \leq \frac{3t}{\beta\gamma} \left(\frac{27\kappa_1^2}{2\gamma} + \frac{32\kappa_2^2}{\beta} \right) e^{Lt} \tag{3.37}$$

where $L = 3L_b^2 + \frac{48L_M^2}{\beta} + 1$. *Here* L_M, L_b *are Lipschitz constants defined in Assumption 3.2.1.*

3.2.2.4 Reducing the pathwise distance in the presence of scale separation.

We consider the potential V with the scale separation in the form of equation (3.6). In [LZ19b] it was shown that the bound in (3.37) may be made sharper (and in particular $O(\varepsilon)$) if the collective variable ξ is chosen to satisfy the *projected orthogonality condition* (POC)

$$(I - \Pi)^\top \nabla V_1 \equiv 0. \tag{POC}$$

The (POC) concerns choosing the rows of the projection $(I - \Pi)$ to be orthogonal to ∇V_1. Under (POC) and the scale separation condition (3.6), Theorem 3.2.11 can be applied to show that:

$$d(Y_t, Z_t) \leq t \left(\frac{C_1 \varepsilon}{K} + \frac{C_2 \varepsilon^2}{K^2} \right) e^{Lt} \sim O(\varepsilon) \tag{3.38}$$

Additionally, under (POC) we note

$$\Pi^\top \nabla V = \Pi^\top \nabla V_0 + \frac{1}{\varepsilon} \nabla V_1 = O(1/\varepsilon), \quad (I - \Pi)^\top \nabla V = (I - \Pi)^\top \nabla V_0. \tag{3.39}$$

This implies that

$$\textit{Operator } \mathcal{L}_0 \textit{ contains large coefficients, while } \mathcal{L}_1 \textit{ does not.} \tag{3.40}$$

We term (3.40) the *coefficient condition* (CC) and remark that (POC) is a special case of (CC). Moreover, in [LZ19b] it was noted that a good choice of collective variable in the most routinely occurring

cases of scale separation falls under (CC).

3.2.2.5 Simulating the effective dynamics

CVs are essential for identifying and replicating the metastable dynamics of the all-atom dynamics (1.51). In particular, given ξ, we may identify metastable states $\tilde{A}, \tilde{B}$ typically corresponding to the wells of the free energy $f(z)$ (3.17). We then lift these metastable states to $\mathbb{R}^N$ by defining $A = \xi^{-1}(\tilde{A})$ and $B = \xi^{-1}(\tilde{B})$. A good CV should yield A and B such that the statistics of the rare transitions between A and B in all-atom space $\mathbb{R}^N$ are replicated by the effective dynamics in CV space $\mathbb{R}^d$. An important statistic is the transition rate ν_{AB} between A and B, defined in (1.57), which measures the per-unit time transitions between A and B. As summarized in Chapter 1 it can also be written in terms of the committor function q in (1.54) as follows:

$$\nu_{AB} = \lim_{T\to\infty} \frac{N_{AB}}{T} = \beta^{-1} Z^{-1} \int_{\Omega_{AB}} \|\nabla q(x)\|^2 \exp(-\beta V(x)) dx. \tag{1.57}$$

Assuming that ξ separates the metastable states A and B, the effective dynamics may be used to numerically calculate ν_{AB} by finding the corresponding transition rate $\tilde{\nu}_{AB}$ for the low-dimensional effective dynamics. Below we describe a numerical method for computing $\tilde{\nu}_{AB}$. Under Assumption 3.2.2, the effective dynamics (3.3) governing Z_t is equivalent to the following dynamics proposed by [MFVEC06]:

$$\widehat{Z}_t = -(M(\widehat{Z}_t)\nabla_z f_\mu(\widehat{Z}_t) + \beta^{-1}\nabla \cdot M(\widehat{Z}_t))\, dt + \sqrt{2\beta^{-1}} M(\widehat{Z}_t)^{1/2}\, dW_t. \tag{3.41}$$

The equivalence between Z_t and $\widehat{Z}_t$ has been noted in [LL10] and it makes Z_t easy to simulate by simulating $\widehat{Z}_t$ instead. This amounts to computing the diffusion tensor M and the free energy, which can be done using well-known numerical methods [MFVEC06, MZK+21, DRGP08, FP17, BBP08]. By utilizing M and f_μ, the committor $\tilde{q}$ in collective variables can be computed as the solution to the following elliptic boundary value problem:

$$\begin{cases} \beta^{-1} e^{\beta f_\mu(z)} \mathsf{div}(e^{-\beta f_\mu(z)} M(z) \nabla \tilde{q}(z)) = 0, & z \in \tilde{\Omega}_{AB}, \\ \tilde{q}(z) = 0, & z \in \tilde{A}, \\ \tilde{q}(z) = 1, & z \in \tilde{B}. \end{cases} \tag{3.42}$$

Here $\tilde{\Omega}_{AB} := \xi(\mathcal{M}) \setminus (\tilde{A} \cup \tilde{B})$. The transition rate $\tilde{\nu}_{AB}$ in the CV space is given by:

$$\tilde{\nu}_{AB} = \beta^{-1} Z_F^{-1} \int_{\tilde{\Omega}_{AB}} (\nabla \tilde{q})^\top M(z) \nabla \tilde{q} \exp\left(-\beta f(z)\right) dz. \tag{3.43}$$

Computing ν_{AB} involves solving the high-dimensional committor problem (1.54) followed by high-dimensional numerical integration of (1.57), which can be unfeasible or undesirable in practice. Instead, we can compute $\tilde{\nu}_{AB}$ by solving the low-dimensional BVP (3.42) and using the formula (3.43). The discrepancy between the original transition rate ν_{AB} and the coarse-grained transition rate $\tilde{\nu}_{AB}$ is [ZHS16]:

$$\nu_{AB} \leq \tilde{\nu}_{AB} = \nu_{AB} + \frac{1}{\beta} \int_{\Omega_{AB}} \nabla [q(x) - \tilde{q}(\xi(x))]^\top M(x) \nabla [q(x) - \tilde{q}(\xi(x))] \mu(x) dx. \tag{3.44}$$

Thus, coarse-graining by a collective variable ξ overestimates the transition rate unless $q(x) \equiv \tilde{q}(\xi(x))$. It was also noted in [ZHS16] that the high-dimensional committor q is the optimal CV. In fact, if $\xi = q$, then the effective dynamics reside in $[0, 1]$. Moreover, for $z \in (0, 1)$, the effective dynamics (3.3) become driftless with a diffusion coefficient $M(z) = \beta\nu_{AB} \exp(\beta f(z))$. As a consequence, the committor is given by $\tilde{q}(z) = z$. Plugging these into (3.43), we get $\tilde{\nu}_{AB} = \nu_{AB}$. Thus q reproduces reaction rates *exactly*. However, computing q accurately by solving the high-dimensional PDE (1.54) is computationally prohibitive. Therefore, different methods are required to design CVs that minimally distort transition rates.

3.2.2.6 Relation to full Langevin dynamics and reaction rate scaling

Typically, MD simulations track not just the positions X_t but also the velocities of atoms v_t (the lower case notation has been purposely chosen to not confuse with the potential V) under the *Langevin dynamics* given by

$$dX_t = v_t\, dt, \tag{3.45}$$

$$m\, dv_t = -\gamma m v_t\, dt - \nabla V(X_t)\, dt + \sqrt{2k_B T\gamma}\, m^{1/2} dW_t. \tag{3.46}$$

Here γ is the friction coefficient, T is the temperature, m is the diagonal matrix of masses of individual particles, and k_B is Boltzmann's constant used to make terms dimensionless. In the regime $\gamma \gg 1$, the Langevin dynamics (3.45), (3.46) may be approximated by the following anisotropic overdamped Langevin dynamics with $\beta^{-1} = k_B T$:

$$dX_t = -m^{-1}\nabla V(X_t)\gamma^{-1}\, dt + \sqrt{2\beta^{-1}\gamma^{-1}} m^{-1/2} dW_t. \tag{3.47}$$

In this case, $\mu \propto \exp(-\beta V)$ is still the invariant measure. To remove the implicit effect of friction, we must rescale time as $\tau = \gamma^{-1} t$. This results in a time-rescaled overdamped Langevin dynamics given by

$$dX_\tau = -m^{-1} \nabla V(X_\tau)\, d\tau + \sqrt{2\beta^{-1}} m^{-1/2} dW_\tau. \tag{3.48}$$

Under the overdamped approximation, the dynamics are anisotropic due to the involvement of the mass matrix m. In this case, the diffusion tensor must feature this mass rescaling [Cam13, Eq. 5] and thus be given by:

$$M(z) := \mathbb{E}_\mu[(D\xi) m^{-1} (D\xi)^\top \mid \xi = z]. \tag{3.49}$$

With this diffusion tensor, the rate formula (3.43) applies for computing the transition rate ν'_{AB}. However, this transition rate is for transitions occurring in the timescale τ; thus to recover the timescale t, we must rescale by the relevant factor of γ. This is pertinent when computing transition rates. For example, as a heuristic, the transition rate ν'_{AB} computed for the dynamics (3.48) would be under the units $1/\tau = \gamma/t \iff t^{-1} = \gamma^{-1}\tau^{-1}$. Therefore, the transition rate ν_{AB} in the units t^{-1} (for instance, in ps^{-1}) may be recovered as

$$\nu_{AB} \approx \gamma^{-1} \nu'_{AB}. \tag{3.50}$$

However, the reaction rate is not the only place where the friction coefficient must be accounted for, because the diffusion tensor M is calculated in practice also with a trajectory X_t indexed by time. This is due to the following reason: given z the string method [MFVEC06] for computing the diffusion

matrix $M(z)$ modifies the potential V with a harmonic restraint with stiffness κ_{String}:

$$V^z(x) := V(x) + \frac{\kappa_{\text{String}}}{2}\|z - \xi(x)\|_2^2. \tag{3.51}$$

With the potential V^z, the process X_t^z is simulated according to the Langevin dynamics (3.45)-(3.46). Then the diffusion tensor is given by:

$$[M(z)]_{jk} := \lim_{T\to\infty} \frac{1}{T}\int_0^T \sum_{i=1}^{3N} m_i^{-1} \frac{\partial \xi_j(X_t^z)}{\partial x_i}\frac{\partial \xi_k(X_t^z)}{\partial x_i}\, dt \tag{3.52}$$

$$= \lim_{T\to\infty} \frac{1}{T}\int_0^T \sum_{i=1}^{3N} m_i^{-1} \frac{\partial \xi_j(X_\tau^z)}{\partial x_i}\frac{\partial \xi_k(X_\tau^z)}{\partial x_i}\, \gamma\, d\tau \tag{3.53}$$

$$\approx \frac{\gamma}{n}\sum_{\ell=1}^{n}\sum_{i=1}^{3N} m_i^{-1} \frac{\partial \xi_j(X_{\ell\Delta\tau}^z)}{\partial x_i}\frac{\partial \xi_k(X_{\ell\Delta\tau}^z)}{\partial x_i} \tag{3.54}$$

$$= \frac{\gamma}{n}\sum_{\ell=1}^{n}\sum_{i=1}^{3N} m_i^{-1} \frac{\partial \xi_j(X_{\ell\gamma^{-1}\Delta t}^z)}{\partial x_i}\frac{\partial \xi_k(X_{\ell\gamma^{-1}\Delta t}^z)}{\partial x_i} \tag{3.55}$$

The third line following the second is the essence of the string method. Here Δt is the time step used for simulating Langevin dynamics–the above calculation suggests that it needs to be additionally smaller to scale with the friction coefficient.

3.3 Proposed methodology for learning CVs

The theory of quantitative coarse-graining reviewed above provides guidelines for CV design through analytical conditions for ξ under which the error estimates (3.28) and (3.31) may hold and be minimized. Two such conditions are given by Assumption 3.2.2 which stipulates that $D\xi D\xi^\top$ is uniformly strictly positive definite and, (OC) and (POC) which make error estimates scale with ε in the potentials of the form $V = V_0 + \varepsilon^{-1}V_1^2$, as in (3.6). In this section, we:

1. prove that under the explicit scale separation condition $V = V_0 + \varepsilon^{-1}V_1^2$, (OC) and (POC) are equivalent,

2. numerically demonstrate that Assumption 3.2.2 is not necessary for the reproduction of transition rates and may be relaxed to the requirement that $\|D\xi\|$ is bounded away from zero, and

3. propose an algorithm based on (OC) for designing CVs for MD data.

3.3.1 Conditions (OC) and (POC) are equivalent

The theory on pathwise estimates and relative entropy provides two conditions, (OC) and (POC), which may be used to improve the error estimates for effective dynamics and therefore to learn a collective variable. We now show that (OC) and (POC) are actually *equivalent* and therefore the same condition provides simultaneous improvements in two different error metrics.

Proposition 3.3.1. *Let ξ be a collective variable and V be given by the decomposition $V = V_0 + \varepsilon^{-1}V_1$ as in (3.6). Then (OC) and (POC) are equivalent.*

Proof. Let $J := D\xi(x)$. Recall that

$$\Phi = \left(JJ^\top\right)^{-1}.$$

Then the entries of $I - \Pi$ are:

$$[I - \Pi]_{lk} = \sum_{i,j} (\Phi^{-1})_{ij} (\nabla\xi_i)_l (\nabla\xi_j)_k = J_l^\top \Phi^{-1} J_k, \tag{3.56}$$

where J_k and J_l are the kth and lth rows of J. Now, let $J = U_r \Lambda_r V_r^\top$ be the truncated SVD of J where

r is the rank of J. Then $\Phi = \left(JJ^\top\right)^{-1} = U_r(\Lambda_r\Lambda_r^\top)^{-1}U_r^\top$. Moreover, note that we can write each column of J in terms of the columns of U with coefficients from the rows of V. The kth column of J is therefore:

$$J_k = \sum_{\alpha=1}^{r} \sigma_\alpha v_{\alpha k} u_\alpha. \tag{3.57}$$

Now we compute:

$$\begin{aligned} \Phi^{-1}J_k &= U_r(\Lambda_r\Lambda_r^\top)^{-1}U_r^\top \sum_{\alpha=1}^{r} \sigma_\alpha v_{\alpha k} u_\alpha \\ &= \sum_{\alpha=1}^{r} \sigma_\alpha v_{\alpha k} U_r(\Lambda_r\Lambda_r^\top)^{-1}U_r^\top u_\alpha \\ &= \sum_{\alpha=1}^{r} \sigma_\alpha v_{\alpha k} u_\alpha \sigma_\alpha^{-2} = \sum_{\alpha=1}^{r} \sigma_\alpha^{-1} v_{\alpha k} u_\alpha. \end{aligned} \tag{3.58}$$

Then $[I - \Pi]_{lk}$ is given by:

$$\begin{aligned} [I-\Pi]_{lk} = J_l^\top \Phi^{-1} J_k &= \left(\sum_{\beta=1}^{r} \sigma_\beta v_{\beta l} u_\beta^\top \sum_{\alpha=1}^{r} \sigma_\alpha^{-1} v_{\alpha k} u_\alpha\right) \\ &= \left(\sum_{1\le\alpha,\beta\le r} \sigma_\beta v_{\beta l}\sigma_\alpha^{-1} v_{\alpha k} u_\beta^\top u_\alpha\right) = \left(\sum_{1\le\alpha\le r} \sigma_\alpha v_{\alpha l}\sigma_\alpha^{-1} v_{\alpha k}\right) \\ &= \sum_{1\le\alpha\le r} v_{\alpha l} v_{\alpha k} = \left[V_r V_r^\top\right]_{lk}. \end{aligned} \tag{3.59}$$

Thus, $[I - \Pi]_{lk} = \left[V_r V_r^\top\right]_{lk}$ so $I - \Pi = V_r V_r^\top$.

Now recall that (OC) states

$$D\xi(x)\nabla V_1(x) = J\nabla V_1 = U_r\Lambda_r V_r^\top \nabla V_1 = 0.$$

Since columns of $U_r\Lambda_r$ are linearly independent, the last equation is equivalent to

$$V_r^\top \nabla V_1 = 0.$$

This, in turn, by the linear independence of columns of V_r, is equivalent to $[I - \Pi]\nabla V_1 = V_r V_r^\top \nabla V_1 = 0$. Thus (OC) is equivalent to (POC). □

3.3.2 The diffusion tensor may be rank-deficient but must not vanish

Assumption 3.2.2, stating that $D\xi D\xi^\top \geq \delta_\xi^2 I_d$ for a positive constant δ_ξ^2, ensures that the characterization of the conditional expectations through the co-area formula (3.12) and therefore the measure $\mu^\xi \propto \exp(-\beta f(z))$ is an invariant measure for the effective dynamics (3.3) which is equivalent to (3.41) under Assumption 3.2.2. However, in the absence of Assumption 3.2.2, the diffusion tensor $M(z)$ may vanish, leading to the loss of ergodicity in the free energy dynamics (3.41). We describe such a case for butane below.

3.3.2.1 A chain-like butane molecule

The all-atom dynamics of the butane molecule C_4H_{10} in a solvent under high friction can be modeled using (1.51) with atomic coordinates X_t residing in $\mathbb{R}^{14\times 3}$ (see Figure 3.4). Specifically, the process $\theta(X_t)$ largely resides in the anti ($\theta = \pi$) or gauche ($\theta = \pi/3, 5\pi/3$) states (Figure 3.5) with rare transitions between them. The anti-gauche transition has therefore received considerable attention in molecular dynamics as a small-scale but significant example of torsional transitions and has served as a benchmark example for algorithms seeking to study conformational dynamics in larger biomolecules such as polymer chains. In practice, butane can be coarse-grained along the *dihedral*

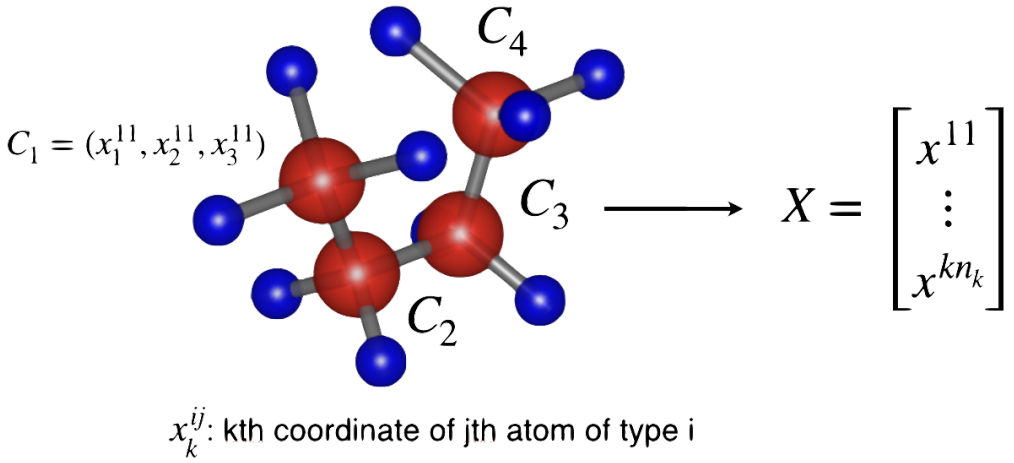


Figure 3.4: The butane molecule C_4H_{10} can be described in atomic coordinates in $\mathbb{R}^{42}$.

angle θ in its carbon backbone, where it exhibits metastability. We simulate butane at 300K under Langevin dynamics with high friction, resulting in dynamics well-approximated by the overdamped regime (1.51). We provide additional details on this approximation in Section 3.2.2.6 where we also describe how the high friction constant should be accommodated for in computing the transition rate via Equation (3.43). The anti-gauche transition rate obtained via brute-force all-atom simulation is $\nu_{AB} = 1.13 \pm 0.08 \times 10^{-2}$ ps^{-1}. Now we consider the following three collective variables based on the dihedral angle θ for reproducing the anti-gauche transition rate (Table 3.1):

1. $\xi_1 = \theta(x)$: The collective variable ξ_1 reproduces the rate as 1.41×10^{-2} ps^{-1}, within 3.5 standard deviations of the reference rate. Notably, the dihedral angle violates Assumption 3.2.2 since it is discontinuous at the point where $\theta = 0$ (i.e the location of the branch cut chosen for the angle).

2. $\xi_2 = \cos\theta(x)$: For the one-dimensional CV ξ_2 resulted in a transition rate of nearly 4.8 standard deviations of the reference rate, at 1.52×10^{-2} ps^{-1}. We posit that this exageration occurs because both the free energy F_2 and one-dimensional diffusion tensorA_2 given by equations (3.17) and (3.5) vanish (see central panel, Figure 3.6) leading to both the drift and diffusion terms

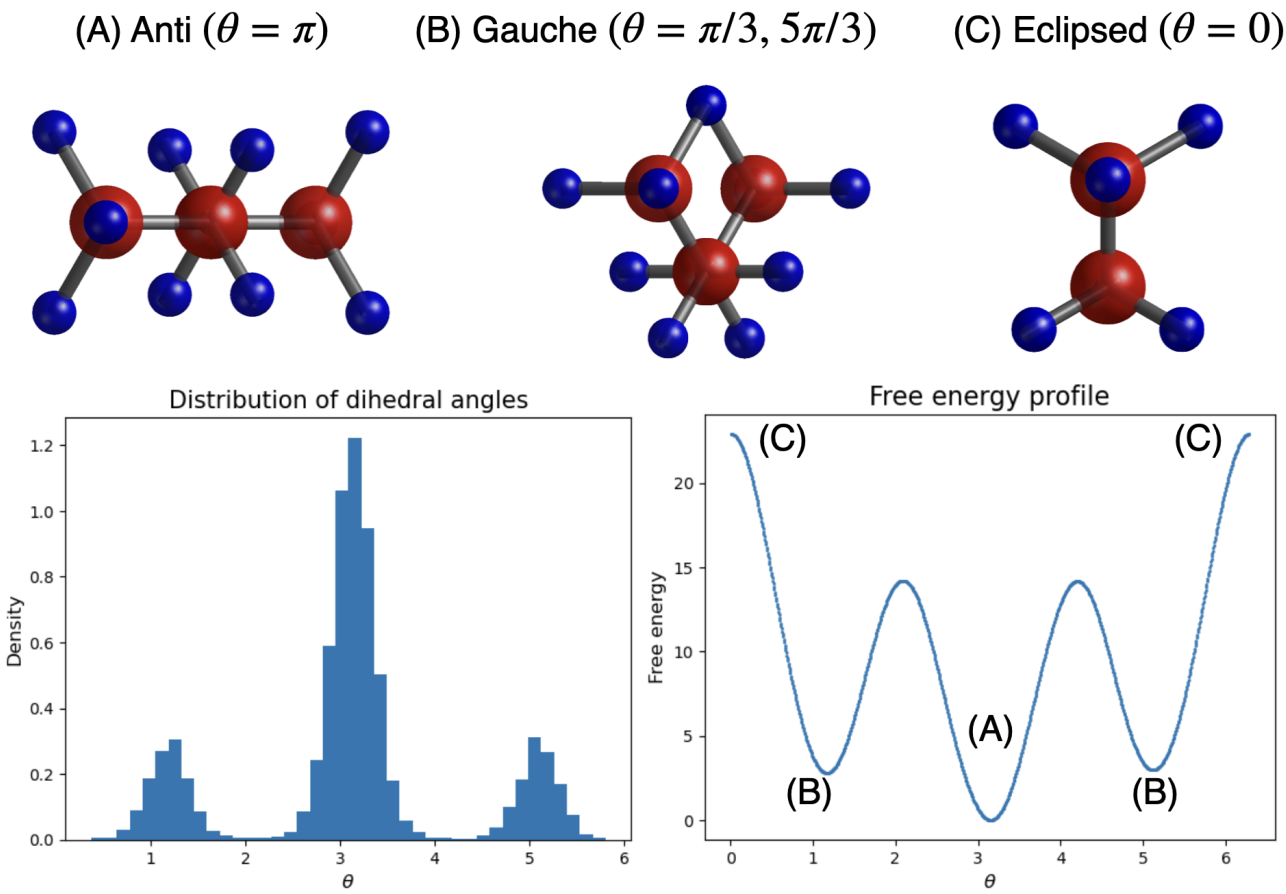


Figure 3.5: The dynamics of butane can be effectively coarse-grained by the dihedral angle θ. In this case, the free energy f has local minima near the anti and gauche states. Here, the system has been simulated at a temperature of 300 K.

in the free-energy dynamics (3.41) to vanish. As a consequence, the dynamics are not irreducible and therefore non-ergodic. Moreover, the chosen gauche configurations form a disjoint set in the atomic dynamics but the CV ξ_2 merges these configurations into one state in the one-dimensional space.

3. $\xi_3 = (\sin\theta(x), \cos\theta(x))$: We observe that for the two-dimensional CV ξ_3, the diffusion tensor A_3 has rank 1 (see right panel, Figure 3.6), but the transition rate is reproduced faithfully, at 1.19 ps^{-1}, less than one standard deviation of the reference. This illustrates that Assumption 3.2.2 is *not necessary* for a CV to reproduce transition rates. Moreover, ξ_3 preserves the disjoint gauche

configurations in CV space, unlike ξ_2.

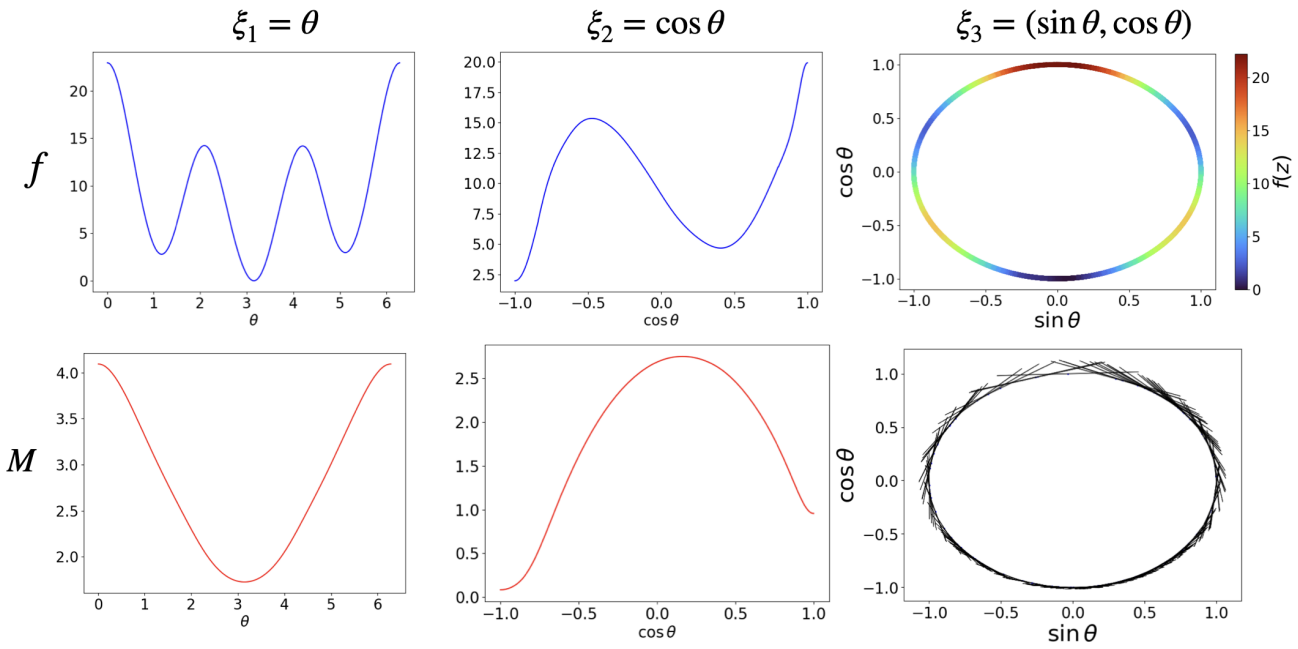


Figure 3.6: Free energy profiles (top) and diffusion tensors (bottom) for the collective variables based on the dihedral angle (left to right). Bottom right: Since the diffusion tensor for the collective variable $\xi = (\sin\theta, \cos\theta)$ is always rank-1, we visualize it as a line segment pointing in the direction of the non-zero eigenvector of M of length given by its non-zero eigenvalue.

Collective variable	Transition rate (ps^{-1})
Reference	$1.13 \pm 0.08 \times 10^{-2}$
Dihedral angle θ	1.41×10^{-2}
$(\sin\theta, \cos\theta)$	1.19×10^{-2}
$\cos\theta$	1.52×10^{-2}

Table 3.1: Collective variables for butane based on the dihedral angle, and the anti-gauche transition rates they yield.

3.3.3 Proposed algorithm

In this section, we propose an algorithm for machine-learning CV ξ based on the orthogonality condition (OC). Cautioned from the example in Section 3.3.2, we aim to ensure that ξ not only preserves metastable states but also observes the condition that $D\xi \neq 0$ for any $x \in \mathcal{M}$. Among the many ways to ensure the non-existence of critical points, a proven effective strategy is to choose ξ to be the encoder in an *autoencoder* [BBM+23]. Additionally, to train such a CV satisfying OC, we would need knowledge of ∇V_1 (the gradient of the confining potential), i.e, the normal vector to the residence manifold $\mathcal{M}$. However, the key obstruction to this approach is that a priori neither the confining term V_1 in (3.6) nor the residence manifold $\mathcal{M}$ are known. In this chapter, we will resolve this obstruction by first obtaining a parameterization $\Psi : \mathbb{R}^N \to \mathbb{R}^{D+1}$ which projects $\mathcal{M}$ into a D dimensional hypersurface $\widehat{\mathcal{M}} := \Psi(\mathcal{M})$. $\widehat{\mathcal{M}}$ will therefore serve as a surrogate for $\mathcal{M}$ and its property as a hypersurface enables learning a surrogate potential $\widehat{\Phi}$ vanishing on $\widehat{\mathcal{M}}$ with gradients aligned normally to the $\widehat{\mathcal{M}}$. Finally, we learn $\widehat{\xi}$ which satisfies the (OC) $\nabla\widehat{\xi} \cdot \nabla\widehat{\Phi} = 0$. Notably, this $\widehat{\xi}$ can be *lifted* to an overall collective variable via composition with the parameterization, given by $\xi := \widehat{\xi} \circ \Psi$. Therefore, our method can be described as recovering a CV by enforcing the orthogonality condition in a latent space $\mathbb{R}^{D+1}$ instead of the ambient space $\mathbb{R}^N$. We summarize our method in Algorithm 3 and elaborate on each step of this method in further sections below.

We remark that Step 1, which involves computing the low-dimensional embedding, is the most technically intricate subroutine of this algorithm. We summarize the subroutine in Algorithm 7 and provide a detailed walkthrough in the section below.

Algorithm 3 Learning CV that satisfies the orthogonality condition (OC)

Require: Data $\mathcal{X} = \{x_i\}_{i=1}^n \subseteq \mathcal{M}$, invariant density $\mu_i \propto \exp(-\beta V(x_i))$.

Step 1: Learn the residence manifold $\mathcal{M}$ as $\widehat{\mathcal{M}} \subseteq \mathbb{R}^{D+1}$ via a *diffusion map* [CL06] accompanied with a *diffusion net* [MSCC19] learned through Algorithm 7

$$\Psi : \mathcal{M} \to \widehat{\mathcal{M}} \subseteq \mathbb{R}^{D+1}$$

.

Step 2: Learn the surrogate potential: Train a neural network

$$\widehat{\Phi} : \mathbb{R}^{D+1} \to \mathbb{R}$$

such that $\widehat{\Phi}$ vanishes on $\widehat{\mathcal{M}}$.

Step 3: Train an autoencoder where the encoder $\widehat{\xi} : \mathbb{R}^{D+1} \to \mathbb{R}$ satisfies

$$\nabla\widehat{\xi} \cdot \nabla\widehat{\Phi} = 0$$

.

Ensure: Collective variable $\xi = \widehat{\xi} \circ \Psi$.

3.4 Learning the residence manifold

Our goal is to learn a function $\Psi : \mathbb{R}^N \to \mathbb{R}^{D+1}$ such that $\mathcal{M}$ may be represented by the surrogate manifold $\widehat{\mathcal{M}} = \Psi(\mathcal{M})$. We will learn this map from a trajectory $\mathcal{X} = \{x_i\}_{i=1}^n$ where $x_i \sim X_{i\Delta T}$, i.e the ith point of a time-series simulation of (1.51). We will adopt diffusion maps [CKL+08] to learn this map Ψ, but this cannot be done straightforwardly on data sampled from the process X_t due to a number of obstacles arising from the nature of MD simulations and the requirements of our proposed Algorithm 3. We highlight the key obstacles below:

1. *Group invariance:* In typical MD simulations, the confining potential V_1 is a combination of several strong pairwise energies such as covalent interactions between atoms and is therefore invariant to actions of SE(3), the group of global translations and rotations of $\mathbb{R}^3$ (see Definition 1.1.10). Therefore, if $X \in \mathcal{M}$, then $gX \in \mathcal{M}$ where g is a representation of an element in SE(3). Taking the action over all group elements, the orbit of any point x under SE(3) actions forms a submanifold of $\mathcal{M}$. But this is not the underlying slow direction we would like to discover.

2. *Hypersurface learning:* Since our goal is to learn a normal vector field on $\Psi(\mathcal{M})$, we will learn the residence manifold as a hypersurface (see Definition 1.1.14) in $\mathbb{R}^{D+1}$ so that it has a normal vector. Not every manifold is embeddable as a hypersurface, so we only hope to achieve this hypersurface locally, allowing for possible self-intersections, knots, or lack of orientability. Therefore, we will only attempt to embed the residence manifold as an *immersed* hypersurface.

3. *Out of sample extension:* Typically, manifold learning algorithms will recover a map $\psi : \mathcal{X} \to \mathbb{R}^{D+1}$. We require that our proposed map Ψ is an extension of ψ to $\mathcal{M}$.

We surmount the above obstacles as follows:

1. *Group invariance:* To account for SE(3) invariances, we use *group invariant features* $\mathcal{F} : \mathbb{R}^N \to \mathbb{R}^{d_\mathcal{F}}$ satisfying $\mathcal{F}(x) = \mathcal{F}(gx)$ for all $g \in G$. To use group invariant features, we first project the raw data $\mathcal{X}$ onto a *feature space* by computing a *featurized point cloud* $\mathcal{X}_\mathcal{F} := \{\mathcal{F}(x_i)\}$. Assuming that $\mathcal{X}_\mathcal{F}$ is sampled from a *featurized manifold* $\mathcal{M}_\mathcal{F}$, we compute a diffusion map $\psi_\mathcal{F}$ on $\mathcal{X}_\mathcal{F}$ instead of $\mathcal{X}$. We describe our choices of feature map in Section 3.4.1.

2. *Hypersurface learning:* We apply the diffusion map algorithm to $\mathcal{X}_\mathcal{F}$ and obtain $\psi_\mathcal{F} : \mathcal{X}_\mathcal{F} \to \mathbb{R}^{D+1}$ such that $\psi_\mathcal{F}(\mathcal{X}_\mathcal{F})$ appears sampled from a hypersurface immersed in $\mathbb{R}^{D+1}$. To achieve such an immersion (see Definition 1.1.12) we use independent eigencoordinate selection [CM19].

3. *Out of sample extension:* Once $\psi_\mathcal{F}$ has been obtained, we extend it to $\mathcal{M}_\mathcal{F}$ via *diffusion nets* [MSCC19], obtaining a neural network $\Psi^\mathcal{F}$. The overall parameterization will then be given by $\Psi := \Psi^\mathcal{F} \circ \mathcal{F}$.

3.4.1 Data post-processing matters

We contrast eight different choices of feature maps $\mathcal{F}$. We will assume $x \in \mathbb{R}^N$ is a point in all-atom space where x is a single vector of all 3D positions of all atoms. This all atom vector may be reshaped into the all-atom matrix $X \in \mathbb{R}^{(N/3)\times 3}$ where the nth row in X is given by $[x_{3n}, x_{3n+1}, x_{3n+2}]$. With this notational change, we describe the feature maps below:

1. No featurization (NoFeaturization): $\mathcal{F}(X) = X$.

2. Recentering: Here $\mathcal{F}(X) = X - \mu(X)$, where $\mu(X)$ is the mean of the rows of the atomic coordinates $X \in \mathbb{R}^{14\times 3}$.

3. Trajectory alignment (TrajAlign): Aligning each configuration X_i at time-step t_i to the previ-

ous configuration X_{i-1} via Procrustes alignment). TrajAlign is a common method for post-processing MD trajectories but it does not correspond to an unambiguous function independent of the data. Consequently, this map is difficult to compute at inference time where a new configuration not in the MD trajectory must be featurized.

4. Computing the gram-matrix of the recentered configurations (GramMatrix): $\mathcal{F}(X) = (X - \mu(X))(X - \mu(X))^\top \in \mathbb{R}^{(N/3)\times(N/3)}$,

5. Computing the gram-matrix of the recentered configurations of the carbon atoms (GramMatrix-Carbons): $\mathcal{F}(X) = (MX - \mu(MX))(MX - \mu(MX))^\top \in \mathbb{R}^{4\times 4}$. Here $M \in \mathbb{R}^{4\times 14}$ is a *mask* which selects the indices corresponding to the carbon atoms.

6. Global translation of the atoms to realign C_1 to the origin followed by rotation of the $C_1 - C_2$ bond onto the x-axis (BondAlign(1,2)),

7. Global translation of the atoms to realign C_2 to the origin followed by rotation of the $C_2 - C_3$ bond onto the x-axis (BondAlign(2,3)),

8. Changing basis of the recentered configurations such that the C_2, C_3, C_4 atoms all lie in the xy-plane (PlaneAlign).

The above feature maps represent changing levels of group invariances, ranging from no invariance to complete invariance to rotations, reflections, and translations (E(3) invariance). In particular, TrajAlign is not a feature map but nonetheless a commonly used pre-processing technique. NoFeaturization has no invariance, Recentering only has translation invariance, and GramMatrixCarbons is only invariant to the actions of E(3) on a subspace of $\mathbb{R}^N$ spanned by the carbon atoms. BondAlign(1,2),

BondAlign(2,3), and PlaneAlign are SE(3) invariant but not E(3) invariant since they separate antipodal configurations. Finally, GramMatrix is E(3) invariant as is well known from Hilbert's classical invariant theory. We visualize BondAlign(1,2), BondAlign(2,3), and PlaneAlign in Figures 3.7, 3.8, and 3.9 respectively.

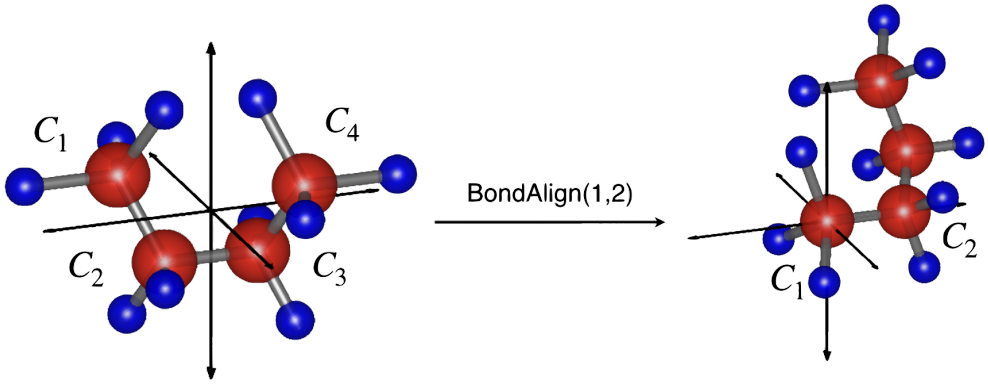


Figure 3.7: In BondAlign(1,2), the $C_1 - C_2$ bond is aligned along the x-axis.

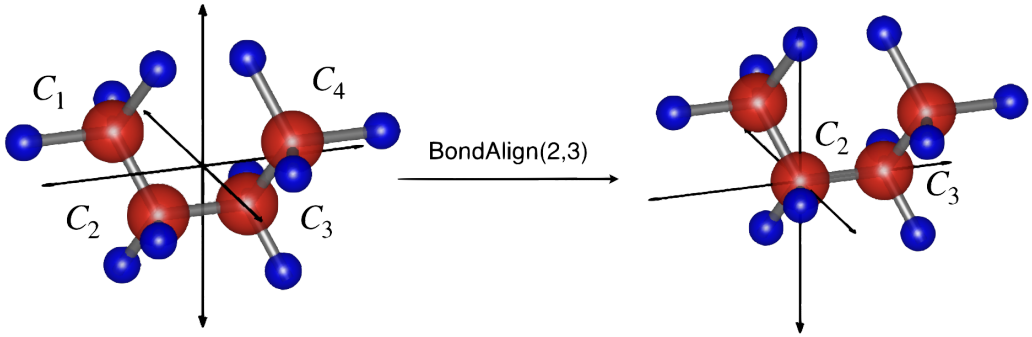


Figure 3.8: In BondAlign(2,3), the $C_2 - C_3$ bond is aligned along the x-axis.

3.4.2 Diffusion maps

We pass the data $\mathcal{X}$ through $\mathcal{F}$ to get a featurized pointcloud $\mathcal{X}_{\mathcal{F}} = \{p_i\}$ where $p_i = \mathcal{F}(x_i)$, which we posit is sampled from the featurized manifold $\mathcal{M}_{\mathcal{F}} := \mathcal{F}(\mathcal{M})$. To embed the featurized manifold in low dimensions we use diffusion maps [CKL+08] introduced in Chapter 1 motivated by

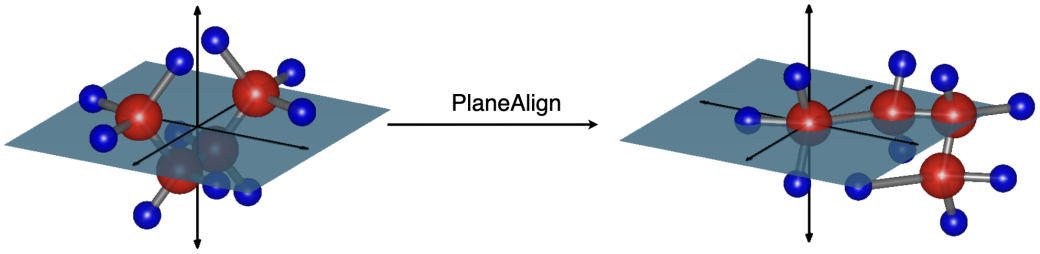


Figure 3.9: In PlaneAlign the basis is changed such that the first three carbon atoms are in the xy-plane.

the fact that we can choose a finite number of coordinates m such that the diffusion map $\tilde{\psi}$ is an embedding in the sense of Definition 1.1.12. Thus, the diffusion map applied to the pointcloud $\mathcal{X}_{\mathcal{F}}$ yields a map $\tilde{\psi}^{n,\epsilon} : \mathcal{X}_{\mathcal{F}} \to \mathbb{R}^m$ given by:

$$\tilde{\psi}^{n,\epsilon}(p_i) := \left(\lambda_1^{n,\epsilon}\tilde{\psi}_1^{n,\epsilon}(p_i), \ldots, \lambda_m^{n,\epsilon}\tilde{\psi}_m^{n,\epsilon}(p_i)\right). \tag{3.60}$$

In an abuse of notation we identify the map $\tilde{\psi}^{n,\epsilon}$ with the $n \times m$ matrix formed by stacking the rows $\tilde{\psi}^{n,\epsilon}(p_i)$.

3.4.3 Independent eigencoordinate selection (IES)

While $\tilde{\psi}^{n,\epsilon}$ approximates the immersion $\tilde{\psi}$ of $\mathcal{M}_{\mathcal{F}}$ in $\mathbb{R}^m$, this m might be too large for the immersed manifold to be a hypersurface. Intuitively, a D-dimensional hypersurface (locally) has large D-dimensional volume but small $D+1$-dimensional volume (for instance a curve in $\mathbb{R}^2$ has a well-defined length but zero area). Thus, given the embedding $\tilde{\psi}^{n,\epsilon}$ (3.60) of n points in m dimensions, we will select S, a subset of the columns of the matrix $\tilde{\psi}^{n,\epsilon}$, $|S| = D+1$, such that the resulting map

$$\psi_{\mathcal{F}} = (\tilde{\psi}_j^{n,\epsilon}(p))_{j\in S} \tag{3.61}$$

forms an embedding in $D+1$ dimensions which locally has large D-dimensional volume but small $D+1$ dimensional volume. To compute these local volumes, we use *independent eigencoordinate selection* (IES) [CM19]. To introduce IES, we need background on differential geometry found in Chapter 1. We assume that $\mathcal{M}_{\mathcal{F}}$ is a Riemannian manifold of dimension $\dim(\mathcal{M}_{\mathcal{F}})$ and metric $\{g(p)\}_{p\in\mathcal{M}_{\mathcal{F}}}$. Let $\tilde{\psi} : \mathcal{M}_{\mathcal{F}} \to \mathbb{R}^m$ be a smooth immersion (Definition 1.1.12) and $d\tilde{\psi}_p : T_p\mathcal{M}_{\mathcal{F}} \to T_{\tilde{\psi}(p)}\mathbb{R}^m$ be its differential at $p \in \mathcal{M}_{\mathcal{F}}$ (Definition 1.1.9). Note that since $\tilde{\psi}$ is an immersion, $\text{rank}(d\tilde{\psi}) = \dim(\mathcal{M}_{\mathcal{F}})$. However, $d\tilde{\psi}_p$ may not be full rank since $m \geq \dim(\mathcal{M}_{\mathcal{F}})$. We can nonetheless compute its pseudo-inverse $d\tilde{\psi}_p^\dagger$ on vectors in $T_{\tilde{\psi}(p)\mathbb{R}^m}$ to define the *pushforward metric* g_*:

$$\langle u, v\rangle_{g_*} := \langle d\tilde{\psi}_p^\dagger u, d\tilde{\psi}_p^\dagger v\rangle_{g(p)},\ u, v \in T_{\tilde{\psi}(p)}\mathbb{R}^m. \tag{3.62}$$

The pushforward metric gives a Riemannian structure to $\tilde{\psi}(\mathcal{M})$ that makes $\tilde{\psi}$ an isometry by definition. In local coordinates in $\mathbb{R}^m$, this results in a quadratic form at each $\tilde{\psi}(p_i)$:

$$\langle u, v\rangle_{g_*} = u^\top G(p_i) v. \tag{3.63}$$

Here $G(p_i) \in \mathbb{R}^{m\times m}$ is a local representation of the metric. Crucially, since $\text{rank}(d\tilde{\psi}_p) < m$, $G(p_i)$ will be positive semi-definite with $\text{rank}(G(p_i)) = \dim(\mathcal{M}_{\mathcal{F}}) = D$. Thus, the first D singular vectors of G will span $T_{\tilde{\psi}(p)}\tilde{\psi}(\mathcal{M}_{\mathcal{F}})$, while the $(D+1)$th singular vector will be contained in $T_{\tilde{\psi}(p)}\tilde{\psi}(\mathcal{M}_{\mathcal{F}})^\perp$. Remarkably, these $G(p_i)$ can be estimated from the data $\mathcal{X}_{\mathcal{F}}$ via Algorithm 4 introduced in [PJM13] and named RMetric.

Using RMetric, we compute these singular vectors $U(p_i) \in \mathbb{R}^{m\times(D+1)}$, noting that m is the number of eigenfunctions computed and $D+1$ is the suggested dimension in which to eventually

Algorithm 4 RMetric

Require: Embedding $\tilde{\psi}^{n,\epsilon} \in \mathbb{R}^{n\times m}$, Laplacian $L^{(n)}_{\epsilon,1}$, target dimension $D+1$.

for all $i = 1 \to n, \alpha = 1 \to m, \beta = 1 \to m$ **do**
  $[\tilde{H}(i)]_{\alpha\beta} = \sum_{j\neq i}[L^{(n)}_{\epsilon,1}]_{ij}(\tilde{\psi}^{n,\epsilon}_{\beta}(p_j) - \tilde{\psi}^{n,\epsilon}_{\beta}(p_i))(\tilde{\psi}^{n,\epsilon}_{\alpha}(p_j) - \tilde{\psi}^{n,\epsilon}_{\alpha}(p_i))$
end for
for $i = 1 \to n$ **do**
  $U(i), \Lambda(i) \leftarrow$ REDUCEDRANKSVD$(\tilde{H}(i), D+1)$
  $H(i) = U(i)\Lambda(i)U(i)^\top$
  $G(i) = U(i)\Lambda(i)^{-1}U(i)^\top$
end for

Ensure: $G(i), H(i) \in \mathbb{R}^{m\times m}$, $U(i) \in \mathbb{R}^{m\times(D+1)}$, $\Lambda(i) \in \mathbb{R}^{(D+1)\times(D+1)}$, for $i \in \{1, 2, \ldots, n\}$.

embed the hypersurface. We now select the subset of rows $S \subseteq [m] := \{1, 2, \ldots, m\}$ of $U(p_i)$ with large D-dimensional volume but small $(D+1)$-dimensional volume. We quantify these volumes in the following way: suppressing the dependence on p_i for a moment, let $U' = U(p_i)[:, [D]]$ be the D column vectors which span $T_{\tilde{\psi}(p_i)}\tilde{\psi}(\mathcal{M}_{\mathcal{F}})$. Next let $U'_{S'} = U[S', :]$ be a selection of the rows of U' where $|S'| = D$. We consider the log of the volume of the parallelepiped formed by the D-dimensional rows of the column-normalized $U'_{S'}$, written as follows:

$$\text{Vol}(U'_{S'}, p_i) = \log\left(\sqrt{\det\left([U'_{S'}]^\top U'_{S'}\right)}\right) - \sum_{j=1}^{D}\log\|U'_{S'}[:, j]\|_2^2. \tag{3.64}$$

To estimate this volume over all points p_i, we take the mean of the normalized projected volume over all the data. Moreover, since S' corresponds to a selection of Laplacian eigenvectors, we also promote selecting eigenvectors with greater smoothness. To do so, we add a regularization term $-\sum_{k\in S'}\lambda_k$ weighted by a parameter ζ which penalizes how much the selected eigenvectors oscillate. Note that we take a minus sign in front since we aim to *maximize* the volume score with the *lowest* frequency

eigenfunctions. This leads to the overall score considered in IES [CM19]:

$$R_\zeta(S') = \frac{1}{n}\sum_{i=1}^{n} \mathrm{Vol}(U'_{S'}, p_i) - \zeta \sum_{k\in S'} \lambda_k. \tag{3.65}$$

In the original implementation of IES, $R_\zeta(S')$ is maximized over all $S' \subseteq [m]$ with $|S| = D$ to obtain an optimal set S_*. Here, we simply take $S_* = [D]$, the first D eigenfunctions. This algorithm is summarized in Algorithm 5.

Algorithm 5 Independent eigencoordinate search (IES)

Require: Tangent bundle $\{U'(p_i)\}_{p_i\in\mathcal{X}_\mathcal{F}} \subseteq \mathbb{R}^{m\times(D)}$, target dimension $|s| \le D$, regularization parameter ζ.

1: **for** $S' \subseteq [m]$ such that $|S'| = s, 1 \in S'$ **do**
2: $\quad R_\zeta(S') \leftarrow 0$
3: $\quad$ **for** $i = 1, \cdots, n$ **do**
4: $\quad\quad U'_{S'} \leftarrow U(p_i)[S', :]$
5: $\quad\quad R_\zeta(S') \leftarrow R_\zeta(S') + n^{-1}\left[\log\left(\sqrt{\det\left([U'_{S'}]^\top U'_{S'}\right)}\right) - \sum_{j=1}^{s}\log\|U'_{S'}[:,j]\|_2^2 - \zeta\sum_{k\in S'}\right]$
6: $\quad$ **end for**
7: **end for**
8: $S^* = \underset{S'}{\mathrm{argmax}} R_\zeta(S')$ {Alternative: $S^* = [D]$}

Ensure: Independent eigencoordinates set S^*

Algorithm 6 Hypersurface search (HyperSearch)

Require: Tangent bundle $\{U(p_i)\}_{p_i\in\mathcal{X}_\mathcal{F}} \subseteq \mathbb{R}^{m\times(D+1)}$, Optimal D independent coordinates S^*, regularization parameter ζ.

1: **for** $k \in [D+1] \setminus S^*$ **do**
2: $\quad S \leftarrow S^* \cup \{k\}, \quad R(S,\zeta) \leftarrow 0$
3: $\quad$ **for** $i = 1, \cdots, n$ **do**
4: $\quad\quad U_S \leftarrow U(p_i)[S, :]$
5: $\quad\quad R_{-\zeta}(S) \leftarrow R_{-\zeta}(S) + n^{-1}\left[\log\left(\sqrt{\det\left([U_S]^\top U_S\right)}\right) - \sum_{j=1}^{s}\log\|U_S[:,j]\|_2^2 + \zeta\sum_{k\in S}\lambda_k\right]$
6: $\quad$ **end for**
7: **end for**
8: $S \leftarrow \underset{S}{\mathrm{argmin}}\, R_{-\zeta}(S)$ {Alternative: $S = [D+1]$}

Ensure: Hypersurface set S

Once S^* has been selected, add the next highest frequency eigenfunction to it to form the set of coordinates $S^* \cup k$, but we need to select one more column k of $\tilde{\psi}^{n,\epsilon}$ so that the $(D+1)$-dimensional volume is small. Thus, we seek $k \in [m] \backslash S^*$ such that $R_{-\zeta}(S^* \cup \{k\})$ is *minimized*. Note that we change the sign of the penalty term in R at the minimization step to penalize for selecting oscillatory eigenvectors as we did in the maximization step. We term this addition to IES as Hypersurface Search (HyperSearch) and summarize it in Algorithm 6.

Algorithm 7 Learning the residence manifold

Require: Featurized data $\mathcal{X}_{\mathcal{F}}$, kernel bandwidth $\epsilon > 0$, number of eigenvectors m, embedding dimension $D+1$, regularization parameter ζ.

1: $\tilde{\psi}^{\epsilon,n}, L_{\epsilon,1}^{(n)} \leftarrow$ DIFFUSION_MAP$(\mathcal{X}_{\mathcal{F}}, \epsilon, m)$
2: $\{U(p_i)\} \leftarrow$ RMETRIC$(\tilde{\psi}^{\epsilon,n}, L_{\epsilon,1}^{(n)}, D+1)$
3: $S_* \leftarrow$ IES$(\{U(p_i)[:, [D]]\}, \zeta)$
4: $S \leftarrow$ HYPERSEARCH$(\{U(p_i)\}, S_*, \zeta)$
5: $\psi_{\mathcal{F}} \leftarrow \psi^{n,\epsilon}[:, S]$

Ensure: Hypersurface set S, Independent coordinates S_*, Hypersurface embedding $\psi_{\mathcal{F}}$

Selecting $D+1$ and $\mathcal{F}$. We summarize the entire pipeline for learning the residence manifold in Algorithm 7. The resulting samples $\{\psi_{\mathcal{F}}(p_i)\}$ are assumed to lie on the surrogate manifold $\widehat{\mathcal{M}}$. Clearly, the validity of this algorithm hinges on an appropriate choice of $D+1$. Ideally, we would like $D+1 = \dim(\mathcal{M}_{\mathcal{F}}) + 1$, so we may estimate the intrinsic dimension of the featurized pointcloud $\mathcal{X}_{\mathcal{F}}$. Alternately, we can consider the $D+1$ which provides the best hypersurface immersion for the given data. This can be achieved as follows: we iterate through the values $2 \leq D+1 \leq m$ and find the $D+1$ for which the difference between the D-dimensional volume and the $D+1$ dimensional volume is the largest. This amounts to calculating

$$\mathsf{HyperSurface}(D+1) := \frac{R_\zeta(U'_{S^*}) - \mathrm{R}_{-\zeta}(U_S)}{|R_\zeta(U'_{S^*})|}. \tag{3.66}$$

The $D+1$ that *maximizes* $\mathsf{HyperSurface}(D+1)$ should be the dimension which provides the best codimension-1 embedding for the data. For simplicity, here we take $S^* = [D]$ and $S = [D+1]$, motivated by the fact that we are not interested necessarily in independent coordinates but simply by those which form an appropriate hypersurface, i.e minimizing (3.66). In Figure 3.10, we plot $\mathsf{HyperSurface}(D+1)$ for the feature maps given in Section 3.4.1.

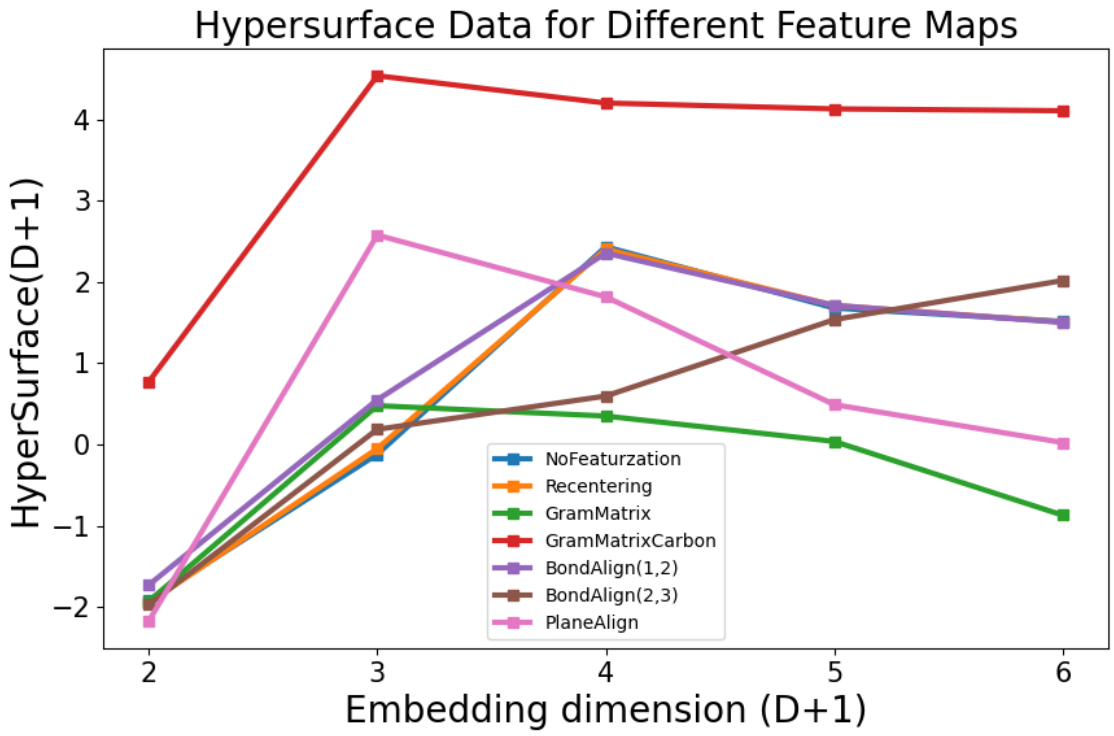


Figure 3.10: Plotting $\mathsf{HyperSurface}(D+1)$ against $D+1$ for different feature maps. Notably, GramMatrixCarbon gives the largest value of $\mathsf{HyperSurface}(\cdot)$ at $D+1=2$ which corresponds to the residence manifold of dimension $D=1$. Moreover PlaneAlign seemingly gives the best hypersurface in 3D

.

3.4.4 Data generation and results on selecting diffusion coordinates

To use Algorithm 7 we first generated a trajectory from an unbiased Langevin dynamics MD simulation of butane with friction coefficient of 10 ps^{-1} at 300K. We collected 10^5 points, each sam-

pled at an interval of 2 femotseconds. To then determine the best feature map, we subsampled this trajectory at every 100th point to obtain a point cloud of size 10^3. This was our raw data $\mathcal{X}$ which we then processed through the different feature maps to obtain the featurized point clouds $\mathcal{X}_{\mathcal{F}}$. The parameter ϵ was tuned via the K-sum test in [ECT22]. On each featurized point cloud we ran Algorithm 7, where the number of eigenvectors was chosen to be $m = 25$ and ζ was fixed at 0.01 to penalize the frequencies of the eigenfunctions. Moreover, the parameter ϵ was tuned according to the kernel sum test (see [ECT22]). Figure 3.10 reveals that, with the GramMatrixCarbon feature map, the best hypersurface is obtained when embedding in two dimensions. Additionally, when using Recentering or NoFeaturization, we require four dimensions to best embed the data with co-dimension 1. We can also flip this analysis and ask: for a given $D + 1$, which feature map $\mathcal{F}$ best embeds the data as a hypersurface? For instance, for both $D + 1 = 2, 3$, the answer is GramMatrixCarbon. For $D + 1 = 3$, the next best feature map is PlaneAlign. A visual inspection of the 3D coordinates why this is the case: both GramMatrixCarbon and PlaneAlign embed as 2D surfaces in 3D but can also be embedded in 2D as curves (see Figure 3.12). For GramMatrixCarbon, this observation is consistent with the HyperSurface score, where the feature map attains a high score for both $D + 1 = 2, 3$. However, in $D + 1 = 2$, PlaneAlign visually embeds as a hypersurface in 2D but *does not* attain a large HyperSurface score. In fact, PlaneAlign provides a better embedding than GramMatrixCarbon because PlaneAlign embeds the metastable states $\theta = 5\pi/3, \pi/3$ as disjoint sets (as they are in the all-atom data), while the GramMatrixCarbon embedding does not. This analysis reveals the inherent limitations of HyperSurface–it is only a heuristic and can identify critical dimensions for a feature map. However, it can lead to *false negatives*, as in the case of PlaneAlign which attains the lowest score for $D + 1 = 2$, but is clearly visually the "best" 2D embedding.

For $D + 1 = 3$, after PlaneAlign, GramMatrixCarbon, BondAlign(1,2) and GramMatrix

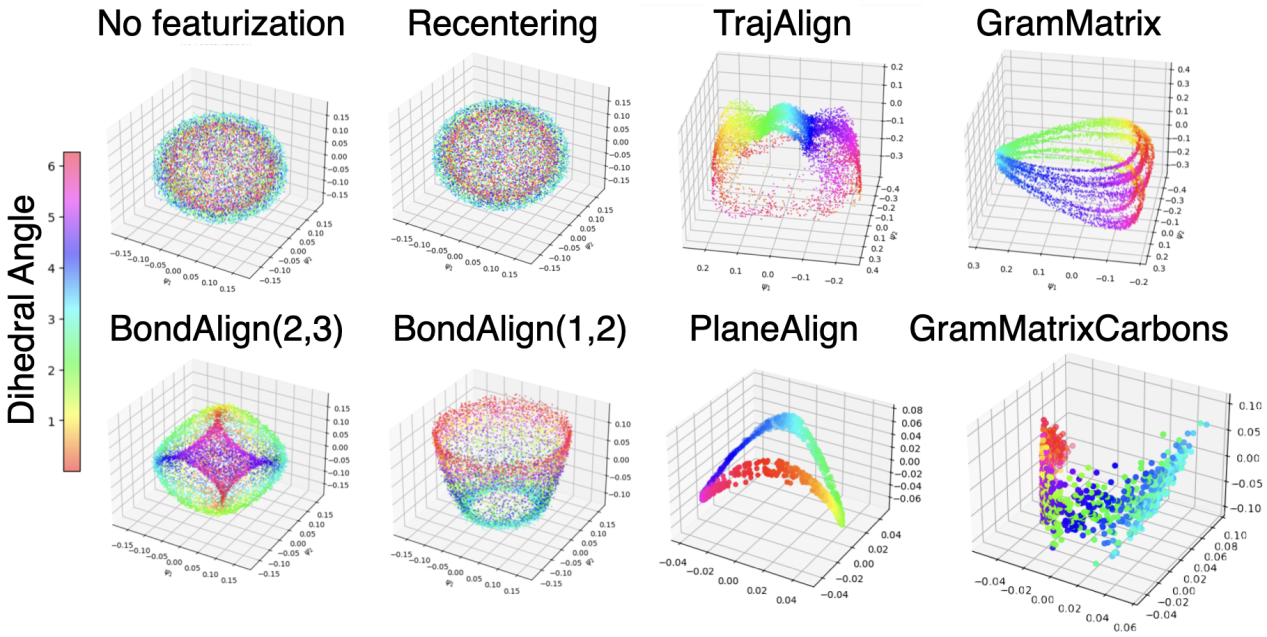


Figure 3.11: Left to right, top to bottom: The diffusion map embeddings $\{\psi(\mathcal{F}_k(x_i)\}_{x_i i \in \mathcal{X}}$ for eight different choices of feature maps with varying degrees of invariance. The points are coloured according to the dihedral angle.

attain the next top 3 scores respectively. The embeddings into 3D are shown in Figure 3.11. Rather intriguingly, GramMatrixCarbon is both embedded as a curve in two dimensions (locally 1-D) and as a surface in three dimensions (locally 2D). Moreover, the BondAlign(1,2) (second row, second column) and GramMatrix (first row, fourth column) embeddings are also locally 2D in three dimensions.

Importance of hydrogen atoms A highly significant aspect of this analysis is the importance of the hydrogen atoms–the embedding obtained via GramMatrixCarbon is parameterized by the $\cos\theta$, which is a somewhat suboptimal collective variable in reproducing transition rates (Table 3.1). However, in GramMatrix when we also include the pairwise inner products with hydrogen atoms, we find that the embedding is smoothly parameterized by θ, where the two disjoint gauche states can be separated.

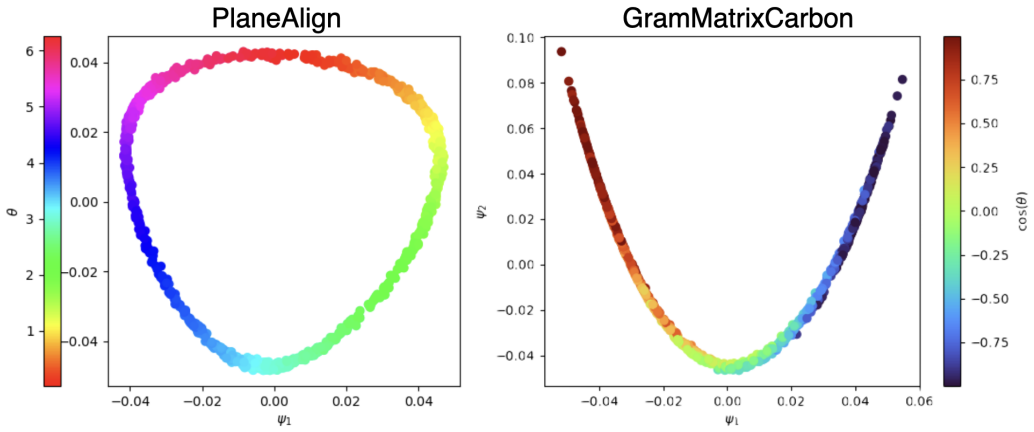


Figure 3.12: Visualizing the first two selected Laplacian eigenvectors for PlaneAlign and GramMatrixCarbon. Both embeddings form hypersurfaces in two dimensions.

3.4.5 Diffusion net

After computing the embedding $\psi_{\mathcal{F}}$, we consider the elements of the pointcloud $\{\psi_{\mathcal{F}}(p_i)\}$ as the evaluations of a diffusion net $\Psi_{\Theta_1} : \mathbb{R}^N \to \mathbb{R}^{D+1}$ where $\Psi_{\Theta_1} = \Psi^{\mathcal{F}}_{\Theta_1} \circ \mathcal{F}$ as a neural network with parameters Θ_1 (the subscript 1 corresponding to Step 1 of Algorithm 3). We train $\Psi^{\mathcal{F}}$ such that (1) $\Psi^{\mathcal{F}}$ matches $\psi_{\mathcal{F}}$ at every point $p_i = \mathcal{F}(x_i)$ and (2) $[\Psi^{\mathcal{F}}_{\Theta_1}]_j$, the jth coordinate of $\Psi^{\mathcal{F}}$ is a Laplacian eigenfunction with eigenvalue λ_j. The training objective (1) can be attained by simple mean squared error loss. To achieve training objective (2), we abuse notation and denote $[\Psi^{\mathcal{F}}_{\Theta_1}]_j$ as the evaluations of $[\Psi^{\mathcal{F}}_{\Theta_1}]_j$ on all points p_i. Then the vector $L_{n,\epsilon}[\Psi^{\mathcal{F}}_{\Theta_1}]_j$ represents the action of the Laplacian on this function. Objective (2) stipulates that $L^{(n)}_{\epsilon,1}[\Psi^{\mathcal{F}}_{\Theta_1}]_j(p_i) \approx \lambda^{n,\epsilon}_j[\Psi^{\mathcal{F}}_{\Theta_1}]_j(p_i)$. This reasoning leads to the following loss function proposed in [MSCC19]:

$$L_{\text{DNet}}(\Theta_1) := \frac{1}{n}\sum_{i=1}^{n} \|\Psi^{\mathcal{F}}_{\Theta_1}(x_i) - \psi^{\mathcal{F}}(p_i)\|_2^2 + \tag{3.67}$$

$$\alpha_{\text{DNet}} \frac{1}{n}\sum_{i=1}^{n}\sum_{j=1}^{D+1} |L^{(n)}_{\epsilon,1}[\Psi^{\mathcal{F}}_{\Theta_1}]_j(p_i) - \lambda^{n,\epsilon}_j[\Psi^{\mathcal{F}}_{\Theta_1}]_j|^2 \tag{3.68}$$

We can also train a decoder Ψ_{Dec} but since $\mathcal{F}$ is not injective, we can only carry $\Psi(x_i)$ back to $\mathcal{F}(x_i)$. In this case, the decoder is trained with the following reconstruction loss:

$$L_{\mathrm{DNet}}^{\mathrm{Dec}}[\Theta_1^{\mathrm{Dec}}, \Theta_1] := \frac{1}{n}\sum_{i=1}^{n} \|\Psi^{\mathrm{Dec}} \circ \Psi(x_i) - \mathcal{F}(x_i)\|_2^2. \tag{3.69}$$

Here our focus remains on the encoder Ψ.

3.4.6 Undoing spurious topological features

Among the SE(3)-invariant feature maps, only BondAlign(2,3) presents a manifold that cannot be embedded as a hypersurface in either two or three dimensions (Figure 3.11, second row, first column). However, the dihedral angle still seems to smoothly parameterize points on the embedding, which resembles a sphere with handles. We find that this phenomenon persists when applying diffusion maps to data pre-processed via BondAlign(2,3) only on the carbon atoms (see top left, Fig. 3.13). A further look by slicing the embedding at various z-coordinates at this embedding reveals *self-intersections* in three dimensions (see top right, Fig 3.13). We additionally embed the data post-processed by the feature map BondAlign(2,3) into four dimensions and visualize its sliced data in three dimensions with the third coordinate being the *fourth* Laplacian eigenfunction ψ_4 in the bottom left of Fig 3.13. This reveals that the self-intersections are actually spurious and that the three-dimensional sliced data forms *loops* which can be parameterized using the dihedral angle. This suggests that there is a functional dependence among the first four coordinates $\psi = (\psi_1, \ldots, \psi_4)$. We quantify this func-

tional dependence via the following energy proposed in [KMVK24]:

$$\mathcal{E}(\Psi) := \sum_{i<j} |\langle \nabla_{\mathbb{R}^{d_{\mathcal{F}}}} \Psi_i, \nabla_{\mathbb{R}^{d_{\mathcal{F}}}} \Psi_j \rangle|^2. \tag{3.70}$$

The functional (3.70) was suggested as a regularization term in a *conformal autoencoder*. We now compute a map $\Psi : \mathbb{R}^{14\times 3} \to \mathbb{R}^{D+1}$ as a neural network trained to be a conformal autoencoder whose coordinates are Laplacian eigenfunctions. In particular, the encoder is trained to minimize the following loss function:

$$L^{\text{Enc}}_{\text{LAPCAE}}[\Theta_1] := \frac{1}{n} \sum_{i=1}^{n} \sum_{j=1}^{D+1} |L^{(n)}_{\epsilon,1} \Psi^{\mathcal{F}}_{\Theta_1}(p_i) - \lambda_j^{n,\epsilon} \Psi^{\mathcal{F}}_{\Theta_1}(p_i)|^2 + \alpha_{\text{LAPCAE}} \mathcal{E}(\Psi_{\Theta_1}). \tag{3.71}$$

We train the overall autoencoder $\Psi^{\text{Dec}} \circ \Psi$ with a weighted combination of $L^{\text{Dec}}_{\text{DNet}}$ (3.69) and $L^{\text{Enc}}_{\text{LAPCAE}}$ (3.71):

$$L_{\text{LAPCAE}}[\Theta^{\text{Dec}}_1, \Theta_1] := L^{\text{Dec}}_{\text{DNet}}[\Theta^{\text{Dec}}_1] + \alpha^{\text{Enc}}_{\text{LAPCAE}} L^{\text{Enc}}_{\text{LAPCAE}}[\Theta_1]. \tag{3.72}$$

After training Ψ on (3.72) we project the all-atom data to obtain a low-dimensional embedding formed by points $\{\Psi^{\mathcal{F}}(p_i)\}_{i=1}^n$ (bottom left, Fig 3.13). Remarkably, the LAPCAE removes the self-intersections, resulting in a cylinder embedded in three dimensions. Moreover, the points are smoothly organized by the dihedral angle. Undoing topological features like spurious knots is an important challenge in data visualization and manifold learning. Our result is therefore of independent interest to geometric data science in addition to the MD community.

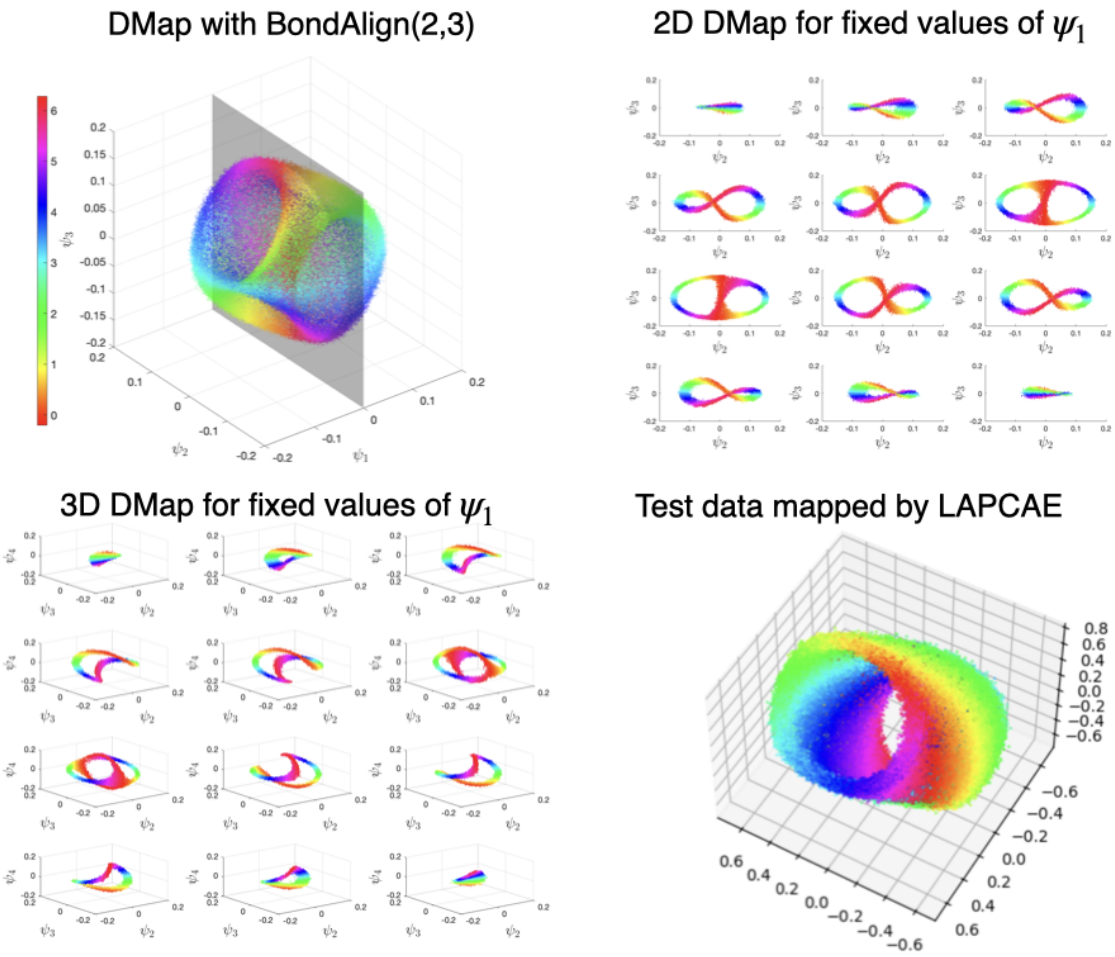


Figure 3.13: Top left: The diffusion map embedding after featurization by recentering the $C_2 - C_3$ bond on the x-axis. Top right: Slices of the embedding by planes parallel to the xy-plane. Bottom left: Visualizing the sliced data along with the corresponding fourth eigenfunction. Bottom right: The corrected embedding after mapping via a Laplacian conformal autoencoder.

3.4.7 Computational details

After acquiring hypersurface coordinates for each feature map, we took every 10th point of our overall trajectory $\{X_{i\Delta t}\}_{i=1}^{10^5}$ as a training set $\mathcal{X}$ of size 10^4. The rest of the points were considered part of the testing set. We then postprocessed this point cloud $\mathcal{X}$ using the group invariant feature maps described in Section 3.4.1 to obtain featurized pointclouds $\mathcal{X}_{\mathcal{F}}$ and computed the diffusion map on the selected hypersurface coordinates. We then trained a diffusion net with encoder Ψ and decoder

Ψ^{Dec}. Both encoders and decoders were four-layer fully connected networks with either $\tanh$ or $x+\sin^2(x)$ activation. We trained these using an ADAM optimizer with learning rate 0.01. For training LAPCAE with the feature map BondAlign applied to the carbon atoms, the constant $\alpha_{\mathrm{DNet}}, \alpha^{\mathrm{Enc}}_{\mathrm{LAPCAE}}$ were chosen to be 1.0. We provide additional details regarding neural network training in Section 3.5.3.

3.5 Learning CVs from the residence manifold

3.5.1 Learning the confining potential

We assume now that the diffusion net Ψ has been trained and that the data $\widehat{\mathcal{X}} = \{y_i\}$ lies on the surrogate manifold $\widehat{\mathcal{M}}$ where $y_i = \Psi(x_i)$. Now we aim to learn a surrogate potential $\widehat{\Phi}$ which vanishes exactly on $\widehat{\mathcal{M}}$. We are aided by the fact that $\widehat{\mathcal{M}}$ can be presented as a hypersurface. In this case, a loss for learning $\widehat{\Phi}$ may be derived using the eikonal equation. In particular, $\widehat{\mathcal{M}}$ may be exhibited (possibly locally) as the boundary of some open set in $\mathbb{R}^{D+1}$. In this case, then the signed distance function (SDF) $\widehat{\Phi}_{\mathrm{SDF}}$ to this open set satisfies $\|\nabla\widehat{\Phi}_{\mathrm{SDF}}\|^2 = 1$. We propose that this SDF be a representation of the desired surrogate potential $\widehat{\Phi}$. This is because $\nabla\widehat{\Phi}_{\mathrm{SDF}}(y) = -\widehat{n}(y)$ where $\widehat{n}(y)$ is the outward pointing normal vector to $\widehat{\mathcal{M}}$ at y, i.e the condition we want the surrogate potential $\widehat{\Phi}$ to satisfy. A loss function for $\widehat{\Phi_{\Theta_2}}$ is given by:

$$L_{\mathrm{eikonal}}[\Theta_2] := \frac{1}{n}\sum_{i=1}^{n}(\|\nabla\widehat{\Phi}_{\Theta_2}(y_i)\|_2^2 - 1)^2 + \alpha_{\mathrm{zero}}\|\widehat{\Phi}_{\Theta_2}(y_i)\|_2^2. \tag{3.73}$$

As an alternate approach, given the point cloud $\widehat{\mathcal{X}}$, the normals at each point $\widehat{n}(y_i)$ may be estimated directly without needing a neural network by taking a neighborhood of y_i and fitting a tangent plane to

this neighborhood. The estimated normal $\widehat{n}_i$ at y_i is then the normal vector to this tangent plane. This method is summarized in Algorithm 8.

Algorithm 8 Estimate Normals via k-NN and Tangent Plane Fitting

Require: Point cloud $\widehat{\mathcal{X}} = \{y_i\}_{i=1}^N \subseteq \mathbb{R}^{D+1}$, number of neighbors k
Ensure: Estimated normals $\{\widehat{n}_i\}_{i=1}^N$

1: Build a spatial index (e.g., KD-tree) for fast nearest-neighbor queries
2: **for** each $y_i \in \widehat{\mathcal{X}}$ **do**
3: Find the k nearest neighbors of y_i: $\{y_j\}_{j=1}^k \leftarrow \texttt{knn}(y_i, k)$
4: Compute the centroid of neighbors:

$$\bar{y} \leftarrow \frac{1}{k}\sum_{j=1}^{k} y_j$$

5: Center the neighbors:

$$Y \leftarrow [y_j - \bar{y}]_{j=1}^k$$

6: Compute the covariance matrix:

$$C \leftarrow \frac{1}{k} Y^\top Y$$

7: Perform eigen-decomposition or SVD of C to get eigenvalues $\lambda_1 \leq \cdots \leq \lambda_d$ and eigenvectors $v_1, \ldots, v_d$
8: Assign the normal vector corresponding to the eigenvector with the smallest eigenvalue:

$$\widehat{n}_i \leftarrow v_1$$

9: **end for**
10: **return** $\{\widehat{n}_i\}_{i=1}^N$

After estimating the point cloud normals $\{\widehat{n}_i\} \subseteq \mathbb{R}^{D+1}$ we use these to supplement L_{eikonal} with an additional regularization term to aid learning the surrogate potential Φ. Thus, the final loss for learning Φ is given by:

$$L_{\text{potential}}[\Theta_2] := L_{\text{eikonal}}[\Theta_2] + \frac{\alpha_{\text{normals}}}{n}\sum_{i=1}^{n} \|\nabla \Phi_{\Theta_2}(y_i) - \widehat{n}_i\|_2^2. \tag{3.74}$$

Here $\alpha_{\text{normals}} > 0$ is a hyperparameter. In Figure 3.14 we present $\nabla \widehat{V}$ learned for the BondAlign(1,2),

BondAlign(2,3), and PlaneAlign feature maps, confirming that the gradients of $\widehat{\Phi}$ are normal to the learned manifold $\widehat{\mathcal{M}}$.

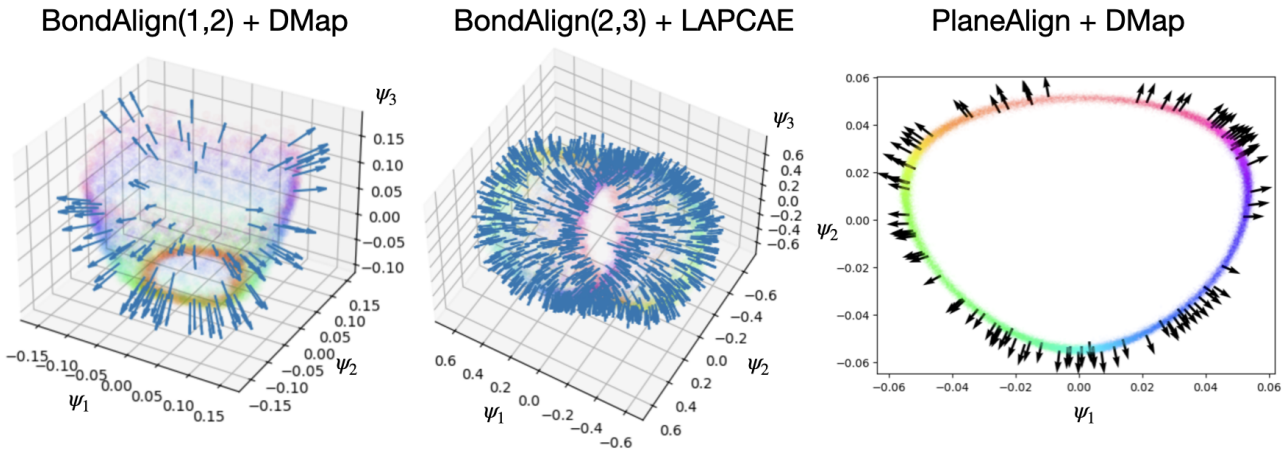


Figure 3.14: Gradients of potential $\nabla\widehat{\Phi}$ learned for BondAlign(1,2), BondAlign(2,3) and PlaneAlign. Here $\widehat{\Phi}$ is learned by minimizing $L_{\text{potential}}$.

3.5.2 Learning collective variable

After acquiring the surrogate potential $\widehat{\Phi}$ that vanishes on the surrogate manifold $\widehat{M}$, we finally train the collective variable $\widehat{\xi}_{\Theta_3}$, given by another neural network with parameters Θ_3 trained to satisfy the orthogonality criterion with respect to $\widehat{\Phi}$. We will proceed to learn CVs following Algorithm 1 only for the two most promising featurizations, BondAlign(2,3) and PlaneAlign. To do so, we consider the surrogate potential $\widehat{\Phi}$ obtained by minimizing $L_{\text{potential}}$ in (3.74). In this case $\widehat{\Phi} \approx \widehat{\Phi}_{\text{SDF}}$. However,

$$\nabla^2\widehat{\Phi}_{\text{SDF}}\nabla\widehat{\Phi}_{\text{SDF}} = 0, \tag{3.75}$$

where $\nabla^2\widehat{\Phi}_{\text{SDF}}$ is the Hessian of the signed distance function. Since $\nabla\widehat{\Phi}$ is the normal vector to $\mathcal{M}$, this implies that the rows of $\nabla^2\widehat{\Phi}$ given by $\nabla\partial_i\widehat{\Phi}$ satisfy the orthogonality condition (OC). Therefore, $\partial_i\widehat{\Phi}$ may be directly used as a collective variable; alternately, a separate encoder $\widehat{\xi}$ may be trained such

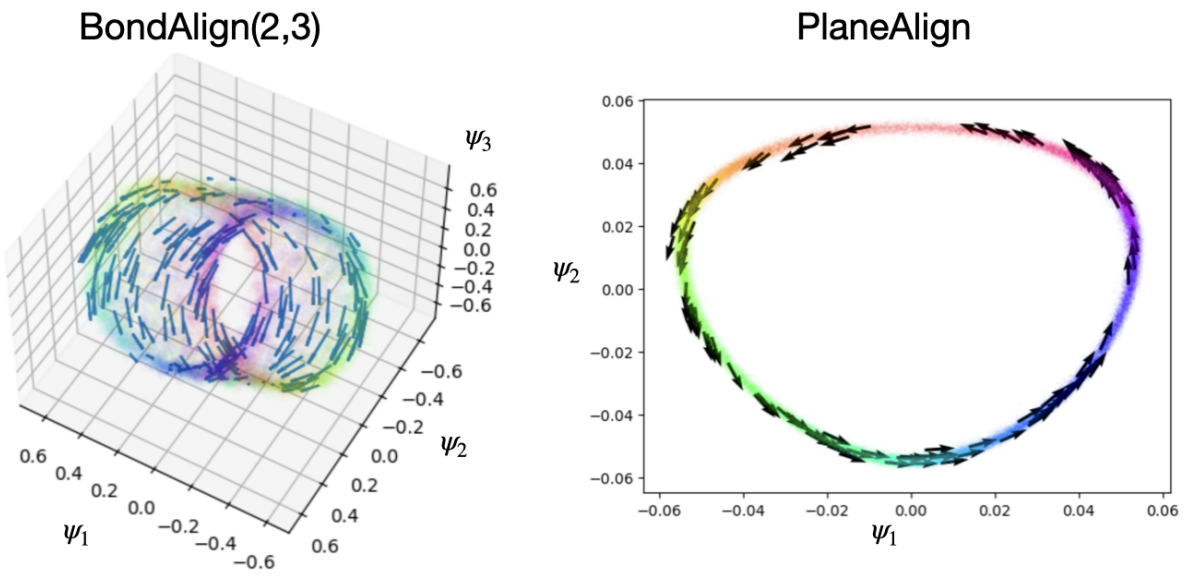


Figure 3.15: Gradients of CVs for BondAlign(2,3) and PlaneAlign.

that the *normalized gradients* of $\widehat{\xi}$ given by $\nabla\widehat{\xi}/\|\nabla\widehat{\xi}\|$ are aligned with the normalized $\nabla\partial_i\widehat{\Phi}$. One way to avoid dividing by $\|\nabla\widehat{\xi}\|$ is to observe that the Cauchy-Schwarz inequality $|\langle\nabla\widehat{\xi}, \nabla\partial_i\widehat{\Phi}\rangle|^2 \leq \|\nabla\widehat{\xi}\|^2\|\nabla\partial_i\widehat{\Phi}\|^2$ is an equality if and only if $\nabla\widehat{\xi}$ and $\nabla\partial_i\widehat{\Phi}$ are parallel. Therefore, a loss function for learning $\widehat{\xi}$ is given by:

$$L_{\text{alignment}}[\Theta_3] = \sum_{i=1}^{n} \|y_i - \xi_{\text{Dec}} \circ \widehat{\xi}(y_i)\|_2^2 + \alpha_{\text{OC}}(|\langle\nabla\widehat{\xi}, \nabla\partial_i\widehat{\Phi}\rangle|^2 - \|\nabla\widehat{\xi}\|^2\|\nabla\partial_i\widehat{\Phi}\|^2)^2. \tag{3.76}$$

In Figure 3.15, we visualize the gradients of $\widehat{\xi}$ for the feature maps BondAlign(2,3) and PlaneAlign which are orthogonal to the surrogate potential $\widehat{\Phi}$ on the learned manifold $\widehat{\mathcal{M}}$. For BondAlign(2,3), we use the loss $L_{\text{alignment}}$ (3.76) with $\partial_1\widehat{\Phi}$, where $\widehat{\Phi}$ is learned by minimizing L_{eikonal}. For PlaneAlign, we find that $\widehat{\xi}(x) = \texttt{arctan2}(x_2, x_1)$ [1] is orthogonal to $\nabla\widehat{\Phi}$ and therefore satisfies (OC). Thus, given an interpretable enough visualization, CVs may be devised by inspection. Interestingly, the CV learned via PlaneAlign correlates significantly with the dihedral angle.

[1] $\texttt{arctan2}(x_2, x_1)$ returns the polar angle of the 2D vector (x_1, x_2) in the range $(-\pi, \pi]$

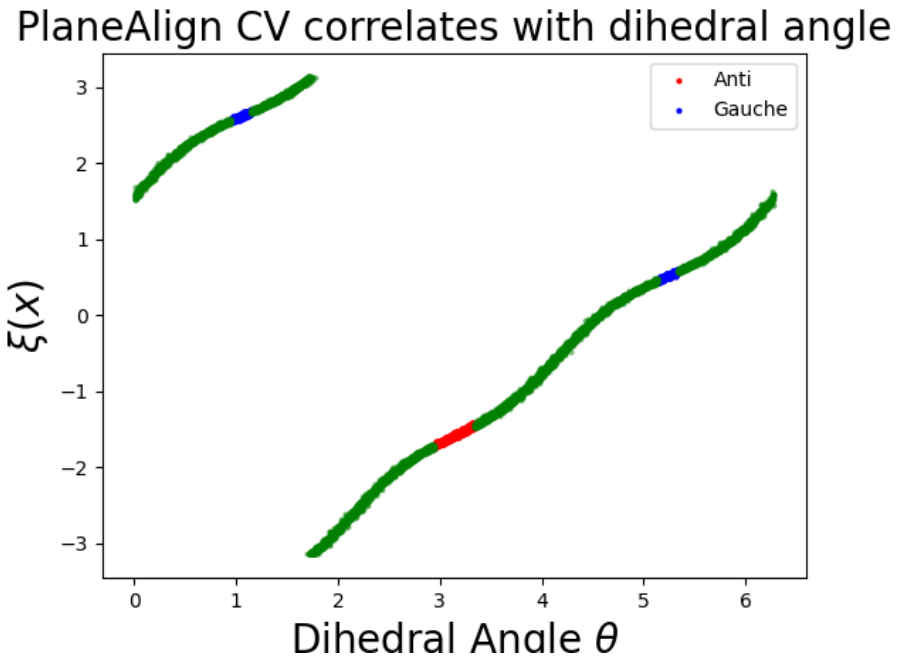


Figure 3.16: We visualize $(\theta(x_i), \xi(x_i))$ for $x_i \in \mathcal{X}$, the training data used for learning ξ. Note that ξ is a nonlinear function of θ–the single discontinuity is due to the use of the `arctan2` function which requires a branch cut at $\theta = 2\pi$.

3.5.3 Computational details

The code for reproducing our results and accessing our trained models has been provided at `https://github.com/ShashankSule/CV_learning_butane`. Below we include details for the numerical computation of the committor function.

Results in Section 3.3. The transition rates in Table 3.1 were computed by numerically solving the boundary value problem (3.42) and then using a quadrature method for computing the integral (3.43). For ξ_1, the dihedral angle, we discretized the elliptic operator $\mathcal{L}$ with a Fourier difference stencil on 10^3 equispaced points in $[0, 2\pi)$. After computing the committor by imposing the relevant boundary conditions, we computed (3.43) using Simpson's rule. For ξ_2, i.e cosine of the dihedral angle, the committor problem amounts to solving the BVP (3.42) in the interval $(\cos(\pi - 0.2), \cos(\pi/3 + 0.1))$

with boundary conditions $q(\cos(\pi - 0.2)) = 0$, $q(\cos(\pi/3 + 0.1)) = 1$. To numerically solve this BVP, we used Chebyshev spectral differentiation, followed by Clenshaw-Curtis quadrature. Finally, for ξ_3, we used a diffusion map-based solver for the committor function proposed for PDEs on manifolds in [ECT22, SEC25]. We then used Monte Carlo integration to compute the rate.

Neural network architectures and training. For each choice of the feature map, we train a diffusion net carrying the data to a surrogate space, then a confining potential vanishing on the surrogate manifold, and finally collective variable(s) which flow orthogonally to the gradients of the confining potential. Below, we enumerate the network architectures, optimizers, and hyperparameters used in training the models.

Feature map	**Model**	**# neurons**	**Activation**	**Optimizer & lr**	**Training epochs**	**Hyper-parameters**
BondAlign (2,3)	LAPCAE	$[12, 32, 32, 32, 4]$	Tanh	Adam, 1e-4	500	$\alpha^{Enc}_{LAPCAE} = 0.5$, $\alpha_{LAPCAE} = 2.0$
	$\widehat{\Phi}$	$[3, 30, 45, 32, 32, 1]$	$x + \sin^2(x)$	Adam, 1e-3	1000	$\alpha_{Zero} = 1.0$ $\alpha_{normals} = 0.0$
	$\widehat{\xi_1}$	$\widehat{\xi_1} = \partial_1 \widehat{\Phi}$	$1 + \cos(2x)$	No training	N.A.	N.A.
	$\widehat{\xi_2}$	$[3, 30, 45, 32, 32, 1]$	$x + \sin^2(x)$	Adam, 1e-2	500	$\alpha_{OC} = 1.0$
PlaneAlign	DNet	$[42, 32, 32, 32, 4]$	Tanh	Adam, 1e-4	500	$\alpha_{DNet} = 1.0$
	$\widehat{\Phi}$	$[2, 30, 45, 32, 32, 1]$	$x + \sin^2(x)$	Adam, 1e-3	1000	$\alpha_{Zero} = 1.0$ $\alpha_{normals} = 0.0$

Table 3.2: Architectures, optimizers, and hyperparameters for the neural networks trained in the paper.

The Free energy and the diffusion tensor. In Section 3.5 and Table 3.3 we study transition rates for the anti-gauche transition from CVs learned via the feature maps BondAlign(2,3) and PlaneAlign. In Figures 3.18 and 3.19 we present the free energy landscapes and diffusion tensors for CVs obtained via Algorithm 3 from data post-processed via these two feature maps.

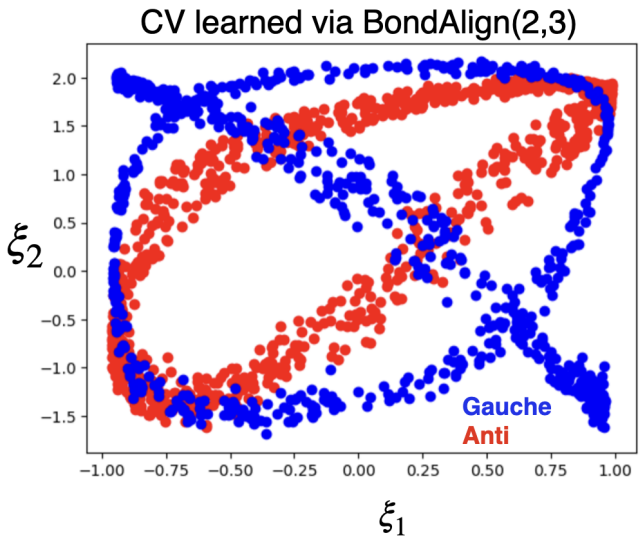


Figure 3.17: When visualizing data in the CV space learned with the BondAlign(2,3) feature map, the anti (red) and gauche (blue) states are not separated.

3.5.4 Reproducing transition rates

Using CVs $\xi := \widehat{\xi} \circ \Psi$ learned from Algorithm 3 we now reproduce the transition rate ν_{AB} for the anti-gauche transition from the set

$$A = \{|\theta - \pi| \leq 0.2\} \quad \text{to} \quad B = \{|\theta - \pi/3| \leq 0.1\} \sqcup \{|\theta - 5\pi/3| \leq 0.1\}.$$

We will do so by computing $\tilde{\nu}_{AB}$ via (3.43). This necessitates computing the free energy f and the diffusion tensor $M(z)$. We compute f via *well-tempered metadynamics* [BBP08] and M via the spring force method [MFVEC06] respectively.

Our results are presented in Table 3.3, where we organize the machine-learned collective variables (ML CVs) by the underlying feature map. We find that although the BondAlign(2,3) feature map results in a coherent manifold, the CVs learned using such a manifold do not separate the metastable states A and B (see Figure 3.17). However, the CV learned using PlaneAlign corresponds nearly ex-

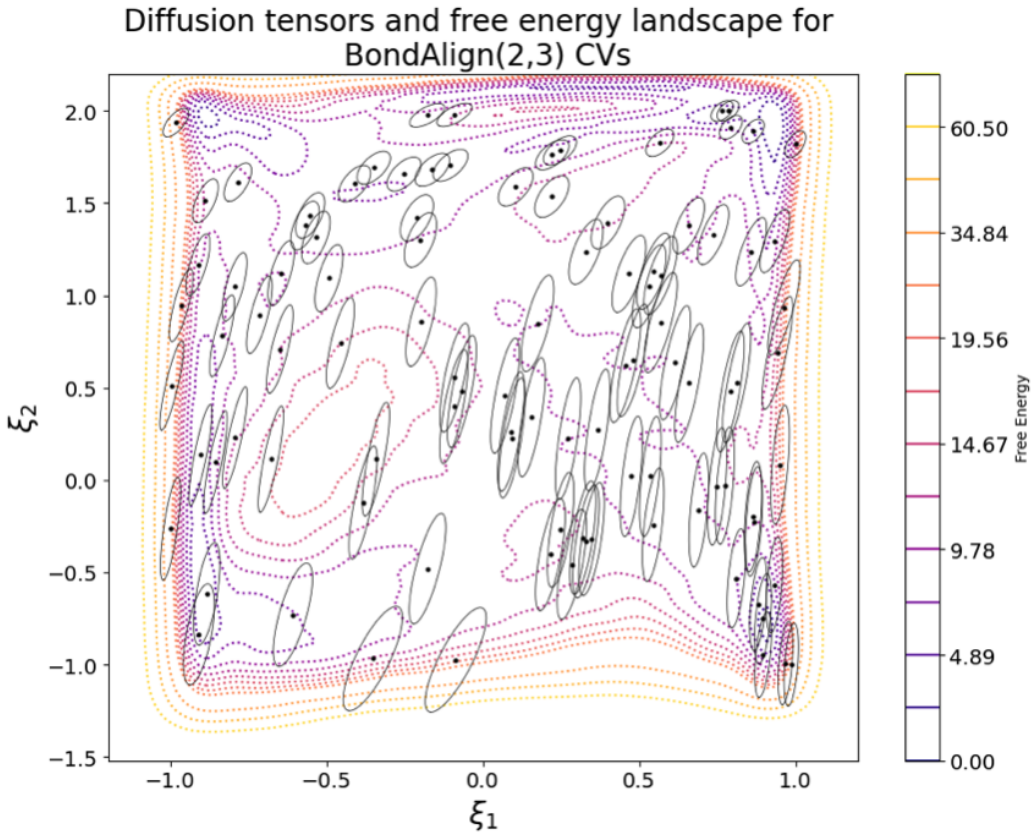


Figure 3.18: Free energy and diffusion tensor for BondAlign(2,3) with LAPCAE as the global coordinate chart parameterizing the residence manifold. Here, the contour lines represent the level sets of the free energy, and the black dots represent points sampled in (ξ_1, ξ_2) space. The ellipse around each such point z represents the diffusion tensor $M(z)$ where the principal axes of the ellipse correspond to the eigenvectors of M and the principal lengths correspond to the eigenvalues.

actly to the dihedral angle, and therefore separates metastable states. In CV space these are identified with the following intervals:

$$A = [-1.75, -1.41],\ B = [0.43, 0.60] \sqcup [2.54, 2.71]. \tag{3.77}$$

Using these and applying formula (3.43) led to a transition rate with less than ten percent relative error. The free energy and diffusion tensors used for computing the transition rates are provided in Figure 3.19. For completeness, we visualize the free energy landscape and diffusion tensors for the

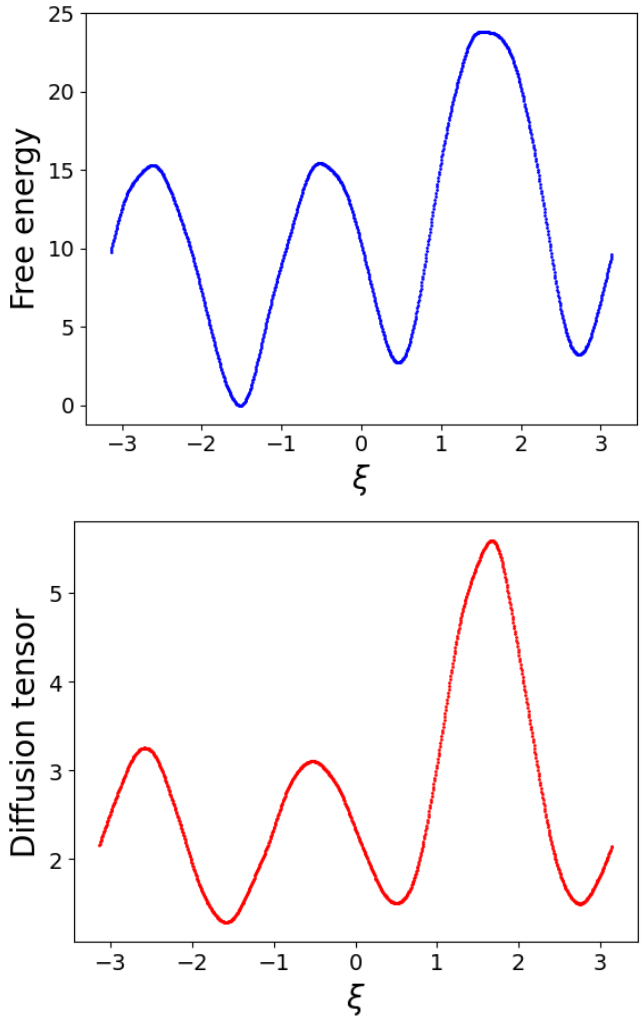


Figure 3.19: Free energy and diffusion tensor for PlaneAlign.

two-dimensional CV learned using BondAlign(2,3) in Figure 3.18 even though we do not compute the anti-gauche transition rate using this CV. Notably the diffusion tensor for the CV learned via PlaneAlign is different than the dihedral angle.

3.5.5 Attributing differences in rates to scale separation

We conduct a spectral analysis of the generators of the collective variables in Table 3.3. In particular, we form the generator $\mathcal{L}$ of the effective dynamics and compute its eigenvalues to visualize their spectral gaps. We visualize the results in Figure 3.20. Here we see that $\cos\theta$ has effectively

Collective variable	Transition rate (ps^{-1})
Reference	$1.13 \pm 0.08 \times 10^{-2}$
Dihedral angle θ	1.41×10^{-2}
$(\sin\theta, \cos\theta)$	1.19×10^{-2}
$\cos\theta$	1.52×10^{-2}
ξ, BondAlign(2,3)	Metastable states not separated
ξ, PlaneAlign	1.25×10^{-2}

Table 3.3: Transition rates for the anti-gauche transition in both classical and machine-learned collective variables. We organize the ML CVs ξ according to the relevant feature map in use.

no large gap between eigenvalues, with the eigenvalues decaying at roughly the same rate. However, the other three collective variables that feature in Table 2 in the chapter show a clear scale separation between the second and third eigenvalue. This observation corroborates the relative errors in the transition rates–the greater the scale separation, the better the transition rate estimate.

3.5.6 Interpreting the collective variables and embeddings

The PlaneAlign CV differs from the dihedral angle in its diffusion tensor. To further discover the underlying physical reason governing this difference, we conduct an analysis to identify the physical interpretation of our learned embeddings and CVs in terms of fundamental molecular degrees of freedom (dihedral angles, bond angles, and bond lengths) below. In particular, butane has 13 bond

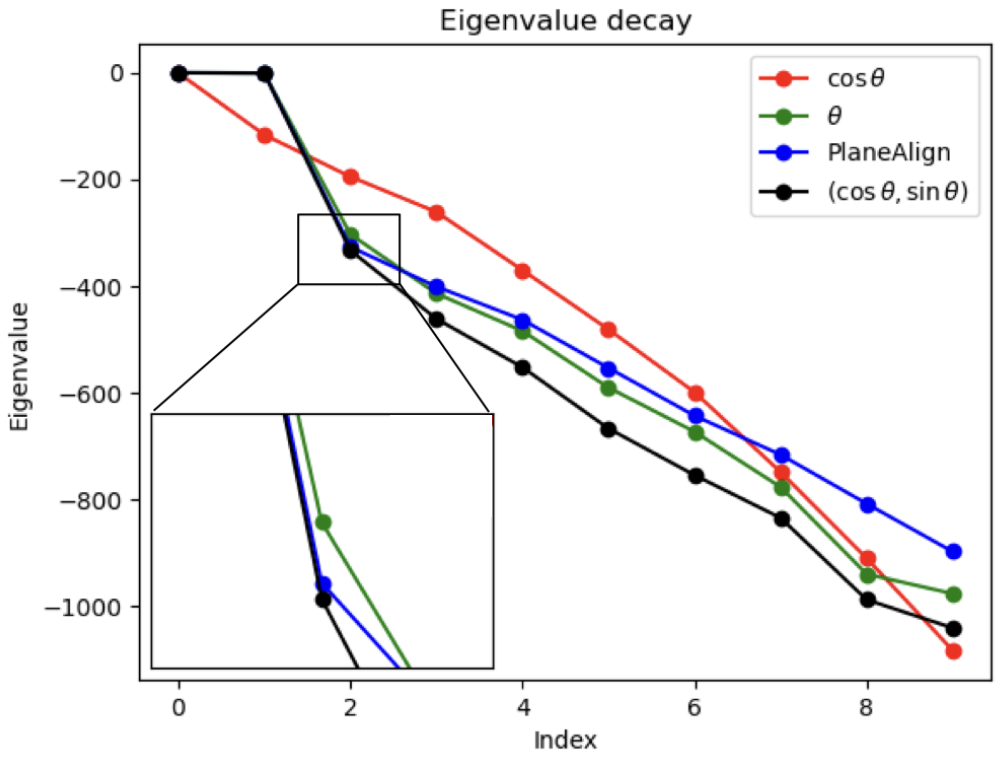


Figure 3.20: We form the generator $\mathcal{L} := (\beta^{-1}\nabla \cdot M - M\nabla F \cdot \nabla) + \mathsf{tr}(M\nabla\nabla)$ of the effective dynamics implied by the different collective variables and compute the top 10 eigenvalues. We visualize these ordered eigenvalues for the different collective variables.

lengths, 24 bond angles (all angles between bonded triples of atoms $A - B - C$), and 24 dihedral angles (the dihedral angle formed by any 4-tuple of bonded atoms). After embedding the angular coordinates through sines and cosines (in order to make them continuous), we obtain a total of 115 internal features f_j. We now take each embedding coordinate ψ_k and attempt to describe it using the internal features by fitting a linear model. In particular, we will approximate:

$$\psi_k(x_i) \approx \widehat{\psi}(x_k) = \beta_0 + \beta_1 f_1(x_i) + \cdots + \beta_{115} f_{115}(x_k). \tag{3.78}$$

In Table 3.4, we provide the results of the regression. We also fit the collective variables (ξ_1, ξ_2) learned using the feature map BondAlign(2,3). We measure the goodness-of-fit of the linear model using the

coefficient of determination R^2 defined below:

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(\psi_k(x_i) - \widehat{\psi}(x_i))^2}{\sum_{i=1}^{n}(\psi_k(x_i) - \overline{\psi_k})^2}, \quad \overline{\psi_k} = \frac{1}{n}\sum_{i=1}^{n}\psi_k(x_i). \tag{3.79}$$

Although this approach has its limitations (due to, for instance, fitting of a linear model), we nonetheless find that:

- The top two diffusion coordinates for the PlaneAlign feature map are well-described by the internal coordinates (with $R^2 = 0.99, 0.98$); moreover $70 - 80\%$ of the regression is carried by the dihedral angle θ in the carbon backbone (this is also clear from Figure 3.16 where we visualize $\arctan 2(\psi_2, \psi_1)$ against θ. Interestingly, in the first two diffusion coordinates, nearly 20% of the regressor norm is contributed to by the $H_{21} - C_2 - C_3 - H_{31}$ and $H_{11} - C_1 - C_2 - C_3$ dihedrals, thus re-emphasizing the role of the hydrogen atoms in meaningfully influencing the diffusion embedding. Our experiments with butane show that we cannot reproduce the transition rate by neglecting the hydrogens–this table clearly shows that the hydrogens are important, especially as the dihedral angle between the carbons and the terminal methyl group hydrogens contribute to the variation in the diffusion embedding and therefore the collective variable.

- In the GramMatrixCarbon embedding, the top diffusion coordinate is parameterized by the cosine of the $C_1 - C_2 - C_3 - C_4$ dihedral angle, which corroborates with the right panel of Figure 3.12. However, as we go deeper into the spectrum we see that there is a transition from rotations to vibrations–the R^2 for the fourth and fifth coordinates remains $> 99\%$ but the dominant features are bond angles and bond lengths in the carbon backbone. On the other hand, the GramMatrix embedding depends quite heavily on the presence of hydrogens, with the dihedral angles

between the terminal hydrogens through the carbon backbone emerging as important features for describing the diffusion coordinates.

- While the CV learned via PlaneAlign is largely accounted for by the $C_1 - C_2 - C_3 - C_4$ dihedral angle, the CV learned via BondAlign(2,3) is not as well-described by the internal coordinates ($R^2 < 0.01$). This paints a rather complicated picture for interpreting the physical meaning behind the CVs involved. In particular, the collective variable so identified depends on the principal directions of the surrogate manifold. However, these principal directions may not always correspond to a well-known internal physical variable.

Table 3.4: Regression results for different feature maps and diffusion coordinates. Here Dmap evec. index indicates the jth diffusion map coordinate which we fit against a linear model of the internal features f_i. Moreover, $0 \leq R^2 \leq 1$ is the explained variance of this linear regression, which indicates how well the linear model explains the diffusion map embedding. $R^2 \approx 1$ indicates a high accuracy of the linear model.

Feature Map	Dmap evec. index	R^2	Top Feature	%
PlaneAlign	1	0.9985	$\sin(\text{DihedralAngle}(C_1, C_2, C_3, C_4))$	85.28%
			$\cos(\text{DihedralAngle}(H_{21}, C_2, C_3, H_{31}))$	20.61%
			$\cos(\text{DihedralAngle}(H_{22}, C_2, C_3, H_{32}))$	16.63%
	2	0.9986	$\cos(\text{DihedralAngle}(C_1, C_2, C_3, C_4))$	83.86%
			$\cos(\text{DihedralAngle}(H_{11}, C_1, C_2, C_3))$	17.53%
			$\sin(\text{DihedralAngle}(H_{22}, C_2, C_3, H_{31}))$	14.27%
	3	0.9991	$\cos(\text{DihedralAngle}(C_2, C_3, C_4, H_{42}))$	52.22%
			$\sin(\text{DihedralAngle}(C_2, C_3, C_4, H_{43}))$	42.00%

Continued on next page

Table 3.4 – continued from previous page

Feature Map	Coord.	R^2	Top Feature	%
			$\cos(\text{DihedralAngle}(C_2, C_3, C_4, H_{41}))$	38.40%
	4	0.9986	$\cos(\text{DihedralAngle}(C_2, C_3, C_4, H_{43}))$	47.16%
			$\cos(\text{DihedralAngle}(C_2, C_3, C_4, H_{42}))$	39.46%
			$\sin(\text{DihedralAngle}(C_2, C_3, C_4, H_{42}))$	37.82%
	5	0.1346	$\sin(\text{DihedralAngle}(H_{12}, C_1, C_2, C_3))$	28.54%
			$\cos(\text{DihedralAngle}(H_{11}, C_1, C_2, C_3))$	27.88%
			$\sin(\text{DihedralAngle}(H_{31}, C_3, C_4, H_{41}))$	24.59%
GramMatrixCarbon	1	0.9976	$\cos(\text{DihedralAngle}(C_1, C_2, C_3, C_4))$	88.75%
			$\cos(\text{DihedralAngle}(H_{21}, C_2, C_3, C_4))$	14.57%
			$\cos(\text{DihedralAngle}(H_{11}, C_1, C_2, H_{22}))$	13.06%
	2	0.4749	$\cos(\text{DihedralAngle}(H_{13}, C_1, C_2, H_{21}))$	28.92%
			$\sin(\text{DihedralAngle}(C_1, C_2, C_3, H_{32}))$	28.19%
			$\cos(\text{DihedralAngle}(H_{31}, C_3, C_4, H_{43}))$	25.40%
	3	0.6227	$\sin(\text{DihedralAngle}(C_1, C_2, C_3, H_{32}))$	28.83%
			$\cos(\text{DihedralAngle}(H_{13}, C_1, C_2, H_{21}))$	28.54%
			$\sin(\text{DihedralAngle}(H_{11}, C_1, C_2, H_{21}))$	26.18%
	4	0.9991	$\cos(\text{BondAngle}(C_2, C_3, C_4))$	59.16%
			$\cos(\text{BondAngle}(C_1, C_2, C_3))$	58.86%
			$\text{BondLength}(C_3, C_4)$	34.20%

Continued on next page

Table 3.4 – continued from previous page

Feature Map	Coord.	R^2	Top Feature	%
	5	0.9943	BondLength(C_3, C_4)	42.58%
			BondLength(C_2, C_3)	41.31%
			BondLength(C_1, C_2)	39.80%
GramMatrix	1	0.8424	$\cos$(DihedralAngle$(H_{31}, C_3, C_4, H_{41})$)	30.43%
			$\sin$(DihedralAngle$(H_{31}, C_3, C_4, H_{43})$)	28.75%
			$\sin$(DihedralAngle(H_{21}, C_2, C_3, C_4))	28.65%
	2	0.8419	$\cos$(DihedralAngle$(H_{31}, C_3, C_4, H_{42})$)	32.12%
			$\cos$(DihedralAngle$(H_{32}, C_3, C_4, H_{41})$)	31.20%
			$\cos$(DihedralAngle$(H_{31}, C_3, C_4, H_{41})$)	28.24%
	3	0.8409	$\cos$(DihedralAngle$(H_{32}, C_3, C_4, H_{43})$)	29.27%
			$\sin$(DihedralAngle$(H_{31}, C_3, C_4, H_{42})$)	26.28%
			$\sin$(DihedralAngle$(H_{31}, C_3, C_4, H_{43})$)	25.83%
	4	0.8351	$\cos$(DihedralAngle(H_{11}, C_1, C_2, C_3))	32.06%
			$\cos$(DihedralAngle$(H_{13}, C_1, C_2, H_{22})$)	30.85%
			$\sin$(DihedralAngle$(H_{12}, C_1, C_2, H_{22})$)	26.08%
	5	0.9763	$\cos$(DihedralAngle(C_1, C_2, C_3, C_4))	31.92%
			$\sin$(DihedralAngle$(H_{13}, C_1, C_2, H_{21})$)	31.61%
			$\sin$(DihedralAngle$(H_{11}, C_1, C_2, H_{21})$)	31.44%
	1	0.0117	$\cos$(DihedralAngle$(H_{32}, C_3, C_4, H_{42})$)	32.97%

Continued on next page

Table 3.4 – continued from previous page

Feature Map	Coord.	R^2	Top Feature	%
			$\sin(\text{DihedralAngle}(H_{32}, C_3, C_4, H_{42}))$	31.06%
			$\sin(\text{DihedralAngle}(H_{32}, C_3, C_4, H_{41}))$	26.74%
	2	0.0135	$\cos(\text{DihedralAngle}(H_{31}, C_3, C_4, H_{41}))$	35.56%
			$\cos(\text{DihedralAngle}(H_{21}, C_2, C_3, H_{32}))$	25.55%
			$\sin(\text{DihedralAngle}(H_{32}, C_3, C_4, H_{42}))$	24.65%
	3	0.9871	$\cos(\text{DihedralAngle}(C_1, C_2, C_3, C_4))$	69.56%
			$\sin(\text{DihedralAngle}(H_{22}, C_2, C_3, C_4))$	25.56%
			$\cos(\text{DihedralAngle}(H_{31}, C_3, C_4, H_{42}))$	18.53%
	4	0.0183	$\sin(\text{DihedralAngle}(H_{22}, C_2, C_3, H_{31}))$	34.83%
			$\sin(\text{DihedralAngle}(H_{21}, C_2, C_3, H_{31}))$	33.59%
			$\sin(\text{DihedralAngle}(H_{22}, C_2, C_3, C_4))$	32.87%
	5	0.0135	$\cos(\text{DihedralAngle}(H_{11}, C_1, C_2, H_{21}))$	38.32%
			$\cos(\text{DihedralAngle}(C_2, C_3, C_4, H_{43}))$	25.32%
			$\sin(\text{DihedralAngle}(C_2, C_3, C_4, H_{42}))$	23.32%
BondAlign(2,3)	1	0.0118	$\cos(\text{DihedralAngle}(H_{32}, C_3, C_4, H_{43}))$	36.19%
			$\cos(\text{DihedralAngle}(H_{21}, C_2, C_3, H_{31}))$	30.17%
			$\sin(\text{DihedralAngle}(H_{32}, C_3, C_4, H_{42}))$	27.88%
	2	0.0101	$\sin(\text{DihedralAngle}(H_{21}, C_2, C_3, H_{31}))$	26.81%
			$\sin(\text{DihedralAngle}(H_{31}, C_3, C_4, H_{43}))$	22.38%

Continued on next page

Table 3.4 – continued from previous page

Feature Map	Coord.	R^2	Top Feature	%
			$\cos(\text{DihedralAngle}(H_{21}, C_2, C_3, C_4))$	22.27%
	3	0.0127	$\sin(\text{DihedralAngle}(H_{31}, C_3, C_4, H_{43}))$	27.85%
			$\cos(\text{DihedralAngle}(H_{32}, C_3, C_4, H_{42}))$	26.15%
			$\sin(\text{DihedralAngle}(H_{32}, C_3, C_4, H_{42}))$	26.06%
	4	0.0107	$\sin(\text{DihedralAngle}(H_{11}, C_1, C_2, H_{22}))$	39.01%
			$\sin(\text{DihedralAngle}(H_{11}, C_1, C_2, H_{21}))$	29.23%
			$\sin(\text{DihedralAngle}(H_{21}, C_2, C_3, H_{32}))$	27.94%
	5	0.7706	$\sin(\text{DihedralAngle}(H_{21}, C_2, C_3, H_{31}))$	34.50%
			$\sin(\text{DihedralAngle}(H_{12}, C_1, C_2, H_{22}))$	33.52%
			$\cos(\text{DihedralAngle}(H_{11}, C_1, C_2, H_{21}))$	28.07%
BondAlign(2,3) CVs	ξ_1	0.0098	$\sin(\text{DihedralAngle}(H_{21}, C_2, C_3, H_{32}))$	37.33%
			$\cos(\text{DihedralAngle}(H_{22}, C_2, C_3, H_{32}))$:	33.27%
			$\cos(\text{DihedralAngle}(H_{21}, C_2, C_3, C_4))$:	29.38%
	ξ_2	0.0123	$\cos(\text{DihedralAngle}(H_{21}, C_2, C_3, H_{31}))$:	35.28%
			$\cos(\text{DihedralAngle}(H_{13}, C_1, C_2, C_3))$	26.27%
			$\sin(\text{DihedralAngle}(H_{13}, C_1, C_2, H_{22}))$	23.73%

3.6 Application to alanine dipeptide

To illustrate the applicability of our general algorithm beyond the butane example, we apply our residence manifold learning procedure to alanine dipeptide, a minipeptide of two amino acid residues,

in vacuum. It is well-known [ECT22] that the two dihedral angles θ_1, θ_2 of the two alanine residues can be used to coarse-grain the atomic coordinates and obtain a free energy surface, enabling the definition of the metastable states C7ax, C7eq which appear as the free energy minima (see Figure 3.21).

Findings on alanine dipeptide:

1. The diffusion map embedding after featurization via PlaneAlign in Figure 3.22 (with alignment of the α-Carbon and its neighbouring Carbon and Nitrogen) is a 2D surface in 3D, whose angular direction is parameterized via the θ_2 angle–this clearly demonstrates the utility of PlaneAlign as a useful featurization for diffusion maps beyond the butane example.

2. We also perform the same exercise as butane in comparing the Gram matrix embeddings with and without the hydrogen atoms for alanine dipeptide. Here we see, in the left panel of Figure 3.23, that with the hydrogen atoms, the two metastable states are separable in the diffusion map embedding. However, they overlap greatly in the diffusion map embedding of the gram matrices with the hydrogens removed (see right panel of Figure 3.23). This conclusion is consistent with the one seen for butane, where the embedding without the hydrogens combined the two disjoint configurations of the gauche state.

3.7 Discussion

In this chapter, we have made several contributions to the theory and application of quantitative coarse-graining. We established the equivalence between the orthogonality condition (OC), which naturally emerges as a mechanism for reducing the modeling error measured via the relative entropy, and the Projected Orthogonality Condition (POC) (Proposition 3.3.1), which ensures that the pathwise

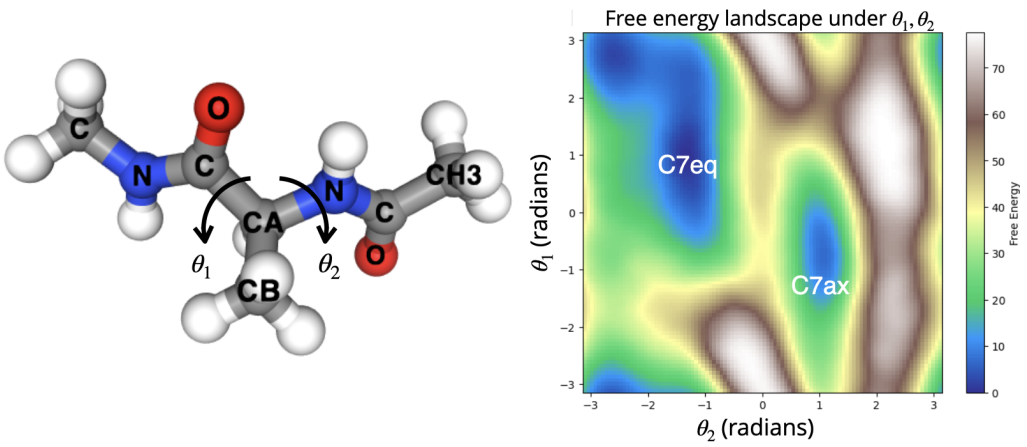


Figure 3.21: Left: A schematic of the alanine dipeptide molecule, with the dihedral angles θ_1, θ_2 visualized. Here the α-Carbon has been denoted as CA. Right: The free energy surface under coarse graining by the 2D collective variable (θ_1, θ_2) reveals two minima– C7eq, C7ax separated by a free energy barrier.

distance remains small. Through our study of butane, we also showed that Assumption 3.2.2 demanding that $D\xi D\xi^\top \geq \delta_\xi^2 I_d$ can be relaxed to accommodate rank-deficient but non-degenerate collective variables (CVs), thereby broadening the applicability of these conditions.

We introduced a framework summarized in Algorithm 3 inspired by the orthogonality condition (OC), to construct CVs that flow along the surrogate manifold $\mathcal{M}$. A crucial component of our approach is the residence manifold learning algorithm (Algorithm 7), which carefully integrates several tools from manifold learning to construct a hypersurface representation. This structured approach facilitates an efficient computation of the normal vector, allowing for the direct construction of a CV whose gradients lie in the tangent space.

The significance of the feature map. The key emergent theme from this approach is the extreme importance of choosing the correct feature map for post-processing the data. The existence of several plausible but structurally different surrogate manifolds suggests that a more intricate design of feature maps beyond standard techniques such as RMSD alignment or pairwise distances can provide significant accelerations in downstream tasks such as transition rate estimation conducted in

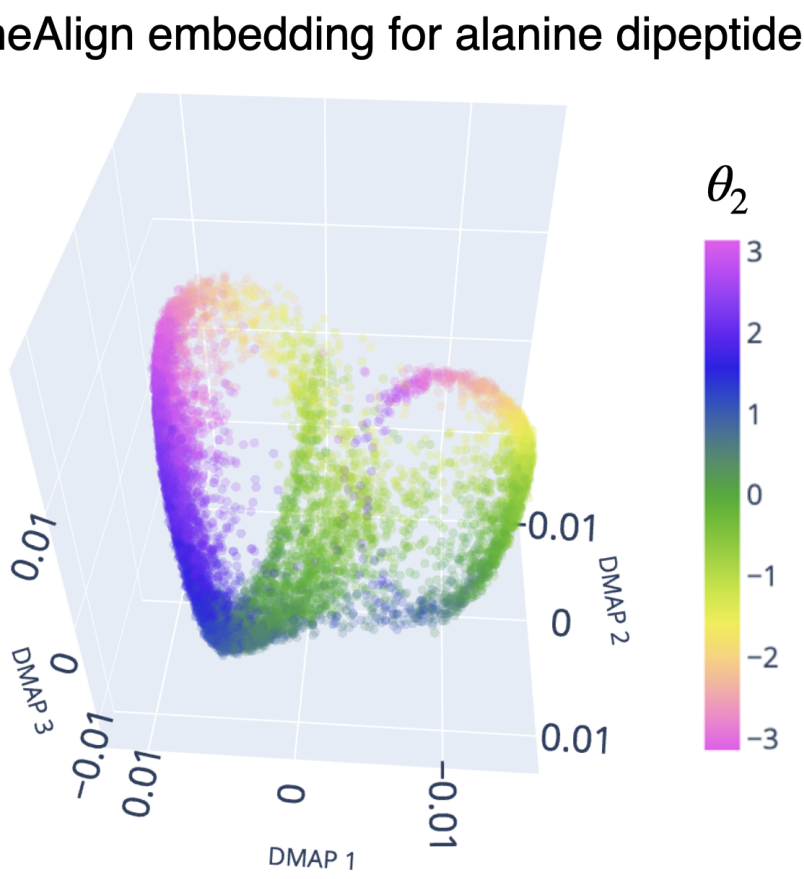


Figure 3.22: The diffusion map embedding on data featurized via PlaneAlign applied to CA and its adjacent carbon and nitrogen atoms reveals a 3D cylinder with angular direction parameterized by the θ_2 angle.

Section 3.5.4. In particular, our hand-crafted PlaneAlign feature map causes the diffusion net embedded to be parameterized by the dihedral angle θ. This parameterization is duly recovered by the collective variable ξ leading to a successful reproduction of the anti-gauche transition rate (see Table 3.3). Furthermore, we provided a heuristic function HyperSurface, (3.66), which combines Laplacian coordinates to embed diffusion map manifolds as hypersurfaces irrespective of the choice of feature map (see Figures 3.11 and 3.12). However, HyperSurface remains purely a heuristic because of (1) false negatives (for instance, PlaneAlign attains a low 2D score but is a good 2D embedding) and (2) self-intersections despite functional independence. We resolved the second issue by simultaneously promoting functional independence in Laplacian eigenfunctions through a neural network loss (3.72)

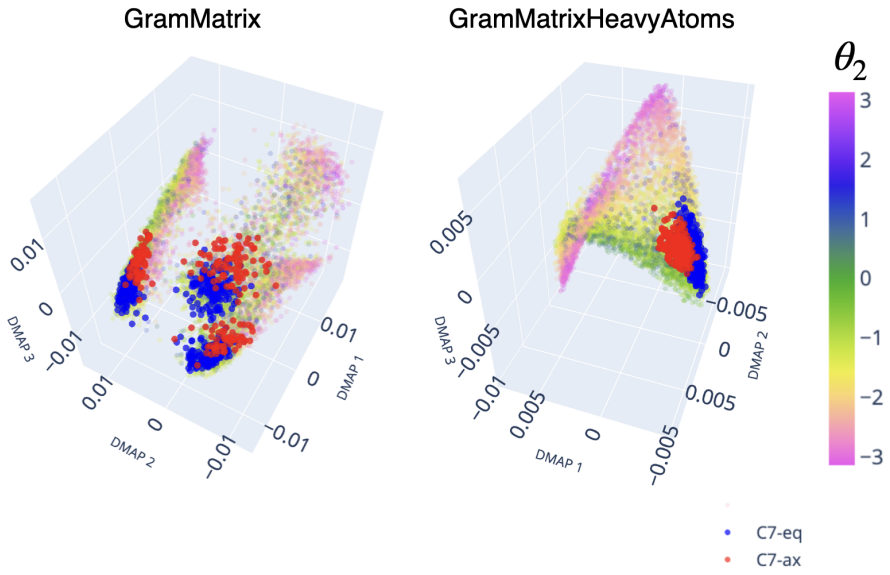


Figure 3.23: Left: The diffusion map embedding on data featurized via the Gram matrix formed by all-atoms. Right: The diffusion map embedding on data featurized via the Gram matrix formed by only the heavy atoms. Note that the metastable states slightly overlap but are still linearly separable in the embedding with hydrogens.

leading to LAPCAE, enabling a well-structured embedding of the surrogate manifold 3.13.

Limitations and scope. Our CV learning algorithm essentially enforces the orthogonality condition (OC)–that the large variations in the CV occur along the principal directions of the data at each point. While this is a general geometric principle that goes beyond the underlying physical model governing the data (such as Langevin or Hamiltonian dynamics), the ability of this principle in reproducing faithful effective dynamics has only been theoretically demonstrated in the limited cases of explicit scale separation in Equation (3.6). Nonetheless, here we have shown that (OC) can be used to train collective variables that reproduce rates in other models governing the data–and therefore holds potential in applications to more complicated systems and alternative stochastic dynamics. Moreover, in addition to the detailed case study of the butane molecule and the feasibility of the residence manifold learning algorithm for alanine dipeptide presented here, CVs trained to satisfy the (OC) have also been proposed for Lennard-Jones systems with permutational symmetries [YSLC25], with the

committor on the CV space being a good reaction coordinate for the forward flux sampling algorithm. However, both these approaches have involved a significant amount of system-based tailoring. For instance, in the butane molecule, we resort to several bespoke strategies for featurizing the data and obtaining the residence manifold–such handcrafting limits the scope of the various steps in Algorithm 3 to reasonably well-understood systems. Thus, while the general idea behind our algorithm (i.e the (OC)) is generalizable to alternative and larger systems, the individual steps of our algorithm require a significant degree of supervision and tailoring. Generalizing these techniques to broader classes of systems is an active research goal.

Additionally, in our framework, the orthogonality condition (OC) is imposed on the surrogate manifold rather than directly in the all-atom space $\mathbb{R}^N$. While it is theoretically possible to define an orthogonality condition in $\mathbb{R}^N$ via a parameterization $\xi = \widehat{\xi} \circ \Psi$ and a function $\Phi = \widehat{\Phi} \circ \Psi$, we found this approach impractical. The presence of numerous ways to satisfy the constraint $\nabla\xi \cdot \nabla\Phi = 0$ in $\mathbb{R}^N$ introduces excessive local minima in the loss function, making the learning of ξ less robust.

Can the surrogate manifold be a CV? Given the ability of diffusion maps to identify the slow manifold of the overdamped Langevin dynamics (1.51), the learned manifold coordinates Ψ could, in principle, serve as CVs. Similar ideas have been implemented in [Ryd24] and [SPTT18] where CVs are directly designed to optimize spectral criteria. Here we use eigenfunctions to reduce the problem to a surrogate space $\mathbb{R}^{D+1}$ and then apply the relevant CV criteria in this surrogate space. We emphasize that enforcing the orthogonality condition (OC) remains beneficial despite the complexity reduction afforded by the map Ψ. In particular, the manifold coordinates Ψ may still be high-dimensional depending on the choice of embedding dimension $D + 1$, making direct usage impractical. Moreover, the orthogonality condition (OC) provides an implicit way to encode functional dependence on the intrinsic parameterization of the surrogate manifold without requiring a coordinate chart in $\mathbb{R}^{D+1}$.

Specifically, locally the surrogate manifold has $D + 1$ degrees of freedom and can be represented as the graph of a function. However, recovering such a function for every point is highly cumbersome and the orthogonality condition (OC) provides a way to implicitly define ξ on these intrinsic degrees of freedom.

Future applications and implementational improvements. The above insights and limitations open numerous directions of future work. One primary direction is to apply our methodology towards studying the conformational dynamics of synthetic peptide chains such as chignolin [HK11] or AIB9, and to the resolution of more advanced challenges such as cryptic pocket discovery or [WHT22] nucleation [WMZT24]. Given the challenges of defining the orthogonality condition directly in $\mathbb{R}^N$, future work could explore alternative formulations that mitigate the issue of excessive local minima in the loss function. This may involve developing regularization techniques or optimizing over constrained subspaces that better capture physically meaningful CVs.

Open theoretical questions. In addition to applications in molecular dynamics, here we also underline some open questions in quantitative coarse-graining and manifold learning prompted by our approach. With regards to quantitative coarse-graining, in Section 3.3, we have demonstrated empirically that the diffusion tensor $M(z)$ need not be full rank to reproduce transition rates. This assumption is ubiquitous in most recent works on quantitative coarse-graining, thus leaving an intriguing avenue to extend the existing theory toward error estimates for degenerate diffusions. Moreover, with regards to manifold learning, we make use of group invariant diffusion maps via a group invariant feature map in Algorithm 7, thus furthering the project started in [RCS23, HK23] of encoding group structure into spectral embeddings. These works also use a group invariant kernel to perform diffusion maps but do so by imparting group invariance via averaging against the Haar measure of the underlying Lie group. This necessitates generating the orbit of every element in the input point cloud. Here we

have presented an alternative approach that does not require these orbits by using a feature map before applying the kernel. A comparison and possible integration of both approaches is therefore interesting and may hold potential for accelerating the accuracy and expressibility of diffusion maps for MD data.

3.8 Conclusion

Our work advances the theory and practice of quantitative coarse-graining by elucidating the role of the orthogonality condition in reducing modeling error and preserving transition rates in overdamped Langevin dynamics. We introduced a learning framework that constructs collective variables by imposing the orthogonality condition on a surrogate space defined via manifold-based representations. We demonstrated that group invariant featurization is critical for defining such representations and therefore heavily affects coarse-graining performance. Through our butane case study, we demonstrated the role of analytical conditions from quantitative coarse-graining and provided practical heuristics, such as HyperSurface and LAPCAE, to address manifold learning challenges in our proposed algorithm. While our approach currently remains limited to simple systems and surrogate spaces, it opens up rich avenues for applying these techniques to more complex biomolecular dynamics and exploring fundamental questions in coarse-graining and manifold learning.

3.9 Acknowledgments

The content of this chapter was partially supported by AFOSR MURI grant FA9550-20-1-0397 and by NSF REU grant DMS-2149913. The author would like to thank Akash Aranganathan and Dr. Eric Beyerle for helpful discussions regarding MD simulations in the high-friction regime.

Chapter 4: Learning the low-dimensional structure of magnetic resonance relaxometry measurements in the human brain

4.1 Introduction

Myelin is a lipid-rich substance that surrounds and insulates nerve axons in the central and peripheral nervous systems, facilitating the rapid and high-fidelity transmission of electrical signals. Therefore, quantification and mapping of myelin provides valuable insights into brain white matter. In addition, myelination in the central nervous system (CNS) is increasingly recognized as being of central importance in a spectrum of neurodegenerative disorders, including Alzheimer's disease and cognitive impairment [FKB+18], making it an important biomarker for neurodegenerative disorders such as Alzheimer's disease, multiple sclerosis (MS), and traumatic brain injury (TBI) [LVM+04, DGVD10, MWA+94, KMJB+12, ML16]. Consequently, the problem of quantification and mapping of the myelin water fraction (MWF) using MRI remains a very active area of investigation, even after its introduction through the pioneering work of MacKay more than 20 years ago [MWA+94].

There are a number of current methods for MWF mapping; relaxation-based methods rely upon use of an MRI signal model that incorporates T2 values of both myelin-associated water and less-restricted water. In this chapter we address multiple gradient-echo or spin-echo pulse sequences [VLL+06, Deo10, OHL+07] where the signal observed from each voxel at an echo time t follows the following exponential model:

$$s(t) = \int_{\mathbb{R}} e^{-t/T_2} p(T_2)\, dT_2. \tag{4.1}$$

Here $p(T_2)$ represents the underlying probability density of the T_2 times and is the object of interest. In the case of MWF mapping, for the remainder of this chapter, we model $p(T_2)$ as a sum of point masses $p(T_2) = c_1\delta_{T_{2,1}} + (1-c_1)\delta_{T_{2,2}}$, representing the rapid transverse relaxation of the less-mobile water entrapped within myelin lamallae represented by $1/T_{2,1}$ as compared to the slower rate of transverse relaxation of more-mobile water represented by $1/T_{2,2}$. This leads to the following biexponential model for $s(t)$ given by:

$$s(t; T_{2,1}, T_{2,2}, c_1, c_2) = c_1 e^{\frac{-t}{T_{2,1}}} + (1-c_1)e^{\frac{-t}{T_{2,2}}}, \tag{4.2}$$

In practice the signal is observed at finitely many echo times under Rician noise (described further in Section 4.2). In effect, the problem is that of separating the two exponential decay signals that comprise the net acquired noise corrupted measurement $\mathbf{s}$. Depending primarily upon the difference in the decay constants of the two underlying signals, this can be an extremely challenging parameter estimation problem [LDr84, SB20]. Consequently, it has attracted several estimation techniques ranging from classical estimation theory to more modern deep learning approaches which attempt to resolve the underlying ill-posedness of separating the two decay constants.

In this chapter, we propose a novel deep learning method for biexponential signal estimation by combining deep learning with classical estimation, by improving a previously proposed deep learning technique termed Input Layer Regularization (ILR) [RPB+22]. The idea behind ILR relies on the somewhat simple but straightforward strategy of *regularized curve fitting* where given a noisy signal, the parameter $\mathbf{p}$ may be estimated by fitting a biexponential curve to the data with the complexity of the parameters being penalized by a regularization term. When this regularization term is the squared norm of the parameters, this is the well-known Tikhonov regularization, and the quality of the parameters so

obtained depends on the weighting between the data fidelity and regularization term, controlled by the all-important hyperparameter $\lambda \geq 0$. The curve fit parameters $\mathbf{p}_\lambda^*$ so obtained will implicitly contain information regarding $\mathbf{p}$; as a consequence, a biexponential signal generated from these curve-fitted parameters can be concatenated with the original noisy measurement to serve as an additional source of information for a neural network. It was seen empirically that this process improves the MSE for a neural network trained to estimate the time decays T_{21}, T_{22} given a constant MWF value.

In the original use of ILR, however, a single λ was chosen for all signals, irrespective of the underlying true parameter combination. For the three parameter biexponential recovery problem, a universal choice of λ tends to hinder rather than help the neural network, as different signals may require different levels of Tikhonov regularization. Our key insight is that if this λ is chosen optimally based on the signal (with optimality being described in Section 4.2) then the parameters $\mathbf{p}_\lambda^*$ obtained from the curve fit will closely approximate the underlying parameters $\mathbf{p}$. The biexponential signal derived from these optimal parameters can then be considered a "denoised" or regularized version of the original signal–we concatenate this denoised or regularized version with the original noisy data and feed the concatenated pair to a neural network which outputs a parameter estimate $\mathbf{p}_{est}$. This procedure can be viewed as augmenting the input layer of a conventional neural network with a regularized signal that contains high-quality information about the underlying true parameters. We further expand on our method's connection to the empirical Bayes method in Section 4.2.6.1.

4.1.1 Contributions

Having briefly described our method, our detailed contributions are as follows:

1. We propose a novel hybrid classical and deep learning method by improving our previously pub-

lished work on ILR [RPB+22] by introducing signal-dependent selection of the regularization hyperparameter λ.

2. We explore two choices for selecting this λ optimally from the signal: (1) following [ACC21] by using a neural network termed λ_{NN} and (2) using generalized cross validation (GCV) [GHW79], leading to the estimator λ_{GCV}. We empirically demonstrate that mean absolute error (MAE) or the L^1 loss is an effective loss function to train the network λ_{NN} which takes a noisy signal to its optimal regularization hyperparameter termed λ_{oracle}. We show that λ_{NN} outperforms λ_{GCV} across all noise levels in terms of matching the underlying distribution of λ_{oracle} (Table 4.3). We remark that both approaches are instances of a bilevel optimization-based framework for studying the optimality of regularization parameters, and to the best our knowledge, the present chapter is the first instance to study the optimal λ-selection problem from this bilevel perspective for any multiexponential parameter estimation problem.

3. Next we study the central inverse problem of estimating the MWF c_1 from the signal $\mathbf{s}$. We show that on synthetically generated data our proposed method of ILR with signal-dependent λ-selection outperforms both classical curve fitting and purely deep learning based approaches (Tables 4.4, 4.5, 4.6).

4. We apply our techniques to in-vivo brain data for two patients and 3 slices per patients, and find consistent improvement via ILR in each slice. A significant aspect of our analysis of brain data is an initial classification of noise level at every voxel followed by voxel filtering by the Akaike Information Criterion (AIC) to restrict the application of our methods to voxels whose signals are better fit with a biexponential model vs a monoexponential model. Furthermore, we find that the ILR strategy with GCV-based selection of λ outperforms ILR with NN-based selection. This

represents an application in which classical methods, including generalized cross validation, are combined with deep learning for performing challenging estimation tasks in brain data.

4.1.2 Related work

Estimation of MWF from the transverse decay signal is known to be an ill-posed problem and to require imposition of a prior in order to obtain meaningful results. One approach is to model the decay as the sum of two discrete decaying exponentials. Extraction of the two decay time constants can be far from trivial, but ultimately yields point estimates for the more rapidly decaying MWF and the more slowly decaying non-myelin associated water. This is in keeping with other techniques yielding single, discrete values for these two water components, and in particular with the widely-used mcDESPOT analysis. That is, in common with biexponential analysis as performed in this and other papers, mcDESPOT [DRA+08, BS16b, GBK+23] yields a single $T2$ value for myelin-associated water and less-restricted water, respectively, as well as single values representing corresponding component fractions.

Alternative modeling choices have also been made, in which the underlying $T2$ distribution has been taken as Gaussian, Laplacian, or Multi-Gaussian, with one study showing that these model choices contribute negligibly to the estimation error in the MWF (see Figure 1 [RPS+14]). A non-parametric approach is based on the regularized non-negative least (NNLS) method, in which there is no assumption regarding the number of exponentially decaying components [MWA+94]. Instead, the location and amplitude of the modes of the T2 distribution, corresponding to distinct water components, are inferred from the recovered distribution. This is most commonly performed using a hard cutoff value, essentially performing a classification task in which a binary assignment to myelin-

associated water or non-myelin associated water is made for each value of T2. The cutoff serves as a hyperparameter, and, depending on the morphology of the recovered distribution, can be of greater or lesser physical plausibility. As with any classification task, and perhaps especially one with a non-validated single value cutoff, there are associated Type I and Type II assignment errors which are typically not modeled. We have found (data not shown) that modeling the transverse decay as two discrete exponentials versus modeling that decay with the non-parameteric approach yields results within a few percent of each other). In any event, the goal of this chapter is not to perform a comparison of NLLS (biexponential, mcDESPOT) with NNLS approaches to MWF estimation but rather to describe a novel NN architecture for the biexponential model, and readily generalizable to other parameter estimation problems

There has also been steady progress in deep learning methods for denoising Rician-noised MR images [LZLL20, GRADM24, TB20, JDV+18, SCC+22, Emi23]. A typical sticking point in these works is the lack of large volumes of real-life MR images with ground truths for model training. Therefore, there also exist methods that focus on alleviating this data paucity problem through synthetic generation of MR images [VWKA24]. Finally, several deep learning methods also perform the denoising and parameter estimation steps in tandem, directly mapping a noisy MR image to the underlying parameters [EHMCM20, CMCM19, ZWL+22, BOBS22, DBJ21, YCRP+21]. This is the approach we will use, and our approach will be benchmarked against another state of the art deep learning approach, Model Informed Machine Learning (MI-ML) [YCRP+21], in Section 4.4.

Thus, point-estimator approaches to MWF quantification, as in this work, are viable and in use. In any event, our goal here is not to compare point estimator methods (mcDESPOT, biexponential) with approaches based on recovery of $T2$ distributions [DLV+21], but rather to further develop ILR for biexponential analysis.

4.1.3 Chapter organization

The outline of the chapter is as follows; this also serves to collect several symbols and definitions. In Section 4.2, describing general features of the problem set up, we first define the biexponential model with Rician noise, and outline the NLLS approach to parameter estimation in this model as an inverse problem. We note the requirement for regularization and outline methods for regularization parameter selection, including a λ selection NN, denoted λ_{NN}. Finally, we describe input layer regularization (ILR).

In Section 4.3, we describe two methods of ILR implementation, based respectively on selection of λ via generalized cross validation (GCV) or via a separate λ selection NN, denoted respectively by λ_{GCV} and λ_{NN}. In both cases, the selection of λ is followed by the implementation of the parameter-estimation (PE) NN using ILR. This involves concatenating the noisy decay (ND) vector with a version of itself regularized with either λ_{GCV} or λ_{NN} to form the input signal to the PE NN, with the concatenated input signals denoted (ND, REG)$_{\text{GCV}}$ or (ND, REG)$_{\text{NN}}$. Since this involves doubling the length of the input ND signal, we compare performance of the two ILR methods with a PE NN for which the input vector is the noisy signal concatenated with itself, deemed (ND, ND).

Although our primary interest is in parameter estimation, the performance of the λ_{NN} is of independent interest, given ongoing research in λ selection for Tikhonov regularization. Therefore, in Section 4.4, we use an oracle λ_{oracle} as a comparison standard for evaluating the performance of GCV and λ_{NN} for regularizer selection on simulated data. We then compare five methods of parameter estimation, focusing on MWF; these are three NN's, with subscripts indicating the method of λ selection: i) (ND, REG)$_{\text{NN}}$, ii) (ND, REG)$_{\text{GCV}}$, and iii) (ND, ND) as described above; iv) Tikhonov-regularized Levenberg-Marquart NLLS (TR-NLLS), using λ_{GCV}; and v) conventional NLLS. These are all com-

pared to TR-NLLS using λ_{oracle}. We then present results for in vivo human brain. In this case, there is no true oracle standard of comparison, so we establish a high-quality MWF map as a reference standard. Of note is that we apply ILR only to pixels for which the decay signal is of biexponential character. We compare the performance of the two ILR PE NN's, using respectively λ_{NN} and λ_{GCV}, finding that the latter exhibits somewhat superior performance, and then compare this with the conventional (ND, ND) PE NN, not using ILR. We acknowledge that our analysis involves several interacting components addressed with abbreviations. Therefore for the convenience of the reader we include a glossary of abbreviations in Table (4.1) below.

In Section 4.5, we highlight additional aspects of the underlying problem and certain limitations of our approach.

4.2 Mathematical formalism

4.2.1 Biexponential signal analysis

We assume that $s(t)$ is measured at a finite number of echo times $\{t_i\}_{i=1}^n$; Eq. (4.2) becomes:

$$\mathbf{G(p)} \equiv (c_1 e^{\frac{-t_i}{T_{2,1}}} + (1-c_1)e^{\frac{-t_i}{T_{2,2}}})_{i=1}^n, \tag{4.3}$$

where $\mathbf{p}$ is the set of parameters $(c_1, T_{2,1}, T_{2,2})$. We will further assume $t_1 = 0$ as is usual for biexponential analysis, although for the conventional MRI experiment data acquisition is initiated at a minimum echo time $t_1 > 0$ which depends primarily on gradient and pulse duration. Additionally, $\mathbf{G(p)}$ is invariably corrupted by noise. In this chapter we will address the conventional case of MR

Table 4.1: Glossary of abbreviations used in the chapter

.

Term	**Abbreviation**
Input Layer Regularization	ILR
Noisy Decay	ND
Regularized Signal	Reg
Parameter Estimation	PE
Neural Network	NN
Non-Linear Least Squares	NLLS
Tikhonov Regularized Non Linear Least Squares	TR-NLLS
Generalized Cross Validation	GCV
Myelin Water Fraction	MWF
Minimum Mean Squared Estimator	MMSE

magnitude imaging, in which the noise model is Rician so that

$$\mathbf{s} = \sqrt{(\mathbf{G(p)} + \eta_2)^2 + \eta_1^2} \tag{4.4}$$

Here η_1 and η_2 are independent Gaussian random variables of zero mean and equal variance. When the signal **s** is normalized to unity, the corresponding signal-to-noise ratio (SNR) will be expressed as the standard deviation σ of these random variables. In other applications, zero-mean additive Gaussian random noise will be the most common noise model. We denote the true underlying parameter set

producing the signal $\mathbf{G}$ as $\mathbf{p}_{true}$. The goal of the analysis is to recover the parameter vector $\mathbf{p}_{true}$ from the noisy measurement vector $\mathbf{s}$.

4.2.2 Parameter Estimation

A straightforward way to recover $\mathbf{p}_{true}$ from $\mathbf{s}$ is *nonlinear least squares* (NLLS) analysis. We note that for Rician data, this deviates from statistically validated maximum likelihood estimator analysis [SDDSVD98, JY03], but is nevertheless conventional and accurate for reasonably high SNR. The NLLS approach involves computing an estimate $\mathbf{p}^*$ that minimizes the mean-squared difference between the model $\mathbf{G(p)}$ and the signal $\mathbf{s}$ for $\mathbf{p}$ in a feasible set of parameters. We can tailor this feasible set based on our problem, in this case constraining $T_{2,1}, T_{2,2} \geq 0$ and $0 \leq c_1 \leq 1$. Therefore the NLLS approach to parameter estimation is mathematically formulated as follows:

$$\mathbf{p}^* \equiv \underset{\mathbf{p}\in F}{\text{argmin}} \ \|\mathbf{G(p)} - \mathbf{s}\|_2^2 \tag{4.5}$$

$$= \underset{T_{2,1},T_{2,2}\geq 0, 0\leq c_1\leq 1}{\text{argmin}} \left[\sum_{i=1}^{m} (c_1 e^{\frac{-t_i}{T_{2,1}}} + (1-c_1)e^{\frac{-t_i}{T_{2,2}}} - s_n)^2\right]. \tag{4.6}$$

However, as stated, the inverse problem (4.5) is highly ill-posed in the sense of Hadamard [ABT18], meaning that distinct parameter choices $\mathbf{p_1}$ and $\mathbf{p_2}$ can map to nearly identical biexponential decay curves [Lan88]. This ill-posedness results in a high sensitivity to noise of derived parameter estimates. Stabilization of the parameter estimation problem is therefore a major topic of investigation in inverse problems.

4.2.3 Regularization

A well-known approach to address the ill-posedness of the problem Eq. (4.5) is *regularization*, in which (4.5) is modified through the introduction of an additional term to the objective function that penalizes the complexity of feasible solutions $\mathbf{p}$ [CS18]. Among the many ways of quantifying complexity, an effective approach is to supplement (4.5) with *Tikhonov* regularization (TR) [GHO99], where the complexity is described through the L^2 norm of the parameters weighted against the fidelity term through the *hyperparameter* λ:

$$\mathbf{p}^*_{\lambda(\mathbf{s})} := \underset{\mathbf{p}\in[0,1]\times[0,+\infty)^2}{\mathsf{argmin}} \|\mathbf{G(p)} - \mathbf{s}\|_2^2 + \lambda\|B\mathbf{p}\|^2. \tag{4.7}$$

Here B is a 3x3 diagonal matrix:

$$B = \mathsf{diag}(100, 1, 1), \tag{4.8}$$

taken to equalize the scales between different parameters, since the $T_{2,1}, T_{2,2}$ values are two orders of magnitude larger than c_1. Along with other variational regularization techniques such as Lasso (L^1 regularization) or elastic net (a linear combination of both L^2 and L^1 regularizers), TR effectively serves to stabilize ill-posed inverse problems. In particular, introduction of the parameter $\lambda \geq 0$ improves the condition number of the parameter estimation problem[TB22, BFBS22] that is, the map of the noisy signal $\mathbf{s}$ to the parameter set $\mathbf{p}^*_\lambda(\mathbf{s})$, as explicitly outlined for this type of system in [BFBS22]. We will henceforth term this the *Tikhonov Regularized NLLS (TR-NLLS)* estimate. In a Bayesian framework, it provides Gaussian priors on the parameters where the TR-NLLS estimate is the maximum a posteriori

(MAP) estimator. In the present case, there is no straightforward statistical or physical interpretation of the TR-NLLS estimator $\mathbf{p}^*_\lambda(\mathbf{s})$ and the application of TR for parameter estimation in Rician-distributed signals is highly non-standard. However, even lacking a true physical basis, regularization applied in (4.7) TR-NLLS would be expected to improve stability.

4.2.4 Regularization Parameter Selection

The appropriate selection of λ, which controls the bias-variance tradeoff in estimating $\mathbf{p}_{true}$ through TR, is a central challenge in inverse problems. Large values lead to simple, low complexity solutions (low variance) but with low fidelity to the underlying signal $\mathbf{s}$ (high bias) and highly biased parameter estimates. Conversely, small values ensure the closest fidelity to the signal (low bias) but high variability to small perturbations in the signal (high variance), such as due to noise. In general, the problem of choosing the optimal λ refers to selecting the λ for which the associated solution to (4.7) $\mathbf{p}^*_\lambda(s)$ is the closest to the true underlying parameter $\mathbf{p}_{true}$. This can be mathematically formulated through the following *bilevel* optimization problem:

$$\lambda_{\text{oracle}}(\mathbf{s}) = \underset{\lambda}{\text{argmin}} \|\mathbf{p}_{true} - \mathbf{p}^*_\lambda(\mathbf{s})\|, \quad \text{w.r.t} \tag{4.9}$$

$$\mathbf{p}^*_\lambda(\mathbf{s}) = \underset{\mathbf{p}\in[0,1]\times[0,+\infty)^2}{\text{argmin}} \|\mathbf{G}(\mathbf{p}) - \mathbf{s}\|_2^2 + \lambda\|\mathbf{p}\|_2^2. \tag{4.10}$$

Here (4.9) is referred to as the *upper level* problem and (4.10) is referred to as the *lower level* problem. Note that the solution to the upper level problem depends upon the noisy signal $\mathbf{s}$ which in turn depends on the map $\mathbf{G}$ and the SNR. Thus the bilevel formulation identifies two sources which affect the optimal λ. We term the optimal $\lambda(\mathbf{s})$ from (4.9) as λ_{oracle} and refer to the following problem as the $\lambda-$*selection*

problem:

$$\text{Recover } \lambda_{\text{oracle}} \text{ from } \mathbf{s} \tag{4.11}$$

Typically it is infeasible to directly solve the bilevel problem (4.9)-(4.10) because $\mathbf{p}_{true}$ is not available in applications. Moreover, establishing theoretical guarantees for the existence and positivity of solutions to the bilevel problem (4.9)-(4.10) is an active area of research [HKB18]. Consequently, a great deal of theory is dedicated to approximating λ_{oracle} using alternative formulations and algorithmic techniques [ADVL+21, The15]. In practice, numerical methods for choosing λ are often implemented, including the L-curve (LC) [EG94], discrepancy principle (MDP) [NTS03], or generalized cross validation (GCV) [GHW79]. Among these, GCV stands out as a theoretically and practically robust method for approximating λ_{oracle} [Luk93, GVM97, HO00].

4.2.5 Neural networks for parameter estimation

Neural Networks (NNs), introdcued in Chapter 1, are well-established in the setting of inverse problems [CHBJ+18] in the biomedical sciences and offer a model-free approach to mapping an unseen signal to parameter estimates. Unlike classical mapping methods (such as TR-NLLS), NNs require training with synthetically or experimentally generated data. Such approaches involving NNs routinely outperform classical methods as long as the underlying NN is well-trained [SSAN16, WSJK18, AV03]. In fact, NNs and classical methods like TR-NLLS may be combined. For example in [ACC21] a supervised learning approach to estimating λ_{oracle} was presented, wherein signals $\mathbf{s}$ are generated from a known underlying $\mathbf{p}_{true}$, followed by direct calculation of the corresponding optimal λ, λ_{oracle}, leading to training pairs of the form $\{\mathbf{s}, \lambda_{\text{oracle}}(\mathbf{s})\}$. A neural network can then be trained on

this data and serve as a surrogate mapping for λ_{oracle}. We will elaborate on this process in Section 4.3. These considerations allow us to define two distinct NNs for the present problem:

1. NNs for λ-selection: Following [ACC21], we develop a NN, termed λ_{NN}, for λ selection. We compare this with λ_{GCV} as defined in 4.2.4.

2. NNs for parameter estimation: We develop a separate NN for estimations of the parameter $\mathbf{p}_{true}$ from the noisy signal $\mathbf{s}_i$.

4.2.6 Input Layer Regularization

In this chapter we combine λ-selection with a NN for parameter estimation into a pipeline for estimating $\mathbf{p}_{true}$. The resulting architecture, which we call *input layer regularization (ILR)* [RPB+22], is defined by concatenation of the noisy signal data (ND) $\mathbf{s}$ with, or augmented by, a regularized (Reg) version given by $G\left(\mathbf{p}^*_{\lambda(\mathbf{s})}\right)$ where $\mathbf{p}^*_{\lambda(\mathbf{s})}$ is a TR-NLLS parameter estimate from (4.7) with hyperparameter $\lambda(\mathbf{s})$ chosen through a user-defined λ-selection method. The augmented input vector is thus given by

$$\mathbf{x} := \left[\mathbf{s}, \mathbf{G}(\mathbf{p}^*_{\lambda(\mathbf{s})}(\mathbf{s}))\right]. \tag{4.12}$$

A NN (termed (ND, REG)) is trained to estimate $\mathbf{p}_{true}$ from $\mathbf{x}$ (see Figure 4.1).

4.2.6.1 Theoretical justifications for ILR

ILR may be conceived as a deep learning-based instantiation of the empirical Bayes estimator of $\mathbf{p}$ [Cas85]. To make this connection precise, first consider the minimum mean squared estimator

(MMSE) of $\mathbf{p}$ from $\mathbf{s}$, given by $\mathbb{E}[\mathbf{p} \mid \mathbf{s}]$, the conditional expectation of $\mathbf{p}$ given $\mathbf{s}$. Moreover, suppose we had a *prior* for $\mathbf{p}$, depending on hyperparameters λ with the MMSE $\mathbb{E}[\mathbf{p} \mid \mathbf{s}, \lambda]$. Note that mean squared errors for both MMSE's (with prior and without prior) can be given in terms of the variances of the conditional distributions of $\mathbf{p}$:

$$\text{MSE}(\mathbb{E}[\mathbf{p} \mid \mathbf{s}]) := \mathbb{E}_{\mathbf{p},\mathbf{s}}[\|\mathbf{p} - \mathbb{E}[\mathbf{p} \mid \mathbf{s}]\|_2^2] = \mathbb{E}_{\mathbf{s}}[\text{Var}(\mathbf{p} \mid \mathbf{s})], \tag{4.13}$$

$$\text{MSE}(\mathbb{E}[\mathbf{p} \mid \mathbf{s}, \lambda]) := \mathbb{E}_{\mathbf{p},\mathbf{s},\lambda}[\|\mathbf{p} - \mathbb{E}[\mathbf{p} \mid \mathbf{s}, \lambda]\|_2^2] = \mathbb{E}_{\mathbf{s},\lambda}[\text{Var}(\mathbf{p} \mid \mathbf{s}, \lambda)]. \tag{4.14}$$

Moreover, note that by the law of total variance, conditioning on λ, we have:

$$\text{Var}(\mathbf{p} \mid \mathbf{s}) = \mathbb{E}_{\lambda}[\text{Var}(\mathbf{p} \mid \mathbf{s}, \lambda)] + \text{Var}_{\lambda}(\mathbb{E}[\mathbf{p} \mid \mathbf{s}, \lambda] \mid \mathbf{s}). \tag{4.15}$$

Taking expectations over the signal $\mathbf{s}$ we get:

$$\text{MSE}(\mathbb{E}[\mathbf{p} \mid \mathbf{s}]) = \text{MSE}(\mathbb{E}[\mathbf{p} \mid \mathbf{s}, \lambda]) + \mathbb{E}_{\mathbf{s}}[\text{Var}(\mathbb{E}[\mathbf{p} \mid \mathbf{s}, \lambda] \mid \mathbf{s})]. \tag{4.16}$$

This proves that under *full knowledge* of the hyperparameters λ, the MMSE that uses prior information does better than the MMSE that does not. In practice, however, we do not have access to the prior. In this case, under the empirical Bayes scheme, we would compute a point estimator $\lambda(\mathbf{s})$ using the data $\mathbf{s}$, and then obtain an estimator for $\mathbf{p}$ given by $\varphi(\mathbf{s}, \lambda(\mathbf{s}))$. Using a decomposition $\mathbf{p} - \varphi = (\mathbf{p} - \mathbb{E}[\mathbf{p} \mid$

$\mathbf{s}, \lambda]) + (\mathbb{E}[\mathbf{p} \mid \mathbf{s}, \lambda] - \varphi)$ we have

$$\text{MSE}(\varphi(\mathbf{s}, \lambda(\mathbf{s})) := \mathbb{E}[\|p - \varphi(\mathbf{s}, \lambda(\mathbf{s}))\|_2^2] \tag{4.17}$$

$$= \text{MSE}(\mathbb{E}(\mathbf{p} \mid \mathbf{s}, \lambda)) + \mathbb{E}[\|\varphi(s, \lambda) - \mathbb{E}[\mathbf{p} \mid \mathbf{s}, \lambda]\|_2^2]. \tag{4.18}$$

Combining this decomposition with (4.16) we have:

$$\text{MSE}(\varphi) = \underbrace{\text{MSE}(\mathbb{E}[\mathbf{p} \mid \mathbf{s}]) - \mathbb{E}_s[\text{Var}(\mathbb{E}[\mathbf{p} \mid \mathbf{s}, \lambda] \mid s)]} \tag{4.19}$$

$$+ \mathbb{E}[\|\varphi(\mathbf{s}, \lambda) - \mathbb{E}[\mathbf{p} \mid \mathbf{s}, \lambda]\|_2^2]. \tag{4.20}$$

The first term grouped under the bracket is *fundamental* and it is strictly smaller than the MMSE of $\mathbf{p}$ without a prior. The term that contributes to the error is the proximity of the estimator φ to the MMSE with prior. Therefore it is beneficial to design an estimator $\lambda(\mathbf{s})$ and an estimator φ depending on both the signal $\mathbf{s}$ and $\lambda(\mathbf{s})$ that is as accurate as possible. This is what we attempt to do in ILR with λ-selection. In particular, we make the following modeling assumptions:

$$\mathbf{s} \sim \mathbb{P}(\mathbf{s} \mid \mathbf{p}), \mathbf{p} \sim \mathbb{P}(\mathbf{p} \mid \lambda) \tag{4.21}$$

Under this setup, a neural network trained to carry $\mathbf{s}$ to $\mathbf{p}$ would be surrogate for $\mathbb{E}[\mathbf{p} \mid \mathbf{s}]$ and the proposed estimator $\varphi(\mathbf{s}, \lambda(\mathbf{s}))$ is our ILR network trained to carry a point estimator of λ and the signal $\mathbf{s}$ to $\mathbf{p}$ via concatenation. Here the point estimator is selected via the bilevel optimization problem (4.9)-(4.10) and is pushed into an estimator, i.e the (ND, REG) network such that the MMSE-empirical bayes error in (4.20) is minimized. However, our framework deviates from the empirical Bayes in two

crucial ways: (1) λ_{oracle} is not necessarily optimal in a statistical framework (like a MAP estimator, for instance), and (2) given Rician noise in $\mathbf{s}$ the lower level program (4.10) is not a maximuim likelihood estimator of $\mathbf{p}$. Despite these estimators not having a direct statistical interpretation, they will nevertheless be maximally informative of $\mathbf{p}$.

Another significant deviation from empirical Bayes arises from the fact that λ_{oracle} *uses* $\mathbf{p}_{true}$; as such this scheme would be could not be implemented in practice at the inference step of the analysis when one does not have access to the underlying true parameter. However, by realizing that we can generate instances of $\mathbf{p}_{true}$ and λ_{oracle}, a neural network can be trained to approximate a point estimator of λ. Therefore at the inference step, we may simply use the observed signal $\mathbf{s}$ to obtain $\lambda(\mathbf{s})$.

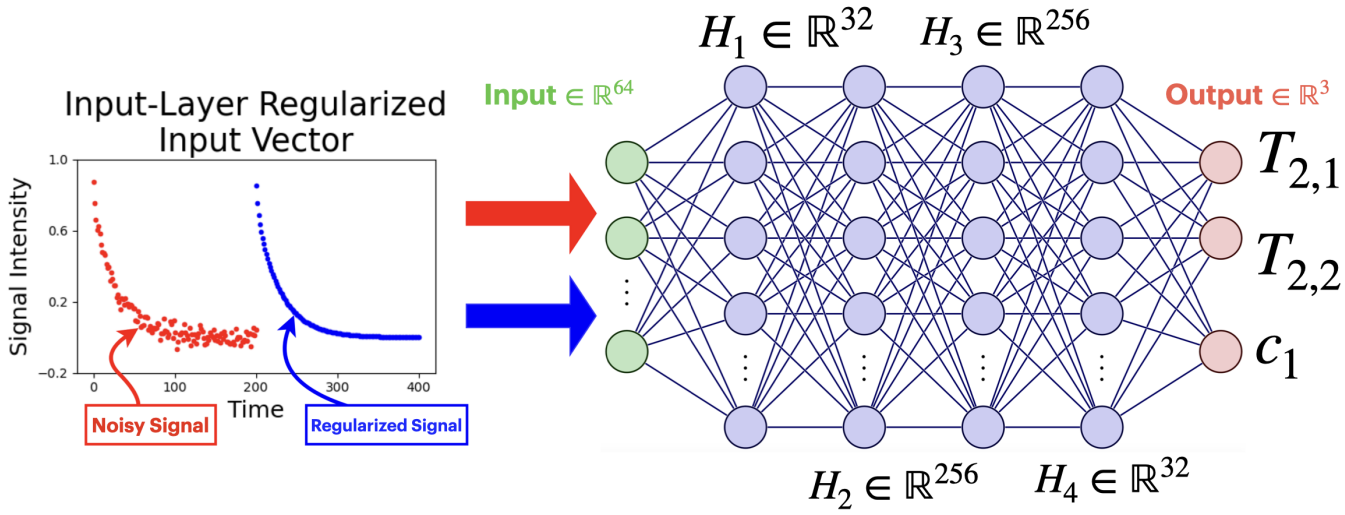


Figure 4.1: The (ND, REG) parameter estimation network is an NN where the inputs are the concatenated vectors of noisy signal s and the signal obtained from parameters estimated by Tikhonov regularization, with regularization parameter $\lambda(\mathbf{s})$ given by either a neural network or GCV. The parameters estimated are $c_1, T_{2,1}$ and $T_{2,2}$.

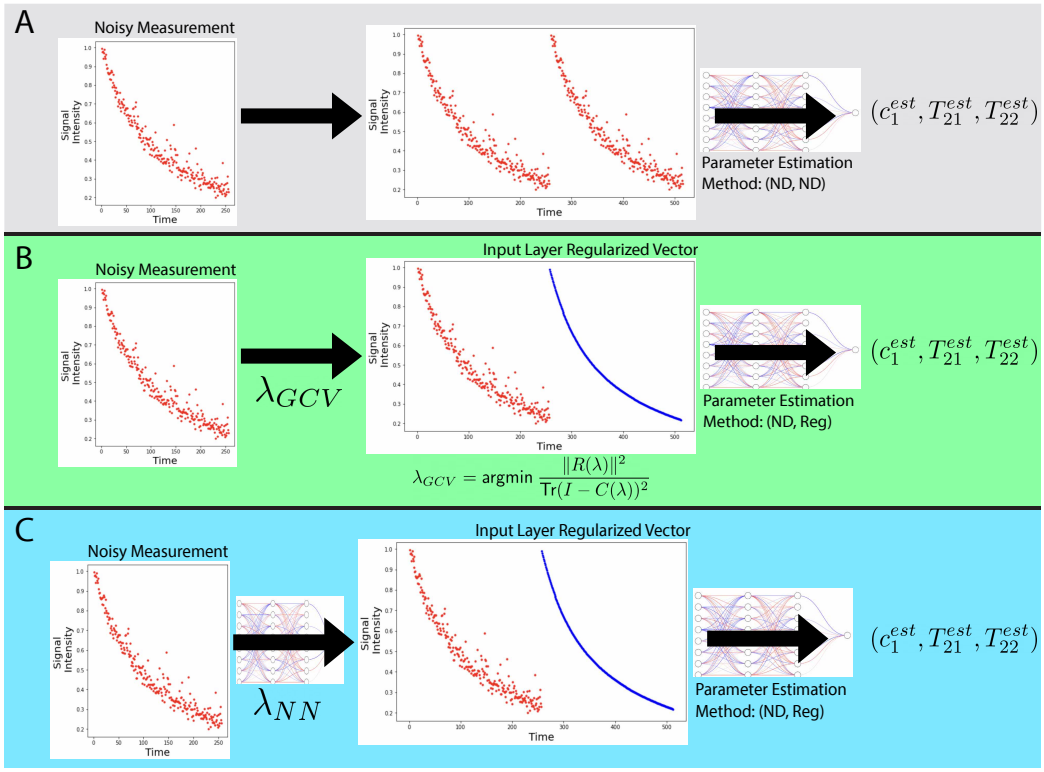


Figure 4.2: We compare the three approaches of parameter estimation. (A): The conventional method of estimating parameters from the noisy decay (ND) itself. (B and C): A novel approach for parameter estimation, where the first step involves selecting either GCV or NN for estimating λ, labeled λ_{GCV} and λ_{NN}, respectively. Training data for the ILR networks (ND, REG) are first passed through a TR-NLLS solver with the signal-dependent $\lambda(\mathbf{s})$ to construct the concatenated vector $\mathbf{x}$ in Eq. (4.12). $C(\lambda)$ is defined according to Eq. (4.26)

4.3 Methods

As above, we propose a methodology combining ILR with signal-based selection of λ for solving the parameter estimation problem. We compare two choices for generating λ: (1) a neural network, following [ACC21] and (2) GCV (outlined in Section 4.3.2). See Figure 4.2. In Section 4.2.2, we discuss the choice between use of λ_{NN} and λ_{GCV}.

We now outline our algorithm for each choice of estimating λ.

4.3.1 Selecting the hyperparameter with a neural network followed by parameter estimation

To estimate λ by a NN, $\lambda(\mathbf{s}) := \lambda_{\text{NN}}(\mathbf{s})$, we train λ_{NN} using synthetically generated training data $\{\mathbf{s}_i, \lambda_{\text{oracle}}(\mathbf{s}_i)\}$. Given $\mathbf{s}$, λ_{oracle} is found using the Brent method which is a combination of golden-section search and parabolic interpolation, across the search space $[10^{-7}, 10^{3}]$, optimizing (4.9). We summarize the data generation process in Appendix 4.3.3. Here λ_{NN} is a convolutional neural network (CNN) with the architecture shown in Figure (4.3). We make use of L^1 loss to train the NN to recognize λ_{oracle} in order not to heavily penalize outliers since the distribution of λ_{oracle} spans several orders of magnitude (Figure 4.5). Thus, the loss function used for training λ_{NN} is given by:

$$L_\lambda := \frac{1}{n}\sum_{i=1}^{n} |\lambda_{\text{oracle}}(\mathbf{s}_i) - \lambda_{\text{NN}}(\mathbf{s}_i)| \tag{4.22}$$

We provide model architecture and training details in 4.3.3.2.

To recover parameter estimates, we generate the concatenated input $\mathbf{x}_i$ according to (4.12) with $\lambda_{\text{NN}}(\mathbf{s})$ in place of $\lambda(\mathbf{s}.)$. Letting $\mathbf{p}_i$ be the parameter vector used for generating the noisy signal $\mathbf{s}_i$, we train a neural network $(\text{ND}, \text{REG})_{\text{NN}}$ for recovering $\mathbf{p}_i$ from $\mathbf{x}_i$ by minimizing mean-squared error. The loss function for training $(\text{ND}, \text{REG})_{\text{NN}}$ is given by:

$$L_{\text{ILR}} := \frac{1}{n}\sum_{i=1}^{n} \|\mathbf{p}_i - (\text{ND}, \text{REG})_{\text{NN}}(\mathbf{x}_i)\|_B^2 \tag{4.23}$$

Here $\|\cdot\|_B^2$ is the *weighted* mean squared error norm given by $\|x\|_B^2 = x^\top B x$, where B is the diagonal matrix (4.8).

4.3.2 Selecting the hyperparameter with GCV followed by parameter estimation

Here we describe the algorithm used for producing $\lambda_{\text{GCV}}(\mathbf{s})$. Recall that $\mathbf{G}$ in (4.3) maps a parameter combination $\mathbf{p} \in \mathbb{R}^3$ to a noiseless signal vector $\mathbf{G}(\mathbf{p}) \in \mathbb{R}^m$. $\mathbf{G}$ is a differentiable map from $\mathbb{R}^3$ to $\mathbb{R}^m$ and admits a Jacobian $D\mathbf{G} : \mathbb{R}^3 \to \mathbb{R}^n$. Next, recall that for a fixed λ, $\mathbf{p}^*_\lambda(\mathbf{s})$ is the TR-NLLS parameter estimate (4.7). This allows us to define a Jacobian $D\mathbf{G}$, a matrix-valued function that depends *only* on λ:

$$J(\lambda) := D\mathbf{G}(\mathbf{p}^*_\lambda(\mathbf{s})) \tag{4.24}$$

Then λ_{GCV} for nonlinear inverse problems is defined as the minimizer of a function involving $J(\lambda)$ [HO00]:

$$\lambda_{\text{GCV}}(\mathbf{s}) = \underset{\lambda \geq 0}{\mathsf{argmin}}\, \mathsf{GCV}(\lambda) \tag{4.25}$$

$$:= \underset{\lambda \geq 0}{\mathsf{argmin}} \frac{\|G\left(\mathbf{p}^*_\lambda\left(\mathbf{s}\right)\right) - \mathbf{s}_i\|_2^2}{\mathsf{Tr}\left[\left(I - J(\lambda)(J(\lambda)^\top J(\lambda) + \lambda I)^{-1} J(\lambda)^\top\right)^2\right]} \tag{4.26}$$

The concatenated ILR input vectors are produced according to (4.12) with $\lambda_{\text{GCV}}(\mathbf{s})$ used place of $\lambda(\mathbf{s})$. The resulting NN trained to minimize the loss (4.23) is termed $(\text{ND}, \text{REG})_{\text{GCV}}$ (Figure 4.2). Note that constructing training data for either the NN or GCV-based approach requires grid-based optimization for obtaining $\lambda_{\text{ORACLE}}(\mathbf{s}_i)$. However, once training is completed, λ_{NN} requires no further grid-based searches for operation, while obtaining $\lambda_{\text{GCV}}(\mathbf{s}_i)$ still requires this at the testing, or implementation, step, in order to solve (4.26).

4.3.3 Computational details

4.3.3.1 Synthetic data generation

We generate the training data used for λ_{NN}, (ND, REG)$_{\text{NN}}$, and (ND, REG)$_{\text{GCV}}$ as follows. For each parameter vector $\mathbf{p}$ in a uniformly spaced grid of possible parameter values specified in Table 4.2 we generate a noiseless signal $\mathbf{G}(\mathbf{p})$ using (4.3). Every noiseless signal is then corrupted with 1000 independent and identically distributed (iid) copies of Rician noise to generate noisy signals resembling experimental MRI data. The noisy signals are normalized to have the initial signal amplitude be 1.0 at the minimum echo time $TE_0 = 8.0$. Finally, we assemble all the noisy signals over all the parameter vectors into the set of training inputs denoted by $\{\mathbf{s}_i\}$. Then the parameters used for generating $\mathbf{s}_i$ will be indexed accordingly and assembled into the set of training outputs given by $\{\mathbf{p}_i\}$. To obtain $\lambda_{oracle}(\mathbf{s}_i)$, we perform a grid search over $\lambda \in [10^{-7}, 10^3]$ for finding minimizers to (4.9). For obtaining λ_{GCV}, we employ another grid search in the range $\lambda \in [10^{-7}, 10^3]$ to find the minimizers in (4.26).

4.3.3.2 NN training

In this paper we trained four distinct NNs:

1. λ_{NN} for estimating λ_{oracle} from the noisy signal $\mathbf{s}$,
2. (ND, REG)$_{\text{NN}}$ for estimating $\mathbf{p}_{true}$ from $(\mathbf{s}, G(\mathbf{p}_{\lambda_{NN}(\mathbf{s})}(\mathbf{s}))$,
3. (ND, REG)$_{\text{GCV}}$ for estimating $\mathbf{p}_{true}$ from $(\mathbf{s}, G(\mathbf{p}_{\lambda_{GCV}(\mathbf{s})}(]\mathbf{s}))$, and
4. (ND, ND) for estimating $\mathbf{p}_{true}$ from $(\mathbf{s}, \mathbf{s})$.

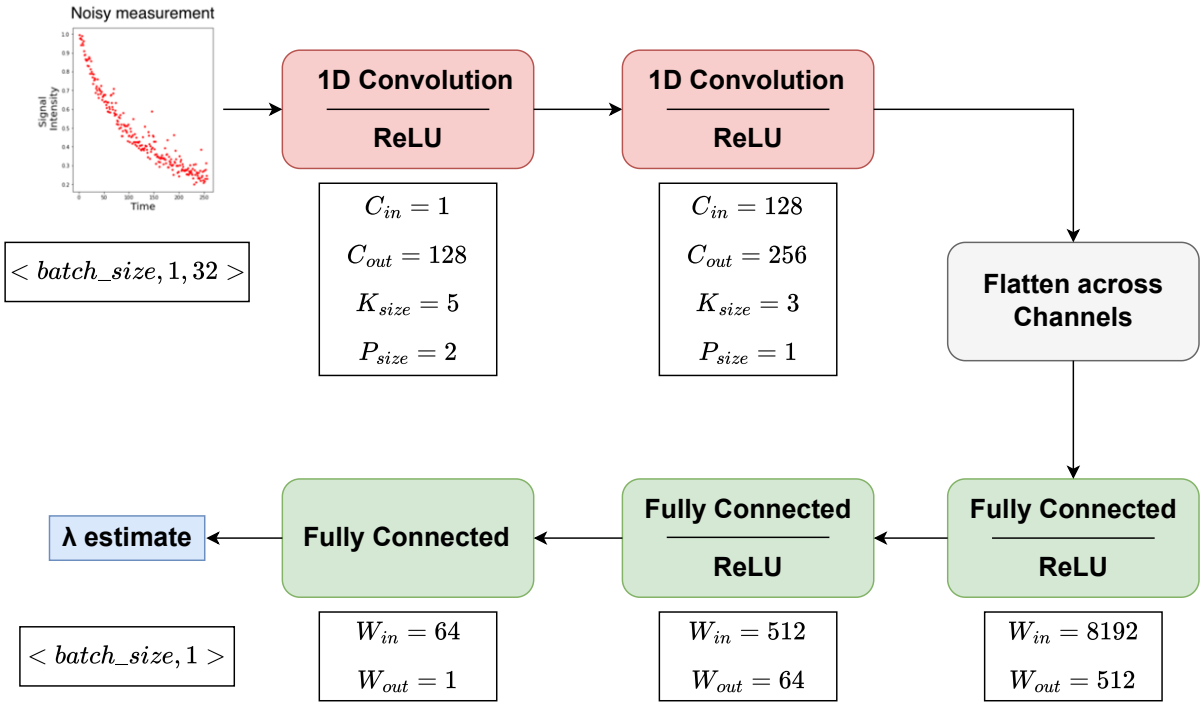


Figure 4.3: λ_{NN} architecture. 1-dimensional convolutional layers are applied to the input noisy signal, followed by fully connected layers to predict λ_{oracle}. C refers to the channel sizes (in/out for input or output channels, respectively). We also define the kernel size as K_{size} and the padding as P_{size}. W refers to the width of a given fully-connected layer. The ReLU activation function is used between each layer.

4.3.3.3 Training λ_{NN}

The architecture of λ_{NN} is provided in Figure 4.3 since it uses a combination of convolutional and fully connected layers. We train it for 30 epochs with the default parameters of an Adam optimizer, with a batch size 64, and a learning rate 10^{-5}.

4.3.3.4 Training ILR networks and (ND, ND)

To train our (ND, ND), (ND, REG)$_{\text{NN}}$, and (ND, REG)$_{\text{GCV}}$ models to learn the parameter estimate from the given input. We use a four layer fully connected NN with ReLU activations and

dimensions $32 \to 256 \to 64 \to 256 \to 3$. We train this network for 30 epochs, and a batch size of 64 with learning rate 10^{-5} using the default parameters of the Adam optimizer.

4.4 Results

4.4.1 Parameter Estimation

4.4.1.1 Estimating λ_{oracle}

Learning the signal-dependent optimal Tikhonov regularization parameter λ_{oracle} is a problem of independent interest in inverse problems. Here we present our results for estimation of the optimal λ for the three-parameter problem Eq. (4.5).

Figure 4.4 shows that the λ-selection network λ_{NN} when trained on the L^1 loss for solving the bilevel problem (4.9) closely approximates λ_{oracle} over unseen examples of noisy signals $\mathbf{s}$. As expected, estimation quality improves with values clustering more closely to the diagonal as SNR increases.

In Table 4.3, we compare λ_{NN} and λ_{GCV} in terms of reproducing the distribution of the values of λ_{oracle} as quantified by the *earth mover's distance*, also known as the Wasserstein-1 distance. We find that the distribution of the ensemble $\{\lambda_{\text{NN}}(\mathbf{s})\}$ is closer to the true ensemble than the distribution of $\{\lambda_{\text{GCV}}(\mathbf{s})\}$ (Table 4.3).

In Figure 4.5 we disaggregate the values in Table 4.3 by comparing the ensembles of $\{\lambda_{\text{NN}}(s)\}$, $\{\lambda_{\text{GCV}}(s)\}$, and $\{\lambda_{\text{oracle}}(s)\}$ values over the testing set. Despite λ_{NN} outperforming λ_{GCV} according to the $\mathcal{W}_1$ distance overall, λ_{GCV} replicates the distribution of λ_{oracle} more faithfully in the regime of low SNR and small λ, $\lambda \in [10^{-7}, 10^{-4}]$. These results can be described by histogram multimodality and spread in

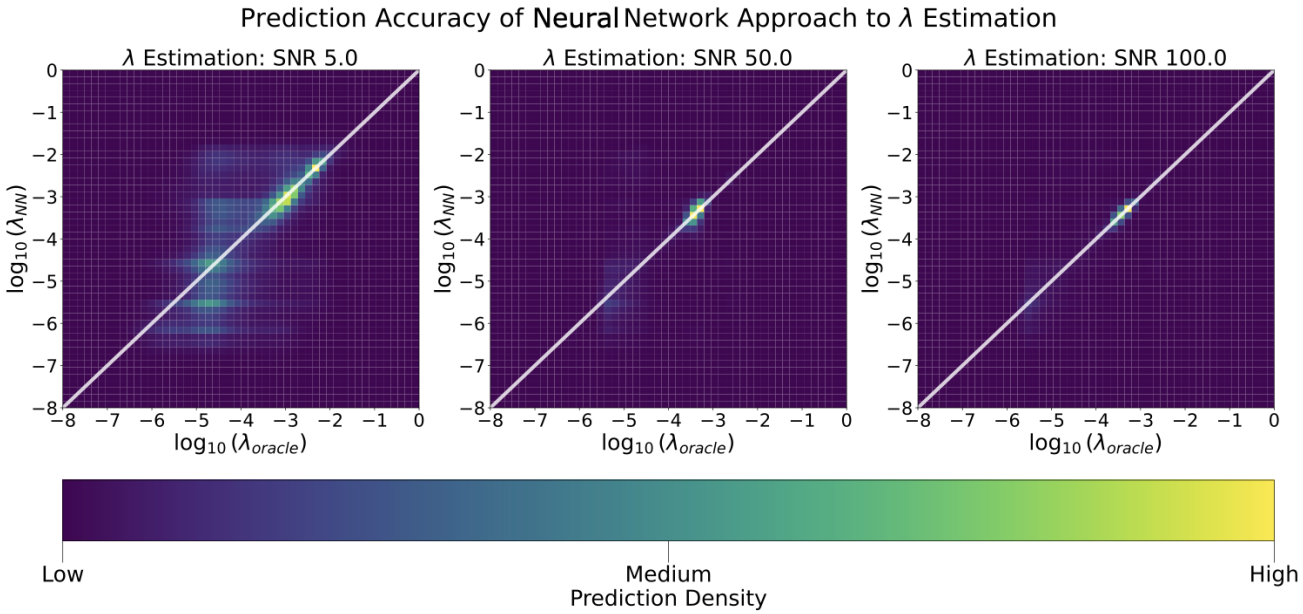


Figure 4.4: 3D Histogram showing prediction density of the λ-selection neural network versus the λ_{oracle} approach. The white line represents the $y = x$ line, indicating the benchmark for perfect prediction. Left to right: SNRs 5 (low), 50 (medium), 100 (high).

this regime, where the λ_{oracle} distribution has greater spread and exhibits multimodality that is more accurately captured by GCV than by the NN λ estimation, which tends to provide a single dominating mode for these lower values of λ (left panel, Figure 4.5). Thus, although the median λ_{oracle} is larger for low SNR as expected, the strategy for estimating λ_{oracle} accurately over the ensemble of noisy signals $\mathbf{s}$ depends on higher order statistics of the λ_{oracle} distribution. This suggests that in real applications, an initial noise-estimation step may be of benefit when deciding between use of λ_{GCV} and λ_{NN}.

4.4.1.2 Accuracy of ILR and Alternative Methods for c_1 Estimation

We have defined the two ILR NN, $(\text{ND}, \text{REG})_{\text{GCV}}$ and $(\text{ND}, \text{REG})_{\text{NN}}$, above. For comparison with conventional NN estimation, we duplicate the native noisy input vector to establish an input of the same length as the ILR NN's and thereby eliminate input signal length as a potential confounder. We call this NN (ND, ND). Additional standards of comparison are the conventional NLLS estimation

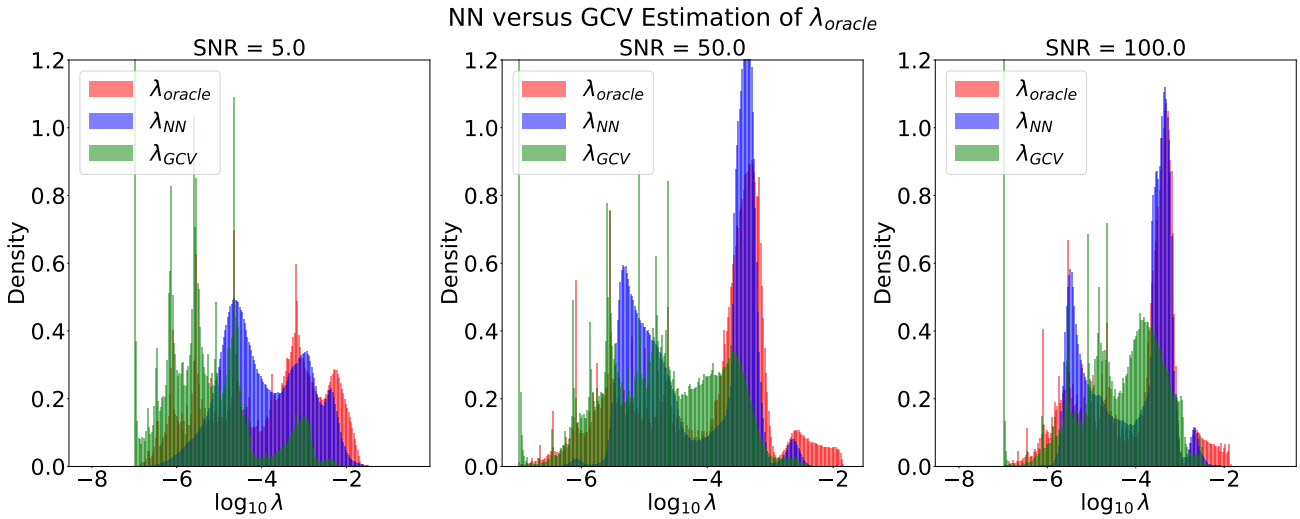


Figure 4.5: Left to right: The true λ distribution (orange) overlaid with the distributions of λ_{NN} and λ_{GCV}. The true distributions at the low SNR level are more multimodal and have greater spread, with GCV providing superior estimates as compared to NN λ selection. This is particularly seen at lower values of λ, where little to no regularization is needed. At the higher SNR's, the optimal λ values are concentrated around a small number of modes, with approximation of λ_{oracle} by the NN now exhibiting performance superior to GCV.

method, as well as the Tiknonov-regularized (TR-NLLS) approach in which regularization is applied to this parameter estimation problem; in our case, we have implemented this with selection of λ via GCV.

Results for estimation of c_1, representing MWF in our motivating application, are shown in Table 4.4. Several points of interest are evident. First, (ND, REG)$_{\text{GCV}}$ and (ND, REG)$_{\text{NN}}$ perform essentially indistinguishably. However, both of these ILR-based NN outperform the NN without ILR, (ND, ND), across all SNR levels. This is our main result. We also note (not shown) that the NN architecture without ILR and without duplication of the input vector, deemed (ND), performs virtually identically with (ND, ND). Further, the most conventional analysis, NLLS, exhibits performance substantially inferior to any of the NN's. In addition, the non-conventional method of TR-NLLS, in spite of the introduction of bias through regularization, provides sufficient stabilization that its overall

performance is markedly superior to that of NLLS. In appendix B, we provide a more detailed study of (ND, REG)$_{\text{NN}}$ and (ND, REG)$_{\text{GCV}}$ by disaggregating the RMSE over all parameter combinations appearing in the testing data.

4.4.1.3 Disaggregating Table 2

Although Table 4.4 shows that our method outperforms both plain deep learning and classical methods overall, it is of interest to define the regions of parameter space (c_1, T_{21}, T_{22}) in which ILR is particularly beneficial over the plain neural network (ND, ND). To investigate this issue, we compared the performance of the two NNs, (ND, REG)$_{NN}$ and (ND, REG)$_{GCV}$ against (ND, ND) as follows. First, let $c_1^{(ND,Reg)}$, $c_1^{(ND,ND)}$ be the c_1 parameter estimates produced by the (ND, REG) and (ND, ND) networks, respectively, where the subscript NN or GCV has been omitted for clarity here and below. Now given a fixed c_1^{true}, for each underlying pair (T_{21}, T_{22}) we calculate the differences between the relative RMSE of the two estimates:

$$\widehat{d}(T_{21}, T_{22}) = RMSE(c_1^{(ND,Reg)}, c_1^{true})/c_1^{true} - \tag{4.27}$$

$$RMSE(c_1^{(ND,ND)}, c_1^{true})/c_1^{true}, \tag{4.28}$$

$$d(T_{21}, T_{22}) = \frac{\widehat{d}(T_{21}, T_{22})}{\max\limits_{T_{21},T_{22}} |\widehat{d}(T_{21}, T_{22})|}. \tag{4.29}$$

Here the $RMSE$ is taken over 400 noise realizations present in the test set. From this, $\widehat{d}(T_{21}, T_{22}) < 0$ corresponds to (ND,REG) outperforming (ND, ND) for the corresponding pair of (T_{21}, T_{22}). We also consider a normalized version of this error metric given by $d(T_{21}, T_{22})$ described in (4.29) where the values of $\widehat{d}$ are rescaled to be in the interval $[-1, 1]$. We visualize $d(T_{21}, T_{22})$ in Figures 4.6 and 4.7.

Our findings are as follows.

(Low SNR) In the low SNR (SNR = 5.0) regime (first panels from top in Figures 4.6 and 4.7), the two (ND, REG) networks markedly exceed the (ND, ND) networks in performance for the $c_1 \geq 0.32$ region while the (ND, ND) networks attain superiority when $c_1 \leq 0.32$. In this latter range of c_1 values, (ND, REG)$_{\text{NN}}$ and (ND, REG)$_{\text{GCV}}$ perform slightly differently. Specifically, when $0.25 \leq c_1 \leq 0.32$, (ND, REG)$_{\text{NN}}$ outperforms (ND, ND) for more values of (T_{21}, T_{22}) than (ND, REG)$_{\text{NN}}$.

Thus, in regions of parameter space where (ND, REG) outperforms (ND, ND), the ILR network with GCV selection given by (ND, REG)$_{\text{GCV}}$ attains greater accuracy than (ND, REG)$_{\text{NN}}$ (thus registering a lower aggregate RMSE in row 1 of Table 4.4). However, (ND, REG)$_{\text{NN}}$ outperforms (ND, ND) for larger regions of parameter space than (ND, REG)$_{\text{GCV}}$.

Medium and high SNR When $SNR = 50, 100$, both of the (ND, REG) networks exhibit distinct patterns in three different ranges of c_1 space:

1. $0.0 \leq c_1 \leq 0.25$: In the low c_1 range the (ND, REG) networks outperform (ND, ND). This gain in performance is specifically more pronounced in the $T_{22} - T_{21} \approx 20$ region, as evidenced by the diagonal blue bands in the lower left corners in the central cells of the SNR 50 and 100 panels in Figures 4.6 and 4.7. However, as the difference $T_{22} - T_{21}$ decreases further to zero, there is a greater prevalence of red pixels in the left corner of each cell, indicating that when the signal is nearly monoexponential, ILR likely introduces inaccuracies stemming from model misspecification. More generally, the difference between the decay times, $T_{22} - T_{21}$, is an indicator of the ill-posedness of the biexponential problem. Our results in Figures 4.6 and 4.7 for this range of c_1 space may thus be qualitatively described as follows: in the regions of parameter space with a large weighting of the faster decay (i.e $c_1 \leq 0.25$), ILR is particularly advantageous

when the problem features an intermediate level of ill-posedness (i.e., $T_{22} - T_{21} \approx 20$). However, when the time constants are nearly identical, (ND, ND) exhibits superior performance.

2. $0.25 \leq c_1 \leq 0.4$: In the intermediate c_1 range, the (ND, ND) networks largely outperform the (ND, REG) networks. Moreover, the (ND, ND) networks specifically attain lower RMSE in the ill-posed region where $T_{21} \approx T_{22}$.

3. $0.4 \leq c_1 \leq 0.6$: This is the region of parameter space where the two time-decays are nearly equally weighted. In this region of equal weighting, ILR is substantially better than plain neural network estimation, with the most significant improvements occurring in the ill-posed regions. This phenomenon persists for both medium and high SNRs for both (ND, REG) networks.

Does ILR behave like classical regularization? For the NLLS estimator, regularization improves parameter estimation in the regions of ill-posedness, that is, when T_{21} and T_{22} are within, roughly, a factor of 2 or 3 of each other. A natural question then is whether ILR acts as a regularizer for the parameter estimation problem; this can be evaluated by examining its performance in ill-posed and well-posed regimes of parameter space. In Figures 4.6 and 4.7, the lower left corners are the most ill-posed regimes. As seen, the effect of ILR is to improve overall performance across the diagrams, but without the improvement generally being concentrated in the ill-posed regime.

4.4.1.4 Estimating time decay constants

Although the MWF parameter c_1 was of central interest in this paper, the estimation of the time decay constants T_{21} and T_{22} is also an independently important problem in magnetic resonance. In Tables 4.5 and 4.6 we include results for estimating T_{21} and T_{22} respectively, noting that our ILR networks (ND, REG)$_{\text{NN}}$ and (ND, REG)$_{\text{GCV}}$ output all three parameters at once. Therefore, no additional

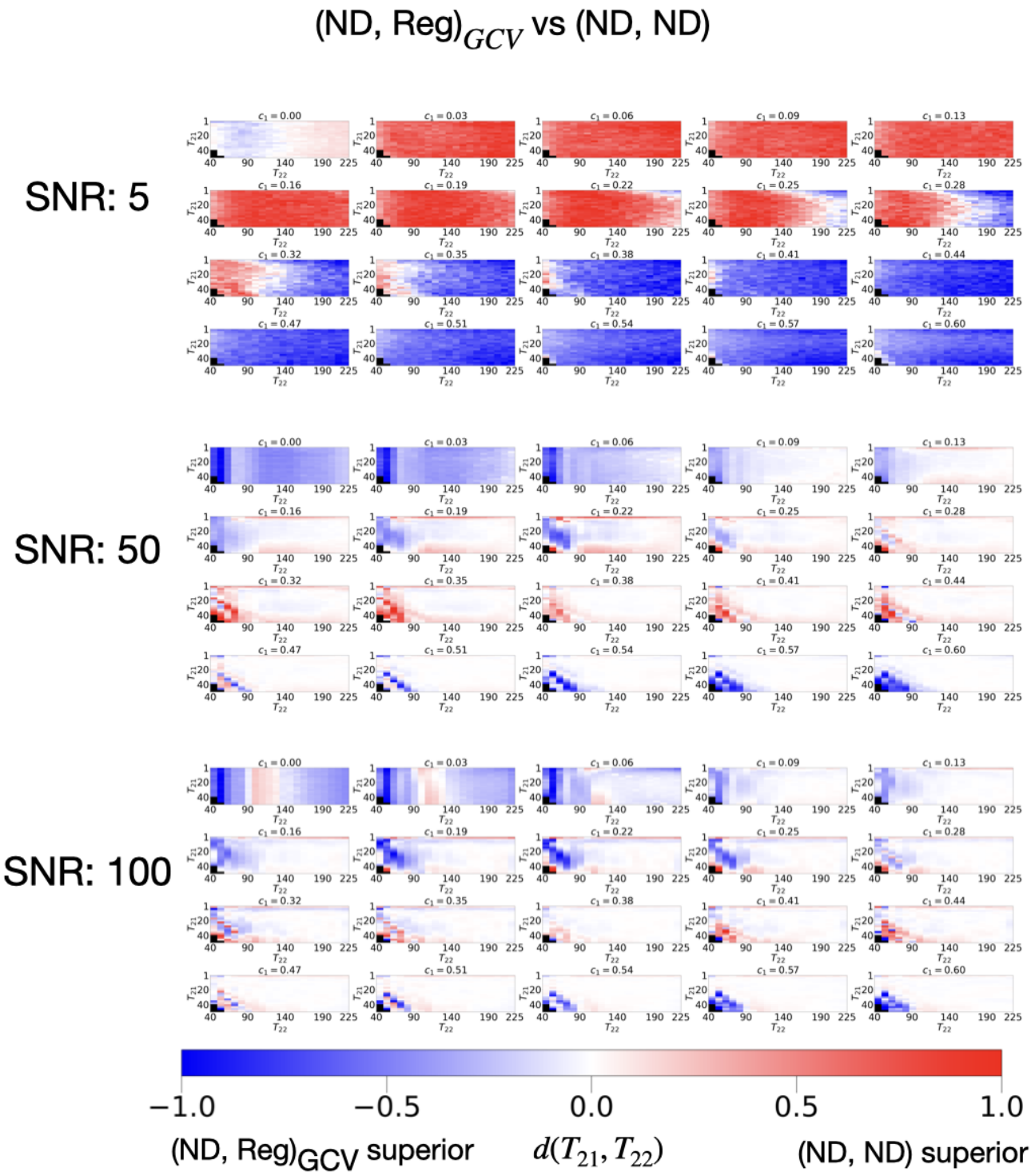


Figure 4.6: Top to bottom: Computing the differences in relative RMSE between (ND, REG)$_{\text{GCV}}$ and (ND, ND). Each panel is an image over the (T_{21}, T_{22}) space where the color blue (resp. red) represents the superiority of (ND, REG)$_{\text{GCV}}$ (resp. (ND, ND)) in estimating the MWF value given by c_1 denoted above the panel.

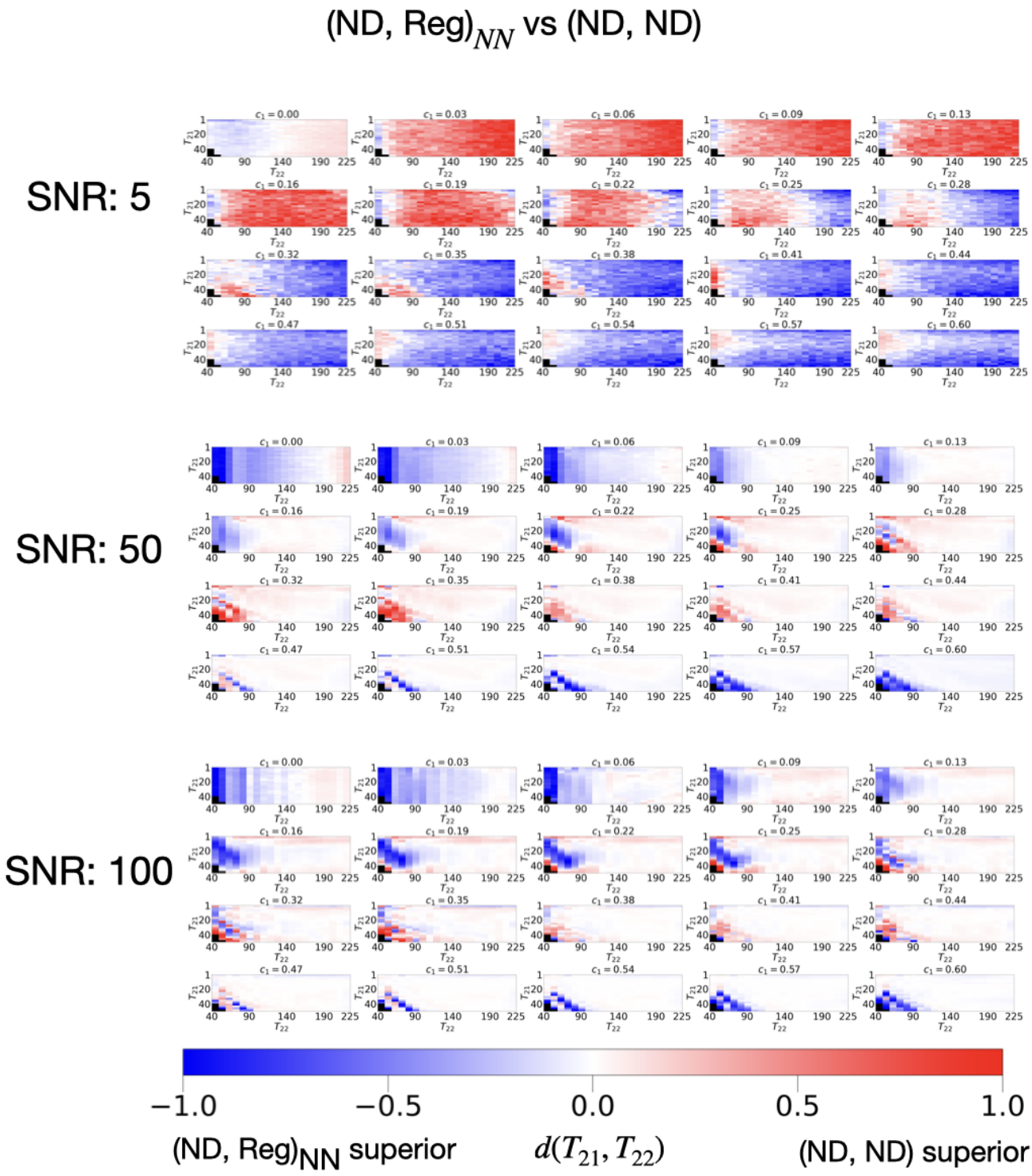


Figure 4.7: Results as for Figure 4.6, but for (ND, REG)$_{\text{NN}}$.

training or design is required to infer T_{21} and T_{22}.

4.4.2 Myelin water fraction estimation

We now apply ILR to the estimation of the MWF from magnetic resonance relaxometry (MRR) signals from the human brain. A multi-spin-echo sequence with 64 values of TE was applied as outlined in the Appendix, with the sampling scheme leading to the decay signal as described in Eq. (4.2).

However, we note that while the white matter of the brain, rich in myelin content, is expected to exhibit biexponential decay, the grey matter, with little-to-no myelin, is more likely to be better described by a monoexponential decay. For the latter, Eq. (4.2) represents an underdetermined model, and no amount of regularization or other manipulation will permit meaningful parameter values to be extracted. The determination of two exponentials will be enforced by the model, with their fractions c_1 and c_2 noise-dependent and therefore random. Therefore, we wish to apply our analysis only to truly biexponential signals. Accordingly, we evaluate each pixel using the Akaike information criterion (AIC) [JWH+13], and analyze signals only from pixels that are better described as biexponential rather than monoexponential. We note that the distribution of the AIC-defined mono- and bi-exponential pixels closely follows the general pattern of cortical grey and interior white matter in the brain.

4.4.2.1 Benchmarking

With this, the time-series for each voxel under consideration is a single noisy biexponential curve with parameters to be estimated via ILR with signal-dependent λ selection. To create a reference set of parameters, we used nonlocal estimation of multispectral magnitudes (NESMA) denoising filter

[BRM+18], followed by five parameter NLLS estimation. The NESMA estimates for c_1 are visualized in Figure 4.8. We recognize that the NESMA-filtered NLLS parameters are limited in reflecting the true underlying parameters; therefore, the simulation data may be more indicative of model performance on the biexponential signal in this sense.

We also benchmark our method against NLLS, TR-NLLS with λ_{GCV} and the deep learning method known as model informed machine learning [YCRP+21] (MI-ML). In MI-ML, a neural network trained to recover $\mathbf{p}_{true}$ from a loss given by a combination of MSE and Wasserstein-1 loss, which compares the distance between $\mathbf{p}_{est}$ and $\mathbf{p}_{true}$ when viewed as distributions. We provide further details on MI-ML in 4.4.2.5. We retrain both (ND, ND) and (ND, REG) models with this loss function.

As suggested in simulations, high quality performance of the (ND, REG) networks is predicated on proper selection of the regularization parameter λ using either GCV or NN; in our case, this means selection of λ that is similar to λ_{oracle}, or at least that provides comparable performance. Furthermore, we illustrated that the choice of GCV or NN can be made based on estimation of SNR. Using a comparable method of SNR quantification, we found median SNR of ≈ 25.0 on the biexponential pixels identified by the AIC, suggesting that the underlying distribution of λ_{oracle} was likely to be spread over orders of magnitudes with a large number of modes, as in Figure 4.5. Table 4.4 indicates that superior results may be obtained from (ND, REG)$_{\text{GCV}}$ as compared to (ND, REG)$_{\text{NN}}$.

Indeed, the left column of Figures 4.9 and 4.10 show that (ND, REG)$_{\text{GCV}}$ is in general superior to (ND, REG)$_{\text{NN}}$. This is also consistent with Figure 4.5, suggesting that λ_{GCV} may more accurately reproduce the modes of the distribution of λ_{oracle}; this capability is not captured by either the L^1 loss or the earth mover's distance. Therefore, λ_{GCV} may be preferable for use with ILR with limited SNR.

Adopting (ND, REG)$_{\text{GCV}}$ as the method of choice for this application, we compared its performance with that of the (ND, ND) network, as shown in the middle column of Figures 4.9 and 4.10. We find that (ND, REG)$_{\text{GCV}}$ outperforms (ND, ND) nearly *everywhere*.

We further expand our significance by benchmarking our best-performing method to the MI-ML model as described in 4.1.2. We see that (ND, REG)$_{\text{GCV}}$ performs superior to MI-ML across almost all pixels, as shown in the right column of 4.9 and 4.10. The summary of root mean squared deviation from MWF^{NESMA} are reported in Table 4.7, with (ND, REG)$_{\text{GCV}}$ performing superiorly for all slices across all patients. From Table 4.8, it is evident that our best performing method is a significant gain over alternative methods. Together, these results indicate the efficacy of ILR to enhance NN-based MWF estimation. It is notable that optimal performance requires combining this parameter estimation NN with a classical technique for selection of the regularization parameter λ, λ_{GCV}, rather than selection according to the λ_{NN}, for generating the augmented part of the noisy input signal.

Below we provide a complete summary of the data and methods used in the above experiment on in-vivo data.

4.4.2.2 MRI acquisition protocol

After obtaining written informed consent, imaging was performed under our IRB protocol #08AG0238 for data collection, imaging was performed with a 3T Philips MRI system (Achieva, Best, The Netherlands), with an internal quadrature bodycoil for transmission and an eight-channel phased array head coil for reception. We used the 3D GRASE (gradient and spin-echo) sequence to acquire data from the brain of a healthy 41-year-old male. Echo data were acquired for 32 echoes, with acquisition times of $t_n = n \times \Delta t$ where the echo spacing $\Delta t = 11.3ms$. An echo planar imaging (EPI) acceleration factor

of 3 was used, with a field of view (FOV) $278mm \times 200mm \times 30mm$, over an acquisition matrix size of $185 \times 133 \times 10$, for a voxel size of $1.5mm \times 1.5mm \times 1.5mm$, with $TR = 1000ms$. The image matrix was zero filled in k-space to yield a $1mm \times 1mm \times 1mm$ reconstruction. The duration of the scan was approximately 10 minutes. Sampling was performed at echo peaks and formed the signal data for analysis.

4.4.2.3 Noise filtering and quantification

Lacking a true "ground truth" for in vivo data, we used the NESMA filter [BRM+18] to obtain high SNR images from which to extract parameter values that serve as our (necessarily flawed) standard for comparison. This also provided a base image to which we added noise to achieve the levels of SNR as described in the main text, allowing us to implement a targeted application of an ILR network trained on data of low, medium, or high SNR. Figure 4.11 displays the calculated SNRs.

4.4.2.4 Akaike information criterion (AIC)

We wish to apply regularization only to signals well-described as biexponential rather than monoexponential. In the latter case, our biexponential model is undetermined, and the resulting parameter variances cannot be remediated by regularization or any other numerical approach. We used the Akaike information criterion (AIC) to identify biexponential signals. The AIC provides a criterion for determining whether the improved numerical fit that inevitably results from an increase in parameters for nested models is statistically meaningful. In the context of least squares regression, the AIC can be derived from the residual sum of squares (RSS). For a linear regression model with normally distributed errors, the likelihood L is proportional to $\exp\left(-\frac{RSS}{2\sigma^2}\right)$. This leads to the following expression for the

AIC:

$$AIC = n\log(\text{RSS}/n) + 2k$$

Here n is the number of data points, RSS is the residual sum of squares, and k is the number of parameters. In the context of MWF imaging, we consider two competing models for explaining the data:

$$\mathbf{G}_{monoexp}(t, c_1, T_{21}, b) = b + c_1 \exp(-t/T_{21}), \tag{4.30}$$

$$\mathbf{G}_{biexp}(t, c_1, T_{21}, c_2, T_{22}, b) = b + c_1 \exp(-t/T_{21}) \tag{4.31}$$

$$+ c_2 \exp(-t/T_{22}). \tag{4.32}$$

Here b is an offset, c_1, c_2 are compartment sizes, and T_{21} and T_{22} are exponential time constants, as defined in the text. The AIC provides a method to compare the suitability of these models. In the context of MWF imaging each voxel can be regarded as containing a mono or bi-exponentially decaying signal corrupted by Rician noise. The RSS is not strictly an estimate of the likelihood of the data for non-Gaussian noise, but our SNR range after NESMA filtering is sufficiently large that the difference between Gaussian and Rician noise can be neglected.

4.4.2.5 Model informed machine learning [YCRP+21]

In MI-ML a neural network f_θ is trained with the following loss function given by a weighted sum of MSE loss with the Wasserstein-1 distance:

$$L_{MI-ML}(\theta) := \frac{1}{n}\sum_{i=1}^{n} \|\mathbf{p}^i_{true} - f_\theta(\mathbf{s}^i)\|_2^2 + \alpha\frac{1}{n}\sum_{i=1}^{n} \mathcal{W}_1(\mathbf{p}^i_{true}, \mathbf{f}_\theta(\mathbf{s}^\mathbf{i})). \tag{4.33}$$

To define $\mathcal{W}_1(\mathbf{p}^i_{true}, \mathbf{f}_\theta(\mathbf{s^i}))$ between the true parameters and the estimated parameters, we first summarize the **Wasserstein-1 distance** (also called the Earth Mover's Distance) between two probability distributions $\mu, \nu \in \mathcal{P}(\mathbb{R})$:

$$\mathcal{W}_1(\mu, \nu) \;=\; \inf_{\gamma \in \Pi(\mu,\nu)} \int_{\mathbb{R}\times\mathbb{R}} |x - y| \, d\gamma(x, y), \tag{4.34}$$

where $\Pi(\mu, \nu)$ denotes the set of all couplings of μ and ν. If F_μ and F_ν denote the cumulative distribution functions (CDFs) of μ and ν, then

$$\mathcal{W}_1(\mu, \nu) \;=\; \int_{-\infty}^{\infty} \big|F_\mu(x) - F_\nu(x)\big| \, dx, \tag{4.35}$$

that is, the Wasserstein-1 distance is exactly the L^1-distance between the two CDFs. Letting $\mathbf{p}_{est} = (c_1^{est}, T_{21}^{est}, T_{22}^{est}) = f_\theta(\mathbf{s}^i)$, we may construct the distribution $\mathcal{P}_{est} := c_1^{est}\delta_{1/T_{21}^{est}} + (1 - c_1^{est})\delta_{1/T_{22}^{est}}$ (and similarly construct $\mathcal{P}_{true}$ from $\mathbf{p}_{true}$. Then

$$\mathcal{W}_1(\mathbf{p}_{true}, \mathbf{p}_{est}) := \mathcal{W}_1(\mathcal{P}_{true}, \mathcal{P}_{est}). \tag{4.36}$$

Note that $\mathcal{W}_1(\mathcal{P}_{true}, \mathcal{P}_{est})$ scales like $O(c_1/T_{21})$ and is therefore at an order of magnitude of 10^{-2}. To keep this order of magnitude commensurate with the MSE loss, we use $\alpha = 50$ so that all terms are $O(1)$.

4.5 Discussion

Combining conventional analytic methodology and domain knowledge with machine learning-based methods has become a common paradigm [ZCD+21, GXH+22, CDCG+22]. Examples include incorporating physical constraints into loss functions [CMW+21], seeding classical solvers with machine-learned initializations [ZHRB23], and designing individual layers of neural networks around the physics of the problem [VHSF+21]. Here we have presented a new approach that is an additional example of such a methodology by integrating the classical technique of Tikhonov regularization [BO04, LW13, SGG+09] and bilevel optimization [CMS07, DZ20] with neural network data concatenation and automated hyperparameter tuning. This leads to a refined approximation of the map from a noisy signal $\mathbf{s}$ to a parameter estimate $\widehat{\mathbf{p}}$. In particular, we have presented a cascaded approach to parameter estimation with two levels: (1) estimating the hyperparameter λ from the noisy signal $\mathbf{s}$ followed by (2) estimating the parameters of a decaying exponential using ILR, with the regularized component of the concatenated signal generated using λ from the previous step. As demonstrated in section 4.2, this can be viewed as an instantiation of the empirical Bayes formulation.

For the first level of analysis, we explored two strategies for recovering λ from the noisy signal $\mathbf{s}$, which can itself be viewed as a parameter estimation problem. The first strategy involves selection of lambda using GCV [GHW79], a common, statistically validated, method that does not require an estimate of signal noise level. This approach requires no training, but rather evaluates each signal individually and estimates the optimal lambda by what is in effect a leave-one-out cross-validation. The second strategy is a neural network [ACC21] trained with a loss function involving λ_{oracle}. Both methods involve computational bottlenecks related to the solutions of optimization problems. The GCV estimate requires minimization over $[0, \infty)$ for the GCV functional, which can be difficult to op-

timize with gradient methods, while the NN approach requires the generation of λ_{oracle} values through an expensive grid search strategy sweeping over several orders of magnitude. Nonetheless, we see that a four-layer CNN can approximate the map $\mathbf{s} \rightarrow \lambda(\mathbf{s})$ accurately, suggesting that even a modest number of layers can suffice for approximating solutions to this highly unstable problem. We have also demonstrated that the L^1 loss is an effective tool to approximate not only the map $\mathbf{s} \rightarrow \lambda_{oracle}(\mathbf{s})$ but also the *ensemble* of λ_{oracle} considered as a probability distribution. The λ_{NN} outperforms λ_{GCV} with respect to both of these error metrics. However, we also find that at lower SNR, λ_{GCV} appears to be superior in deriving parameters from noisy signals that do not require substantial regularization. The effect of NN depth, loss functions, and selection of activation functions on estimation of optimal λ values remains an open question.

For the second level of our analysis pipeline, we used λ_{GCV} and λ_{NN} to construct a regularized signal which was concatenated with the noisy signal, with this concatenated construct used to train the ILR networks (ND, REG)$_{\text{NN}}$ and (ND, REG)$_{\text{GCV}}$. Both of these (ND, REG) networks, differing only in the manner in which λ was selected, attained over 5–10% or greater improvements in accuracy compared to the basic ND NN and TR-NLLS estimation. This indicates that ILR is a promising approach for overcoming the limitations of traditional methods when performing parameter estimation from noisy decay signals. In particular, our results suggest that ILR may significantly improve the accuracy and reliability of parameter estimation problems in magnetic resonance relaxometry, a topic of great current interest for gaining insights into the pathophysiology of the central nervous system.

In actual brain data, there is no gold standard for parameter estimation, and surrogates for actual underlying values must be used instead. In the present case, we applied an effective noise-reduction filter to obtain the benchmark results shown in Figure 4.8. Nevertheless, application to an actual in vivo system is an important part of demonstrating the potential utility of the ILR approach, especially

since actual data is unlikely to strictly obey any imposed signal model. We find that the “hybridized" approach–where the λ is picked with GCV followed by NN estimation with ILR is superior to pure deep learning or pure classical estimation.

The data-concatenation approach we have developed and implemented demonstrates a substantial improvement in the accuracy of parameter estimation for the biexponential decay signal. Nevertheless, there are certain limitations to our study which may form the basis for additional investigation. The present analysis does not incorporate the spatial correlations between pixels which exist in any natural image. Instead, our λ_{NN} networks are trained to process data arising from individual decay curves. Thus, although λ_{NN} provides meaningful estimates of λ_{oracle}, further optimization of its architecture, including potential modifications of the penalty function to penalize rapid spatial variation, may improve results. The challenges in this approach are to avoid the introduction of bias and of blurring.

Additionally, although we have carefully tuned the hyperparameters for our various NN’s, this remains something of a black art in the literature and results could change somewhat with additional tuning. We have elected to perform our investigations using the Rician noise model as the basic model in MRI, and there is nothing in our analysis that would be specific to this choice. Nevertheless, investigation of other models, and in particular the Gaussian, would be of interest. In addition, as noted above, we have demonstrated our method on actual brain data, but these results are somewhat inferential, since there is no true gold standard for underlying parameter values. We also note that one of our comparison methods, TR-NLLS, is highly nonstandard. We have however found this to be an effective modification of NLLS in many settings and are exploring this more systematically. Further, we applied the AIC to define pixels for which signal regularization makes sense; this appears to be a novel approach in the literature and has proved to be very effective here. However, there are several

choices, including the F-test, the Bayesian information criterion, and reduced chi-squared, to address this issue, and another choice may exhibit superior performance. Lastly, we highlight the fact that our work as presented here explores only the biexponential model, with emphasis on determination of the c_1 parameter. Other signal models and parameter estimation problems should also be evaluated in this framework.

In summary, we have addressed the recovery of the parameter c_1 in the biexponential model (4.3) for MWF estimation in the central nervous system, a topic of ongoing research interest. On both simulated data and actual brain data, we find ILR to offer a significant improvement in parameter estimation accuracy.

Table 4.2: Parameters defined for dataset creation

.

Parameter	**Symbol**	**Value**
Number of signal acquisition times	N_t	32
Lower and upper bounds on regularization parameter search space	$\log_{10}(\lambda_{oracle})$	-7,3
Initial guess for λ_{oracle}-solver (regularized-nonlinear least squares solver)	$\mathbf{z}$	$(T_{2,1}^{true}, T_{2,2}^{true}, c_1^{true})$
Lower and upper bounds on target fraction of more rapidly decaying component	c_1^l, c_1^h	[0.0,0.6] (Training), [0.0, 0.5] (Validation & Testing)
Fraction of more slowly decaying component	c_2	$1.0 - c_1$
Lower and upper bounds on target $T_{2,1}$	$T_{2,1}^l, T_{2,1}^h$	[1.0, 50.0] (Training), [5.0, 45.0] (Validation & Testing)
Lower and upper bounds on target $T_{2,2}$	$T_{2,2}^l, T_{2,2}^h$	[40.0, 225.0] (Training), [45.0, 200.0] (Validation & Testing)
Number of uniformly spaced T_2 and c_1 values	N_p	20 (per parameter)
Range of signal acquisition times		$[8.0, 256.0]$
Number of noise realizations	N_ω	1000 (Training), 200 (Validation), 400 (Testing)

SNR	5.0	50.0	100.0
$W_1(\lambda_{\text{NN}}(\mathbf{s}), \lambda_{\text{oracle}}(\mathbf{s}))$	0.348	0.246	0.145
$W_1(\lambda_{\text{GCV}}(\mathbf{s}), \lambda_{\text{oracle}}(\mathbf{s}))$	1.386	0.862	0.560

Table 4.3: The Wasserstein-1, or earth mover's distance (EMD), between the ensembles of λ values produced by the NN (upper row) and GCV (lower row). Note that the NN outperforms GCV in matching the distribution to λ_{oracle} for every SNR level. However, the multiple modes towards low values of λ are captured more accurately by GCV.

SNR	**5.0**	**50.0**	**100.0**
(ND, REG)$_{\text{NN}}$ RMSE$_{c_1}$	0.1652	0.0931*	0.0796*
(ND, REG)$_{\text{GCV}}$ RMSE$_{c_1}$	0.1643*	0.0945	0.0817
(ND, ND) RMSE$_{c_1}$	0.1671	0.1079	0.0935
TR-NLLS RMSE$_{c_1}$, λ_{oracle}	0.2328	0.2092	0.2078
TR-NLLS RMSE$_{c_1}$, λ_{GCV}	0.3687	0.3487	0.3347
NLLS RMSE$_{c_1}$	0.3844	0.3471	0.3453

Table 4.4: Root mean squared error (RMSE) in c_1 for each indicated parameter estimation method. * indicates superior performance for the given SNR category.

RMSE$_{T_{2,1}}$/SNR	**5.0**	**50.0**	**100.0**
(ND, Reg)$_{NN}$	14.45*	10.41*	8.55*
(ND, Reg)$_{GCV}$	14.45*	10.42	8.63
(ND, ND)	14.46	11.03	8.99
TR-NLLS, λ_{oracle}	23.03	18.46	18.40
TR-NLLS, λ_{GCV}	92.39	66.40	54.79
NLLS	38.18×10^6	41.32×10^5	16.71×10^5

Table 4.5: Root mean squared error (RMSE) in $T_{2,1}$ for each parameter estimation method. * indicates superior performance for the indicated SNR.

RMSE$_{T_{2,2}}$/SNR	**5.0**	**50.0**	**100.0**
(ND, Reg)$_{NN}$	36.77*	8.49*	5.31*
(ND, Reg)$_{GCV}$	36.80	8.50	5.31*
(ND, ND)	36.86	8.57	5.35
TR-NLLS, λ_{oracle}	32.85	7.99	7.53
TR-NLLS, λ_{GCV}	15.29×10^3	59.20×10^2	53.68×10^2
NLLS	41.61×10^7	12.86×10^7	13.69×10^7

Table 4.6: Root mean squared error (RMSE) in $T_{2,2}$ for each parameter estimation method. * indicates superior performance for the indicated SNR.

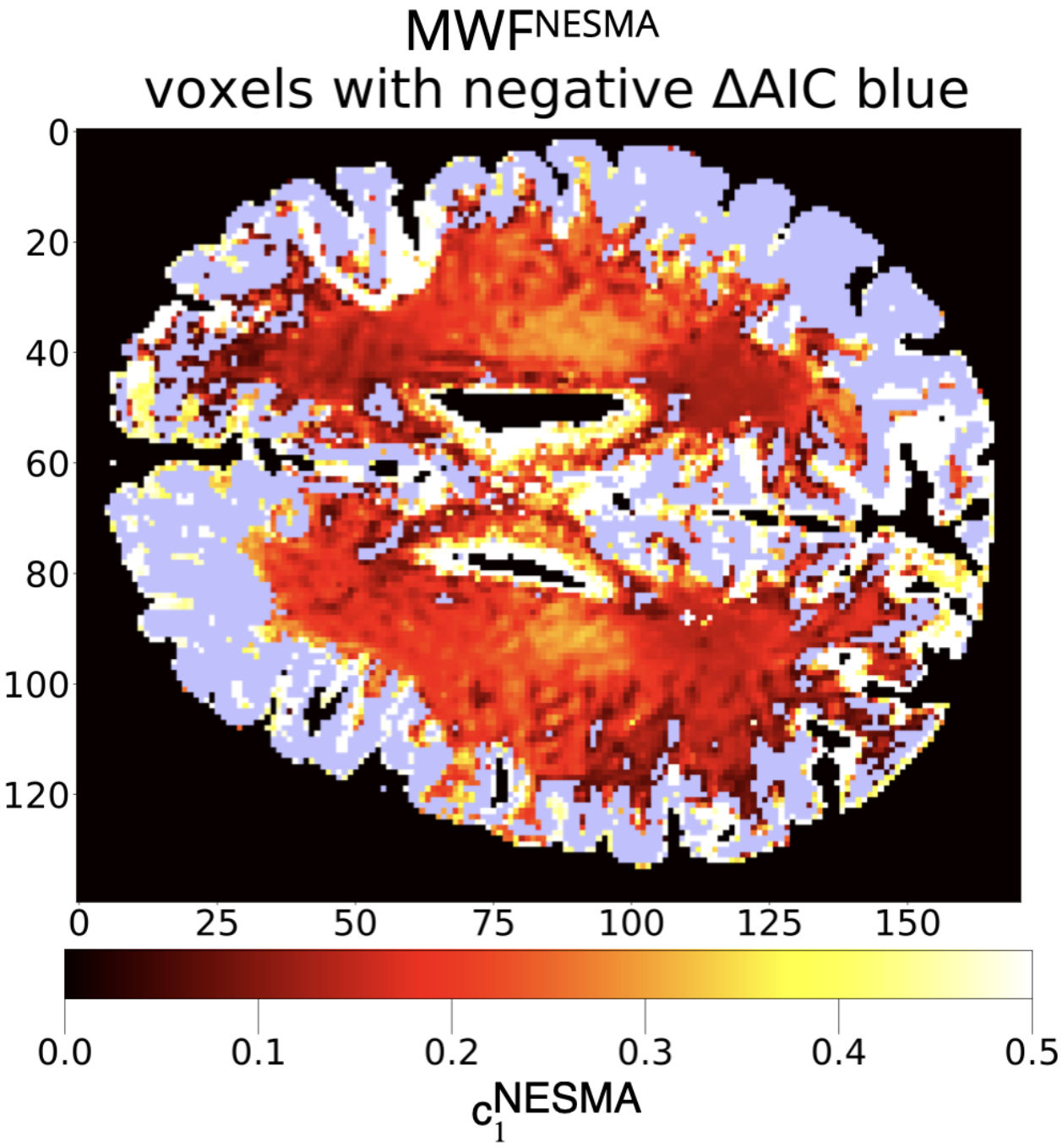


Figure 4.8: Comparison standard for c_1 for pixels determined to be biexponential according to the AIC (color bar); purple pixels are those determined to be of a monoexponential character. Each voxel in the 288 $\times$ 288 image of the brain is characterized by a noisy signal $\mathbf{s}$ with 64 acquisition times. Each of these voxel signals was evaluated according to a 5-parameter model after NESMA filtering [BRM+18]. Estimates for c_1 were taken as a comparison standard.

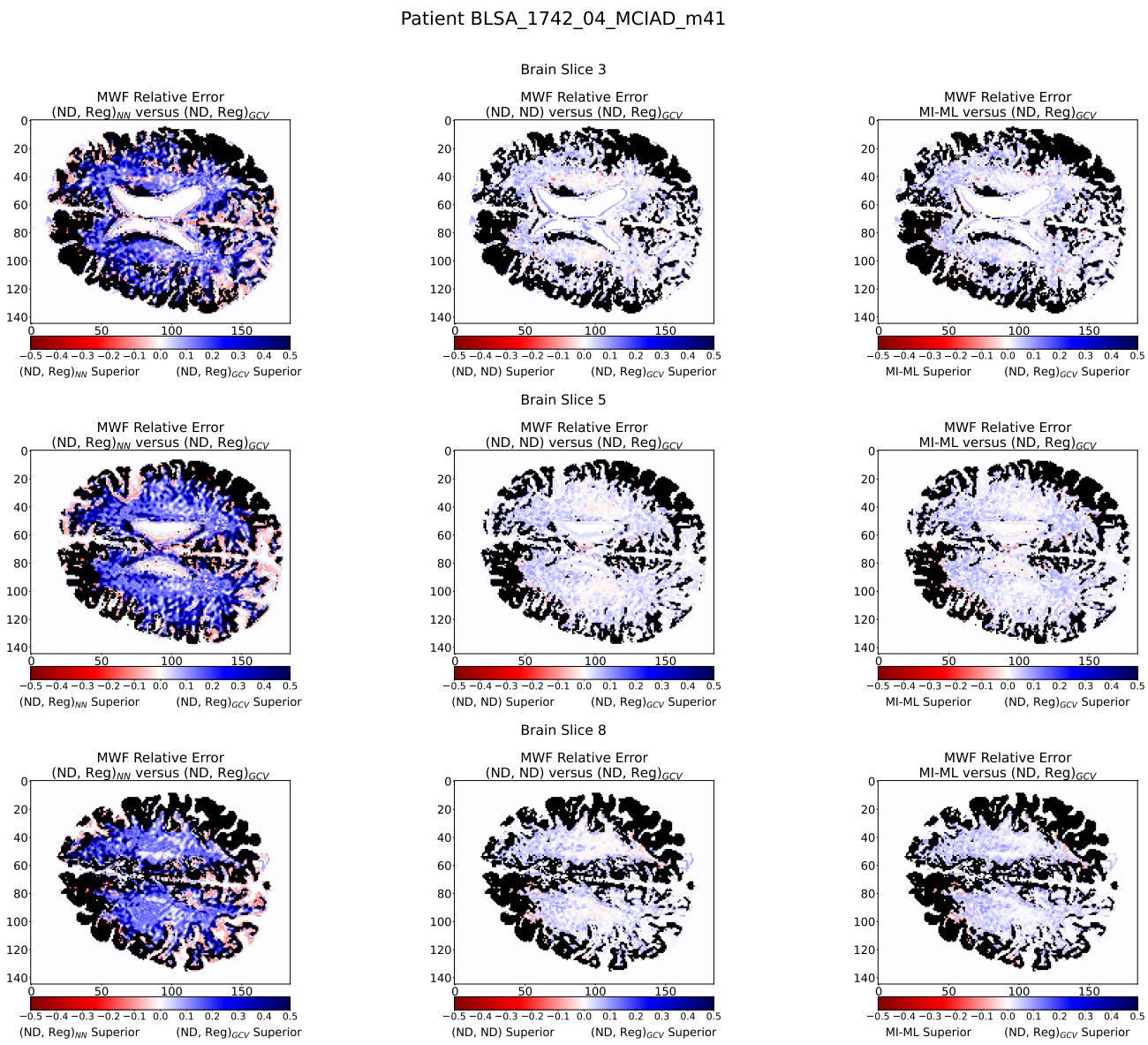


Figure 4.9: Comparison of c_1 estimation according to the two methods of λ selection and the MI-ML benchmarking model. The metric illustrated is the difference in absolute values of $|Method1_{c_1}^{est} - c_1^{\text{NESMA}}| - |Method2_{c_1}^{est} - c_1^{\text{NESMA}}|$. For example, the left column is computed via $|((\text{ND, REG})_{\text{NN}}^{c_1^{est}}) - c_1^{\text{NESMA}}| - |((\text{ND, REG})_{\text{GCV}}^{c_1^{est}}) - c_1^{\text{NESMA}}|$ calculated over pixels with a biexponential character. As seen, $(\text{ND, REG})_{\text{GCV}}$ is superior overall.

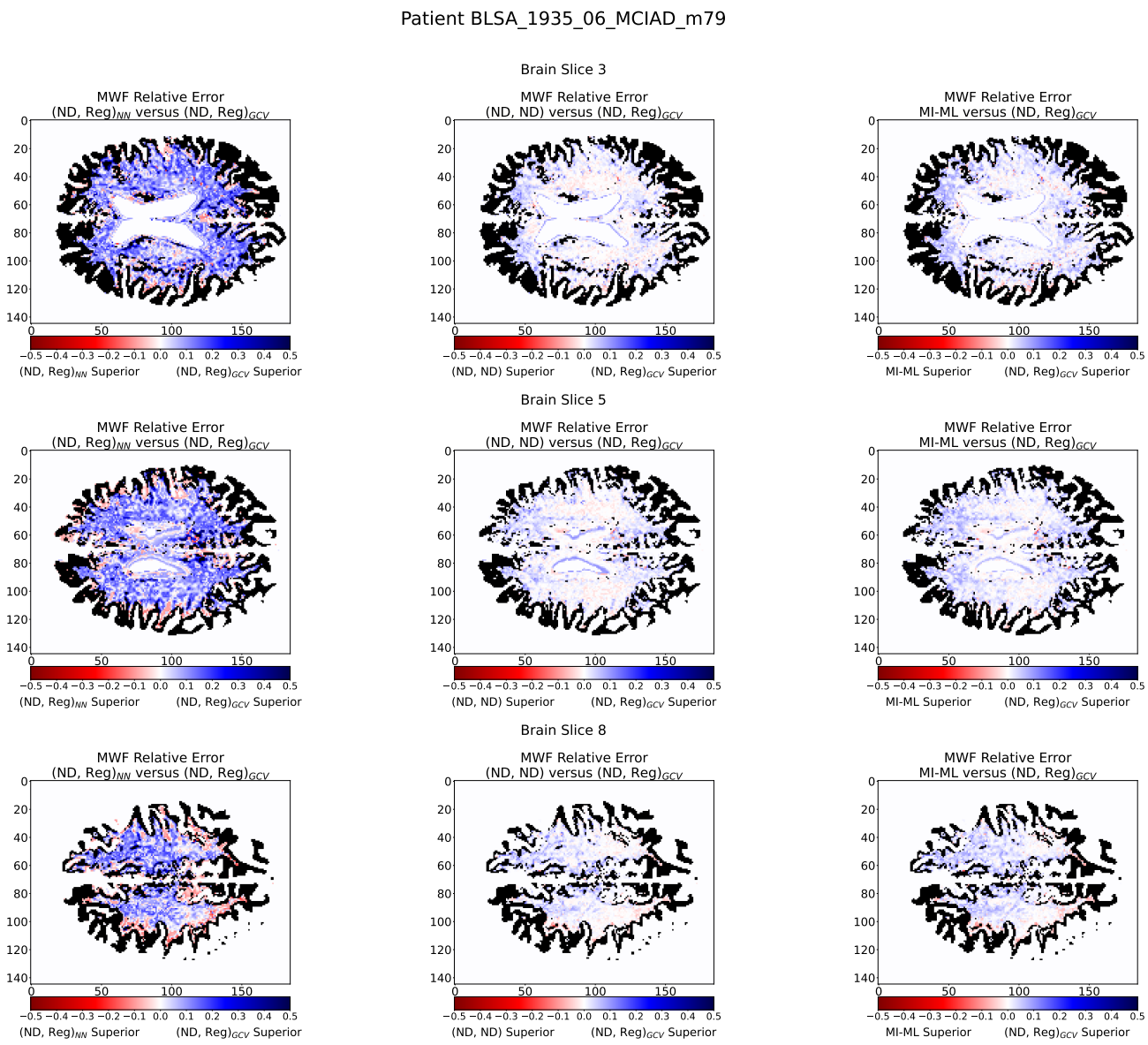


Figure 4.10: Comparison of c_1 estimation according to the two methods of λ selection and the MI-ML benchmarking model. The metric illustrated is the difference in absolute values of $|Method1_{c_1}^{est} - c_1^{\text{NESMA}}| - |Method2_{c_1}^{est} - c_1^{\text{NESMA}}|$. For example, the left column is computed via $|((\text{ND, REG})_{\text{NN}}^{c_1^{est}}) - c_1^{\text{NESMA}}| - |((\text{ND, REG})_{\text{GCV}}^{c_1^{est}}) - c_1^{\text{NESMA}}|$ calculated over pixels with a biexponential character. As seen, $(\text{ND, REG})_{\text{GCV}}$ is superior overall.

	Root Mean Squared Deviation from $\mathbf{MWF^{NESMA}}$						
Brain Slice	**(ND, ND)**	$(\text{ND, REG})_{\text{GCV}}$	$(\text{ND, REG})_{\text{NN}}$	**NLLS**	**MI-ML**	$\mathbf{MI-ML}(\text{ND, REG})_{\text{GCV}}$	$\mathbf{MI-ML}(\text{ND, REG})_{\text{NN}}$
m41-3	0.1277	**0.1147**	0.1818	0.2462	0.1273	0.1264	0.2348
m41-5	0.0744	**0.0673**	0.1118	0.1433	0.0740	0.0757	0.1422
m41-8	0.1004	**0.0898**	0.1449	0.2093	0.1016	0.1010	0.1966
m79-3	0.1015	**0.0944**	0.1406	0.2206	0.1038	0.1043	0.2267
m79-5	0.1046	**0.0973**	0.1390	0.2243	0.1066	0.1063	0.2242
m79-8	0.0717	**0.0666**	0.0871	0.1673	0.0731	0.0733	0.1506
Aggregate	0.0967	**0.0884**	0.1342	0.2018	0.0977	0.0978	0.1959

Table 4.7: [
Root mean squared deviation from $\text{MWF}^{\text{NESMA}}$ in MWF for each parameter estimation method]Root mean squared deviation from $\text{MWF}^{\text{NESMA}}$ in MWF for each parameter estimation method. The bolded values represent superior performance. The aggregate row represents the averaged result over all slices and patients.

	Percentage Improvement of Input Layer Regularization over Alternative Methods		
Brain Slice	(ND, REG)$_{GCV}$ vs **NLLS**	(ND, REG)$_{GCV}$ vs **(ND, ND)**	(ND, REG)$_{GCV}$ vs **MI-ML**
m41-3	53.41%	10.18%	9.90%
m41-5	53.04%	9.54%	9.05%
m41-8	57.10%	10.56%	11.61%
m79-3	57.21%	7.00%	9.06%
m79-5	56.62%	6.98%	8.72%
m79-8	60.20%	7.11%	8.89%
Aggregate	56.26%	8.56%	9.54%

Table 4.8: Percentage improvement of the best performing method (ND, REG)$_{GCV}$ over the respective alternative method. The aggregate row represents the averaged result over all slices and patients.

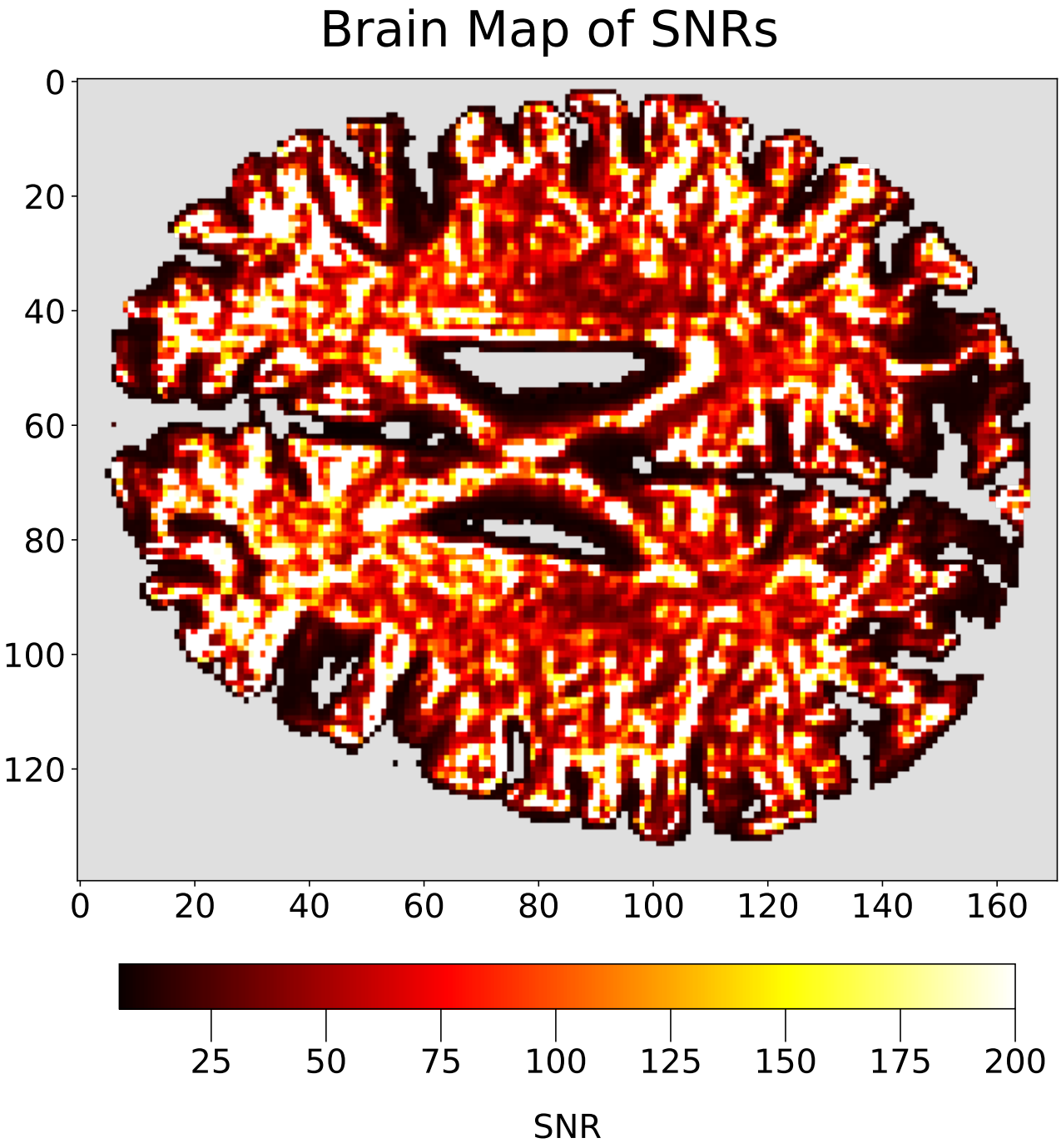


Figure 4.11: The SNR for each voxel was calculated by dividing the signal amplitude at the first echo time by the signal floor (offset value) as estimated with NLLS: $\frac{SA_{TE_0}}{offset/\sqrt{\pi/2}}$. Excluded voxels are colored grey.

Chapter 5: Emergence of the singular value decomposition in explaining neural networks

5.1 Introduction

Explaining or interpreting the mapping between inputs and outputs of trained neural networks (NNs) remains one of the major unsolved problems in machine learning and is the focus of significant research effort [YCN+15, SVZ14]. Recently these efforts have focused on associating individual components of NNs (such as layers or groups of neurons) with mathematical transformations (such as the Fourier transform [NCL+23] or wavelet shrinkage [KGSM23]). Such achievements in explaining NN weights have relied on sophisticated strategies for processing the given NN in conjunction with its training data. Moreover, these explanations have required considerable experimentation and as such, the existence of a general method for identifying the deterministic transformations within the hidden layers of NNs is still an elusive goal in machine learning. An intriguing solution towards this goal was recently proposed in [AKCK21] which introduced *descrambling transformations*, the subject of this chapter.

5.1.1 Neural Network Descrambling

NN descrambling is a mathematical model for explaining trained neural networks. Specifically, the goal of NN descrambling is to create human/expert readable visualizations of the trained weights $\mathbf{W}_k$. To this end, let $f : \mathbb{R}^{n_0} \to \mathbb{R}^{n_L}$ be a *bias-free* L-layer neural network given by the alternating

compositions of linearities and nonlinearities:

$$f(x) := \mathbf{W}_L \phi \cdots \phi \mathbf{W}_1 x. \tag{5.1}$$

It is assumed that the set of NN weights $\{\mathbf{W}_i\}_{i=1}^L$ is obtained after optimizing some loss function on the input-output data $\mathbf{X} \in \mathbb{R}^{n_0 \times n}$ and $\mathbf{Y} \in \mathbb{R}^{n_L \times n}$. Here each column of $\mathbf{X}$ (resp. $\mathbf{Y}$) represents an input (resp. output) point. The weights $\{\mathbf{W}_i\}_{i=1}^L$ are considered uninterpretable or *scrambled*. In NN descrambling, each weight matrix $\mathbf{W}_k$ is brought into a readable or *descrambled* form $\widehat{\mathbf{P}}\mathbf{W}_k$, through multiplication by a matrix $\widehat{\mathbf{P}}$–the *descrambler*. The descrambler is constructed as follows: in the chain of compositions (5.1) we insert the identity matrix factorized as $\mathbf{I} = \mathbf{P}^{-1}\mathbf{P}$ in the kth layer so that the network's output $\mathbf{F}$ remains unchanged after propagating $\mathbf{X}$ column-wise:

$$\mathbf{F} = \mathbf{W}_L \cdots \phi \mathbf{P}^{-1} \mathbf{P} \mathbf{W}_k \cdots \mathbf{W}_1 \mathbf{X} = \mathbf{W}_L \cdots \phi \mathbf{P}^{-1} \mathbf{P} f_k(\mathbf{X}), \tag{5.2}$$

where $f_k(\mathbf{X}) = \mathbf{W}_k \phi \cdots \phi \mathbf{W}_1 \mathbf{X} \in \mathbb{R}^{n_k \times n}$ is the matrix of *preactivations* given by the intermediate data in the network at the kth layer. The matrix $\mathbf{P}$ thus acts on the preactivations; in NN descrambling we choose a $\mathbf{P}$ which transforms the preactivation data optimally for minimizing a well-chosen explainability loss function η over a specified matrix group G:

$$\widehat{\mathbf{P}} := \widehat{\mathbf{P}}\,(\eta, G, k, \mathbf{X}, n) = \underset{\mathbf{P} \in G}{\operatorname{argmin}}\, \eta\,(\mathbf{P} f_k(\mathbf{X})). \tag{5.3}$$

An important choice for η proposed in [AKCK21] was the smoothness descrambling loss (referred to as *smoothness criterion (SC)* in [AKCK21]), where the preactivations $f_k(\mathbf{X})$ are rearranged to be

optimally smooth:

$$\eta_{SC}\left(\mathbf{P}f_k(\mathbf{X})\right) = \|\mathbf{D}\mathbf{P}f_k(\mathbf{X})\|_F^2. \tag{5.4}$$

Here $\mathbf{D}$ is taken to be either a spectral or finite-difference second order differentiation matrix [Tre00], so minimizing η_{SC} corresponds to choosing the orthogonal change of basis such that the discrete second derivative of the signal $\mathbf{P}f_k(\mathbf{X})$ is minimized. The motivation behind this choice of NN interpretation is that while in signal/audio processing contexts, the incoming and outgoing data is usually smooth and has an intelligible time-ordered structure, the intermediate data often loses this structure. As such, the role of the descrambler is to change basis so that the intermediate signal is itself smooth and ordered with respect to the indices of the output or input dimension. Using the formulation (5.3) the descramblers $\widehat{\mathbf{P}}_{SC}$ are defined as the minimizers of η_{SC} over the orthogonal matrices $O(n_k)$, where n_k is the number of neurons in the kth layer:

$$\widehat{\mathbf{P}}_{SC} := \widehat{\mathbf{P}}_{SC}(k, \mathbf{X}, n) := \widehat{\mathbf{P}}(\eta_{SC}, O(n_k), k, \mathbf{X}, n) = \underset{\mathbf{P}^\top\mathbf{P}=\mathbf{I}}{\operatorname{argmin}} \|\mathbf{D}\mathbf{P}f_k(\mathbf{X})\|_F^2. \tag{5.5}$$

The optimization problem (5.5) is an example of a quadratically constrained quadratic program (QCQP). We will demonstrate that (5.5) admits several global minima. Given that we identify $\widehat{\mathbf{P}}_{SC}(k, \mathbf{X}, n)$ with *explanations* of the weights $\mathbf{W}_k$, the non-uniqueness of the minimizers means that there may be *multiple* explanations of the kth layer weights. To select a particular explanation, the authors in [AKCK21] suggested a gradient-based optimization technique based on the Cayley transform such that the initial guess starts from "no explanation", $\mathbf{P} = \mathbf{I}$, and then iteratively minimizes the explainability loss.

Smoothness descrambling was used to explain the two-layer network *DEERNet* [WSJK18] trained to solve an integral equation arising in double electron-electron resonance (DEER). After descrambling, the first layer of the network was deemed to represent a cubic time-distance conversion in the DEER model and the second layer was judged to represent projection in the basis of Chebyshev polynomials (see Figs. 5.4 & 5.5 or [AKCK21, Figures 2,4]). In an ablation study, these layers were replaced by the identified mathematical transformations such as notch filtering, leading to a deterministic method for processing DEER data and removing the need for an NN entirely.

5.1.2 Goals of this chapter and summary of results

We aim to give a theoretical account for how descramblers explain the weights $\mathbf{W}_k$. Specifically, our goal is not to reverse engineer networks but to characterize the $\widehat{\mathbf{P}}_{SC}(k, \mathbf{X}, n)$ which enable the aforementioned reverse-engineering. Moreover, we will suggest an alternative method to find the minimizers of (5.5) based on an eigendecomposition and show that it may provide clearer explanations of the network weights. We also restrict ourselves to cases where we can take the limit of the descramblers as $n \to \infty$. Our results can be summarized as follows: **1.** For NN descrambling using η_{SC} with a general $n_k \times n_k$ matrix $\mathbf{D}$, there exists a sequence of descramblers $\widehat{\mathbf{P}}_{SC}(k, \mathbf{X}, n)$ that converges almost surely to $\mathbf{T}\mathbf{U}^\top$ as the number of points $n \to \infty$, where $\mathbf{T}$ are the right singular vectors of $\mathbf{D}$ arranged in ascending order of the singular values and $\mathbf{U}$ is the matrix of the principal components of the autocorrelation matrix of the preactivations $f_k(x)$, i.e $\mathbf{U}\mathbf{\Lambda}\mathbf{U}^\top = \mathbb{E}_x[f_k(x)f_k(x)^\top]$, where the columns of $\mathbf{U}$ are arranged in *descending* order of the singular values $\mathbf{\Lambda}$ (Theorem 5.2.4). This reveals that smoothness descrambling explains network weights by projecting the weight matrix column basis onto the principal components of the hidden data, followed by expansion in the basis

that diagonalizes the operator $\mathbf{D}$. In particular, when $\mathbf{D}$ is differentiation stencil (as in the case of smoothness descrambling), we will indicate that $\mathbf{T}$ forms a trigonometric basis, and this is what leads to highly interpretable representations of the weights as Fourier-domain filters (Figure 5.1).

2. For smoothness descrambling where $\mathbf{D}$ is a second order differentiation stencil, we characterize descramblers for cases where (1) data $\mathbf{X}$ is described by isotropic distributions or noisy measurements of signals, (2) NN is trained under the Saxe-Mclelland-Ganguli (SMG) model [SMG14] for a linear signal estimation problem and (3) the network is convolutional. Although these represent highly idealized situations, we note that they nonetheless occur quite commonly in deep learning workflows.

3. We revisit the empirical results from [AKCK21], showing that we can recover the notch filter in the first layer of DEERNet and an orthogonal polynomial basis in the second layer via an eigendecomposition (instead of solving the problem (5.5) via an optimizer). Moreover, we illustrate that although there can be *multiple* ways/descramblers to explain the NN corresponding to multiple minimizers of the problem (5.5), the explanations provided by them are nearly the same. The code to generate data and figures has been published at `https://github.com/ShashankSule/ESVD`.

4. We apply the singular value decomposition of first layer weights as an explainability tool to understand input layer regularization (ILR) for biexponential parameter recovery. Specifically, we show that the dominant right singular vectors of the first layer weights are themselves biexponential curves, revealing that both (ND, ND) and (ND, REG) networks learn an input signal library consistent with their training data. Crucially, (ND, REG) networks learn *markedly different* biexponential shapes in the noisy and smooth halves of each singular vector, whereas (ND, ND) networks learn nearly identical shapes in both halves—demonstrating that ILR induces a richer first-layer function class (Figure 5.7). We further exploit this post-hoc explanation to improve network training by pre-factoring the first layer weights in SVD form and fixing or initializing the right singular vectors to signal-model-

informed bases (`deconv` and `QR`). We find that networks with *fixed* right singular vectors trained under a soft orthogonality constraint achieve at least 40% lower test error than the benchmark (Table 5.2).

5. We apply singular vector explainability to neural STFT phase retrieval networks trained to recover human speech from undersampled spectrograms [CJSW24]. Inspecting the first layer weights reveals a highly structured banded organization in the singular matrices (i.e., singular vectors reshaped according to the time-frequency layout of the STFT input). Computing the SVD of these singular matrices separates time and frequency contributions, and shows that in the severely undersampled regime $m = d/2$, the time-direction singular vectors are smooth and non-oscillating—consistent with the network performing smooth interpolation of missing time-frequency measurements to reconstruct an invertible spectrogram (Figures 5.8, 5.9).

5.1.3 Related work

Explainability techniques generally fall into two categories [MSM18]: ante-hoc methods, which are inherently interpretable models like linear models [MSM18], decision trees [Qui86], offering transparency but often less predictive power; and post-hoc methods, applied after training to explain predictions. Post-hoc approaches are further divided into local and global explanations.

Local post-hoc methods aim to explain model predictions on or around a specific input point. Prominent examples include LIME (Local Interpretable Model-agnostic Explanations) [RSG16] and SHAP (SHapley Additive exPlanations) [LL17]. LIME approximates a black-box model's behavior around a specific input with a simpler, interpretable surrogate, revealing feature contributions for that prediction. SHAP, gives a game-theoretic attribution to how much a model's output at a given query

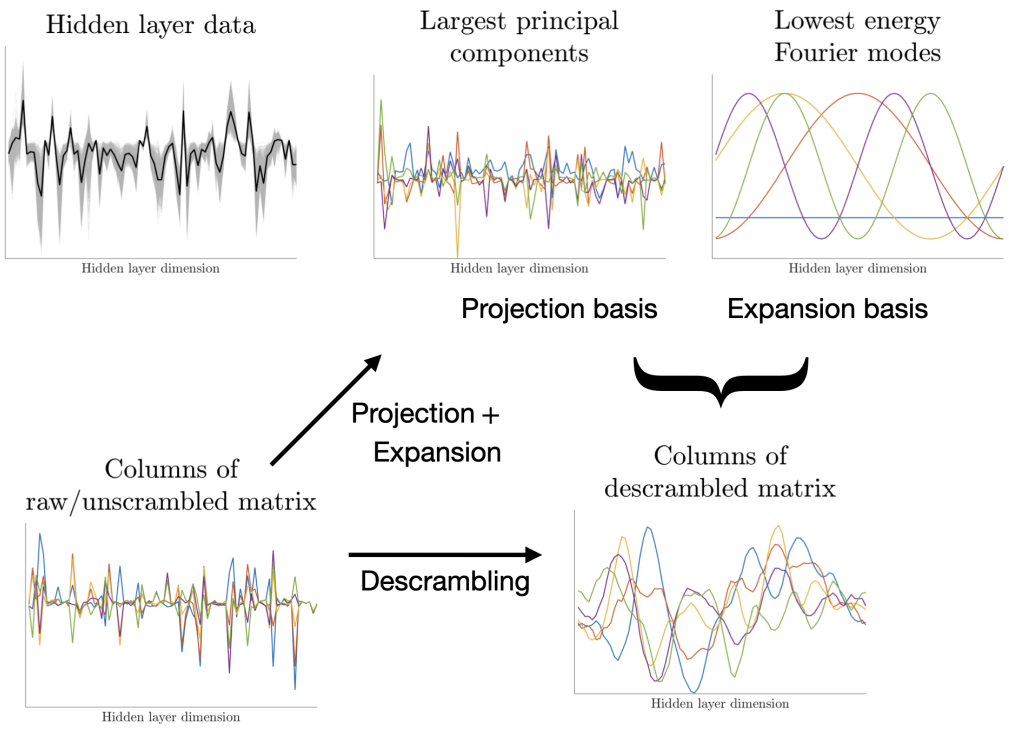


Figure 5.1: How neural network descrambling works: Theorem 5.2.4 shows that the explanation brings the column basis of network weights into interpretable form, by expressing the columns in an interpretable basis, with coefficients given by principal components projections.

point contributes to its deviation from the average of the model output on some data. While effective for single predictions or when regression variables have an underlying meaning, LIME and SHAP can struggle with data like time-series. For instance, SHAP can attribute how the nth measurement in time contributes to the model output, but this would offer limited explainability to an interested practitioner. While ongoing explainability research is aimed at surmounting such obstacles, at present there is no canonical method for adapting local frameworks like SHAP or LIME for settings like scientific machine learning where the data itself has a complicated structure.

In contrast, global explanation methods like NN descrambling explain the abstract transformation embodied by the neural network's weight matrix. It utilizes the entire training dataset to compute an explanation that remains independent of any specific input instance. Rather than competing with local methods, global methods complement local explanations by explicating the model's overall learned behavior. The global approach is also appealing for scientific machine learning: black-box models have achieved remarkable success in solving complex scientific and mathematical problems, from AlphaFold's protein structure predictions [JEP+21] to Physics-Informed Neural Networks (PINNs) solving differential equations [RPK19], and even discovering new theorems in knot theory [DVB+21]. Understanding how these models arrive at their solutions, by attributing known transformations to their individual components or *circuits* can reveal new insights into elusive physical or mathematical phenomena.

5.2 Results

The descramblers (5.5) for the kth layer and data $\mathbf{X}$ with n i.i.d. columns are denoted by $\widehat{\mathbf{P}}_{SC}(k, \mathbf{X}, n)$. We will show that these matrices admit almost sure limits as $n \to \infty$ for a variety

of choices of distributions for $\mathbf{X}$ and layer number k. To characterize such minimizers, first we state and prove a general result which will relate the smoothness objective function (5.4) to the Brockett cost function widely applied in graph matching, complexity theory, and optimization on matrix manifolds [Bro89, AMS08]:

Lemma 5.2.1. *Fix $\mathbf{S} \in \mathbb{R}^{d\times n}$ and $\mathbf{A} \in \mathbb{R}^{d\times d}$ such that $\mathbf{A}^\top = \mathbf{A}$. Define $\widehat{\mathbf{P}}$ to be the solution to the following minimization problem over the orthogonal group $O(d)$:*

$$\widehat{\mathbf{P}} \in \underset{\mathbf{P}\in O(d)}{\textit{argmin}}\ \mathit{Tr}(\mathbf{P}^\top \mathbf{APSS}^\top). \tag{5.6}$$

Let $\mathbf{A} = \mathbf{T}\boldsymbol{\Omega}\mathbf{T}^\top$ be an eigendecomposition of $\mathbf{A}$ sorted in ascending order of the eigenvalues where $\boldsymbol{\Omega}$ is the diagonal matrix of eigenvalues of $\mathbf{A}$ given by $\{\omega_i\}_{i=1}^d$. Then $\widehat{\mathbf{P}}$ is a minimizer of (5.6) if there exists a singular value decomposition of $\mathbf{S} = \mathbf{U}_S\boldsymbol{\Lambda}_S\mathbf{V}_S^\top$ such that $\widehat{\mathbf{P}} = \mathbf{T}\mathbf{U}_S^\top$.

Remark 5.2.2. *Note that if we set $\mathbf{S} = f_k(\mathbf{X}) \subseteq \mathbb{R}^{n_k\times n}$ where $f_k(\mathbf{X})$ are the kth layer preactivations then the smoothness descrambling functional $\eta_{SC}(\mathbf{P}f_k(\mathbf{X}))$ is given by*

$$\eta_{SC}(\mathbf{P}f_k(\mathbf{X})) := \|\mathbf{DPS}\|_F^2 = \mathit{Tr}(\mathbf{S}^\top\mathbf{P}^\top\mathbf{D}^\top\mathbf{DPS}) = \mathit{Tr}(\mathbf{P}^\top\mathbf{D}^\top\mathbf{DPSS}^\top). \tag{5.7}$$

This is exactly the objective function in (5.6) if we set $\mathbf{A} = \mathbf{D}^\top\mathbf{D}$. In general for $\mathbf{A}, \mathbf{B}$ symmetric, the function $\mathit{Tr}(\mathbf{P}^\top\mathbf{APB})$ is the Brockett cost function. Remarkably, NN descrambling connects the Brockett function and NN explainability. Lemma 5.2.1 has also been proved in more general settings using Lagrange multipliers and interlacing inequalities. For completeness, below we present another proof characterizing the minimizers on the orthogonal manifold by relaxation to a generalized eigenvalue problem.

Proof. First, using the cyclic property of traces, we get

$$F(\mathbf{P}) := \mathsf{T}r(\mathbf{P}^\top \mathbf{A}\mathbf{P}\mathbf{S}\mathbf{S}^\top) = \mathsf{T}r(\mathbf{S}^\top \mathbf{P}^\top \mathbf{A}\mathbf{P}\mathbf{S}). \tag{5.8}$$

Let $\mathbf{S} = \mathbf{U}_S \mathbf{\Lambda}_S \mathbf{V}_S^\top$ be an SVD of $\mathbf{S}$. Then, since $\mathbf{V}_S \mathbf{V}_S^\top = \mathbf{I}$, we have

$$\mathsf{T}r(\mathbf{S}^\top \mathbf{P}^\top \mathbf{A}\mathbf{P}\mathbf{S}) = \mathsf{T}r(\mathbf{S}^\top \mathbf{P}^\top \mathbf{A}\mathbf{P}\mathbf{S}\mathbf{V}_S \mathbf{V}_S^\top) = \mathsf{T}r(\mathbf{V}_S^\top \mathbf{S}^\top \mathbf{P}^\top \mathbf{A}\mathbf{P}\mathbf{S}\mathbf{V}_S). \tag{5.9}$$

Now consider the change of variable $\mathbf{Y} = \mathbf{P}\mathbf{S}\mathbf{V}_S$. Then we can reformulate $F(\mathbf{P})$ in terms of $\mathbf{Y}$ as follows:

$$F(\mathbf{P}) = \mathsf{T}r(\mathbf{Y}^\top \mathbf{A}\mathbf{Y}) =: G(\mathbf{Y}), \tag{5.10}$$

where $\mathbf{Y} \in \mathbb{R}^{d \times n}$ and $\mathbf{Y}^\top \mathbf{Y} = \mathbf{\Lambda}_S^\top \mathbf{\Lambda}_S \in \mathbb{R}^{n \times n}$. Thus, the minimum in problem (5.6) can be bounded below by

$$\min_{\mathbf{P} \in O(d)} F(\mathbf{P}) \geq \min_{\mathbf{Y}^\top \mathbf{Y} = \mathbf{\Lambda}_S^\top \mathbf{\Lambda}_S} \mathsf{T}r(\mathbf{Y}^\top \mathbf{A}\mathbf{Y}). \tag{5.11}$$

Thus we have relaxed (5.6) to a generalized eigenvalue problem. To attain the minimum on the RHS of (5.11), we proceed as follows: the first $r := \mathsf{rank}(\mathbf{\Lambda}_S)$ columns of the minimizers of $G(\mathbf{Y})$ are given by the first r eigenvectors of $\mathbf{A}$ renormalized by λ_i such that $\|y_i\|_2^2 = \lambda_i^2$. Thus the first r columns of $\mathbf{Y}$ are given by $\lambda_i t_i$ where t_i are the eigenvectors corresponding to the smallest r eigenvalues of $\mathbf{A}$ (including multiplicities and negative eigenvalues). The rest of the $n - r$ columns of $\mathbf{Y}$ are given by

$y_i = 0$. Thus, overall the solution may be written as $\widehat{\mathbf{Y}} = \mathbf{T}\mathbf{\Lambda}_S$. Plugging this into G we get

$$G(\widehat{\mathbf{Y}}) = \sum_{i=1}^{r} \lambda_i^2 \omega_i^2. \tag{5.12}$$

Thus, for a fixed ordering of the non-zero singular values of $\mathbf{S}$ the minimum of G is attained by such $\widehat{\mathbf{Y}}$. To minimize the RHS of (5.12) over all possible arrangements of the non-zero singular values of $\mathbf{S}$, we note the following: since ω_i necessarily range from smallest to largest, to minimize (5.12) we must choose λ_i^2 to range from largest to smallest due to the rearrangement inequality. Finally, we consider the original problem. Let $\widehat{\mathbf{P}}$ be such that $\widehat{\mathbf{Y}} = \mathbf{T}\mathbf{\Lambda}_S = \widehat{\mathbf{P}}\mathbf{S}\mathbf{V}_S$. Then we may write $\mathbf{S} = \widehat{\mathbf{P}}^\top \mathbf{T}\mathbf{\Lambda}_S\mathbf{V}_S$. This is a singular value decomposition for $\mathbf{S}$, so there exists some $\mathbf{U}_S \in O(d)$ such that $\widehat{\mathbf{P}}^\top\mathbf{T} = \mathbf{U}_S$ and $\mathbf{S} = \mathbf{U}_S\mathbf{\Lambda}_S\mathbf{V}_S^\top$. Therefore $\widehat{\mathbf{P}} = \mathbf{T}\mathbf{U}_S^\top$, where the columns of $\mathbf{T}$ (resp. $\mathbf{U}_S$) are arranged in the ascending (resp. descending) order of the singular values. □

Remark 5.2.3. *Note that the solutions to* (5.12) *are not unique and depend (at least) on the selection of the singular vectors of* $\mathbf{A}$ *and* $\mathbf{S}$.

5.2.1 Neural network descrambling: general results

Using Lemma 5.2.1 the large data limits of the matrices $\widehat{\mathbf{P}}_{SC}$ in (5.5) are characterized in a similar manner to the consistency of principal component analysis:

Theorem 5.2.4. *Let* $\mathbf{\Sigma} = \mathbb{E}_x[f_k(x)f_k(x)^\top]$ *be the autocorrelation matrix of* $f_k(x)$ *and assume* $\mathbf{\Sigma}$ *has distinct eigenvalues. Moreover, let* $\mathbf{U}\mathbf{\Lambda}\mathbf{U}^\top = \mathbf{\Sigma}$ *be an eigendecomposition of* $\mathbf{\Sigma}$. *There exists a sequence of minimizers* $\widehat{\mathbf{P}}_{SC}(k, \mathbf{X}, n)$ *to* (5.5) *such that as* $n \to \infty$, $\widehat{\mathbf{P}}_{SC}(k, \mathbf{X}, n) \to \mathbf{T}\mathbf{U}^\top$ *almost surely in the measure induced by the data* x.

Proof. Since a factor of n^{-1} does not change the minimizer in (5.5), we compute:

$$n^{-1}\|\mathbf{DP}f_k(\mathbf{X})\|_F^2 = n^{-1}\sum_{i=1}^{n}\|\mathbf{DP}f_k(x_i)\|_2^2 \tag{5.13}$$

$$= \mathsf{T}r\left(\mathbf{P}^\top\mathbf{D}^\top\mathbf{DP}\left[n^{-1}\sum_{i=1}^{n} f_k(x_i)f_k(x_i)^\top\right]\right) \tag{5.14}$$

$$=: \mathsf{T}r\left(\mathbf{P}^\top\mathbf{D}^\top\mathbf{DP}\boldsymbol{\Sigma}_n\right). \tag{5.15}$$

Here $\boldsymbol{\Sigma}_n$ is the empirical autocorrelation matrix given by

$$\boldsymbol{\Sigma}_n := n^{-1}\sum_{i=1}^{n} f_k(x_i)f_k(x_i)^\top. \tag{5.16}$$

Note that $\boldsymbol{\Sigma}_n \to \boldsymbol{\Sigma} := \mathbb{E}_x[f_k(x)f_k(x)^\top]$ almost surely as $n \to \infty$ due to the law of large numbers. Next we define $\delta = \inf\{\lambda_{i+1} - \lambda_i\}$ where λ_i are the eigenvalues of $\boldsymbol{\Sigma}$. Since by assumption these are distinct, $\delta > 0$. Furthermore, since the eigenvalues of $\boldsymbol{\Sigma}_n$ converge to those of $\boldsymbol{\Sigma}$ (see, for instance [Lax07, Theorem 6, Chapter 9]) the eigenvalues of $\boldsymbol{\Sigma}_n$ are also distinct for large enough n. Assuming such an n large enough, the eigenspaces of $\boldsymbol{\Sigma}_n$ are all distinct. Then by the modified Davis-Kahan theorem [YWS14, Theorem 2], for each index j, there exists an eigenvector of u_n^j of $\boldsymbol{\Sigma}_n$ corresponding to the jth eigenspace such that

$$\|u_n^j - u^j\|_2^2 \leq \frac{8\|\boldsymbol{\Sigma}_n - \boldsymbol{\Sigma}\|_F^2}{\delta^2}. \tag{5.17}$$

Here u^j is the jth eigenvector of $\boldsymbol{\Sigma}$. Now iterating the above bound over all the n_k eigenvectors u_n^j's

we get that there exist eigenvectors of $\mathbf{U}_n$ of $\mathbf{\Sigma}_n$ such that

$$\|\mathbf{U}_n - \mathbf{U}\|_F \leq \frac{2^{3/2}\sqrt{n_k}\|\mathbf{\Sigma} - \mathbf{\Sigma}_n\|_F}{\delta}. \tag{5.18}$$

Now, we may write $\widehat{\mathbf{P}}_{SC}(k, \mathbf{X}, n) = \mathbf{T}\mathbf{U}_n^\top$ and from Lemma 5.2.1 verify that $\widehat{\mathbf{P}}_{SC}(k, \mathbf{X}, n)$ minimizes the objective function (5.15) and is therefore a descrambler as per (5.3). As a consequence we have that $\widehat{\mathbf{P}}_{SC}(k, \mathbf{X}, n) \to \mathbf{T}\mathbf{U}^\top$ almost surely because $\mathbf{U}_n \to \mathbf{U}$ almost surely. □

Remark 5.2.5. *While the statement of Theorem 5.2.4 only proves convergence in the Frobenius norm of a particular selection of descramblers, a stronger statement is true if we considered the alignment between subspaces instead. In particular, under the hood of (5.17) lies the conclusion of the Davis-Kahan theorem which gives a bound for the alignment between singular subspaces:*

$$|\sin(u_n^j, u^j)|^2 \leq \frac{8\|\mathbf{\Sigma}_n - \mathbf{\Sigma}\|_F^2}{\delta^2}. \tag{5.19}$$

The above is true for any *singular vector of $\mathbf{\Sigma}_n$. Since the descramblers are given by $\widehat{\mathbf{P}}_{SC}(k, \mathbf{X}, n) = \mathbf{T}\mathbf{U}_n^\top$, we conclude that the one-dimensional* subspaces *spanned by the columns of* any descrambler *align with those spanned by the columns of $\mathbf{T}\mathbf{U}^\top$. Subspace alignment has been previously explored as an ingredient in the design of interpretable models [BFS22] and it is interesting to observe that it naturally emerges under NN descrambling.*

5.2.2 Variations in input distribution and network architecture

We apply Theorem 5.2.4 to a variety of conditions for f_k and distributions of the input-output pairs (x, y). We additionally define the hidden layer function $h_k(x)$ by

$$h_k(x) = \phi(f_{k-1}(x)). \tag{5.20}$$

Here ϕ is the nonlinear activation function. To obtain the hidden layer data $h_k(\mathbf{X})$ we apply the function h_k column-wise to $\mathbf{X}$. Note that $f_k(\mathbf{X}) = \mathbf{W}_k h_k(\mathbf{X})$. Then since the columns of $h_k(\mathbf{X})$ are i.i.d. samples of $h_k(x)$, the distribution of $h_k(x)$ can be prescribed and the limiting behaviour of the descramblers $\widehat{\mathbf{P}}_{SC}(k, \mathbf{X}, n)$ can be studied. We start with the case where hidden data the $h_k(x)$ has a whitened distribution with autocorrelation matrix $\mathbf{I}$. Whitened hidden data is a common assumption in many theoretical analyses of NNs [BDFS22, GMMM19]. To apply Theorem 5.2.4 we must understand the eigendecomposition of $\mathbf{D}^\top \mathbf{D}$ where $\mathbf{D}$ is a second-order finite-difference/Fourier differentiation matrix of dimension n_k. In both cases, $\mathbf{D}$ is a real and circulant matrix so it will admit singular vectors $\mathbf{T}$ given by the Hartley transform [KSU03]. For example, if $\mathbf{D}$ is a periodic finite-difference Laplacian then the columns of $\mathbf{T}$ sorted in ascending order of eigenvalues can be characterized as follows:

$$T_m(l) = \begin{cases} \frac{1}{\sqrt{n_k}} \left(\cos \frac{\pi m l}{n_k} + \sin \frac{\pi m l}{n_k} \right), & m \text{ even}, \\ \frac{1}{\sqrt{n_k}} \left(\cos \frac{\pi (m+1) l}{n_k} + \sin \frac{\pi (m+1) l}{n_k} \right), & m \text{ odd}. \end{cases} \tag{5.21}$$

The following result shows that in the large data limit, the action of the descrambled weight matrix is given by basis projection followed by expansion in the trigonometric basis given by (5.21):

Theorem 5.2.6 (Isotropic data)**.** *Let f_k be given by (5.2). If $n_{k-1} \geq n_k$ and the singular values of $\mathbf{W}_k$*

are distinct, then there exist descramblers $\widehat{\mathbf{P}}_{SC}(k, \mathbf{X}, n)$ *such that the descrambled weight matrices* $\widehat{\mathbf{P}}_{SC}(k, \mathbf{X}, n)\mathbf{W}_k \to \widehat{\mathcal{W}}_k$ *as* $n \to \infty$ *where the action of* $\widehat{\mathcal{W}}_k$ *is given by:*

$$\widehat{\mathcal{W}}_k x = \sum_{i=1}^{n_k} T_i \lambda_i^k \langle v_i^k, x \rangle, \tag{5.22}$$

where v_i^k *is the right singular vector of* $\mathbf{W}_k$ *and* λ_i^k *is the corresponding singular value.*

Proof. The sample autocorrelation matrix $\mathbf{\Sigma}_n$ of the data $f_k(\mathbf{X})$ is given by

$$\mathbf{\Sigma}_n := \mathbf{W}_k \left[\frac{1}{n} \sum_{i=1}^{n} (h_k(x))_i (h_k(x))_i^\top \right] \mathbf{W}_k^\top. \tag{5.23}$$

Now, since $h_k(x_i)$ are samples of a zero mean isotropic random variable, we get that $\mathbf{\Sigma}_n \to \mathbf{W}_k \mathbf{W}_k^\top$ almost surely. Since $\mathbf{W}_k$ has distinct singular values and its output dimension $n_k \leq n_{k-1}$ by assumption, we get that the eigenvalues of $\mathbf{W}_k \mathbf{W}_k^\top$ are distinct. Therefore, Theorem (5.2.4) applies and we have that there exists a sequence of descramblers $\widehat{\mathbf{P}}_{SC}(k, \mathbf{X}, n)$ such that $\widehat{\mathbf{P}}_{SC}(k, \mathbf{X}, n)\mathbf{W}_k \to \mathbf{T}\mathbf{U}^\top \mathbf{W}_k$ where $\mathbf{U}_k$ is the matrix of singular vectors of $\mathbf{W}_k$. Relabelling $\mathbf{T}\mathbf{U}^\top \mathbf{W}_k$ as $\widehat{\mathcal{W}}_k$ we have that $\widehat{\mathcal{W}}_k = \mathbf{T}\mathbf{U}_k^\top \mathbf{U}_k \mathbf{\Lambda}_k \mathbf{V}_k^\top = \mathbf{T}\mathbf{\Lambda}_k \mathbf{V}_k^\top$ where $\mathbf{W}_k = \mathbf{U}_k \mathbf{\Lambda}_k \mathbf{V}_k^\top$ is the SVD of $\mathbf{W}_k$. □

Remark 5.2.7. *Theorem 5.2.6 shows that NN descrambling mathematically captures the process of representing the directions of largest variance of the preactivation data through trigonometric functions of lowest energy.*

Remark 5.2.8. *Although the descramblers* $\widehat{\mathbf{P}}_{SC}(k, \mathbf{X}, n)$ *may be non-unique, this degeneracy may be circumvented in the large data limit in the sense that the action of the descrambled weights can be described by projection onto one-dimensional subspaces followed by expansion in a trigonometric basis. We must also assume the distinctness of the singular values of the raw/scrambled weights* $\mathbf{W}_k$.

In practice this is not a restriction since neural networks whose weights have distinct singular values are dense in the space of all networks.

Next we consider the case when the hidden layer vectors h_k are noisy measurements of some parameter (with a given prior) under isotropic noise with a prescribed SNR α_{SNR}. Specifically, let $z \in \mathbb{R}^d$ be a random variable with finite moments. Let $s : \mathbb{R}^d \to \mathbb{R}^{n_k}$ such that $\mathbb{E}_z[\|s(z)\|_2^2] < \infty$. We assume that distribution of hidden layer inputs is given by $h_k(x) \sim s(z) + \alpha_{\text{SNR}}^{-1}\varsigma$. This situation arises widely in inverse problems and signal processing, and indeed the original motivation for NN descrambling was to study a network trained to recover z from $x = h_0$. In this case, we show that as the data gets noisier, the descrambled weight matrices converge to the form given in (5.22).

Theorem 5.2.9 (Noisy signal input)**.** *Let f_k be given by (5.2) and assume $\mathbf{W}_k$ has distinct singular values where $n_k \leq n_{k-1}$. Let $h_k(x) \sim s(z) + \alpha_{\text{SNR}}^{-1}\varsigma$ where z and ς are independent distributions, ς is isotropic and zero mean. Then for small enough α_{SNR}, as $n \to \infty$ there exist descrambled weights $\widehat{\mathbf{P}}_{SC}(k, \mathbf{X}, n)\mathbf{W}_k \to \widehat{\mathcal{W}}_k$ where for every $x \in \mathbb{R}^{n_k}$,*

$$\widehat{\mathcal{W}}_k x = \sum_{i=1}^{n_k} T_i \lambda_i^k \langle v_i^k, x\rangle + O(\alpha_{\text{SNR}}^2). \tag{5.24}$$

where v_i^k is the right singular vector of $\mathbf{W}_k$ and λ_i^k is the corresponding singular value.

Proof. Note that multiplying the hidden data by α_{SNR} will not change the minimizers in (5.3). Due to the zero mean property and isotropy of ς we get that $\mathbb{E}[h_k h_k^\top] = \mathbb{E}[s(z)s(z)^\top] + \alpha_{\text{SNR}}^{-2}\mathbf{I}$. Therefore the autocorrelation matrix of the rescaled hidden data $\alpha_{\text{SNR}} f_k = \alpha_{\text{SNR}}\mathbf{W}_k h_k$ is $\mathbf{\Sigma}_{\alpha_{\text{SNR}}} = \alpha_{\text{SNR}}^2 \mathbf{W}_k \mathbb{E}[s(z)s(z)^\top]\mathbf{W}_k^\top + \mathbf{W}_k\mathbf{W}_k^\top$. For small enough α_{SNR}, $\mathbf{\Sigma}_{\alpha_{\text{SNR}}}$ will have distinct eigenvalues because it is a small perturbation of $\mathbf{W}_k\mathbf{W}_k^\top$. Therefore, Theorem 5.2.4 applies and we get that the descramblers $\widehat{\mathbf{P}}_{SC}(k, \mathbf{X}, n) \to$

$\mathbf{T}\mathbf{U}_{\alpha_{\text{SNR}}}^{\top}$ where $\mathbf{U}_{\alpha_{\text{SNR}}}$ are the eigenvectors of $\mathbf{\Sigma}_{\alpha_{\text{SNR}}}$. Since $\mathbf{\Sigma}_{\alpha_{\text{SNR}}} \to \mathbf{\Sigma}_0 = \mathbf{W}_k\mathbf{W}_k^{\top}$ as $\alpha_{\text{SNR}} \to 0$, denoting the singular vectors of $\mathbf{\Sigma}_0$ as $\mathbf{U}_0$ and using the Davis Kahan bound (5.18) on $\mathbf{\Sigma}_0$ and $\mathbf{\Sigma}_{\alpha_{\text{SNR}}}$ we conclude that there exist singular vectors $\mathbf{U}_{\alpha_{\text{SNR}}}$ of $\mathbf{\Sigma}_{\alpha_{\text{SNR}}}$ such that $\|\mathbf{U}_{\alpha_{\text{SNR}}} - \mathbf{U}_0\| = O(\alpha_{\text{SNR}}^2)$. But since $\mathbf{\Sigma}_0 = \mathbf{W}_k\mathbf{W}_k^{\top}$, $\mathbf{U}_0$ are the singular vectors of $\mathbf{W}_k$ and since the large data limit descrambled weights are given by $\mathbf{T}\mathbf{U}_{\alpha_{\text{SNR}}}^{\top}\mathbf{W}_k$, (5.24) holds readily. □

5.2.3 NN descrambling in the presence of training

Finally, we tackle a case where we unite NN descrambling with NN training. Here we make a number of simplifying assumptions:

(A1) The NN is a *two-layer linear network* with one hidden layer, i.e the NN is $f = \mathbf{W}_2\mathbf{W}_1 x$ and the f_k under consideration for descrambling are $f_1 = \mathbf{W}_1 x$ and $f_2 = f$. Such two-layer linear networks are examples of the highly-studied deep linear networks [SMG14, SMG19, BDFS22].

(A2) f is trained with finitely many examples $(x_i, y_i)_{i=1}^n$ with mean squared error loss:

$$\mathcal{L}_{\text{MSE}}(\mathbf{W}_1, \mathbf{W}_2) = \frac{1}{n}\sum_{i=1}^{n} \|y_i - \mathbf{W}_2\mathbf{W}_1 x_i\|_2^2. \tag{5.25}$$

The loss $\mathcal{L}_{\text{MSE}}$ is minimized under a *gradient flow* with learning rate $1/\tau$:

$$\tau\frac{d\mathbf{W}_1}{dt} = -\nabla_{\mathbf{W}_1}\mathcal{L}_{\text{MSE}},\ \tau\frac{d\mathbf{W}_2}{dt} = -\nabla_{\mathbf{W}_2}\mathcal{L}_{\text{MSE}}. \tag{5.26}$$

(A3) The eigenvectors of the input-input correlations $\mathbf{\Sigma}_n^{xx}$ coincide with the right singular vectors of

the input-output correlations $\mathbf{\Sigma}_n^{xy}$:

$$\mathbf{\Sigma}_n^{xx} = \frac{1}{n}\sum_{i=1}^{n} x_i x_i^\top, \quad \mathbf{\Sigma}_n^{xy} = \frac{1}{n}\sum_{i=1}^{n} y_i x_i^\top, \tag{5.27}$$

$$\mathbf{\Sigma}_n^{xx} = \mathbf{V}_n^x \mathbf{\Lambda}_n^x [\mathbf{V}_n^x]^\top, \quad \mathbf{\Sigma}_n^{xy} = \mathbf{U}_n^{xy} \mathbf{\Lambda}_n^{xy} [\mathbf{V}_n^x]^\top. \tag{5.28}$$

Note here that we denote matrices involved in self-correlations with the notation xx in the superscript and those with correlations between x and y with xy in the superscript. The assumption (A3) therefore states that the eigenvectors that diagonalize the auto correlation matrix $\mathbf{\Sigma}_n^{xx}$ also appear as the right singular vectors of the cross-correlation matrix $\mathbf{\Sigma}_n^{xy}$. This alignment is an important assumption for the convergence of neural training dynamics as was shown in [SMG14].

(A4) The weights $\mathbf{W}_1, \mathbf{W}_2$ are initialized with an initial condition

$$\mathbf{W}_1 = \mathbf{R}\mathbf{\Lambda}_1(0)[\mathbf{V}_n^x]^\top, \quad \mathbf{W}_2 = \mathbf{U}_n^{xy}\mathbf{\Lambda}_1(0)\mathbf{R}^\top. \tag{5.29}$$

Here $\mathbf{R}$ is an arbitrary orthogonal matrix. In [SMG19] and [SMG14] it is shown that the NN weights initialized using layerwise pretraining or random orthogonal initialization approximately satisfy the conditions (5.29) when trained under the gradient flow (5.26). Here we assume that the matrices $\mathbf{W}_1$ and $\mathbf{W}_2$ satisfy this form exactly to simplify our calculations.

The assumptions $(A1) - (A4)$ provide a setting in which the dynamics of two-layer linear networks can be described exactly.

Proposition 5.2.10 ([SMG14]). *Under the assumptions* $(A1 - A4)$ *the dynamics of* $\mathbf{W}_1(t)$ *and* $\mathbf{W}_2(t)$

evolve according to the following dynamics:

$$\mathbf{W}_1(t) = \mathbf{R}\mathbf{\Lambda}_1(t)[\mathbf{V}_n^x]^\top, \quad \mathbf{W}_2(t) = \mathbf{U}_n^{xy}\mathbf{\Lambda}_2(t)\mathbf{R}^\top. \tag{5.30}$$

Here $\mathbf{R}$ *is the arbitrary fixed orthogonal matrix used for initialization in* (5.29) *and* $\mathbf{\Lambda}_1, \mathbf{\Lambda}_2$ *are diagonal matrix-valued functions defined on* $[0, +\infty)$.

Using the training dynamics of the weights, we can now describe the dynamics of the *descrambled weights*:

Theorem 5.2.11. *Suppose* $(A1-A4)$ *are satisfied. Then for every* $t \in [0, +\infty)$ *there exist descrambled weight matrices given by:*

$$\widehat{\mathbf{P}}(1, \mathbf{X}, n)\mathbf{W}_1 = \mathbf{T}\mathbf{\Lambda}_1'[\mathbf{V}_n^x]^\top, \quad \widehat{\mathbf{P}}(2, \mathbf{X}, n)\mathbf{W}_2 = \mathbf{T}\mathbf{\Lambda}_2'\mathbf{R}^\top. \tag{5.31}$$

Here $\mathbf{\Lambda}_1', \mathbf{\Lambda}_2'$ *are diagonal matrix valued functions defined on* $[0, +\infty)$.

Proof. (Theorem 5.2.11) The NN weights $\mathbf{W}_2$ and $\mathbf{W}_1$ are described by (5.30). Denoting by $\mathbf{X}$ the $n_0 \times n$ matrix of i.i.d samples of x, we proceed layer by layer. For $k = 1$, let $\mathbf{\Sigma}_n$ be the autocorrelation matrix of $\mathbf{S} = \mathbf{W}_1\mathbf{X}$. Note that due to (5.30) and assumption (A3) (5.28),

$$\mathbf{\Sigma}_n = \mathbf{R}\mathbf{\Lambda}_1(t)\mathbf{\Lambda}_n^x\mathbf{R}^\top \tag{5.32}$$

We assign $\mathbf{\Lambda}_1'(t) := \mathbf{\Lambda}_1(t)\mathbf{\Lambda}_n^x$. To keep notation clean we suppress the dependence on t. From Theorem

5.2.4 we have that for fixed n the descramblers are given by $\mathbf{T}\mathbf{R}^\top$ and the descrambled weights are

$$\mathbf{T}\mathbf{R}^\top \mathbf{W}_1 = \mathbf{T}\mathbf{\Lambda}_1'[\mathbf{V}_n^x]^\top \tag{5.33}$$

For $k = 2$, the autocorrelation matrix of the finite samples is given by $\mathbf{U}_n^{xy}\mathbf{\Lambda}(t)[\mathbf{U}_n^{xy}]^\top$. Labeling $\mathbf{\Lambda}_2'(t) = \mathbf{\Lambda}_1(t)\mathbf{\Lambda}_n^{xy}$, by the same argument as for $k = 1$, the descramblers are given by $\mathbf{T}[\mathbf{U}_n^{xy}]$ and the descrambled weights by $\mathbf{T}\mathbf{\Lambda}_2'\mathbf{R}^\top$. □

In Theorem 5.2.11 if the sample input-input correlation matrix converges to a limit with distinct eigenvalues then Theorem 5.2.4 may be applied to obtain a limiting descrambled matrix for the first layer. Theorem 5.2.11 shows that under the SMG model, the descrambled first layer reflects the data (through the influence of $[\mathbf{V}_n^x]^\top$) while the descrambled second layer reflects the influence of the network *initialization* (through the influence of $\mathbf{R}$). The assumptions A1, A2, and A4 are regularly used settings for studying the training dynamics of neural networks. The assumption A3 on the training data is quite restrictive and is usually not satisfied exactly. However, it has been illustrated numerically [SMG19, BH89] that the training dynamics (5.30) are stable with respect to small deviations in A3. This means that the NN still approximately obeys (5.30) when A3 is approximately satisfied. A case of approximate satisfaction of A3 is given by noisy linear measurements of a signal under isotropic noise. Let the random variable $z \in \mathbb{R}^{n_2}$ model a signal to be estimated such that $\mathbb{E}[zz^\top] = \vartheta\mathbf{I}$ for some $\vartheta \in \mathbb{R}$. Moreover, let $\mathbf{K} : \mathbb{R}^{n_2} \to \mathbb{R}^{n_0}$ be a linear transformation and ς a zero-mean isotropic random variable. Set the NN inputs to be i.i.d samples from $x = \mathbf{K}z + \varsigma$ and the outputs as $y = z$. Then the

training data (x, y) satisfies assumption $A3$:

$$\mathbf{\Sigma}^{xy} = \mathbb{E}[zz^\top]\mathbf{K}^\top = \vartheta\mathbf{K}^\top, \tag{5.34}$$

$$\mathbf{\Sigma}^{xx} = \mathbf{K}\mathbb{E}[zz^\top]\mathbf{K}^\top + \mathbf{I} = \vartheta\mathbf{K}\mathbf{K}^\top + \mathbf{I}. \tag{5.35}$$

$[\mathbf{\Sigma}^{xy}]^\top\mathbf{\Sigma}^{xy}$ and $\mathbf{\Sigma}^{xx}$ have the same eigenspaces. As a consequence, n i.i.d samples of (x, y) will approximately satisfy the condition A3. This observation suggests that descrambling NNs trained for noisy signal estimation (such as DEERNet) might reveal the training dynamics of the weights. We will provide a numerical illustration of this idea in Section 5.3.2.

5.2.4 Descrambling as rearrangement of neurons

The action of the descrambler $\widehat{\mathbf{P}}_{SC}(k, \mathbf{X}, n)$ on the weights $\mathbf{W}_k$ can be modeled as a symmetry that "rearranges" the neurons in the kth layer for optimal explainability. We describe a case where this rearrangement is a permutation. Notably, this occurs when using convolutional networks.

Proposition 5.2.12 (Convolutional NNs)**.** *Let f_k be given by (5.2) and assume the hidden data $h_k(\mathbf{X})$ is whitened. Moreover, assume $\mathbf{W}_k$ represents a convolution with stride 1 where $\mathbf{W}_k x = w * x$. Then $\widehat{\mathbf{P}}_{SC}(k, \mathbf{X}, n)\mathbf{W}_k = \mathbf{\Pi}^\top\mathbf{W}_k$ where $\mathbf{\Pi}^\top$ is a permutation.*

Proof. Let $\mathbf{\Sigma}_n^h$ be the autocorrelation matrix of the data $h_k(\mathbf{X})$. By assumption $\mathbf{\Sigma}_n^h = \mathbf{I}$ so the autocorrelation matrix of the preactivation data $f_k(\mathbf{X}) = \mathbf{W}_k h_k(\mathbf{X})$ is $\mathbf{\Sigma}_n = \mathbf{W}_k\mathbf{W}_k^\top$ (5.16). Since $\mathbf{W}_k$ represents a circular convolution $\mathbf{W}_k\mathbf{W}_k^\top$ must be diagonalized by the real Fourier basis given by $\mathbf{T}$ in (5.21) [KSU03]. To apply Theorem 5.2.4 we must rearrange the columns of $\mathbf{T}$ through a permutation matrix $\mathbf{\Pi}$ to match the descending order of the eigenvalues of $\mathbf{W}_k\mathbf{W}_k^\top$. Representing this rearranged

eigenbasis as $\mathbf{T\Pi}$ the descramblers are given as $\widehat{\mathbf{P}}_{SC}(k, \mathbf{X}, n) = \mathbf{T}(\mathbf{T\Pi})^\top = \mathbf{T\Pi}^\top\mathbf{T}^\top = \mathbf{\Pi}^\top$ since $\mathbf{TT}^\top = \mathbf{I}$. □

Remark 5.2.13. *The assumption of whitened finite samples of hidden data $h_k(\mathbf{X})$ is stronger than the assumption of whitened distributions of hidden data made in Theorem 5.2.6. However, there are several contexts in which hidden data is isotropic or made to be isotropic for enhancing the learning capabilities of fully connected NNs [Luo17, WN11, YEH+20].*

5.3 Numerical results

In this section we will descramble the network DEERNet given by the architecture $\mathsf{DEERNet}(x) = \mathsf{logsigmoid}(\mathbf{W}_2\mathsf{tanh}(\mathbf{W}_1 x))$ as described in [AKCK21] [1]. The NN was trained to solve the following integral equation arising in double electron-electron resonance (DEER):

$$\Gamma(t) = \int_\Omega p(r)\gamma(r,t)\,dr + \varsigma. \tag{5.36}$$

Here ς is considered to be white noise with a preset signal-to-noise ratio (SNR) and $\gamma(r,t)$ is the DEER kernel given by

$$\gamma(r,t) := \sqrt{\frac{\pi}{6At}}\left[\cos[At]C\left[\sqrt{\frac{6At}{\pi}}\right] + \sin[At]S\left[\sqrt{\frac{6At}{\pi}}\right]\right], \tag{5.37}$$

$$A := \frac{\mu_0}{4\pi}\frac{\gamma_1\gamma_2 h}{r^3};\ C(x) = \int_0^x \cos(t^2)\,dt;\ S(x) = \int_0^x \sin(t^2)\,dt. \tag{5.38}$$

DEERNet was trained to approximate the solution operator taking Γ to p using synthetically generated input-output pairs (Γ_i, p_i).

[1] The code to generate data and figures has been published at `https://github.com/ShashankSule/ESVD`

5.3.1 Non-uniqueness of descramblers

In order to apply descrambling to a specific NN we first have to address the issue of non-uniqueness of descramblers. Lemma 5.2.1 combined with (5.7) shows that the solutions to the NN descrambling problem are only unique up to the choices of orthogonal bases for the singular subspaces of $\mathbf{S}\mathbf{S}^\top$ (the data autocorrelation matrix) and $\mathbf{D}^\top\mathbf{D}$ (the differentiation stencil, or more generally, a context dependent linear operator). A scheme to select a particular minimizer to act as an explanation for the weights $\mathbf{W}_k$ was formulated in [AKCK21] where it was proposed that the smoothness objective function η_{SC} be minimized over $O(n_k)$ through the Cayley transform:

$$\mathbf{P}(\mathbf{Q}) = (\mathbf{I} - \mathbf{Q})(\mathbf{I} + \mathbf{Q})^{-1},\ \mathbf{Q} = -\mathbf{Q}^\top. \tag{5.39}$$

The Cayley transform is an injective parameterization of the Lie group $O(d)$ through its Lie Algebra $S_d^-(\mathbb{R})$, the vector space of skew-symmetric matrices on $\mathbb{R}$. Through the Cayley transform, the smoothness descrambling problem on the orthogonal group may be restated as a nonlinear optimization problem on the vector space $S_d^-(\mathbb{R})$:

$$\widehat{\mathbf{Q}}(k, \mathbf{X}, n) = \underset{\mathbf{Q}\in S_{n_k}^-(\mathbb{R})}{\operatorname{argmin}} \|\mathbf{D}(\mathbf{I} - \mathbf{Q})(\mathbf{I} + \mathbf{Q})^{-1} f_k(\mathbf{X})\|_F^2 := \underset{\mathbf{Q}\in S_{n_k}^-(\mathbb{R})}{\operatorname{argmin}} G(\mathbf{Q}). \tag{5.40}$$

The advantage of the formulation (5.40) is that now the descramblers $\widehat{\mathbf{P}}_{SC}(k, \mathbf{X}, n)$ may be found using a gradient flow:

$$\mathbf{Q}'(t) = \tau \nabla_{S_d^-} G(\mathbf{Q}(t)), \quad \mathbf{Q}(0) = \mathbf{Q}_0, \tag{5.41}$$

$$\widehat{\mathbf{Q}}(k, \mathbf{X}, n) = \widehat{\mathbf{Q}}_{\mathbf{Q}_0} := \lim_{t \to \infty} \mathbf{Q}(t). \tag{5.42}$$

Given that $G(\mathbf{Q})$ has multiple minimizers, the choice of the initial point $\mathbf{Q}_0$ is essential for determining the solution $\widehat{\mathbf{Q}}$ as long as the stepsize τ is sufficiently small. Alternately, following Theorem 5.2.4 the gradient-based minimization approach (5.41) may be discarded entirely in favour of using an eigensolver, computing $\mathbf{U}_S$ and then writing the descrambler as $\widehat{\mathbf{P}} = \mathbf{T}\mathbf{U}_S^\top$ for some fixed choice of $\mathbf{T}$. In Figure 5.2 we compare these approaches to finding the descramblers of the first layer of DEERNet by exploring three different initializations for the gradient flow (5.41) and directly using an eigensolver for descrambling. We find that although the descamblers are different in all four cases, the explanations of $\mathbf{W}_1$ they produce upon multiplication are nearly identical. In Figure 5.3 it can be observed that the time-domain representations of the descrambled matrices $\widehat{\mathbf{P}}_{SC}(1, \mathbf{X}, n)\mathbf{W}_1$ feature wave-like interlocking patterns. To understand these patterns we use the trick in [AKCK21] by inspecting the Fourier domain representations of these matrices as filters, revealing a notch filter at the base frequency. However, we find that the different methods of computing the descramblers lead to slightly different qualities of notch filter. In particular, the weight matrix descrambled through the eigensolver has a sharp frequency cutoff outside the $[-10, 10]$ band. Although the weight matrices descrambled through the three other methods also contain the same bandpass filtering feature, the frequency cutoff is certainly noisier and less pronounced than the eigensolver method (Table 5.1). As a consequence, directly computing the solutions to (5.3) via Theorem 5.2.4 may be more beneficial for explainabil-

ity than using the gradient-descent based method suggested in [AKCK21]. Moreover, substituting the gradient-based optimization approach with eigendecomposition can scale up NN descrambling for high-dimensional problems, through the use of randomized or sketching based SVDs [HMT09].

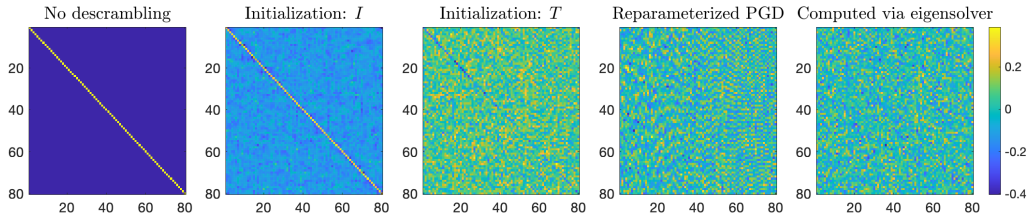


Figure 5.2: Left to Right: The descamblers $\widehat{\mathbf{P}}_{SC}(1, \mathbf{X}, n)$ were computed using four different strategies (a) Projected Gradient Descent (PGD) on (5.41) with $\mathbf{P}(\mathbf{Q}_0) = \mathbf{I}$, (b) Warm start, i.e PGD with $\mathbf{P}(\mathbf{Q}_0) = \mathbf{T}$, (c) Rescaled PGD where $\mathbf{D}^\top\mathbf{D} = \mathbf{T}\mathbf{\Omega}\mathbf{T}^\top$ was replaced with a diagonal matrix $\mathbf{\Omega}$ and solutions were interpreted through multiplication by $\mathbf{T}$, and (d) Direct eigendecomposition of $\mathbf{S} = f_1(\mathbf{X}) = \mathbf{W}_1\mathbf{X}$.

	No desc.	Init.: $\mathbf{I}$	Init.: $\mathbf{T}$	Reparam	Eigensolver
Rel. norm of coefficients where $\omega_1 \notin [-10, 10]$	7.86×10^{-1}	5.87×10^{-2}	$5.40\times10^{e-2}$	7.63×10^{-2}	$\mathbf{9.66}\times10^{-3}$

Table 5.1: Figure 5.3 displays a bandpass in the $\omega_1 \in [-10, 10]$ frequency band in the descrambled weights. We additionally find that in the descrambled weights, less than 5% of the norm is concentrated outside this band. Moreover, the weights descrambled via the eigensolver have the sharpest band cut off.

5.3.2 Assessing the validity of interpretation

To what extent is the notch filter in Figure 5.3 inherent to the network weights $\mathbf{W}_1$ and not an artefact of (a) the influence of the differentiation matrix $\mathbf{D}$, or (b) the training data $\mathbf{X}$? We explore this question by using Theorem 5.2.11. In particular, suppose DEERNet was a two-layer linear network

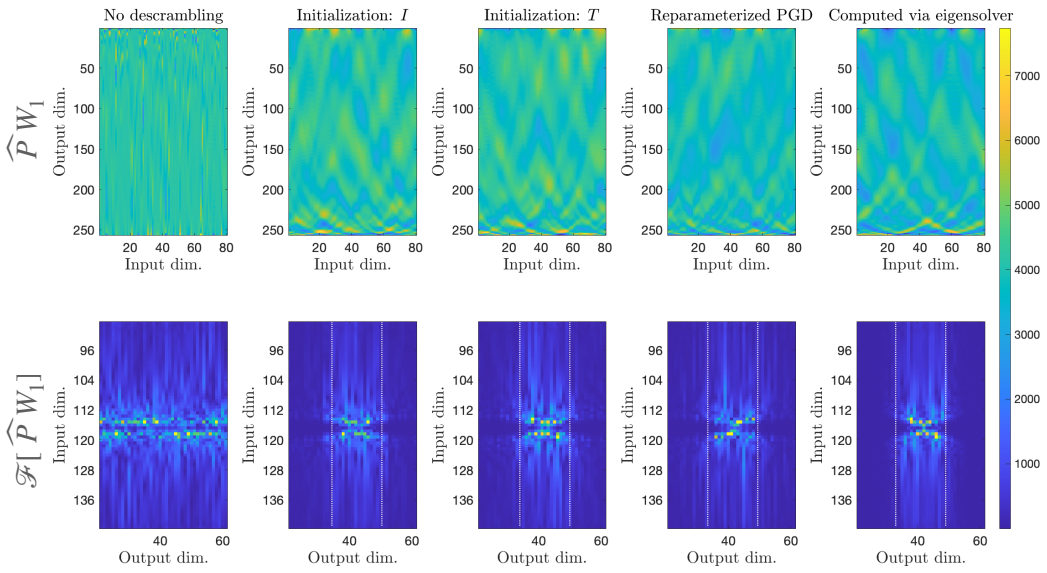


Figure 5.3: Top: Descrambled weight matrices $\widehat{\mathbf{P}}_{SC}(1, \mathbf{X}, n)$ computed for the four strategies outlined in Figure 5.2. Notably, while the descramblers are quite different, the descrambled weights themselves are quite similar. Bottom: This similarity among explanations can be quantified by moving to the Fourier domain where nearly all descrambled weights have a notch at the zero frequency and a bandpass filter along the output dimension in approximately the $\omega_1 \in [-10, 10]$ frequency band.

given by $f = \mathbf{W}_2\mathbf{W}_1 x$. Since it is trained on data of the form $x = \mathbf{K}z + \alpha_{\text{SNR}}^{-1}\varsigma$ where $\mathbf{K}$ is the discretization of an integral kernel, if $\mathbb{E}[zz^\top] = \mathbf{I}$ then the solutions to NN descrambling given by Theorem 5.2.11 would approximately hold. Under this approximation, computing the SVD of $\mathbf{W}_1$ as $\mathbf{W}_1 = \mathbf{U}_1\mathbf{\Lambda}_1\mathbf{V}_1$, we would have $\mathbf{V}_1 = \mathbf{V}_n^x$, where $\mathbf{V}_n^x$ are the eigenvectors of the input autocorrelation matrix $\mathbf{\Sigma}_n^{xx}$. As a result, the descrambled weight matrix would be described as:

$$\widehat{\mathbf{P}}_{SC}(1, \mathbf{X}, n)\mathbf{W}_1 \approx \mathbf{T}\mathbf{\Lambda}_1'[\mathbf{V}_n^x]^\top. \tag{5.43}$$

According to (5.35), $\mathbf{\Sigma}_n^{xx} \approx \mathbf{K}\mathbf{K}^\top$, so there exist $\mathbf{V}_n^x \approx \mathbf{U}_K$, where $\mathbf{U}_K$ are the eigenvectors of $\mathbf{K}\mathbf{K}^\top$. In Figure 5.4 we confirm that this Ansatz is good: in particular, we take the SVD of $\mathbf{W}_1 = \mathbf{U}_1\mathbf{\Lambda}_1\mathbf{V}_1^\top$. Then we find cubic streaks in the Fourier domain representation of $\mathbf{\Lambda}_1\mathbf{V}_1^\top$ mimicking those

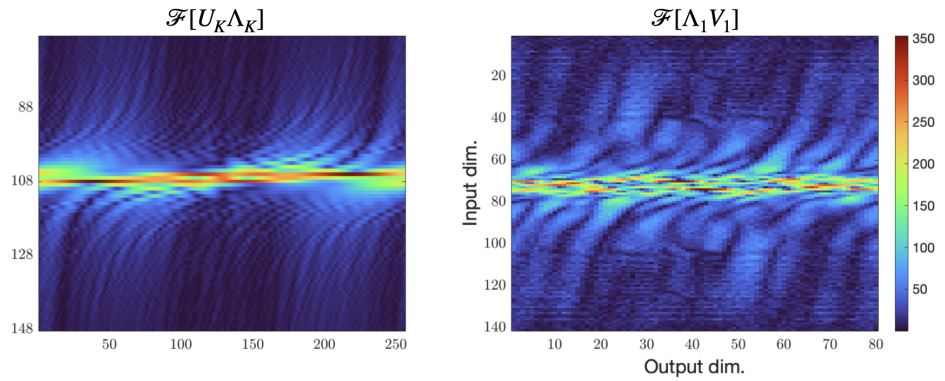


Figure 5.4: Left: The 2D FFT of the singular vectors of the discretized integral kernel in (5.36). Right: The 2D FFT of the scaled right singular vectors of the trained weights. Note that the panel on the right resembles a noisier version of the left panel, suggesting that the first layer of the NN acts as a pseudo-inverse.

in the Fourier domain representation of $\mathbf{U}_K \mathbf{\Lambda}_K$, thus validating the interpretation of (a part of) $\mathbf{W}_1$ as a notch filter in Figure 5.4. Thus, the projection induced by principal components–now viewed as descrambling–helps us identify a model for the training process of the first layer.

5.3.3 Discovering motifs within singular vectors of higher layers

Theorems 5.2.6 and 5.2.9 are limited due to the restrictive assumptions on the distribution of the hidden layer activations. Usually these assumptions are only satisfied by the training data itself, thus making the theoretical characterizations of the descramblers $\widehat{\mathbf{P}}_{SC}(k, \mathbf{X}, n)$ only applicable for the first layer. Nonetheless, in [AKCK21] it was illustrated that the *second* layer of DEERNet regularizes the incoming signal by projection onto the Chebyshev basis. However, these features were discovered by descrambling both the rows and columns of the second weight matrix $\mathbf{W}_2$ simultaneously. Here we illustrate that the emergence of this orthogonal basis is much more straightforward: in particular, by resorting to Theorem 5.2.4 we visualize the left singular vectors $\mathbf{U}_S$ of the second layer activations $\mathbf{S} = f_2(\mathbf{X})$. We discover that the largest singular vectors of $\mathbf{U}_S$ resemble a system of orthogonal polynomials. This is because (1) the singular vectors at odd (resp. even) indices tend to resemble

odd (resp. even) functions and (2) the singular vectors, although oscillating, are off-phase so a polynomial structure is more plausible than a trigonometric structure. We confirm this fact in Figure 5.5. We also test the hypothesis that these singular vectors represent noisy realizations of the Chebyshev polynomials C_i by computing the pairwise inner products $\langle C_i, u_S^j \rangle$ and find a banded structure in the inner product matrix, suggesting that the jth singular vector may indeed reside in the subspace formed by some $C_{j-k}, \ldots, C_{j+k}$ for some small k. In any case, here we have illustrated that a complicated optimization procedure via the Cayley transform may be circumvented in favour of an eigendecomposition. This eigendecomposition, when viewed by a subject matter expert, can nevertheless lead to helpful explanations of the mechanisms of the NN at the kth layer.

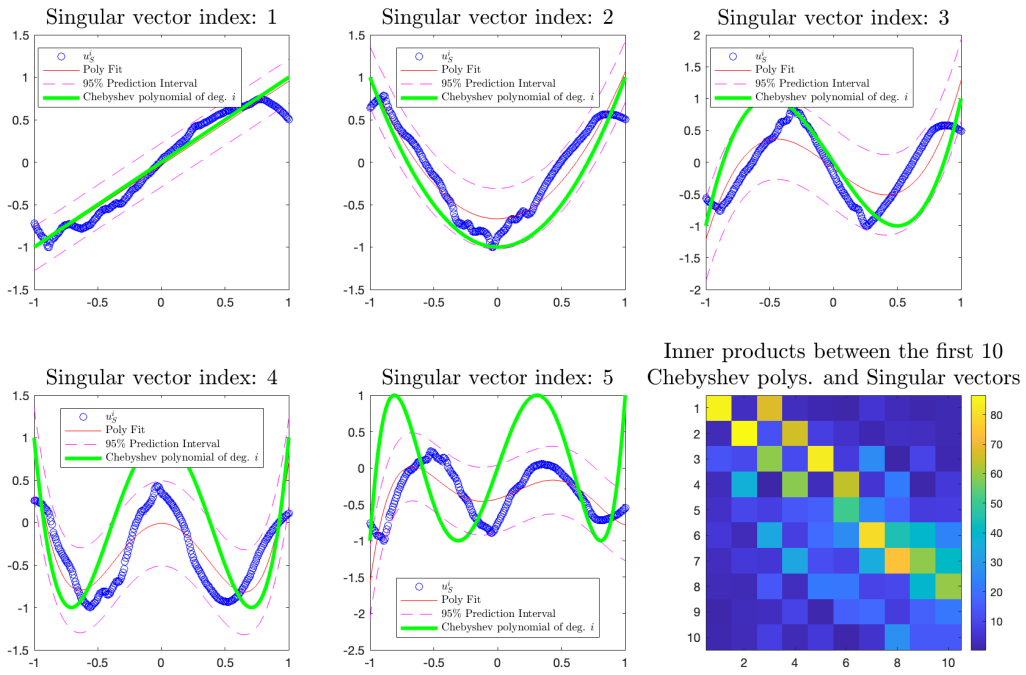


Figure 5.5: The top five principal components $\{u_S^i\}_{i=1}^5$ of the second layer preactivation data $\mathbf{S} = \mathbf{W}_2 f_1(x)$ resemble a system of orthogonal polynomials. In [AKCK21] a powerful plausibility argument suggested these can be fit by Chebyshev polynomials. Bottom right: After shifting and rescaling the principal components, the inner product matrix between the top 20 Chebyshev polynomials C_n and the principal components u_S^i is approximately banded.

5.3.4 Limitations of NN descrambling

NN descrambling amounts to computing a linear transformation of NN weights to bring them into readable form. The linear nature of the explanation process inherently limits the explanatory capacity of the algorithm. This is especially pertinent when attempting to understand the behaviour of a network layer after the nonlinearity has been applied. To illustrate this, below we conduct the following experiment. For the two layer *DEERNet*, we will attempt to understand the hidden layer dimension with 80 neurons. To do so, we descramble at the first layer before and after the nonlinearity. This amounts to computing:

$$\widehat{\mathbf{P}}_{SC}(1, \mathbf{X}, n) := \underset{\mathbf{P}^\top \mathbf{P} = \mathbf{I}}{\operatorname{argmin}} \|\mathbf{D}\mathbf{P}\mathbf{W}_1(\mathbf{X})\|_F^2, \tag{5.44}$$

$$\tilde{\mathbf{P}}_{SC}(2, \mathbf{X}, n) := \underset{\mathbf{P}^\top \mathbf{P} = \mathbf{I}}{\operatorname{argmin}} \|\mathbf{D}\mathbf{P}\phi(\mathbf{W}_1(\mathbf{X}))\|_F^2. \tag{5.45}$$

Here $\phi(x) = \mathsf{tanh}(x)$, the hyperbolic tangent nonlinearity. After computing $\widehat{\mathbf{P}}$ and $\tilde{\mathbf{P}}$ as above (with the data and dimension dependence suppressed) we compute $\tilde{\mathbf{W}}_2 := \mathbf{W}_2\tilde{\mathbf{P}}^\top$ and $\mathbf{W}_1 = \widehat{\mathbf{P}}\mathbf{W}_1$. This process was suggested in [AKCK21] to understand the hidden, or "link" dimension of the NN. In Figure 5.6 we visualize this experiment where we compare the raw weights, descrambled weights, and their top singular vectors before and after descrambling. In the top row, we find that descrambling brings the first weight matrix into readable form and descrambles a noisy singular vector into a smoothly oscillating signal, which can now be understood as part of a filter bank. However, the same process fails to provide any reasonable explanation for when the data is passed through the nonlinearity–the data remains uninterpretable after descrambling. In fact, as shown in Figure 5.5. Thus, while descrambling gives a modicum of interpretation to the hidden dimension as a 'time-axis",

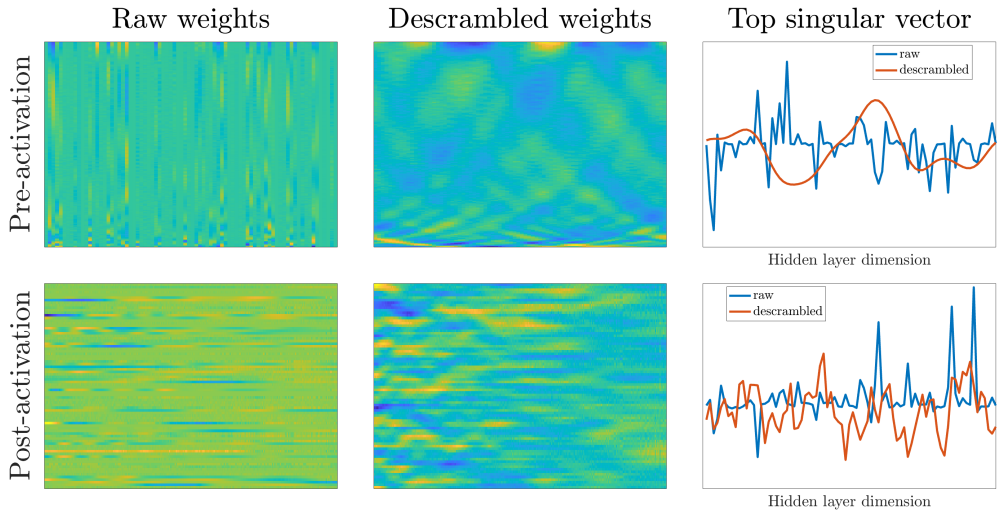


Figure 5.6: We compare descrambling in layer 1 before and after the nonlinearity tanh. Note that post-nonlinearity, neither the descrambled singular vectors nor the descrambled weights are understandable, unlike pre-nonlinearity, where an explanation of the first layer as a frequency domain filter is much more plausible

it fails to give an explanation for how the second weight matrix acts on this time axis. This is because descrambling is a linear technique, and the activation function represents a highly nonlinear deformation of the input axis to the second layer weights. Therefore, a linear rearrangement of this input axis is unlikely to bring the succeeding weight matrix in readable or explainable form.

5.4 Explaining input layer regularization

The preceding sections recast the singular value decomposition as an *interpretable* factorization of neural network weights, since the singular vectors seem to reveal the motifs learned from the training data. We now use this understanding of the SVD of network weights to explain input layer regularization introduced in Chapter 4. Specifically, we will give a plausibility argument for why concatenation by a smooth signal might improve generalization error. Recall that ILR involves concatenating a noisy signal $\mathbf{s}$ with a smooth version and feeding the resulting concatenated vector to a neural network termed (ND, REG). Our results in Chapter 4 showed that such (ND, REG) networks attain better generaliza-

tion error than (ND, ND) networks trained without this smooth signal, on two copies of the noisy data. In Section 4.2 we offered a plausibility argument for this improvement by connecting the (ND, REG) networks with an empirical Bayes estimator of the true parameter(s). In particular, we argued that the smooth vector encodes a prior on the unknown parameters $\mathbf{p}$ and an empirical Bayes estimators uses this prior information in addition to the data. However, it remained unclear *how* a neural network in particular could use this additional prior information to improve the mean squared error performance. To underline this point, in [RPB+22], ILR networks were implemented for the two-parameter problem where c_1 was known to be 0.6. Here, despite the λ being chosen suboptimally (the value was fixed at $\lambda = 10^{-4}$ obtained by a grid-based cross-validation search on training data), the (ND, REG) networks still outperformed the (ND, ND) networks. In this section we offer an explanation for why this might be the case. In particular, mirroring the experimental setup of [RPB+22], we train a four-layer NN to recover the parameters T_{21}, T_{22} from the noisy measurements of a biexponential signal with $c_1 = 0.6$:

$$\mathbf{s} := (0.4\exp(-t_i/T_{21}) + 0.6\exp(-t_i/T_{22}))_{i=1}^{m} + \eta_3. \tag{5.46}$$

Here $\eta_3 \sim \mathcal{N}(0, I_m)$ is additive white Gaussian noise. Note that recovering (T_{21}, T_{22}) from $\mathbf{s}$ with additive noise is a considerably simpler problem than the one discussed so far and has limited clinical applicability. Nonetheless, due to its status as the simplest non-trivial ill-posed problem (see [Lan88]) it is an important test problem for neural networks used in clinical settings. We trained both (ND, ND) and (ND, REG) networks for this problem and to explain the improved generalization capacity of (ND, REG) we computed the singular value decomposition of the weight matrices of the *first* layer of both networks, since they differed only in their input structure. We present the results in Figure 5.7. Here we show that the right singular vectors corresponding to the largest singular val-

ues of the first layer are themselves biexponential curves, thus showing that the networks learn an input signal library in the class of its training data. Most notably, the (ND, REG) networks learn two markedly different shapes of biexponential for the noisy and smooth parts while the corresponding parts in the singular vectors of the first layer weights of the (ND, ND) network nearly identical. Thus, (ND, REG) networks learn a richer function class in the first layer, corresponding to an improved estimation capacity.

5.4.1 Improving training with explainability

We now address the question of how *post-hoc* explanations–explanations of a network after it has been trained–can be exploited to improve the training of subsequent networks in the context of the two-parameter networks discussed above. Specifically, it has been shown empirically and theoretically (in a limited set of cases) that network training evolves the singular vectors of the first layer weights to the same signal model as the input data. To this end, we test whether pre-setting these singular vectors to the signal model can help generalization. We set up the following experiment: the first layer weights are prefactored as $\mathbf{W_1} = \mathbf{U_1}\mathbf{\Lambda_1}V_1^\top$ and instead of training $\mathbf{W_1}$ we train $\mathbf{U_1}, \mathbf{\Lambda_1}, \mathbf{V_1}$ under orthogonality and diagonality constraints. As such, there are two central ways in which we enforce these constraints.:

Hard constraint. The constraint is enforced directly, and all the matrices are exactly orthogonal or diagonal. We can parameterize the orthogonal matrices $\mathbf{U_1}, \mathbf{V_1}$ through the Cayley transform (5.39) and the singular value matrix multiplication as $\texttt{diag}(\lambda)X$ and train on s. Additionally, since λ must be positive, we parameterize it as $\lambda = \text{softmax}(\ell)$ or $\lambda = \ell^2$ where ℓ is the underlying trainable parameter.

Soft constraint. The orthogonality constraint is enforced through the loss function, by adding a penalty

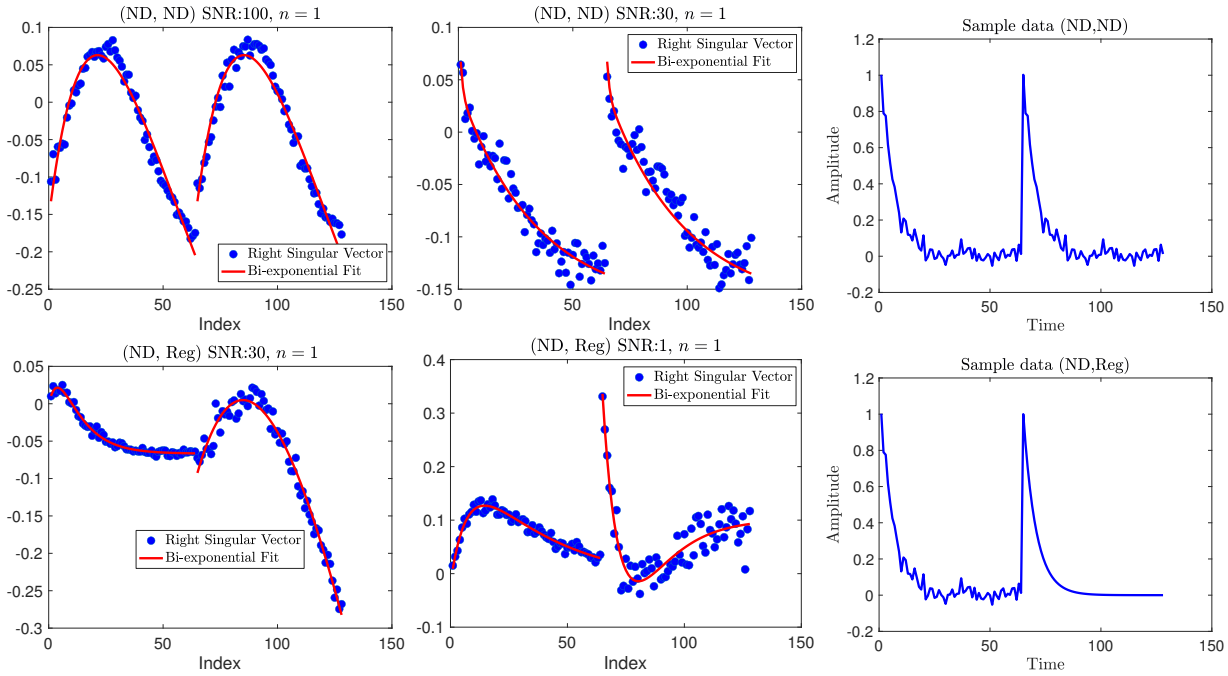


Figure 5.7: We visualize the singular vectors of the first layer of ILR networks trained on concatenations of noisy time-series data with the functional form $c_1 \exp(-t/T_{2,1}) + c_2 \exp(-t/T_{2,2})$ for $c_1 = 0.6, c_2 = 0.4$. Top row, left and center: The top singular vectors of the (ND, ND) networks for the first layer, for $\lambda^{-1} = 1, 100$. Top row, right: A sample training point in the (ND, ND) network. Bottom row: Same analysis as top row, for (ND, ND) networks, for $\lambda^{-1} = 1, 30$. Bottom row, right: A sample training point for the (ND, REG) network. Note that the second half of the signal is a smooth version of the noisy first half. We find that the dominant singular vectors of the first layer weights themselves can be fit by curves of the same algebraic form as the noiseless version of the training data, allowing for the c_i's to be negative. However, while the (ND, ND) networks learn the same biexponentials in each halves of the singular vectors, each right singular vector of the weights of the (ND, REG) networks learn exhibits markedly different biexponentials in either half. This phenomenon occurs across a wide range of SNR's and for both classes of networks–(ND, ND) and (ND, REG). Here (ND, REG) refers to networks trained on data points given by a concatentation of noisy and Tikhonov regularized copies of the same data, while (ND, ND) refers to networks trained on concatenation of the noisy data to itself.

term of the form $\|\mathbf{U}_1^\top \mathbf{U}_1 - \mathbf{I}\|$ (and similarly for $\mathbf{V_1}$). This is used in Yang et al 2021 for neural network compression.

Now, we need to incorporate the knowledge that the first layer right singular vectors $\mathbf{V_1}$ should come from the underlying signal model. Once again, we do this in two ways:

Frozen $\mathbf{V_1}$. In this case, we fix $\mathbf{V_1}$ to either be:

(a) The 32 top singular vectors of the covariance matrix of the data. In particular, let $\overline{\mathbf{\Sigma}}_n^{xx}$ be the covariance matrix of the input data and $\overline{\mathbf{\Sigma}}_n^{xx} = \overline{\mathbf{V}}_n^{xx}\overline{\mathbf{\Lambda}}_n^{xx}[\overline{\mathbf{V}}_n^{xx}]^\top$ be its SVD. Then we pick $\mathbf{V_1}$ to be $\overline{\mathbf{V}}_n^{xx}$ (or the first n_1 columns of $\overline{\mathbf{V}}_n^{xx}$ where n_1 is the prescribed layer width). The idea behind this is as follows: the first layer is a linear map described by $\mathbf{X} \to \mathbf{W}_1\mathbf{X}$. If $\mathbf{X} = \overline{\mathbf{V}}_n^{xx}\mathbf{Z}$ where $\mathbf{Z}$ is some latent variable and $\mathbf{W_1} = \mathbf{R}_1[\overline{\mathbf{V}}_n^{xx}]^\top$ then $\mathbf{W_1X} = \mathbf{R_1}[\overline{\mathbf{V}}_n^{xx}]^\top\overline{\mathbf{V}}_n^{xx}\mathbf{Z} = \mathbf{R}_1\mathbf{Z}$ where Y is the underlying latent variable, so now the linear map in $\mathbf{X}$ is a linear map in the latent variable $\mathbf{Z}$. This is basically PCA; the same idea underlies the concept of neural network deconvolution proposed in [YEH+20]. To represent this parallel, we will call this strategy `deconv`.

(b) The orthogonal basis of the subspace spanned by 32 randomly selected training points. This orthogonal basis is picked through a QR decomposition. We will call this strategy `QR`.

Trainable $\mathbf{V_1}$. We *initialize* $\mathbf{V_1}$ to be the set of signals given by `deconv` or `QR` followed by network training either with the hard constraint or soft constraint detailed above.

As a reference, we also trained a neural network with its first layer factorized in SVD without introducing the influence of the signal model. The results are presented in Table 5.2. The aforementioned fully factorized SVD without the signal model attains the lowest test error, but the individual singular vector matrices being trained are far from being orthogonal as measured by the orthogonality loss—this means that the network was probably treating $\mathbf{U_1}$ and $\mathbf{V_1}$ as linear layers, so it is not entirely surprising that adding slightly more parameters helped in improving test error. Keeping this case aside, it is notable that the networks with fixed right singular vectors (either via `deconv` or `QR`) perform at least 40% better over the benchmark. This suggests that explicitly setting the singular vectors to a fixed basis helps instead of hinders the network performance overall.

5.5 Understanding neural STFT phase retrieval

Short Time Fourier Transform (STFT) phase retrieval is a fundamental problem in signal processing arising in diverse applications including audio processing, X-ray crystallography, and optics [JEH16, FS20, Mil90, LXD12, AJD18]. The STFT problem amounts to finding a vector $x \in \mathbb{R}^d$ from its *spectrogram*, denoted $\mathrm{STFT}^w_{a,M}(x)$, characterized by a window w, hop length a and a number of frequency bins M defined as follows:

$$\mathrm{STFT}^w_{a,M}(x)[\ell, k] = \left| \sum_{j=0}^{m-1} x_{j+ak} \overline{w}_j \exp\left(-2\pi i \frac{j\ell}{M}\right) \right|. \tag{5.47}$$

Here the number of hops represent measurements over time and the number of frequency bins represent measurements over space. Thus the spectrogram is a local time-frequency representation of a signal modulated by the window and it is possible to reconstruct the signal given enough time-frequency measurements. The total number of measurements in (5.47) is $m = (\lfloor d/a \rfloor + 1)(\lfloor d/M \rfloor + 1)$ and the STFT may be rearranged as a non-negative matrix in $\mathbb{R}^{(\lfloor d/a \rfloor + 1) \times (\lfloor d/M \rfloor + 1)}$. The nonlinear nature of the STFT operator severely complicates the well-posedness of the recovery of x from $\mathrm{STFT}^w_{a,M}(x)$. In particular, the problem is hampered by non-uniqueness even beyond a global phase ambiguity where two possible solutions x, x' are considered equivalent up to global phase if and only if $x = \alpha x'$ where $|\alpha| = 1$. There is in fact a sharp characterization for the conditions under which STFT phase retrieval is possible (and vice versa, impossible). In particular, a celebrated result of [BCE06] shows that STFT phase retrieval is unique if and only if $m \geq (\text{resp. } \leq) 2d - 2$. Nonetheless, in applications such as audio signal processing (where STFT phase retrieval is known as *spectrogram inversion*) it is still of interest to recover the correct vector modeling an audio signal from the infinitely many solutions to the STFT

phase retrieval problem in the non-unique setting when the number of measurements does not exceed $2d-2$. Besides this practical need, this problem also has some mathematical grounding: in particular, if the vectors x arise from a suitably low-dimensional subspace or manifold then the complexity of phase retrieval may not be as high as $2d-2$. To probe this question empirically, in [CJSW24] we trained a four layer neural network PR_Θ to recover examples of human speech $x_i \in \mathbb{R}^{8095}$ from undersampled STFT measurements $y_i = \text{STFT}^w_{a,M}(x_i)$ such that $m < 2d-2$. To do so, we designed a novel loss function that was invariant to global phase factors:

$$L_{\text{PR}} := \sum_{i=1}^{n-1} \|\text{STFT}^w_{a',M'}(x_i) - \text{STFT}^w_{a',M'}(\text{PR}_\Theta(y_i))\|_F^2. \tag{5.48}$$

Here we picked a', M' such that $\text{STFT}^w_{a',M'}$ is an invertible nonlinear function so $L_{\text{PR}} = 0$ *if and only if* $x_i = \alpha_i PR_\Theta(y_i)$ for any choice of $|\alpha_i| = 1$. L_{PR} is therefore an attractive loss function because a perfectly zero-loss model exactly solves the phase retrieval problem up to global phase ambiguity. We trained four different networks, for four different sampling regimes with $m = 4d, 2d, 2d, d/2$ to understand the effect of undersampling on phase retrival. When trained on the loss 5.48 our networks recovered human speech up to -8.0 dB even in the severely undersampled case $m = d/2$, i.e with only half as many measurements as the original signal. To probe how a neural network might be recovering a signal in this "impossible" regime of phase retrieval, we yet again turn to singular vector based explainability for a plausibility argument and inspect the first layer weights $\mathbf{W_1}$ for clues.

The results are presented in Figure 5.8. Note that since the inputs to the networks are STFT matrices ordered according to time and frequency, we can also order the singular vectors (columns of $\mathbf{V_1}$) according to the same time-frequency ordering as a matrix. We now address these singular vectors as *singular matrices* since these are reshaped. In Figure 5.8 we find a fascinating banded structure

to the singular matrices, revealing that learning converges the network weights to specific form of self-organization. Moreover, the row and column axes of the singular matrices can be associated with the time and frequency directions in the spectrogram. To separate out these time and frequency contributions, we compute the singular vectors of the singular matrices–this will result in right and left singular vectors arranged along the frequency and time directions respectively. We present the results in Figure 5.9 and note that the $m = d/2$ and $m = d$ cases *diverge*: in particular the time-ordered singular vectors for the $m = d/2$ networks are smooth and mildly oscillating while the corresponding singular vectors for the $m = d$ networks oscillate rapidly. To explain this phenomenon, recall that our loss 5.48 requires that the network construct a signal such that it reproduces the correct *oversampled* spectrogram from an undersampled one. This can be rephrased as a problem of interpolation: the network must reconstruct a dense time-frequency representation from a coarsely sampled one. Thus, in the $m = d/2$ case, it 'chooses' to interpolate between the different temporal measurements smoothly with the smooth and non-oscillating singular vectors exhibited in Figure 5.9.

We conclude this section with a note of caution–the explanations offered above are *plausibility arguments* and may be tested more rigorously with ablation studies not performed here. Nevertheless, we have demonstrated here that weight matrix singular vectors–recast as the optimal descramblers for the weights–can be inspected to find motifs for the inductive biases of a trained neural network.

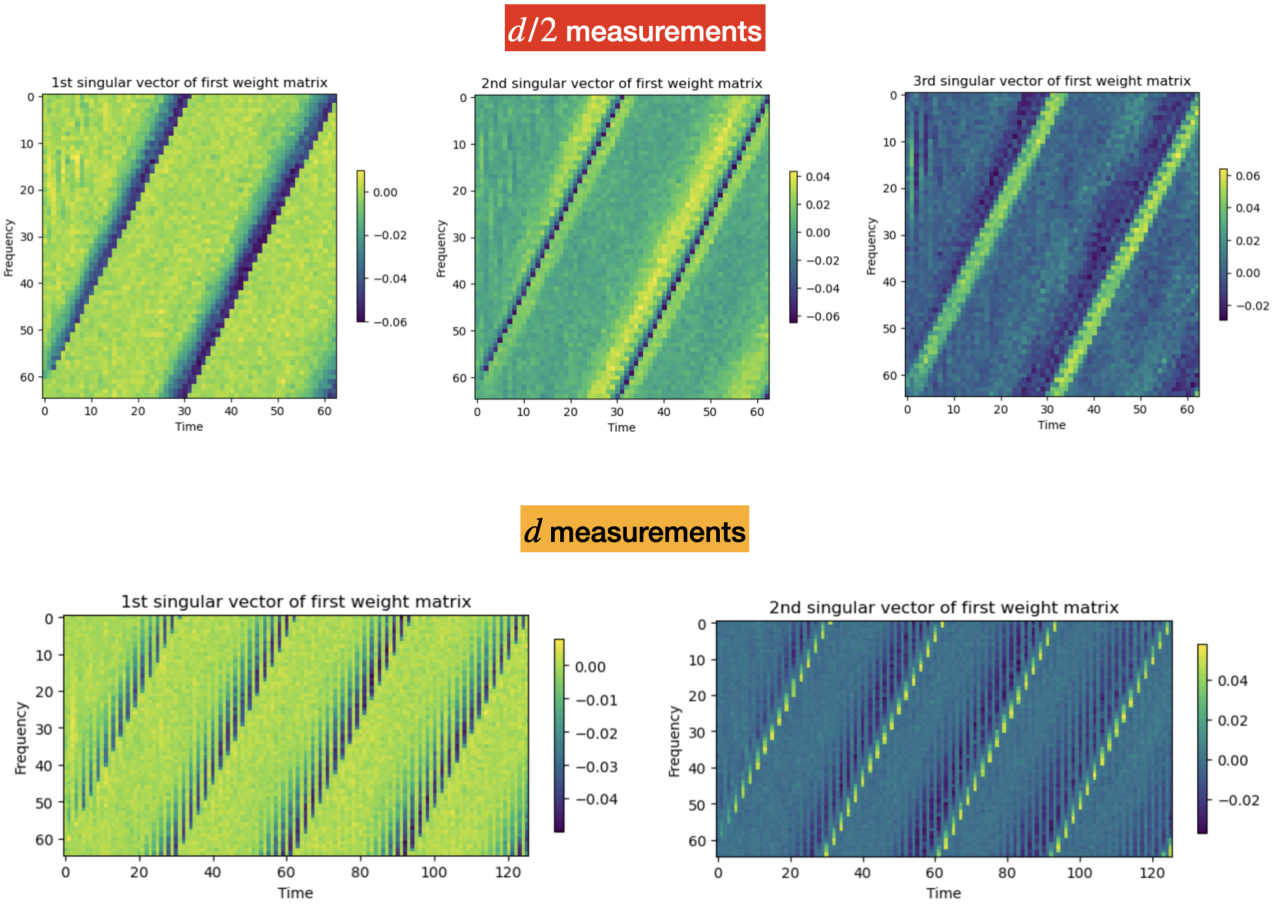


Figure 5.8: We visualize the top 3 singular vectors of $\mathbf{W_1}$ for the phase retrieval networks in [CJSW24] corresponding to the undersampled cases where the dimension of the underlying audio signal was $d = 8095$. We find a highly fascinating banded structure to the singular vectors, providing insights into how the networks perform phase retrieval

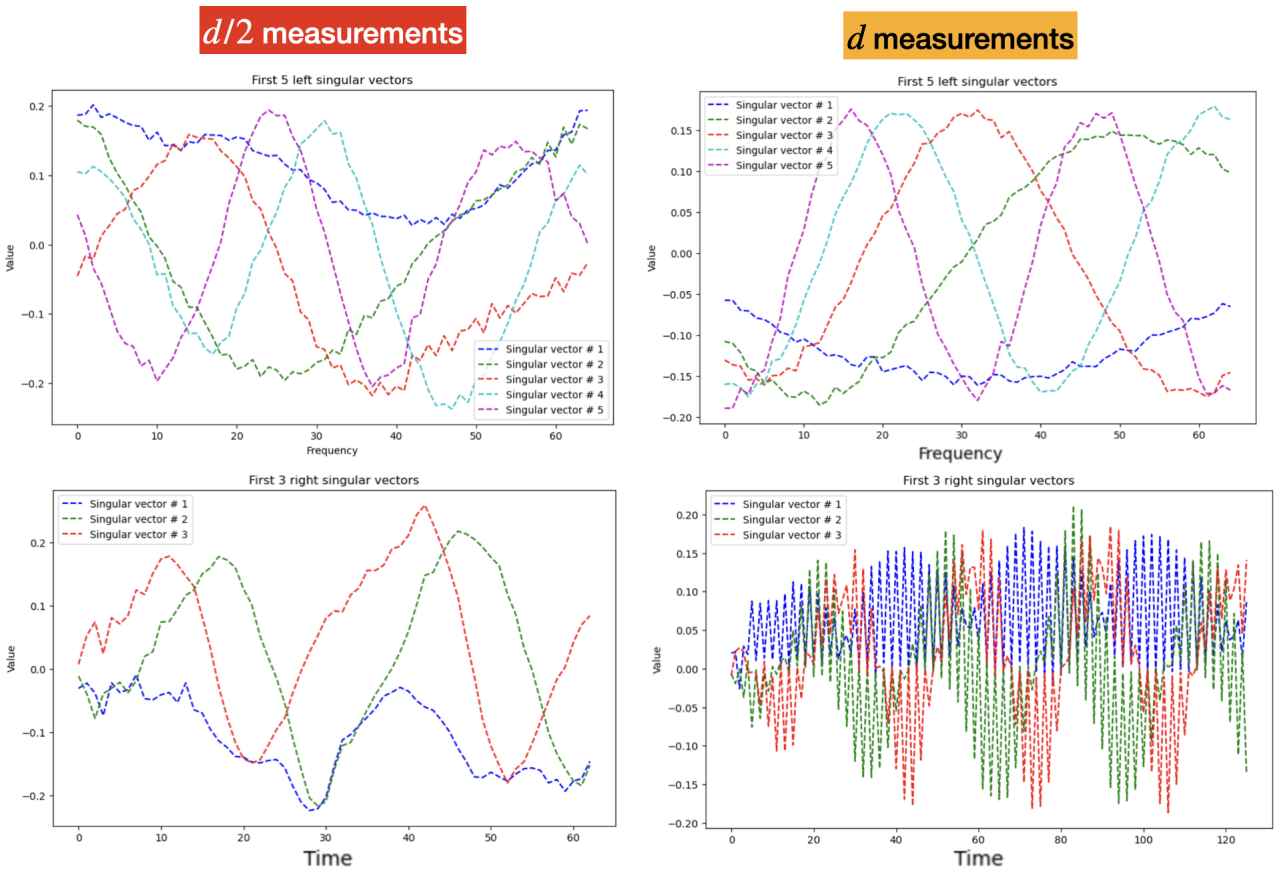


Figure 5.9: Singular vectors of the singular matrices formed by rearranging the SVD of the first layer weights from Figure 5.8. Computing the SVD of the singular matrices separates the representations along time and frequency. We find that the time and frequency signals learned by the $d/2$ network in the undersampled case are smooth and non-oscillating, indicating how the network smoothly interpolates the missing time-frequency measurements required to impute an invertible spectrogram.

Title	Training RMSE	Validation RMSE	Test RMSE	Orthogonality
Benchmark (4 layer ILR architecture, SNR = 100, T=64, reg=1.6e-4)	152	—	564.66	—
$\mathbf{V_1}$ fixed as `Deconv`, $\mathbf{U_1}$ through Cayley, $\mathbf{\Sigma_1}$ through softmax	148	33638.33	1186775.56	—
$\mathbf{V_1}$ fixed as `QR`, $\mathbf{U_1}$ through Cayley, $\mathbf{\Sigma_1}$ through softmax	159	68116.29	197281.51	—
$\mathbf{V_1}$ fixed as `Deconv`, $\mathbf{U_1}$ through Cayley, $\mathbf{\Sigma_1}$ through squared	146	75375.32	116256.50	—
$\mathbf{V_1}$ fixed as `QR`, $\mathbf{U_1}$ through Cayley, $\mathbf{\Sigma_1}$ through squared	153	79721.25	139523.68	—
$\mathbf{V_1}$ fixed as `Deconv`, $\mathbf{U_1}$, $\mathbf{\Sigma_1}$ through soft reg	158	251.49	327.69	8.64
$\mathbf{V_1}$ fixed as `QR`, $\mathbf{U_1}$, $\mathbf{\Sigma_1}$ through soft reg	155	251.45	324.21	5.31
Unconstrained SVD train	406	259.71	276.47	19.55, 194.35
$\mathbf{V_1}$ initialized as `Deconv`, all trained through soft reg	180	218.24	349.14	23.59, 0.29
$\mathbf{V_1}$ initialized as `QR`, all trained through soft reg	172	315.11	538.54	17.55, 0.53
$\mathbf{W_1}$ initialized as `Deconv` (orthogonal initialization)	153	408.68	525.66	N/A
$\mathbf{W_1}$ initialized as `QR` (orthogonal initialization)	153	336.69	528.60	N/A

Table 5.2: We present training and test RMSE with relevant orthogonality losses for (ND, REG) networks trained to solve the two-parameter biexponential recovery problem (5.46) with prefactored first layer weights to encode explainability.

Conclusion

This dissertation has developed mathematical perspectives on the analysis and design of data-driven algorithms across three scientific domains: molecular dynamics, magnetic resonance relaxometry, and neural network weight spaces. A unifying theme throughout is that high-dimensional scientific data possesses intrinsically low-dimensional structure—arising from metastability in molecular systems, the biophysical signal model in MRI, and the inductive biases of neural network architectures—and that principled algorithmic design requires understanding and exploiting this structure through rigorous mathematical analysis. Each chapter has identified a precise mathematical question, developed theory to answer it, and used that theory to motivate or improve a concrete algorithm.

Chapter 2 established sharp, pointwise consistency estimates for the TMDmap generator, with explicit analytical formulas for bias and variance prefactors, and proved that consistency implies stability for the committor boundary value problem. Chapter 3 provided a precise mathematical rationale for orthogonality-based collective variable design via relative entropy and pathwise distance control, and introduced the Laplacian Conformal Autoencoder (LAPCAE) to resolve topological artifacts in manifold learning embeddings. Chapter 4 rationalized input layer regularization (ILR) as an empirical Bayes estimator computing a maximally informative prior, motivating bilevel optimization for automated regularization tuning and yielding substantial improvements over both classical and standard deep learning baselines. Chapter 5 characterized the large-sample limit of neural network descram-

blers as a composition of PCA projection with expansion in the eigenbasis of a differentiation stencil, explaining empirically observed sparsity in the Fourier domain and demonstrating that signal-model-informed singular vectors improve generalization.

Open Problems and Future Directions

The results of this dissertation raise a number of mathematically precise open problems, organized below by theme.

Manifold Learning and Kernel Methods. The consistency analysis of Chapter 2 applies to the TMDmap generator under biased sampling, but several important extensions remain open. First, the analysis assumes a fixed kernel bandwidth; a rigorous theory of *adaptive* bandwidth selection—balancing bias and variance in regions of low sampling density—would significantly strengthen the practical applicability of diffusion map methods. Moreover, most sampling algorithms–including those in molecular dynamics–arise from *Hamiltonian* dynamics like the full Langevin dynamics outlined in Section 1.2.4. Data from these trajectories contains invaluable *kinetic* information that has yet not been used to inform spectral nonlinear dimensionality reduction algorithms. On a separate note, the diffusion map algorithm treats the time-series samples (either from biased or unbiased simulations) as statically sampled from the invariant distribution, which is often inaccessible in practice. Designing a diffusion map algorithms for time-varying densities is therefore a highly attractive goal in manifold learning.

Finally, Chapter 3 demonstrates empirically that group-invariant featurization is essential for collective variable discovery in molecular systems, but a general consistency theory for diffusion maps on group-invariant feature spaces remains to be developed. Equivariant graph neural networks [SHW21]

offer a natural parametric class for learning such kernels, and understanding their spectral approximation properties is an open and practically important question.

Collective Variables and Enhanced Sampling. The CV learning algorithm introduced in Chapter 3 has been validated on small benchmark systems (n-butane and alanine dipeptide), but its performance on larger multiscale systems such as fast-folding proteins is unknown. Scaling manifold learning to such systems will likely require bottom-up or hierarchical approaches to constructing high-quality data manifolds, and the development and analysis of such approaches is an open problem.

More broadly, Chapter 3 proves that the orthogonality condition (OC) on collective variables controls relative entropy and pathwise distances between projected and effective dynamics. Extending these results to orthogonality conditions with higher than a single degree of scale separation or to systems with implicit scale separation is thus an interesting problem to study.

Bilevel Optimization and Inverse Problems.

Chapter 4 motivates learning the regularization parameter via bilevel optimization and demonstrates strong empirical performance. On the theoretical side, a key open question is characterizing positivity of the optimal regularization parameter $\lambda(s)$, defined as:

$$\lambda(s) \in \operatorname*{argmin}_{\lambda \in [0,+\infty)} \|p(\lambda) - p\|_{\mathcal{H}}^2, \quad p(\lambda) \in \operatorname*{argmin}_{p \in \mathcal{F}} \|G(p) - s\|_{\mathcal{K}}^2 + \lambda \|p\|_{\mathcal{H}}^2. \tag{5.49}$$

Conditions for $\lambda(s) > 0$ are known when G is linear and invertible [EGS23], establishing when regularization genuinely improves over the least squares estimator. Extending these results to *bi-Lipschitz* forward maps—where standard smoothness arguments are unavailable but the map is still well-conditioned in a metric sense—would substantially broaden the theoretical foundations of learned regularization. A related direction concerns *holomorphic equilibrium propagation* (HEP), which com-

putes derivatives of minimizers via the Cauchy integral formula and Wirtinger derivatives [LZ22]. The interplay of HEP with gradient descent in bilevel problems remains unquantified; precisely characterizing the conditions under which HEP provides advantage over standard automatic differentiation could prove useful for training energy-based models and for hyperparameter optimization more broadly.

Neural Network Explainability and Weight Space Geometry. Chapter 5 characterizes descrambling in fully connected networks via singular value decomposition. Several natural extensions remain open. First, the descrambling framework can be generalized by replacing the orthogonal group with *Parseval frames* or *quadratic Lie groups* satisfying $P^\top \Pi P = \Pi$ [ML02]; such groups arise naturally in convolutional architectures, and developing a parallel singular vector theory in this setting would extend the explainability results of Chapter 5 to a much broader class of models.

Second, singular vectors have been used to probe training dynamics via average gradient outer products [BSMB24] and transformer block coupling [AMSP24]. It is an open question whether alternative matrix factorizations—such as non-negative matrix factorization, tensor decompositions, or structured sparse factorizations—can identify hidden invariants in weight spaces that correlate more directly with generalization or downstream task performance.

Closing Perspective

The gap between theory and practice in data-driven scientific computing remains wide, and bridging it will require sustained engagement with both the mathematical foundations and the scientific applications. The problems outlined above—in manifold learning, collective variable design, bilevel optimization, and neural network explainability—represent, in the author's view, some of the most tractable and consequential points of entry for future mathematical work in this space.

Bibliography

[ABG06] Hernan Alonso, Andrey A Bliznyuk, and Jill E Gready. Combining docking and molecular dynamic simulations in drug design. *Medicinal research reviews*, 26(5):531–568, 2006.

[ABH21] Harbir Antil, Tyrus Berry, and John Harlim. Fractional diffusion maps. *Applied and Computational Harmonic Analysis*, 54:145–175, 2021.

[ABT18] Richard C Aster, Brian Borchers, and Clifford H Thurber. *Parameter estimation and inverse problems*. Elsevier, 2018.

[ACC21] Babak Maboudi Afkham, Julianne Chung, and Matthias Chung. Learning regularization parameters of inverse problems via deep neural networks. *Inverse Problems*, 37(10):105017, sep 2021.

[ACM12] William K Allard, Guangliang Chen, and Mauro Maggioni. Multi-scale geometric methods for data sets ii: Geometric multi-resolution analysis. *Applied and computational harmonic analysis*, 32(3):435–462, 2012.

[ADVL+21] Giovanni S Alberti, Ernesto De Vito, Matti Lassas, Luca Ratti, and Matteo Santacesaria. Learning the optimal tikhonov regularizer for inverse problems. *Advances in Neural Information Processing Systems*, 34:25205–25216, 2021.

[AJD18] Sercan Ö Arık, Heewoo Jun, and Gregory Diamos. Fast spectrogram inversion using multi-head convolutional neural networks. *IEEE Signal Processing Letters*, 26(1):94–98, 2018.

[AKCK21] Jake L Amey, Jake Keeley, Tajwar Choudhury, and Ilya Kuprov. Neural network interpretation using descrambler groups. *Proceedings of the National Academy of Sciences*, 118(5), 2021.

[AMS08] P-A Absil, Robert Mahony, and Rodolphe Sepulchre. *Optimization algorithms on matrix manifolds*. Princeton University Press, 2008.

[AMSP24] Murdock Aubry, Haoming Meng, Anton Sugolov, and Vardan Papyan. Transformer block coupling and its correlation with generalization in llms. *arXiv preprint arXiv:2407.07810*, 2024.

[AON+08] Alex Altis, Martin Otten, Phuong H. Nguyen, Rainer Hegger, and Gerhard Stock. Dihedral angle principal component analysis of molecular dynamics simulations. *The Journal of Chemical Physics*, 128(24):245102, 2008.

[AV03] Jonas S Almeida and Eberhard O Voit. Neural-network-based parameter estimation in s-system models of biological networks. *Genome Informatics*, 14:114–123, 2003.

[Bal12] Pierre Baldi. Autoencoders, unsupervised learning, and deep architectures. In *Proceedings of ICML workshop on unsupervised and transfer learning*, pages 37–49. JMLR Workshop and Conference Proceedings, 2012.

[Bat14] Jonathan Bates. The embedding dimension of laplacian eigenfunction maps. *Applied and Computational Harmonic Analysis*, 37(3):516–530, 2014.

[BBM+23] Zineb Belkacemi, Marc Bianciotto, Hervé Minoux, Tony Lelièvre, Gabriel Stoltz, and Paraskevi Gkeka. Autoencoders for dimensionality reduction in molecular dynamics: Collective variable dimension, biasing, and transition states. *The Journal of Chemical Physics*, 159(2), 2023.

[BBP08] Alessandro Barducci, Giovanni Bussi, and Michele Parrinello. Well-tempered metadynamics: a smoothly converging and tunable free-energy method. *Physical review letters*, 100(2):020603, 2008.

[BBP11] Alessandro Barducci, Massimiliano Bonomi, and Michele Parrinello. Metadynamics. *Wiley Interdisciplinary Reviews: Computational Molecular Science*, 1(5):826–843, 2011.

[BCE06] Radu Balan, Pete Casazza, and Dan Edidin. On signal reconstruction without phase. *Applied and Computational Harmonic Analysis*, 20(3):345–356, 2006.

[BCMS06] James C Bremer, Ronald R Coifman, Mauro Maggioni, and Arthur D Szlam. Diffusion wavelet packets. *Applied and Computational Harmonic Analysis*, 21(1):95–112, 2006.

[BDFS22] Lukas Braun, Clémentine Dominé, James Fitzgerald, and Andrew Saxe. Exact learning dynamics of deep linear networks with prior knowledge. *Advances in Neural Information Processing Systems*, 35:6615–6629, 2022.

[BDH16] Anton Bovier and Frank Den Hollander. *Metastability: a potential-theoretic approach*, volume 351. Springer, 2016.

[BFBS22] Chuan Bi, Kenneth Fishbein, Mustapha Bouhrara, and Richard G Spencer. Stabilization of parameter estimates from multiexponential decay through extension into higher dimensions. *Scientific reports*, 12(1):1–16, 2022.

[BFS22] Moritz Böhle, Mario Fritz, and Bernt Schiele. B-cos networks: Alignment is all we need for interpretability. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10329–10338, 2022.

[BH89] Pierre Baldi and Kurt Hornik. Neural networks and principal component analysis: Learning from examples without local minima. *Neural Networks*, 2(1):53–58, 1989.

[BH16] Tyrus Berry and John Harlim. Variable bandwidth diffusion kernels. *Applied and Computational Harmonic Analysis*, 40(1):68–96, 2016.

[BK17] Ralf Banisch and Péter Koltai. Understanding the geometry of transport: Diffusion maps for lagrangian trajectory data unravel coherent sets. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 27(3), 2017.

[BLT13] Zdravko I Botev, Pierre L'Ecuyer, and Bruno Tuffin. Markov chain importance sampling with applications to rare event probability estimation. *Statistics and Computing*, 23:271–285, 2013.

[BN03] Mikhail Belkin and Partha Niyogi. Laplacian eigenmaps for dimensionality reduction and data representation. *Neural computation*, 15(6):1373–1396, 2003.

[BN06] Mikhail Belkin and Partha Niyogi. Convergence of laplacian eigenmaps. *Advances in neural information processing systems*, 19, 2006.

[BN08] Mikhail Belkin and Partha Niyogi. Towards a theoretical foundation for laplacian-based manifold methods. *Journal of Computer and System Sciences*, 74(8):1289–1308, 2008.

[BO04] Martin Burger and Stanley Osher. Convergence rates of convex variational regularization. *Inverse problems*, 20(5):1411, 2004.

[BOBS22] Chuan Bi, M Yvonne Ou, Mustapha Bouhrara, and Richard G Spencer. Span of regularization for solution of inverse problems with application to magnetic resonance relaxometry of the brain. *Scientific reports*, 12(1):20194, 2022.

[BPP21] Luigi Bonati, GiovanniMaria Piccini, and Michele Parrinello. Deep learning the slow modes for rare events sampling. *Proceedings of the National Academy of Sciences*, 118(44):e2113533118, 2021.

[BRM+18] Mustapha Bouhrara, David A Reiter, Michael C Maring, Jean-Marie Bonny, and Richard G Spencer. Use of the nesma filter to improve myelin water fraction mapping with brain MRI. *Journal of neuroimaging*, 28(6):640–649, 2018.

[Bro89] Roger W Brockett. Least squares matching problems. *Linear Algebra and its Applications*, 122:761–777, 1989.

[BS16a] Tyrus Berry and Timothy Sauer. Local kernels and the geometric structure of data. *Applied and Computational Harmonic Analysis*, 40(3):439–469, 2016.

[BS16b] Mustapha Bouhrara and Richard G Spencer. Improved determination of the myelin water fraction in human brain using magnetic resonance imaging through bayesian analysis of mcdespot. *Neuroimage*, 127:456–471, 2016.

[BSMB24] Daniel Beaglehole, Peter Súkeník, Marco Mondelli, and Mikhail Belkin. Average gradient outer product as a mechanism for deep neural collapse. *Advances in Neural Information Processing Systems*, 37:130764–130796, 2024.

[BTB+20] Ralf Banisch, Zofia Trstanova, Andreas Bittracher, Stefan Klus, and Péter Koltai. Diffusion maps tailored to arbitrary non-degenerate itô processes. *Applied and Computational Harmonic Analysis*, 48(1):242–265, 2020.

[Cam13] Maria K Cameron. Estimation of reactive fluxes in gradient stochastic systems using an analogy with electric circuits. *Journal of Computational Physics*, 247:137–152, 2013.

[Cas85] George Casella. An introduction to empirical bayes data analysis. *The American Statistician*, 39(2):83–87, 1985.

[CDCG+22] Salvatore Cuomo, Vincenzo Schiano Di Cola, Fabio Giampaolo, Gianluigi Rozza, Maziar Raissi, and Francesco Piccialli. Scientific machine learning through physics–informed neural networks: Where we are and what's next. *Journal of Scientific Computing*, 92(3):88, 2022.

[CF18] Wei Chen and Andrew L Ferguson. Molecular enhanced sampling with autoencoders: On-the-fly collective variable discovery and accelerated free energy landscape exploration. *Journal of computational chemistry*, 39(25):2079–2102, 2018.

[CHBJ+18] Travers Ching, Daniel S Himmelstein, Brett K Beaulieu-Jones, Alexandr A Kalinin, Brian T Do, Gregory P Way, Enrico Ferrero, Paul-Michael Agapow, Michael Zietz, Michael M Hoffman, et al. Opportunities and obstacles for deep learning in biology and medicine. *Journal of The Royal Society Interface*, 15(141):20170387, 2018.

[CJSW24] Wojciech Czaja, Canran Ji, Shashank Sule, and Matthias Wellershoff. Neural network-based speech reconstruction from undersampled stft magnitude data. In *2024 32nd European Signal Processing Conference (EUSIPCO)*, pages 406–410. IEEE, 2024.

[CKL+08] Ronald R Coifman, Ioannis G Kevrekidis, Stéphane Lafon, Mauro Maggioni, and Boaz Nadler. Diffusion maps, reduction coordinates, and low dimensional representation of stochastic systems. *Multiscale Modeling & Simulation*, 7(2):842–864, 2008.

[CL06] Ronald R Coifman and Stéphane Lafon. Diffusion maps. *Applied and computational harmonic analysis*, 21(1):5–30, 2006.

[Clo14] Alexander Cloninger. *Exploiting data-dependent structure for improving sensor acquisition and integration*. PhD thesis, University of Maryland, College Park, 2014.

[CM17] Miles Crosskey and Mauro Maggioni. Atlas: A geometric approach to learning high-dimensional stochastic systems near manifolds. *Multiscale Modeling & Simulation*, 15(1):110–156, 2017.

[CM19] Yu-Chia Chen and Marina Meila. Selecting the independent coordinates of manifolds with large aspect ratios. *Advances in Neural Information Processing Systems*, 32, 2019.

[CMCM19] EL Christian, Saïd Moussaoui, Guylaine Collewet, and Maja Musse. Spatially regularized multi-exponential transverse relaxation times estimation from magnitude magnetic resonance images under rician noise. In *2019 IEEE International Conference on Image Processing (ICIP)*, pages 1143–1147. IEEE, 2019.

[CMS07] Benoît Colson, Patrice Marcotte, and Gilles Savard. An overview of bilevel optimization. *Annals of operations research*, 153:235–256, 2007.

[CMW+21] Shengze Cai, Zhiping Mao, Zhicheng Wang, Minglang Yin, and George Em Karniadakis. Physics-informed neural networks (pinns) for fluid mechanics: A review. *Acta Mechanica Sinica*, 37(12):1727–1738, 2021.

[CPGM15] Rémi Cuchillo, Kevin Pinto-Gil, and Julien Michel. A collective variable for the rapid exploration of protein druggability. *Journal of Chemical Theory and Computation*, 11(3):1292–1307, 2015.

[CS18] Daniela Calvetti and Erkki Somersalo. Inverse problems: From regularization to bayesian inference. *Wiley Interdisciplinary Reviews: Computational Statistics*, 10(3):e1427, 2018.

[CSF19] Wei Chen, Hythem Sidky, and Andrew L Ferguson. Nonlinear discovery of slow molecular modes using state-free reversible vampnets. *The Journal of chemical physics*, 150(21), 2019.

[CSP+07] John D. Chodera, N. Singhal, Vijay S. Pande, Ken A. Dill, and William C. Swope. Automatic discovery of metastable states for the construction of markov models of macromolecular conformational dynamics. *The Journal of Chemical Physics*, 126(15):155101, 2007.

[CW22a] Xiuyuan Cheng and Hau-Tieng Wu. Convergence of graph laplacian with knn self-tuned kernels. *A Journal of the IMA*, 11:889–957, 2022.

[CW22b] Xiuyuan Cheng and Hau-Tieng Wu. Convergence of graph laplacian with knn self-tuned kernels. *Information and Inference: A Journal of the IMA*, 11(3):889–957, 2022.

[CW22c] Xiuyuan Cheng and Nan Wu. Eigen-convergence of gaussian kernelized graph laplacian by manifold heat interpolation. *Applied and Computational Harmonic Analysis*, 61:132–190, 2022.

[DBJ21] Gerhard S Drenthen, Walter H Backes, and Jacobus FA Jansen. Estimating myelin-water content from anatomical and diffusion images using spatially undersampled myelin-water imaging through machine learning. *NeuroImage*, 226:117626, 2021.

[Deo10] Sean CL Deoni. Quantitative relaxometry of the brain. *Topics in Magnetic Resonance Imaging*, 21(2):101–113, 2010.

[DGVD10] Adrienne N Dula, Daniel F Gochberg, Heather L Valentine, and Mark D Does. Multi-exponential t2, magnetization transfer, and quantitative histology in white matter tracts of rat spinal cord. *Magnetic Resonance in Medicine*, 63(4):902–909, 2010.

[DLP+18] Manh Hong Duong, Agnes Lamacz, Mark A Peletier, André Schlichting, and Upanshu Sharma. Quantification of coarse-graining error in langevin and overdamped langevin dynamics. *Nonlinearity*, 31(10):4517, 2018.

[DLV+21] Adam V Dvorak, Emil Ljungberg, Irene M Vavasour, Lisa Eunyoung Lee, Shawna Abel, David KB Li, Anthony Traboulsee, Alex L MacKay, and Shannon H Kolind. Comparison of multi echo t2 relaxation and steady state approaches for myelin imaging in the central nervous system. *Scientific Reports*, 11(1):1369, 2021.

[DRA+08] Sean CL Deoni, Brian K Rutt, Tarunya Arun, Carlo Pierpaoli, and Derek K Jones. Gleaning multicomponent t1 and t2 information from steady-state imaging data. *Magnetic Resonance in Medicine: An Official Journal of the International Society for Magnetic Resonance in Medicine*, 60(6):1372–1387, 2008.

[DRGP08] Eric Darve, Daniel Rodríguez-Gómez, and Andrew Pohorille. Adaptive biasing force method for scalar and vector free energy calculations. *The Journal of Chemical Physics*, 128(14):144120, 2008.

[DVB+21] Alex Davies, Petar Veličkovi'c, Lars Bocking, Matt Blackwell, Sai Zheng, Nenad Tomasev, Corina Grabska-Barwi'nska, Keith Wegerif, Hubert Theophane, Demis van der Ploeg, et al. Advancing mathematics by guiding human intuition with ai. *Nature*, 600(7889):70–74, 2021.

[DZ20] Stephan Dempe and Alain Zemkoho. Bilevel optimization. In *Springer optimization and its applications*, volume 161. Springer, 2020.

[ECT22] Luke Evans, Maria K Cameron, and Pratyush Tiwary. Computing committors via mahalanobis diffusion maps with enhanced sampling data. *The Journal of Chemical Physics*, 157(21):214107, 2022.

[ECT23] Luke Evans, Maria K Cameron, and Pratyush Tiwary. Computing committors in collective variables via mahalanobis diffusion maps. *Applied and Computational Harmonic Analysis*, 64:62–101, 2023.

[EG92] L.C. Evans and R.F. Gariepy. *Measure Theory and Fine Properties of Functions. Studies in Advanced Mathematics*. CRC Press, Boca Raton, 1992.

[EG94] Heinz W Engl and Wilhelm Grever. Using the l–curve for determining optimal regularization parameters. *Numerische Mathematik*, 69(1):25–31, 1994.

[EGS23] Matthias J Ehrhardt, Silvia Gazzola, and Sebastian J Scott. On optimal regularization parameters via bilevel learning. *arXiv preprint arXiv:2305.18394*, 2023.

[EHMCM20] Christian El Hajj, Said Moussaoui, Guylaine Collewet, and Maja Musse. Multi-exponential transverse relaxation times estimation from magnetic resonance images under rician noise and spatial regularization. *IEEE Transactions on Image Processing*, 29:6721–6733, 2020.

[Emi23] Jeremiah Emidih. *Graph-Based Data Fusion With Applications to Magnetic Resonance Imaging*. PhD thesis, University of Maryland, College Park, 2023.

[EPGB21] Dennis Elbrächter, Dmytro Perekrestenko, Philipp Grohs, and Helmut Bölcskei. Deep neural network approximation theory. *IEEE Transactions on Information Theory*, 67(5):2581–2623, 2021.

[Eva10] Lawrence C. Evans. *Partial differential equations*. American Mathematical Society, Providence, R.I., 2010.

[EVE06] Weinan E and Eric Vanden-Eijnden. Towards a theory of transition paths. *Journal of statistical physics*, 123(3):503, 2006.

[FCR+22] Sebastian Falkner, Alessandro Coretti, Salvatore Romano, Phillip Geissler, and Christoph Dellago. Conditioning normalizing flows for rare event sampling. *arXiv preprint arXiv:2207.14530*, 2022.

[FKB+18] Tobias D Faizy, Dushyant Kumar, Gabriel Broocks, Christian Thaler, Fabian Flottmann, Hannes Leischner, Daniel Kutzner, Simon Hewera, Dominik Dotzauer, Jan-Patrick Stellmann, et al. Age-related measurements of the myelin water fraction derived from 3d multi-echo grase reflect myelin content of the cerebral white matter. *Scientific reports*, 8(1):14991, 2018.

[FP17] Sean A. Fitzgerald and Michele Parrinello. Diffusion tensors in collective variables spaces: Application to metadynamics. *The Journal of Chemical Physics*, 147(15):152716, 2017.

[FS20] Albert Fannjiang and Thomas Strohmer. The numerics of phase retrieval. *Acta Numerica*, 29:125–228, 2020.

[GBK+23] Zhaoyuan Gong, Murat Bilgel, Matthew Kiely, Curtis Triebswetter, Luigi Ferrucci, Susan M Resnick, Richard G Spencer, and Mustapha Bouhrara. Lower myelin content is associated with more rapid cognitive decline among cognitively unimpaired individuals. *Alzheimer's & Dementia*, 19(7):3098–3107, 2023.

[GHO99] Gene H Golub, Per Christian Hansen, and Dianne P O'Leary. Tikhonov regularization and total least squares. *SIAM journal on matrix analysis and applications*, 21(1):185–194, 1999.

[GHW79] Gene H Golub, Michael Heath, and Grace Wahba. Generalized cross-validation as a method for choosing a good ridge parameter. *Technometrics*, 21(2):215–223, 1979.

[GKS04] Dror Givon, Raz Kupferman, and Andrew Stuart. Extracting macroscopic dynamics: model problems and algorithms. *Nonlinearity*, 17(6):R55, 2004.

[GMMM19] Behrooz Ghorbani, Song Mei, Theodor Misiakiewicz, and Andrea Montanari. Limitations of lazy training of two-layers neural network. *Advances in Neural Information Processing Systems*, 32, 2019.

[GPAM+20] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks. *Communications of the ACM*, 63(11):139–144, 2020.

[GQ20] JEAN QUAINTANCE Gallier and Jocelyn Quaintance. *Differential geometry and lie groups*, volume 12. Springer, 2020.

[GRADM24] Javier Gurrola-Ramos, Teresa Alarcon, Oscar Dalmau, and José V Manjón. Mri rician noise reduction using recurrent convolutional neural networks. *IEEE Access*, 2024.

[GTGHS20] Nicolás García Trillos, Moritz Gerlach, Matthias Hein, and Dejan Slepčev. Error estimates for spectral convergence of the graph laplacian on random geometric graphs toward the laplace–beltrami operator. *Foundations of Computational Mathematics*, 20(4):827–887, 2020.

[GVM97] Gene H Golub and Urs Von Matt. Generalized cross-validation for large-scale problems. *Journal of Computational and Graphical Statistics*, 6(1):1–34, 1997.

[GXH+22] Rui Guo, Song Xue, Jiaxi Hu, Hasan Sari, Clemens Mingels, Konstantinos Zeimpekis, George Prenosil, Yue Wang, Yu Zhang, Marco Viscione, et al. Using domain knowledge for robust and generalizable deep learning-based ct-free pet attenuation and scatter correction. *Nature Communications*, 13(1):5882, 2022.

[GZ07] Alan Grossfield and Daniel M. Zuckerman. Application of principal components analysis to distortion of biomolecular complexes. *Theoretical Chemistry Accounts*, 116:44–50, 2007.

[HAL05] Matthias Hein, Jean-Yves Audibert, and Ulrike von Luxburg. From graphs to manifolds–weak and strong pointwise consistency of graph laplacians. In *International Conference on Computational Learning Theory*, pages 470–485. Springer, 2005.

[HAL07] Matthias Hein, Jean-Yves Audibert, and Ulrike von Luxburg. Graph laplacians and their convergence on random neighborhood graphs. *Journal of Machine Learning Research*, 8(6), 2007.

[HK11] Ryuhei Harada and Akio Kitao. Exploring the folding free energy landscape of a β-hairpin miniprotein, chignolin, using multiscale free energy landscape calculation method. *The Journal of Physical Chemistry B*, 115(27):8806–8812, 2011.

[HK20] Konrad Hinsen and Gerald R. Kneller. Hydrogen bonds vs rmsd: Geometric reaction coordinates for the simulation of conformational transitions. *The Journal of Chemical Physics*, 162(7):074107, 2020.

[HK23] Paulina Hoyos and Joe Kileel. Diffusion maps for group-invariant manifolds. *arXiv preprint arXiv:2303.16169*, 2023.

[HKB18] Gernot Holler, Karl Kunisch, and Richard C Barnard. A bilevel approach for parameter learning in inverse problems. *Inverse Problems*, 34(11):115012, 2018.

[HLS+22] Jérôme Hénin, Tony Lelièvre, Michael R. Shirts, Omar Valsson, and Lucie Delemotte. Enhanced sampling methods for molecular dynamics simulations [Article v1.0]. *Living Journal of Computational Molecular Science*, 4(1):1583, December 2022.

[HMT09] Nathan Halko, Per-Gunnar Martinsson, and Joel A Tropp. Finding structure with randomness: Stochastic algorithms for constructing approximate matrix decompositions. *arXiv preprint arXiv:0909.4061*, 909, 2009.

[HNS20] Carsten Hartmann, Lara Neureither, and Upanshu Sharma. Coarse graining of nonreversible stochastic differential equations: Quantitative results and connections to averaging. *SIAM Journal on Mathematical Analysis*, 52(3):2689–2733, 2020.

[HO00] Eldad Haber and Douglas Oldenburg. A gcv based method for nonlinear ill-posed problems. *Computational Geosciences*, 4:41–63, 2000.

[Hsu02] Elton P Hsu. *Stochastic analysis on manifolds*. Number 38. American Mathematical Soc., 2002.

[HWK+20] Nana Heilmann, Moritz Wolf, Mariana Kozlowska, Elaheh Sedghamiz, Julia Setzler, Martin Brieg, and Wolfgang Wenzel. Sampling of the conformational landscape of small proteins with monte carlo methods. *Scientific reports*, 10(1):18211, 2020.

[HWSS+18] Carlos X Hernández, Hannah K Wayment-Steele, Mohammad M Sultan, Brooke E Husic, and Vijay S Pande. Variational encoding of complex dynamics. *Physical Review E*, 97(6):062412, 2018.

[Jac12] Dunham Jackson. *Fourier series and orthogonal polynomials*. Courier Corporation, 2012.

[JDV+18] Dongsheng Jiang, Weiqiang Dou, Luc Vosters, Xiayu Xu, Yue Sun, and Tao Tan. Denoising of 3d magnetic resonance images with multi-channel residual learning of convolutional neural network. *Japanese journal of radiology*, 36(9):566–574, 2018.

[JEH16] Kishore Jaganathan, Yonina C Eldar, and Babak Hassibi. Phase retrieval: An overview of recent developments. *Optical compressive imaging*, pages 279–312, 2016.

[JEP+21] John Jumper, Richard Evans, Alexander Pritzel, Tim Green, Michael Figurnov, Olaf Ronneberger, Kathryn Tunyasuvunakool, Russ Bates, Augustin 'Azquez "Amaro, Chris Ringrose, et al. Highly accurate protein structure prediction with alphafold. *Nature*, 596(7873):583–589, 2021.

[JH23] Shixiao Willing Jiang and John Harlim. Ghost point diffusion maps for solving elliptic pdes on manifolds with classical boundary conditions. *Communications on Pure and Applied Mathematics*, 76(2):337–405, 2023.

[JWH+13] Gareth James, Daniela Witten, Trevor Hastie, Robert Tibshirani, et al. *An introduction to statistical learning*, volume 112. Springer, 2013.

[JY03] Lei Jiang and Wenhui Yang. Adaptive magnetic resonance image denoising using mixture model and wavelet shrinkage. *Dicta*, pages 831–838, 2003.

[Käs11] Johannes Kästner. Umbrella sampling. *Wiley Interdisciplinary Reviews: Computational Molecular Science*, 1(6):932–942, 2011.

[KB14] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[KCM21] Dhruv Kohli, Alexander Cloninger, and Gal Mishne. Ldle: Low distortion local eigenmaps. *The Journal of Machine Learning Research*, 22(1):12914–12977, 2021.

[KDKD15] Sang Beom Kim, Carmeline J Dsilva, Ioannis G Kevrekidis, and Pablo G Debenedetti. Systematic characterization of protein folding pathways using diffusion maps: Application to trp-cage miniprotein. *The Journal of chemical physics*, 142(8), 2015.

[KGSM23] Zahra Kadkhodaie, Florentin Guth, Eero P Simoncelli, and Stéphane Mallat. Generalization in diffusion models arises from geometry-adaptive harmonic representation. *arXiv preprint arXiv:2310.02557*, 2023.

[KIRP21] Tarak Karmakar, Michele Invernizzi, Valerio Rizzi, and Michele Parrinello. Collective variables for the study of crystallisation. *Molecular Physics*, 119(19-20):e1893848, 2021.

[KMJB+12] Shannon Kolind, Lucy Matthews, Heidi Johansen-Berg, M Isabel Leite, Steven CR Williams, Sean Deoni, and Jackie Palace. Myelin water imaging reflects clinical variability in multiple sclerosis. *Neuroimage*, 60(1):263–270, 2012.

[KMVK24] George A Kevrekidis, Mauro Maggioni, Soledad Villar, and Yannis G Kevrekidis. Thinner latent spaces: Detecting dimension and imposing invariance through autoencoder gradient constraints. *arXiv preprint arXiv:2408.16138*, 2024.

[KPKP92] Peter E Kloeden, Eckhard Platen, Peter E Kloeden, and Eckhard Platen. *Stochastic differential equations*. Springer, 1992.

[KS07] Leonid Koralov and Yakov G Sinai. *Theory of probability and random processes*. Springer Science & Business Media, 2007.

[KSU03] Herbert Karner, Josef Schneid, and Christoph W Ueberhuber. Spectral decomposition of real circulant matrices. *Linear Algebra and Its Applications*, 367:301–311, 2003.

[KTLW19] Ori Katz, Ronen Talmon, Yu-Lun Lo, and Hau-Tieng Wu. Alternating diffusion maps for multimodal data fusion. *Information Fusion*, 45:346–360, 2019.

[Laf04] Stéphane S Lafon. *Diffusion maps and geometric harmonics*. Yale University, 2004.

[Lan88] Cornelius Lanczos. *Applied analysis*. Courier Corporation, 1988.

[Lax07] Peter D. Lax. *Linear Algebra and Its Applications*. Wiley-Interscience, Hoboken, NJ, second edition, 2007.

[LDr84] EM Landaw and JJ DiStefano 3rd. Multiexponential, multicompartmental, and noncompartmental modeling. ii. data analysis and statistical considerations. *American Journal of Physiology-Regulatory, Integrative and Comparative Physiology*, 246(5):R665–R677, 1984.

[LF19] Andrew W Long and Andrew L Ferguson. Landmark diffusion maps (l-dmaps): Accelerated manifold learning out-of-sample extension. *Applied and Computational Harmonic Analysis*, 47(1):190–211, 2019.

[LJHY21] Senwei Liang, Shixiao W Jiang, John Harlim, and Haizhao Yang. Solving pdes on unknown manifolds with machine learning. *arXiv preprint arXiv:2106.06682*, 2021.

[LL10] Frédéric Legoll and Tony Lelievre. Effective dynamics using conditional expectations. *Nonlinearity*, 23(9):2131, 2010.

[LL17] Scott M Lundberg and Su-In Lee. A unified approach to interpreting model predictions. *Advances in neural information processing systems*, 30, 2017.

[LL18] Rongjie Lai and Jianfeng Lu. Point cloud discretization of fokker–planck operators for committor functions. *Multiscale Modeling & Simulation*, 16(2):710–726, 2018.

[LLCC20] Henry Li, Ofir Lindenbaum, Xiuyuan Cheng, and Alexander Cloninger. Variational diffusion autoencoders with random walk sampling. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XXIII 16*, pages 362–378. Springer, 2020.

[LLO17] Frédéric Legoll, Tony Lelièvre, and Stefano Olla. Pathwise estimates for an effective dynamics. *Stochastic Processes and their Applications*, 127(9):2841–2863, 2017.

[LLS19] Frédéric Legoll, Tony Lelièvre, and Upanshu Sharma. Effective dynamics for non-reversible stochastic differential equations: a quantitative study. *Nonlinearity*, 32(12):4779, 2019.

[LMR17] Anna V Little, Mauro Maggioni, and Lorenzo Rosasco. Multiscale geometric methods for data sets i: Multiscale svd, noise and curvature. *Applied and Computational Harmonic Analysis*, 43(3):504–567, 2017.

[LP02a] Alessandro Laio and Michele Parrinello. Escaping free-energy minima. *Proceedings of the National Academy of Sciences*, 99(20):12562–12566, 2002.

[LP02b] Alessandro Laio and Michele Parrinello. Escaping free-energy minima. *Proceedings of the national academy of sciences*, 99(20):12562–12566, 2002.

[LSYZ21] Jianfeng Lu, Zuowei Shen, Haizhao Yang, and Shijun Zhang. Deep network approximation for smooth functions. *SIAM Journal on Mathematical Analysis*, 53(5):5465–5506, 2021.

[Luk93] Mark A Lukas. Asymptotic optimality of generalized cross-validation for choosing the regularization parameter. *Numerische Mathematik*, 66:41–66, 1993.

[Luo17] Ping Luo. Learning deep architectures via generalized whitened neural networks. In *International Conference on Machine Learning*, pages 2238–2246. PMLR, 2017.

[LVM+04] Cornelia Laule, Irene M Vavasour, Graham RW Moore, Joel Oger, David KB Li, and Donald W Paty. Water content and myelin water fraction in multiple sclerosis. *Brain*, 127(10):2209–2218, 2004.

[LW13] Dirk Lorenz and Nadja Worliczek. Necessary conditions for variational regularization schemes. *Inverse Problems*, 29(7):075016, 2013.

[LXD12] Z-C Liu, Rui Xu, and Y-H Dong. Phase retrieval in protein crystallography. *Foundations of Crystallography*, 68(2):256–265, 2012.

[LZ19a] T. Lelièvre and W. Zhang. Pathwise estimates for effective dynamics: The case of nonlinear vectorial reaction coordinates. *Multiscale Modeling & Simulation*, 17(3):1019–1051, 2019.

[LZ19b] Tony Lelièvre and Wei Zhang. Pathwise estimates for effective dynamics: the case of nonlinear vectorial reaction coordinates. *Multiscale Modeling & Simulation*, 17(3):1019–1051, 2019.

[LZ22] Axel Laborieux and Friedemann Zenke. Holomorphic equilibrium propagation computes exact gradients through finite size oscillations. *Advances in neural information processing systems*, 35:12950–12963, 2022.

[LZLL20] Sanqian Li, Jinjie Zhou, Dong Liang, and Qiegen Liu. Mri denoising using progressively distribution-based neural network. *Magnetic resonance imaging*, 71:55–68, 2020.

[MB79] Klaus Müller and Leo D Brown. Location of saddle points and minimum energy paths by a constrained simplex optimization procedure. *Theoretica chimica acta*, 53:75–93, 1979.

[MFVEC06] Luca Maragliano, Alexander Fischer, Eric Vanden-Eijnden, and Giovanni Ciccotti. String method in collective variables: Minimum free energy paths and isocommittor surfaces. *The Journal of chemical physics*, 125(2), 2006.

[Mil90] Rick P Millane. Phase retrieval in crystallography and optics. *Journal of the Optical Society of America A*, 7(3):394–411, 1990.

[ML02] Luıs Miguel Machado and F Silva Leite. Optimization on quadratic matrix lie groups. *Pré-Publicaçoes do Departamento de Matemática*, (02-21), 2002.

[ML16] Alex L MacKay and Cornelia Laule. Magnetic resonance of myelin water: an in vivo marker for myelin. *Brain plasticity*, 2(1):71–91, 2016.

[MM05a] K. W. Morton and D. F. Mayers. *Linear second order elliptic equations in two dimensions*, page 194–234. Cambridge University Press, 2 edition, 2005.

[MM05b] Keith W Morton and David Francis Mayers. *Numerical solution of partial differential equations: an introduction*. Cambridge university press, 2005.

[MPWN18] Andreas Mardt, Luca Pasquali, Hao Wu, and Frank Noé. Vampnets for deep learning of molecular kinetics. *Nature communications*, 9(1):5, 2018.

[MSCC19] Gal Mishne, Uri Shaham, Alexander Cloninger, and Israel Cohen. Diffusion nets. *Applied and Computational Harmonic Analysis*, 47(2):259–285, 2019.

[MSM18] Gr'egoire Montavon, Wojciech Samek, and Klaus-Robert Müller. *Explainable AI: Interpreting, explaining and visualizing deep learning*. Springer, 2018.

[MSP17] Mohammad M. Sultan and Vijay S Pande. tica-metadynamics: accelerating metadynamics by using kinetically selected collective variables. *Journal of chemical theory and computation*, 13(6):2440–2447, 2017.

[MSP+25] Mirage Modi, Shashank Sule, Jonathan Palumbo, Michael Rozowski, Mustapha Bouhrara, Wojciech Czaja, and Richard G Spencer. Input layer regularization and automated regularization hyperparameter tuning for myelin water estimation using deep learning. *arXiv preprint arXiv:2501.18074*, 2025.

[MVE06] Luca Maragliano and Eric Vanden-Eijnden. A temperature accelerated method for sampling free energy and determining reaction pathways in rare events simulations. *Chemical physics letters*, 426(1-3):168–175, 2006.

[MWA+94] Alex MacKay, Kevin Whittall, John Adler, David Li, Donald Paty, and Dale Graeb. In vivo visualization of myelin water in brain by magnetic resonance. *Magnetic Resonance in Medicine*, 31(6):673–677, 1994.

[MZK+21] Kai R. Müller, Wenkai Zhang, Madhusudhanan Krishnamoorthy, Cecilia Clementi, and Frank Noé. Estimating the diffusion tensor of collective variables from short molecular dynamics trajectories. *The Journal of Chemical Physics*, 154(23):234111, 2021.

[NCL+23] Neel Nanda, Lawrence Chan, Tom Lieberum, Jess Smith, and Jacob Steinhardt. Progress measures for grokking via mechanistic interpretability. *arXiv preprint arXiv:2301.05217*, 2023.

[NKPH+14] Feliks Nuske, Bettina G Keller, Guillermo Pérez-Hernández, Antonia SJS Mey, and Frank Noé. Variational approach to molecular kinetics. *Journal of chemical theory and computation*, 10(4):1739–1752, 2014.

[NLCK06] Boaz Nadler, Stéphane Lafon, Ronald R Coifman, and Ioannis G Kevrekidis. Diffusion maps, spectral clustering and reaction coordinates of dynamical systems. *Applied and Computational Harmonic Analysis*, 21(1):113–127, 2006.

[NN13] Frank Noé and Feliks Nüske. A variational approach to modeling slow processes in stochastic dynamical systems. *Multiscale Modeling & Simulation*, 11(2):635–655, 2013.

[NTM+22] Neha, Vikas Tiwari, Soumya Mondal, Nisha Kumari, and Tarak Karmakar. Collective variables for crystallization simulations–from early developments to recent advances. *ACS omega*, 8(1):127–146, 2022.

[NTS03] M Thamban Nair, Ulrich Tautenhahn, and E Schock. Morozov's discrepancy principle under general source conditions. *Zeitschrift für Analysis und ihre Anwendungen*, 22(1):199–214, 2003.

[OA23] Koji Ooka and Munehito Arai. Accurate prediction of protein folding mechanisms by simple structure-based statistical mechanical models. *Nature Communications*, 14(1):6338, 2023.

[OHL+07] Jiwon Oh, Eun T Han, Michael C Lee, Sarah J Nelson, and Daniel Pelletier. Multicomponent t2 relaxation and magnetization transfer in the normal brain and in cerebral gliomas. *Journal of Neuroimaging*, 17(4):273–280, 2007.

[PC14] Jordane Preto and Cecilia Clementi. Fast recovery of free energy landscapes via diffusion-map-directed molecular dynamics. *Physical Chemistry Chemical Physics*, 16(36):19181–19191, 2014.

[PHPG+13] Guillermo Pérez-Hernández, Fabian Paul, Toni Giorgino, Gianni De Fabritiis, and Frank Noé. Identification of slow molecular order parameters for markov model construction. *The Journal of Chemical Physics*, 139(1):015102, 2013.

[PJM13] Dominique Perraul-Joncas and Marina Meila. Non-linear dimensionality reduction: Riemannian metric estimation and the problem of geometric discovery. *arXiv preprint arXiv:1305.7255*, 2013.

[PNV19] Sanjib Paul, Nisanth N Nair, and Harish Vashisth. Phase space and collective variable based simulation methods for studies of rare events. *Molecular Simulation*, 45(14-15):1273–1284, 2019.

[Qui86] J Ross Quinlan. Induction of decision trees. *Machine learning*, 1:81–106, 1986.

[RA07] P. R. Romero and G. E. Arnold. Essential dynamics for the characterization of protein conformational ensembles. *Current Opinion in Structural Biology*, 17(2):169–174, 2007.

[RBWT18] João Marcelo Lamim Ribeiro, Pablo Bravo, Yihang Wang, and Pratyush Tiwary. Reweighted autoencoded variational bayes for enhanced sampling (rave). *The Journal of chemical physics*, 149(7), 2018.

[RCS23] Eitan Rosen, Xiuyuan Cheng, and Yoel Shkolnisky. G-invariant diffusion maps. *arXiv preprint arXiv:2306.07350*, 2023.

[RCV23] Jakub Rydzewski, Ming Chen, and Omar Valsson. Manifold learning in atomistic simulations: A conceptual review. *arXiv preprint arXiv:2303.08486*, 2023.

[RPB+22] Michael Rozowski, Jonathan Palumbo, Jay Bisen, Chuan Bi, Mustapha Bouhrara, Wojciech Czaja, and Richard G Spencer. Input layer regularization for magnetic resonance relaxometry biexponential parameter estimation. *Magnetic Resonance in Chemistry*, 60(11. Special issue rapid communication.):1076–1086, 2022.

[RPK19] 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.

[RPS+14] Ashish Raj, Sneha Pandya, Xiaobo Shen, Eve LoCastro, Thanh D Nguyen, and Susan A Gauthier. Multi-compartment t2 relaxometry using a spatially constrained multigaussian model. *PLoS One*, 9(6):e98391, 2014.

[RSG16] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. Why should I trust you? Explaining the predictions of any classifier. *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, pages 1135–1144, 2016.

[Ryd24] Jakub Rydzewski. Spectral map for slow collective variables, markovian dynamics, and transition state ensembles. *Journal of Chemical Theory and Computation*, 20(18):7775–7784, 2024.

[RZMC11] Mary A. Rohrdanz, Wenwei Zheng, Mauro Maggioni, and Cecilia Clementi. Determination of reaction coordinates via locally scaled diffusion map. *The Journal of Chemical Physics*, 134(12):124116, 2011.

[SB20] Richard G Spencer and Chuan Bi. A tutorial introduction to inverse problems in magnetic resonance. *NMR in Biomedicine*, 33(12):e4315, 2020.

[SC08] Amit Singer and Ronald R Coifman. Non-linear independent component analysis with diffusion maps. *Applied and Computational Harmonic Analysis*, 25(2):226–239, 2008.

[SC25] Shashank Sule and Wojciech Czaja. Neumann eigenmaps for landmark embedding. In *2025 International Conference on Sampling Theory and Applications (SampTA)*, pages 1–7. IEEE, 2025.

[SCC+22] Hong Song, Lei Chen, Yutao Cui, Qiang Li, Qi Wang, Jingfan Fan, Jian Yang, and Le Zhang. Denoising of mr and ct images using cascaded multi-supervision convolutional neural networks with progressive training. *Neurocomputing*, 469:354–365, 2022.

[SDDSVD98] Jan Sijbers, Arnold J Den Dekker, Paul Scheunders, and Dirk Van Dyck. Maximum-likelihood estimation of rician distribution parameters. *IEEE Transactions on Medical Imaging*, 17(3):357–361, 1998.

[SDWMG15] Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *International conference on machine learning*, pages 2256–2265. PMLR, 2015.

[SEC25] Shashank Sule, Luke Evans, and Maria Cameron. Sharp error estimates for target measure diffusion maps with applications to the committor problem. *Applied and Computational Harmonic Analysis*, page 101803, 2025.

[SEKC09] Amit Singer, Radek Erban, Ioannis G Kevrekidis, and Ronald R Coifman. Detecting intrinsic slow variables in stochastic dynamical systems by anisotropic diffusion maps. *Proceedings of the National Academy of Sciences*, 106(38):16090–16095, 2009.

[SG21] Steffen Schultze and Helmut Grubmüller. Time-lagged independent component analysis of random walks and protein dynamics. *Journal of Chemical Theory and Computation*, 17(9):5766–5776, 2021.

[SGG+09] Otmar Scherzer, Markus Grasmair, Harald Grossauer, Markus Haltmeier, and Frank Lenzen. *Variational methods in imaging*, volume 167. Springer, 2009.

[Sha17] U. Sharma. *Coarse-graining of Fokker-Planck equations*. PhD thesis, Technische Universiteit Eindhoven, 2017.

[SHW21] Vıctor Garcia Satorras, Emiel Hoogeboom, and Max Welling. E (n) equivariant graph neural networks. In *International conference on machine learning*, pages 9323–9332. PMLR, 2021.

[Sin06] Amit Singer. From graph to manifold laplacian: The convergence rate. *Applied and Computational Harmonic Analysis*, 21(1):128–134, 2006.

[SK11] Vojtěch Spiwok and Blanka Králová. Metadynamics in the conformational space nonlinearly dimensionally reduced by isomap. *The Journal of chemical physics*, 135(22), 2011.

[SMC] Shashank Sule, Arnav Mehta, and Maria K Cameron. Learning collective variables that preserve transition rates. *SIAM Multiscale Modeling and Simulation*.

[SMG14] A Saxe, J McClelland, and S Ganguli. Exact solutions to the nonlinear dynamics of learning in deep linear neural networks. In *Proceedings of the International Conference on Learning Represenatations*. International Conference on Learning Represenatations 2014, 2014.

[SMG19] Andrew M Saxe, James L McClelland, and Surya Ganguli. A mathematical theory of semantic development in deep neural networks. *Proceedings of the National Academy of Sciences*, 116(23):11537–11546, 2019.

[SMH23] Subarna Sasmal, Martin McCullagh, and Glen M Hocky. Reaction coordinates for conformational transitions using linear discriminant analysis on positions. *Journal of chemical theory and computation*, 19(14):4427–4435, 2023.

[SP13] Christian R. Schwantes and Vijay S. Pande. Improvements in markov state model construction reveal many non-native interactions in the folding of ntl9. *Journal of Chemical Theory and Computation*, 9(4):2000–2009, 2013.

[SP15] Christian R Schwantes and Vijay S Pande. Modeling molecular kinetics with tica and the kernel trick. *Journal of chemical theory and computation*, 11(2):600–608, 2015.

[SPTT18] Zachary Smith, Debabrata Pramanik, Sun-Ting Tsai, and Pratyush Tiwary. Multi-dimensional spectral gap optimization of order parameters (sgoop) through conditional probability factorization. *The Journal of Chemical Physics*, 149(23), 2018.

[SSAN16] Elham Saniei, Saeed Setayeshi, Mohammad Esmaeil Akbari, and Mitra Navid. Parameter estimation of breast tumour using dynamic neural network from thermal pattern. *Journal of Advanced Research*, 7(6):1045–1055, 2016.

[SSC25] Shashank Sule, Richard G Spencer, and Wojciech Czaja. On the limits of neural network explainability via descrambling. *Applied and Computational Harmonic Analysis*, 79:101793, 2025.

[SV07] Daniel W Stroock and SR Srinivasa Varadhan. *Multidimensional diffusion processes*. Springer, 2007.

[SVZ14] K Simonyan, A Vedaldi, and A Zisserman. Deep inside convolutional networks: visualising image classification models and saliency maps. In *Proceedings of the International Conference on Learning Representations (ICLR)*. ICLR, 2014.

[SW12] Amit Singer and H-T Wu. Vector diffusion maps and the connection laplacian. *Communications on pure and applied mathematics*, 65(8):1067–1144, 2012.

[SWW07] Oleg G Smolyanov, Heinrich v Weizsäcker, and Olaf Wittich. Chernoff's theorem and discrete time approximations of brownian motion on manifolds. *Potential Analysis*, 26(1):1–29, 2007.

[TB20] Prasun Chandra Tripathi and Soumen Bag. Cnn-dmri: a convolutional neural network for denoising of magnetic resonance images. *Pattern Recognition Letters*, 135:57–63, 2020.

[TB22] Lloyd N Trefethen and David Bau. *Numerical linear algebra*. SIAM, 2022.

[The15] Sergios Theodoridis. *Machine learning: a Bayesian and optimization perspective*. Academic press, 2015.

[Tre00] Lloyd N Trefethen. *Spectral methods in MATLAB*. SIAM, 2000.

[VEE06] Eric Vanden-Eijnden and Weinan E. Towards a theory of transition paths. *Journal of statistical physics*, 123(3):503–523, 2006.

[VHSF+21] Soledad Villar, David W Hogg, Kate Storey-Fisher, Weichi Yao, and Ben Blum-Smith. Scalars are universal: Equivariant machine learning, structured like classical physics. *Advances in Neural Information Processing Systems*, 34:28848–28863, 2021.

[VLBB08] Ulrike Von Luxburg, Mikhail Belkin, and Olivier Bousquet. Consistency of spectral clustering. *The Annals of Statistics*, pages 555–586, 2008.

[VLL+06] Irene M Vavasour, Cornelia Laule, David KB Li, Anthony L Traboulsee, and Alex L MacKay. Is the magnetization transfer ratio a marker for myelin in multiple sclerosis? *Journal of Magnetic Resonance Imaging*, 24(6):809–814, 2006.

[VTP16] Omar Valsson, Pratyush Tiwary, and Michele Parrinello. Enhancing important fluctuations: Rare events and metadynamics from a conceptual viewpoint. *Annual review of physical chemistry*, 67(1):159–184, 2016.

[VW19] Santosh Vempala and Andre Wibisono. Rapid convergence of the unadjusted langevin algorithm: Isoperimetry suffices. *Advances in neural information processing systems*, 32, 2019.

[VWD22] Bodhi P Vani, Jonathan Weare, and Aaron Dinner. Computing transition path theory quantities with trajectory stratification. *The Journal of Chemical Physics*, 2022.

[VWKA24] Serge Didenko Vasylechko, Simon K Warfield, Sila Kurugol, and Onur Afacan. Improved myelin water fraction mapping with deep neural networks using synthetically generated 3d data. *Medical image analysis*, 91:102966, 2024.

[Wan15] Xu Wang. Spectral convergence rate of graph laplacian. *arXiv preprint arXiv:1510.08110*, 2015.

[WAS20] Robert J Webber, David Aristoff, and Gideon Simpson. A splitting method to reduce mcmc variance. *arXiv preprint arXiv:2011.13899*, 2020.

[WDD+08] Demian Wassermann, Maxime Descoteaux, Rachid Deriche, et al. Diffusion maps clustering for magnetic resonance q-ball imaging segmentation. *International Journal of Biomedical Imaging*, 2008, 2008.

[WHT22] Yihang Wang, Lukas Herron, and Pratyush Tiwary. From data to noise to data for mixing physics across temperatures with generative artificial intelligence. *Proceedings of the National Academy of Sciences*, 119(32):e2203656119, 2022.

[WKR15] Matthew O Williams, Ioannis G Kevrekidis, and Clarence W Rowley. A data-driven approximation of the koopman operator: Extending dynamic mode decomposition. *Journal of Nonlinear Science*, 25:1307–1346, 2015.

[WMZT24] Ruiyu Wang, Shams Mehdi, Ziyue Zou, and Pratyush Tiwary. Is the local ion density sufficient to drive nacl nucleation from the melt and aqueous solution? *The Journal of Physical Chemistry B*, 128(4):1012–1021, 2024.

[WN11] Simon Wiesler and Hermann Ney. A convergence analysis of log-linear training. *Advances in Neural Information Processing Systems*, 24, 2011.

[WN18] Christoph Wehmeyer and Frank Noé. Time-lagged autoencoders: Deep learning of slow collective variables for molecular kinetics. *The Journal of chemical physics*, 148(24), 2018.

[WQB+24] Dedi Wang, Yunrui Qiu, Eric R Beyerle, Xuhui Huang, and Pratyush Tiwary. Information bottleneck approach for markov model construction. *Journal of chemical theory and computation*, 20(12):5352–5367, 2024.

[WR21] Caroline L Wormell and Sebastian Reich. Spectral convergence of diffusion maps: Improved error bounds and an alternative normalization. *SIAM Journal on Numerical Analysis*, 59(3):1687–1734, 2021.

[WSJK18] Steven G Worswick, James A Spencer, Gunnar Jeschke, and Ilya Kuprov. Deep neural network processing of deer data. *Science Advances*, 4(8):eaat5218, 2018.

[WT21] Dedi Wang and Pratyush Tiwary. State predictive information bottleneck. *The Journal of Chemical Physics*, 154(13), 2021.

[WVE04] E[1] Weinan and Eric Vanden-Eijnden. Metastability, conformation dynamics, and transition pathways in complex systems. In *Multiscale modelling and simulation*, pages 35–68. Springer, 2004.

[YCN+15] Jason Yosinski, Jeff Clune, Anh Nguyen, Thomas Fuchs, and Hod Lipson. Understanding neural networks through deep visualization. *Published in the Deep Learning Workshop, 31 st International Conference on Machine Learning, Lille, France.*, 2015.

[YCRP+21] Thomas Yu, Erick Jorge Canales-Rodríguez, Marco Pizzolato, Gian Franco Piredda, Tom Hilbert, Elda Fischi-Gomez, Matthias Weigel, Muhamed Barakovic, Meritxell Bach Cuadra, Cristina Granziera, et al. Model-informed machine learning for multicomponent t2 relaxometry. *Medical image analysis*, 69:101940, 2021.

[YEH+20] Chengxi Ye, Matthew Evanusa, Hua He, Anton Mitrokhin, Tom Goldstein, James A Yorke, Cornelia Fermuller, and Yiannis Aloimonos. Network deconvolution. In *International Conference on Learning Representations*, 2020.

[YSLC25] Jiaxin Yuan, Shashank Sule, Yeuk Yin Lam, and Maria Cameron. Learning collective variables that respect permutational symmetry. *The Journal of Chemical Physics*, 163(12):124101, 09 2025.

[YSZ+19] Yi Isaac Yang, Qiang Shao, Jun Zhang, Lijiang Yang, and Yi Qin Gao. Enhanced sampling in molecular dynamics. *The Journal of chemical physics*, 151(7), 2019.

[YWS14] Y. Yu, T. Wang, and R. J. Samworth. A useful variant of the Davis–Kahan theorem for statisticians. *Biometrika*, 102(2):315–323, 04 2014.

[ZCD+21] Ying Zheng, Haoyu Chen, Qingyang Duan, Lixiang Lin, Yiyang Shao, Wei Wang, Xin Wang, and Yuedong Xu. Leveraging domain knowledge for robust deep reinforcement learning in networking. In *IEEE INFOCOM 2021-IEEE Conference on Computer Communications*, pages 1–10. IEEE, 2021.

[ZHRB23] Mo Zhou, Jiequn Han, Manas Rachh, and Carlos Borges. A neural network warm-start approach for the inverse acoustic obstacle scattering problem. *Journal of Computational Physics*, 490:112341, 2023.

[ZHS16] Wei Zhang, Carsten Hartmann, and Christof Schütte. Effective dynamics along given reaction coordinates and reaction rate theory. *Faraday Discuss.*, 195, 10 2016.

[ZNP+19] Yue-Yu Zhang, Haiyang Niu, GiovanniMaria Piccini, Dan Mendels, and Michele Parrinello. Improving collective variables: The case of crystallization. *The Journal of chemical physics*, 150(9), 2019.

[ZRC13] Wenwei Zheng, Mary A. Rohrdanz, and Cecilia Clementi. Rapid exploration of configuration space with diffusion-map-directed molecular dynamics. *The Journal of Physical Chemistry B*, 117(42):12769–12776, 2013.

[ZS23] Wei Zhang and Christof Schütte. Understanding recent deep-learning techniques for identifying collective variables of molecular dynamics. *PAMM*, 23(4):e202300189, 2023.

[ZS24] Wei Zhang and Christof Schütte. On finding optimal collective variables for complex systems by minimizing the deviation between effective and full dynamics. *arXiv preprint arXiv:2405.02001*, 2024.

[ZVR+13] Wenwei Zheng, Attilio Vittorio Vargiu, Mary A Rohrdanz, Paolo Carloni, and Cecilia Clementi. Molecular recognition of dna by ligands: Roughness and complexity of the free energy profile. *The Journal of chemical physics*, 139(14), 2013.

[ZWL+22] Xin-Xiang Zhou, Xin-Yu Wang, En-Hui Liu, Lan Zhang, Hong-Xia Zhang, Xiu-Shi Zhang, Yue-Min Zhu, and Zi-Xiang Kuai. An unsupervised deep learning approach for dynamic-exponential intravoxel incoherent motion mri modeling and parameter estimation in the liver. *Journal of Magnetic Resonance Imaging*, 56(3):848–859, 2022.