UNIVERSITY OF CALIFORNIA

Los Angeles

Inverse Problems in Mathematical Imaging with Applications in Multiplicative Noise Removal, Remotely Sensed Atmospheric Wind Velocity Estimation, and Turbulent Fluid Simulation

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

by

Joel Ryan Barnett

2024

ABSTRACT OF THE DISSERTATION

Inverse Problems in Mathematical Imaging with Applications in Multiplicative Noise Removal, Remotely Sensed Atmospheric Wind Velocity Estimation, and Turbulent Fluid Simulation

by

Joel Ryan Barnett
Doctor of Philosophy in Mathematics
University of California, Los Angeles, 2024
Professor Andrea Bertozzi, Co-Chair
Professor Luminita Aura Vese, Co-Chair

This thesis considers three inverse problems originating in mathematical imaging, remote sensing and simulation. First, it addresses a challenging image processing task: recovering images corrupted by multiplicative noise. Motivated by the success of multiscale hierarchical decomposition methods (MHDM) in image processing, a variety of both classical and new multiplicative noise removing models are adapted to the MHDM form. On the basis of previous work, tight and refined extensions of the multiplicative MHDM process are proposed. Existence and uniqueness of solutions for the proposed models is discussed, in addition to convergence properties. Moreover, the work introduces a discrepancy principle stopping criterion which prevents recovering excess noise in the multiscale reconstruction. The validity of all the proposed models is qualitatively and quantitatively evaluated through comprehensive numerical experiments and comparisons across several images degraded by multiplicative noise. By construction, these multiplicative multiscale hierarchical decomposition methods have the added benefit of recovering many scales of an image, which can provide features of interest beyond image denoising.

Second, this work considers an applied mathematical imaging problem in remote sensing. Accurate estimation of atmospheric wind velocity plays an important role in weather forecasting, flight safety assessment and cyclone tracking. Atmospheric data captured by infrared and microwave satellite instruments provide global coverage for weather analysis. Extracting wind velocity fields from such data has traditionally been done through feature tracking, correlation/matching or optical flow means from computer vision. However, these recover either sparse velocity estimates, oversmooth details or are designed for quasi-rigid body motions which over-penalize vorticity and divergence within the often turbulent weather systems. This thesis proposes a texture based optical flow procedure tailored for water vapor data. The method implements an $L1$ data term and total variation regularizer and employs a structure-texture image decomposition to identify key features which improve recoveries and help preserve the salient vorticity and divergence structures.

Based on the success of texture-features at improving the flow estimation, this procedure is extended to a multi-fidelity scheme which incorporates additional image features into the optical flow scheme. Both flow estimation methods are tested on simulated over-ocean mesoscale convective systems and convective and extratropical cyclone datasets, each of which have accompanying ground truth wind velocities allowing quantitative comparisons of each method's performance against existing optical flow methods.

Third, the thesis considers a data-driven reduced-order modeling problem. It is well known that full-resolution simulations of turbulence is computationally expensive (and often intractable), motivating the development of reduced-order models for practical engineering and science applications. However, developing accurate reduced models for turbulent flow is impeded by the loss of unresolved scales which critically influence the full state. The Mori-Zwanzig formalism is a mathematically exact strategy for performing reduced-order modeling in which reduced-order variables (observables) are evolved by memory kernels and an orthogonal dynamics term. Recently, frameworks for extracting Mori-Zwanzig kernels from simulated data have shown promise for reduced turbulence models. However, these same works highlight the importance of choosing an appropriate reduced-order observables set, and how this choice can substantially affect the model's overall success. This thesis proposes a joint-learning process to discover an improved set of observables in tandem with extracting the Mori-Zwanzig memory kernels. The process is tested on simulated two-dimensional turbulence data, and results show that the discovered set of observables enhances the Mori-Zwanzig based turbulence model's capabilities at predicting turbulent structures and statistics within the resolved space, both in short and long trajectories.

The dissertation of Joel Ryan Barnett is approved.

Igor Yanovsky

Marcus Leigh Roper

Stanley J. Osher

Luminita Aura Vese, Committee Co-Chair

Andrea Bertozzi, Committee Co-Chair

University of California, Los Angeles

2024

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGMENTS

I would like to thank all my committee members who have remained a constant source of encouragement throughout my time at UCLA. My advisors, Professors Andrea Bertozzi and Luminita Vese, have both continually seen and cultivated potential beyond what my initial confidence could allow. I am grateful for their tireless encouragement and optimism in addition to academic direction and support. I am thankful for Dr. Igor Yanovsky, who volunteered to help mentor and advise a green graduate student and provided beneficial industry perspectives throughout.

I am also extremely grateful to Professor Stanley Osher for encouraging the engineering side of my mathematical pursuits and for extending my interest in the imaging space. I am very grateful to Professor Marcus Roper for his cheerful encouragement, especially throughout coursework early on in the program. Professor Hayden Schaeffer provided invaluable mentoring throughout my research and I am grateful for the good talks through the hallways of the Mathematical Sciences building.

Additionally, I would like to acknowledge my mentors from Los Alamos National Laboratory (LANL): Drs. Yifeng Tian, Yen Ting Lin and Daniel Livescu. They provided an incredible experience during LANL's 2023 Applied Machine Learning Summer Fellowship, and put in effort above and beyond, continuing to be a readily available resource long after the internship ended. They have helped me grow in new fields substantially. To all those who provided encouragement and resources while job-searching as well as invaluable references, thank you!

I would also like to thank my many good friends from UCLA who have made my time in the department so much more pleasant: Elisa Negrini, Giulio Farolfi, Dominic Yang, Benjamin Bowman, Bon-Soon Lin, Xia Li, Kaiyan Peng, Jacob Moorman, Raymond Chu,

Jerry Luo, Jason Brown, Yotam Yaniv, Grace Li, Peter Xu, office mates Josheph Breen, Thierry Laurens, James Chapman, Lisang Ding, Bohan Chen, Xinzhe Zuo and housemates Patrick Hiatt and Alexander Kastner. In addition, thank you to members of the unofficial Math Department IM Ultimate team, the *Whitney Disc Trick*, who provided a needed break from the office to go throw a disc.

Additionally, my friends from outside the UCLA mathematics department have made life in Los Angeles so much sweeter. Your support and encouragement has been distinctly impactful. I am especially grateful for those from Cornerstone Church West Los Angeles, members of the Cahoon, Darlington, and B-T-S community groups, CWLA softball team, CWLA weekly workouts group, as well as those involved in the Navigators ministry during my time in Los Angeles.

Finally, I want to thank most fervently my family. It is because of your endless support that I am able to be where I am today. Thank you Mom and Dad, I am so grateful for years of care, encouragement and wise direction. To T.C., my siblings, grandparents and close friends from Santa Rosa, CA, whom I love dearly, I am grateful for the practical and emotional cheer-leading throughout the entirety of this endeavor. Without all of your support, I would not have fared nearly so well, especially during the isolation in the COVID years.

The research in Chapter 2 is a version of [BLR23] and is joint work with Wen Li, Elena Resmerita and Luminita Vese. Luminita Vese oversaw the project. Wen Li and Elena Resmerita introduced a multiscale scheme for additive-noise schemes in a prior work with Luminita Vese. Elena Resmerita contributed to convergence theory of multiscale hierarchical decomposition iterates. Wen Li assisted with background model research and consulted on coding implementation. I developed well posedness theory for the multiscale hierarchical techniques, formulated the numerical treatment and implemented and ran experiments for all the methods within this chapter. We would also like to thank Tobias Wolf (University of Klagenfurt) for proofreading assistance on this project.

Chapter 3 is a version of [BBV23b, BBV23a]. This is work done with Andrea Bertozzi, Luminita Vese and Dr. Igor Yanovsky. Profs. Andrea Bertozzi and Luminita Vese advised on the project, and Igor Yanovsky provided the motivation along with access to datasets for the work. I developed the Texture-Flow and Multi-Fidelity techniques and performed all coding and experiments.

Chapter 4 is a collaboration with Yifeng Tian, Yen Ting Lin and Daniel Livescu. The project was started as a 2023 LANL Applied Machine Learning Summer Research Fellowship topic and continued after the program ended. Dr. Livescu and Dr. Lin advised on the project and Dr. Tian provided mentoring and starter code. I developed the observable learning process and performed all experiments and implementations throughout.

The work done in Chapters 2 and 3 was supported by the National Science Foundation (DMS 2012868). We would like to thank the JPL Observing System Simulation Experiment group and the University of Arizona for providing the nature-run datasets used for experiments within Chapter 3. Research presented in Chapter 4 was supported by the Laboratory Directed Research and Development program of Los Alamos National Laboratory under project number 20220104DR. Funding was additionally provided by the 2023 LANL Applied Machine Learning Summer Research Fellowship. The work is released under LA-UR-24-24745.

VITA

2016 A.S. (Mathematics) and A.S. (Physics), Santa Rosa Junior College, Santa Rosa, California.

2018 B.S. (Applied Mathematics), University of California, Davis, California.

2020 M.A. (Mathematics), University of California, Los Angeles, California.

2019–2023 Teaching Assistant, Mathematics Department, University of California, Los Angeles, California.

2020–2024 Research Assistant, Mathematics Department, University of California, Los Angeles, California.

2023 Applied Machine Learning Fellow, Los Alamos National Laboratory, Los Alamos, New Mexico.

PUBLICATIONS

Joel Barnett, Wen Li, Elena Resmerita and Luminita Vese. "Multiscale hierarchical decomposition methods for images corrupted by multiplicative noise." *Submitted Nov, 2023.*

Joel Barnett, Andrea Bertozzi, Luminita A. Vese and Igor Yanovsky."Incorporating Texture Features Into Optical Flow for Atmospheric Wind Velocity Estimation." *IGARSS 2023-2023 IEEE International Geoscience and Remote Sensing Symposium*, Pasadena, CA, July 2023, pp. 3784-3787. IEEE, 2023.

Joel Barnett, Andrea Bertozzi, Luminita A. Vese and Igor Yanovsky.“Texture-based Optical Flow for Wind Velocity Estimation from Water Vapor Data.” *Ocean Sensing and Monitoring XV*, W. Hou and L. J. Mullen, Eds., International Society for Optics and Photonics, Vol. 12543. pp. 164-175. SPIE, 2023.

CHAPTER 1

Introduction

This thesis addresses three separate problems within the mathematical imaging, remote sensing, and fluid simulation fields. First, we discuss an image denoising problem for a particular form of noise present in radar and medical imaging. Then, we discuss a remote sensing task pertinent to atmospheric sciences. Lastly, we address a reduced order modeling problem for simulation. Each involves solving an inverse problem (often ill-posed), and each has a clear application within the science community.

In Chapter 2, we address a classical imaging task: denoising a corrupted signal. While there is an abundance of research for handling additive noise, much less work has been done to remove multiplicative noise. Multiplicative or impulse noise is a common interference found, for example, within radar (especially Synthetic Aperture Radar (SAR)) imaging and ultrasound imaging systems. It is formed by the signal experiencing a corruption proportional to itself, often the result of constructive or destructive interference from the diffuse reflection of wavefronts off of the sensed terrain. The result is a speckle pattern degrading the image which is challenging to remove.

We initially discuss the existing approaches for multiplicative denoising, and identify several baseline models to use for comparison. Inspired by multiscale techniques for additive noise, we introduce a general multiscale hierarchical decomposition method (MHDM) suitable for multiplicative interference. The goal of the MHDM approach is to not only provide a multiplicative denoising model, but also to develop a technique which can be flexibly adapted to a variety of existing or yet-to-come models for such types of corruption. An MHDM approach has several benefits. By construction, the process requires little to no parameter tuning as is commonly needed for variational image denoising methods. Moreover, because the image is recovered in successively refined scales, one can set a desired level of detail to retain, or extract only scales of interest, and thus the method can be extended to applications beyond image denoising. Now, without appropriate stopping conditions, multiscale approaches can recover excess noise. We address this by providing a suitable stopping criterion based on the image noise level. We discuss the general behavior of the reconstructed scales along with their convergence properties for general fidelity and regularization functionals, and additionally consider the well-posedness of several specific models. Tight and refined extensions of the standard multiscale approach allow for stronger convergence properties, and can recover slightly more textured features within an image. Detailed approaches to numerically solving the induced minimization problems are discussed and we conduct extensive numerical experiments for comparison.

In Chapter 3, we consider an applied imaging problem: estimating the wind velocity of a region using remote sensing data. This problem is relevant for atmospheric modeling and weather prediction. Computing accurate wind estimates with physical instruments is expensive and infeasible if one needs detailed spatial coverage at a variety of altitudes and over large regions. A natural approach to this problem would leverage remote sensing data which can provide high-fidelity coverage over a wide swaths of the globe. Our approach makes use of data recording the water vapor levels within the atmosphere at a specified pressure level (altitude). This vapor concentration is a quantity which can be determined by microwave and infrared enabled remote sensing satellites. The data itself comes from direct numerical simulations from a comprehensive weather forecasting model, which provides not only the water vapor data needed, but also accompanying wind velocity information so that we have a basis for comparison.

We discuss the optical flow problem of determining the displacement map between two consecutive images which record a dynamic scene. The displacement map, known as the flow, can be seen as the velocity field transporting one image into the next. Classically, this flow field has been determined through correlation or block-matching means, which produce spatially sparse flow estimates. As opposed to these *local* methods, dense estimates can be found through *global* optical flow methods found in computer vision literature. We propose modifying the traditional *optical flow constraint equation* used within these dense flow methods, and including additional data-features which aid flow extraction. These features are similar to choosing a hierarchical level in the multiscale decomposition method discussed in Chapter 2. We design two techniques to perform improved wind velocity estimations by incorporating these features. The first, *texture-flow*, defines a multi-step process which initially tracks the movement of the textured portion of the dataset independently from the full data. The flow can then be refined by considering the full dataset. Second, we propose a *multi-fidelity flow* process which incorporates many features into the optical flow constraint equation simultaneously. Finally, a feasible numerical approach to handle the more complicated fidelity terms is developed, and its construction is agile enabling it to easily be modified to accept additional features as needed.

In Chapter 4, we consider a reduced order modeling problem for dynamical systems. In particular, we consider turbulent fluid simulation where we want to evolve forward a reduced-order representation of the system's state. Reduced order models are often necessary for fluid simulations because fully resolving all the scales present in turbulent flow is infeasible for even mildly large domains.

Our approach to forming a reduced order model is inspired by the recent works of [LTL21, TLA21, LTP23] which leverage theory from the Mori-Zwanzig formalism to develop a data-driven process for turbulence modeling. The Mori-Zwanzig formalism is a mathematically exact description of reduced order dynamics, but it depends on a choice of observables (reduced order functions of the full state) which are to be evolved. The authors in [TLA21] found empirically that this choice can dramatically influence the quality of a model's success, yet there is no systematic way of choosing these features optimally.

This is the motivation for a data-driven observable discovery method. Selecting the observable set is an inverse problem, whose corresponding forward problem is the forward simulation of the system using the discovered observables. We discuss the popular Koopman approach to reduced order modeling before providing an overview of the Mori-Zwanzig formalism, deriving its generalized Langevin dynamics equation and Markov and memory kernel relationships. Importantly, we establish the generalized fluctuation-dissipation relation which relates memory kernels to the dynamics orthogonal to the resolved space. We also discuss a discrete Mori-Zwanzig formulation which can be adapted for data-driven approaches to learn the Markov and memory kernels. Given a numerical approach to Mori-Zwanzig memory based forward simulation, we consider the inverse problem of determining observables which will improve the accuracy of the forward problem.

We propose a convolutional neural network to learn the observable function, which the forward process will evolve, and jointly train the network and the memory kernels via an alternating minimization scheme. We test the process on two-dimensional, fully resolved direct numerical simulations of turublent flow, and compare the output to coarse-grain computational fluid dynamics simulations as well as Mori-Zwanzig based approaches using a predefined observable set.

CHAPTER 2

Multiscale Hierarchical Decomposition Methods for Images Corrupted by Multiplicative Noise[1]

2.1 Introduction to Image Denoising

Signal corruption is typically unavoidable during data collection. Blur can be introduced during image formation by lens misalignment or jitter, while noise is accumulated from random affects on the sensor, during image transmission, storage or compression. Born from this reality are a myriad of image processing techniques aimed at cleaning such corrupted signals.

The challenge of eliminating noise from an image (or signal) is characterized by a frequently encountered reality in mathematical disciplines: being easy to state but difficult to solve. Indeed, the problem of image denoising (with possible blur) can be simply formulated. Given a noisy signal f^δ,

$$f^\delta = Tz + n, \tag{2.1}$$

recover an approximation of the clean image $z : \Omega \to \mathbb{R}$, where $n \sim \mathcal{N}_\Omega$ represents some *added* noise on the domain Ω, and $T : L^2(\Omega) \to L^2(\Omega)$ is a bounded linear operator which applies blur. Here, Ω (typically $\subseteq \mathbb{R}^2$) determines the domain of the image, and δ is a parameter related to the noise level, which we will address in subsequent sections.

[1] This chapter is adapted from [BLR23], which is under review.

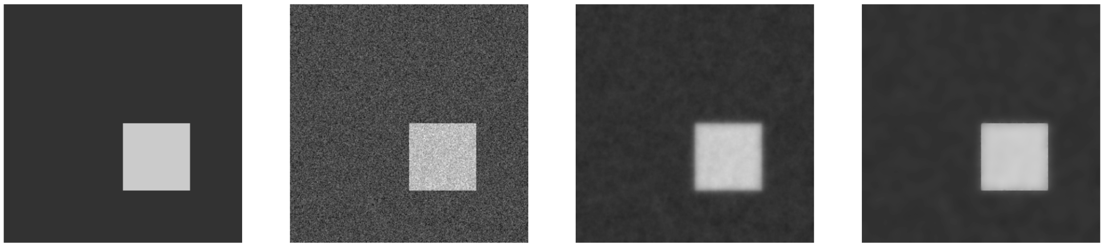


Figure 2.1: Tikhonov and Total Variation regularized denoising examples. From left to right: z, the clean image; f^δ, the noise degraded image; the Tikhonov recovery approximating z via (2.2); the recovery to (2.5) with total-variation regularization. The noise is Gaussian of mean 0 and standard deviation 0.1.

Recovering z from f^δ in (2.1) has been the subject of much research. The problem (2.1) is well posed if existence, uniqueness, and continuous dependence of the solution on the data holds (i.e., if T is injective, T^{-1} exists and is continuous). If these fail to hold, then inference of z is an ill-posed inverse problem. To solve (2.1), early methods involve filtering or least squares [Hun73], but suffer from over-smoothing or Gibbs phenomena and cannot recover sharp edges well. To overcome ill-posedness, regularity conditions on u can be added. For example, quadratic Tikhonov regularization attempts to determine z by finding a u near f^δ (in a least squares sense), with added spatial regularization $|\nabla u|^2$

$$u = \arg\min_u \lambda \|f^\delta - Tu\|^2 + \|\nabla u\|^2. \tag{2.2}$$

However, an image, u, recovered via (2.2) has smeared out edges, as show in Figure 2.1. This undesirable effect occurs because u is enforced to be continuous, and thus no brightness jumps—i.e. edges and texture—within an image are allowed.

2.1.1 Total Variation Image Denoising

The seminal paper [ROF92] by Rudin, Osher and Fatemi (ROF) approaches (2.1) by looking to recover an approximation, u, of z which lives in a sufficiently regular function space to remove noise, but still allows adequate variation for u to be a reasonable natural image. The proposed space is $BV(\Omega)$, the set of functions of bounded variation.

By $BV(\Omega)$, we mean functions for which $TV(u) < \infty$, where the *total variation*, $TV(\cdot)$, is defined as follows.

Definition 2.1.1. *The total variation of an image is defined by duality. For a function $u \in L^1_{loc}(\Omega)$, the total variation is given by*

$$TV(u) := \sup_{\phi} \left\{ -\int_{\Omega} u \ \mathrm{div}(\phi) dx : \phi \in C_c^{\infty}(\Omega; \mathbb{R}^N), |\phi(x)| \leq 1 \, \forall x \in \Omega \right\} \tag{2.3}$$

$BV(\Omega)$ is endowed with the norm $\|u\|_{BV(\Omega)} = \|u\|_{L^1(\Omega)} + TV(u)$.

Remark 2.1.2. *For a smooth function $u \in C^1(\Omega)$ (or one where $\nabla u \in L^1(\Omega)$),*

$$-\int_{\Omega} u \ \mathrm{div}(\phi) dx = \int_{\Omega} \phi \cdot \nabla u,$$

and the supremum over all $\|\phi\|_{L^{\infty}(\Omega)} \leq 1$ is $TV(u) = \int_{\Omega} |\nabla u|$.

We will abuse notation slightly and use $\int_{\Omega} |\nabla u|$ for $TV(u)$ formally, with ∇u indicating both the gradient (when it is defined) and the distribution associated with TV as needed.

Importantly, functions in $BV(\Omega)$ can be discontinuous. For example, on $[0, 1] \subset \mathbb{R}$ the step function $u(x) = 0$ for $x \in [0,1)$, $u(x) = 1$ for $x \in [1/2, 1]$ has total variation

$$\begin{aligned} TV(u) &= \sup_{\phi} - \int_0^1 u(x)\phi_x dx \\ &= \sup_{\phi} - \left(\int_0^{1/2} 0 \cdot \phi_x dx + \int_{1/2}^1 1 \cdot \phi_x dx \right) \\ &= \sup_{\phi} -\phi\Big|_{1/2}^{1} \\ &= \sup_{\phi} \left(\phi(1/2) - \phi(1)\right) = 1. \end{aligned}$$

The last equality holds recalling ϕ is compactly supported (i.e. $\phi(1) = 0$) and $|\phi(x)| \leq 1$ on $[0,1]$. Hence $u \in BV([0,1])$. This property of allowing discontinuous functions makes $BV(\Omega)$ an ideal candidate space for images which may contain edges and features where jumps in intensity are desirable.

Armed with a suitable function space, the ROF method enforces a recovered image, u, to be in $BV(\Omega)$ by introducing a regularizing functional $J(\cdot) = TV(\cdot)$. Referring to (2.1), the goal is to find the smoothest (or most regular) function u which fits the data $f^\delta \approx Tu + n$ appropriately (i.e., u may require jumps to *fit* well). Smoothness is dictated by the regularity condition $J(\cdot)$, and *smoothest* implies minimizing $J(u)$ subject to some fitting constraint. If some statistics for the noise are known, one can be more precise by what *fitting* the data means. Since $f^\delta - T(z) = n$, $\int_\Omega (f^\delta - Tz)^2 = \mathbb{V}[n]|\Omega|$ (assuming $\mathbb{E}[n] = 0$). For now, we assume $|\Omega| = 1$, and rewrite the objective as

$$\min_u J(u) \text{ subject to } \int (Tu - f^\delta)^2 = \sigma^2, \tag{2.4}$$

where σ is the standard deviation of n. By introducing a Lagrange multiplier, λ, the constrained problem can be rewritten as solving $\nabla_{u,\lambda}\mathcal{L}(u,\lambda) = 0$ where

$$\mathcal{L}(u,\lambda) = \lambda \int \left[(Tu - f^\delta)^2 - \sigma^2\right] + J(u).$$

Naturally, by ∂_u we mean first variation and $\partial_u \mathcal{L}(u, \lambda) = 0$ is a criticality condition for the unconstrained optimization

$$\min_u \lambda \int (Tu - f^\delta)^2 + J(u), \tag{2.5}$$

which is the form frequently encountered in imaging texts. Solving (2.5) with $J(\cdot) = TV(\,\cdot\,)$, one can clearly see the greater edge detail retained by the ROF method when compared with the Tikhonov recoveries (2.2) in Figure 2.2.

Forgetting the Lagrange multiplier approach momentarily, the format (2.5) can equivalently be seen as finding a balance between fitting the data closely—appropriately, $\int (Tu - f^\delta)^2$ is often referred to as the *data fidelity term*—while still remaining smooth enough, and the balance is controlled by the weighting λ. The choice of the weight can force recoveries u to match the data very closely (large λ) at the cost of retaining more noise, or obtain very cartoonish representations of f (small λ) which have high regularity. Many approaches to solve (2.5) exist, and a useful compilation of Euler-Lagrange, augmented Lagrangian, primal-dual, gradient descent, splitting and graph-cut approaches are given in [CCN09].

The observation that BV is a suitable function space for image recovery combined with variational methods spawned a plethora of research [RO94b, AV94, Ves01, Mey01, VL16] for total-variation based methods for image denoising and deblurring tasks. In [CMM00], a higher order penalty is proposed to combat the stair-case effect that can crop up with the piece-wise constant solutions (2.5) produces. This modified penalty balances a total variation term to preserve large edges, and an adaptive penalty ($\Delta u / \sqrt{|\nabla u|^2 + \epsilon}$ for example) which acts heavily in regions of small gradients (i.e. where smooth gradations should exist), but is reduced near large gradients (i.e. where true edges should be).

Additional works extended the ROF approach to image decomposition techniques [CL97, Mey01, LV05] and segmentation approaches [CV99, CV00, CV01, VC02, LV08] extending the popular Mumford-Shah Segmentation [MS85, MS89].

2.1.2 Multiscale Methods for Image Restoration

In 2004, Tadmor, Nezar and Vese introduced a multiscale image restoration method [TNV04] aimed at decomposing an image into features of several scales. The authors extended decomposition methods which typically broke an image, z, into pieces $z = u + v$, where u captured the essential features of z while v contained fine scale components of z (and potentially noise if we have a corrupted signal f^δ instead of z). The extension iterated on the prior process, recovering a representation of z by a sum of parts u_j, each which represented finer and finer scaled features of z. This decomposition strategy was leveraged for image denoising.

Starting with a signal, $f^\delta = Tz + n$, perturbed by additive noise, let λ_0 be a positive number and u_0 be a solution of the ROF problem (2.5) with $\lambda = \lambda_0$. Then, define the sequence of functions $\{u_k\} \subseteq BV(\Omega)$ recursively, where each function satisfies

$$u_k \in \arg\min_u \{\lambda_k \|Tu - v_{k-1}\|^2 + TV(u)\}, \tag{2.6}$$

with $\lambda_k = 2^k \lambda_0$ and $v_{k-1} = f^\delta - \sum_{j=0}^{k-1} Tu_j$, and thus $f^\delta = Tu_0 + Tu_1 + \cdots + Tu_{k-1} + v_{k-1}$. Equivalently, procedure (2.2.1) can be expressed as

$$\min_u \{\lambda_k \|T(u + x_{k-1}) - f^\delta\|^2 + TV(u)\},$$

for $k \geq 0$, where $x_{k-1} = \displaystyle\sum_{j=0}^{k-1} u_j$ and $x_{-1} = 0$ (see also [MNR19]). Overall, (2.2.1) can be thought of as fitting the residual v_{k-1} at each step.

The process compiles multiscale portions u_j, which form a hierarchical assembly of scales of the image f^δ. Taking many segments (i.e., $x_k = \sum_{j=0}^{k} u_j$ with k large) retains much of the detail of f^δ, but potentially contains noise. Retaining fewer scales (i.e., only small or moderate k) holds the main features of f^δ, but may miss finer features.

Convergence rates of (Tx_k) to Tz have been analyzed in [MNR19,LRV21,TNV04,TNV08], while improved versions of this multiscale process have been introduced and studied in

[MNR19, LRV21, TH16], including a stopping criteria for choosing k in [LRV21].

As an introduction to image denoising, we have briefly covered variational approaches to decluttering a signal in the presence of noise. Additionally, we introduced the multiscale hierachical decomposition process (2.2.1) and described how the process can be extended to denoising images. However, all the prior works are concerned with *additive* noise. The problem we wish to address is one of multiplicative noise, which we now turn to.

2.2 The Multiplicative Noise Problem

While the literature on denoising and deblurring images affected by additive noise is quite rich, the study of images corrupted by multiplicative noise still requires attention. This type of noise is inherent in radar, synthetic-aperture radar (SAR) and ultrasound images, cf. [Goo76,Bur78]. Our aim is to contribute to the topic not only by removing such noise from images, but also by proposing multiscale decomposition strategies for those images, similar to the contributions of [TNV04, TNV08, MNR19, LRV21] in case of additive corruption. More precisely, this work expands multiscale methods to the multiplicative-noise domain. While we apply such techniques to foundational methods from [RO94a, AA08, SO08], the resulting procedures could translate to other approaches aimed at multiplicative corruption. The importance of providing decompositions of medical images that separate the coarse and fine scales has been clearly highlighted in the case of image registration when significant levels of noise are involved [PLS06] (see also [MNR19, DLV23]). The reader is referred also to [BBF97, RB96] that address astronomical imaging and the need of recovering objects of very different sizes.

In a multiplicative noise problem, a clean image $z : \Omega \subseteq \mathbb{R}^2 \to \mathbb{R}$ is degraded by multiplicative noise $\eta : \Omega \to \mathbb{R}$ of mean 1 and (possibly) blurred by an ill-posed, linear, bounded operator $T : L^2(\Omega) \to L^2(\Omega)$, where Ω is a domain in $\mathbb{R}^2$. These together form the degraded data

$$f^\delta = (Tz) \cdot \eta, \tag{2.7}$$

where δ is a parameter relating to the size of the noise, referred to hereafter as the noise level. Throughout this work, we denote by $f \in L^2(\Omega)$ the *exact data* satisfying $Tz = f$.

A natural way to approach multiplicative noise is to manipulate the problem into a familiar form and apply existing techniques. At least in the pure denoising case, the logarithm transforms the problem $f^\delta = z \cdot \eta$ to an additive noise system $\log(f^\delta) = \log(z) + \log(\eta)$ for which a plethora of denoising methods exist. Indeed, this idea has been tried—the reader can find in [AA08] a discussion of a log-additive model. However, as pointed out in [AA08], blindly applying the log transform and employing algorithms for additive noise removal do not necessarily provide reasonable reconstructions, because the reconstruction means are often much smaller than those of the original images. This is due to the primary assumption of additive-noise methods, namely, zero-mean noise. However, the noise term in the transformed problem, $\log(\eta)$, is not necessarily zero-mean. To be more precise, by Jensen's inequality one has $\log(\mathbb{E}[\eta]) \geq \mathbb{E}[\log(\eta)]$. If a restoration of the system $\log(f^\delta) = \log(z) + \log(\eta)$ is found under the assumption that $\mathbb{E}[\log(\eta)] = 0$, then $\log(\mathbb{E}[\eta]) \geq 0$. Consequently, one has $\mathbb{E}[\eta] \geq 1$. Moreover, since $\log(\cdot)$ is a strictly concave function, Jensen's inequality is strict as soon as there is any noise, and we conclude $\mathbb{E}[\eta] > 1$, a contradiction to η having mean 1.

One can estimate this change in expectation by expanding $\log(\eta)$ about $\mathbb{E}[\eta]$ and taking expectation,

$$\mathbb{E}[\log(\eta)] \approx \log(\mathbb{E}[\eta]) - \frac{\mathbb{V}[\eta]}{2\mathbb{E}[\eta]^2},$$

whenever the distribution of η allows such quantities to be defined. Such restorations u will satisfy $f^\delta \approx u \cdot \eta$, implying $\mathbb{E}[u]\mathbb{E}[\eta] \approx \mathbb{E}[f^\delta]$, and necessarily $\mathbb{E}[u] \lessapprox \mathbb{E}[f^\delta] = \mathbb{E}[z]$, indicating a shift in the reconstruction mean $\mathbb{E}[u]$ from the image mean $\mathbb{E}[z]$. Therefore, designing novel algorithms which address directly the multiplicative noise is highly desirable.

2.2.1 Multiplicative Noise Models

Let us review below several variational models for restoring images corrupted by multiplicative noise. Rudin and Osher [RO94a] introduced in 1994 the following model for image denoising by imposing constraints on the mean and the variance of the noise,

$$\min_u \left\{ TV(u) + \lambda \int_\Omega \left(\frac{f^\delta}{u} - 1 \right)^2 \right\}, \tag{2.8}$$

where TV is the total variation and the minimization is performed in the space of bounded variation functions $BV(\Omega)$. Note that problem (2.8) is well-defined when $f^\delta \in L^\infty(\Omega)$ and $\inf_\Omega f^\delta > 0$ (see [Cha23, CL95]), and the unique minimizer u verifies $\inf_\Omega f^\delta \leq u \leq \sup_\Omega f^\delta$ a.e.

In 2008, Aubert and Aujol [AA08] proposed minimizing the energy

$$E(u) = TV(u) + \lambda \int_\Omega \left(\log(u) + \frac{f^\delta}{u} \right) \tag{2.9}$$

over the set $\{u \in BV(\Omega) : u > 0\}$ for denoising images degraded by a gamma-law speckle noise, with $f^\delta > 0$ as well. We will call this the AA model. The authors demonstrated that minimizers of (2.9) exist, however, the data fidelity term $\int_\Omega \left(\log(u) + \frac{f^\delta}{u} \right)$ is only strictly convex for $u \in (0, 2f^\delta)$ a.e., and not globally convex, so the minimization problem may not have a unique solution. They also noted that (2.9) can be extended to deblurring by involving an appropriate operator T,

$$E(u) = TV(u) + \lambda \int_\Omega \left(\log(Tu) + \frac{f^\delta}{Tu} \right).$$

Concurrently, Shi and Osher [SO08] introduced two multiplicative noise removal models.

The first one, which looks for

$$\arg\min_u \left\{ TV(u) + \lambda \int_\Omega \left(a\frac{f^\delta}{u} + \frac{b}{2}\left(\frac{f^\delta}{u}\right)^2 + c\log(u) \right) \right\}, \tag{2.10}$$

is a more general AA formulation which can be reduced to (2.9) by setting $b = 0$ and $a = c$. Again, the fidelity term is not globally convex. To address this, Shi and Osher let $w = \log(u)$ within the fidelity term of (2.10) and replaced $TV(u)$ with $TV(w)$, thus producing the second model which is convex (in w),

$$\arg\min_w \left\{ TV(w) + \lambda \int_\Omega \left(af^\delta e^{-w} + \frac{b}{2}(f^\delta)^2 e^{-2w} + cw \right) \right\}. \tag{2.11}$$

Having solved the now convex minimization problem for w, the true image estimate can be recovered by $u = e^w$. It is worth emphasizing that this partial transformation, which replaces $TV(u) = TV(e^w)$ with $TV(w)$, shifts the regularization to the logarithm of the image intensity. Consequently, the majority of smoothing is performed on image intensities near 0, while larger intensities are much less smoothed.

There have been several extensions of the works [AA08,SO08] which enforce convexity of the objective functional or tackle the efficient computation of the minimizers. For instance, the authors in [HNW09] studied (2.11) with $a = c = 1$ and $b = 0$, splitting the regularizing and fidelity terms, and adding a quadratic fitting term. A primary reason for the formulation in [HNW09] is the numerical efficiency in solving the minimization with an iterative alternating scheme.

Rather than transform $w = \log(u)$ to gain convexity, Dong and Zeng [DZ13] introduced an additional quadratic penalty term to the AA model

$$E_T(u) := \lambda TV(u) + \int_\Omega \left(\log(Tu) + \frac{f^\delta}{Tu} \right) + \alpha \int_\Omega \left(\sqrt{\frac{Tu}{f^\delta}} - 1 \right)^2, \tag{2.12}$$

thus ensuring convexity of the fidelity term for $\alpha \geq \frac{2\sqrt{6}}{9}$, as well as coercivity of the objective functional for the more general problem of deblurring. Hereafter, we refer to (2.12) as the DZ model. We mention also the interesting approach for multiplicative noise removal in [ST10], that uses a data fidelity which is typical for eliminating Poisson noise, and incorporates total variation or nonlocal means as regularizers. Additionally, in recent years there have been new approaches for removing multiplicative noise from images with or without blur. In [UCK17] and [JWM22], the authors made use of a fractional-order total variation and a total generalized variation penalty, respectively. The paper [WLL20] considered a convex scheme for structured multiplicative noise, [LF16] proposed an improved algorithm for the DZ model [DZ13], and [ZLG22] adapted Euler's elastica to the multiplicative noise problem. The reader is referred further to the introduction and the included references on the multiplicative noise topic in [DMR22]. There are also methods addressing denoising of color images degraded by speckle noise, which employ a total variation function adapted to red-green-blue (RGB) and hue-saturation-value (HSV) images (see [UCK17] and [WYN21], respectively). Studies on using convolutional neural networks for speckle noise removal can be found in [CDI23,NW22].

As mentioned above, our aim goes beyond the need of reconstructing images corrupted by multiplicative noise. That is, we focus also on obtaining decompositions of such images along several scales in a variational manner. For simplicity, we consider here the case of additive noise removal (see again from [TNV04, TNV08]), starting from the ROF model (2.5) reproduced below

$$\min_u \{\lambda_0 \|Tu - f^\delta\|^2 + TV(u)\}.$$

It is not easy to determine an appropriate parameter λ_0 to ensure that the cartoon (the main features of the image) is well extracted and also the image texture is well preserved while removing the noise. The advantage of the hierarchical process is that it enables separation of noise and image texture in increasingly refined scales by updating parameters, since the texture can be seen as cartoon at finer scales. As a result, the method provides an approximation of the original image f by a sum of image components, that is $f \approx \sum u_j$. As explained in [TNV04] (see also the more recent work [KRW23]), the approximation $\sum u_j$ obtained at the k-th hierarchical step involving the regularization parameter λ_k does not necessarily coincide with the one-step ROF minimization corresponding to the parameter λ_k. This shows the versatile role of the hierarchical decompositions versus single-step variational models. Motivated by stronger theoretical properties and better restoration effects, tight and refined versions of the multiscale hierarchical decomposition for denoising and deblurring images with additive noise were proposed in [LRV21] (see also [MNR19] for a more general tight version). Moreover, [LRV21] proposed for the first time the discrepancy principle for early stopping in the original, tight and refined MHDM.

2.2.2 Contributions and Organization

In this study, we introduce, test, and provide convergence properties for several hierarchical decomposition procedures designed to recover structured and textured images with multiple scales, when affected by multiplicative noise. Specifically, we propose four multiscale hierarchical decomposition methods for multiplicative noise removal, called SO MHDM, AA MHDM, AA-log MHDM and TNV-log MHDM. Thus, we first formulate a direct MHDM extension of the Shi-Osher model (2.11), which we abbreviate as SO MHDM, allowing us to adapt the summed-MHDM denoising techniques from [LRV21] to the new data fidelity setting, which is no longer quadratic (see Remark 2.3.3). Secondly, we proceed similarly with the AA model (AA MHDM, for short), and additionally introduce a penalty-modified adaptation of the AA model (2.9) (abbreviated as AA-log MHDM) which handles multiplicative gamma noise and blurring. Finally, we introduce a new variational model, that is a modified RO model, in which the TV penalty is replaced by TV(log). Then we derive its multiscale adaptation, thus yielding the so-called TNV-log MHDM. In order to promote more details in the reconstruction of the images perturbed by multiplicative noise, we introduce also tight and refined MHDM versions, and emphasize their effect on images with more texture.

We expect that the proposed multiplicative multiscale hierarchical decomposition methods can be extended to applications beyond image restoration, such as image fusion [CFX15, BD18], image representation [TA09], image registration and inverse problems [XAN14,MNR19].

The current work is organized as follows. In Section 2.3, we lay out the general strategy of hierarchical decomposition for multiplicative noise degraded images. We justify well-definedness, convergence properties and stopping rules of such schemes in Sections 2.4 and 2.5. Tight and refined modifications of the recovery schemes are analyzed in Section 2.6. We propose several numerical discretizations of the methods in Section 2.7, present detailed numerical results in Section 2.8, and point out the robustness of the proposed procedures, as well as the advantages of using one method or another, depending on the structure of the given image.

2.3 MHDM for Multiplicative Noise

In the multiplicative denoising problem, recovering the true image z in $BV(\Omega)$ amounts to solving the equation

$$f^\delta = (Tz) \cdot \eta$$

in a stable way, where z is assumed to contain features at different scales, as happens for example, in natural images. Our aim is to derive multiscale hierarchical decomposition methods for images affected by multiplicative noise, inspired by the idea developed in [TNV04, TNV08].

For images degraded by multiplicative noise, the only multiscale hierarchical decomposition we know about is the one from [TNV04] and [TNV08], which uses an increasing weighting parameter λ_k in the iteration-adapted Rudin-Osher model (2.8). Namely, one starts with

$$u_0 \in \arg\min_u \left\{ \lambda_0 \int_\Omega \left(\frac{f^\delta}{u} - 1 \right)^2 + TV(u) \right\},$$

where λ_0 is a positive parameter, and proceeds further with a similar minimization problem by doubling λ_0 and considering the new residual f^δ/u_0 which might contain more features of the original image, and so on. The minimizers u_k obtained iteratively as

$$u_k \in \arg\min_u \left\{ \lambda_k \int_\Omega \left(\frac{f^\delta}{u u_0 \cdots u_{k-1}} - 1 \right)^2 + TV(u) \right\}$$

for $k \geq 0$ (with $u_{-1} = 1$) are well-defined [CL95] and can be characterized as shown in [TNV08].

To set up a general multiscale hierarchical decomposition method for multiplicative noise, we work with a general data fidelity term in order to provide analysis in a unifying setting. Assume that $J : L^2(\Omega) \to [0, \infty]$ is a proper function and H is a non-negative data fitting term to be specified later. Let u_k be defined as follows:

$$u_k \in \arg\min_u E_k(u), \text{ with } \quad E_k(u) = \lambda_k H(f^\delta, T(u x_{k-1})) + J(u), \tag{2.13}$$

where $x_{k-1} = \displaystyle\prod_{j=0}^{k-1} u_j$, $x_{-1} = 1$ and $\lambda_{k+1} = 2\lambda_k$, if $k \geq 1$. For example, choosing the data fidelity

$$H(f^\delta, Tu) = \left\| \frac{f^\delta}{Tu} - 1 \right\|^2 \tag{2.14}$$

yields the Rudin-Osher variational method (2.8) for deblurring images, while

$$H(f^\delta, Tu) = \int_\Omega \left(\frac{f^\delta}{Tu} + \log(Tu) - \log(f^\delta) - 1 \right) \tag{2.15}$$

is the Itakura-Saito divergence that leads to the Aubert-Aujol model, with a general regularization term $J(\cdot)$. Note that this divergence is the Bregman distance

$$D_F(f^\delta, u) = F(f^\delta) - F(u) - \langle \nabla F(u), f^\delta - u \rangle,$$

with associated Burg entropy $F(u) = -\log(u)$. The resulting distance is non-negative due to the convexity of the entropy, $-\log(u)$. We will mention later more properties of H that will be helpful in the analysis regarding convergence of (Tx_k) to the exact data f. More properties of the multiplicative MHDM schemes introduced in this work, e.g. error estimates, will be shown when using penalty functionals J satisfying

$$J(uv) \leq J(u) + J(v), \qquad J(u) = J\left(\frac{1}{u}\right), \qquad J(1) = 0, \tag{2.16}$$

for any $u, v \in dom\, J = \{u \in L^2(\Omega) : J(u) < \infty\}$. An example of such a function is $J(u) = \varphi(\log(u))$, where φ is a seminorm (e.g., the total variation or the $*$-norm introduced in Sec. 2.6.2).

For the moment, we assume that minimizers u_k in (2.13) exist, and instead focus on the analysis of the multiscale decomposition method. Note that existence results will be pointed out for the particular denoising models we deal with in Section 2.4, while existence of solutions for the deblurring models (that is $T \neq I$) will be considered in more detail in our future research.

Remark 2.3.1. *Let us discuss the choice of the data-fidelity term in* (2.13)*. The first iterate u_0 is just a minimizer of $\lambda_0 H(f^\delta, Tu) + J(u)$. When searching for u_1, we can consider two possibilities. The first one consists of looking for u_1 such that the misfit between $T(u_1u_0)$ and f^δ becomes smaller than the one between Tu_0 and f^δ, and corresponds to the choice $\lambda_1 H(f^\delta, T(uu_0)) + J(u)$ used in* (2.13)*. Thus, the clean data f will be approximated by $T(u_0u_1 \dots u_k)$. The second possibility addresses the "new" data f^δ/Tu_0 and amounts to finding u_1 as a minimizer of $\lambda_1 H(f^\delta/Tu_0, Tu) + J(u)$. In this case, it is desired that the product $Tu_0Tu_1 \dots Tu_k$ converges in some sense to f. Our work focuses on the first version, since it is hoped that the product $u_0u_1 \dots u_k$ might approximate the true image z in both the denoising and deblurring case.*

Remark 2.3.2. *The Itakura-Saito divergence occuring in the AA-model has the interesting property of being scale invariant in the following sense:* $H(\lambda u, \lambda v) = H(u, v)$ *for any* $\lambda > 0$*. Therefore, in the denoising case, it holds that* $H(f^\delta, uu_0) = H(f^\delta/u_0, u)$*, showing that the two approaches from Remark 2.3.1 coincide. The same holds for* H *used in the Rudin-Osher model.*

Remark 2.3.3. *For clarity, we will at times refer to hierarchical decompositions which break an image down into a sum* $\sum_j u_j$ *as summed-MHDM (like the those studied in [TNV04, LRV21]). We introduce this vocabulary to distinguish from the decomposition techniques which use a multiplicative hierarchical representation* $\prod_j u_j$ *of an image.*

2.4 Well-definedness of several models for multiplicative noise removal

Recall we are focusing on multiscale hierarchical decompositions applied to variational denoising models that address multiplicative noise. Before listing those models, we verify the following equivalence that will ensure well-definedness for some schemes of type (2.13) involving particular penalties $J = TV(\log)$.

Proposition 2.4.1. *The following minimization problems in* $BV(\Omega)$

$$u^* \in \arg\min_u \left\{ E(u) := \lambda H(f^\delta, u) + TV(\log(u)) \right\} \tag{2.17}$$

and

$$w^* \in \arg\min_w \left\{ \tilde{E}(w) := \lambda H(f^\delta, e^w) + TV(w) \right\} \tag{2.18}$$

are equivalent (that is, they have the same minimum values). Moreover, the following holds: If u^* *is a minimizer of* (2.17)*, then* $\log(u^*)$ *minimizes* (2.18)*, and if* w^* *is a minimizer of* (2.18)*, then* e^{w^*} *minimizes* (2.17)*.*

Proof. Note that, whenever E and $\tilde{E}$ are defined, one has $E(u) = \tilde{E}(\log(u))$ and $\tilde{E}(w) = E(e^w)$, and furthermore, the minimum values of (2.17) and (2.18) are finite. Indeed, one can easily substitute the constant functions $u = 1$ or $w = 0$ to get a finite energy. To show that minimizers of $\tilde{E}$ lead to minimizers of E, let $w^* \in BV(\Omega)$ minimize (2.18). Since $e^{w^*} \in BV(\Omega)$ holds by a chain rule property (see [Vol67]), we propose this as a candidate minimizer of (2.17). Indeed, suppose by contradiction that there exists $u \in BV(\Omega)$ so that

$$E(u) < E(e^{w^*}) = \tilde{E}(w^*).$$

Since $E(e^{w^*}) < \infty$, we have $E(u) < \infty$ and so $\log(u) \in BV(\Omega)$. Consequently, $\log(u)$ is feasible for $\tilde{E}$ and

$$\tilde{E}(\log(u)) = E(u) < \tilde{E}(w^*),$$

a contradiction to the minimality of $\tilde{E}(w^*)$. We conclude $u^* = e^{w^*}$ is feasible and minimizes (2.17).

For the reverse implication, consider $u^* \in BV(\Omega)$ minimizing (2.17) and suppose there is a $w \in BV(\Omega)$ with

$$\tilde{E}(w) < E(u^*).$$

But then, $e^w \in BV(\Omega)$ and consequently $E(e^w) = \tilde{E}(w) < E(u^*)$, a contradiction. Furthermore, $w^* = \log(u^*)$ is in $BV(\Omega)$ by the finiteness of $E(u^*)$, so w^* minimizes $\tilde{E}$. □

We focus on the following variational models, among which the TNV-log is based on a new energy functional. This is the first work that considers and analyzes these multiscale hierarchical adaptations.

1. A particular Shi-Osher (SO) MHDM model: One can replace the total variation penalty in (2.9) by $J(u) = TV(\log(u))$ and substitute $w = \log(u)$, thus obtaining the *convex* optimization problem

$$\min_w \left\{ TV(w) + \lambda_0 \int_\Omega \left(f^\delta e^{-w} + w \right) \right\}. \tag{2.19}$$

This is (2.11) for $a = c = 1$ and $b = 0$. The paper [JY10] showed existence and uniqueness of the minimizer w_0 when the data $f^\delta \in L^\infty(\Omega)$ satisfy $\inf_\Omega f^\delta > 0$. Moreover, the minimizer w_0 verifies $\inf_\Omega(\log(f^\delta)) \leq w_0 \leq \sup_\Omega(\log(f^\delta))$. We can now apply summed-MHDM, that is solving

$$w_k = \arg\min_w \left\{ \lambda_k \int_\Omega \big(f^\delta e^{-(y_{k-1}+w)} + y_{k-1} + w - \log(f^\delta) - 1 \big) + TV(w) \right\}, \tag{2.20}$$

where $y_{k-1} = \sum_{j=0}^{k-1} w_j$ for $k \in \mathbb{N}$, with $y_{-1} = w_{-1} = 0$. Note that our data fidelity also incorporates the term $-\log(f^\delta) - 1$ in order to build the Itakura-Saito divergence, which is non-negative.

As in the case of w_0, existence and uniqueness can be shown for w_1 (and for further iterations), since the updated data f^δ / e^{w_0} are also away from zero, and the remaining iterates continue to satisfy this relation inductively.

2. AA MHDM model: It was shown in [AA08] that minimizers u_0 of the AA model (2.9) exist in $BV(\Omega)$ for data $f^\delta \in L^\infty(\Omega)$ which satisfy $\inf_\Omega f^\delta > 0$. Moreover, any minimizer u_0 obeys $\inf_\Omega f^\delta \leq u_0 \leq \sup_\Omega f^\delta$. In order to obtain existence of u_1 and of further MHDM iterates, one takes into account that f^δ/u_0 belongs also to $L^\infty(\Omega)$ and verifies $\inf_\Omega f^\delta/u_0 > 0$. Hence, by the same argument in [AA08], u_1 minimizing the AA model with data f^δ/u_0 exists in $BV(\Omega)$, and so on. The generated AA MHDM scheme given by

$$u_k \in \arg\min_u \left\{ \lambda_k \int_\Omega \left(\frac{f^\delta}{u x_{k-1}} + \log(u x_{k-1}) - \log(f^\delta) - 1 \right) + TV(u) \right\} \tag{2.21}$$

will briefly be discussed theoretically and numerically in the upcoming sections.

3. The AA-log MHDM model: One can employ directly the penalty $J(u) = TV(\log(u))$ in the AA model,

$$\min_u \lambda \int_\Omega \left(\frac{f^\delta}{u} + \log(u) - \log(f^\delta) - 1 \right) + TV(\log(u)). \tag{2.22}$$

Clearly, the substitution $w = \log(u)$ yields the SO model. By taking into account the latter and by applying Proposition 2.4.1, problem (2.22) has a unique minimizer. It follows that the MHDM problem

$$u_k \in \arg\min_u \left\{ \lambda_k \int_\Omega \left(\frac{f^\delta}{ux_{k-1}} + \log(ux_{k-1}) - \log(f^\delta) - 1 \right) + TV(\log(u)) \right\} \tag{2.23}$$

is also well-defined in this case.

Despite transforming into the convex SO model under the appropriate substitution, we include the AA-log method because it extends to deblurring, and in the presence of blur the log-transformation no longer produces a convex problem.

4. The TNV-log model: We propose a version of the Rudin-Osher minimization problem, where the penalty $J(u) = TV(\log(u))$ is used instead of just TV. It reads as

$$\min_u \left\{ \lambda_0 \int_\Omega \left(\frac{f^\delta}{u} - 1 \right)^2 + TV(\log(u)) \right\}. \tag{2.24}$$

Since the TNV method is a multiscale adaptation of the RO model, correspondingly, we call RO-log model's multiscale form the TNV-log model, given by

$$u_k \in \arg\min_u \left\{ \lambda_k \int_\Omega \left(\frac{f^\delta}{ux_{k-1}} - 1 \right)^2 + TV(\log(u)) \right\}. \tag{2.25}$$

The existence of minimizers u_0 can be shown via Proposition 2.4.1 and the following Proposition 2.4.2.

Proposition 2.4.2. *Let $f^\delta \in L^\infty(\Omega)$ such that $\inf_\Omega f^\delta > 0$. Then, there exists at least one solution $w \in BV(\Omega)$ of the problem*

$$\min_w \left\{ \lambda_0 \int_\Omega \left(f^\delta e^{-w} - 1 \right)^2 + TV(w) \right\}, \tag{2.26}$$

such that $\inf_\Omega \log f^\delta \leq w \leq \sup_\Omega \log f^\delta$ a.e.

Proof. Let $h(x) = (ae^{-x} - 1)^2$, where $a > 0$ and $x \in \mathbb{R}$. One can prove the result by following the techniques from [AA08, Theorem 4.1], taking into account that the function h is nonincreasing on $(-\infty, \log(a))$ and nondecreasing on $(\log(a), \infty)$. □

As opposed to the situation of the AA model where the $w = \log(u)$ transformation produces a convex problem (2.19), we do not focus on the form (2.26) since it does not exhibit special properties, and in practice the recoveries are the same or slightly worse than those from the TNV-log model (2.24).

2.5 Convergence properties of the multiplicative MHDM

We will consider a general data fidelity H and a penalty, J, which for the moment does not necessarily satisfy (2.16). Moreover, supposing that the general multiscale hierarchical decomposition schemes (2.13) are well-defined (minimizers exist, but might not be unique), we focus on convergence properties of the corresponding iterates. We assume in what follows that the given noisy data f^δ verify

$$H(f^\delta, f) \leq \delta^2, \quad \delta > 0, \tag{2.27}$$

where f denotes the exact—that is, non-noisy but potentially blurred—data. Moreover, existence of a clean image z satisfying $Tz = f$ and $J(z) < \infty$ is also assumed.

The lemma below shows a couple of basic properties for procedure (2.13) (including the SO model, after the logarithm substitution), whenever the iterates are well-defined.

Lemma 2.5.1. *Assume that $J(1) = 0$ and that the iterates $x_k = \prod_{j=0}^{k} u_j$ given by* (2.13) *are well-defined. Then the following inequality holds for any $k \geq 0$,*

$$\lambda_k H(f^\delta, Tx_k) + J(u_k) \leq \lambda_k H(f^\delta, Tx_{k-1}),$$

and the residual $H(f^\delta, Tx_k)$ decreases for increasing k. If (2.27) *is additionally satisfied and $z/x_{k-1} \in dom\, J$ for any $k \geq 0$, then*

$$\lambda_k H(f^\delta, Tx_k) + J(u_k) \leq \lambda_k \delta^2 + J\left(\frac{z}{x_{k-1}}\right) \tag{2.28}$$

holds.

Proof. According to (2.13), one has

$$\lambda_k H(f^\delta, T(x_k)) + J(u_k) \leq \lambda_k H(f^\delta, T(ux_{k-1})) + J(u),$$

for any feasible u. Using $u = 1$ we immediately get

$$\lambda_k H(f^\delta, T(x_k)) + J(u_k) \leq \lambda_k H(f^\delta, Tx_{k-1}).$$

Moreover, dropping $J(u_k)$, gives $H(f^\delta, T(x_k)) \leq H(f^\delta, Tu_{k-1})$, with strict inequality whenever $J(u_k) > 0$ (i.e. provided u_k is not constant). Hence, the fidelity decreases for increasing k. Finally, letting $u = z/x_{k-1}$ in (2.13), one obtains

$$\lambda_k H(f^\delta, T(x_k)) + J(u_k) \leq \lambda_k H(f^\delta, Tz) + J\left(\frac{z}{x_{k-1}}\right) \leq \lambda_k \delta^2 + J\left(\frac{z}{x_{k-1}}\right).$$

□

Remark 2.5.2. *Note that the condition $z/x_{k-1} \in BV(\Omega)$ holds when x_{k-1} is bounded away from zero, since the product of the two bounded variation functions z and $1/x_{k-1}$ has bounded variation, according to [AD90]. Indeed, $1/x_{k-1}$ belongs to $BV(\Omega)$ based on the chain rule for $\varphi \circ x_{k-1} = 1/x_{k-1}$, since $\varphi(s) = 1/s$ is Lipschitz when s is bounded away from zero (see [Vol67]). Therefore, Lemma 2.5.1 works for the corresponding AA and RO models. Moreover, it is also applicable to the* log *models approached in Section 2.4 due to* (2.16) *for $J = TV(\log)$, as $J(z/x_k) \leq J(z) + J(1/x_k) = J(z) + J(x_k) < \infty$. Last but not least, recall that well-definedness of x_k is ensured in all these models when $T = I$ and the data $f^\delta \in L^\infty$ satisfy* $\inf_\Omega f^\delta > 0$.

Additional convergence properties for the multiplicative MHDM can be shown if the penalty J has the properties (2.16).

Proposition 2.5.3. *If* (2.16) *and* (2.27) *are satisfied, and the iterates x_k given by* (2.13) *are well-defined, then the following estimate holds for any $k \geq 0$,*

$$H(f^\delta, Tx_k) \leq \delta^2 + \frac{2J(z)}{(k+1)\lambda_0}. \tag{2.29}$$

Proof. Since J satisfies (2.16) and $u_k = x_k/x_{k-1}$, one has for any $k \geq 0$,

$$J(z/x_k) - J(z/x_{k-1}) \leq J(u_k).$$

This inequality combined with (2.28) yields

$$H(f^\delta, Tx_k) + \frac{1}{\lambda_k} J(z/x_k) \leq \delta^2 + \frac{2}{\lambda_k} J(z/x_{k-1}) = \delta^2 + \frac{1}{\lambda_{k-1}} J(z/x_{k-1}). \tag{2.30}$$

By writing (2.30) for indices $0, 1, ..., k$ and summing up, one has for any $k \geq 0$,

$$(k+1)H(f^\delta, Tx_k) + \frac{1}{\lambda_k} J(z/x_k) \leq \sum_{j=0}^{k} H(f^\delta, Tx_j) + \frac{1}{\lambda_k} J(z/x_k) \leq (k+1)\delta^2 + \frac{2}{\lambda_0} J(z),$$

where the left inequality follows from the monotonicity of the data fidelity term in k (cf. Lemma 2.5.1), and the right one follows from $x_{-1} = 1$. This yields (2.29). □

Clearly, inequality (2.29) holds for the log-based penalty approaches in Section 2.4, as explained in Remark 2.5.2, but not necessarily for the AA and RO models.

Summed-MHDM for non-quadratic data fidelity The work [LRV21] provided error estimates for MHDM in case of quadratic data-fidelity. Fortunately, the proof techniques can be similarly employed in the case of non-quadratic data-fidelities $H(f^\delta, e^w)$ as long as the existence of minimizers w_k is guaranteed. Hence, the following result holds for the SO MHDM defined by (2.20) (compare to [LRV21, Proposition 3.1]).

Proposition 2.5.4. *Let $f^\delta \in L^\infty(\Omega)$ be such that $\inf_\Omega f^\delta > 0$. Then the data-fidelity H is monotonically decreasing for increasing k and*

$$H(f^\delta, e^{w_k}) \leq \delta^2 + \frac{2TV(\log(z))}{\lambda_0(k+1)},\ k \in \mathbb{N}.$$

2.5.1 Discrepancy principle stopping rule

Computing too many multiscale hierarchical iterations can result in getting back more and more noise in the reconstructed image. Therefore, stopping the procedure early enough is necessary. In view of this, we propose a stopping rule for (2.13) and show convergence properties. Let us define the following stopping index,

$$k^*(\delta) := \max\{k \in \mathbb{N} : H(f^\delta, Tx_k) \geq \tau\delta^2\}, \quad \text{for some}\, \tau > 1. \tag{2.31}$$

As shown below, this index exists and convergence of the data fidelity to zero is guaranteed.

Proposition 2.5.5. *Assume that* (2.16) *and* (2.27) *are satisfied, and the iterates* x_k *given by* (2.13) *are well-defined. Then the stopping index* (2.31) *is finite. If* $(k^*(\delta))$ *is unbounded as* $\delta \to 0$*, then* $\lim\limits_{\delta\to 0} H(f^\delta, Tx_{k^*(\delta)}) = 0$ *holds.*

Proof. By writing (2.29) for $k = k^*(\delta)$ and using (2.31), it follows that

$$\tau\delta^2 \leq \delta^2 + \frac{2}{\lambda_0(k^*(\delta)+1)} J(z)$$

and thus, the stopping index is finite:

$$k^*(\delta) \leq \frac{2J(z)}{\lambda_0(\tau-1)\delta^2} - 1.$$

If $(k^*(\delta))$ is unbounded, then (2.29) written for $k = k^*(\delta)$ implies $\lim\limits_{\delta\to 0} H(f^\delta, Tx_{k^*(\delta)}) = 0$. □

2.5.2 Convergence of multiplicative MHDM for particular models

This subsection deals with convergence of the MHDM iterates for the particular models considered in the current study. Note that the residual H converges to zero when the procedure is stopped earlier at $k^*(\delta)$ cf. (2.31), as seen in the previous subsection. We now analyze the implications of this convergence in case of the two data fidelities employed in the proposed MHDM, namely the quadratic term of the RO model and the Itakura-Saito distance. As opposed to the MHDM concerning additive noise in images, where convergence is shown with respect to the L^2 norm, we can prove only pointwise convergence on subsequences a.e. for the MHDM corresponding to multiplicative noise.

Proposition 2.5.6. *Assume that* (2.16) *and* (2.27) *are satisfied, and the iterates* x_k *given by* (2.13) *are well-defined, whenever the data fidelity* H *is defined by* (2.14) *or* (2.15)*. If* $(k^*(\delta))$ *is unbounded as* $\delta \to 0$*, then* $(Tx_{k^*(\delta)})$ *converges a.e. to* f *on a subsequence. In particular for the denoising case, one has a.e. convergence of* $(x_{k^*(\delta)})$ *on a subsequence to the true image.*

Proof. According to Proposition 2.5.5, one has $\lim\limits_{\delta\to 0} H(f^\delta, Tx_{k^*(\delta)}) = 0$. If the data fidelity is given by (2.14), then $\left(\frac{f^\delta}{Tx_{k^*(\delta)}}\right)$ converges strongly to 1 in $L^2(\Omega)$. This yields a.e. convergence of $(\frac{f^\delta}{Tx_{k^*(\delta)}})$ to 1 on a subsequence, thus a.e. convergence of $(Tx_{k^*(\delta)})$ to f on a subsequence. Now consider H given by (2.15). Then the convergence of the residual (cf. Proposition 2.5.5) implies that the positive sequence $(d(f^\delta, Tx_{k^*(\delta)}))$ converges to zero in the $L^1(\Omega)$ norm, where $d(f^\delta, Tx_{k^*(\delta)}) = \frac{f^\delta}{Tx_{k^*(\delta)}} + \log(Tx_{k^*(\delta)}) - \log(f^\delta) - 1$. Consequently, it converges a.e. to zero on a subsequence. It follows that the sequence $(Tx_{k^*(\delta)})$ is (a.e.) pointwise bounded in $[0, \infty)$, otherwise a subsequence would diverge to $+\infty$, which would contradict $d(f^\delta, Tx_{k^*(\delta)}) \to 0$. Therefore, $(Tx_{k^*(\delta)})$ converges on a subsequence to some nonnegative function g a.e., implying a.e. convergence of $(d(f^\delta, Tx_{k^*(\delta)}))$ to $d(f, g)$. Uniqueness of the limit yields $d(f, g) = 0$ a.e., that is $g = f$ a.e., due to the strict convexity of the Burg entropy which defines the (pointwise) Itakura Saito distance d. □

2.6 Extensions of the multiplicative MHDM

2.6.1 A tight multiplicative MHDM

In this section, we adapt to the multiplicative noise case the tight hierarchical decomposition method [MNR19] proposed in the additive noise context. That tight version incorporated an additional penalization, namely on the entire approximation (x_k), in order to obtain better convergence properties of (x_k). Since this section follows the structure of the tight MHDM in the case of additive noise [LRV21], we introduce the tight method in the new setting by omitting proof details.

Let (a_k) be a sequence of nonnegative numbers such that for any $k \geq 1$,

$$\lim_{k\to\infty} a_k = 0 \quad \text{and} \quad a_k \leq a_{k-1}. \tag{2.32}$$

Set λ_0 to be a positive number and let $(\lambda_k) \subset (0, \infty)$ verify the following relaxed inequality

$$2\lambda_k \leq \lambda_{k+1}, \quad k \geq 0, \tag{2.33}$$

rather than the equality $2\lambda_k = \lambda_{k+1}$. Finally, determine $u_k \in BV(\Omega)$ as a solution of

$$\min_u F_k(u), \quad \text{with} \quad F_k(u) = \lambda_k H(f^\delta, T(ux_{k-1})) + \lambda_k a_k J(ux_{k-1}) + J(u),$$

with, as before, $x_{k-1} = \prod_{j=0}^{k-1} u_j$, $x_{-1} = 1$. The tight formulation, then, is augmented by a new penalization term $\lambda_k a_k J(ux_{k-1})$.

Remark 2.6.1. *The tight versions of the denoising models presented in Section 2.4 are also well-defined (similar arguments).*

Under the assumptions of Lemma 2.5.1, one can derive similarly the following inequalities,

$$\lambda_k H(f^\delta, Tx_k) + \lambda_k a_k J(x_k) + J(u_k) \leq \lambda_k H(f^\delta, Tx_{k-1}) + \lambda_k a_k J(x_{k-1}), \tag{2.34}$$

$$\lambda_k H(f^\delta, Tx_k) + \lambda_k a_k J(x_k) + J(u_k) \leq \lambda_k a_k J(z) + J(z/x_{k-1}) + \lambda_k \delta^2, \quad k \geq 0.$$

Note that (2.34) yields the decreasing monotonicity of $H(f^\delta, Tx_k) + a_k J(x_k)$, which is a type of residual in the tight method. If we further require

$$\sum_{k=0}^{\infty} a_k < \infty \tag{2.35}$$

and define the stopping index also by a discrepancy rule

$$k^*(\delta) := \max\{k \in \mathbb{N} : H(f^\delta, Tx_k) + a_k J(x_k) \geq \tau\delta^2\}, \quad \text{for some}\, \tau > 1, \tag{2.36}$$

then the results below can be established in a similar manner to the ones for the multiplicative MHDM when J verifies (2.16).

Proposition 2.6.2. *Let conditions* (2.16)*,* (2.27)*,* (2.32) *and* (2.33) *be satisfied. Then the following estimate holds for any* $k \geq 0$*,*

$$H(f^\delta, Tx_k) + a_k J(x_k) \leq \delta^2 + \left(\sum_{j=0}^{k} a_j\right) \frac{J(z)}{k+1} + \frac{2J(z)}{(k+1)\lambda_0}.$$

Moreover, if (2.35) *is verified, then the stopping index defined by* (2.36) *is finite. Additionally,*

1. *If* $(k^*(\delta))$ *is unbounded, then* $\lim_{\delta \to 0} H(f^\delta, Tx_{k^*(\delta)}) = 0$ *and* $\lim_{\delta \to 0} a_{k^*(\delta)} J(x_{k^*(\delta)}) = 0$*.*
2. *If the stopping index is chosen as* $k^*(\delta) \sim \frac{1}{\delta^2}$*, then*

$$H(f^\delta, Tx_{k^*(\delta)}) + a_{k^*(\delta)} J(x_{k^*(\delta)}) = O(\delta^2).$$

By adapting the techniques from [LRV21, Section 4] to the multiplicative noise case with the help of the condition $\limsup_{k\to\infty} \frac{2^k}{\lambda_k a_k} = 0$, one can show $J(x_{k^*(\delta)}) \to J(z)$, demonstrating that the recoveries have the same level regularity as the clean image. Compare also to [MNR19, Theorem 2.5] which addresses the tight summed-MHDM. Additionally, the convergence in the sense of Subsection 2.5.2 holds.

2.6.2 A refined multiplicative MHDM

In order to promote specific properties of the u_k components, we propose a multiplicative counterpart of the refined method introduced in [LRV21]. Thus, we allow the penalization on the hierarchical component to be a functional different from J, that is different from TV or $TV(\log)$. Although it can vary in every iteration as stated in [LRV21], we consider it fixed (for fixed J) hereafter and denote it by R.

In particular, we require $R : L^2(\Omega) \to \mathbb{R} \cup \{\infty\}$ to be a seminorm which is weakly lower semicontinuous and verifies the following inequality for some $c > 0$:

$$R(u) \leq cJ(u), \forall u \in dom\, R.$$

Construct a sequence $(u_k) \subset BV(\Omega)$ with u_k as a solution of

$$\min_u F_k(u), \quad \text{with} \quad F_k(u) = \lambda_k H(f^\delta, T(ux_{k-1})) + \lambda_k a_k J(ux_{k-1}) + R(u),$$

where λ_k and a_k are defined as in the tight formulation. One can similarly derive the estimate

$$H(f^\delta, Tx_k) + a_k J(x_k) \leq \delta^2 + \left(\sum_{j=0}^{k} a_j \right) \frac{J(z)}{k+1} + \frac{2R(z)}{(k+1)\lambda_0},$$

as well as the same convergence results under the same assumptions, in addition to the ones above for R. An improved behavior (as compared to the tight and the regular MHDM versions) will be shown numerically by considering $R = \|\cdot\|_*$ or $R = \|\log(\cdot)\|_*$ when $J = TV$ or $J = TV(\log)$, respectively, where

$$\|u\|_* = \sup_{\phi:\ TV(\phi)>0} \frac{\langle u, \phi \rangle}{TV(\phi)},$$

is the dual norm.

2.7 Numerical schemes for multiplicative MHDM minimization

Here we introduce numerical discretizations for the three classes of MHDM problems we consider: Shi-Osher adaptations, AA-like models, and TNV inspired methods.

2.7.1 Discretization Notation

We will make repeated use of common finite differences in the sections that follow. To aid with notational brevity, we introduce the following symbolic representations. For functions

$u, v : \Omega \to \mathbb{R}$ taking values on a grid with meshsize $\Delta x \times \Delta y$, i.e. $u_{ij} = u(x_i, y_j)$ for $(x_i, y_j) = (x_0 + i\Delta x, y_0 + j\Delta y) \in \Omega$, define,

$$D_x^{\pm} u_{ij} := \pm(u_{i\pm1,j} - u_{ij})/\Delta x \tag{2.37a}$$

$$D_y^{\pm} u_{ij} := \pm(u_{i,j\pm1} - u_{ij})/\Delta y \tag{2.37b}$$

$$D_x^0 u_{ij} := \frac{u_{i+1,j} - u_{i-1,j}}{2\Delta x} \tag{2.37c}$$

$$D_y^0 u_{ij} := \frac{u_{i,j+1} - u_{i,j-1}}{2\Delta y} \tag{2.37d}$$

$$c_\epsilon(u_{ij}) := \frac{1}{\sqrt{(D_x^+ u_{ij})^2 + (D_y^+ u_{ij})^2 + \epsilon^2}} \tag{2.37e}$$

$$c_u^{ij} := 2c_\epsilon(u_{ij}) + c_\epsilon(u_{i-1,j}) + c_\epsilon(u_{i,j-1}) \tag{2.37f}$$

$$d_\epsilon(u_{ij}) := \frac{c_\epsilon(u_{ij})}{|u_{ij}|} \tag{2.37g}$$

$$d_u^{ij} := 2d_\epsilon(u_{ij}) + d_\epsilon(u_{i-1,j}) + d_\epsilon(u_{i,j-1}) \tag{2.37h}$$

$$\chi[d_\epsilon, i, j]_v^u := d_\epsilon(u_{ij})v_{i+1,j} + d_\epsilon(u_{i-1,j})v_{i-1,j} + d_\epsilon(u_{ij})v_{i,j+1} + d_\epsilon(u_{i,j-1})v_{i,j-1}, \tag{2.37i}$$

and we will at times drop the explicit ϵ-dependence in equations (2.37) when it is clear by context. With these notations in place, we proceed with the numerical approaches to the SO, AA and TNV MHDM adaptations, along with their tight and refined formulations.

2.7.2 Shi-Osher MHDM

Discretization of Euler-Lagrange equations

We develop a numerical scheme for the Shi-Osher (SO) model (2.19) adapted to multiscale hierarchical decomposition (MHDM). Our ultimate goal is to recover a k-level multiscale approximation, $x_k = e^{y_k}$, to the clean image z from the noise-degraded signal $f^\delta = (Tz) \cdot \eta$. The approximation $x_k = \prod_{j=0}^{k} u_j$ is multiscale, composed of individual scales $u_j = e^{w_j}$ where each w_j satisfies (2.20). Recall, the SO model is a convex recasting of the AA model (2.22) in the case of no blur, so we will take T to be the identity for the SO MHDM sections.

For $k \geq 1$, we assume we are given a partial reconstruction $y_{k-1} := \sum_{j=0}^{k-1} w_j$, and seek a sufficiently regular w_k—as imposed by $TV(\cdot)$—so that the sum $w_k + y_{k-1}$ fits f^δ according to the data fidelity term. The Euler-Lagrange equation associated with the minimization problem (2.20) is found by taking the first variation. We include the computation here for completeness, but exclude it from subsequent sections. Let $E_k(\cdot)$ be the functional to be minimized in (2.20). Then, we seek w so that

$$\left.\frac{dE_k(w+sv)}{ds}\right|_{s=0} = 0 \tag{2.38}$$

for any $v \in BV(\Omega)$. Consequently, by the divergence theorem

$$\begin{aligned}
\left.\frac{dE_k(w+sv)}{ds}\right|_{s=0} &= \lambda_k \int_\Omega \left(1 - f^\delta e^{-(w+y_{k-1})}\right) v + \int_\Omega \frac{\nabla w}{|\nabla w|} \nabla v \\
&= \lambda_k \int_\Omega \left(1 - f^\delta e^{-(w+y_{k-1})}\right) v + \int_{\partial\Omega} v \frac{\nabla w \cdot \vec{n}}{|\nabla w|} - \int_\Omega \operatorname{div}\left(\frac{\nabla w}{|\nabla w|}\right) v \\
&= \int_\Omega \left(\lambda_k \left(1 - f^\delta e^{-(w+y_{k-1})}\right) - \operatorname{div}\left(\frac{\nabla w}{|\nabla w|}\right)\right) v = 0. \qquad (2.39)
\end{aligned}$$

The condition (2.39) holds for arbitrary v, provided $\partial w / \partial \vec{n} = 0$ on $\partial\Omega$, allowing us to conclude the following PDE must hold for w:

$$\begin{cases} \operatorname{div}\left(\frac{\nabla w}{|\nabla w|}\right) - \lambda_k (1 - f^\delta e^{-(w+x_{k-1})}) = 0 & \text{in } \Omega, \\ \frac{\partial w}{\partial \vec{n}} = 0 & \text{in } \partial\Omega. \end{cases} \tag{2.40}$$

We can approach (2.40) by introducing artificial time $w = w(x,t)$, and iterating w in time till it reaches equilibrium exactly when (2.40) is satisfied. This gives the following gradient descent scheme with Neumann boundary conditions

$$\begin{cases} \frac{\partial w}{\partial t} = \operatorname{div}\left(\frac{\nabla w}{|\nabla w|}\right) - \lambda_k (1 - f^\delta e^{-(w+y_{k-1})}) & \text{in } \Omega, \\ \frac{\partial w}{\partial \vec{n}} = 0 & \text{in } \partial\Omega. \end{cases} \tag{2.41}$$

We discretize (2.41) with added ϵ-regularization to the $\nabla w / |\nabla w|$ terms by applying the forward and backward spatial differencing operators, D^+ and D^-, to the right-hand-side of

(2.41)

$$\begin{aligned}\frac{w_{ij}^{n+1}-w_{ij}^{n}}{\Delta t} &= -\lambda_k\left(1-f_{ij}^{\delta}e^{-(w_{ij}^{n}+(x_{k-1})_{ij})}\right)+\\ &\qquad D_x^{-}\left(\frac{D_x^{+}w_{ij}^{n}}{\sqrt{\epsilon^2+(D_x^{+}w_{ij}^{n})^2+(D_y^{+}w_{ij}^{n})^2}}\right)+D_y^{-}\left(\frac{D_y^{+}w_{ij}^{n}}{\sqrt{\epsilon^2+(D_x^{+}w_{ij}^{n})^2+(D_y^{+}w_{ij}^{n})^2}}\right).\end{aligned}$$

The added superscript w_{ij}^n indicates time (or iterate number) for grid variable w, and we have introduced a time step-size Δt. We also take $\Delta x = \Delta y = 1$. Using the notation introduced in Section 2.7.1, we expand to get

$$\begin{aligned}\frac{w_{ij}^{n+1}-w_{ij}^{n}}{\Delta t} &= -\lambda_k\left(1-f_{ij}^{\delta}e^{-(w_{ij}^{n}+(x_{k-1})_{ij})}\right)+\\ &\qquad c(w_{ij}^{n})D_x^{+}w_{ij}^{n}-c(w_{i-1,j}^{n})D_x^{+}w_{i-1,j}^{n}+c(w_{ij}^{n})D_y^{+}w_{ij}^{n}-c(w_{i,j-1}^{n})D_y^{+}w_{i,j-1}^{n}\\ &= -\lambda_k\left(1-f_{ij}^{\delta}e^{-(w_{ij}^{n}+(x_{k-1})_{ij})}\right)+\\ &\qquad c(w_{ij}^{n})w_{i+1,j}^{n}+c(w_{i-1,j}^{n})w_{i-1,j}^{n}+c(w_{ij}^{n})w_{i,j+1}^{n}+c(w_{i,j-1}^{n})w_{i,j-1}^{n}+\\ &\qquad w_{ij}^{n}\left(c(w_{ij}^{n})+c(w_{i-1,j}^{n})+c(w_{i,j}^{n})+c(w_{i,j-1}^{n})\right)\\ &= -\lambda_k\left(1-f_{ij}^{\delta}e^{-(w_{ij}^{n}+(x_{k-1})_{ij})}\right)+\chi[c_\epsilon,i,j]_{w^n}^{w^n}-w_{ij}^{n}c_{w^n}^{ij}. \qquad (2.42)\end{aligned}$$

To mitigate the potential for instability in the highly non-linear discretized PDE, an implicit scheme is preferred. However, solving (2.42) with right-hand-side w^n swapped for w^{n+1} is non-trivial. Instead, we propose a semi-implicit scheme by isolating all terms linear in w_{ij}^n, and promoting them to w_{ij}^{n+1}. Then, one can solve for w_{ij}^{n+1} in a Gauss-Seidel like fashion. The final iterative scheme

$$w_{ij}^{n+1}=\frac{1}{1+\Delta t\cdot c_{w^n}^{ij}}\left[w_{ij}^{n}-\lambda_k\Delta t\left(1-f_{ij}^{\delta}e^{-(w_{ij}^{n}+(x_{k-1})_{ij})}\right)+\Delta t\chi[c_\epsilon,i,j]_{w^n}^{w^n}\right], \qquad (2.43)$$

solves (2.41) on interior points (i,j) within Ω. Boundary points are obtained to ensure $\partial w/\partial\vec{n}=0$ on $\partial\Omega$. We can numerically determine w_k by running w^n to steady state, followed by updating $y_k = w_k + y_{k-1}$. Having recovered y_k, we subsequently obtain the reconstruction

x_k via an exponential transform. To initialize process (2.43), see the discussion in Subsection 2.7.9.

Let $\texttt{SO\_MHDM}(f, y_{k-1}, \Delta t, \lambda_k, \epsilon, \texttt{maxIter})$ be the numerical solution to (2.41) after running $n = \texttt{maxIter}$ times. Then the image restoration algorithm proceeds as follows:

Algorithm 1 SO MHDM

INPUT: noisy image $f^\delta = z \cdot \eta$, where η is some multiplicative noise and z is the original image.

OUTPUT: $x_{\texttt{numScales}}$, an approximation to z.

```
Initialize: $y_{-1} = 0$, $\lambda_0 = 0.01$, $\epsilon = 0.01$ (or some small constant)
Choose maxIter
for $k = 0, 1, 2, \ldots, \texttt{numScales}$ do
    $w_k \leftarrow \texttt{SO\_MHDM}(f^\delta, y_{k-1}, \Delta t, \lambda_k, \epsilon, \texttt{maxIter})$
    $y_k \leftarrow w_k + y_{k-1}$
    $\lambda_{k+1} \leftarrow 2\lambda_k$
end for
return : $x_{\texttt{numScales}} = e^{y_{\texttt{numScales}}}$
```

ADMM for Shi-Osher MHDM

In addition to the Euler-Lagrange approach, we consider the popular alternating direction method of multipliers (ADMM) for the convex optimization problem obtained by the Shi-Osher formulation. Recall, given y_{k-1} and λ_k, we solve (2.20) to form the multiscale reconstruction $x_k = e^{y_k}$ with $y_k = \sum_{j=0}^{k} w_j$. We split the problem into simpler subproblems, minimizing the data fidelity term $\lambda_k \int \left(f^\delta e^{-(w+y_{k-1})} + (w + y_{k-1}) \right)$ and the regularizing term $TV(w)$ separately, subject to the condition these minimizers match. This gives the ADMM formulation

$$\theta^{j+1} = \arg\min_{\theta} \lambda_k \int \left(f^\delta e^{-(\theta+y_{k-1})} + (\theta + y_{k-1})\right) + \frac{\rho}{2}\|\theta - \psi^j + \vartheta^j\|_2^2, \tag{2.44}$$

$$\psi^{j+1} = \arg\min_{\psi} TV(\psi) + \frac{\rho}{2}\|\theta^{j+1} - \psi + \vartheta^j\|_2^2, \tag{2.45}$$

$$\vartheta^{j+1} = \vartheta^j + \theta^{j+1} - \psi^{j+1}, \tag{2.46}$$

which proceeds iteratively in j, forming solution $w_k = \theta^\infty$. Here, ρ is a constant parameter of the scheme. The stopping condition for ADMM is determined by some tolerance $\epsilon > 0$ and is satisfied whenever

$$\max\left\{\|\theta^{j+1} - \theta^j\|^2, \|\vartheta^{j+1} - \vartheta^j\|^2, \|\psi^{j+1} - \psi^j\|^2, \|\theta^{j+1} - \psi^{j+1}\|^2\right\} < \epsilon.$$

To perform the minimizations in (2.44), we use Newton's iteration. Noticing (2.45) is a ROF denoising problem, we use an exact total variation minimization routine [CD09] for solving the ROF problem provided at `http://www.cmap.polytechnique.fr/~antonin/software/`. To use the ADMM formulation for SO MHDM, use Algorithm 1 with the output of (2.44) used in place of $\texttt{SO\_MHDM}(f^\delta, y_{k-1}, \Delta t, \lambda_k, \epsilon, \texttt{maxIter})$.

2.7.3 Shi-Osher Tight MHDM

Discretization of Euler-Lagrange equations

For the tight SO scheme, we consider the modified objective function

$$w_k = \arg\min_{w} \left\{TV(w) + \lambda_k a_k TV(w + y_{k-1}) + \lambda_k \int \left(f^\delta e^{-(w+y_{k-1})} + (w + y_{k-1})\right)\right\}. \tag{2.47}$$

Notice that the only alteration from the standard SO MHDM model is the additional $TV(w+y_{k-1})$ term, so the resulting Euler-Lagrange equations will be modified solely by this term. Additionally, the boundary condition will require $\vec{n} \cdot \nabla(w + y_{k-1}) = 0$. However, since

$y_{k-1} = \sum_{j=0}^{k-1} w_j$ where $\vec{n} \cdot \nabla w_j = 0$ on the boundary, we need only to impose $\vec{n} \cdot \nabla w = 0$ on $\partial\Omega$.

Consequently, the Euler-Lagrange equation for (2.47) after considering artificial time is

$$\begin{cases} \frac{\partial w}{\partial t} = \text{div}\left(\frac{\nabla w}{|\nabla w|}\right) + \lambda_k a_k \text{div}\left(\frac{\nabla(w+y_{k-1})}{|\nabla(w+y_{k-1})|}\right) - \lambda_k(1 - fe^{-w-y_{k-1}}) & \text{in } \Omega, \\ \frac{\partial w}{\partial \vec{n}} = 0 & \text{in } \partial\Omega. \end{cases} \tag{2.48}$$

Our strategy in discretizing (2.48) is to introduce as many implicit terms as possible by isolating w_{ij}^n. Using blended forward and backward differences, the discretization would be identical to (2.42), except for the additional term with $\lambda_k a_k$ as a coefficient. We focus first on this term, noting by linearity of divergence

$$\text{div}\left(\frac{\nabla(w + y_{k-1})}{|\nabla(w + y_{k-1})|}\right) = \text{div}\left(\frac{\nabla(w)}{|\nabla(w + y_{k-1})|}\right) + \text{div}\left(\frac{\nabla(y_{k-1})}{|\nabla(w + y_{k-1})|}\right). \tag{2.49}$$

For notational compactness, we introduce the variable $v^n = w^n + y_{k-1}$.

With the split divergence terms, we proceed with the discretization, recalling again the shorthand notation we introduced in (2.37) in Section 2.7.1

$$
\begin{aligned}
\frac{w_{ij}^{n+1}-w_{ij}^{n}}{\Delta t} &= -\lambda_k\left(1-f_{ij}^{\delta}e^{-(w_{ij}^n+(x_{k-1})_{ij})}\right)+\chi[c_\epsilon,i,j]_{w^n}^{w^n}+w_{ij}^n c_{w^n}^{ij}+\lambda_k a_k\Bigg[\\
&D_x^-\left(\frac{D_x^+ w_{ij}^n}{\sqrt{\epsilon^2+(D_x^+v_{ij}^n)^2+(D_y^+v_{ij}^n)^2}}\right)+D_y^-\left(\frac{D_y^+ w_{ij}^n}{\sqrt{\epsilon^2+(D_x^+v_{ij}^n)^2+(D_y^+v_{ij}^n)^2}}\right)+\\
&D_x^-\left(\frac{D_x^+ (y_{k-1})_{ij}}{\sqrt{\epsilon^2+(D_x^+v_{ij}^n)^2+(D_y^+v_{ij}^n)^2}}\right)+D_y^-\left(\frac{D_y^+ (y_{k-1})_{ij}}{\sqrt{\epsilon^2+(D_x^+v_{ij}^n)^2+(D_y^+v_{ij}^n)^2}}\right)\Bigg]\\
&= -\lambda_k\left(1-f_{ij}^{\delta}e^{-(w_{ij}^n+(x_{k-1})_{ij})}\right)+\chi[c_\epsilon,i,j]_{w^n}^{w^n}-w_{ij}^n c_{w^n}^{ij}+\lambda_k a_k\Bigg[\\
&c(v_{ij}^n)w_{i+1,j}^n+c(v_{i-1,j}^n)w_{i-1,j}^n+c(v_{ij}^n)w_{i,j+1}^n+c(v_{i,j-1}^n)w_{i,j-1}^n-\\
&w_{ij}^n\left(c(v_{ij}^n)+c(v_{i-1,j}^n)+c(v_{i,j}^n)+c(v_{i,j-1}^n)\right)+\\
&c(v_{ij}^n)(y_{k-1})_{i+1,j}+c(v_{i-1,j}^n)(y_{k-1})_{i-1,j}+c(v_{ij}^n)(y_{k-1})_{i,j+1}+c(v_{i,j-1}^n)(y_{k-1})_{i,j-1}-\\
&\qquad(y_{k-1})_{ij}\left(c(v_{ij}^n)+c(v_{i-1,j}^n)+c(v_{i,j}^n)+c(v_{i,j-1}^n)\right)\Bigg]\\
&= -\lambda_k\left(1-f_{ij}^{\delta}e^{-(w_{ij}^n+(x_{k-1})_{ij})}\right)+\chi[c_\epsilon,i,j]_{w^n}^{w^n}-w_{ij}^n c_{w^n}^{ij}+\lambda_k a_k\Bigg[\\
&\chi[c_\epsilon,i,j]_{w^n}^{v^n}-w_{ij}^n c_{v^n}^{ij}+\chi[c_\epsilon,i,j]_{y_{k-1}}^{v^n}-(y_{k-1})_{ij}c_{v^n}^{ij}\Bigg].
\end{aligned}
\tag{2.50}
$$

Isolating the w_{ij}^n terms and exchanging for w_{ij}^{n+1} produces the following semi-implicit scheme for SO Tight MHDM

$$
\begin{aligned}
w_{ij}^{n+1}=\frac{1}{1+\Delta t(c_{w^n}^{ij}+\lambda_k a_k c_{v^n}^{ij})}&\Bigg\{w_{ij}^n+\Delta t\Bigg[-\lambda_k\left(1-f_{ij}^{\delta}e^{-(w_{ij}^n+(x_{k-1})_{ij})}\right)+\\
&\chi[c_\epsilon,i,j]_{w^n}^{w^n}+\lambda_k a_k\Bigg(\chi[c_\epsilon,i,j]_{w^n}^{v^n}+\chi[c_\epsilon,i,j]_{y_{k-1}}^{v^n}-(y_{k-1})_{ij}c_{v^n}^{ij}\Bigg)\Bigg]\Bigg\}.
\end{aligned}
\tag{2.51}
$$

Compared with the Shi-Osher MHDM, the Shi-Osher Tight MHDM has the same initialization (see Section 2.7.9) and parameters except $\lambda_{k+1}=3\lambda_k$, with $a_0=1$, $a_k=\frac{a_0}{(k+1)^{3/2}}$. The algorithm for SO Tight MHDM is summarized below

Algorithm 2 SO Tight MHDM

INPUT: noisy image $f^\delta = z \cdot \eta$, where η is some multiplicative noise and z is the original image.

OUTPUT: $x_{\texttt{numScales}}$, an approximation to z.

```
**Initialize:** $y_{-1} = 0$, $\lambda_0 = 0.01$, $a_0 = 1$, $\epsilon = 0.01$ (or some small constant)
Choose maxIter
**for** $k = 0, 1, 2, \dots, \texttt{numScales}$ **do**
    $a_k \leftarrow \frac{a_0}{(k+1)^{3/2}}$
    $w_k \leftarrow \texttt{SO\_Tight\_MHDM}(f^\delta, y_{k-1}, \Delta t, \lambda_k, a_k, \epsilon, \texttt{maxIter})$    ▷ output of (2.51)
    $y_k \leftarrow w_k + y_{k-1}$
    $\lambda_{k+1} \leftarrow 3\lambda_k$
**end for**
**return** : $x_{\texttt{numScales}} = e^{y_{\texttt{numScales}}}$
```

ADMM for Shi-Osher tight MHDM

We can also leverage ADMM for the SO Tight MHDM process. The utility of ADMM is it's ability to handle minimization problems consisting of sums of functionals which, in congregate, are difficult to handle. The SO Tight MHDM consists of three terms—a data fidelity term, the tight regularization term and standard regularizer—and we address each term separately. Given y_{k-1}, λ_k, a_k, the ADMM proceeds as follows:

$$\theta^{j+1} = \arg\min_{\theta} \lambda_k \int \left(f^\delta e^{-(\theta + y_{k-1})} + (\theta + y_{k-1}) \right) + \frac{\rho}{2} \|\theta - \frac{\psi_1^j + \psi_2^j}{2} + \vartheta^j\|_2^2, \tag{2.52}$$

$$\psi_1^{j+1} = \arg\min_{\psi} \lambda_k a_k TV(\psi + y_{k-1}) + \frac{\rho}{2} \|\theta^{j+1} - \frac{\psi + \psi_2^j}{2} + \vartheta^j\|_2^2, \tag{2.53}$$

$$\psi_2^{j+1} = \arg\min_{\psi} TV(\psi) + \frac{\rho}{2} \|\theta^{j+1} - \frac{\psi_1^{j+1} + \psi}{2} + \vartheta^j\|_2^2, \tag{2.54}$$

$$\vartheta^{j+1} = \vartheta^j + \theta^{j+1} - \frac{\psi_1^{j+1} + \psi_2^{j+1}}{2}, \tag{2.55}$$

iteratively forming the solution $w_k = \theta^\infty$. Similar to ADMM for SO MHDM, the stopping criterion is met whenever

$$\|\theta^{j+1} - \theta^j\|^2, \|\vartheta^{j+1} - \vartheta^j\|^2, \|\psi_1^{j+1} - \psi_1^j\|^2, \|\psi_2^{j+1} - \psi_2^j\|^2, \|\theta^{j+1} - \frac{1}{2}(\psi_1^{j+1} + \psi_2^{j+1})\|^2 < \epsilon$$

is satisfied for some $\epsilon > 0$.

We approach (2.52) and (2.54) with Newton's method and the ROF solve as before. For (2.53) we use $g = \psi + y_{k-1}$ and rescale the problem. This gives

$$g^{j+1} = \arg\min_g \frac{4a_k\lambda_k}{\rho} TV(g) + \frac{1}{2}\|2\theta^{j+1} + 2\vartheta^j + y_{k-1} - \psi_2^j - g\|^2 \ ,$$

from which $\psi_1^{j+1} = g^{j+1} - y_{k-1}$ can be recovered. To use the ADMM formulation for SO Tight MHDM, use Algorithm 2 with the final output of (2.52) in place of $\texttt{SO\_Tight\_MHDM}(f^\delta, y_{k-1}, \Delta t, \lambda_k, a_k, \epsilon, \texttt{maxIter})$.

2.7.3.1 Shi-Osher Refined MHDM

For the refined version, we consider

$$w_k = \arg\min_w \left\{ \|w\|_* + \lambda_k a_k TV(w + y_{k-1}) + \lambda_k \int \left(f^\delta e^{-(w+y_{k-1})} + (w + y_{k-1}) \right) \right\}, \quad (2.56)$$

where $\|w\|_* = \sup_{TV_\epsilon(\phi)\neq 0} \frac{\langle w,\phi\rangle}{TV(\phi)}$. Thus, the main modification is the $*$-norm term. Our approach towards minimizing (2.56) is alternatingly considering $\|\cdot\|_*$ and the remaining terms separately. Assuming w is given, we can determine ϕ by studying the Euler-Lagrange equation associated with maximizing

$$\frac{\langle w, \phi\rangle}{TV(\phi)} \quad (2.57)$$

over ϕ, as done in [LRV21]. Likewise, given ϕ, the Euler-Lagrange equation for the refined problem (2.56) is modified from (2.48) only by the $*$-norm term. Thus, we can find w_k through the Euler-Lagrange equation for (2.56). We will solve for ϕ and w by alternatingly time-stepping in each variable, using a similar semi-implicit discretization as with in the regular and tight formulations.

To maximize (2.57), ϕ will induce a vanishing first variation, implying

$$\begin{aligned}\frac{d}{ds}\frac{\langle w,\phi+s\theta\rangle}{TV_\epsilon(\phi+s\theta)}\bigg|_{s=0} &= \frac{\int w\theta}{TV_\epsilon(\phi)} - \frac{\int w\phi}{TV_\epsilon(\phi)^2}\int\frac{\nabla\phi\cdot\nabla\theta}{\sqrt{\epsilon^2+|\nabla\phi|^2}} \\ &= \frac{\int w\theta}{TV_\epsilon(\phi)} + \frac{\int w\phi}{TV_\epsilon(\phi)^2}\int\theta\mathrm{div}\frac{\nabla\phi}{\sqrt{\epsilon^2+|\nabla\phi|^2}} = 0,\end{aligned} \tag{2.58}$$

where we have ϵ-regularized the total variation term $TV_\epsilon(w) := \int\sqrt{\epsilon^2+|\nabla w|^2}$. Equation (2.58) holds for any $\theta\in BV(\Omega)$, provided $\vec{n}\cdot\nabla\phi = 0$ on $\partial\Omega$ stemming from the integration by parts in (2.58). Therefore, ϕ satisfies

$$\begin{cases}\frac{w}{TV_\epsilon(\phi)} + \frac{\langle w,\phi\rangle}{TV_\epsilon(\phi)^2}\mathrm{div}\left(\frac{\nabla\phi}{\sqrt{\epsilon^2+|\nabla\phi|^2}}\right) = 0, & \text{in } \Omega \\ \vec{n}\cdot\nabla\phi = 0, & \text{on } \partial\Omega,\end{cases}$$

or equivalently,

$$\begin{cases}w + \frac{\langle w,\phi\rangle}{TV_\epsilon(\phi)}\mathrm{div}\left(\frac{\nabla\phi}{\sqrt{\epsilon^2+|\nabla\phi|^2}}\right) = 0, & \text{in } \Omega \\ \vec{n}\cdot\nabla\phi = 0, & \text{on } \partial\Omega.\end{cases} \tag{2.59}$$

We can numerically determine ϕ by time-stepping

$$\begin{cases}\frac{\partial\phi}{\partial t} = \frac{\langle w,\phi\rangle}{TV_\epsilon(\phi)}\mathrm{div}\left(\frac{\nabla\phi}{\sqrt{\epsilon^2+|\nabla\phi|^2}}\right) + w, & \text{in } \Omega \\ \vec{n}\cdot\nabla\phi = 0, & \text{on } \partial\Omega\end{cases} \tag{2.60}$$

towards steady state as previously done with w in SO and SO Tight MHDM.

Supposing we can solve (2.60) for ϕ, the Euler-Lagrange equation for the refined problem (2.56) is modified from (2.48) only by the $*$-norm term, producing the time dependent ϵ-regularized equation

$$\begin{cases}\frac{\partial w}{\partial t} = -\frac{\phi}{TV_\epsilon(\phi)} + \lambda_k\alpha_k\mathrm{div}\left(\frac{\nabla(w+x_{k-1})}{|\nabla(w+x_{k-1})|}\right) - \lambda_k(1-f^\delta e^{-w-x_{k-1}}) & \text{in } \Omega \\ \frac{\partial w}{\partial\vec{n}} = 0 & \text{in } \partial\Omega.\end{cases} \tag{2.61}$$

Given the gradient descent problems (2.58),(2.61) associated with the Euler-Langrange equations of their respective minimization problems, we can construct two semi-implicit numerical schemes to iteratively solve for ϕ and w_k. We start by addressing (2.61) assuming we have the iterate ϕ^{n+1} and are seeking w^{n+1}. Subsequently, we will discuss determining ϕ^{n+1} given w^n.

The semi-implicit numerical scheme for (2.61) contains all the terms in the SO Tight MHDM update (2.51), with those associated with discretizing $\text{div}(\nabla w/|\nabla w|)$ exchanged for $-\phi/TV_\epsilon(\phi)$. It follows then that

$$\begin{aligned} w_{ij}^{n+1} = \frac{1}{1+\Delta t\lambda_k a_k c_{v^n}^{ij}} \cdot \Bigg\{ & w_{ij}^n - \Delta t \frac{\phi_{ij}^{n+1}}{TV_\epsilon(\phi^{n+1})} - \Delta t\lambda_k \left(1 - f_{ij}^\delta e^{-(w_{ij}^n + y_{k-1,ij})}\right) + \\ & \Delta t\lambda_k a_k \bigg(\chi[c_\epsilon, i, j]_{w^n}^{v^n} + \chi[c_\epsilon, i, j]_{y_{k-1}}^{v^n} - (y_{k-1})_{ij} c_{v^n}^{ij} \bigg) \Bigg\}, \end{aligned} \tag{2.62}$$

where

$$TV_\epsilon(\phi) = \sum_{k,l} \sqrt{\epsilon^2 + (D_x^+ \phi_{kl})^2 + (D_y^+ \phi_{kl})^2}, \tag{2.63}$$

is the update scheme for determining $w_k = w^\infty$ given ϕ^{n+1}. It remains to be shown how to determine the iterates ϕ^n from (2.60). Note that (2.60) has more non-linearity than (2.61), and we have two options in creating a semi-impicit method. Using the notation from Section 2.7.1, observe that

$$\left(\frac{\langle w, \phi \rangle}{TV_\epsilon(\phi)} \text{div} \left(\frac{\nabla \phi}{\sqrt{\epsilon^2 + |\nabla \phi|^2}} \right) \right)_{ij} = \frac{\sum_{k,l} w_{kl} \phi_{kl}}{TV_\epsilon(\phi)} \cdot \left(\chi[c_\epsilon, i, j]_\phi^\phi - \phi_{ij} c_\phi^{ij} \right).$$

So, (2.60) becomes

$$\begin{aligned}
\frac{\phi_{ij}^{n+1} - \phi_{ij}^{n}}{\Delta t} &= w_{ij}^n + \frac{\sum_{k,l} w_{kl}^n \phi_{kl}^n}{TV_\epsilon(\phi^n)} \cdot \left(\chi[c_\epsilon, i, j]_{\phi^n}^{\phi^n} - \phi_{ij}^n c_{\phi^n}^{ij}\right) \qquad (2.64)\\
&= w_{ij}^n + \frac{1}{TV_\epsilon(\phi^n)} \left(\sum_{k,l \neq i,j} w_{kl}^n \phi_{kl}^n + w_{ij}^n \phi_{ij}^n\right) \left(\chi[c_\epsilon, i, j]_{\phi^n}^{\phi^n} - \phi_{ij}^n c_{\phi^n}^{ij}\right)\\
&= w_{ij}^n + \frac{1}{TV_\epsilon(\phi^n)} \Bigg[\chi[c_\epsilon, i, j]_{\phi^n}^{\phi^n} \sum_{k,l \neq i,j} w_{kl}^n \phi_{kl}^n +\\
&\qquad \phi_{ij}^n \left(w_{ij}^n \chi[c_\epsilon, i, j]_{\phi^n}^{\phi^n} - c_{\phi^n}^{ij} \sum_{k,l \neq i,j} w_{kl}^n \phi_{kl}^n\right) - (\phi_{ij}^n)^2 c_{\phi^n}^{ij} w_{ij}^n\Bigg]. \qquad (2.65)
\end{aligned}$$

If we wish to create a semi-implicit scheme, the isolated ϕ_{ij}^n terms on the right-hand-side of the above expression can have time-index n swapped for $n+1$.

However, we have a choice. One can make the exchange solely within the parenthetical term on the right-hand-side of (2.64), creating scheme

$$\phi_{ij}^{n+1} = \frac{1}{1 + \Delta t c_{\phi^n}^{ij} \frac{\sum_{k,l} w_{kl}^n \phi_{kl}^n}{TV_\epsilon(\phi^n)}} \left\{\phi_{ij}^n + \Delta t \cdot \left(w_{ij}^n + \frac{\sum_{k,l} w_{kl}^n \phi_{kl}^n}{TV_\epsilon(\phi^n)} \cdot \chi[c_\epsilon, i, j]_{\phi^n}^{\phi^n}\right)\right\}. \qquad (2.66)$$

We make use of (2.66) for SO Refined MHDM.

However, one can also consider the form in (2.65) where the bracketed terms within the right-hand-side are quadratic in ϕ_{ij}^n. Promoting $\phi_{ij}^n \to \phi_{ij}^{n+1}$ requires a quadratic solve at each n:

$$\begin{aligned}
0 =& (\phi_{ij}^{n+1})^2 \underbrace{\Delta t c_{\phi^n}^{ij} \sum_{k,l \neq i,j} w_{kl}^n \phi_{kl}^n}_{A} +\\
& \phi_{ij}^{n+1} \underbrace{\left(TV_\epsilon(\phi^n) + \Delta t \left(c_{\phi^n}^{ij} \sum_{k,l \neq i,j} w_{kl}^n \phi_{kl}^n - w_{ij}^n \chi[c_\epsilon, i, j]_{\phi^n}^{\phi^n}\right)\right)}_{B} + \qquad (2.67)\\
& \underbrace{\left[-TV_\epsilon(\phi^n)\phi_{ij}^n - \Delta t \left(TV_\epsilon(\phi^n) w_{ij}^n + \chi[c_\epsilon, i, j]_{\phi^n}^{\phi^n} \sum_{k,l \neq i,j} w_{kl}^n \phi_{kl}^n\right)\right]}_{C}.\\
\Rightarrow& \phi_{ij}^{n+1} = \frac{-B \pm \sqrt{B^2 - 4AC}}{2A}. \qquad (2.68)
\end{aligned}$$

Remark 2.7.1. *For computational speed, the term* $\sum_{k,l\neq i,j} w^n_{kl}\phi^n_{kl}$ *need not be computed at each* (i,j) *pair. Instead, consider* $\sum_{k,l} w^n_{kl}\phi^n_{kl} - w^n\phi^n$ *is a new grid function, whose* (i,j) *element has value* $\sum_{k,l\neq i,j} w^n_{kl}\phi^n_{kl}$.

Denoting the output from (2.62) by `SO_Refined_MHDM`$(f^\delta, y_{k-1}, \phi^{n+1}, \Delta t, \lambda_k, a_k, \epsilon)$ and (2.66) (or (2.67)) by `Phi_Update`$(w^n, \phi^n, \Delta t, \epsilon)$, we are able to write the Shi Osher Refined MHDM algorithm as follows.

Algorithm 3 SO Refined MHDM

INPUT: noisy image $f^\delta = z \cdot \eta$, where η is some multiplicative noise and z is the original image.

OUTPUT: $x_{\texttt{numScales}}$, an approximation to z.

```
Initialize: $y_{-1} = 0$, $\lambda_0 = 0.01$, $a_0 = 1$, $\epsilon = 0.01$ (or some small constant)
Choose maxIter
for $k = 0, 1, 2, \ldots,$ numScales do
    Initialize: $w_k$ (see Sec. 2.7.9)
    Initialize: $\phi = 0$
    $a_k \leftarrow \frac{a_0}{(k+1)^{3/2}}$
    for n=0,1,2,..., maxIter do
        $\phi \leftarrow$ Phi_Update$(w_k, \phi, \Delta t, \epsilon)$        ▷ output of 2.66 (or 2.67)
        $w_k \leftarrow$ SO_Tight_MHDM$(f^\delta, y_{k-1}, \phi, \Delta t, \lambda_k, a_k, \epsilon)$        ▷ output of (2.62)
    end for
    $y_k \leftarrow w_k + y_{k-1}$
    $\lambda_{k+1} \leftarrow 3\lambda_k$
end for
return : $x_{\texttt{numScales}} = e^{y_{\texttt{numScales}}}$
```

A discussion of initializations can be found in Subsection 2.7.9.

2.7.4 AA MHDM

We extended the original AA model (2.9) to an MHDM method, as given in (2.21). We use a semi-implicit method to solve the Euler-Lagrange equation for (2.9),

$$\begin{cases} \frac{\partial u}{\partial t} = \operatorname{div}\left(\frac{\nabla u}{|\nabla u|}\right) - \lambda_k T^*\left(\frac{1}{T(ux_{k-1})} - \frac{f^\delta}{[T(ux_{k-1})]^2}\right) x_{k-1}, & \text{on } \Omega, \\ \nabla u \cdot \vec{n} = 0 & \text{on } \partial\Omega, \end{cases} \tag{2.69}$$

as formulated in [AA08]. The discretization details are omitted here due to similarity with the subsequent AA-log schemes.

2.7.5 AA-log MHDM with $TV(\log(u))$ penalty term

The AA-log model directly addresses images corrupted by blur and multiplicative noise, since now with blur, the substitution $w = \log u$ no longer produces a convex problem as was true for the SO formulation. We seek a multiscale solution $x_k = \prod_{j=0}^{k} u_j$ approximating z, where each u_k satisfies (2.23), and we have access to the corrupted data $f^\delta = (Tz)\eta$.

Assuming x_{k-1} is determined, we use the Euler-Lagrange equation associated with (2.23) to obtain a time dependent PDE to determine u_k,

$$\begin{cases} \frac{\partial u}{\partial t} = \operatorname{div}\left(\frac{\nabla u}{|u||\nabla u|}\right) + \frac{|\nabla u|}{u|u|} - \lambda_k T^*\left(\frac{1}{T(ux_{k-1})} - \frac{f^\delta}{[T(ux_{k-1})]^2}\right) x_{k-1}, & \text{on } \Omega \\ \nabla u \cdot \vec{n} = 0, & \text{on } \partial\Omega. \end{cases} \tag{2.70}$$

Remark 2.7.2. *Because of the several u and $|u|$ terms found in denominators within* (2.70)*, it is recommended to shift any input data away from zero before proceeding with a discretization.*

A numerical scheme for (2.70) proceeds in much the same manner as for the SO models, comprising of composed forward and backward finite difference operators for computing the divergence-of-gradient terms. We observe that the discretization

$$\left(\operatorname{div}\left(\frac{\nabla u}{|u||\nabla u|}\right)\right)_{ij} = \left(\chi[d_\epsilon, i, j]_u^u - u_{ij} d_u^{ij}\right) \tag{2.71}$$

isolates the terms linear in u_{ij}, where the notation d_u^{ij} and $d_\epsilon(\cdot)$ are introduced in (2.37) from Section 2.7.1. For $|\nabla u|/u|u|$, one can use centered or forward differences. Keeping in mind (2.71), if $|\nabla u|/u|u|$ is thought of as part of the data fidelity term, the discretization of (2.70) has direct analogue with (2.43). Isolating terms linear in u_{ij}^n and exchanging them for u_{ij}^{n+1} to make a semi-implicit scheme gives the update for the AA-log MHDM method:

$$\begin{aligned} u_{ij}^{n+1} = \frac{1}{1+\Delta t d_{u^n}^{ij}} \cdot \Bigg\{ & u_{ij}^n - \Delta t \lambda_k T^* \left[\frac{1}{T(u^n x_{k-1})} - \frac{f^\delta}{[T(u^n x_{k-1})]^2}\right]_{ij} \cdot x_{k-1,ij} \\ & + \Delta t \frac{\sqrt{D_x^0(u_{ij}^n)^2 + D_y^0(u_{ij}^n)^2}}{u_{ij}^n |u_{ij}^n|} + \Delta t \chi[d_\epsilon, i, j]_{u^n}^{u^n} \Bigg\}. \end{aligned} \tag{2.72}$$

Here, ϵ-regularization is implicitly implied in the $d_{u^n}^{ij}$ term. For a given x_{k-1}, (2.72) is iterated in n until convergence, returning $x_k = u^\infty \cdot x_{k-1}$, where by u^∞ we indicate the converged iterate.

To initialize the MHDM process, we set $x_{-1,ij} = 1$ and choose u^0 according to Subsection 2.7.9. Denoting the solution of (2.72) after $n = \texttt{maxIter}$ iterations by $\texttt{AA\_log\_MHDM}(f^\delta, x_{k-1}, \Delta t, \lambda_k, a_k, T, \epsilon, \texttt{maxIter})$, we present the basic workflow for using the AA-log model in the following algorithm.

Algorithm 4 AA-log MHDM

INPUT: noisy image $f^\delta = (Tz) \cdot \eta$, where η is some multiplicative noise, T is a blurring operator and z is the original image.

OUTPUT: $x_{\texttt{numScales}}$, an approximation to z.

```
Initialize: $x_{-1} = 1$, $\lambda_0 = 0.01$, and $\epsilon = 0.01$ (or some small constant)
Choose maxIter
for $k = 0, 1, 2, \ldots,$ numScales do
    $u_k \leftarrow \texttt{AA\_log\_MHDM}(f^\delta, x_{k-1}, \Delta t, \lambda_k, T, \epsilon, \texttt{maxIter})$    ▷ output of (2.72)
    $x_k \leftarrow u_k \cdot x_{k-1}$
    $\lambda_{k+1} \leftarrow 2\lambda_k$
end for
return : $x_{\texttt{numScales}}$
```

2.7.5.1 AA-log Tight MHDM

The tight version modifies the objective function by adding an additional regularizing term

$$u_k \in \arg\min_u TV(\log(u)) + \lambda_k a_k TV(\log(u x_{k-1})) + \lambda_k \int_\Omega \left(\log(T(u x_{k-1})) + \frac{f^\delta}{T(u x_{k-1})} \right). \tag{2.73}$$

The Euler-Lagrange equation for (2.73) gives the time dependent PDE (2.74) for u_k, which we can run to equilibrium:

$$\begin{cases} \dfrac{\partial u}{\partial t} = -\lambda_k T^* \left(\dfrac{1}{T(u x_{k-1})} - \dfrac{f^\delta}{[T(u x_{k-1})]^2} \right) x_{k-1} + \operatorname{div}\left(\dfrac{\nabla u}{|u||\nabla u|} \right) \\ \qquad + \dfrac{|\nabla u|}{u|u|} + \lambda_k a_k x_{k-1} \left[\operatorname{div}\left(\dfrac{\nabla (u x_{k-1})}{|u x_{k-1}||\nabla (u x_{k-1})|} \right) + \dfrac{|\nabla (u x_{k-1})|}{u x_{k-1}|u x_{k-1}|} \right] & \text{on } \Omega, \\ \nabla u \cdot \vec{n} = 0, & \text{on } \partial\Omega \end{cases} \tag{2.74}$$

To discretize (2.74), we introduce the notation $Z^n = u^n x_{k-1}$ and follow the same semi-implicit strategy from before (recalling notation in (2.37)), giving

$$u_{ij}^{n+1} = \frac{1}{1 + \Delta t(d_{u^n} + \lambda_k a_k x_{k-1,ij}^2 d_{Z^N}^{i,j})} \Bigg\{ u_{ij}^n - \Delta t \lambda_k T^* \left[\frac{1}{T(u^n x_{k-1})} - \frac{f^\delta}{[T(u^n x_{k-1})]^2} \right]_{ij} \cdot x_{k-1,ij}$$
$$+ \Delta t \left(\chi[d,i,j]_{u^n}^{u^n} + \frac{\sqrt{D_x^0(u_{ij}^n)^2 + D_y^0(u_{ij}^n)^2}}{u_{ij}^n |u_{ij}^n|} \right)$$
$$+ \Delta t \lambda_k a_k x_{k-1,ij} \left(\chi[d_\epsilon, i, j]_{Z^n}^{Z^n} + \frac{\sqrt{D_x^0(Z_{ij}^n)^2 + D_y^0(Z_{ij}^n)^2}}{Z_{ij}^n |Z_{ij}^n|} \right) \Bigg\}. \tag{2.75}$$

The AA-log Tight MHDM has the same initialization and parameters as the AA-log MHDM, except $a_0 = 1$, $a_k = \frac{a_0}{(k+1)^{3/2}}$, and $\lambda_{k+1} = 3\lambda_k$. Denoting the output of (2.75) after convergence (capped by $n =$ `maxIter` iterations) by `AA_log_Tight_MHDM`$(f^\delta, , x_{k-1}, \Delta t, \lambda_k, a_k, T, \epsilon,$ `maxIter`$)$, we give the AA-log Tight MHDM algorithm below.

Algorithm 5 AA-log Tight MHDM

INPUT: noisy image $f^\delta = (Tz) \cdot \eta$, where η is some multiplicative noise, T is a blurring operator and z is the original image.

OUTPUT: $x_{\texttt{numScales}}$, an approximation to z.

```
Initialize: $x_{-1} = 1$, $\lambda_0 = 0.01$, and $\epsilon = 0.01$ (or some small constant)
Choose maxIter
for $k = 0, 1, 2, \ldots,$ numScales do
    $a_k \leftarrow \frac{a_0}{(k+1)^{3/2}}$
    $u_k \leftarrow$ AA_log_Tight_MHDM($f^\delta, x_{k-1}, \Delta t, \lambda_k, a_k, T, \epsilon,$ maxIter)        ▷ output of (2.75)
    $x_k \leftarrow u_k \cdot x_{k-1}$
    $\lambda_{k+1} \leftarrow 3\lambda_k$
end for
return : $x_{\texttt{numScales}}$
```

2.7.6 AA-log Refined MHDM

The refinement of the AA-log Tight MHDM scheme includes the weaker $*$-norm in place of $TV(\log(\cdot))$ for the regularizing term $J(\cdot)$. The minimization problem reads as follows,

$$u_k \in \arg\min_u \|\log(u)\|_* + \lambda_k a_k TV(\log(ux_{k-1})) + \lambda_k \int_\Omega \left(\log(T(ux_{k-1})) + \frac{f^\delta}{T(ux_{k-1})}\right). \tag{2.76}$$

The Euler-Lagrange equation for (2.76),

$$\begin{aligned} \frac{\partial u}{\partial t} &= -\lambda_k T^* \left(\frac{1}{T(ux_{k-1})} - \frac{f^\delta}{(T[ux_{k-1}])^2}\right) x_{k-1} - \\ &\quad \frac{\phi}{uTV(\phi)} + \lambda_k \alpha_k x_{k-1} \left[\operatorname{div}\left(\frac{\nabla(ux_{k-1})}{|ux_{k-1}||\nabla(ux_{k-1})|}\right) + \frac{|\nabla(ux_{k-1})|}{ux_{k-1}|ux_{k-1}|}\right] \\ \nabla u \cdot \vec{n} &= 0, \quad \text{on } \partial\Omega, \end{aligned} \tag{2.77}$$

is modified from the tight formulation only by the $\|\log(\cdot)\|_*$ term. A discretization of (2.77) is determined by the same process as for the AA-log tight method combined with alternating time-stepping with a test-function ϕ, as done in the SO Refined MHDM (compare also to the refined version of the summed-MHDM in [LRV21]). To determine ϕ^{n+1} given ϕ^n and u^n, solve (2.66) (or (2.67)) with w^n exchanged for $\log(u^n)$. Then, compute u^{n+1} given ϕ^{n+1} via

$$\begin{aligned} u_{ij}^{n+1} = \frac{1}{1 + \Delta t \lambda_k a_k x_{k-1,ij}^2 d_{Z^N}^{i,j}} \Bigg\{ & u_{ij}^n - \Delta t \lambda_k T^* \left[\frac{1}{T(u^n x_{k-1})} - \frac{f^\delta}{[T(u^n x_{k-1})]^2}\right]_{ij} \cdot x_{k-1,ij} \\ & + \Delta t \left[\frac{-\phi_{ij}^{n+1}}{u_{ij}^n TV_\epsilon(\phi^{n+1})} + \lambda_k a_k x_{k-1,ij} \left(\chi[d_\epsilon, i, j]_{Z^n}^{Z^n} + \frac{\sqrt{D_x^0(Z_{ij}^n)^2 + D_y^0(Z_{ij}^n)^2}}{Z_{ij}^n |Z_{ij}^n|}\right)\right] \Bigg\}. \end{aligned} \tag{2.78}$$

As with AA-log Tight MHDM, $x_{-1} = 1$ and u^0 can be initialized as discussed in Section 2.7.9. The full AA-log Refined MHDM process is given in Algorithm 6.

Algorithm 6 AA-log Refined MHDM

INPUT: noisy image $f^\delta = (Tz) \cdot \eta$, where η is some multiplicative noise, T is a blurring operator and z is the original image.

OUTPUT: $x_{\texttt{numScales}}$, an approximation to z.

```
**Initialize:** $x_{-1} = 1$, $\lambda_0 = 0.01$, $a_0 = 1$, $\epsilon = 0.01$ (or some small constant)
Choose maxIter
**for** $k = 0, 1, 2, \ldots,$ numScales **do**
    Initialize: $u_k$ (see Sec. 2.7.9)
    Initialize: $\phi = 0$
    $a_k \leftarrow \frac{a_0}{(k+1)^{3/2}}$
    **for** n=0,1,2,..., maxIter **do**
        $\phi \leftarrow$ Phi_Update$(\log(u_k), \phi, \Delta t, \epsilon)$        ▷ output of 2.66 (or 2.67)
        $u_k \leftarrow$ AA_log_Refined_MHDM$(f^\delta, x_{k-1}, \phi, \Delta t, \lambda_k, a_k, T, \epsilon)$        ▷ output of (2.78)
    **end for**
    $x_k \leftarrow u_k \cdot x_{k-1}$
    $\lambda_{k+1} \leftarrow 3\lambda_k$
**end for**
**return** : $x_{\texttt{numScales}}$
```

2.7.7 TNV-log Models

We now address the TNV-log model (2.25), which consists of the TNV fidelity term, $\int(\frac{f}{Tu} - 1)^2$, augmented with a $TV(\log(u))$ penalty, and adapted to the multiscale form. This model handles blurring in addition to noise, so we assume we have access to the perturbed signal $f^\delta = (Tz) \cdot \eta$, and as before we seek a multiscale recovery, x_k, approximating z of the form $x_k = \prod_{j=0}^{k} u_j$.

The dynamical PDE from the Euler-Lagrange equations for (2.25) is

$$\begin{cases} \frac{\partial u}{\partial t} = \operatorname{div}\left(\frac{\nabla u}{|u||\nabla u|}\right) + \frac{|\nabla u|}{u|u|} - 2\lambda_k T^* \left[\frac{f^\delta}{[T(ux_{k-1})]^2}\left(1 - \frac{f^\delta}{T(ux_{k-1})}\right)\right] x_{k-1} & \text{in } \Omega, \\ \nabla u \cdot \vec{n} = 0 & \text{in } \partial\Omega. \end{cases} \tag{2.79}$$

This, differing from the AA-log MHDM by only its fidelity term, takes on a similar discretization and algorithm, with updates for u^{n+1} given by

$$\begin{aligned} u_{ij}^{n+1} = \frac{1}{1+\Delta t d_{u^n}^{ij}} \cdot \Bigg\{ & u_{ij}^n - 2\Delta t \lambda_k T^* \left[\frac{f^\delta}{[T(u^n x_{k-1})]^2}\left(1 - \frac{f^\delta}{T(u^n x_{k-1})}\right)\right]_{ij} \cdot x_{k-1,ij} + \\ & \Delta t \frac{\sqrt{D_x^0(u_{ij}^n)^2 + D_y^0(u_{ij}^n)^2}}{u_{ij}^n |u_{ij}^n|} + \Delta t \chi[d_\epsilon, i, j]_{u^n}^{u^n} \Bigg\}, \end{aligned} \tag{2.80}$$

with algorithm shown below. Initialization of u^0 is discussed in Section 2.7.9.

Algorithm 7 TNV-log MHDM

INPUT: noisy image $f^\delta = (Tz) \cdot \eta$, where η is some multiplicative noise, T is a blurring operator and z is the original image.

OUTPUT: $x_{\texttt{numScales}}$, an approximation to z.

```
**Initialize:** $x_{-1} = 1$, $\lambda_0 = 0.01$, and $\epsilon = 0.01$ (or some small constant)
Choose maxIter
**for** $k = 0, 1, 2, \ldots,$ numScales **do**
    $u_k \leftarrow$ TNV_log_MHDM($f^\delta, x_{k-1}, \Delta t, \lambda_k, T, \epsilon,$ maxIter)    ▷ output of (2.80)
    $x_k \leftarrow u_k \cdot x_{k-1}$
    $\lambda_{k+1} \leftarrow 2\lambda_k$
**end for**
**return** : $x_{\texttt{numScales}}$
```

2.7.8 TNV-log Tight

Likewise, with the TNV-log tight formulation the Euler-Lagrange equation is the same as that for the AA-log tight MHDM scheme (2.74), except with the appropriate fidelity term swapped for TNV, as was done in (2.79). The discretization is akin to that of the AA-log tight MHDM

$$\begin{aligned} u_{ij}^{n+1} =& \frac{1}{1+\Delta t(d_{u^n} + \lambda_k a_k x_{k-1,ij}^2 d_{Z^N}^{i,j})} \Bigg\{ u_{ij}^n - \\ & 2\Delta t \lambda_k T^* \left[\frac{f^\delta}{[T(u^n x_{k-1})]^2} \left(1 - \frac{f^\delta}{T(u^n x_{k-1})} \right) \right]_{ij} \cdot x_{k-1,ij} + \\ & \Delta t \left(\chi[d,i,j]_{u^n}^{u^n} + \frac{\sqrt{D_x^0 (u_{ij}^n)^2 + D_y^0 (u_{ij}^n)^2}}{u_{ij}^n |u_{ij}^n|} \right) + \\ & \Delta t \lambda_k a_k x_{k-1,ij} \left(\chi[d_\epsilon, i, j]_{Z^n}^{Z^n} + \frac{\sqrt{D_x^0 (Z_{ij}^n)^2 + D_y^0 (Z_{ij}^n)^2}}{Z_{ij}^n |Z_{ij}^n|} \right) \Bigg\}. \end{aligned} \tag{2.81}$$

For completeness, we include Algorithm 8 describing the TNV-log Tight MHDM method.

Algorithm 8 TNV-log Tight MHDM

INPUT: noisy image $f^\delta = (Tz) \cdot \eta$, where η is some multiplicative noise, T is a blurring operator and z is the original image.

OUTPUT: $x_{\texttt{numScales}}$, an approximation to z.

```
**Initialize:** $x_{-1} = 1$, $\lambda_0 = 0.01$, $a_0 = 1$, and $\epsilon = 0.01$ (or some small constant)
Choose maxIter
**for** $k = 0, 1, 2, \dots,$ numScales **do**
    $a_k \leftarrow \frac{a_0}{(k+1)^{3/2}}$
    $u_k \leftarrow$ TNV_log_Tight_MHDM($f^\delta, x_{k-1}, \Delta t, \lambda_k, a_k, T, \epsilon,$ maxIter)    ▷ output of (2.81)
    $x_k \leftarrow u_k \cdot x_{k-1}$
    $\lambda_{k+1} \leftarrow 3\lambda_k$
**end for**
**return** : $x_{\texttt{numScales}}$
```

2.7.9 Initializations

We take a moment to discuss initializing our MHDM schemes. Each of the proposed iterative processes minimize an energy as shown in (2.13), starting with some initialization u^0 (or w^0 in the case of SO) which, hopefully, is close to the minimizer. In this work, we propose two types of choices with the following motivation.

Fidelity minimizing initializations

One approach is to choose u^0 (or w^0) to minimize the fidelity $H(f^\delta, Tu^0 x_{k-1})$ without regard to the penalty term, which leads to non-constant initializations. For the SO MHDM schemes, if we aim to minimize $\int \left(f^\delta e^{-(w^0 + y_{k-1})} + (w^0 + y_{k-1}) \right)$, then the optimal initialization is $w^0 = \log(f^\delta) - y_{k-1}$. For the AA MHDM, AA-log MHDM and TNV-log schemes, the optimal initialization u^0 minimizing $H(f^\delta, Tux_{k-1})$ is u such that $Tux_{k-1} = f^\delta$, which amounts to solving a deblurring problem. For simplicity, we will just use $u^0 = f^\delta/x_{k-1}$.

Penalty minimizing initializations

On the other hand, we can choose u^0 to minimize the penalty $J(\cdot)$. For $J(\cdot) = TV(\log(\cdot))$ or $TV(\cdot)$, these would be constant functions whose values we can optimally choose to minimize the remaining data fidelity $H(f^\delta, Tux_{k-1})$. For the tight and refined schemes, the additional terms $a_k\lambda_k TV(\log(u^0 x_{k-1}))$ and $\|\log(u^0)\|_*$ do not necessarily vanish as was the case before. In the tight schemes, however, this does not affect the choice of a constant u^0 since $TV(\log(u^0x_{k-1})) = TV(\log(u^0) + \log(x_{k-1})) = TV(\log(x_{k-1}))$. However, $\|\log(u^0)\|_*$ is finite only if $\int_\Omega \log(u^0) = 0$ (see Sec. 3.1.4, Lemma 1 in [VL16]), which would force $u^0 = 1$ for a constant initialization, with no choice on optimizing the fidelity further.

We let $w^0 = \log\left(\frac{1}{|\Omega|}\int f^\delta e^{-y_{k-1}}\right)$ for the SO MHDM regular and tight schemes, and $w^0 = 0$ for the refined one. In the AA MHDM and AA-log MHDM regular and tight schemes, we take $u^0 = \frac{1}{|\Omega|}\int f^\delta/Tx_{k-1}$, while for the refined scheme we need $u^0 = 1$. Finally, we choose $u^0 = \frac{\|f^\delta/Tx_{k-1}\|^2_{L^2(\Omega)}}{\int f^\delta/Tx_{k-1}}$ for the TNV-log schemes.

In practice, we use the *penalty minimizing* initializations for the regular and tight SO MHDM, AA MHDM, AA-log MHDM and TNV-log schemes, while the *fidelity minimizing* initializations are considered for the refined SO MHDM and AA-log schemes. In the case of images with blur, we find that the *penalty minimizing* initialization $u^0 = 1$ works marginally better for the refined AA-log MHDM recovery.

Remark 2.7.3. *One could also try to balance the fidelity and penalty minimizing approaches. While less systematic, it would be tractable to initialize with a smoothed version of the fidelity minimizing initialization discussed above. For instance with AA-log MHDM, one could take* $u^0 = K * (f^\delta/x_{k-1})$ *where* K *is some Gaussian blurring kernel and* $*$ *indicates convolution. However, some experimenting or prior would be needed to properly choose the kernel widths.*

2.8 Numerical Results

In this section we examine the numerical results of multiscale image restoration in the case of multiplicative noise, using the regular, tight and refined schemes for the SO MHDM, AA MHDM, AA-log MHDM and TNV-log models. We also compare these against the TNV multiscale method [TNV08] and the DZ model [DZ13].

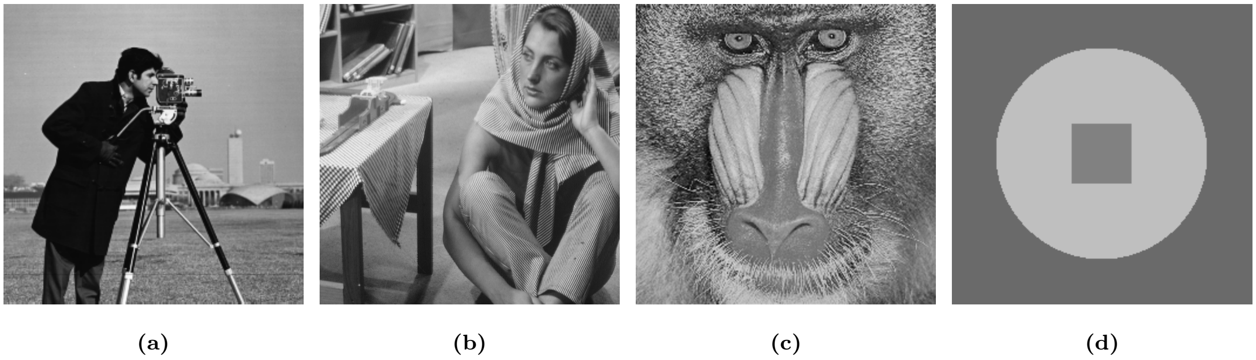


Figure 2.2: Original images: (a) "Cameraman", (b) "'Barbara", (c) "Mandrill", (d) "Geometry".

We chose three natural grayscale images with edges and textures ranging from smooth to detailed, and one synthetic test image as shown in Fig. 2.2. The original *Barbara* image comes from Alan Gersho's lab at the University of California, Santa Barbara; *Cameraman* is available through the `scikit-image` library's `skimage.data` test image dataset, and is

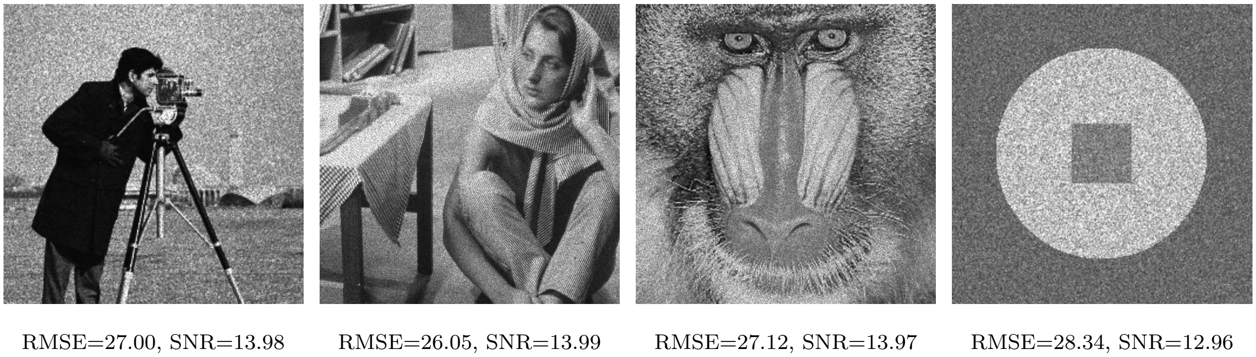


Figure 2.3: Images with multiplicative gamma noise.

released under CC0 by the photographer Lav Varshney; *Mandrill* is available through the University of Southern California Signal and Image Processing Institute's image database [SI]. The *Geometry* image is the author's. The models proposed in this work are aimed at removing multiplicative gamma noise and (possibly) blurring. Accordingly, we degrade the test images with gamma noise

$$g(x; a) = \frac{a^a}{\Gamma(a)} x^{a-1} e^{-ax} \mathbb{1}_{x \geq 0} \,,$$

with shape parameter $a = 25$ (mean 1 and standard deviation $1/\sqrt{a} = 0.2$), as shown in Fig. 2.3. Our choice of a serves to compare with the original Aubert and Aujol paper [AA08]. While the standard deviation for the Gamma noise used in [AA08] is not given, this noise profile produces signal-to-noise ratios near the sample images used therein. Similar or lower noise levels are also used in [UCK17]. Comprehensive comparisons are made at this noise level, but we also include some high and severe noise cases in Figures 2.12 and 2.18 which demonstrate the MHDM's ability to handle more aggressive corruption. Restorations are evaluated on the root-mean-squared-error (RMSE) and signal-to-noise ratio (SNR) between recovered x_k and original images z:

$$RMSE = \frac{\|x_k - z\|}{\sqrt{N}}, \quad SNR = 10 \times \log_{10} \left(\frac{\|z\|^2}{\|x_k - z\|^2} \right),$$

where $\| \cdot \|$ is the Euclidean norm and N is the total number of pixels in the image. We also examine how many multiscales are required for reconstruction, as well as the effectiveness of the stopping criteria.

The MHDM recoveries in the following sections are all performed with the same parameter values (as much as the models allow). We choose $\lambda_0 = 0.01$, then $\lambda_k = \lambda_0 q^k$ with $q = 2$ for the regular and $q = 3$ for the tight/refined schemes, as they satisfy the convergence prerequisites given in Section 2.5 and 2.6. For the tight and refined formulations, there is the additional parameter $a_k = \frac{a_0}{(1+k)^{3/2}}$ with $a_0 = 1$. If blurring is considered, we use a 5×5

Gaussian kernel T with variance 2. The parameter λ_k serves as the weight of the fidelity term, and larger λ_k leads to more textural details in u_k. In general, to avoid restoring noise in the image during the first few ranks of the hierarchy, a relatively small λ_0 should be chosen. As λ_k gradually increases, the texture within the image is restored at finer and finer scales. The multiscale hierarchical nature of the methods also increases the robustness of the recoveries; that is to say, it dilutes the influence of parameter changes on the image restoration. For example, if we choose a smaller λ_0, an acceptable restoration can be obtained with more hierarchy ranks (larger k). The time step size $\Delta t = 0.01$, gradient regularization $\epsilon = 0.01$, and the maximum number of iterations `maxIters`=1000. Using SO MHDM (EL) as an example, we give Fig. 2.4 to briefly mention the numerical convergence of gradient descent across multiscale indices k for `maxIters`=1000, where $E_k(w^n)$ is given by the energy to be minimized in (2.20) evaluated at the n-th iteration w^n. Lower k values may take more than 1000 steps to fully converge (c.f. $k = 6$ in Fig. 2.4), however, in aggregate across all k, convergence is achieved. Increasing `maxIters` or implementing a relative-energy-change exit condition has little impact on RMSE or SNR values at k_{min}, as shown in Table 2.4. We also observe similar behavior for the AA-log MHDM methods. Overall, the schemes exhibit robust nature due to the hierarchical construction of multiscale methods.

Table 2.1: k_{min}, RMSE and SNR values of SO MHDM (cameraman image) with different stopping criteria for gradient descent.

Stopping criteria	k_{min}	RMSE at k_{min}	SNR at k_{min}
`maxIters`=100	9	12.0223	21.0058
`maxIters`=1000	9	10.8598	21.8892
`maxIters`=10000	9	10.9102	21.8489
$\dfrac{\lvert E_k(w^{n+1}) - E_k(w^n)\rvert}{E_k(w^{n+1})} < 10^{-8}$	9	10.9159	21.8444

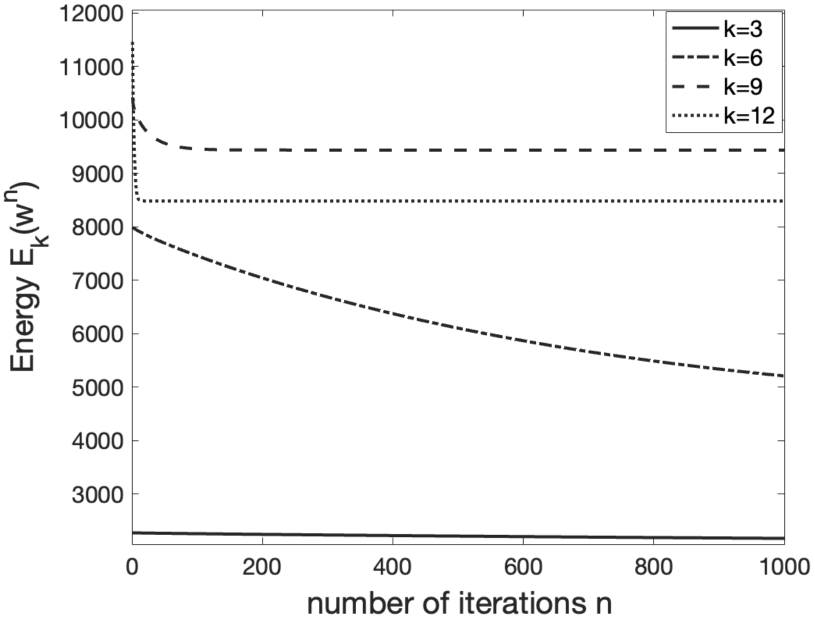


Figure 2.4: SO MHDM (cameraman image) energy versus iteration number during gradient descent, with multiscale numbers $k = 3, 6, 9, 12$.

2.8.1 Shi-Osher Models

Here and in the subsequent sections, to differentiate between the Euler-Lagrange PDE discretization and the ADMM recoveries, we will append (EL) or (ADMM) to the appropriate schemes, such as tight SO MHDM (ADMM) and refined SO MHDM (EL).

The purpose of the MHDM recovery is to retain textures of the original images at different scales while eliminating noise. As an example of a typical recovery, we show the progression of the multiscales for the “Cameraman” image in Fig. 2.5. The importance of correctly selecting a stopping point is clear, since too many multiscales will recover the majority of the noise as finer and finer levels of texture are added back into the image, while too few leave the image without textural details. This can be seen visually in Fig. 2.5 and numerically in Fig. 2.7a. The individual multiplicative scales are given in Fig. 2.6 and demonstrate how the images are built up. Recall each x_k in Fig. 2.5 is the product $\prod_{j=0}^{k} u_j$, with each u_j contributing features at different scales. Based on the multiplicative construction, light regions within u_j (higher pixel values) are promoted, while darker ones (lower pixel values) are suppressed relative to middle-toned regions in the image. As one can see in Fig. 2.6, the multiplicative process segments the large scale “cartoon” features at early scales before separating the texture and eventually noise within the image. Moreover, the proposed stopping index $k^*(\delta) = 9$ aligns with a sensible contribution from u_9 which adds back sufficient detail within the cameraman’s clothing and tripod before increasing the fine details—and noise—in the grass and background with u_j, $j > 9$. For a user with definite sense of the scale of features in the true data, the u_j pieces could provide a visual method of choosing k^*, whereby one increases k until u_k is emphasizing details at the preferred size.

Throughout the following discussion, we will look at both the multiscale index k_{min} which minimizes RMSE and the proposed stopping index $k^* = k^*(\delta)$ defined in (2.31), (2.36).

Recall that the noise parameter δ satisfies $H(f^\delta, Tz) \leq \delta^2$. In practice, we take $\delta^2 = H(f^\delta, Tz)$ for our numerical experiments.

In the case of SO MHDM, we transform the summed-MHDM reconstruction y_k back to the image approximation $x_k = e^{y_k}$, and then we compute k^* accordingly. The stopping criterion is determined by choosing k^* to be the penultimate multiscale to $H(f^\delta, x_k)/H(f^\delta, z)$ dropping below the value 1, as shown in Fig. 2.7b for the SO MHDM regular, tight and refined recoveries. In the “Cameraman” restoration, one has $k^* = k_{min}$ for regular and tight SO MHDM, while $k^*(\delta) = k_{min} + 1$ for the refined version. We emphasize that the tight and refined schemes require fewer multiscales to recover the images, and that Fig. 2.7a demonstrates the importance of the stopping criteria at preventing excess noise from being recovered. The SO MHDM (EL) recoveries are shown in Fig. 2.8, where we observe slight improvements transitioning from the regular to tight and finally to refined methods. Fig. 2.9

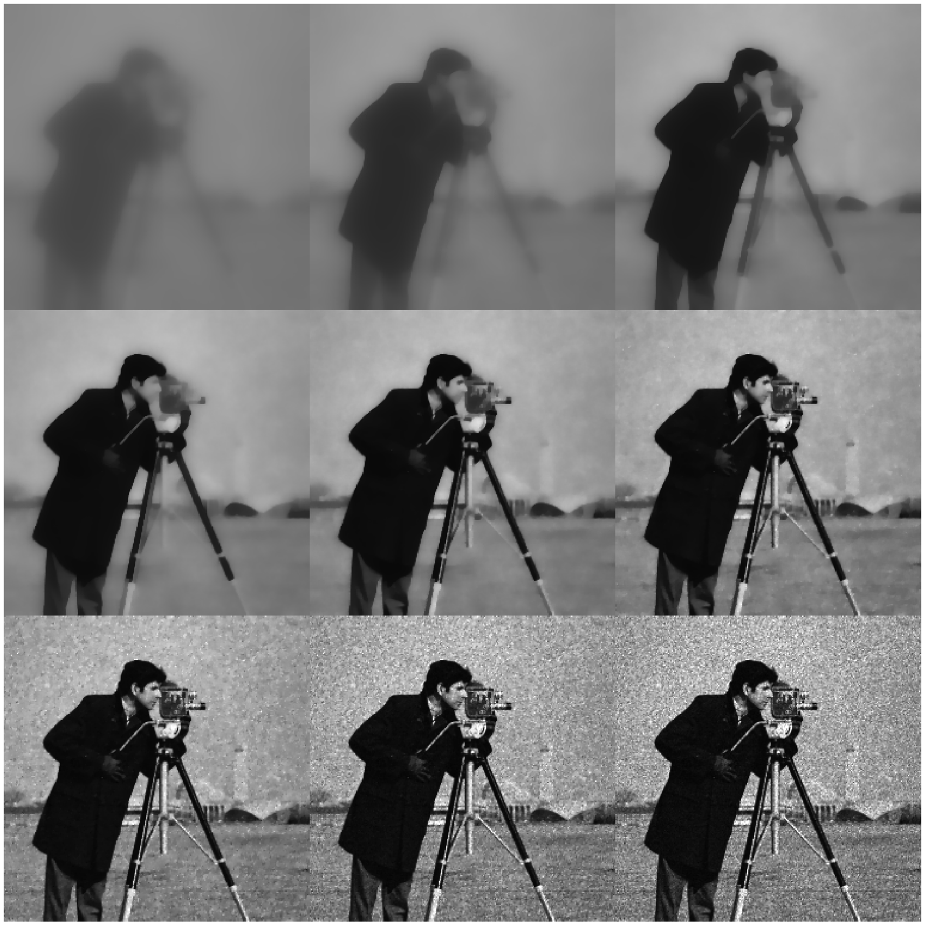


Figure 2.5: Multiscales x_k for $k = 4, 5, \ldots, 12$ in the SO MHDM recovery of the cameraman image. In this example, our proposed stopping index $k^*(\delta) = 9 = k_{min}$ is optimal (Refer to the diamond-labeled curves in Fig. 2.7).

exhibits a more textured image at the k_{min} and $k^*(\delta)$ scales. Again, the number of multiscales required in the SO MHDM (EL) reconstructions decreases going from regular to tight and then to refined SO MHDM in a uniform manner, consistent with Fig. 2.7. We also see increasingly improved restorations when moving from regular to refined schemes. Notably, with this more textured image the stopping index $k^*(\delta)$—which is generally within one step of k_{min}—produces restorations which are visually very close to the optimal ones.

However, the difference between k_{min} and $k^*(\delta)$ can significantly affect the restoration

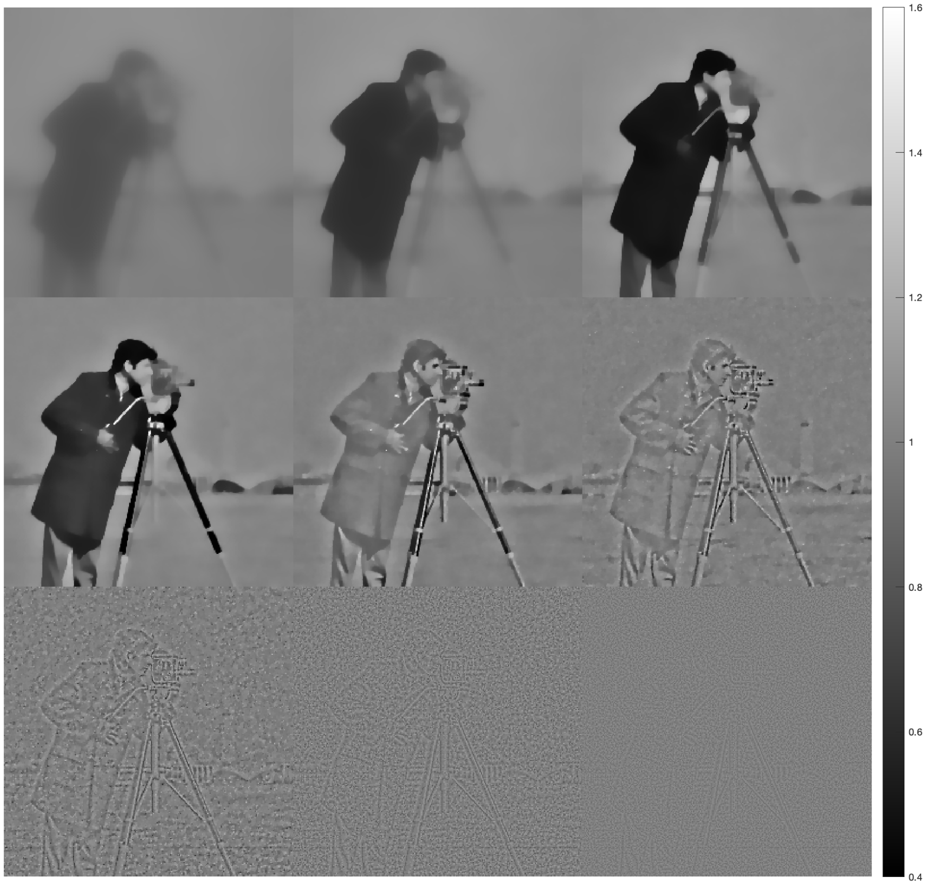
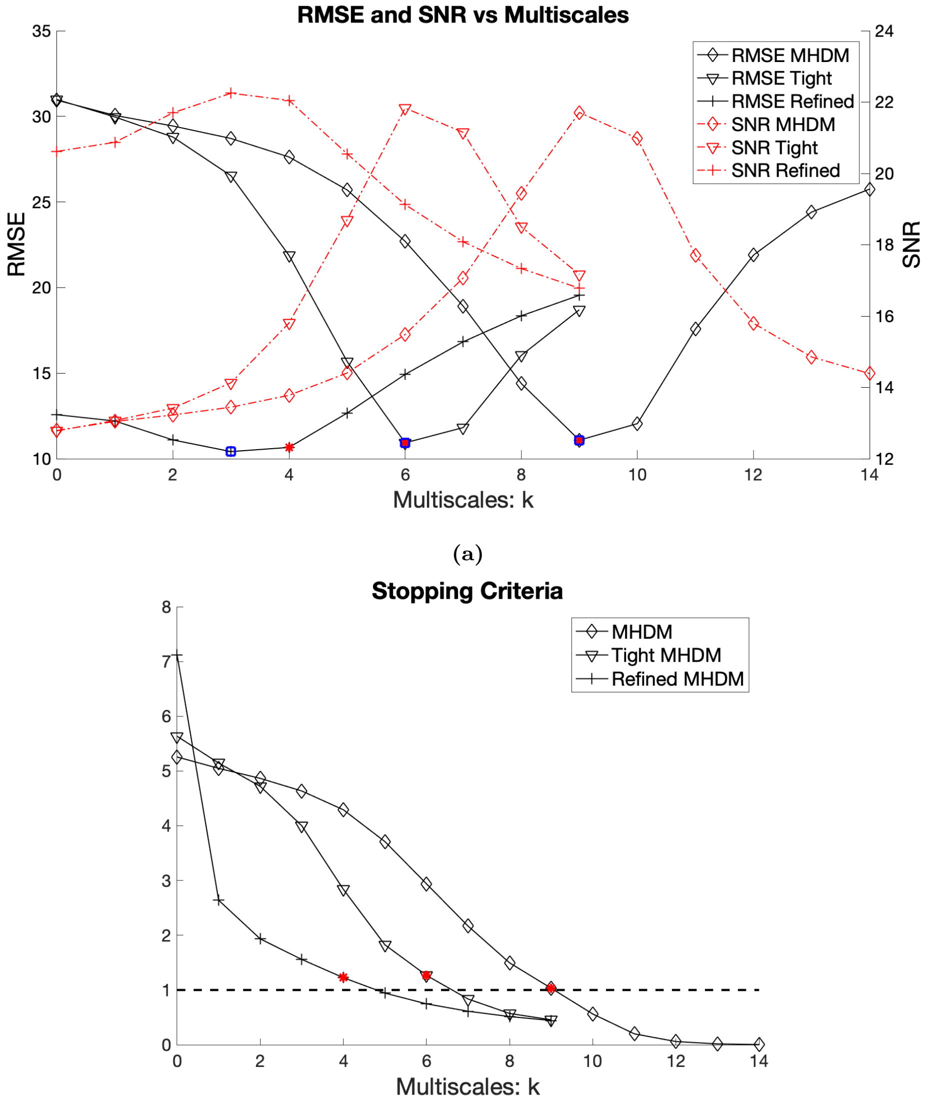


Figure 2.6: Multiplicative scales $u_j = e^{y_j}$ from the SO MHDM recovery of the cameraman image. The product $\prod_{j=0}^{k} u_j =: x_k$ constructs the multiscales shown in Fig. 2.5. These are displayed on the interval $[0.4, 1.6]$ for increased contrast—true range $[0.22, 1.63]$—and shown with a common legend.

in some cases—see "Geometry" in second column of Fig. 2.10, which compares recoveries obtained from the SO MHDM ADMM and EL approaches. For this low texture image, the SO MHDM (ADMM) restorations are significantly better than the EL derived counterparts. We also take a moment to mention that the SO MHDM recoveries in Fig. 2.10 preserve the mean image intensity. For models that make use of the logarithm to transform multiplicative noise into additive noise, there is a downward shift in the mean intensity of the recoveries, as explained in [AA08]. Our multiscale method effectively eliminates this mean intensity shift.



(a)

(b)

Figure 2.7: (a) RMSE and SNR versus multiscales index across methods when restoring the "cameraman" image. The optimal multiscale index k_{min} for each method is shown as the blue square and the stopping criteria k^* are given by red asterisks. (b) The stopping criteria k^* (shown as red asterisks) are the maximal k before $H(f^\delta, x_k)/H(f^\delta, z) \geq \tau > 1$ (for SO MHDM) or $(H(f^\delta, x_k) + a_k\lambda_k J(x_k))/H(f^\delta, z) \geq \tau > 1$ (for tight and refined SO MHDM) is no longer satisfied, as indicated by crossing under the horizontal dotted line in (b).

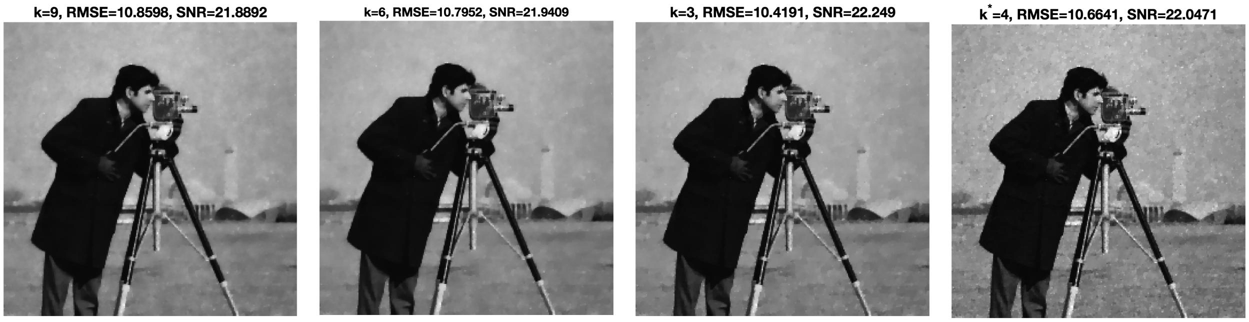

k=9, RMSE=10.8598, SNR=21.8892
k=6, RMSE=10.7952, SNR=21.9409
k=3, RMSE=10.4191, SNR=22.249
k^*=4, RMSE=10.6641, SNR=22.0471

Figure 2.8: SO MHDM (EL) image recoveries. From left to right: regular, tight, and refined restorations at k_{min}, and finally, the refined restoration at k^*. Recall that $k^* = k_{min}$ for regular and tight recoveries.

For the remaining higher-texture images, detailed recovery comparisons across all the SO MHDM models will be discussed in Section 2.8.7. In summary, the best restorations (lowest RMSE and highest SNR) among the SO MHDM models are tight SO MHDM (ADMM) for 'Cameraman' and 'Geometry', and refined SO MHDM (EL) for 'Barbara' and 'Mandrill'. This confirms that the refined version is suitable for recovering images with more texture.

2.8.2 AA-log Models: Denoising

In contrast to the SO models, the AA MHDM and AA-log MHDM models address the noise directly, building a multiscale restoration through multiplicative decompositions. They are also well equipped to handle blurring, and perform comparably to SO in the regular and tight formulations. The AA MHDM method behaves similarly to the AA-log schemes (see Table 2.2), so the majority of discussion is dedicated to the latter. We note that with the same timestep as the AA-log scheme, a threshholding step to ensure that the iterates continue to satisfy $\inf_\Omega f^\delta \leq x_k \leq \sup_\Omega f^\delta$ helps with numerical stability.

In Fig. 2.11 we give the AA-log MHDM restorations, and note the reduced performance

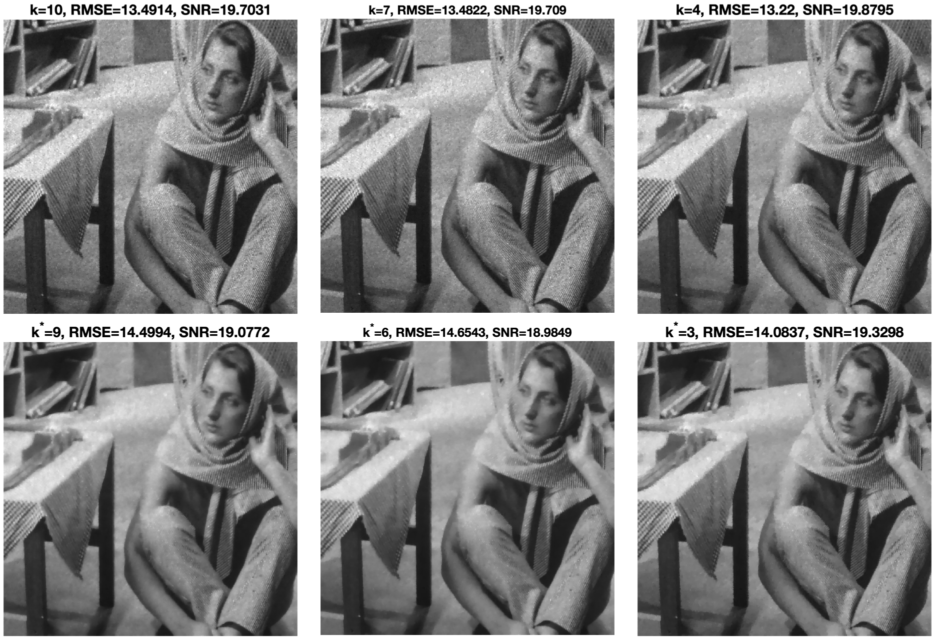

k=10, RMSE=13.4914, SNR=19.7031
k=7, RMSE=13.4822, SNR=19.709
k=4, RMSE=13.22, SNR=19.8795
k*=9, RMSE=14.4994, SNR=19.0772
k*=6, RMSE=14.6543, SNR=18.9849
k*=3, RMSE=14.0837, SNR=19.3298

Figure 2.9: From left to right: the regular, tight and refined SO MHDM (EL) recoveries. Row one is the k_{min} restoration while row two is the k^* recovery (if $k^* \neq k_{min}$).

of the refined method, specifically on the smooth “Geometry” image, is likely due to the weaker $\|\log(u)\|_*$ penalty. As a result, it may provide insufficient regularization to remove adequate noise, especially in smoother images. For the regular and tight formulations, the AA-log MHDM scheme produces better recoveries than its SO MHDM (EL) counterparts for the “Geometry” image. We also include the tight k^* recovery in Fig. 2.11, which visually is a better restoration despite not obtaining the lowest RMSE.

Worth remarking is the ability of the AA-log MHDM methods to recover corners and edges in the “Geometry”, even with very high noise levels. This is shown in Fig. 2.12, which compares the original AA recovery to the AA-log MHDM tight restoration of a severely noise-degraded image (gamma noise $g(x;1)$, standard deviation 1), as tested in Figures 2 and 3 from [AA08].

Remark 2.8.1. *As noted in [TNV08], the presence of* $\log(Tu)$ *and* f^δ/Tu *terms in the*

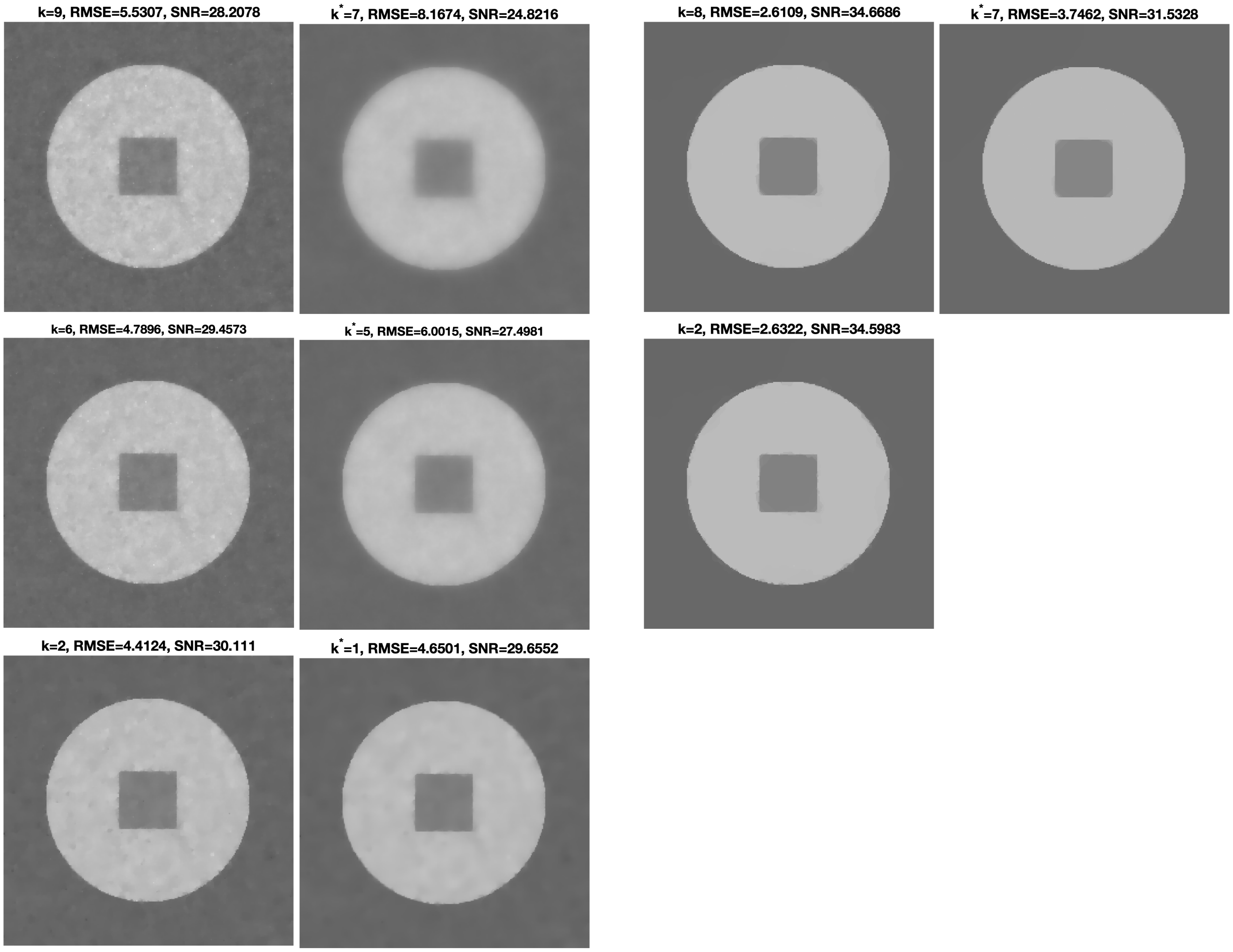

k=9, RMSE=5.5307, SNR=28.2078
k*=7, RMSE=8.1674, SNR=24.8216
k=8, RMSE=2.6109, SNR=34.6686
k*=7, RMSE=3.7462, SNR=31.5328
k=6, RMSE=4.7896, SNR=29.4573
k*=5, RMSE=6.0015, SNR=27.4981
k=2, RMSE=2.6322, SNR=34.5983
k=2, RMSE=4.4124, SNR=30.111
k*=1, RMSE=4.6501, SNR=29.6552

Figure 2.10: SO MHDM EL (left) and ADMM (right) restored images. Rows 1, 2 and 3 are the regular, tight and refined model recoveries, respectively. Columns 1 and 3 are the k_{min} restorations while columns 2 and 4 are the recoveries at the stopping criterion index k^* (if $k^* \neq k_{min}$).

AA-log MHDM models require the images to take strictly positive pixel values. Accordingly, images should be shifted away from zero, processed, and then shifted back appropriately when near-black pixels are expected.

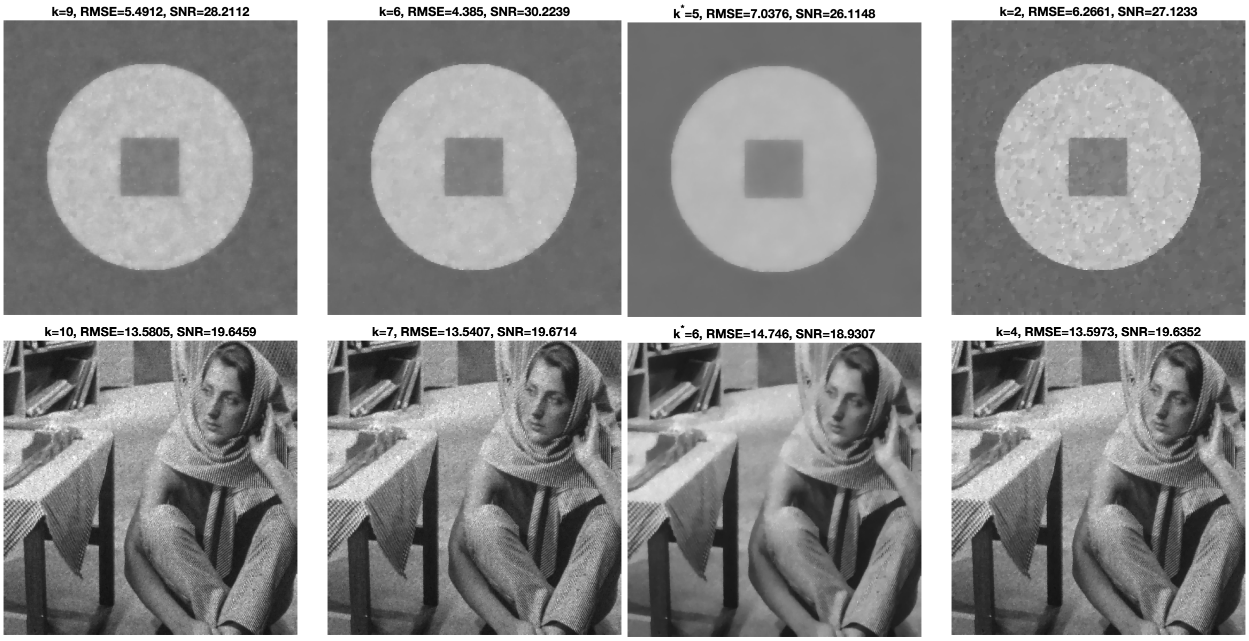


Figure 2.11: AA-log MHDM denoised images. Columns 1, 2, and 4 are the k_{min} regular, tight and refined restorations, respectively. Columns 3 is the tight recovery at the stopping criterion k^*.

2.8.3 AA-log Models: Denoising-Deblurring

One of the primary advantages of the AA-log MHDM models is handling deblurring in addition to denoising. Figure 2.13 gives the blurry, noisy counterparts of the test images, and Fig. 2.14 shows the AA-log MHDM recoveries. All images are blurred with a 5×5 Gaussian filter with standard deviation $\sqrt{2}$. The method effectively sharpens edges (see "Cameraman's" jacket and "Geometry") while maintaining texture (the tablecloth in "Barbara" and whiskers in "Mandrill"). The refined version with its weaker $*$-norm suffers from greater numerical instability, exhibiting a slight drop from the tight version in SNR.

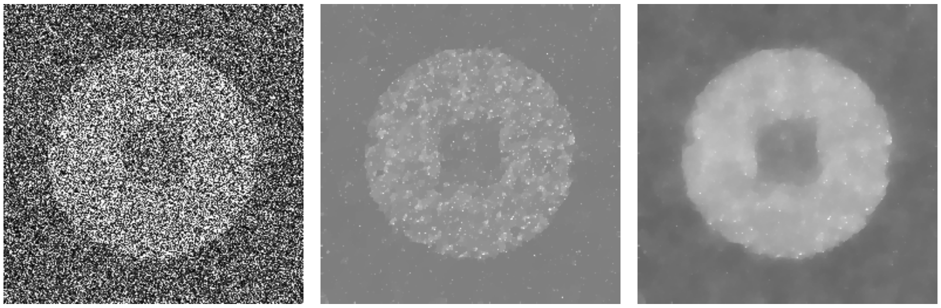
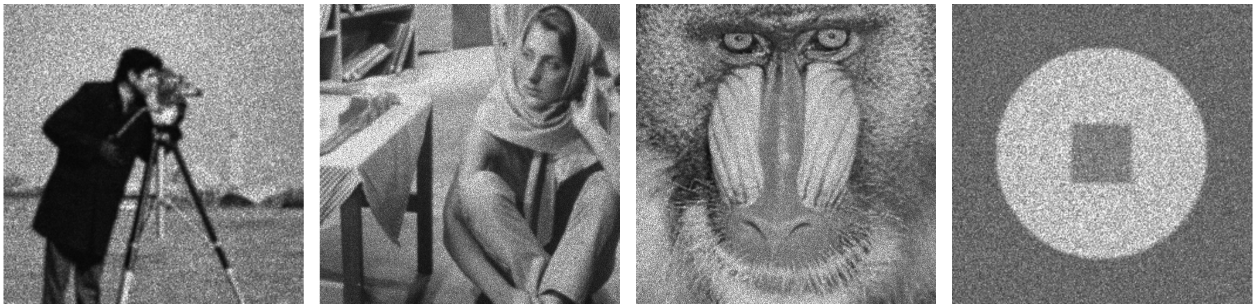


Figure 2.12: Recoveries from severe noise (standard deviation 1). From left to right: noisy image ($SNR = -0.079$), original AA model [AA08] ($SNR = 13.42$), and the AA-log MHDM tight recovery ($k_{min} = 4$, $SNR = 20.22$).



RMSE= 30.58, SNR=12.90 RMSE=30.06, SNR=12.74 RMSE=29.50, SNR=13.24 RMSE=28.84, SNR=13.86

Figure 2.13: Images degraded with multiplicative gamma noise and Gaussian blur. Blurring from a 5×5 filter with standard deviation $\sqrt{2}$. Noise as before with previous images.

2.8.4 TNV-log Models: Denoising

The TNV-log model is an adaptation of the TNV model by using the $TV(\log(u))$ penalty. Recall that TNV is a multiscale procedure based on [RLO03], which aimed to recover a degraded image $f = u \cdot \eta$ by minimizing $TV(u)$ subject to the constraints $\int f/u = 1$ (mean) and $\int (f/u - 1)^2 = \sigma^2$ (variance). Notice that no assumptions are required on the noise distribution (besides its mean and variance), so this method is suitable for more general multiplicative noise restorations. TNV [TNV08] dropped the mean constraint and converted the result to a multiscale method. We follow this lead while adding the contribution of a mod-

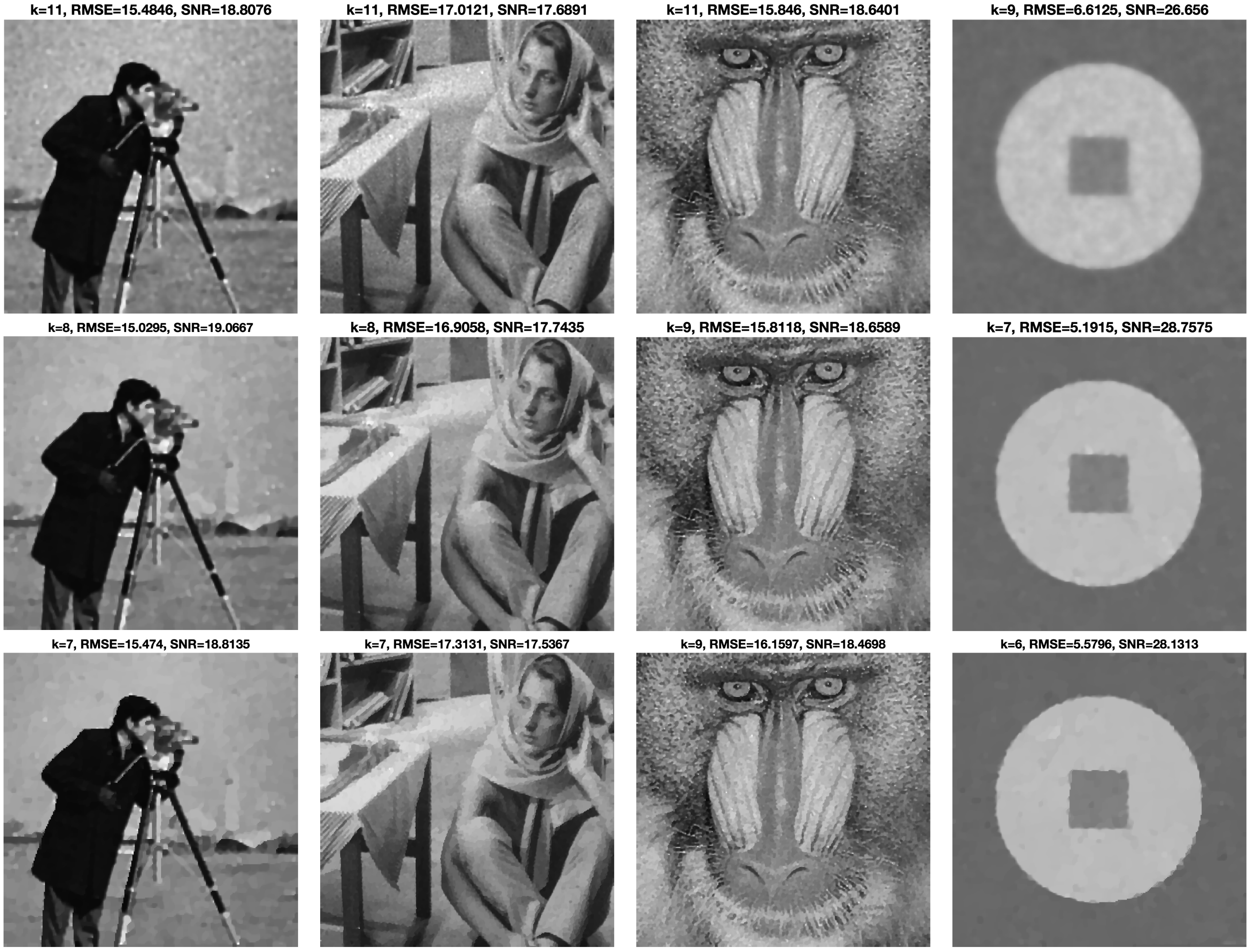

k=11, RMSE=15.4846, SNR=18.8076
k=11, RMSE=17.0121, SNR=17.6891
k=11, RMSE=15.846, SNR=18.6401
k=9, RMSE=6.6125, SNR=26.656
k=8, RMSE=15.0295, SNR=19.0667
k=8, RMSE=16.9058, SNR=17.7435
k=9, RMSE=15.8118, SNR=18.6589
k=7, RMSE=5.1915, SNR=28.7575
k=7, RMSE=15.474, SNR=18.8135
k=7, RMSE=17.3131, SNR=17.5367
k=9, RMSE=16.1597, SNR=18.4698
k=6, RMSE=5.5796, SNR=28.1313

Figure 2.14: AA-log denoised-deblurred images. Rows one, two and three are the regular, tight and refined AA-log MHDM models.

ified penalty. We omit a refined scheme because of the reduced performance from the weaker regularization $\| \log(\cdot)\|_*$, but there is hope, though, that other types of discretizations might provide better results in the refined case. Figure 2.15 shows the TNV-log recoveries. While the method does not outperform the multiscale methods designed for gamma distributed noise, TNV-log performs well against the DZ model on highly textured images.

The paper [AA08] pointed out that empirically the RO model did not preserve the average image intensity, with the recoveries shifting to lighter—higher mean—values. We check the mean intensity of the TNV-log "Geometry" restorations and observe a similar, yet reduced trend, with only minor upward shifts in the average intensity of both the regular (+3.3) and tight (+5.8) methods compared with the original image (an 8-bit grayscale image).

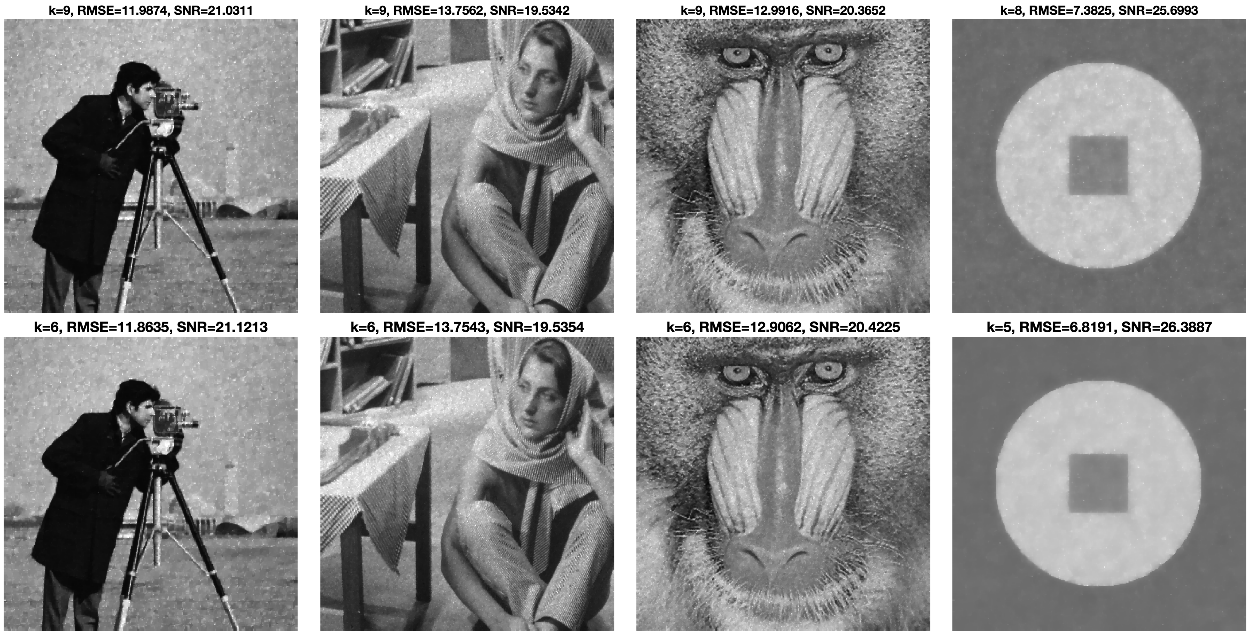

k=9, RMSE=11.9874, SNR=21.0311
k=9, RMSE=13.7562, SNR=19.5342
k=9, RMSE=12.9916, SNR=20.3652
k=8, RMSE=7.3825, SNR=25.6993
k=6, RMSE=11.8635, SNR=21.1213
k=6, RMSE=13.7543, SNR=19.5354
k=6, RMSE=12.9062, SNR=20.4225
k=5, RMSE=6.8191, SNR=26.3887

Figure 2.15: TNV-log denoised images. Row one is the regular and row two is the tight recoveries.

2.8.5 TNV-log Models: Denoising-Deblurring

The TNV-log model can also tackle denoising-deblurring, as shown in Fig. 2.16. We see that the performance is not as good as with the AA-log methods. For example, the edges are not that well preserved and the "Cameraman" image has more noise than the AA-log in Fig. 2.14. Yet, this is expected for a technique not tailored for gamma noise.

2.8.6 Sensitivity to Initialization

To aid the reader in using the MHDM procedure, we briefly discuss a heuristic for initializing each method and provide some intuition for these choices. The AA-log MHDM methods are

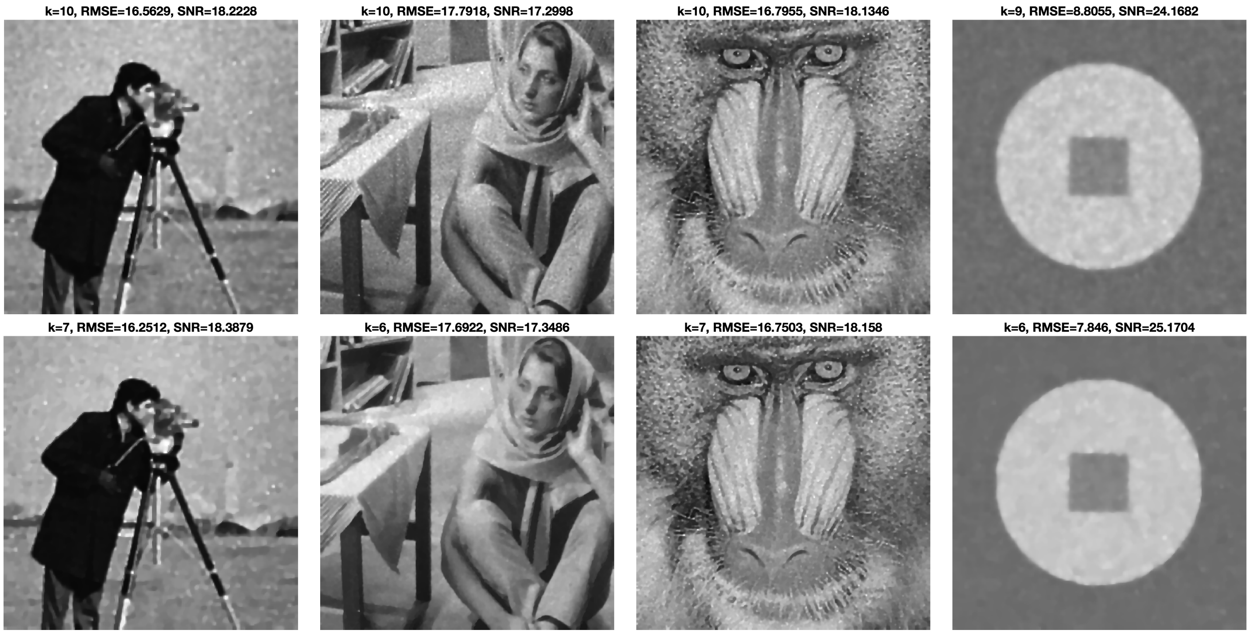

k=10, RMSE=16.5629, SNR=18.2228
k=10, RMSE=17.7918, SNR=17.2998
k=10, RMSE=16.7955, SNR=18.1346
k=9, RMSE=8.8055, SNR=24.1682
k=7, RMSE=16.2512, SNR=18.3879
k=6, RMSE=17.6922, SNR=17.3486
k=7, RMSE=16.7503, SNR=18.158
k=6, RMSE=7.846, SNR=25.1704

Figure 2.16: TNV-log denoised-deblurred images: regular (row 1) and tight (row 2) recoveries.

not convex, and consequently initialization can affect recoveries. We find that the penalty minimizing initializations, which generally are smoother (see Subsection 2.7.9), work well for the regular and tight AA-log MHDM schemes, while the fidelity minimizing initialization f^δ/x_{k-1} produces poor restorations and appears to be near a suboptimal local minimum. The refined AA-log MHDM method, however, does well with the less smooth fidelity minimizing initializations, likely because the $*$-norm is finite only for zero-mean functions (see Subsection 2.7.9).

The SO MHDM methods are convex, and as a result exhibit much less dependence on initialization. All SO methods perform well with both the penalty and fidelity minimizing initializations. We do see a slight improvement following the convention employed for the AA-log MHDM and use this pattern for the results in this work.

Like the AA-log MHDM methods, the TNV-log schemes are not convex. We observe a similar trend to the AA-log method's dependence on initialization, preferring the smoother penalty minimizing initialization for the TNV-log regular and tight schemes.

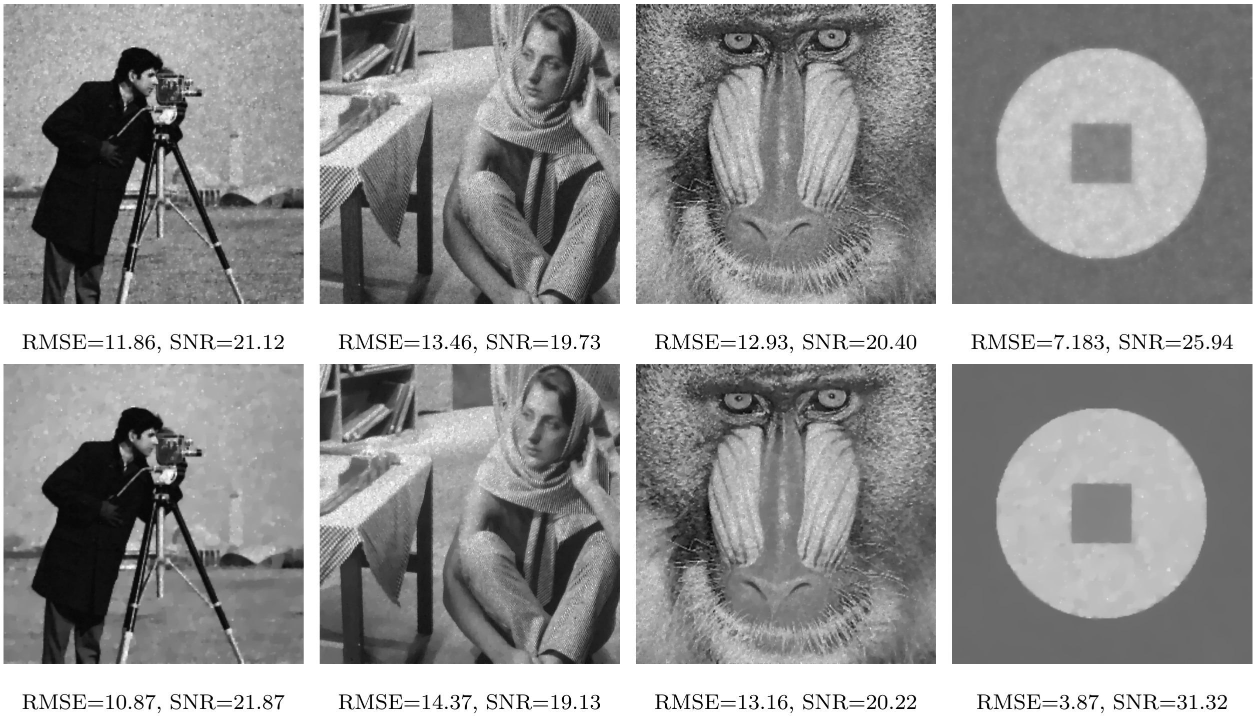
2.8.7 Denoising Comparisons

Having seen the individual recoveries across the proposed methods, we now compare them against one another, and also with the existing TNV [TNV08] and DZ [DZ13] models for both denoising and deblurring tasks. For DZ, we choose $\alpha = 16$ in (2.12) for all images and select the best weight λ by means of a grid search (see Fig. 2.17).

We present the TNV and the DZ recoveries in Fig. 2.17. The DZ model recovers the smoother, more cartoon image features well, but fails to capture the textural details in the way a multiscale method like the TNV model does. However, the latter suffers from restoring perhaps too much noise within smoother images. We examine our proposed MHDM schemes against the TNV and DZ models. To easily compare across all models, we list the SNR values of the denoising restorations at k_{min} and k^* in Table 2.2. We note that the refined SO MHDM recovery performs best on images with more detail and texture ("Barbara" and "Mandrill"), while the ADMM schemes recover best those with larger smooth regions ("Cameraman" and "Geometry").

Figure 2.19 gives a detailed crop of each method's recovery for the test images. The multiscale reconstruction's ability to recover greater texture is clear in the "Barbara" recovery, where one can see that the DZ recovery flattens and mutes the details. The SO MHDM (ADMM) schemes are very effective at removing noise in smooth regions ("Geometry"). In a more textured image such as the "Mandrill", we remark the importance of the tight adjustment preventing over-smoothing, as for example, is seen with the "Mandrill" image's nose for SO MHDM (ADMM).

For greater comparison, we also note that the MHDM schemes handle higher noise levels quite well ($g(x; 10)$, standard deviation 0.32), as shown for a few methods in Fig. 2.18, in



RMSE=11.86, SNR=21.12 RMSE=13.46, SNR=19.73 RMSE=12.93, SNR=20.40 RMSE=7.183, SNR=25.94

RMSE=10.87, SNR=21.87 RMSE=14.37, SNR=19.13 RMSE=13.16, SNR=20.22 RMSE=3.87, SNR=31.32

Figure 2.17: Row one: TNV recoveries at $k_{min} = 9$ for all images except "Geometry", for which $k_{min} = 8$. Row two: DZ model denoised images. Parameters for the DZ model recoveries (refer to (2.12)): (Cameraman) $\lambda = 0.06$, (Barbara) $\lambda = 0.05$, (Mandrill) $\lambda = 0.05$, and (Geometry) $\lambda = 0.13$. For all images, $\alpha = 16$.

comparison with the DZ recovery. Importantly, SNR and $RMSE$ are improved, and texture is retained throughout the image when compared with the DZ method.

2.8.8 Deblurring-Denoising Comparisons

Table 2.3 lists comparisons for denoising-deblurring. The AA-log MHDM models perform better overall. The DZ model greatly over-smooths details compared to the multiscale schemes, especially in the detailed crops of Figures 2.20–2.21, and naturally, it performs well on "Geometry". We emphasize the improvement of the TNV and the TNV-log methods over the DZ model on textured images in either SNR or the visual metric, demonstrating

Table 2.2: SNR values from various denoising recoveries at the minimizing indices k_{min} and the stopping criteria $k^*(\delta)$. Bold entries are the maximum of their respective columns.

	Cameraman		Barbara		Mandrill		Geometry	
SNR at	k_{min}	k^*	k_{min}	k^*	k_{min}	k^*	k_{min}	k^*
SO MHDM (EL)	21.89	21.89	19.70	19.08	20.38	20.06	28.21	24.82
SO Tight (EL)	21.94	21.94	19.71	18.98	20.37	19.89	29.46	27.50
SO Refined (EL)	22.25	22.05	**19.88**	19.33	**20.73**	20.32	30.11	29.66
SO MHDM (ADMM)	22.16	**22.16**	19.31	19.12	19.93	19.93	**34.67**	31.53
SO Tight (ADMM)	**22.31**	21.97	19.74	**19.36**	20.46	20.19	34.60	**34.60**
AA MHDM	21.76	19.45	19.53	19.14	20.21	20.05	28.21	24.77
AA-log MHDM	21.74	21.74	19.65	19.04	20.35	20.02	28.21	24.83
AA-log Tight	21.48	21.48	19.67	18.93	20.37	19.79	30.22	26.11
AA-log Refined	21.68	21.19	19.64	19.21	20.55	**20.41**	27.12	13.81
TNV-log	21.03	20.13	19.53	17.97	20.37	19.02	25.70	23.17
TNV-log Tight	21.12	12.61	19.54	17.35	20.42	17.89	26.39	21.02
TNV	21.12		19.73		20.40		25.94	
DZ	21.87		19.13		20.22		31.32	

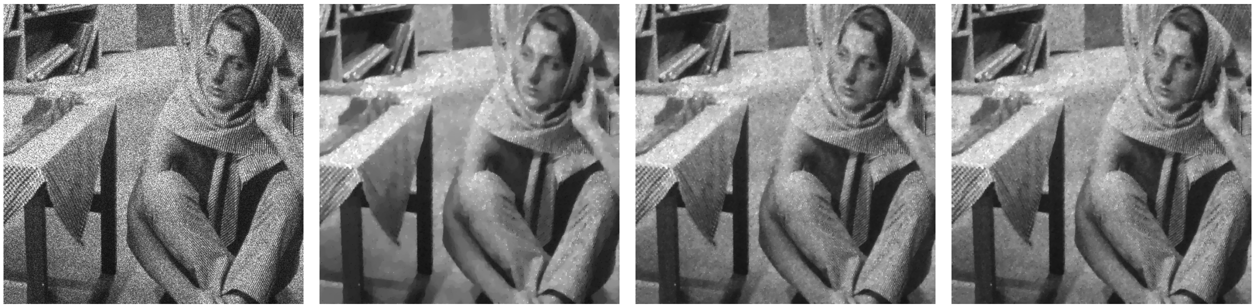


Figure 2.18: Recoveries from high noise (standard deviation 0.32). From left to right: noisy image ($RMSE = 41.25$, $SNR = 10.00$), DZ model ($RMSE = 17.5233$, $SNR = 17.44$, $\lambda = 0.08$), the SO MHDM tight ($k_{min} = 6$, $RMSE = 16.37$, $SNR = 18.03$) and AA-log MHDM tight recoveries ($k_{min} = 6$,$RMSE = 16.38$, $SNR = 18.02$).

the effectiveness of multiscale recoveries.

2.9 Conclusion

In this chapter, we introduced several multiscale hierarchical decomposition methods for images degraded by multiplicative noise which are able to retain texture and image features at different scales while reducing noise. We demonstrated that the fidelity terms decrease monotonically with increasing hierarchical depth, and propose an effective stopping criterion which limits restoring excess noise. Additionally, we considered extensions of the multiplicative MHDM (the so-called tight and refined versions), which are shown theoretically to push the regularity of recoveries to match that of the original image and empirically demonstrate better convergence properties of the iterates. The AA MHDM and AA-log MHDM methods are aimed specifically at gamma noise, and additionally handle deblurring tasks quite well, outperforming the existing DZ and TNV models in our tests. The convex SO MHDM models are quite robust with respect to initialization. They behave exceptionally well for images with smooth regions and still prevent over-smoothing on regions with oscillating patterns

Table 2.3: SNR values for restoring noisy-blurred images at the k_{min}. Bold entries are the maximums of their respective columns.

SNR at k_{min}	Cameraman	Barbara	Mandrill	Geometry
AA MHDM	18.81	17.69	18.65	26.45
AA-log MHDM	18.81	17.69	18.64	26.66
AA-log Tight	**19.07**	**17.74**	**18.66**	**28.76**
AA-log Refined	18.81	17.54	18.47	27.12
TNV-log	18.22	17.30	18.13	24.17
TNV-log Tight	18.39	17.35	18.16	25.17
TNV	18.26	17.32	18.17	24.23
DZ	17.14	17.36	17.24	28.32

when implemented with ADMM, while the Euler-Lagrange method retains slightly more details in very textured images. Finally, we consider the TNV-log method which handles blurring and noise without assuming a specific distribution for the noise (e.g. gamma). Accordingly, it is suitable for images corrupted by more general multiplicative noises. While TNV-log does not top the other MHDM methods when tested on gamma noise corrupted samples, it still outperforms the DZ model when deblurring and denoising more textured images, while continuing to maintain fine-scaled features. Collectively, these MHDM schemes provide a means to address multiplicative noise-degraded images by constructing decompositions across several scales. It is hoped that the schemes and the included analysis might be helpful for the reader to identify applications beyond the denoising and deblurring tasks investigated here, and to extend to additional schemes outside those studied within. In the future, we aim to extend the proposed models to image segmentation and to vector-valued data, for instance, to restore color images perturbed by multiplicative noise and blurring.

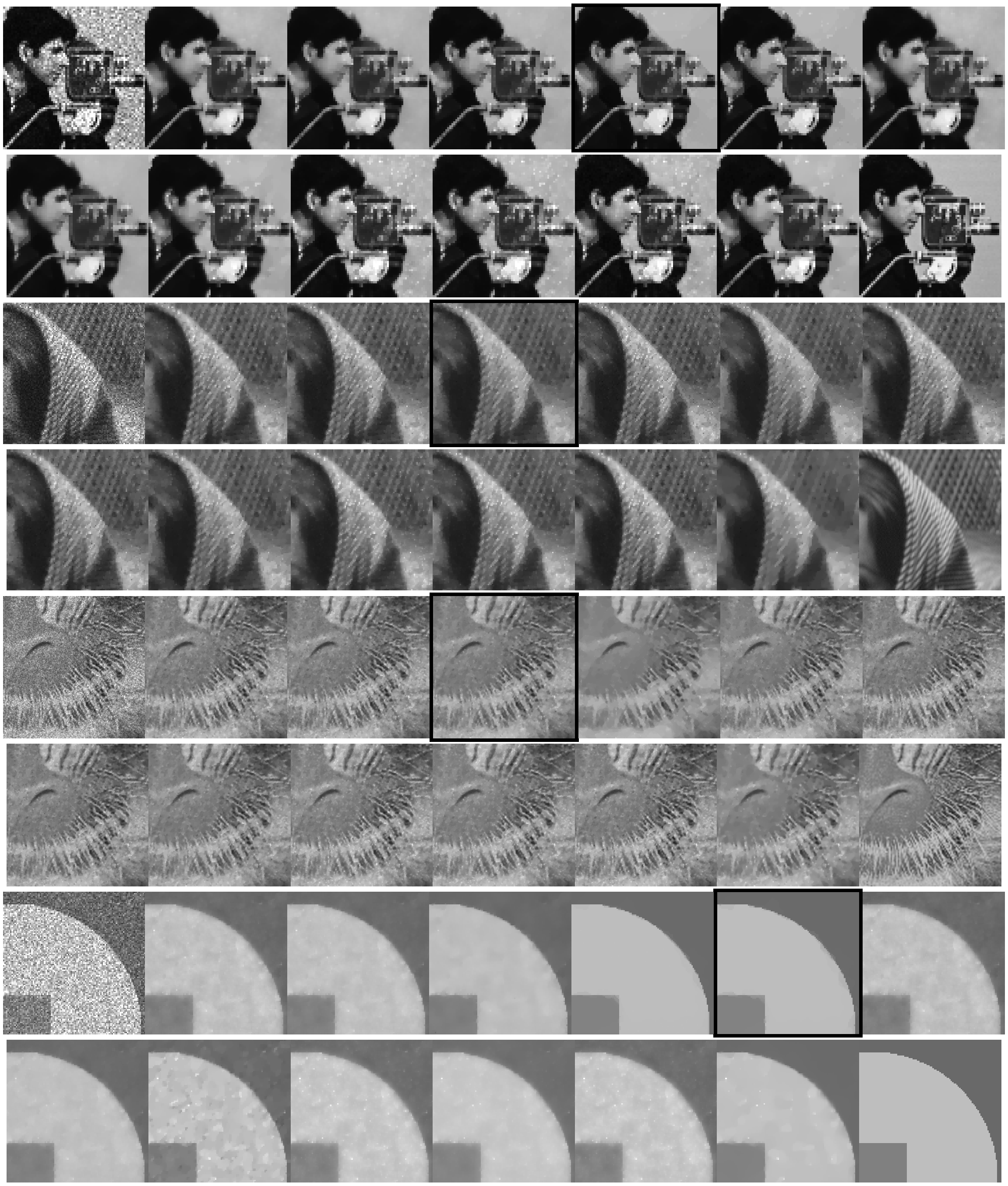


Figure 2.19: Detailed MHDM denoising. From left to right, top to bottom: noisy images; SO MHDM (EL) regular, tight, and refined versions; SO MHDM (ADMM) regular and tight versions; AA-log MHDM regular, tight, and refined versions; TNV-log regular and tight versions; TNV, DZ and original images. Black borders indicate the best recoveries (highest SNR) for each image.

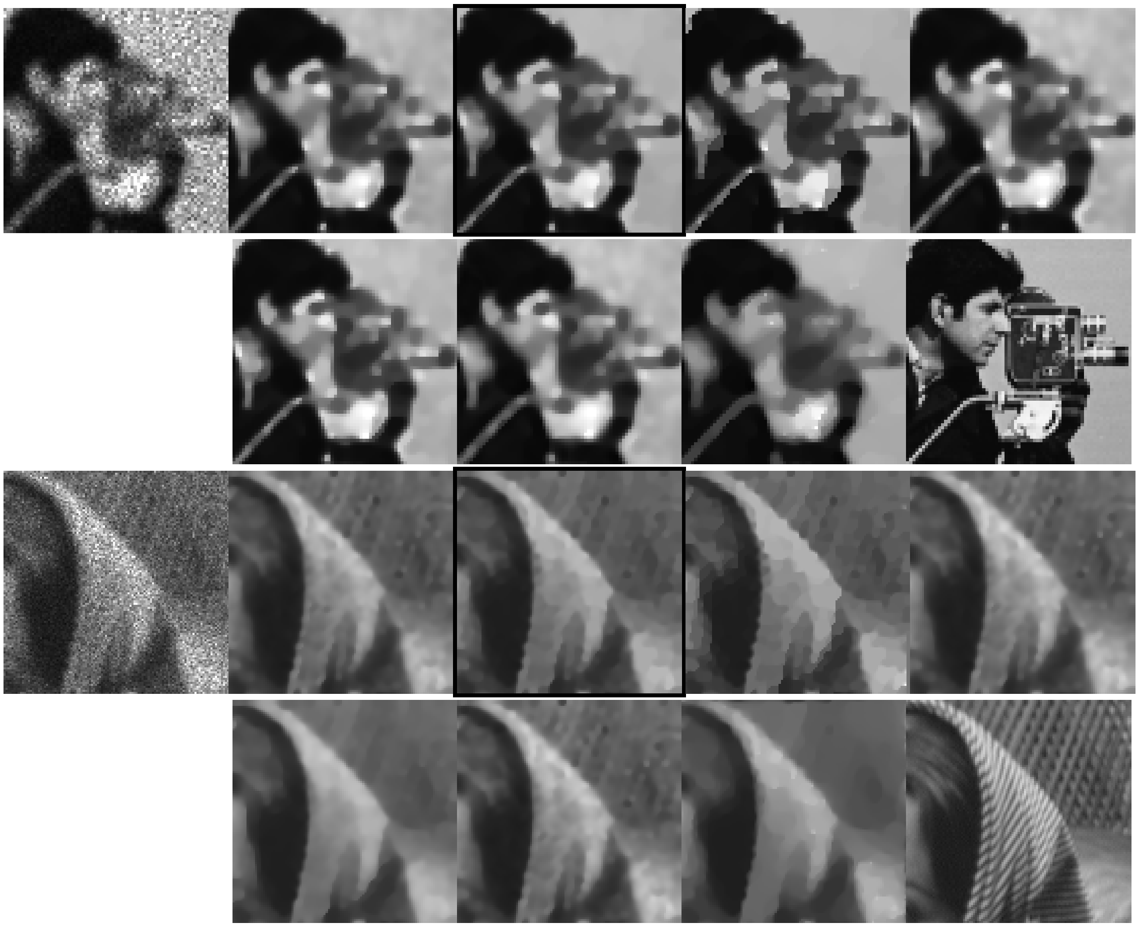


Figure 2.20: Detailed MHDM denoising-deblurring results for Cameraman and Barbara. From left to right, top to bottom: noisy images; AA-log MHDM regular, tight, and refined versions; TNV-log regular and tight versions; TNV, DZ, and original images. Black borders indicate the best recoveries (highest SNR) for each image.

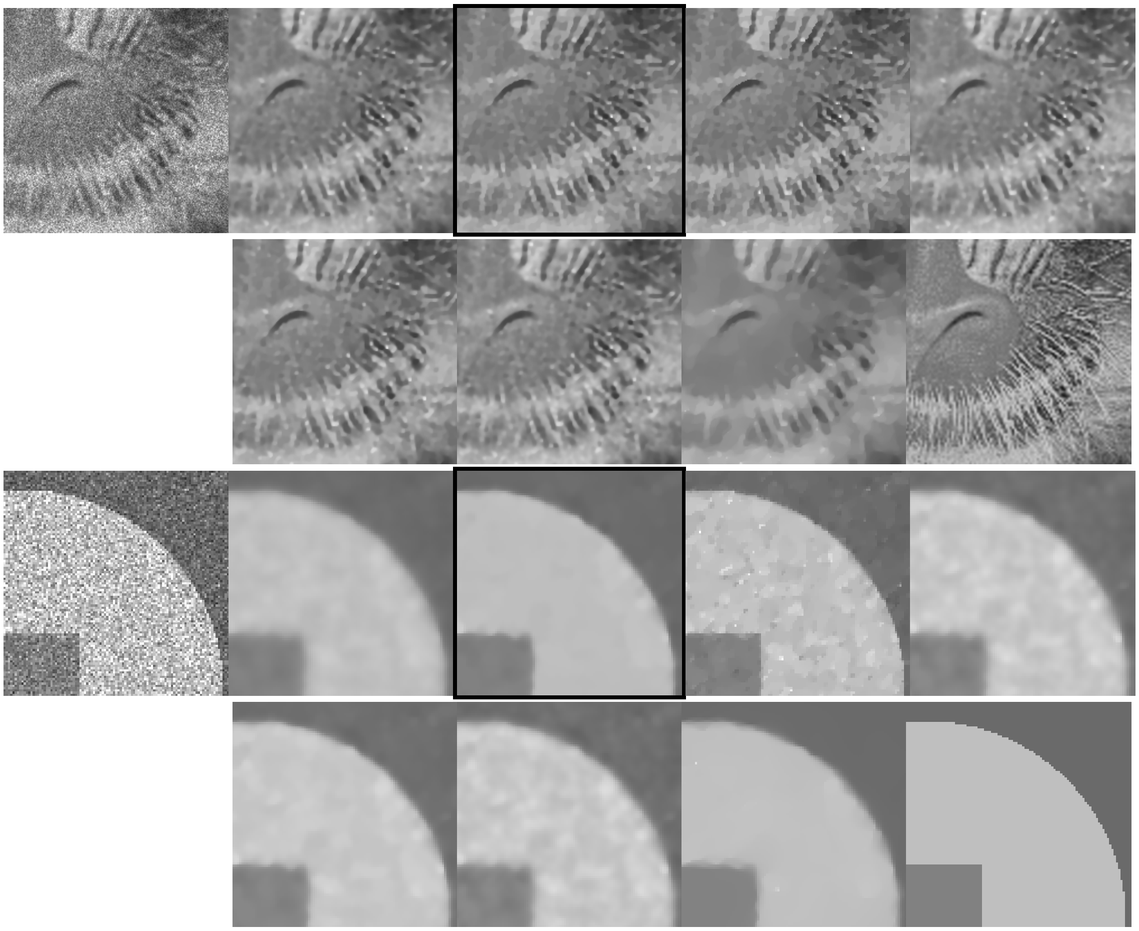


Figure 2.21: MHDM denoising-deblurring recoveries for Mandril and Geometry images. From left to right, top to bottom: noisy images; AA-log MHDM regular, tight, and refined versions; TNV-log regular and tight versions; TNV, DZ, and original images. Black borders indicate the best recoveries (highest SNR) for each image.

CHAPTER 3

Optical Flow for Atmospheric Motion Estimation[1]

3.1 Introduction

Optical flow has a long and successful history in computer vision applications as a motion estimator. The method has found a place in object tracking including such applications as optical computer-mice, stereo-vision, and scene motion estimation. In general, an optical flow method attempts to extract a displacement field which describes the motion between two scenes (often two consecutive frames from an image sequence or video). While the method is traditionally used on natural imagery, it has been applied as a tool in studying fluids by examining particle-laden flows, a practice known as particle image velocimetry (PIV) [Adr05, LYZ21], as well as for cloud and weather tracking [CBK15, WW17]. Applications like PIV and cloud tracking prove challenging because of the complicated nature of fluid motion—in particular, there are no longer stable features or quasi-rigid motions typical of natural imagery.

We are interested in applying optical flow towards water vapor data with the aim of recovering atmospheric wind velocities. The ability to extract such estimates from satellite data is of significant interest since microwave and infrared instrument equipped satellites provide comprehensive coverage of weather phenomenon without the need for in-place instruments (e.g., ocean drifters, physical wind sensors), which are both costly and provide

[1] This chapter is adapted from [BBV23b] with permission from SPIE, and [BBV23a] ©2023 IEEE.

only sparse estimates.

Our data comes from direct numerical simulations of atmospheric conditions from the Weather Research and Forecasting III model [SKD08]. Importantly, because we use data from simulations, it has accompanying ground truth wind velocities allowing us to evaluate any flow estimate against the correct velocity field.

Water vapor from three weather scenarios is extracted from this nature-run data and used in our experiments. First, we have a mesoscale convective system (MCS) over the eastern Pacific which provides higher spatial resolution but with larger time intervals between frames. Second, we consider tropical convection (TC) over the maritime continent. Last, we consider an extratropical cyclone (ETC) event over the western Atlantic. The size, spatial and temporal resolution of each dataset is given in Table 3.1. We note that, as is typical of remote sensing data, the spatial resolution is quite coarse (on the order of 3km to 12km per pixel). Furthermore, for the MCS dataset, the frames of data are separated by 15 minute intervals. This means that for moderate wind speeds (15-20 m/s, or 33-45 mph), water vapor transported by the wind could travel across many pixels. Large displacement flows (in terms of pixels) is a known challenge for optical flow methods. Moreover, for the lower spatial resolution datasets with shorter time-separation intervals, wind-driven displacement frequently is on the sub-pixel scale which also poses a challenge for resolving low velocity

Table 3.1: Description of water vapor datasets.

Name	Image Size	Spatial Resolution	Time Resolution
MCS	850×1850 px^2	3000 m/px	900 s
TC	999×1299 px^2	3500 m/px	72 s
ETC	480×480 px^2	12000 m/px	120 s

regions.

3.2 Background

The primary assumption of optical flow methods is one of *brightness constancy.* That is, the brightness of an object in a scene does not change over short time intervals, and an optical flow algorithm attempts to track the rearrangement or mixing of these intensity-unvarying objects (or pixels in the case of image-like data) with time. To frame this constancy assumption more precisely, let $I(\cdot, t) : \Omega \subset \mathbb{R}^2 \to \mathbb{R}$, with $I(\boldsymbol{x}, t)$ representing the image intensity at some point $\boldsymbol{x} \in \Omega$ for each fixed time t. Here, $\boldsymbol{x} = (x, y)$. If we assume the object at $\boldsymbol{x}$ is displaced by $\boldsymbol{d} = \boldsymbol{d}(\boldsymbol{x}, t)$ as time advances to $t + \Delta t$, then brightness constancy assumes $I(\boldsymbol{x} + \boldsymbol{d}, t + \Delta t) = I(\boldsymbol{x}, t)$. Taylor expanding I about $(\boldsymbol{x}, t)$ gives the so-called optical flow constraint equation (OFCE)

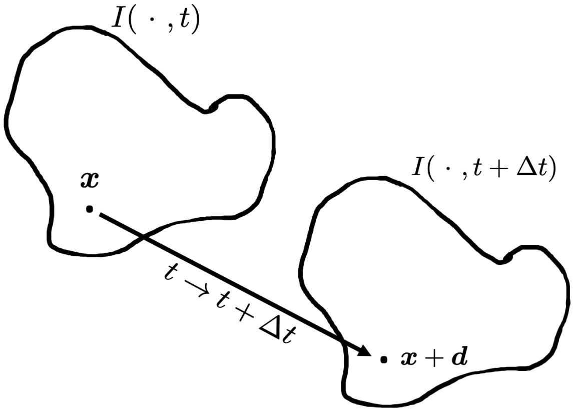

$I(\,\cdot\,, t)$
$\boldsymbol{x}$
$I(\,\cdot\,, t + \Delta t)$
$t \to t + \Delta t$
$\boldsymbol{x} + \boldsymbol{d}$

Figure 3.1: Brightness constancy assumption: if I takes value $I(\boldsymbol{x}, t)$ at time t, and the object at $\boldsymbol{x}$ moves to $\boldsymbol{x} + \boldsymbol{d}$ as $t \to t + \Delta t$, then the brightness/intensity at the displaced points match: $I(\boldsymbol{x} + \boldsymbol{d}, t + \Delta t) = I(\boldsymbol{x}, t)$.

$$I_t(\boldsymbol{x}, t) + \boldsymbol{u}^T \nabla I(\boldsymbol{x}, t) = 0, \tag{3.1}$$

where $\boldsymbol{u} = \boldsymbol{d}/\Delta t$ is the velocity field describing the displacement. This is depicted in Figure 3.1. By letting $\Delta t \to 0$, the left-hand side of (3.1) can also be seen as the total time-derivative of $I(\boldsymbol{x}, t)$, where it is understood that $(\boldsymbol{x}, t) = (\boldsymbol{x}(t), t)$ traces out the space-time path objects take as they move within the scene. Since the transport equation (3.1) is obtained through linearization, it is sometimes called the *linear OFCE.* The non-linear version is simply

$$I(\boldsymbol{x} + \boldsymbol{d}, t + \Delta t) - I(\boldsymbol{x}, t) = 0. \tag{3.2}$$

In practice, when working with video or imagery data it is standard to take $\Delta t = 1$ (in which case $\boldsymbol{u} = \boldsymbol{d}$) and define frame one by $I_1(\boldsymbol{x}) = I(\boldsymbol{x}, t + 1)$ and frame zero by $I_0(\boldsymbol{x}) = I(\boldsymbol{x}, t)$. An optical flow method aims to find the flow field $\boldsymbol{u} = (u_1, u_2)^T$ given the images I_0 and I_1.

As it stands, solving (3.1) for the unknown $\boldsymbol{d}$ is ill-posed, since there is only one equation but multiple unknowns. This feature of optical flow constraint equation is known as the aperture problem, and can be understood visually in Figure 3.2. When restricted to viewing a scene through an"aperture", there are multiple feasible directions a patch within the scene could be moving.

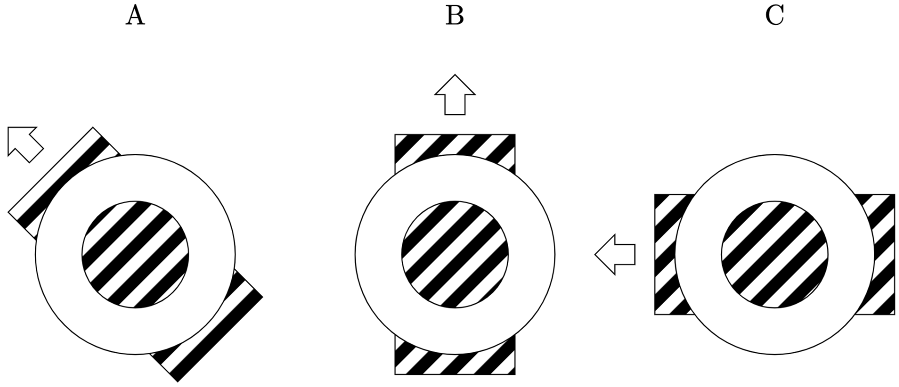


Figure 3.2: The aperture problem. In each of diagrams A, B, or C, if the patterned background is pulled through the viewing aperture in the indicated direction, the resulting motion appears identical.

Because (3.1) is ill-posed, additional constraints are required to improve the conditioning of the problem and determine the flow. This can be remedied by assuming the flow is constant on neighborhoods as was done by Lucas and Kanade [LK81], or imposing regularization on the flow as was introduced by Horn and Schunck [HS81] (HS) in 1981. Methods which impose constraints on local regions, e.g. Lucas and Kanade's flow, block matching or correlation techniques [LMM15, PWM19], are known as local methods which generally produce a sparse flow field describing the motion within a scene. Conversely, schemes which place requirements on the whole flow are known as global methods, and produce dense velocity fields. Additionally, hybrid methods [HHM08, BWS05, HMS10, LSF20] which blend correlation and variational techniques have been tested. Because we are interested in recovering dense flow estimates for wind velocities, we focus on global variational optical flow methods, which have outperformed cross correlation techniques for atmospheric tracking (see [CBK15]).

Subsequent optical flow techniques have primarily been based on the seminal work of Horn and Schunck, which solves the minimization problem

$$\min_{\boldsymbol{u}=(u_1,u_2)} \lambda \int_{\Omega} \left(I_t + \boldsymbol{u}^T \nabla I\right)^2 d\boldsymbol{x} + \int_{\Omega} \left(|\nabla u_1|^2 + |\nabla u_2|^2\right) d\boldsymbol{x}. \tag{3.3}$$

These have been applied to PIV [Adr05, LYZ21], tailored for fluid flow through relaxed vorticity and divergence penalties [CZS15], tested for weather forecasting [WW17] and utilized in atmospheric tracking [BD06, CBK15, YSP22, YPW23, BBV23a, OZW21].

Each of these methods can broadly be categorized as the minimization problem

$$\min_{\boldsymbol{u}} \lambda \int_{\Omega} \psi(\rho(\boldsymbol{u}; I)) d\boldsymbol{x} + J(\boldsymbol{u}) \tag{3.4}$$

where ρ represents the data fitting term, ψ a non-negative penalizing function and $J(\cdot)$ is a regularization term. For instance, the HS flow is characterized by $\psi(x) = x^2$, $\rho(\boldsymbol{u}) = I_t + \boldsymbol{u}^T \nabla I$ and $J(\boldsymbol{u}) = \int_{\Omega} \left(|\nabla u_1|^2 + |\nabla u_2|^2\right)$. The positive constant λ weights the emphasis of the data over the regularization.

3.2.1 Large Displacement Optical Flow

For scenes with large displacements between frames, the linearized OFCE is often inaccurate, so either non-linear schemes are devised (see [SPC09] for e.g.), or a method of "warping" is utilized, whereby $I(\boldsymbol{x} + \boldsymbol{u_0})$ is computed via interpolation (typically bicubic) for some approximation $\boldsymbol{u}_0$ to the desired flow. The OFCE is then reformed by linearizing (3.2) around this most recent approximation, $\boldsymbol{x} + \boldsymbol{u}_0$. This "warping" process is repeated until a final flow is obtained.

Additionally, to further assist with large displacements, optical flow algorithms commonly make use of a hierarchical pyramid multi-resolution scheme (see [ABH93]). First, images are spatially smoothed through a low-pass filter before being spatially downsampled and stored at each level, forming the so-called multiresolution hierarchy pyramid. The purpose in reducing the resolution is to resolve the motion within the scene to one or two pixels. If an object moves 16 pixels between frames at full resolution, then downsampling spatially by a factor of 8 or 16 will result in the same object moving only 1-2 pixels within the new coarse grain data.

Images are then used to form flow estimates, starting with the coarsest resolution first, and working to more resolved scales. The flow estimates from the coarse scales serve to initialize flow estimates for higher resolution images. This process is repeated iteratively until the original resolution is reached (see [MS13] for a HS implementation).

3.2.2 Limitations of the Optical Flow Constraint Equation for Fluid Motion

Applying HS directly to fluid data yields less than satisfactory results, as shown in Figure 3.3. We note that throughout this chapter, we will visualize a vector field using a *color plot* which assigns a vector's direction a color and the saturation determines the vector magnitude. This can be decoded by the circular color-chart in Figure 3.3. The underperformance of HS is not surprising, as the method is designed more for quasi-rigid motions, with the regularizing term $\|\nabla \boldsymbol{u}\|^2$ penalizing turbulent flows harshly [CHA06, CZS15]. Specifically, when viewed as a minimization penalty, $\int_\Omega |\nabla u_1|^2 + |\nabla u_2|^2 d\boldsymbol{x}$ is equivalent with a quadratic penalty on the divergence and vorticity of $\boldsymbol{u}$. To see this, we must look at the Euler-Lagrange equations associated with minimizing the two penalties. Note that for any u

$$\begin{aligned} \frac{d}{ds}\left(\int_\Omega |\nabla(u+s\phi)|^2 d\boldsymbol{x}\right)\Bigg|_{s=0} &= 2\int_\Omega \nabla u \cdot \nabla \phi d\boldsymbol{x} \\ &= -2\int_\Omega \phi \Delta u d\boldsymbol{x} \end{aligned}$$

for all ϕ vanishing on $\partial\Omega$. Thus, the necessary condition for minimizing $\|\nabla \boldsymbol{u}\|^2$ is

$$\begin{cases} -2\Delta u_1 = 0, \\ -2\Delta u_2 = 0, \end{cases} \tag{3.5}$$

on Ω.

Now consider $\text{div}(\boldsymbol{u})^2 + \text{vort}(\boldsymbol{u})^2 = |\nabla u_1|^2 + |\nabla u_2|^2 + 2\partial_x u_1 \partial_y u_2 - 2\partial_y u_1 \partial_x u_2$. We rewrite the mixed derivative terms as $F(u_1, u_2) = 2\nabla u_1 \cdot (\nabla u_2)^\perp$, where $(\nabla u_2)^\perp$ is the vector, $(\partial_y u_2, -\partial_x u_2)^T$, normal to ∇u_2, and the vorticity $\text{vort}(\boldsymbol{u}) = \partial_x u_2 - \partial_y u_1$ as usual.

The optimality condition on $\boldsymbol{u}$ for $F(u_1, u_2)$ vanishes, as can be seen by taking the first variation

$$\begin{aligned} \frac{d}{ds}\int_\Omega F(u_1 + s\phi, u_2 + s\psi) d\boldsymbol{x}\big|_{s=0} &= 2\int_\Omega \nabla\phi \cdot (\nabla u_2)^\perp + \nabla u_1 \cdot (\nabla \psi)^\perp d\boldsymbol{x} \\ &= 2\int_\Omega \nabla\phi \cdot (\nabla u_2)^\perp - (\nabla u_1)^\perp \cdot \nabla\psi d\boldsymbol{x} \\ &= -2\int_\Omega \phi \nabla \cdot (\nabla u_2)^\perp - \nabla \cdot (\nabla u_1)^\perp \psi d\boldsymbol{x} = 0, \end{aligned}$$

since the normal vectors $(\nabla u_1)^\perp$ and $(\nabla u_1)^\perp$ are divergence free. The second equality makes use of the perpendicular-operation $\perp$ which is anti-symmetric, so $\nabla u_1 \cdot (\nabla \psi)^\perp = -(\nabla u_1)^\perp \cdot$

$\nabla\psi$. Consequently, a quadratic divergence-vorticity penalty

$$\int_\Omega \left(\operatorname{div}(\boldsymbol{u})^2 + \operatorname{vort}(\boldsymbol{u})^2\right) d\boldsymbol{x} = \int_\Omega \left(|\nabla u_1|^2 + |\nabla u_2|^2 + F(u_1, u_2)\right) d\boldsymbol{x}$$

will have the same Euler-Langrange equation (3.5) as a quadratic gradient penalty. Given the restrictions of the HS penalty, a natural starting place to try and improve flow estimates is to modify $\psi(\rho)$ and J to be more suited for the data. A more robust data term making use of the L^1 norm, which the authors in [SMF13, WPZ09, ZPB07] consider, in addition to total variation regularization, better adapt to illumination changes and promote discontinuities common in complicated flows. Such a scheme would have $\psi(\rho) = |\rho|$, $\rho(\boldsymbol{u}; I) = I_t + \boldsymbol{u}^T \nabla I$ and regularization $J(\boldsymbol{u}) = \int_\Omega (|\nabla u_1| + |\nabla u_2|) d\boldsymbol{x}$ in (3.4).

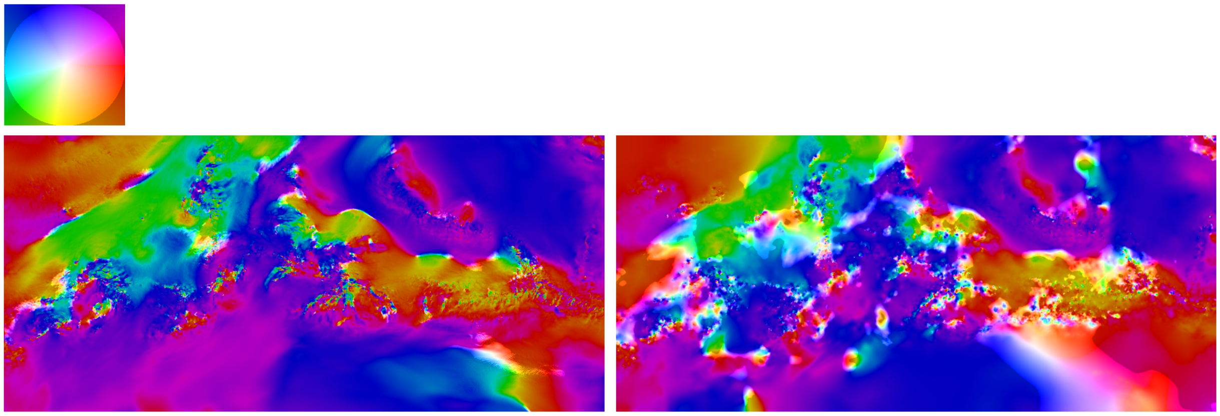


Figure 3.3: HS optical flow derived wind velocity estimate. Ground truth (left) and HS (right) flows. HS, with a root-mean-square-vector-distance of 3.400m/s, exhibits significant fragmentation. Note: the figure depicts a vector field using color to indicate direction, and saturation to indicate speed. Direction and magnitude can be inferred from the circular legend at the top-left.

Efficient implementations of this so named TV-$L1$ optical flow method are discussed in [SMF13, WPZ09, ZPB07], along with a non-linear scheme which requires no warping in [SPC09]. A Python language implementation is available from the Scikit-image [WSN14] library. The method has also been proposed for atmospheric motion vector estimation, and the authors in [YSP22, YPW23] show that TV-$L1$ performs favorably compared with conventional atmospheric motion tracking methods (see Figure 3.4). TV-$L1$ improves flow estimates in our data compared with HS and is able to extract the general behavior of the flow, but struggles to recover flow in areas of low image variation and still over-smooths the divergence and vorticity present in the flow.

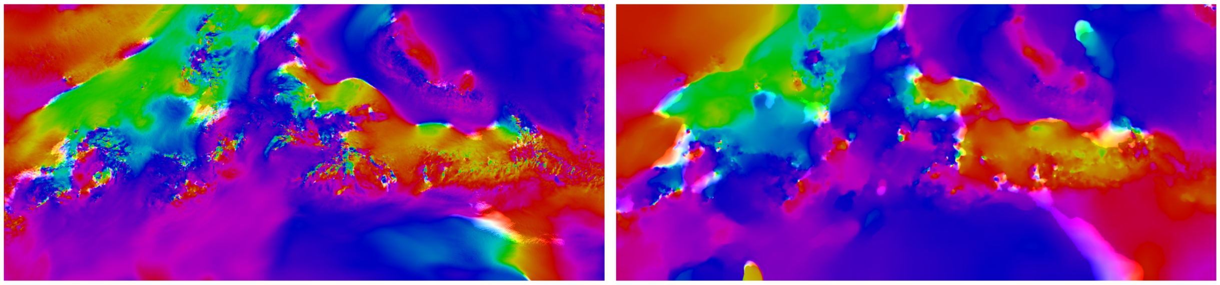


Figure 3.4: TV-$L1$ optical flow derived wind velocity estimate. Ground truth (left) and TV-$L1$ (right) flows. TV-$L1$ captures the structure of the flow, and reduces average error to 2.483 m/s.

While additionally modifying the penalty ψ and regularization J may yield some improvements, we consider more generally whether the OFCE alone is appropriate for recovering the desired flow. Because we have the ground truth data $\boldsymbol{u}_{GT}$ available, we are able to test the performance of the ground truth and any computed flow $\boldsymbol{u}$ at minimizing (3.4) as well as satisfying the data term $\rho(\,\cdot\,; I)$. We make the critical observation that the TV-$L1$ computed flows outperform the ground truth, both at minimizing (3.4) and satisfying the OFCE (both linear and non-linear adaptations), yet do not match the desired flow satisfactorily. That is,

$$\int_\Omega |I_t + \boldsymbol{u}^T \nabla I| d\boldsymbol{x} + J(\boldsymbol{u}) < \int_\Omega |I_t + \boldsymbol{u}_{GT}^T \nabla I| d\boldsymbol{x} + J(\boldsymbol{u}_{GT}),$$

for $\boldsymbol{u}$ a minimizer of the TV-$L1$ problem.

This indicates that for our data, the OFCE alone is insufficient for guiding flow discovery. Driving $\boldsymbol{u}$ to satisfy the OFCE more exactly can move $\boldsymbol{u}$ further from $\boldsymbol{u}_{GT}$, and additions or modifications to the constraint equation are needed if we hope to recover improved estimates.

3.2.3 Texture features for flow estimation

In an attempt to provide more context to the flow estimator in regions of uniform image-intensity, we propose decomposing the image into texture and structure components, thereafter extracting a velocity field from a combination of these components. We note that using the texture portion of images for optical flow has been proposed as a way to combat the effects of brightness variation on flow estimation (i.e. illumination changes which break the brightness constancy assumption) [WPZ09], however in the context of water-vapor imagery, the data is not illumination based and so our motivation for such a decomposition is distinct.

A standard image decomposition can be formed by the iconic Rudin-Osher-Fatemi total variation denoising model, where an image I is split into the sum $I = I^S + I^{Text}$, with the structure portion I^S minimizing

$$\min_{f \in BV(\Omega)} \lambda \int_\Omega (I - f)^2 d\boldsymbol{x} + TV(f), \tag{3.6}$$

where $BV(\Omega)$ is the space of functions of bounded variation over Ω. However, a decomposition with more oscillatory texture is desirable because this provides more features and variation to add context for flow extraction, especially in regions where the image is primarily very smooth. One such texture-structure image decomposition was proposed by Meyer [Mey01], in which $I^{Text} = I - I^S$ belongs to the weaker space $G(\Omega) = \{\text{div}(\boldsymbol{g}) : \boldsymbol{g} \in L^\infty(\Omega)^2\}$. Oscillations such as texture have small norms in $G(\Omega)$, and thus are not penalized by such a model. The decomposition is determined by solving

$$I^S = \underset{f \in BV(\Omega), \boldsymbol{g} \in L^\infty(\Omega)^2}{\arg\min} \lambda \int_\Omega (I - (f + \text{div}(\boldsymbol{g}))^2 d\boldsymbol{x} + TV(f) + \mu \|\boldsymbol{g}\|_\infty. \tag{3.7}$$

Details on numerically determining (3.7) are found in [OSV03], where a p-norm relaxation of (3.7) is utilized. We use this approach to find $I^{Text} = I - I^S$ for our data.

3.2.4 Texture-flow

Armed with a structure-texture decomposition, how might one use these features to improve flow extraction? We propose a texture-flow process which tracks the motion of the texture portion of an image (or data), and uses this flow to improve the overall atmospheric motion vector estimate.

The first task is incorporating texture or structure from such a decomposition into the optical flow process. We test both the performance of I^{Text} and I^S at producing flow estimates when used as the data within the TV-$L1$ algorithm, individually. Heuristically, we find that I^{Text} alone improves flow estimates while I^S does not (over the original flow from I itself). A possible explanation of this is that the full image, I, contains primarily the information of I^S, with only small contribution of texture from I^{Text} which exist at a much smaller scales and magnitudes than both I and I^S. Thus, the motion of large scale features and structures within the water vapor data are primarily captured by I (or I^S), and the penalty within $\rho(\boldsymbol{u}; I)$ from miss-aligning the motion seen by I^{Text} is minimal. Whereas, when considering $\rho(\boldsymbol{u}; I^{Text})$ separately, the penalty is entirely due to the motion within I^{Text}, and so the ensuing flow conforms to this more strongly.

We would like to capitalize on the texture-features within I^{Text}. Starting with I^{Text} computed for both frames I_0 and I_1, we compute the TV-$L1$ texture-flow $\boldsymbol{u}^{Text}$ determined by these textural components (i.e. solve (3.4) with $\psi(x) = |x|$, $\rho(\boldsymbol{u}; I^{Text}) = I_1^{Text}(\boldsymbol{x} + \boldsymbol{u}) - I_0^{Text}(\boldsymbol{x})$ and $J(\boldsymbol{u}) = TV(u_1) + TV(u_2)$). This “texture-based” flow represents only the motion of the I^{Text} features. To further improve our flow estimate, we make use of the warping process when solving the non-linear OFCE problem to form $I_1(\boldsymbol{x} + \boldsymbol{u}^{Text})$. This warped, full image now incorporates the motion determined from I^{Text}, and we can compare the remaining mismatch (according to $\psi(\rho(\cdot))$) between I_0 and $I_1(\boldsymbol{x} + \boldsymbol{u}^{Text})$ to form a correction to $\boldsymbol{u}^{Text}$. We call this finding this correction *flow refinement*, and denote the derived flow by $\boldsymbol{u}^R$. Finally, we compile the total flow $\boldsymbol{u} = \boldsymbol{u}^{Text} + \boldsymbol{u}^R$. The hope is that in concert, the texture within I_0^{Text} and I_1^{Text} affords enough information to determine $\boldsymbol{u}^{Text}$ more accurately in regions of uniform image intensity, while $\boldsymbol{u}^R$ corrects for the motions of the larger structures present within I_0 and I_1. This texture-flow process is encoded in Algorithm 9.

Algorithm 9 Texture-Flow

```
INPUT: Images $I_1$ and $I_0$ and texture components $I_1^{Text}$, $I_0^{Text}$.
OUTPUT: $\boldsymbol{u}$, the flow between $I_1$ and $I_0$.
$\boldsymbol{u}^{Text} \leftarrow \arg\min_{\boldsymbol{u}} \lambda \int_\Omega \psi(\rho(\boldsymbol{u}; I^{Text}))d\boldsymbol{x} + J(\boldsymbol{u})$
$I_1 \leftarrow I_1(\boldsymbol{x} + \boldsymbol{u}^{Text})$ (via bicubic uplook)
$\boldsymbol{u}^R \leftarrow \arg\min_{\boldsymbol{u}} \lambda \int_\Omega \psi(\rho(\boldsymbol{u}; I))d\boldsymbol{x} + J(\boldsymbol{u})$
return : $\boldsymbol{u} = \boldsymbol{u}^{Text} + \boldsymbol{u}^R$
```

One could also imagine performing a texture-flow like process with different features which we expect to satisfy a constancy-like assumption. Different fidelity flow constraint functions $\rho(\boldsymbol{u}; I)$ could also be considered for each feature. However, including many features and fidelity terms together quickly gives rise to many questions and complications, such as which order to process each feature. In order to consider incorporating many features, a more unifying approach is needed.

3.2.5 Multi-fidelity flow

Multi-fidelity flow is a step towards this unification. One can consider reconstructing the full flow in a single process by concurrently including the original and textural portions of the data I_0 and I_1 within the optical flow minimization problem. An approach to this is to create a data fidelity term which incorporates multiple features—in this case, the raw data

I_0, I_1 and it's texture I_0^{Text}, I_1^{Text}—from the data simultaneously, as given in the following minimization problem

$$\arg\min_{\boldsymbol{u}} \lambda_1 \int_\Omega |\rho(\boldsymbol{u}; I)| + \lambda_2 \int_\Omega |\rho(\boldsymbol{u}; I^{Text})| d\boldsymbol{x} + TV(u_1) + TV(u_2), \tag{3.8}$$

where the λ_i are weights for each fidelity term. We appropriately refer to (3.8) as a *multi-fidelity* flow problem, since the fidelity term contains a sum of multiple elements.

In [BBP04], the authors discuss that additional image features such as image gradient and higher order features like Hessian and Laplacian, should satisfy the constancy assumption. Shifts in gray-level intensity in an image will break brightness constancy, but gradients of the image will remain constant under such global illumination changes. Additionally, scaling the intensity levels within I will cause scaled change within the image gradient lengths, but will leave the gradient directions unchanged and ∇I can be seen to satisfy it's own constancy assumption. Similarly, higher order features will be invariant under affine (or higher) transformations of I, and a constancy assumption on these is natural.

Additionally, a gradient can be decomposed into magnitude and direction. Under rotational transformations, the magnitude will remain fixed with direction alone changing. It follows that one may also consider the gradient-magnitude as a feature which tracks a constancy assumption.

To tackle considering many features simultaneously, we propose a reframing of the optical flow problem into the *multi-fidelity* structure

$$\arg\min_{\boldsymbol{u}} \sum_{j=1}^{N} \lambda_j \int_\Omega \psi(\rho_j(\boldsymbol{u}; I)) d\boldsymbol{x} + J(\boldsymbol{u}), \tag{3.9}$$

where $J(\cdot)$ remains the regularization term and now there are $j = 1, 2, \ldots, N$ data fidelity terms $\psi(\rho_j(\boldsymbol{u}; I))$ with $\psi(\cdot)$ some penalty and $\rho_j(\cdot)$ a data-fitting term such as the OFCE. The HS method would have $N = 1$, $\rho_1(\boldsymbol{u}; I) = I_t + \boldsymbol{u}^T \nabla I$, $\psi(x) = x^2$ and $J(\boldsymbol{u}) = \int_\Omega (|\nabla u_1|^2 + |\nabla u_2|^2) d\boldsymbol{x}$, while TV-$L1$ has $\psi(x) = |x|$ and $J(\boldsymbol{u}) = \int_\Omega (|\nabla u_1| + |\nabla u_2|) d\boldsymbol{x}$.

3.3 Numerical Minimization

Our objective is solving the minimization problem (3.4) as it appears within Algorithm 9 and (3.9). For quadratic (or smooth) ψ and well-conditioned regularizing functionals $J(\,\cdot\,)$, straightforward techniques of gradient descent can be followed after forming the associated Euler-Lagrange equations for each minimization problem. In general, though, these can be difficult optimization problems, especially if ρ is non-linear in $\boldsymbol{u}$ and when $\psi(\,\cdot\,)$ is non-smooth. However, we are interested in the robust penalty $\psi(\,\cdot\,) = |\,\cdot\,|$ and given the potential for large displacements for our data, we must consider non-linear OFCE approaches and make use of the iterative warping and multi-resolution techniques mentioned in Section 3.2.1

3.3.1 Texture-Flow Numerics

To solve for the flow in Algorithm 9, we must minimize (3.4) for a specific choice of ψ, ρ and J. In TV-$L1$ flow, the corresponding problem is

$$\min_{\boldsymbol{u}} \lambda \int_{\Omega} |\rho(\boldsymbol{u}; I)| d\boldsymbol{x} + TV(u_1) + TV(u_2),$$

where $\rho(\boldsymbol{u}; I) = I_1(\boldsymbol{x} + \boldsymbol{u}) - I_0(\boldsymbol{x})$ is highly nonlinear in $\boldsymbol{u}$. To approach this minimization, we linearize ρ about a current flow estimate $\boldsymbol{u}_0$,

$$\rho(\boldsymbol{u}; I) \approx \rho(\boldsymbol{u}; \boldsymbol{u}_0, I) := I_1(\boldsymbol{x} + \boldsymbol{u}_0) - I_0(\boldsymbol{x}) + (\boldsymbol{u} - \boldsymbol{u}_0)^T \nabla I(\boldsymbol{x} + \boldsymbol{u}_0),$$

where, with some abuse of notation we refer to the linearized version by $\rho(\boldsymbol{u}; \boldsymbol{u}_0, I)$. At times when the context is clear, we will drop the explicit dependence of ρ on I. Now, linearization on it's own can be inaccurate for large displacement flows, so the current flow estimate $\boldsymbol{u}_0$ must be near the desired flow. Consequently, we will repeatedly update $\boldsymbol{u}_0$ and warp the image I_1 by this updated displacement estimate throughout the recovery process in order to iteratively step $\boldsymbol{u}_0$ nearer the desire flow. This process of warping is described in Algorithm 10 for the multi-fidelity flow process, but is also carried out in the texture-flow recoveries.

Given the current $\boldsymbol{u}_0$, we approach minimizing

$$\min_{\boldsymbol{u}} \lambda \int_\Omega |I_1(\boldsymbol{x}+\boldsymbol{u}_0) - I_0(\boldsymbol{x}) + (\boldsymbol{u}-\boldsymbol{u}_0)^T \nabla I(\boldsymbol{x}+\boldsymbol{u}_0)| d\boldsymbol{x} + TV(u_1) + TV(u_2) \tag{3.10}$$

by splitting the fidelity and regularizing terms, alternatingly solving the two proximal problems (3.11) and (3.12)

$$\boldsymbol{u}^{k+1} \in \arg\min_{\boldsymbol{u}} \lambda \int_\Omega |\rho(\boldsymbol{u};\boldsymbol{u}_0)| d\boldsymbol{x} + \frac{1}{2\theta}\|\boldsymbol{u}-\boldsymbol{v}^k\|^2 \tag{3.11}$$

$$\boldsymbol{v}^{k+1} \in \arg\min_{\boldsymbol{v}} \frac{1}{2\theta}\|\boldsymbol{u}^{k+1}-\boldsymbol{v}\|^2 + J(\boldsymbol{v}), \tag{3.12}$$

with $\boldsymbol{u}^0$ (note the superscript to indicate iterate, whereas the subscript denotes current flow estimate) initialized to some guess for the flow (typically $\boldsymbol{u}_0$). The tightness parameter θ forces $\boldsymbol{v}^k$ near $\boldsymbol{u}^k$, and letting $\theta \to 0$ the split minimization problems given in (3.11) and (3.12) is equivalent with (3.10). The reason for splitting is that the individual minimization problems are easier when considered separately. Indeed, (3.12) decouples along the components of $\boldsymbol{v}$, giving two ROF total variation denoising problems. These have a well documented (see [Cha04]) fast dual-projection method,

$$\boldsymbol{p}_i^{n+1} = \frac{\boldsymbol{p}_i^n + \tau\nabla(\mathrm{div}\boldsymbol{p}_i^n - u_i^k/\theta)}{1+\tau|\nabla(\mathrm{div}\boldsymbol{p}_i^n - u_i^k)|} \tag{3.13}$$

$$v_i^{k+1} = u_i^k - \theta\mathrm{div}\boldsymbol{p}_i^N, \tag{3.14}$$

which can be used to solve for each component, v_i^{k+1}, of $\boldsymbol{v}^{k+1}$ after running n in (3.13) to some convergence criterion $\tilde{N}$. For (3.11), we follow a shrinkage-like thresholding procedure

(3.15) detailed in [ZPB07, WPZ09, SMF13]

$$\boldsymbol{u}^{k+1} = \boldsymbol{v}^k + \begin{cases} \lambda\theta\nabla I_1(\boldsymbol{x}+\boldsymbol{u}_0) & \text{if } \rho(\boldsymbol{v}^k;\boldsymbol{u}_0) < -\lambda\theta|\nabla I_1(\boldsymbol{x}+\boldsymbol{u}_0)|^2 \\ -\lambda\theta\nabla I_1(\boldsymbol{x}+\boldsymbol{u}_0) & \text{if } \rho(\boldsymbol{v}^k;\boldsymbol{u}_0) > \lambda\theta|\nabla I_1(\boldsymbol{x}+\boldsymbol{u}_0)|^2 \\ -\rho(\boldsymbol{v}^k;\boldsymbol{u}_0)\nabla I_1(\boldsymbol{x}+\boldsymbol{u}_0)/|\nabla I_1(\boldsymbol{x}+\boldsymbol{u}_0)|^2 & \text{if } |\rho(\boldsymbol{v}^k;\boldsymbol{u}_0)| \le \lambda\theta|\nabla I_1(\boldsymbol{x}+\boldsymbol{u}_0)|^2. \end{cases} \tag{3.15}$$

This process which solves (3.10) can be used in both the minimization steps in the *texture-flow* procedure Algorithm 9. We use the Python TV-$L1$ implementation available from the Scikit-image image processing library [WSN14] when solving (3.10).

3.3.2 Multi-Fidelity Numerics

We use a splitting-like approach for the multi-fidelity minimization problem (3.9) as well. Splitting strategies, in addition to providing a tractable approach to (3.4), have the additional benefit of simplifying the process of including additional terms in the fidelity (see [BBV23a] for instance). This is a distinct advantage over the numerical approach in [BBP04], where a several-term fidelity is considered for optical flow tasks. The minimization approach in [BBP04] requires defining a new discretization based on the Euler-Lagrange equation for each fidelity and regularization functional used therein. This is not only inconvenient, but each specific model may require significant changes in discretization strategies in order to maintain numerical stability, and semi-implicit or fully implicit approaches may be necessary.

However, with a splitting approach, one can quickly stack several fidelity terms and easily test the influence of, for example, including a texture-based feature in the *multi-fidelity* problem (3.8). We are able to solve problems in the form of (3.11) and (3.12), so our goal is to use splitting to recast (3.9) into these forms.

We introduce new variables $\boldsymbol{u}^j$ for $j = 1, 2, \ldots, N+1$—one for each of the N fidelity terms and $J(\cdot)$—and consider the constrained optimization problem,

$$\begin{aligned} &\underset{\boldsymbol{u}^1,\boldsymbol{u}^2,\ldots,\boldsymbol{u}^{N+1}}{\arg\min} \sum_{j=1}^{N} \lambda_j \int_\Omega \psi\left(\rho_j(\boldsymbol{u}^j;\boldsymbol{u}_0,I)\right) d\boldsymbol{x} + J\left(\boldsymbol{u}^{N+1}\right) \\ &\text{subject to } \ \boldsymbol{u}^1 = \boldsymbol{u}^2 = \cdots = \boldsymbol{u}^{N+1}. \end{aligned} \tag{3.16}$$

This is equivalent with (3.9) and we can relax (3.16) by enforcing the quadratic penalty

$$Q_{N+1}(x_1, x_2, \ldots, x_{N+1}, \theta) := \frac{1}{2\theta} \sum_{i=1}^{N+1} \sum_{j>i} \|x_i - x_j\|^2,$$

giving the unconstrained problem

$$\underset{\boldsymbol{u}^1,\boldsymbol{u}^2,\ldots,\boldsymbol{u}^{N+1}}{\arg\min} \sum_{j=1}^{N} \lambda_j \int_\Omega \psi\left(\rho_j(\boldsymbol{u}^j;\boldsymbol{u}_0,I)\right) d\boldsymbol{x} + J\left(\boldsymbol{u}^{N+1}\right) + Q_{N+1}\left(\boldsymbol{u}^1, \boldsymbol{u}^2, \ldots, \boldsymbol{u}^{N+1}, \theta\right). \tag{3.17}$$

The parameter $\theta > 0$, and as $\theta \downarrow 0$, (3.17) becomes equivalent with (3.9) and (3.16).

Splitting (3.17) into simple subproblems is not immediately clear, since the quadratic penalty term, Q_{N+1}, now contains $N(N+1)/2$ distinct quadratic terms and for any fixed k, $Q_{N+1}(\boldsymbol{u}^1, \boldsymbol{u}^2, \ldots, \boldsymbol{u}^{N+1}, \theta)$ contains N terms which depend on $\boldsymbol{u}^k$. However, Lemma 3.3.1 will allow splitting to proceed since it shows for any fixed k, Q_{N+1} can be equivalently viewed as a quadratic difference between $\boldsymbol{u}^k$ and the average of the remaining $\boldsymbol{u}^j$, $j \neq k$.

Lemma 3.3.1. *Let $\theta > 0$ and $N \geq 1$ be a fixed integer. For any $k \in \{1, 2, \ldots, N+1\}$, the quadratic function*

$$Q_{N+1}\left(\boldsymbol{u}^1, \boldsymbol{u}^2, \ldots, \boldsymbol{u}^{N+1}, \theta\right) := \frac{1}{2\theta} \sum_{i=1}^{N+1} \sum_{j>i}^{N+1} \|\boldsymbol{u}^i - \boldsymbol{u}^j\|^2 = \frac{N}{2\theta} \left\| \boldsymbol{u}^k - \frac{1}{N} \sum_{j\neq k} \boldsymbol{u}^j \right\|^2 + c(\boldsymbol{u}^{j\neq k}), \tag{3.18}$$

where $c(\boldsymbol{u}^{j\neq k})$ is a function independent of $\boldsymbol{u}^k$.

Proof. The result is direct. Suppose $\theta > 0$ and $N \geq 1$. Let $k \in \{1, 2, \ldots, N+1\}$. Since Q is symmetric about its first $N+1$ arguments, we may assume $k = 1$ without loss of generality.

Now,

$$
\begin{aligned}
N\left\|\boldsymbol{u}^k - \frac{1}{N}\sum_{j\neq k}\boldsymbol{u}^j\right\|^2 &= N\left\|\boldsymbol{u}^1 - \frac{1}{N}\sum_{j=2}^{N+1}\boldsymbol{u}^j\right\|^2 \\
&= N\left(\|\boldsymbol{u}^1\|^2 - 2\left\langle \boldsymbol{u}^1, \frac{1}{N}\sum_{j=2}^{N+1}\boldsymbol{u}^j\right\rangle + \left\|\frac{1}{N}\sum_{j=2}^{N+1}\boldsymbol{u}^j\right\|^2\right) \\
&= \sum_{j=2}^{N+1}\left(\|\boldsymbol{u}^1\|^2 - 2\langle \boldsymbol{u}^1, \boldsymbol{u}^j\rangle + \|\boldsymbol{u}^j\|^2\right) + c(\boldsymbol{u}^{j\neq 1}) \\
&= \sum_{j=2}^{N+1}\left\|\boldsymbol{u}^1 - \boldsymbol{u}^j\right\|^2 + c(\boldsymbol{u}^{j\neq 1}), \qquad (3.19)
\end{aligned}
$$

where we redefine $c(\boldsymbol{u}^{j\neq 1})$ from line to line to absorb terms dependent on $\boldsymbol{u}^j$, $j \neq k = 1$ as necessary. Then, by definition

$$
\begin{aligned}
Q_{N+1}\left(\boldsymbol{u}^1, \boldsymbol{u}^2, \ldots, \boldsymbol{u}^{N+1}, \theta\right) &= \frac{1}{2\theta}\sum_{i=1}^{N+1}\sum_{j>i}^{N+1}\|\boldsymbol{u}^i - \boldsymbol{u}^j\|^2 \\
&= \frac{1}{2\theta}\sum_{j=2}^{N+1}\left\|\boldsymbol{u}^1 - \boldsymbol{u}^j\right\|^2 + \frac{1}{2\theta}\sum_{i=2}^{N+1}\sum_{j>i}^{N+1}\|\boldsymbol{u}^i - \boldsymbol{u}^j\|^2 \\
&= \frac{1}{2\theta}\sum_{j=2}^{N+1}\left\|\boldsymbol{u}^1 - \boldsymbol{u}^j\right\|^2 + c(\boldsymbol{u}^{j\neq 1}). \qquad (3.20)
\end{aligned}
$$

Multiplying (3.19) by $1/2\theta$ and substituting in (3.20), one immediately has the result. □

We can now approach (3.17) by alternatingly minimizing over each of the $N+1$ variables $\boldsymbol{u}^1,\ \boldsymbol{u}^2, \ldots, \boldsymbol{u}^{N+1}$. In the k-th subproblem, we will only include the terms of (3.17) which directly depend on $\boldsymbol{u}^k$, and because of the lemma these subproblems are simplified into a fidelity term plus a single quadratic. Our split iterative scheme is given below.

$$
\begin{aligned}
\boldsymbol{u}^{1,n+1} &\in \arg\min_{\boldsymbol{u}} \lambda_1 \int_\Omega \psi\left(\rho_1(\boldsymbol{u};\boldsymbol{u}_0,I)\right)d\boldsymbol{x} + \frac{1}{2\theta}\left\|\boldsymbol{u} - \frac{1}{N}\sum_{j=2}^{N+1}\boldsymbol{u}^{j,n}\right\|^2 \\
\boldsymbol{u}^{2,n+1} &\in \arg\min_{\boldsymbol{u}} \lambda_2 \int_\Omega \psi\left(\rho_2(\boldsymbol{u};\boldsymbol{u}_0,I)\right)d\boldsymbol{x} + \frac{1}{2\theta}\left\|\boldsymbol{u} - \frac{\boldsymbol{u}^{1,n+1}}{N} - \frac{1}{N}\sum_{j=3}^{N+1}\boldsymbol{u}^{j,n}\right\|^2 \\
&\vdots \\
\boldsymbol{u}^{k,n+1} &\in \arg\min_{\boldsymbol{u}} \lambda_k \int_\Omega \psi\left(\rho_k(\boldsymbol{u};\boldsymbol{u}_0,I)\right)d\boldsymbol{x} + \frac{1}{2\theta}\left\|\boldsymbol{u} - \frac{1}{N}\sum_{j<k}\boldsymbol{u}^{j,n+1} - \frac{1}{N}\sum_{j>k}^{N+1}\boldsymbol{u}^{j,n}\right\|^2 \\
&\vdots \\
\boldsymbol{u}^{N,n+1} &\in \arg\min_{\boldsymbol{u}} \lambda_N \int_\Omega \psi\left(\rho_N(\boldsymbol{u};\boldsymbol{u}_0,I)\right)d\boldsymbol{x} + \frac{1}{2\theta}\left\|\boldsymbol{u} - \frac{1}{N}\sum_{j<N}\boldsymbol{u}^{j,n+1} - \frac{1}{N}\boldsymbol{u}^{N+1,n}\right\|^2 \\
\boldsymbol{u}^{N+1,n+1} &\in \arg\min_{\boldsymbol{u}} J(\boldsymbol{u}) + \frac{1}{\theta}\left\|\boldsymbol{u} - \frac{1}{N}\sum_{j=1}^{N}\boldsymbol{u}^{j,n+1}\right\|^2,
\end{aligned}
\tag{3.21}
$$

where we iterate this process in n, starting with some initialization for $\boldsymbol{u}^{j,0}$, $j = 1, \ldots, N+1$ (usually the current flow estimate $\boldsymbol{u}_0$). Again, the problem (3.17) has been reduced into proximal problems and an ROF total variation denoising problem, which can be solved via the explicit formula (3.15) and dual projection procedure (3.13) when $\psi(\cdot)$ and $J(\cdot)$ are chosen to be the TV-$L1$ components.

Computing the derivative terms in these schemes is done with finite differences. For speed we make use of the Python `NumPy` built-in gradient function, which uses central differences to compute gradient terms in (3.15) while forward and backward differences are composed when computing the gradient-of-divergence terms in (3.13).

In order to overcome inaccuracies from large displacements (in terms of pixels), we also incorporate a hierarchical pyramid scheme (see [ABH93]), downsampling I_1 and I_0 until the resolution is reduced sufficiently so that the displacements no longer span many gridpoints.

Then, recoveries are determined at each resolution level, starting from the coarsest and working towards the original resolution, with each recovery acting as an initialization for the next pyramid level. The full algorithm with warping for the multi-fidelity flow (3.17) using the form (3.21) is given in Algorithm 10.

3.4 Results

In this section we discuss the results of our texture-based and multi-fidelity flows and compare them against existing optical flow methods. In particular, we will compare against the classic HS optical flow method as well as the popular TV-$L1$ scheme, which recently has shown success [YSP22, YPW23] over the conventional feature-tracking algorithm which is widely used, including for the National Oceanic and Atmospheric Administration (NOAA) Geostationary Operational Environmental Satellites (GOES) cloud motion retrieval. However, since the texture-flow and multi-fidelity flow approaches are based on general fidelity and regularization functionals (ψ, ρ and J) , the same approaches can be applied to other more sophisticated optical flow processes.

Recall we are interested in determining a vector field describing the atmospheric wind velocity given a sequence of water-vapor images. A typical water vapor image is given in Fig. 3.5. In all our experiments, we take in two images to guide flow retrieval which are separated by $2\Delta t$, where Δt is the time between frames in the image sequence. The wind velocity associated with the intermediate frame is taken as the ground truth, and the estimated vector field is compared against the ground truth using root-mean-squared-vector-distance (RMSVD). The RMSVD between two vector fields $\boldsymbol{u}, \boldsymbol{v}$ defined on an $M \times N$ pixel grid is

$$RMSVD(\boldsymbol{u}, \boldsymbol{v}) = \sqrt{\frac{1}{MN} \sum_{i=1}^{M} \sum_{j=1}^{N} (v_{1,ij} - u_{1,ij})^2 + (v_{2,ij} - u_{2,ij})^2}.$$

Algorithm 10 Multi-fidelity Optical Flow

INPUT: images I_0 and I_1, initial flow estimate $\boldsymbol{u}_0$, parameters N_{warp}, N_{iter}, N_{levels}, θ, and λ_j for $j = 1, 2, \ldots, N$. Here, N is the number of fidelity terms $\rho_j(\cdot)$ used. Precompute the features used within each $\rho_j(\cdot)$ at this stage (for instance I_0^{Text}, I_1^{Text}, and any gradients).

OUTPUT: flow estimate $\boldsymbol{u}$

```
Form downsampled pyramid levels $I_{0,l}$, $I_{1,l}$, $\boldsymbol{u}_{0,l}$, $l = 1, 2, \ldots, N_{levels}$.
for $l = N_{levels}, \ldots, 2, 1$ do
    for  $k = 0, 1, \ldots, N_{warp} - 1$ do
        Compute $I_{1,l}(\boldsymbol{x} + \boldsymbol{u}_{k,l})$, $\nabla I_{1,l}(\boldsymbol{x} + \boldsymbol{u}_{k,l})$, and the equivalent for any additional features used by each $\rho_j(\cdot)$ (for e.g. $I_{1,l}^{Text}(\boldsymbol{x} + \boldsymbol{u}_{k,l})$ ) via bicubic uplook.
        $\boldsymbol{u}^{j,0} \leftarrow \boldsymbol{u}_{k,l}$, $j = 1, 2, \ldots, N + 1$
        for $n = 0, 1, \ldots, N_{iter} - 1$ do
            for j=1,2,...,N do
                $$\boldsymbol{u}^{j,n+1} \in \arg\min_{\boldsymbol{u}} \lambda_j \int_\Omega \psi\left(\rho_j(\boldsymbol{u}; \boldsymbol{u}_0, I)\right) d\boldsymbol{x} + \frac{1}{2\theta} \left\| \boldsymbol{u} - \frac{1}{N} \sum_{i<j} \boldsymbol{u}^{i,n+1} - \frac{1}{N} \sum_{i>j}^{N+1} \boldsymbol{u}^{i,n} \right\|^2$$
            end for
            $$\boldsymbol{u}^{N+1,n+1} \in \arg\min_{\boldsymbol{u}} J(\boldsymbol{u}) + \frac{1}{\theta} \left\| \boldsymbol{u} - \frac{1}{N} \sum_{i=1}^{N} \boldsymbol{u}^{i,n+1} \right\|^2$$
        end for
        $\boldsymbol{u}_{k+1,l} \leftarrow \boldsymbol{u}^{N+1,N_{iter}}$
    end for
    $\boldsymbol{u}_{0,l+1} \leftarrow \text{Upsample}(\boldsymbol{u}_{N_{warp},l})$
end for
Return $\boldsymbol{u}_{N_{warp},N_{levels}}$
```

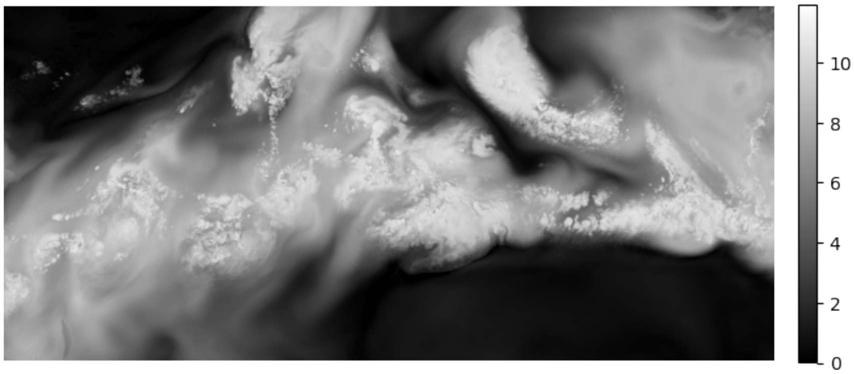


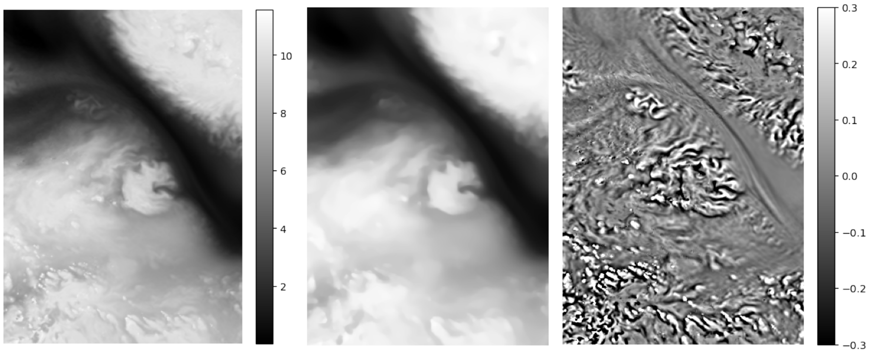
Figure 3.5: A typical water-vapor image, in this case from the MCS dataset. Pixel values correspond to the water vapor mixing ratio in g/kg.

We also make use of a hierarchical pyramid scheme for each method, downsampling by a factor of 2 for 10 levels (or until the minimum dimension of the image is 16 pixels).

As discussed in Section 3.2, using optical flow for wind velocity estimation can be challenging for a variety of reasons. We are considering several datasets, and the ground truth displacements between frames can be large (in terms of pixels), violating the linearization assumptions when forming the OFCE. On the other hand, in the case of very coarse spatial resolution such as with the ETC dataset, the displacements can be quite small, so that the ground truth motion vectors are at a subpixel level. These issues are handled by a combination of the hierarchical pyramid and warping routines which are used in concert with the texture and multi-fidelity flow methods proposed here.

3.4.1 Texture-Flow Results

In addition the the standard challenges of velocity estimation within a scene, atmospheric motion is complex and turbulent, and large regions of uniform water vapor within the data domain provide little information for a flow estimator to extract a velocity estimate accurately. This observation is the primary motivation for incorporating image texture as a feature for an optical flow procedure. In our texture-flow scheme, we decompose an image into its structural and textural portions, as shown for a crop of the MCS dataset in Fig. 3.6, and use the textured portion for motion estimation. In Fig. 3.7, we visualize the wind velocity field using a colormap, where color indicates the direction of the wind and saturation indicates the speed. Our tests extracting wind velocity estimates using the textured portion, I^{Text}, of an image alone demonstrate that texture is an excellent feature for atmospheric motion tracking. That is, recovering $\boldsymbol{u}^{Text}$ without any additional refinement in Algorithm 9. This is made clear in Fig. 3.7d when compared with 3.7c. Fig. 3.7d is obtained from only the information contained in I^{Text}, while 3.7c is formed from the whole image, yet 3.7d outperforms the TV-$L1$ estimate 3.7c. Of note, we see sizable improvement within the boxed regions of 3.7d, which represent portions of I with relatively uniform water-vapor levels (see Fig. 3.5). This indicates that texture indeed provides good context within these challenging regions, and our motivation for including texture in the flow retrieval process is valid. Adding the refinement process as described in Algorithm 9 provides a small additional improvement, as shown in 3.7e.



(a) **(b)** **(c)**

Figure 3.6: Structure-texture decomposition of a cropped portion of Fig. 3.5. The original image I is given in (a), it's structure portion I^S in (b), and textural portion I^{Text} in (c). Recall $I = I^S + I^{Text}$.

Similar flow visualizations are given for the TC and ETC datasets in Figures 3.9 and 3.10. The texture-flow velocity fields produce 6–42% improved flow estimates compared with HS and 1–20% improvements over the TV-$L1$ methods across the three datasets. Full results are recorded in Table 3.2. We remark that lower performance in units m/s does not necessarily imply lower performance relative to the resolution of the data collected. For instance, since the ETC dataset has a very coarse spatial resolution (12,000m per pixel) and moderate temporal resolution ($\Delta t = 120$s), an error of 1 m/s corresponds to 0.01 px/Δt, so results need to be taken in context with the data specifications. Within the MCS dataset, we see the greatest improvements (20% over TV-$L1$ and 42% over HS), as the conventional optical flow estimates produce flow estimates with errors near 1 px/Δt.

3.4.2 Multi-Fidelity Results

For the multi-fidelity optical flow process described in Section 3.3.2, we consider incorporating I^{Text}, ∇I and $|\nabla I|$ as features in addition to the full image data I. That is, in (3.21) we take $N = 2, 3, 4$ fidelity terms and define $\rho_1(\boldsymbol{u}; I) = I_1(\boldsymbol{x} + \boldsymbol{u}) - I_0(\boldsymbol{x})$, $\rho_2(\boldsymbol{u}; I) = I_1^{Text}(\boldsymbol{x}+\boldsymbol{u}) - I_0^{Text}(\boldsymbol{x})$, $\rho_3(\boldsymbol{u}; I) = \nabla I_1(\boldsymbol{x}+\boldsymbol{u}) - \nabla I_0(\boldsymbol{x})$ and $\rho_4(\boldsymbol{u}; I) = |\nabla I_1|(\boldsymbol{x}+\boldsymbol{u}) - |\nabla I_0|(\boldsymbol{x})$. We keep $\psi(x) = |x|$ and set $J(\,\cdot\,)$ to be the total variation penalty. For brevity, we'll denote the multi-fidelity scheme assuming constancy on I and I^T by MF(I, I^T), MF$(I, I^{Text}, \nabla I)$ when we additionally consider the gradient, and MF$(I, I^{Text}, \nabla I, |\nabla I|)$ when including all the features.

The MCS data, which has the lowest temporal resolution (15 minutes between frames), has the largest displacements and consequently provides a better measurement of the accuracy within a flow estimate since the displacement ratio in pixels/Δt is the largest. That is, equally sized errors in velocity (m/s) create the largest noticed error within the data (pixels/Δt) for the MCS dataset. So, we test the multi-fidelity process for all the

Table 3.2: RMSVD in m/s and px/Δt between flow estimations and ground truth wind velocity. Here, Δt is the time between frames for each respective dataset. Bold entries indicate lowest error in each row.

	HS		TVL1		Texture-flow		MF(I, I^T)	
	m/s	px/Δt	m/s	px/Δt	m/s	px/Δt	m/s	px/Δt
MCS	3.400	1.020	2.483	0.745	**1.985**	**0.596**	**1.985**	**0.596**
TC	2.903	0.060	2.462	0.051	2.361	0.049	**2.299**	**0.047**
ETC	6.205	0.062	5.855	0.059	5.838	0.058	**5.675**	**0.057**

Table 3.3: Results of additional features on the MCS dataset. Bold entries indicate lowest error.

TVL1		Texture-flow		MF(I, I^T)		MF$(I, I^T, \nabla I)$		MF$(I, I^T, \nabla I, \|\nabla I\|)$	
m/s	px/Δt	m/s	px/Δt	m/s	px/Δt	m/s	px/Δt	m/s	px/Δt
2.483	0.745	1.985	0.596	1.985	0.596	1.748	0.524	**1.745**	**0.524**

features on the MCS data, as shown in Figures 3.7f, 3.7g, 3.7h. A vector field plot of MF$(I, I^{Text}, \nabla I, |\nabla I|)$ is also shown in Figure 3.8 which gives a better sense of the motion within the recovered flow.

We additionally test MF(I, I^{Text}) on the TC and ETC datasets shown in Figures 3.9e and 3.10e. We see improved performance as features are added. In particular, for the MCS dataset, we see a boost to 20–30% improvement over TV-$L1$ and 42–49% over HS. For the ETC and TC datasets, we see 4-7% improvement over TV-$L1$ and 9–20% reduction in error over HS from MF(I, I^{Text}). The results for MF(I, I^{Text}) across all datasets are recorded in Table 3.2 and the impact of additional features on the MCS dataset are summarized in Table 3.3.

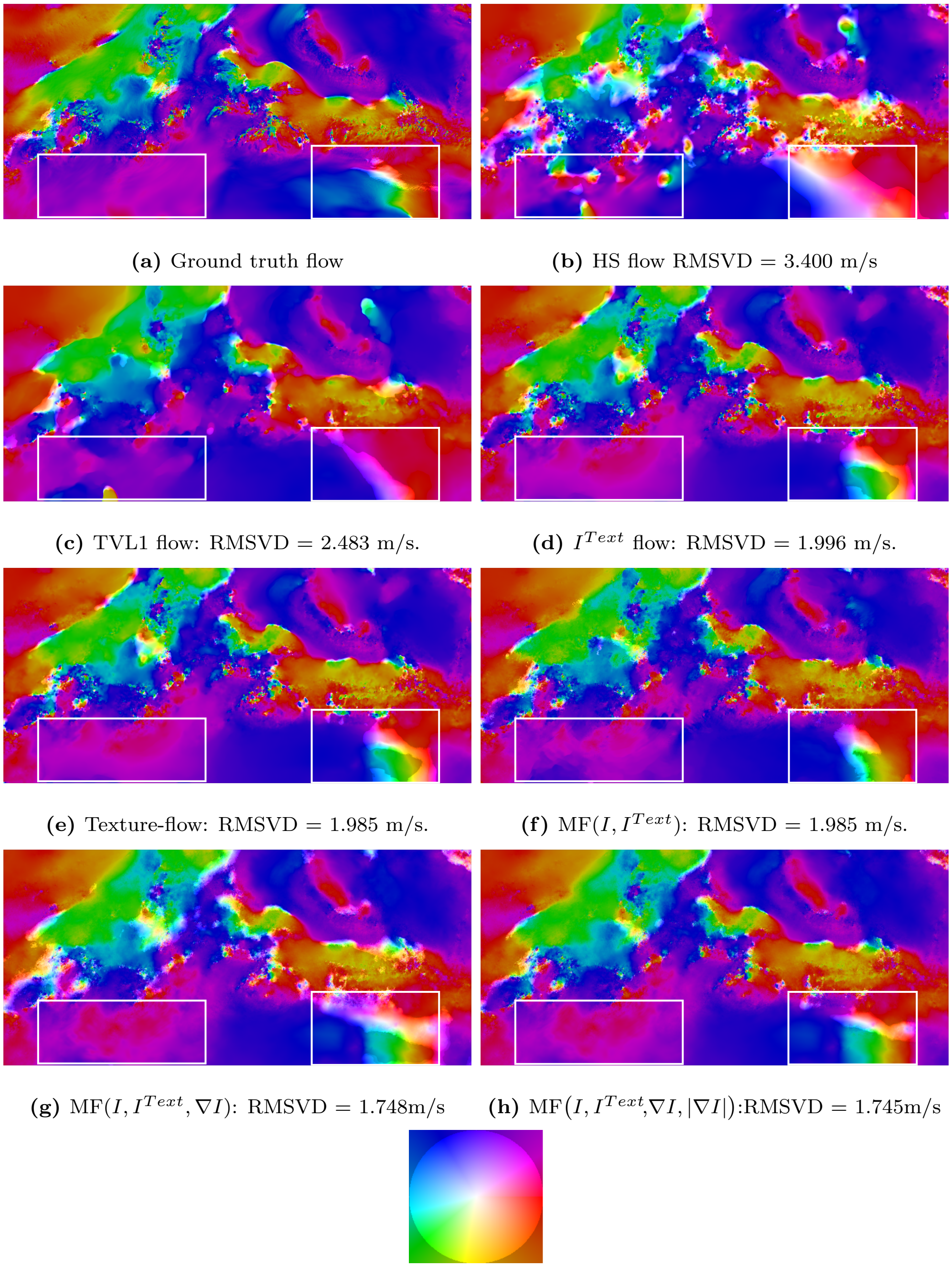
3.5 Conclusion

In this work we demonstrate that optical flow can effectively perform wind velocity estimation from remote sensing water-vapor data. Additionally, we show that texture is a pertinent feature for motion estimation and we introduce two texture-based optical flow procedures.

The first method, “texture-flow”, arrived from observing that the optical flow constraint equation (OFCE) on which the vast majority of variational optical flow methods are built, cannot recover the desired complex flow of atmospheric motion alone. Modifying the regularization term or considering a non-linear OFCE approach is insufficient, and modified or additional data fidelity terms should be considered. We determine that image-texture is a viable feature for optical flow, and we incorporate this feature by augmenting the fidelity with a textural portion of the data.

A second “multi-fidelity” method extends the idea from texture-flow by developing a convenient algorithm which has the advantage of easily handling additional fidelity terms, and can be extended to include many more image-features in the flow retrieval process. In future work, we would like to test additional image-features as well as apply this multi-fidelity approach to other optical flow approaches including non-local methods.

Collectively, these methods perform well on convective and cylconic weather systems, and this research would benefit the microwave and infrared instrument systems that commonly record such data. Both methods outperform the Horn Schunck and TV-$L1$ schemes across varying spatial and temporal resolutions. Moreover, the dense flow estimates only require two frames of data, and as such can produce “real-time” weather updates without the need to collect extended-duration data samples.



(a) Ground truth flow

(b) HS flow RMSVD = 3.400 m/s

(c) TVL1 flow: RMSVD = 2.483 m/s.

(d) I^{Text} flow: RMSVD = 1.996 m/s.

(e) Texture-flow: RMSVD = 1.985 m/s.

(f) MF(I, I^{Text}): RMSVD = 1.985 m/s.

(g) MF$(I, I^{Text}, \nabla I)$: RMSVD = 1.748m/s

(h) MF$\big(I, I^{Text}, \nabla I, |\nabla I|\big)$:RMSVD = 1.745m/s

(i) Color-key

Figure 3.7: MCS flow visualizations. Boxed regions indicate areas of uniform water-vapor levels where HS and TV-$L1$ algorithms struggle to accurately estimate the flow. Figs. 3.7d-3.7f show the improvement texture features add to velocity estimation. The color in 3.7i indicates velocity direction and saturation indicates speed. Partially reproduced from [BBV23a] ©2023 IEEE.

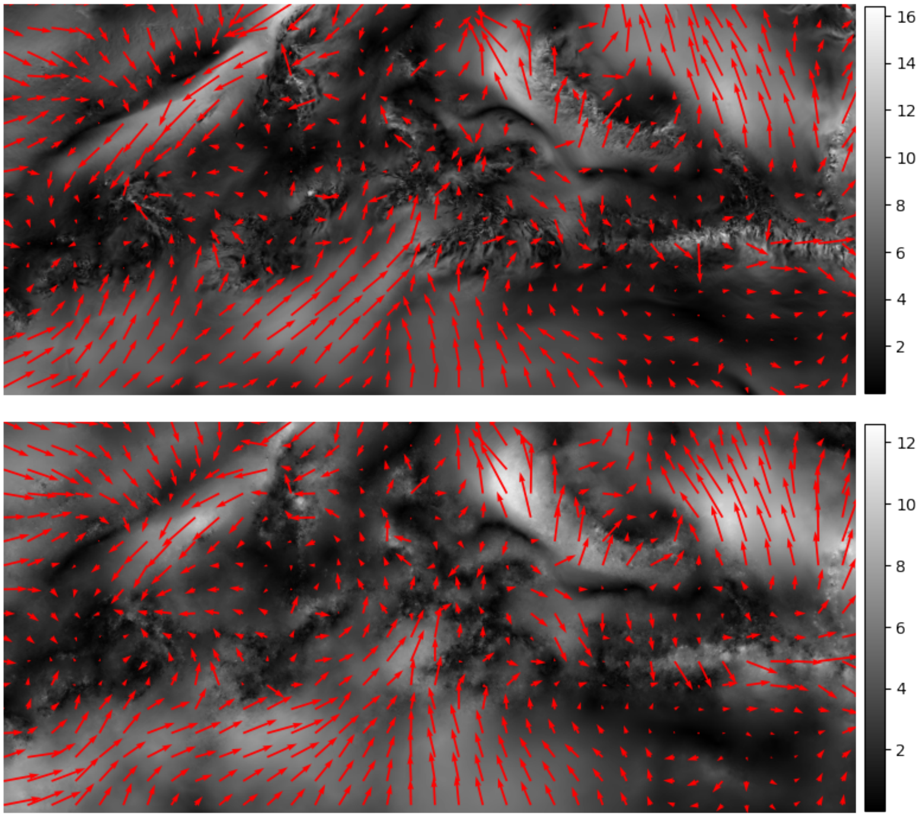
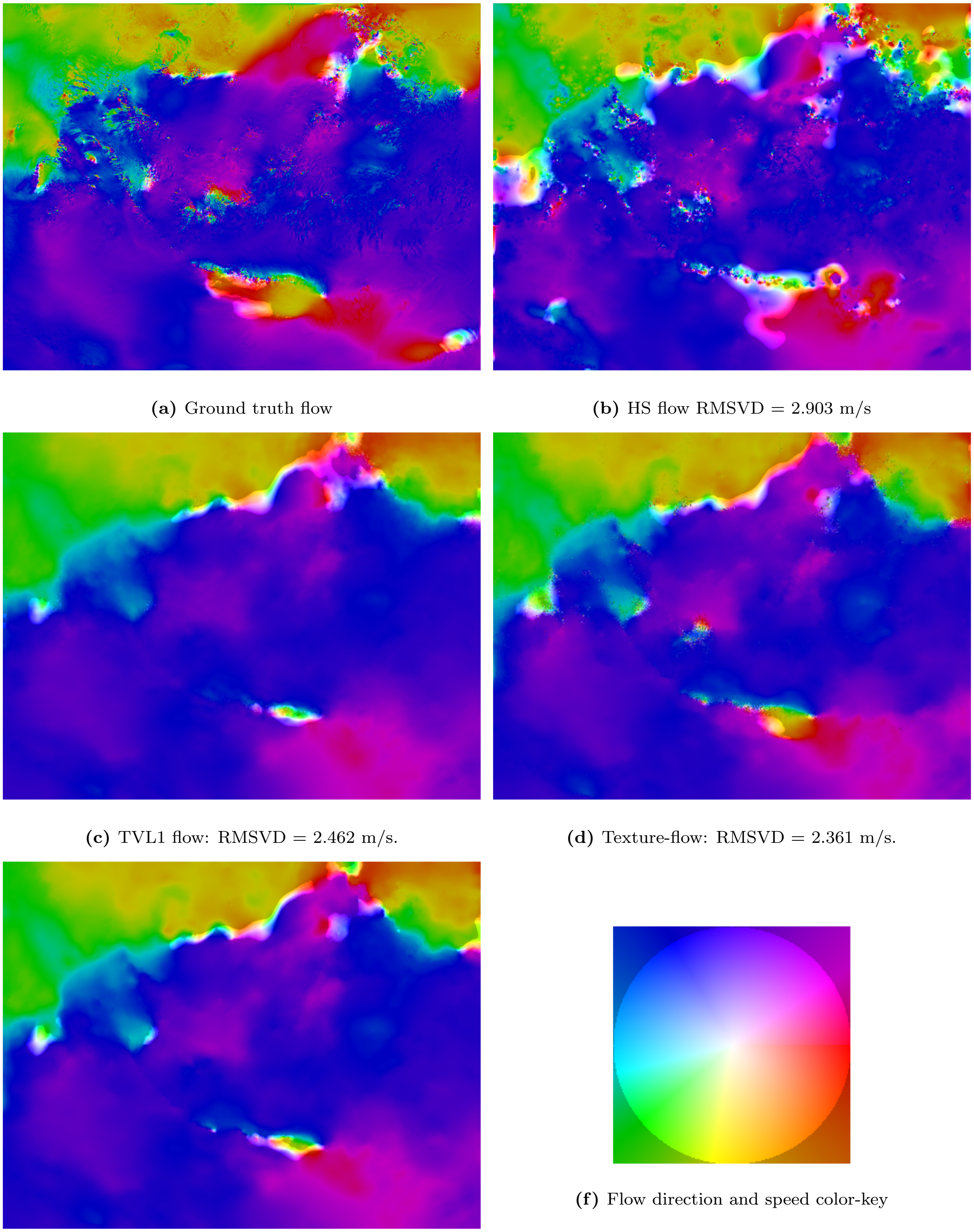


Figure 3.8: Ground truth (top) and MF$(I, I^T, \nabla I, |\nabla I|)$ (bottom) vector-fields overlaid on wind-speed. Image intensity is wind-speed in m/s. RMSVD = 1.745 m/s. Reproduced from [BBV23a] ©2023 IEEE.



(a) Ground truth flow

(b) HS flow RMSVD = 2.903 m/s

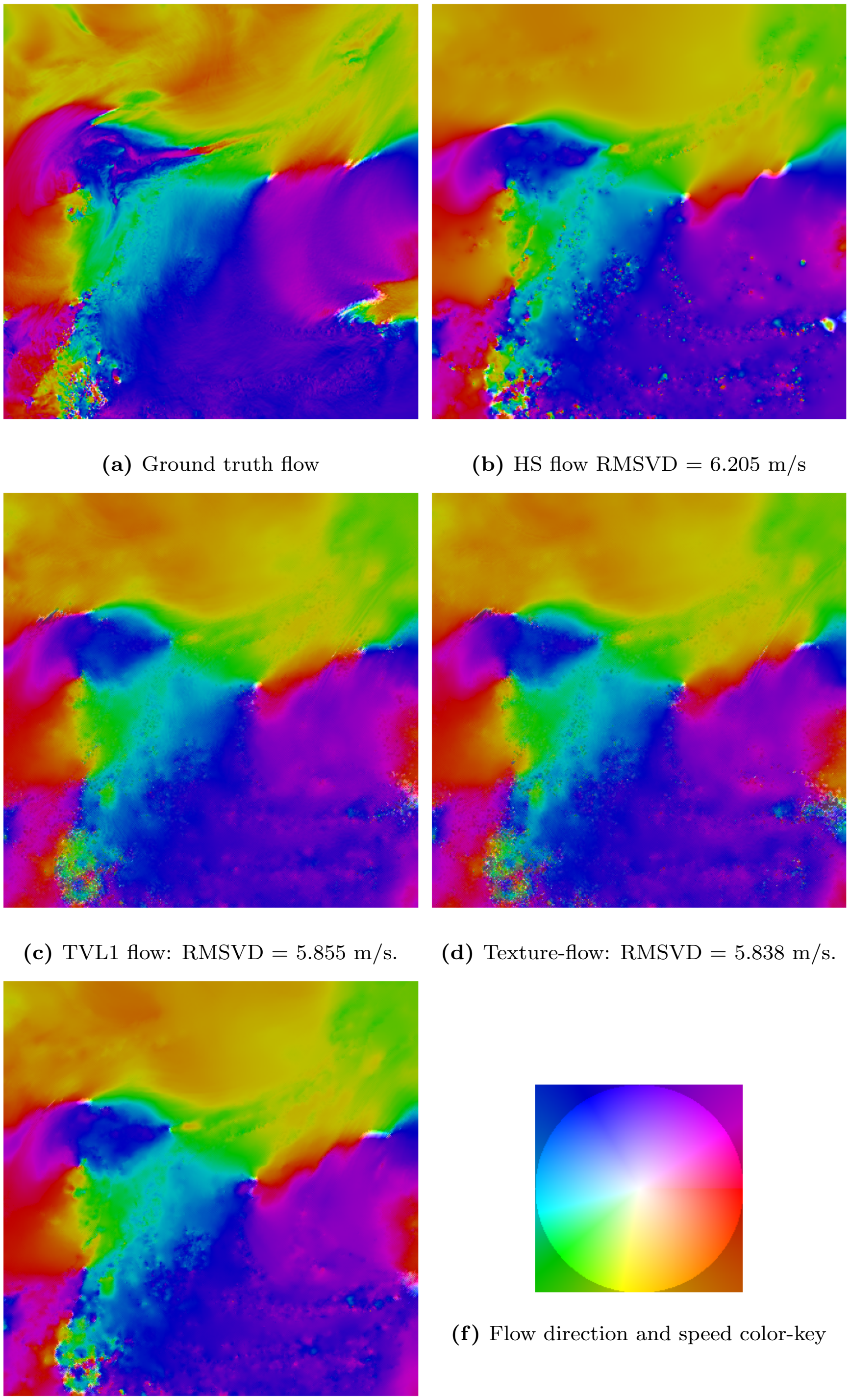
(c) TVL1 flow: RMSVD = 2.462 m/s.

(d) Texture-flow: RMSVD = 2.361 m/s.

(e) MF(I, I^{Text}): RMSVD = 2.299 m/s.

(f) Flow direction and speed color-key

Figure 3.9: Flow visualization for tropical convection (TC) data.



(a) Ground truth flow

(b) HS flow RMSVD = 6.205 m/s

(c) TVL1 flow: RMSVD = 5.855 m/s.

(d) Texture-flow: RMSVD = 5.838 m/s.

(f) Flow direction and speed color-key

(e) MF(I, I^{Text}): RMSVD = 5.675 m/s.

Figure 3.10: Flow visualization of extratropical cylcone (ETC) data.

CHAPTER 4

Data Driven Observable Discovery for Reduced-Order Modeling of Turbulence Based on the Mori-Zwanzig Formalism [1]

4.1 Introduction

In the prior chapters, we were concerned with inverse problems which extract some desired information after it has been disguised by some forward process (e.g. the formation of a noise-corrupted image or the evolution of wind-driven vapor fields). The extracted information (a restored image or the wind vector field) was the ultimate aim. In this chapter, we consider another inverse problem which has a different goal. Instead of retaining the inverted quantity, we use it to improve a forward (simulation) process.

In particular, the forward problem we consider is reduced-order modeling for turbulent fluid simulation. Given velocity and pressure components of a fluid, the Navier-Stokes equations dictate the resulting fluid motion. However, the flow is notoriously difficult to evolve forward in time due to the dimension of the system becoming prohibitively large when simulations are run at sufficiently high resolutions to form accurate flows. At lower resolutions, things are no easier since the dynamics cannot be fully resolved and inaccuracies are rapidly

[1] Research presented in this chapter was supported by the Laboratory Directed Research and Development program of Los Alamos National Laboratory under project number 20220104DR. It is released under LA-UR-24-24745.

introduced due to the missing subgrid information. In response, a plethora of subgrid models have been introduced which attempt to determine the influence of scales below that of the current resolution.

Recent work [LTL21, TLA21, LTP23] has demonstrated that a model reduction process based on the Mori-Zwanzig formalism can be successfully employed to capture subgrid effects. However, inherent to the method is a choice of reduced order variables, and that choice can dramatically affect the quality of the resulting reduced order model's success. We are unaware of any method which provides an optimal way to choose these resolved variables for Mori-Zwanzig based procedures. This inverse problem—determining an appropriate set of reduced order variables—is the motivation for this work.

Before beginning, we remark that there is a similar problem within Koopman based approaches for dynamical systems. A Koopman operator is a linear operator which can advance a non-linear (finite dimensional) dynamical system, but on an infinite-dimensional Hilbert space consisting of *observation* functions of the state. Several approaches based on the Koopman theory have leveraged the universal approximation capabilities of neural networks to help find a reduced (finite dimensional) *observable* set which can be linearly advanced by an approximate Koopman operator [WKR15, YKH19, LKB18, OR19]. These techniques, aptly dubbed *Deep Koopman*, make use of auto-encoders to determine potential observables.

In a similar vein, a very recent paper [GSS23] imitates Deep Koopman but for a Mori-Zwanzig based approach. However, we distinguish [GSS23] from our work in two ways. First, the specific structure required by the Mori-Zwanzig formalism is not enforced in [GSS23] when learning the memory kernels. Rather, a black-box Long Short Term Memory (LSTM) cell is used to incorporate memory effects as a correction to a Koopman based Markov model. This is an important distinction and is emphasized in [LTP23], which notes research such as [MW18] introduce the Mori-Zwanzig formalism as a theoretical basis for reduced order modeling before turning to a recurrent neural network with LSTM blocks to incorporate memory into their model. No structure is built in to ensure that the forward model, indeed, follows the generalized fluctuation-dissipation relationship as specified within a Mori-Zwanzig system. A similar oversight is performed in [HJL21]. While memory dependence is a key property of the Mori-Zwanzig approach, not all memory based methods fit the Mori-Zwanzig structure and some care must be taken.

Second, the full state is provided to the model in [GSS23]. While the reduced order variables are functions of the full state within the Mori-Zwanzig formalism, in practice access to the full state cannot always be expected, and so it is worthwhile to consider models which rely only on reduced versions of the known state which might be anticipated to be accessible.

In what follows, we propose a data driven approach to discovering a *good* set of these reduced order variables, where by *good* we mean ones which improve the Mori-Zwanzig based reduced order model performance. We make use of a convolutional neural network to select features from reduced order versions of the full state, and evolve these features forward using Markov and memory kernels derived from regression-based projections as detailed in the Mori-Zwanzig formalism. In Section 4.2, we introduce the Koopman and Mori-Zwanzig approaches to modeling dynamical systems. Then, we discuss the data driven process for a Mori-Zwanzig framework in Section 4.3 and the resulting inverse problem of choosing a set of reduced order variables. Details and results are discussed in Section 4.4.

4.2 Modeling of Dynamical Systems

Dynamical systems are adept at representing the evolution of a physical system. The evolved variables can be interpreted as physical quantities representing, for instance, position, momentum, velocity or acceleration of a particle within a many-body system. The value of the variables at any point in time describe the *state* of the system, and that name is given to the collection of these variables. In this chapter, we consider an autonomous dynamical system

$$\frac{d}{dt}\boldsymbol{\phi}(t) = \boldsymbol{R}(\boldsymbol{\phi}),\ \boldsymbol{\phi}(0) = \boldsymbol{\phi}_0, \tag{4.1}$$

where the state is denoted $\boldsymbol{\phi} : \mathbb{R} \to \mathbb{R}^D$, $\boldsymbol{R} : \mathbb{R}^D \to \mathbb{R}^D$ encodes the flow of the system, and the dynamics do not depend explicitly on time. The state components ϕ_i, $i = 1, \ldots, D$ are thought of as physical-space variables and we denote a solution of (4.1) satisfying the initial condition $\boldsymbol{\phi}_0 \in \mathbb{R}^D$ by $\boldsymbol{\phi}(t; \boldsymbol{\phi}_0)$.

The system may consist of many degrees of freedom ($D \gg 1$) or be governed by a complex and non-linear flow function $\boldsymbol{R}$ (often both) and consequently, a simplified model for (4.1) may be of interest, especially to those wishing to numerically evolve such a system.

To address a high dimension or complex system, a common approach is to consider a reduced order model (ROM). Roughly speaking, a ROM replaces some known mathematical model by a “smaller” model which still captures the essential dynamics of the original system. More precisely, one can consider the evolution of $\bar{\boldsymbol{\phi}} : \mathbb{R} \to \mathbb{R}^M$, where $\bar{\boldsymbol{\phi}}$ depends on $\boldsymbol{\phi}$ in some way, but lives in a significantly reduced space (i.e. $M \ll D$). Consider the motivating example

$$\begin{cases} \dot{\phi}_1(t) & = \phi_2(t),\ \phi_1(0) = 1 \\ \dot{\phi}_2(t) & = \phi_2^2/\phi_1 + \phi_1,\ \phi_2(0) = 1. \end{cases} \tag{4.2}$$

In the language of (4.1), this system has two states ($D = 2$) and $\boldsymbol{R} : \mathbb{R}^2 \to \mathbb{R}^2$ is a non-linear function of these states. The system has solution $\phi_1(t) = e^{t+t^2/2}$ and $\phi_2(t) = e^{t+t^2/2}(t+1)$, but in general it is extremely challenging to find an analytic solution to higher dimensional, non-linear differential equations. A reduced order variable for (4.2) could be the scalar function $\bar{\phi} : \mathbb{R} \to \mathbb{R}$, $\bar{\phi}(\phi_1(t), \phi_2(t)) = \phi_2/\phi_1$, since $\bar{\phi}$ satisfies

$$\begin{aligned}\frac{d\bar{\phi}}{dt} &= \frac{\partial\bar{\phi}}{\partial\phi_1}\frac{d\phi_1}{dt} + \frac{\partial\bar{\phi}}{\partial\phi_2}\frac{d\phi_2}{dt} \\ &= -\frac{\phi_2}{\phi_1^2}\dot{\phi}_1 + \frac{\dot{\phi}_2}{\phi_1} = -\bar{\phi}^2 + \bar{\phi}^2 + 1 = 1,\end{aligned}$$

with initial condition $\bar{\phi}(\phi_1(0), \phi_2(0)) = \phi_2(0)/\phi_1(0) = 1$. The ROM can be written,

$$\frac{d\bar{\phi}}{dt} = \bar{R}(\bar{\phi}) = 1, \; \bar{\phi}(0) = 1, \tag{4.3}$$

and tracking $\bar{\phi}$ reduces the size of the problem to half ($M = D/2$) that of evolving ϕ_1 and ϕ_2. In general, determining the reduced order variable $\bar{\phi}$ for which we can explicitly write the reduced flow function $\bar{R}$ which depends only on $\bar{\phi}$ (e.g. as in (4.3)) is not straightforward (or even possible). The variable $\bar{\phi}$ is often referred to as the resolved, coarse-grain or reduced variable and may not be as interpretable as a physical-space variable. For example, in (4.2) ϕ_1 might represent position while ϕ_2 velocity of a particle. Then, $\bar{\phi} = \phi_2/\phi_1$ is the ratio of velocity to position, and we see that this ratio $\bar{\phi}$ evolves with a constant velocity $\dot{\bar{\phi}} = 1$. Determining $\phi_i(t)$ themselves from $\bar{\phi}$ may not be possible, but the quantity may be of interest nonetheless.

Keeping in mind the objective of this chapter, we emphasize the *inverse problem* that could be posed to find the reduced order system (4.3). The forward problem is, given $\bar{\phi} = \bar{\phi}(\phi_1, \phi_2)$ and reduced flow map $\bar{R}$, evolve $\bar{\phi}$ forward in time from initial condition $\bar{\phi}(0)$ (e.g. Euler's method). Call this forward process up to time t, $\mathfrak{F}(\bar{\phi}(0), \bar{R}, t)$. The inverse problem, then, is given a forward process $\mathfrak{F}$ for system (4.2), infer a reduced order function $\bar{\phi}(\phi_1, \phi_2)$ and flow map $\bar{R}$ so that when the forward procedure is applied taking $\bar{\phi}(0)$ to $\bar{\phi}(t)$, $\mathfrak{F}(\bar{\phi}(0), \bar{R}, t) \approx \bar{\phi}(\phi_1(t), \phi_2(t))$. That is, solve

$$\underset{\bar{\phi}, \bar{R}}{\arg\min} \, \|\mathfrak{F}(\bar{\phi}(0), \bar{R}, \cdot) - \bar{\phi}(\phi_1(\cdot), \phi_2(\cdot))\|$$

where the minimization is taken over some class of order reductions $\bar{(\cdot)}$ and some norm $\|\cdot\|$.

4.2.1 The Koopman Approach

The Koopman approach to (4.1) is to consider a system which is updated by a linear operator $\mathcal{K}$ which acts on observables. Observables, which we denote $\boldsymbol{g}$, are L^2-integrable functions which sample or "observe" the full state $\boldsymbol{\phi}$. The operator $\mathcal{K} : L^2(\mathbb{R}^D, \mu) \to L^2(\mathbb{R}^D, \mu)$ gets its name from Bernard Koopman, who originally introduced the idea in 1931 to study Hamiltonian systems [Koo31]. Since its inception, a myriad of approaches have adopted Koopman theory for dynamical systems [Mez05, MM16], and this has spawned a plethora of numerical approaches including dynamic mode decomposition (DMD) [Sch10, BBP16] and extended DMD (eDMD) [WKR15] which make use of Koopman eigenfuctions to track the dynamics of a system.

Supplied with an inner product, the space of all observable functions of the state-space forms a Hilbert space, $\mathcal{H}$, on which the Koopman operator acts. If we denote the flow map $\mathcal{F}_t$ taking initial condition $\boldsymbol{\phi}_0$ to a solution $\boldsymbol{\phi}(t; \boldsymbol{\phi}_0)$, $t > 0$ of (4.1), then the time-continuous Koopman operator at $t > 0$ is

$$(\mathcal{K}_t \boldsymbol{g})(\boldsymbol{\phi}_0) := \boldsymbol{g} \circ \mathcal{F}_t(\boldsymbol{\phi}_0) = \boldsymbol{g}(\boldsymbol{\phi}(t; \boldsymbol{\phi}_0)), \tag{4.4}$$

for any function $\boldsymbol{g} \in L^2(\mathbb{R}^D, \mu)$ and initial condition $\boldsymbol{\phi}_0 \in \mathbb{R}^D$. Equation (4.4) can be interpreted two ways. On one hand, the left hand side is evolving the initial observation function, $\boldsymbol{g}$, so that one obtains a time-varying expression $(\mathcal{K}_t \boldsymbol{g})(t)$ which returns the time-appropriate value of an observation of the state which started with initial condition $\boldsymbol{\phi}_0$. On the other hand, the right equality of (4.4) considers a fixed observation function which is evaluated at the evolving state $\boldsymbol{\phi}(t; \boldsymbol{\phi}_0)$. That is, the second view considers evolving physical-space variables, while the first considers an output (or observation) function evolving.

In [MM16] it's observed the collection $\{\mathcal{K}_t\}$ forms a semigroup of linear operators, and the generator $\mathcal{K}$ can be computed by limiting

$$\mathcal{K} := \lim_{t\to 0^+} \frac{\mathcal{K}_t - I}{t}.$$

For our system (4.1), the action of $\mathcal{K}$ on an observable $\boldsymbol{g}$ can be computed

$$(\mathcal{K}\boldsymbol{g})(\boldsymbol{x}) = \frac{\partial}{\partial t}(\mathcal{K}_t\boldsymbol{g})(\boldsymbol{x})\big|_{t=0} = \frac{\partial}{\partial t}\boldsymbol{g}(\mathcal{F}_t(\boldsymbol{x}))\big|_{t=0} = \sum_{i=1}^{D} \frac{\partial \boldsymbol{g}}{\partial x_i}\frac{\partial \mathcal{F}_t(\boldsymbol{x})}{\partial t}\big|_{t=0} = \sum_{i=1}^{D} \frac{\partial \boldsymbol{g}}{\partial x_i} R_i(\boldsymbol{x}), \quad (4.5)$$

where R_i are the components of the flow $\boldsymbol{R}$ in (4.1). Thus, $\mathcal{K}$ is a linear operator for the evolution of $\boldsymbol{g}(\boldsymbol{\phi}(t;\boldsymbol{\phi}_0))$,

$$\frac{\partial \boldsymbol{g}}{\partial t} = \sum_{i=1}^{D} \frac{\partial \boldsymbol{g}}{\partial x_i} R_i = \mathcal{K}\boldsymbol{g}, \quad (4.6)$$

where the left equality follows from differentiating $\boldsymbol{g}$ and recalling $\boldsymbol{\phi}$ solves (4.1), and the second is from (4.5).

The Koopman operator can be studied by the behavior of its eigenfunctions. Let an eignenfuction ψ of $\mathcal{K}_t$ satisfy $(\mathcal{K}_t\psi) = e^{\lambda t}\psi$, with eigenvalue λ. For any two eigen-pairs (ψ_1, λ_1) and (ψ_2, λ_2), the following $(m\lambda_1 + n\lambda_2, \psi_1^m\psi_2^n)$ with $m, n \in \mathbb{N}$ is also an eigen-pair, and hence the spectra of $\mathcal{K}_t$ is infinite. Likewise, the Koopman operator shares these eigenfuctions: $\mathcal{K}\psi = \lambda\psi$. Consequently, $\mathcal{K}$ has infinite-dimensional eigenspace on which its action is invariant.

Thus, we see the dynamics of the observables in the Koopman framework are a reduction in the complexity of (4.1), since (4.6) is linear. However, this comes at the expense of dimension, since now $\boldsymbol{g}$ lives in an infinite-dimensional Hilbert space $\mathcal{H}$.

Going back to our example, (4.2), if, without the insight to try $g = \phi_2/\phi_1$, we define a collection of observables $g_{k,l}(\phi_1, \phi_2) = \phi_1^k \cdot \phi_2^l$, then

$$\frac{dg_{k,l}}{dt} := \mathcal{K}g_{k,l} = \frac{\partial g_{k,l}}{\partial \phi_1} R_1(\phi_1, \phi_2) + \frac{\partial g_{k,l}}{\partial \phi_2} R_2(\phi_1, \phi_2) \quad (4.7)$$

$$= k\phi_1^{k-1}\phi_2^l[\phi_2] + l\phi_2^{l-1}\phi_1^k[\phi_2^2/\phi_1 + \phi_1] \quad (4.8)$$

$$= (k+l)g_{k-1,l+1} + lg_{k+1,l-1} \quad (4.9)$$

which forms a *linear*, but infinite dimensional system of $g_{k,l}(t)$ with $k, l \in \mathbb{N}$. Trying to determine a closed set (known as a closure model) of observables for a dynamical system is unfortunately very difficult, and many approximation methods are used to close the dynamics.

4.2.2 Mori-Zwanzig Formalism

The Mori-Zwanzig (MZ) formalism [Mor65, Zwa01] is a mathematically exact approach to reduced order modeling. Unlike Koopman, MZ is able to close the dynamics with a finite set of observables through projection operators which act on the current and past states of the system. In this section we introduce the primary results of MZ, deriving the generalized Langevin equation (GLE) and general fluctuation dissipation (GFD) relationship. We also provide the discrete MZ formulation from the derivations in [DSK09, LL21, LTL21, TLA21] and we make use of this form in computational experiments. In [LTL21], the authors also connect the MZ formalism with Koopman, and provide some geometric intuition for the GLE.

Recall in the system (4.1), $\boldsymbol{\phi}$ is high-dimensional and $\boldsymbol{R}$ is non-linear. In contrast to Koopman approaches, we want to develop a low-dimensional representation of (4.1). We will study the evolution of a set of observables $\boldsymbol{g} = \boldsymbol{g}(\boldsymbol{\phi}(t; \boldsymbol{\phi}_0)) : \mathbb{R}^D \to \mathbb{R}^M$, where $\boldsymbol{\phi}(t; \boldsymbol{\phi}_0) \in \mathbb{R}^D$ solves (4.1) with initial condition $\boldsymbol{\phi}(0) = \boldsymbol{\phi}_0$. We think of $\boldsymbol{g}$ as a M-dimensional representation of $\boldsymbol{\phi}$ with $M < D$, reducing the size of the problem. As with Koopman, we will assume $\boldsymbol{g} \in \mathcal{H}$, a Hilbert space with inner product

$$\langle f, g \rangle = \int_{\mathbb{R}^D} f(\boldsymbol{x}) g(\boldsymbol{x}) d^D \mu(\boldsymbol{x}),$$

and the goal of Mori-Zwanzig is to derive the dynamics for a set of these M *resolved* components $g_1, g_2, \ldots, g_M$. Since $M < D$, there are insufficient terms to uniquely specify the full system's (4.1) state, and there are $D - M$ un-resolved components.

MZ requires a projection operator $\mathcal{P}$ which maps functions of the full state, $\boldsymbol{\phi} \in \mathbb{R}^D$, to functions of the resolved space $\mathbb{R}^M$. That is, for any $f \in L^2(\mathbb{R}^D, \mu)$, $\mathcal{P}f : \mathbb{R}^M \to \mathbb{R}$ and we think of $\mathcal{P}f \in \text{span}\{g_1, g_2, \ldots, g_M\}$. The corresponding orthogonal complement to $\mathcal{P}$ is $\mathcal{Q} := I - \mathcal{P}$, so that $\mathcal{P}\mathcal{Q} = \mathcal{Q}\mathcal{P} = 0$, since $\mathcal{P}^2 = \mathcal{P}$. MZ will use this projection to study the dynamics of $\boldsymbol{g}$.

Remark 4.2.1. *An example of the resolved space could be* $\boldsymbol{g}(\boldsymbol{\phi}) = (\phi_1, \phi_2, \ldots, \phi_M)^T$, *and* $\mathcal{P}g_i = g_i\left((\phi_1, \phi_2, \ldots, \phi_M)^T\right) = (\phi_1, \phi_2, \ldots, \phi_M)^T$.

Recall, the dynamics of $\boldsymbol{g}$ satisfy (4.6), where $\mathcal{K}$ is the Koopman operator

$$\mathcal{K} = \sum_{i=1}^{D} R_i(\boldsymbol{x})\partial_{x_i} \tag{4.10}$$

with $\boldsymbol{x}$ a dummy variable. The solution to (4.6) is $\boldsymbol{g}(\boldsymbol{\phi}(t; \boldsymbol{\phi}_0)) = e^{\mathcal{K}t}\boldsymbol{g}(\boldsymbol{\phi}_0)$, and $e^{\mathcal{K}t} = \mathcal{K}_t$ by definition, since $\mathcal{K}_t$ promotes functions of $\boldsymbol{\phi}_0$ to be evaluated after the flow-map $\mathcal{F}_t$ is applied to input $\boldsymbol{\phi}_0$, as in (4.4). Since $\mathcal{K}$ and $\mathcal{K}_t = e^{\mathcal{K}t}$ commute, we have

$$\frac{\partial}{\partial t}\boldsymbol{g} = \frac{\partial}{\partial t}e^{\mathcal{K}t}\boldsymbol{g}(\boldsymbol{\phi}_0) = \mathcal{K}\mathcal{K}_t\boldsymbol{g}(\boldsymbol{\phi}_0) = e^{\mathcal{K}t}\mathcal{K}\boldsymbol{g}(\boldsymbol{\phi}_0). \tag{4.11}$$

We would like to split the evolution into projected components and orthogonal dynamics. To do this, we consider the evolution of the operator $A(t) = e^{-t\mathcal{K}}e^{t\mathcal{Q}\mathcal{K}}$. Differentiating A leaves

$$\frac{dA(t)}{dt} = -\mathcal{K}e^{-t\mathcal{K}}e^{t\mathcal{Q}\mathcal{K}} + e^{-t\mathcal{K}}\mathcal{Q}\mathcal{K}e^{t\mathcal{Q}\mathcal{K}} = -e^{-t\mathcal{K}}\mathcal{P}\mathcal{K}e^{t\mathcal{Q}\mathcal{K}},$$

where we recall $\mathcal{P} = I - \mathcal{Q}$. Integration from 0 to t gives

$$A(t) := e^{-t\mathcal{K}}e^{t\mathcal{Q}\mathcal{K}} = I - \int_0^t e^{-s\mathcal{K}}\mathcal{P}\mathcal{K}e^{s\mathcal{Q}\mathcal{K}}ds.$$

Finally, collecting the integral with $A(t)$ and multiplying by $e^{t\mathcal{K}}$ leads to the Dyson identity,

$$e^{t\mathcal{K}} = e^{t(\mathcal{P}+\mathcal{Q})\mathcal{K}} = e^{t\mathcal{Q}\mathcal{K}} + \int_0^t e^{(t-s)\mathcal{K}}\mathcal{P}\mathcal{K}e^{s\mathcal{Q}\mathcal{K}}ds. \tag{4.12}$$

Taking (4.12) with (4.11) one obtains

$$\begin{aligned}
\frac{\partial}{\partial t} e^{\mathcal{K}t} \boldsymbol{g}(\boldsymbol{\phi}_0) &= e^{\mathcal{K}t}\mathcal{K}\boldsymbol{g}(\boldsymbol{\phi}_0) \\
&= e^{\mathcal{K}t}\mathcal{P}\mathcal{K}\boldsymbol{g}(\boldsymbol{\phi}_0) + e^{\mathcal{K}t}\mathcal{Q}\mathcal{K}\boldsymbol{g}(\boldsymbol{\phi}_0) \\
&= e^{\mathcal{K}t}\left[\mathcal{P}\mathcal{K}\boldsymbol{g}\right](\boldsymbol{\phi}_0) + \left[e^{\mathcal{Q}\mathcal{K}t}\mathcal{Q}\mathcal{K}\boldsymbol{g}\right](\boldsymbol{\phi}_0) + \int_0^t e^{(t-s)\mathcal{K}}\left[\mathcal{P}\mathcal{K}e^{s\mathcal{Q}\mathcal{K}}\mathcal{Q}\mathcal{K}\boldsymbol{g}\right](\boldsymbol{\phi}_0)ds \quad (4.13) \\
&= \left[\mathcal{P}\mathcal{K}\boldsymbol{g}\right](\boldsymbol{\phi}(t)) + \left[e^{\mathcal{Q}\mathcal{K}t}\mathcal{Q}\mathcal{K}\boldsymbol{g}\right](\boldsymbol{\phi}_0) + \int_0^t \left[\mathcal{P}\mathcal{K}e^{s\mathcal{Q}\mathcal{K}}\mathcal{Q}\mathcal{K}\boldsymbol{g}\right](\boldsymbol{\phi}(t-s))ds. \quad (4.14)
\end{aligned}$$

Considering the time-evolution of $\boldsymbol{g}(t;\boldsymbol{\phi_0}) = \boldsymbol{g}(\boldsymbol{\phi}(t;\boldsymbol{\phi}_0)$, the above can be rewritten as the generalized Langevin equation (GLE)

$$\frac{\partial}{\partial t}\boldsymbol{g}(t;\boldsymbol{\phi}_0) = \boldsymbol{M}(\boldsymbol{g}(t;\boldsymbol{\phi}_0)) - \int_0^t \boldsymbol{K}\left(\boldsymbol{g}(t-s;\boldsymbol{\phi}_0), s\right)ds + \boldsymbol{F}(\boldsymbol{\phi}_0, t), \quad (4.15)$$

with

$$\boldsymbol{M}(\boldsymbol{g}) := \mathcal{P}\mathcal{K}\boldsymbol{g} \quad (4.16a)$$

$$\boldsymbol{K}\left(\boldsymbol{g}, s\right) := -\mathcal{P}\mathcal{K}e^{s\mathcal{Q}\mathcal{K}}\mathcal{Q}\mathcal{K}\boldsymbol{g} \quad (4.16b)$$

$$\boldsymbol{F}(\boldsymbol{\phi}_0, t) := e^{\mathcal{Q}\mathcal{K}t}\mathcal{Q}\mathcal{K}\boldsymbol{g}(\boldsymbol{\phi}_0). \quad (4.16c)$$

We refer to $\boldsymbol{M}$ as the *Markov term*, since it acts on $\boldsymbol{g}$ at the current time, while $\boldsymbol{K}(\cdot, s)$ are *memory kernels* which encode the influence of observables s-units in history. The so called *orthogonal dynamics* term, $\boldsymbol{F}$, represents the contribution from the un-resolved space. Critically $\mathcal{P}\boldsymbol{F} = 0$, and hence the name *orthogonal*, since $\mathcal{P}e^{\mathcal{Q}\mathcal{K}t}\mathcal{Q}\mathcal{K} = \mathcal{P}\mathcal{Q}\mathcal{K}e^{\mathcal{Q}\mathcal{K}t} = 0$.

Collectively, (4.16) comprise the primary compents of the MZ formalism, with (4.15) providing the exact dynamics of the observable $\boldsymbol{g}$. We note that we've included a negative sign within the definition of (4.16b), known as the general fluctuation-dissipation (GFD) equation, which is consistent with the convention in [Mor65, Zwa01]. The GFD relates the kernels to the orthogonal term, since $\boldsymbol{K}(\boldsymbol{g}(\boldsymbol{\phi}), s) = -\mathcal{P}\mathcal{K}\boldsymbol{F}(\boldsymbol{\phi}, s)$.

In practice, one will apply MZ to a system which requires a numerical discrete representation

$$\boldsymbol{\phi}^{n+1} = \boldsymbol{R}_\Delta(\boldsymbol{\phi}^n), \quad \boldsymbol{\phi}^0 = \boldsymbol{\phi}_0. \tag{4.17}$$

In this case, the discrete dynamical system (4.17) has flow map $\mathcal{F}^0_\Delta = \boldsymbol{R}_\Delta$, and $\mathcal{F}^n_\Delta$ has action $\mathcal{F}^n_\Delta(\boldsymbol{\phi}^0) := \boldsymbol{\phi}^n$. Here, Δ is the uniform stepsize for (4.17), and solutions $\boldsymbol{\phi}^n = \boldsymbol{\phi}(n\Delta; \boldsymbol{\phi}_0)$.

Discrete $\boldsymbol{g}^n$ provide observations of the states $\boldsymbol{\phi}^n$ (i.e. $\boldsymbol{g}^n = \boldsymbol{g}(\boldsymbol{\phi}^n)$), and the associated discrete-time Koopman operator $\mathcal{K}^n_\Delta$ (with generator $\mathcal{K}_\Delta$) is built off the discrete-time flow map giving the familiar

$$\mathcal{K}^n_\Delta \boldsymbol{g}^0 := \boldsymbol{g}^0 \circ \mathcal{F}^n_\Delta(\boldsymbol{\phi}^n) = \boldsymbol{g}^n.$$

To obtain a discrete MZ formulation, we make use of the discrete Dyson identity [LL21, LTL21, TLA21]

$$\mathcal{K}^{n+1}_\Delta = (\mathcal{Q}\mathcal{K}_\Delta)^{n+1} + \sum_{l=0}^{n} \mathcal{K}^{n-l}_\Delta \mathcal{P}\mathcal{K}_\Delta (\mathcal{Q}\mathcal{K}_\Delta)^l. \tag{4.18}$$

Applying (4.18) to the evolution of $\boldsymbol{g}^n$ gives

$$\begin{aligned}
\boldsymbol{g}^{n+1} = \mathcal{K}^{n+1}_\Delta \boldsymbol{g}^0 &= (\mathcal{Q}\mathcal{K}_\Delta)^{n+1}\boldsymbol{g}^0 + \sum_{l=0}^{n} \mathcal{K}^{n-l}_\Delta \mathcal{P}\mathcal{K}_\Delta (\mathcal{Q}\mathcal{K}_\Delta)^l \boldsymbol{g}^0 \\
&= \mathcal{K}^n_\Delta[\mathcal{P}\mathcal{K}_\Delta \boldsymbol{g}](\boldsymbol{\phi}^0) + \sum_{l=1}^{n} \mathcal{K}^{n-1}_\Delta \left[\mathcal{P}\mathcal{K}_\delta (\mathcal{Q}\mathcal{K}_\Delta)^l \boldsymbol{g}\right](\boldsymbol{\phi}^0) + \left[(\mathcal{Q}\mathcal{K}_\Delta)^{n+1}\boldsymbol{g}\right](\boldsymbol{\phi}^0) \\
&= [\mathcal{P}\mathcal{K}_\Delta \boldsymbol{g}](\boldsymbol{\phi}^n) + \sum_{l=1}^{n} \left[\mathcal{P}\mathcal{K}_\delta (\mathcal{Q}\mathcal{K}_\Delta)^l \boldsymbol{g}\right](\boldsymbol{\phi}^{n-l}) + \left[(\mathcal{Q}\mathcal{K}_\Delta)^{n+1}\boldsymbol{g}\right](\boldsymbol{\phi}^0) \\
&= \boldsymbol{\Omega}^{(0)}(\boldsymbol{g}^n) + \sum_{l=1}^{n} \boldsymbol{\Omega}^{(l)}(\boldsymbol{g}^{n-l}) + \boldsymbol{W}_n(\boldsymbol{\phi}^0).
\end{aligned} \tag{4.19}$$

The discrete analogue of the Markov, memory kernel and orthogonal dynamics terms in (4.16) are given by the kernels $\boldsymbol{\Omega}^{(0)}$, $\boldsymbol{\Omega}^{(l)}$ $l \geq 1$, and $\boldsymbol{W}$ in (4.19), and we define these terms in (4.20)

$$\boldsymbol{\Omega}^{(0)} := \mathcal{P}\mathcal{K}_\Delta \tag{4.20a}$$

$$\boldsymbol{\Omega}^{(l)} := \mathcal{P}\mathcal{K}_\Delta(\mathcal{Q}\mathcal{K}_\Delta)^l,\ l = 1, 2, \dots \tag{4.20b}$$

$$\boldsymbol{W}_n := (\mathcal{Q}\mathcal{K}_\Delta)^{n+1}\boldsymbol{g}. \tag{4.20c}$$

The discrete GFD relationship

$$\boldsymbol{\Omega}^{(l)}(\boldsymbol{g}) = \mathcal{P}\mathcal{K}_\Delta\boldsymbol{W}_{l-1},\ \forall l \geq 1 \tag{4.21}$$

follows from the definitions (4.20b) and (4.20c), and this relationship will be critical for developing a data-driven approach which utilizes the MZ framework. Importantly, (4.21) imposes a connection between the memory kernels $\boldsymbol{\Omega}^{(l)}$ and orthogonal dynamics terms, $\boldsymbol{W}_{l-1}$. Recall that $\boldsymbol{W}_{l-1}$ is a function of the full initial state $\boldsymbol{\phi}_0$. So, evaluating left and right hand sides of (4.21) at $\boldsymbol{\phi}_0$ gives

$$\begin{aligned}
\boldsymbol{\Omega}^{(l)}(\boldsymbol{g}(\boldsymbol{\phi}_0)) = \mathcal{P}\mathcal{K}_\Delta\boldsymbol{W}_{l-1}(\boldsymbol{\phi}_0) &= \mathcal{P}\left[\mathcal{K}_\Delta(\mathcal{Q}\mathcal{K}_\Delta)^l\right](\boldsymbol{g}(\boldsymbol{\phi}_0))\\
&= \mathcal{P}\left[(\mathcal{Q}\mathcal{K}_\Delta)^l\right](\boldsymbol{g}\circ\mathcal{F}_\Delta)(\boldsymbol{\phi}_0)\\
&= \mathcal{P}\left[(\mathcal{Q}\mathcal{K}_\Delta)^l\right](\boldsymbol{g}(\boldsymbol{\phi}_1))\\
&= \mathcal{P}\boldsymbol{W}_{l-1}(\boldsymbol{\phi}_1),\ \forall l \geq 1.
\end{aligned} \tag{4.22}$$

Notice that the evaluated form (4.22) differs from (4.21), which defines a function $\boldsymbol{\Omega}^{(l)}(\boldsymbol{g}(\cdot))$ depending on a projection $\mathcal{P}$, discrete Koopman operator $\mathcal{K}_\Delta$ and observable function $\boldsymbol{g}$. Instead, (4.22) provides an evaluation of this function which does not require access to the Koopman operator. Given sufficient data-points, one can leverage (4.22) to learn the memory kernels' action.

4.3 Observable Discovery: A Data Driven Approach

In this section, we outline the setup for using the Mori-Zwanzig formalism to predict turbulent fluid flow dynamics. A critical component of this process will be the choice of the observable functions, $\boldsymbol{g}$, which the discrete GLE will evolve forward in time. We propose a novel approach towards learning these observables from data which improves trajectories based on a MZ framework.

4.3.1 Mori-Zwanzig Projection

In [TLA21], the authors demonstrate a data-driven approach based on the algorithm described in [LTL21] for learning the kernels $\boldsymbol{\Omega}^{(l)}$ in (4.19). The form of the kernels depends on the choice of projection operator, $\mathcal{P}$. Several projections can be considered. Mori [Mor65] proposed a linear $\mathcal{P}$ defined on a given set of linearly independent resolved observables $\boldsymbol{g} = \{g_i\}_{i=1}^{M}$, which we emphasize are fixed for fixed $\mathcal{P}$. For example, these could be the component maps $g_i(\boldsymbol{\phi}) = \phi_i$. Completing the space are the $D - M$ functions $\tilde{\boldsymbol{g}} = \{\tilde{g}_j\}_{j=1}^{D-M}$, so that $\mathrm{span}\{\boldsymbol{g}, \tilde{\boldsymbol{g}}\} = \mathcal{H}$. The projection takes functions $f = f(\boldsymbol{g}, \tilde{\boldsymbol{g}})$ to $\mathcal{P}f = [\mathcal{P}f](\boldsymbol{g}) \in \mathrm{span}\{g_1, g_2, \dots, g_M\}$, defined by

$$[\mathcal{P}f](\boldsymbol{g}) := \sum_{i,j=1}^{M} \frac{\langle g_i, f\rangle}{\langle g_i, g_j\rangle} g_j. \tag{4.23}$$

Here, we've used scalar f, but the projection extends to vector $\boldsymbol{f}$ in a component-wise fashion. The linearity of Mori's projection produces linear Markov and memory kernel terms in the GLE (4.15) and (4.19), a property that has lead to Mori's projection being widely used [TLA21, MW20, MO07, LTL21]. Because of this linearity, one can explicitly determine the memory kernels using time series of the fully resolved dynamics, and experiments in [LTL21] demonstrate MZ with Mori's projection produce higher-order corrections compared with data driven Koopman approaches. However, the linearity of Mori's projection limits the power of

MZ, and cannot be combined with many non-linear closure methods nor accommodate more expressive non-linear memory kernels based on neural network approaches used in modern machine learning frameworks.

Zwanzig [Zwa73, Zwa01] also proposed a projection based on conditional expectation $[\mathcal{P}f](\boldsymbol{h}) = \mathbb{E}_\mu\left[f(\boldsymbol{g}(\boldsymbol{\phi}), \tilde{\boldsymbol{g}}(\boldsymbol{\phi}))|\boldsymbol{g}(\boldsymbol{\phi}) = \boldsymbol{h}\right]$ which in general can be non-linear and result in non-linear Markov and memory kernel terms in the GLE. This projection is shown to be optimal [CHK02], but in practice, using the Zwanzig projection on anything but simple dyamical systems is intractable. Other projections have been proposed based on truncation [PD17] and Wiener projections [LL21].

Recently, Lin et al. [LTP23] introduced a regression-based projection which attempts to fill the gap between the simplistic, linear Mori projection and the unmanageable but optimal Zwanzig approach. This projection regresses functions f onto the span of a basis consisting of (potentially non-linear) combinations of the resolved observables $\boldsymbol{g}$.

4.3.2 The Problem of Observable Choice

Tian et al. [TLA21] chose Mori's linear projection to develop a data-driven approach for turbulent fluid simulation based on an MZ framework. Besides the limitation of linearity, the process is strongly influenced by the pre-chosen observable functions onto which the dynamics are projected. For isotropic turbulence, [TLA21] consider four sets of observables:

- Observable set 1: $\bar{v}_i$ (filtered velocity)
- Observable set 2: $\bar{\boldsymbol{v}}$, $\bar{v}_i\bar{v}_j$, $\bar{v_i v_j} - \bar{v}_i\bar{v}_j$
- Observable set 3: $\bar{v}_i$, $\sum_i \bar{v}_i\bar{v}_i$, S_{ij}, W_{ij}, $S_{ij}S_{ij}$, $W_{ij}W_{ij}$, where $S_{ij} = \frac{1}{2}\left(\frac{\partial\bar{v}_i}{\partial x_j} + \frac{\bar{v}_j}{\partial x_i}\right)$, $W_{ij} = \frac{1}{2}\left(\frac{\partial\bar{v}_i}{\partial x_j} - \frac{\bar{v}_j}{\partial x_i}\right)$.

- Observable set 4: $\bar{v}_i$, $\frac{\partial \bar{v_i v_j}}{\partial x_i}$, $\frac{\partial \bar{v_i v_j}}{\partial x_j}$, $\frac{\partial \bar{p}}{\partial x_i}$.

Each set is taken over i, j modulo linearly dependent terms, and the constant observable $g_0 = 1$ is added to each observable set. Using these, the authors found little difference between sets 1–3, where adding strain rate tensor, vorticity, kinetic energy, subgrid stress terms etc. provided neglibible improvement over the filtered velocity components alone. However, with observable set 4 which adds the actual filtered governing equation terms (including pressure), there was over a 50% improvement over the other observable choices.

Choosing the appropriate observables is a shared problem with the learning framework used in Koopman approximation, for which Deep Koopman approaches have been tried as described in the introduction. Additionally, this problem extends equally to other projections, such as the regression-based approach.

The discussion henceforth will be on setting up a data-driven framework for determining a good set of observables to pair with the MZ framework.

4.3.3 The Forward Problem: Learning Memory Kernels

We start by setting up the forward problem generating the dynamical system (4.1) via a MZ framework. Suppose we have T trajectories of our full state $\{\boldsymbol{\phi}^{[i]}\}_{i=1}^{T} \subset \mathbb{R}^D$, each starting from initial condition $\boldsymbol{\phi}^{[i]}(0) = \boldsymbol{\phi}_0^{[i]} \sim \mu$ and sampled at time points $t_n = n\Delta$. Suppose further that we have defined some observable function $\boldsymbol{g}$, giving resolved data $\boldsymbol{g}^{[i]}(n) = \boldsymbol{g}(\boldsymbol{\phi}^{[i]}(t_n))$. At each time t_n, we combine $\boldsymbol{g}^{[i]}(n) \in \mathbb{R}^M$ for all $i = 1, \ldots, T$ realizations into an $T \times M$ vector $\boldsymbol{g}^n$ of data at time t_n, and likewise refer to $\boldsymbol{\phi}(t_n) = \boldsymbol{\phi}^n$ as the ensemble of all state trajectories $\boldsymbol{\phi}^{[i]}$ at time t_n.

We are interested in evolving forward these observables $\boldsymbol{g}^n$. However, we may only wish to track a few quantities of interest within $\boldsymbol{g}$ corresponding to (reduced order) physical-space variables. To emphasize this, we define $\boldsymbol{G}^{[i]}$ to be a collection of coarse grain quantities of interest (vorticity, velocity etc.) from the filtered state. As such, $\boldsymbol{G}^{[i]}$ is a subset of components of $\boldsymbol{g}^{[i]}$. We call $\boldsymbol{G}^{[i]} \in R^{\tilde{M}}$ a *generating observable* because it will seed the formation of a full observable set $\boldsymbol{g}$, as described in Section 4.3.4. As with $\boldsymbol{g}^n$, we consider the ensemble of realizations at time t_n for the generating observable $\boldsymbol{G}^n$.

Given the observable function, $\boldsymbol{g}(\cdot)$, we specify a regression-based projection $\mathcal{P}$ (inspired by [LTP23]) which will in turn specify Markov and memory kernels according to the GLE and (4.20). First, we choose a family of functions, $\boldsymbol{f}(\cdot;\theta)$, parameterized by θ with which to perform the regression. To define a regression problem for pairs $\{\boldsymbol{g}(\boldsymbol{\phi}^{[i]}(0)), \boldsymbol{y}^{[i]}\}$ of regressors $\boldsymbol{g}(\boldsymbol{\phi}^{[i]}(0))$ at t_n, and targets, $\boldsymbol{y}^{[i]}$, one solves

$$\theta_* = \arg\min_{\theta} C\left(\boldsymbol{f}(\boldsymbol{g}(\boldsymbol{\phi}^0);\theta), \boldsymbol{y}\right), \tag{4.24}$$

where $\boldsymbol{y} = \{\boldsymbol{y}^{[i]}\}$ and $C(\cdot,\cdot)$ is some cost function. Typically, we take $\boldsymbol{f}(\cdot;\theta)$ to be some family of simple functions (linear, polynomial of degree d etc.), but can also be a fixed neural network architecture, and C to be mean-squared-error. For a function $\boldsymbol{h}(\boldsymbol{\phi})$ giving targets $\boldsymbol{y}^{[i]} = \boldsymbol{h}(\boldsymbol{\phi}^{[i]}(t_k))$, the projection

$$[\mathcal{P}\boldsymbol{h}](\cdot) = \boldsymbol{f}(\cdot\ ;\theta_*),$$

where θ_* solves (4.24) with $\boldsymbol{y} = \boldsymbol{h}(\boldsymbol{\phi}^k)$.

Remark 4.3.1. *The operator $\mathcal{P}$ as defined is indeed a projection (i.e. $\mathcal{P}^2 = \mathcal{P}$), since $\mathcal{P}^2\boldsymbol{h} = [\mathcal{P}\boldsymbol{f}](\cdot;\theta_*) = \boldsymbol{f}(\cdot\ ;\theta_*)$. This follows because repeating the regression*

$$\arg\min_{\theta} C\left(\boldsymbol{f}(\boldsymbol{g}(\boldsymbol{\phi}^0);\theta), \boldsymbol{f}(\boldsymbol{g}(\boldsymbol{\phi}^0);\theta_*)\right)$$

returns the same θ_*.

With the projection specified, we can proceed with the MZ learning process. Our goal is to evolve the quantities of interest, $\boldsymbol{G}^n$, forward in time using the MZ formalism and observable set $\boldsymbol{g}^k$, $k \leq n$, so that it matches the data $\boldsymbol{G}^{n+1}$ at the next timestep. Starting with $n = 0$, looking to (4.19) we have

$$\boldsymbol{G}^1(\boldsymbol{\phi}^0) = \boldsymbol{\Omega}^{(0)}(\boldsymbol{g}^0(\boldsymbol{\phi}^0)) + \boldsymbol{W}_0(\boldsymbol{\phi}^0).$$

By definition, $\boldsymbol{\Omega}^{(0)}(\boldsymbol{g}^0) = \mathcal{P}\mathcal{K}_\Delta \boldsymbol{g}^0 = \mathcal{P}\boldsymbol{g}^1$, so the Markov kernel $\boldsymbol{\Omega}^{(0)}$ encodes the contribution of $\boldsymbol{g}^0$ towards $\boldsymbol{G}^1$ within the expressivity constraints of the regression family $\boldsymbol{f}(\cdot;\theta)$. With the Markov term evaluated, the initial orthogonal dynamics term is the residual

$$\boldsymbol{W}_0(\boldsymbol{\phi}_0) = \boldsymbol{G}^1(\boldsymbol{\phi}^0) - \mathcal{P}\boldsymbol{g}^1 = \boldsymbol{G}^1(\boldsymbol{\phi}^0) - \boldsymbol{f}(\boldsymbol{g}^1(\boldsymbol{\phi}^0);\theta_*^{(0)}) = \boldsymbol{G}^1(\boldsymbol{\phi}^0) - \boldsymbol{\Omega}^{(0)}(\boldsymbol{g}^0(\boldsymbol{\phi}^0)).$$

For $n = 1$,

$$\boldsymbol{G}^2 = \boldsymbol{\Omega}^{(0)}(\boldsymbol{g}^1) + \boldsymbol{\Omega}^{(1)}(\boldsymbol{g}^0) + W_1(\boldsymbol{\phi}^0),$$

and $\boldsymbol{\Omega}^{(1)}$ and $\boldsymbol{W}_1$ are, as of yet, undetermined. To determine the successive memory kernel, we turn to the relationship (4.22),

$$\boldsymbol{\Omega}^{(1)}(\boldsymbol{g}^0) = \mathcal{P}\boldsymbol{W}_0(\boldsymbol{\phi}^1) = \mathcal{P}\left(\boldsymbol{G}^1(\boldsymbol{\phi}^1) - \boldsymbol{\Omega}^{(0)}(\boldsymbol{g}^0(\boldsymbol{\phi}^1))\right) = \mathcal{P}\left(\boldsymbol{G}^2(\boldsymbol{\phi}^0) - \boldsymbol{\Omega}^{(0)}(\boldsymbol{g}^1(\boldsymbol{\phi}^0))\right).$$

Solving the regression problem for the first memory kernel gives $\boldsymbol{f}(\cdot\ ;\theta_*^{(1)})$, and the next orthogonal dynamics term $\boldsymbol{W}_1(\boldsymbol{\phi}^0)$ is defined as the ensuing residual. It is worth emphasizing that the memory kernels re-project the residual when the currently learned forward model (i.e. $\boldsymbol{G}^1(\cdot) = \boldsymbol{\Omega}^{(0)}(\boldsymbol{g}^0(\cdot))$) is applied to the next time-step, $\boldsymbol{\phi}^1$.

The process to determine memory kernels and orthogonal terms continues inductively, with a general update procedure at step n of

$$\boldsymbol{\Omega}^{(n+1)}(\boldsymbol{g}^0) = \mathcal{P}\boldsymbol{W}_n(\boldsymbol{\phi}^1) = \mathcal{P}\left[\boldsymbol{G}^{n+2}(\boldsymbol{\phi}^0) - \sum_{l=0}^{n} \boldsymbol{\Omega}^{(l)}(\boldsymbol{g}^{n-l+1}(\boldsymbol{\phi}^0))\right] \tag{4.25}$$

$$\boldsymbol{W}_{n+1}(\boldsymbol{\phi}^0) = \boldsymbol{G}^{n+2}(\boldsymbol{\phi}^0) - \sum_{l=0}^{n+1} \boldsymbol{\Omega}^{(l)}\left(\boldsymbol{g}^{n-l+1}(\boldsymbol{\phi}^0)\right), \tag{4.26}$$

given one has already computed $\boldsymbol{\Omega}^{(l)}$, $l = 1, 2, \ldots, n$ and $\boldsymbol{W}_n(\cdot)$. By construction, this process enforces the GFD relationship $\boldsymbol{\Omega}^{(n)}(\boldsymbol{g}^0) = \mathcal{P}\mathcal{K}_\Delta \boldsymbol{W}_{n-1}$ and satisfies the orthogonality condition $\mathcal{P}\boldsymbol{W}_n = 0$.

We call this Mori-Zwanzig forward operation $MZ(\ \cdot\ ) = MZ(\cdot\ ; \boldsymbol{\Omega}^{(0)}, \boldsymbol{\Omega}^{(1)}, \ldots, \boldsymbol{\Omega}^{(K)})$, where the Markov and K memory kernels are learned from some data. $MZ(\ \cdot\ )$ relies on a specified projection and a choice of observables. In practice, we will truncate the number of kernels used so it does not stretch the length of the training trajectory. The process (4.25),(4.26) for learning the Markov and memory kernels is captured in Algorithm 11.

4.3.4 The Inverse Problem: Learning Observables

With a well defined forward process $MZ(\ \cdot\ )$, we can now address the problem of observable choice. Predicted trajectories using $MZ(\ \cdot\ )$ accumulate error from the truncation of the orthogonal term. The size of error is dependent on two factors:

- The set of observables chosen to study the dynamics on. This choice of observables is akin to selecting a subgrid closure model. While some intuition and physics can help guide observable selection, in general it is very difficult to specify a *good* set in a prescribed way. Providing exact terms to close the governing equations of the dynamical system, as was done for the fourth observable set in experiments by [TLA21], can produce good results, but in practice one cannot expect to have access to the full state needed to form these non-linear closure terms.

- The expressivity of the projection operator (and consequently the Markov and memory kernels) can influence the error, since a highly flexible projection such as a high order regression or a neural network can capture more non-linear dynamics. This can compensate for a poor observable set choice. Since our goal is observable discovery, we choose simple projection based on linear regression in order to emphasize the benefit a better observable set has on predictions.

As mentioned above, we start with a set of quantities of interest, $\boldsymbol{G}$, which we call the *generating observables*, and we wish to track these quantities as they evolve in time. $\boldsymbol{G}$ lives in a reduced/resolved space, $\mathbb{R}^{\tilde{M}}$. To create the full observable set, $\boldsymbol{g}$, we use a neural network function $NN(\cdot\,; \Theta)$ with learnable parameters Θ to produce the remainder of the observable terms from $\boldsymbol{G}$. That is, $\boldsymbol{g}^{[i]} = (\boldsymbol{G}^{[i]}, NN(\boldsymbol{G}^{[i]}; \Theta)) \in \mathbb{R}^M$ for each training trajectory $i = 1, \ldots, T$.

In the MZ theory, observables may depend directly on the full state. However, in practice we restrict the input to $\boldsymbol{G}$ to a filtered version of the full state, $\bar{\boldsymbol{\phi}}^{[i]}$. Under this construction, the learned feature set $\boldsymbol{g}^n = (\boldsymbol{G}^n, NN(\boldsymbol{G}^n))$ is never aware of the full resolution data, $\boldsymbol{\phi}(t_n)$. This prevents the network from learning a direct simulation of the full resolution data in tandem with a down-sampling operation to the resolved dimension.

Armed with a set of time-series data $\{\boldsymbol{G}^n\}_{n=1}^N$ and a trainable set of observables $\boldsymbol{g}^n = (\boldsymbol{G}^n, NN(\boldsymbol{G}^n; \Theta))$, we propose an alternating scheme to learn both $\boldsymbol{g}^n$ and the Markov and memory kernels $\boldsymbol{\Omega}^{(l)},\ l = 0, 1, \ldots, K$. First, we generate $\boldsymbol{g^n}$ with Θ randomly initialized. Given these observables, we fix Θ and proceed with defining the memory kernels via the projection (4.25). With $MZ(\cdot\,; \boldsymbol{\Omega}^{(0)}, \boldsymbol{\Omega}^{(1)}, \ldots, \boldsymbol{\Omega}^{(K)})$ learned, we freeze the kernels and use gradient descent to update the network weights Θ. Since Θ is learned through descent, several steps should be taken before returning to update the kernels, and we find in practice that slowly increasing this number as the learning rate decreases is beneficial. Proceeding in this iterative fashion, alternating between training the network and regressing to obtain the kernels, allows both appropriate observables along with the MZ Markov and memory kernels to be determined simultaneously. This process is encapsulated in Algorithm 11 and visualized in Figure 4.1.

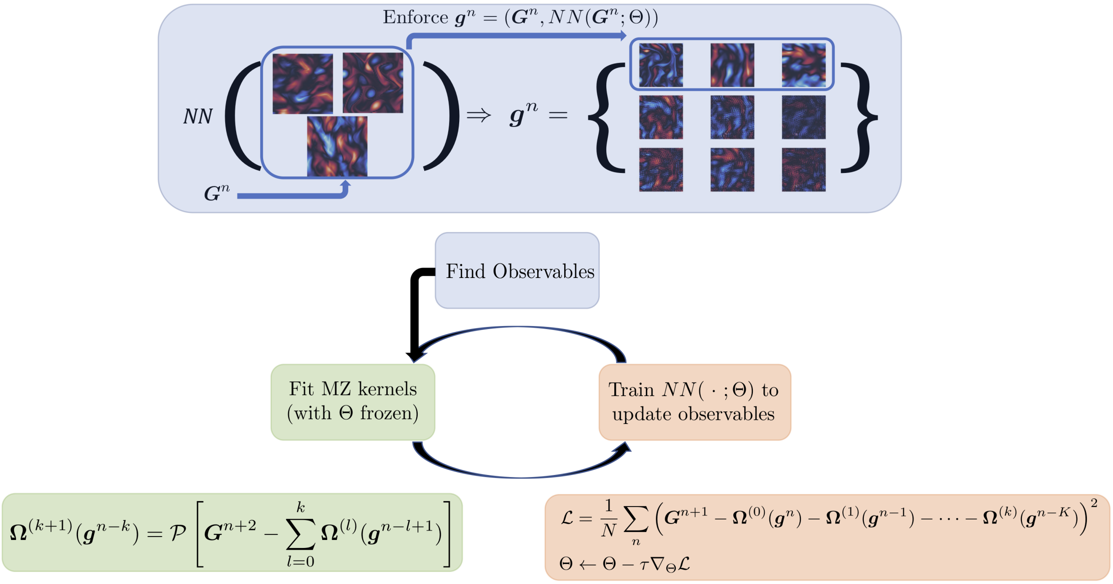

Enforce $\boldsymbol{g}^n = (\boldsymbol{G}^n, NN(\boldsymbol{G}^n;\Theta))$
NN
$\boldsymbol{G}^n$
$\Rightarrow \boldsymbol{g}^n =$
Find Observables
Fit MZ kernels (with Θ frozen)
Train $NN(\,\cdot\,;\Theta)$ to update observables
$\boldsymbol{\Omega}^{(k+1)}(\boldsymbol{g}^{n-k}) = \mathcal{P}\left[\boldsymbol{G}^{n+2} - \sum_{l=0}^{k}\boldsymbol{\Omega}^{(l)}(\boldsymbol{g}^{n-l+1})\right]$
$\mathcal{L} = \frac{1}{N}\sum_n \left(\boldsymbol{G}^{n+1} - \boldsymbol{\Omega}^{(0)}(\boldsymbol{g}^n) - \boldsymbol{\Omega}^{(1)}(\boldsymbol{g}^{n-1}) - \cdots - \boldsymbol{\Omega}^{(k)}(\boldsymbol{g}^{n-K})\right)^2$
$\Theta \leftarrow \Theta - \tau\nabla_\Theta\mathcal{L}$

Figure 4.1: The MZ kernel learning process with observable discovery. Observables are specified from a generating set, $\boldsymbol{G}$, as shown in the blue block. Then, an alternating learning process proceeds, where MZ kernels $\boldsymbol{\Omega}^{(l)}$, $l = 0, 1, \ldots, K$ are learned keeping Θ fixed (green block) followed by updating the observables via gradient descent on the parameters Θ, keeping $\boldsymbol{\Omega}^{(l)}$ fixed (orange block).

One may ask why construct $\boldsymbol{g}$ as specified, and not let the neural network determine $\boldsymbol{g}$ entirely? There are several subtle reasons for constructing the observables in this fashion, where $\boldsymbol{G}$ is explicitly included as part of $\boldsymbol{g}$. Primarily, if the observables are entirely determined by $NN(\cdot)$, e.g. $\boldsymbol{g}^n = NN(\bar{\boldsymbol{\phi}}^n)$, then the alternating minimization process could easily learn the zero (or constant) function for $NN(\cdot)$. That is, when $\boldsymbol{g}^n = NN(\bar{\boldsymbol{\phi}}^n) = 0$ for each n, then the kernels learned via Algorithm 11 would be essentially unconstrained, since

$$0 = NN(\bar{\boldsymbol{\phi}}^{n+1}) = MZ(0; \boldsymbol{\Omega}^{(0)}, \boldsymbol{\Omega}^{(1)}, \ldots, \boldsymbol{\Omega}^{(K)}),$$

automatically satisfied for any kernels $\boldsymbol{\Omega}^{(l)}$ vanishing at zero. So, we avoid this degeneracy by specifying a portion of $\boldsymbol{g}$. Moreover, beyond avoiding trivial learning, this specification aids training since the early kernel-learning steps (4.25) will be able to at minimum project onto the components, $\boldsymbol{G}$, and advance the system without subgrid corrections. Therefore, the system is in a reasonable regime where $MZ(\,\cdot\,)$ is able to produce ballpark predictions of the next timestep at initialization. Then, when the kernels are frozen and the network $NN(\cdot; \Theta)$ is trained, it can fine-tune the observables $\boldsymbol{g}$ to address the subgrid error. Finally, specifying $\boldsymbol{G}$ allows control over the the quantities one wishes to track. For example, a system may have a constant or steadily decaying spatial average in time, and so $\boldsymbol{g}^n = NN(\bar{\boldsymbol{\phi}}^n)$ might quickly learn to return the average value of $\bar{\boldsymbol{\phi}}$. However, if a user intends to study a direct reduced-order representation of the state, they can specify so within $\boldsymbol{G}$ and the observable-learning network must adapt to find features which will allow tracking this quantity.

Algorithm 11 MZ with Observable Discovery

INPUT: Sequence of generating observables $\{\boldsymbol{G}^n\}_{n=1}^N$. Number of memory kernels: K, and family of regression functions $\boldsymbol{f}(\cdot;\theta)$. Assumes the observable generating function $NN(\cdot\,;\Theta)$ is a trainable neural network with parameters Θ and loss function $\mathcal{L}$.

OUTPUT: Markov and memory kernels $\boldsymbol{\Omega}^{(l)} = \boldsymbol{f}(\ \cdot\ ;\theta^{(l)})$, $l = 0, 1, \ldots, K$ and trained $NN(\ \cdot\ ;\Theta)$

```
for $i = 1 : N_{alternate}$ do
    Set $\boldsymbol{g}^n = (\boldsymbol{G}^n, NN(\boldsymbol{G}^n;\Theta))$, $n = 1, \ldots, N$                ▷ with $\Theta$ frozen
    $\boldsymbol{\Omega}^{(0)} = \boldsymbol{f}(\cdot\ ;\theta^{(0)})$, where $\theta^{(0)} = \arg\min_\theta \frac{1}{N-1}\sum_{n=1}^{N-1} C(\boldsymbol{f}(\boldsymbol{g}^n;\theta), \boldsymbol{G}^{n+1})$
    for $k = 0 : K-1$ do                                                      ▷ with $\Theta$ frozen
        $\boldsymbol{\Omega}^{(k+1)} = \boldsymbol{f}(\ \cdot\ ;\theta^{(k+1)})$, where
            $\theta^{(k+1)} \leftarrow \arg\min_\theta \frac{1}{N-k-2} \sum_{n=1}^{N-k-2} C\left(\boldsymbol{f}(\boldsymbol{g}^{n-k};\theta), \boldsymbol{G}^{n+k+2} - \sum_{l=0}^{k} \boldsymbol{\Omega}^{(l)}(\boldsymbol{g}^{n-l+1})\right)$
    end for
    for $j = 0 : N_{train}$ do                                 ▷ with $\theta^{(l)},\ l = 0, \ldots, K$ fixed
        $\Theta \leftarrow \Theta - \tau\nabla_\Theta \mathcal{L}(\{\boldsymbol{G}^n\}, \{\boldsymbol{g}^n\})$, where $\boldsymbol{g}^n \leftarrow (\boldsymbol{G}^n, NN(\boldsymbol{G}^n;\Theta))$ and
            $\mathcal{L} = \frac{1}{N-K} \sum_{n=K+1}^{N-1} \left(\boldsymbol{G}^{n+1} - \sum_{l=0}^{K} \boldsymbol{\Omega}^{(l)}\left(\boldsymbol{g}^{n-l}\right)\right)^2$
    end for
end for
return $\boldsymbol{\Omega}^{(l)}$, $l = 0, 1, \ldots, K$ and $NN(\ \cdot\ ;\Theta)$.
```

4.4 Results

For experiments, we test the observable learning process for MZ-based simulation using direct numerical simulation (DNS) of a 2D vorticity field. The ground truth data is a pseudo-spectral forced Navier-Stokes simulation of isotropic turbulence, which evolves the vorticity, ω, according to

$$\frac{\partial \omega}{\partial t} + (\boldsymbol{u} \cdot \nabla)\omega = \nu \nabla^2 \omega + \operatorname{curl}(\boldsymbol{F}), \tag{4.27}$$

where viscosity $\nu = 10^{-3}$ and $\boldsymbol{F}$ is a forcing function. The psuedo-spectral approach has built in filtering and anti-aliasing to maintain stability. For details, see [KSA21][2]. In two dimensions, the relationship between velocity, $\boldsymbol{u} = (u, v)^T$, and vorticity is

$$\omega := \operatorname{curl}(\boldsymbol{u}) = \frac{\partial v}{\partial x} - \frac{\partial u}{\partial y}.$$

We take periodic boundary conditions, and the simulation is performed on a 265^2 grid of a $[0, 2\pi]^2$ domain. Snapshots of the flow are recorded after reaching stationarity for $T = 7$ differently initialized trajectories, forming the data $\boldsymbol{\phi}^n = \{(\omega^{[i]}(t_n), u^{[i]}(t_n), v^{[i]}(t_n))^T\}_{i=1}^T$ at each timestep $t_n = n \cdot \Delta$, with $\Delta = 0.0015$s.

Given this data, we apply spatial filtering at each timestep giving $\bar{\boldsymbol{\phi}}^n$. The generating observable set $\boldsymbol{G}^n = \boldsymbol{G}(\bar{\boldsymbol{\phi}}^n)$ is a 64^2 grid coarse-grain observation of the quantities of interest from our dataset, in this case velocity and vorticity components. These hand-selected quantities are likely to be insufficient for learning a successful MZ model, so we extend to a full observable set $\boldsymbol{g}^n = (\boldsymbol{G}^n, NN(\boldsymbol{G}^n; \Theta))$. For our experiments, we use a deep convolutional neural network with fifteen 7 × 7 filters in each layer, 5 hidden layers, and an output layer where we can specify the number of additional observables to add to the generating set, $\boldsymbol{G}^n$.

[2] The codebase for generating the ground truth data comes from `https://github.com/google/jax-cfd`, based on the paper [KSA21].

We will evaluate our observable-discovery based predictions against a more comprehensive, but predefined set of observables, $\boldsymbol{g}_{PO}$, where the subscript stands for *predefined observables.* These consist of the coarse-grain vorticity and velocity components along with their spatial gradients. We'll refer to predictions made using these observables as by MZ_PO. For the MZ_PO predictions, the MZ kernels are defined by a quadratic regression-based projection, so nonlinear combinations of components of $\boldsymbol{g}_{PO}$ are used when predictions $\boldsymbol{g}_{PO}^{n+1}$ are made from $\{\boldsymbol{g}_{PO}^{n}, \boldsymbol{g}_{PO}^{n-1}, \ldots, \boldsymbol{g}_{PO}^{n-K}\}$ via $MZ(\ \cdot\ )$. For predictions made using the *learned observables* (henceforth referred to by MZ_LO), we will use the subscript $\boldsymbol{G}_{LO}^{n+1}$, and when simulating the vorticity and velocity components from the computational fluid dynamics program at coarse-grain directly, we denote the components $\boldsymbol{G}_{CFD}^{n+1}$.

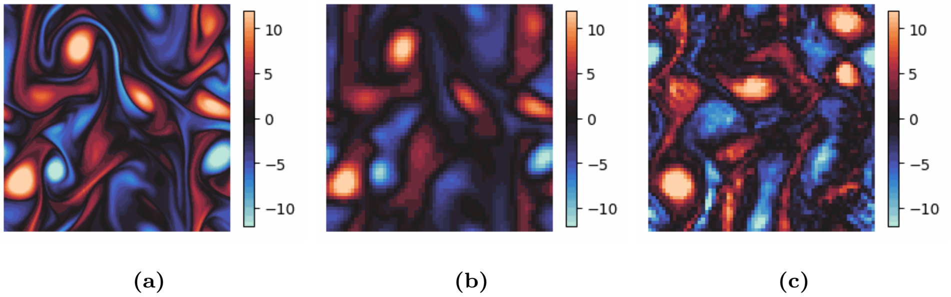
We remark that turbulence is an ideal dataset to test reduced order modeling on. The celebrated description of turbulence due to Richardson [Ric22]

> *Big whorls have little whorls, which feed on their velocity, and little whorls have lesser whorls, and so on to viscosity.*

highlights one of the chief reasons. Turbulent flows contain a broad range of scales (whorls), which exist from huge eddies down to dissipation at viscosity, and resolving all these dynamics would require a prohibitively large number of degrees of freedom (on the order of Avogadro's number). Consequently, for many applications it is necessary to run simulations at a lower (resolved) resolution. However, applying a simple linear spatial filtering operation, $\bar{\cdot}$, to (4.27) (without downsampling) gives the filtered vorticity transport equation

$$\frac{\partial \bar{\omega}}{\partial t} + (\bar{\boldsymbol{u}} \cdot \nabla)\bar{\omega} = \nu \nabla^2 \bar{\omega} + \overline{\operatorname{curl}(\boldsymbol{F})} + \tau, \tag{4.28}$$

with $\tau = (\bar{\boldsymbol{u}} \cdot \nabla)\bar{\omega} - \overline{(\boldsymbol{u} \cdot \nabla)\omega}$ the subgrid term. In general, one cannot determine τ without knowledge of the full-resolution data ω and $\boldsymbol{u}$ at each timestep—i.e. we cannot explicitly write $\partial_t \bar{\omega} = \bar{R}(\bar{\omega})$—and so (4.28) requires a model of τ in order to advance. Considering downsampling in addition to filtering increases the modeling complexity. Ignoring this subgrid term and directly running the CFD on a coarse-grain initial condition, unfortunately, produces undesirable results, even when the CFD has built in filtering to induce stability. This is due to the wide range of interacting scales present in turbulent flow. It is easy to see that a substantial number of small scales are lost when filtering and downsampling the state by comparing Figures 4.2a and 4.2b, and the resulting error that accumulates when running the CFD at the coarse resolution as shown in Figure 4.2c.



(a) **(b)** **(c)**

Figure 4.2: Three vorticity samples generated from the same initial condition. Full resolution DNS of ω is generated according to (4.27), and shown in 4.2a. This is filtered and downsampled in 4.2b. In 4.2c, the vorticity is evolved at the coarse grain level, and the lack of subgrid corrections is evident.

Since the flow has a forcing term, and our goal is to learn a sugbrid correction to the CFD (i.e. learn τ), in practice we train on the error in the CFD applied to coarse-grain data because the CFD has $\boldsymbol{F}$ built in. That is, find Θ and $\boldsymbol{\Omega}^{(0)}, \boldsymbol{\Omega}^{(1)}, \dots, \boldsymbol{\Omega}^{(K)}$ so that

$$MZ\left(\boldsymbol{g}^n, \boldsymbol{g}^{n-1}, \dots, \boldsymbol{g}^{n-K}; \boldsymbol{\Omega}^{(0)}, \boldsymbol{\Omega}^{(1)}, \dots, \boldsymbol{\Omega}^{(K)}\right) \approx \boldsymbol{G}^{n+1} - CFD(\boldsymbol{G}^n),$$

for all n, recalling $\boldsymbol{g}^n = (\boldsymbol{G}^n, NN(\boldsymbol{G}^n; \Theta))$. With the kernels and Θ fixed, one step predictions are then made via

$$\boldsymbol{G}_{LO}^{n+1} = CFD(\boldsymbol{G}^n) + MZ\left(\boldsymbol{g}^n, \boldsymbol{g}^{n-1}, \dots, \boldsymbol{g}^{n-K}; \boldsymbol{\Omega}^{(0)}, \boldsymbol{\Omega}^{(1)}, \dots, \boldsymbol{\Omega}^{(K)}\right). \tag{4.29}$$

Our goal is to determine observables $\boldsymbol{g}$ which produce a *good* model $MZ(\,\cdot\,;\boldsymbol{\Omega}^{(0)},\boldsymbol{\Omega}^{(1)},\cdots,\boldsymbol{\Omega}^{(K)})$, where *good* is evaluated in two ways:

1. How well evolving $\boldsymbol{G}^n$ at this lower dimension produces good agreement with coarse-grain observations of the ground truth at the next timestep, $\boldsymbol{G}^{n+1}$. That is, check $\boldsymbol{G}_{LO}^{n+1} \approx \boldsymbol{G}^{N+1}$, where $\boldsymbol{G}_{LO}^{n+1}$ is generated according to (4.29) with coarse-grain ground truth data plugged in to $CFD(\cdot)$ and $MZ(\cdot)$.

2. How well the discovered model is able to generate long trajectories which remain stable and match the flow statistics well. Long trajectories are formed by starting with some initial data, $\boldsymbol{G}_{LO}^{0} = \boldsymbol{G}^0, \boldsymbol{G}_{LO}^{2} = \boldsymbol{G}^1, \ldots, \boldsymbol{G}_{LO}^{K} = \boldsymbol{G}^K$, and iteratively forming $\boldsymbol{G}_{LO}^{K+1}$, $\boldsymbol{G}_{LO}^{K+2}, \ldots$ accumulating error along the way

$$\boldsymbol{g}_{LO}^{n+1} = (\boldsymbol{G}_{LO}^{n+1}, NN(\boldsymbol{G}_{LO}^{n+1} l; \Theta)), \tag{4.30}$$

$$\boldsymbol{G}_{LO}^{n+1} = CFD(\boldsymbol{G}_{LO}^{n}) + MZ\left(\boldsymbol{g}_{LO}^{n}, \boldsymbol{g}_{LO}^{n-1}, \ldots, \boldsymbol{g}_{LO}^{n-K}; \boldsymbol{\Omega}^{(0)}, \boldsymbol{\Omega}^{(1)}, \ldots, \boldsymbol{\Omega}^{(K)}\right). \tag{4.31}$$

   The updates (4.30) and (4.31) should remain stable after over many n and match flow statistics despite $\boldsymbol{G}_{LO}^{n}$ potentially de-correlating from the ground truth $\boldsymbol{G}^n$.

4.4.1 One-Step Predictions

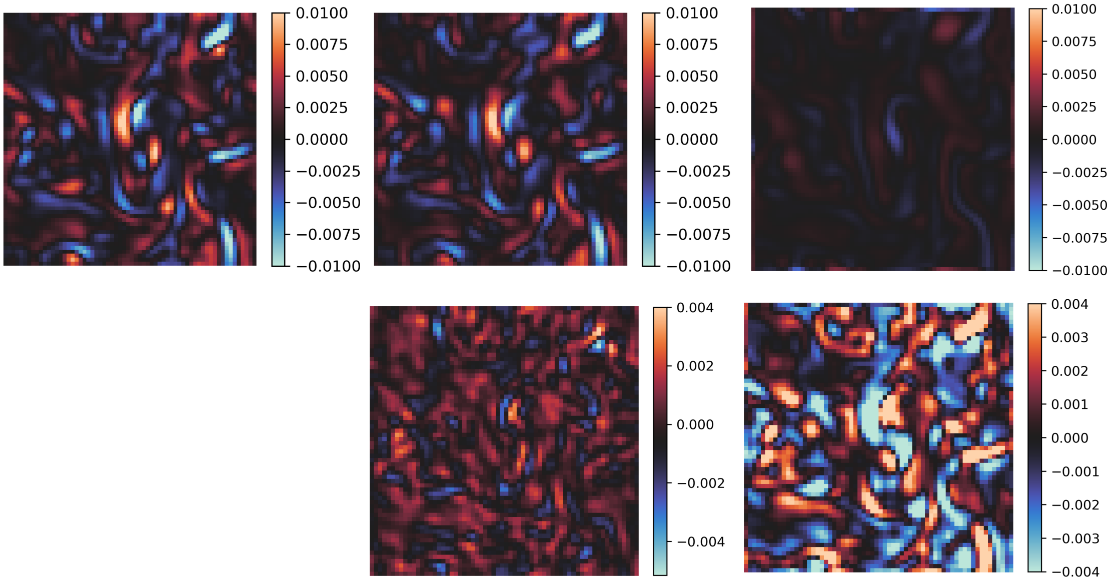


Figure 4.3: One-step predictions. Top row: (left) subgrid error from CFD; (middle) MZ_LO predictions of subgrid term; (right) MZ_PO prediction of subgrid term. Bottom row: corresponding errors for the quantities above when predicting the subgrid term for both learned and predefined observables.

The learned function $MZ(\,\cdot\,)$ in tandem with $NN(\,\cdot\,)$ produce predictions one timestep in the future. The error incurred by the CFD in a single timestep is shown in the top-left of Figure 4.3. This subgrid error is corrected by (4.29) with learned observables, and the predicted subgrid term very accurately mimics its target. Using the predefined observables $\boldsymbol{g}_{PO}$ instead, the subgrid term is sorely underestimated. The predicted subgrid terms and the corresponding errors are shown Figure 4.3.

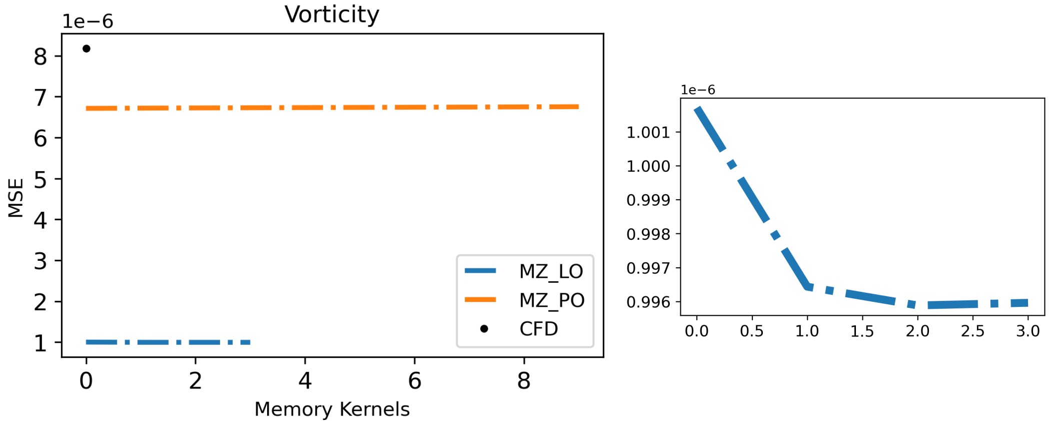


Figure 4.4: Left: Error versus number of memory kernels, $\boldsymbol{\Omega}^{(l)}$ used in the MZ learned observable (MZ_LO) and predefined observable (MZ_PO) 1-step predictions. One-step error for the CFD (which has no memory kernels) is also shown. Right: MZ_LO plotted on its own axis showing the decreasing error for increased memory depth.

The mean-squared errors over the test set for one-step vorticity predictions using CFD, predefined observables and learned observables are shown in Figure 4.4. Using $\boldsymbol{g}_{PO}$, the MZ_PO method provides a modest 19.3% reduction in MSE over the single-step error accumulated by the CFD on coarse-grain data. Learning $\boldsymbol{g}_{LO}^{n}$, however, improves predictions by 87.9% over CFD and 85.2% over predictions made using $\boldsymbol{g}_{PO}$, confirming that the alternating optimization process in Algorithm 11 is a functioning observable discovery mechanism, and that these learned observables greatly boost the prediction capabilities of a truncated MZ operator.

A typical set of observables is shown in Figure 4.5. The additional terms, $NN(\boldsymbol{G}^{n};\Theta)$, exhibit significantly more fragmentation than the vorticity and velocity components making up $\boldsymbol{G}^{n}$ (top row Fig. 4.5). We believe this added fine-scale structure is filling in information at the subgrid scale.

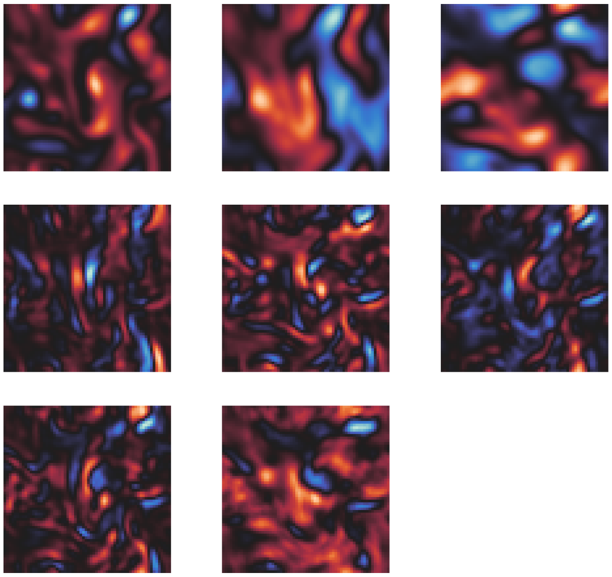


Figure 4.5: Visualization of $\boldsymbol{g}_{LO}^{n}$. The generating set, $\boldsymbol{G}^n$ is shown in the top row. The remaining images are from the learned additions $NN(\boldsymbol{G}^n; \Theta)$.

4.4.2 Long Trajectories

While one-step predictions are a confirmation that the process is able to learn correctly, long-term predictions of a dynamical system are the true measure of a model's success. Following (4.30) and (4.31), we generate a 5000 timestep ($5000\Delta = 7.5$s) trajectory, and compare the effects of using learned observables, $\boldsymbol{g}_{LO}$, with the predefined set, $\boldsymbol{g}_{PO}$, or direct CFD. The eddy turnover time, t_L of the system is approximately $L/\bar{U}$, where L is the lengthscale of the large eddies and $\bar{U}$ is the root-mean-square velocity of the filtered flow. We take $L = \pi/2$ (based on forcing function $\boldsymbol{F}(x, y) = (\sin(4y), 0))^T$) and compute $\bar{U} \sim 1.43s^{-1}$, giving $t_L \sim 1.1$s. Hence, $5000\Delta \sim 6.8t_L$, and conservatively we can use $1000\Delta \sim t_L$. MSE plots over the extended trajectory are given in Figure 4.6.

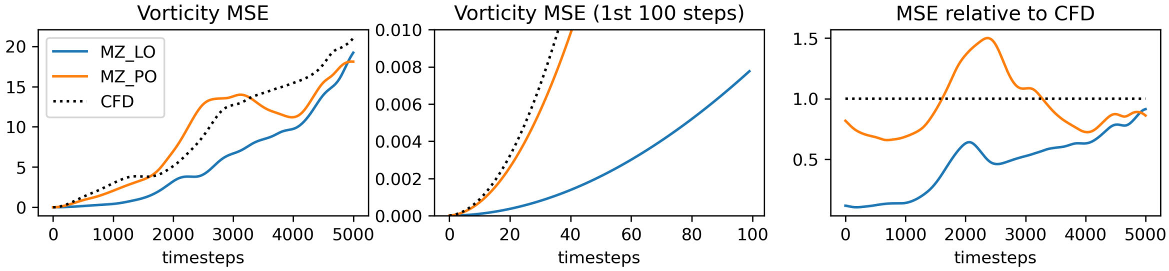


Figure 4.6: Mean squared error for long term trajectories (left) of learned observables (MZ_LO) compared with CFD and predefined observable trajectories (MZ_PO). MZ_LO improves predictions considerably, especially evident in the early timesteps plot (middle). The relative error plot (right) shows that over the first 1000 timesteps ($1.36t_L$), where the MZ_LO trajectories have less than 20% of the error a CFD trajectory run directly on coarse-grain data. Total trajetory length is $6.8t_L$.

We note that MZ_LO appears visually correlated with the ground truth out to intermediate time-scales (around 2000Δ cf. Fig. 4.8), and over this range it significantly outperforms both CFD and MZ_PO. For the first 1000 timesteps, MZ_LO maintains less than 20% the error of CFD.

At later periods, all trajectories have decorrelated from the ground truth, and consequently have high MSE. This is expected, however over these intervals, one hopes to match turbulent flow statistics. We compute several statistics for the trajectories, and display them in Figure 4.7. The histograms of the velocity and velocity-gradient components of the flow for each trajectory match the ground truth well. However, enstrophy, turbulent kinetic energy and the energy spectra of the flow reveal that MZ_LO is maintaining much better proximity to the ground truth. The CFD enstrophy shoots rapidly away from the ground truth, whereas MZ_LO and MZ_PO are more stable. One can see the filtering and anti-aliasing built into the CFD by examining the kinetic energy spectra in Figure 4.7, where there's a harsh cutoff at the Nyquist limit. However, through mid-wavenumbers MZ_LO holds tightly to the GT, and maintains lower high frequency terms out to the Nyquist limit. The higher frequencies in the CFD are evident in Figure 4.8, where we see highly oscillatory terms cropping up. There we can also see MZ_PO successfully mitigates these high-frequency aberrations.

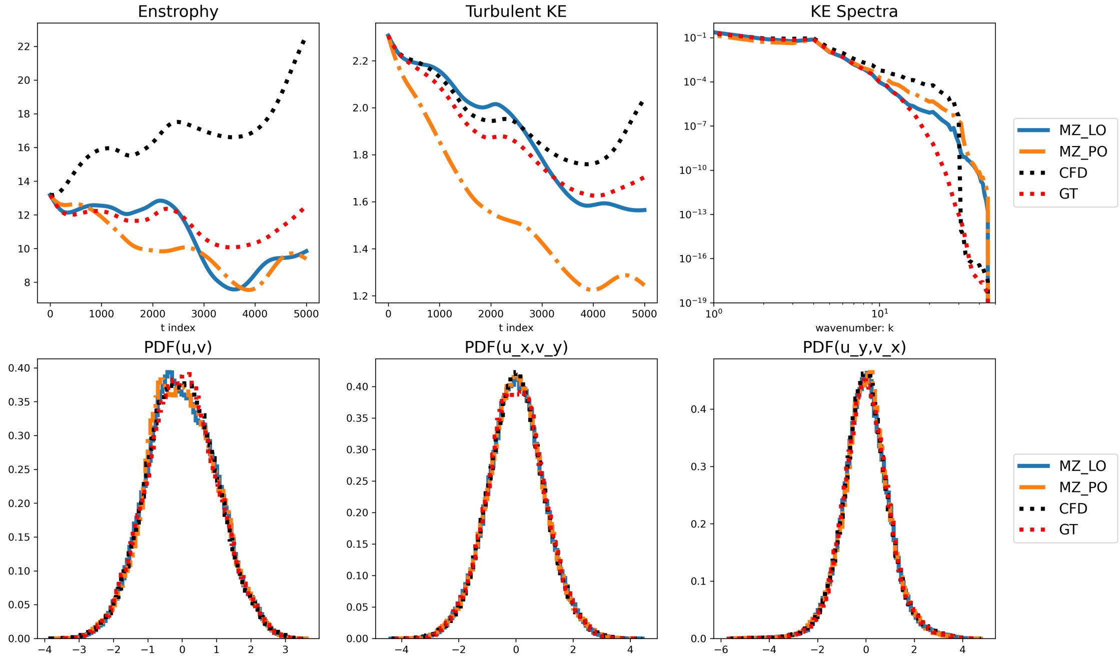


Figure 4.7: Turbulent flow statistics for 5000 step MZ_LO, MZ_PO and CFD trajectories. First row: enstrophy, turbulent kinetic energy and kinetic energy spectra for each trajectory compared against the ground truth (GT). Second row: PDF's are histograms of the velocity components (u, v) and their gradients terms (u_x, v_y) and (u_y, v_x).

Snapshots of each 5000 step trajectory are shown in Figure 4.8. We see that both MZ_PO and MZ_LO are able to maintain reasonable representations of a vorticity field, while the CFD quickly develops highly oscillatory behavior. This is evidence of the stability the Markov and memory kernels add. The addition of learned observables, however, increases the accuracy of the trajectory in addition to this stability, and MZ_LO is highly correlated through the middle frame (2000Δ).

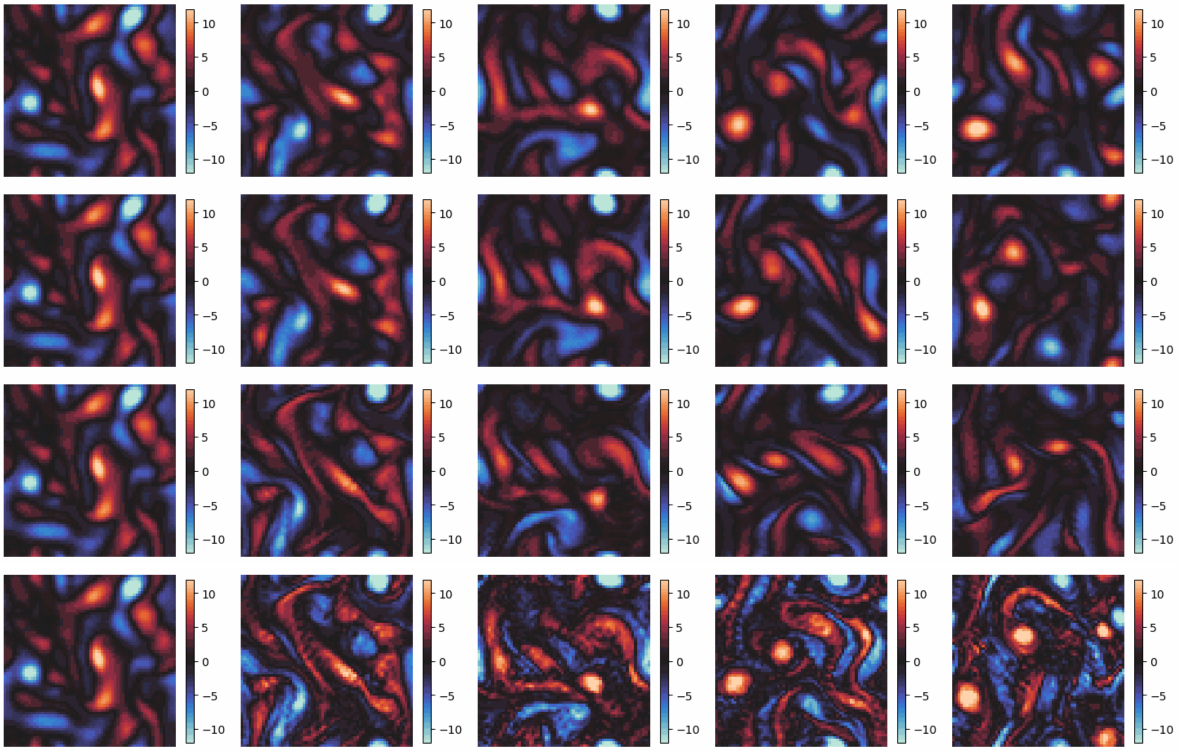


Figure 4.8: Evolution of long trajectories of the vorticity field. First row: ground truth; second row: MZ with learned observables; third row: MZ with predefined observables; fourth row: coarse-grain CFD. Frames are spaced at $1000\Delta \sim 1.36 t_L$ intervals.

4.5 Conclusion

In this chapter, we demonstrate a novel data-driven observable discovery method which successfully defines a resolved space on which to apply the generalized Langevin dynamics as determined by the Mori-Zwanzig formalism. We pair this observable discovery with data-driven approaches [TLA21, LTP23] for performing reduced order modeling on dynamical systems. In particular, we consider turbulent, isotropic, forced Navier-Stokes flow in two-dimensions, and apply the Mori-Zwanzig reduced order model on a 16-fold reduced order system.

Observables, $\boldsymbol{g} = (\boldsymbol{G}, NN(\boldsymbol{G}; \Theta))$, are determined by a convolutional neural network, $NN(\cdot)$ and seeded with a generating observable set, $\boldsymbol{G}$. By using the generating set, we control the quantities of interest we want to evolve forward via the Mori-Zwanzig kernels, and are able to directly track reduced-order representations of the system state (such as vorticity and velocity). Without this generating set, the observable discovery process could learn trivial (i.e. zero maps) or less desirable quantities to track, such as average velocity. Moreover, the generalized fluctuation-dissipation relationship (GFD) is enforced by the construction of the memory kernels, which are defined by a regression-based projection operator as in [LTP23]. This is an important distinction from other memory based data-driven approaches which are inspired by the Mori-Zwanzig formalism, but do not ensure the memory mechanism obeys the GFD.

Overall, the addition of learned observables decreases one-step prediction error by 87.9% compared with a computational fluid dynamics program run directly on the reduced-order state. We also compare against a Mori-Zwanzig kernel learning approach which uses a predefined observable set consisting of the velocity, vorticity and their gradients. Even with a more expressive projection-operator, the hand-picked set is outperformed by the learned observables, which on average produce over 85% error reduction in single-step predictions. The learned observables paired with memory kernels also creates a stable subgrid correction to the CFD, and long trajectories (above six eddy turnover times) remain stable long after decorrelating with the ground truth.

In the future, we would like to extend these experiments to turbulence in three-dimensions as well as to other dynamical systems, such as Kuramoto-Sivashinsky flow. Importantly, we have not yet provided an interpretation of the discovered observables and the MZ operators they are evolved by. Fortunately, the Markov and memory kernels can be kept very simple (in our case linear), so there is direct way to understand the contribution of the observable components if the observables themselves can be deciphered. This is an important area to explore, as interpreting the observables in terms of the physical state could lead to improved, explicit subgrid models for turbulence modeling.

CHAPTER 5

Conclusion

In this thesis we studied three separate inverse problems: denoising an image corrupted by multiplicative noise, extracting wind velocity fields from remote sensing data, and determining an appropriate set of observables with which to perform turbulent flow simulations using Mori-Zwanzig based operators.

The first is a classical mathematical imaging task for a difficult type of corruption common in synthetic aperture radar and medical imaging applications. We establish a general multiscale hierarchical decomposition method (MHDM) for multiplicative noise, and apply it to several multiplicative image denoising models, including a novel formulation. The MHDM construction recovers images at multiple scales. This property allows both smooth and textured regions to be processed, and overall proves successful at retaining texture features which existing methods oversmooth. The multiscale property has implications beyond denoising and could be applied to image decomposition tasks as well. The MHDM architecture additionally simplifies weighting parameter choice during optimization. The weighting parameter, λ, which balances the emphasis of the data fidelity and regularization terms in the objective functional, typically would require tedious estimations in non-multiscale methods in order to obtain good results.

We demonstrate well-posedness for the MHDM process applied to several models. Theoretical guarantees of decreasing fidelity terms with increasing hierarchical depth are established, along with a viable stopping criterion which prevents excess noise from being recovered. In addition to the standard MHDM process, we develop tight and refined extensions which further control the convergence of our multi-scale approximations of the clean image. Numerical schemes are laid out in detail for several models, and overall, we see improved performance in both natural, textured and smooth synthetic images. Most importantly, the MHDM procedure's general design allows it to be applied to many models, and so it is not limited to the fundamental models tested within.

Secondly, we consider whether optical flow can be used to determine wind velocities from sequential image data and develop two tools which improve atmospheric motion estimation. The optical flow objective is an ill-posed inverse problem. For water vapor datasets, we find that TV-$L1$ approaches struggle to accurately determine the velocity within regions of uniform data. To correct this, we propose using a structure-texture decomposition to extract features which may improve the flow estimates. We establish that texture is a viable feature, and build a *Texture-Flow* scheme which finds an initial flow estimate from the texture-features of the image data before refining the estimate using the original image. This two-step process improves estimates, especially in trouble-regions compared with TV-$L1$.

Building on the insight from Texture-Flow, we establish a Multi-Fidelity Flow process which accomplishes two objectives. First, it allows the incorporation of a texture-based feature into the fidelity term in concert with the original data, eliminating the two-step process required in Texture-Flow. Second, Multi-Fidelity flow is flexible and can accept many features into the fidelity term simultaneously. Leveraging this, we determine that image texture, gradient and gradient-norm are all helpful features for the wind velocity extraction problem. Overall, testing on several datasets representing convective and cyclonic weather systems, we see 3-20% improvements when incorporating texture alone. Adding additional features increases accuracy improvements by an additional 10% (from 20% to 30%) over TV-

$L1$ for the mesoscale convective system. Both of our methods outperform the Horn Schunck and TV-$L1$ schemes over data of various spatial and temporal resolutions with complex motions. Additionally, our estimates are spatially dense and require only two frames of data, which makes them well suited for real-time weather "now-casting" applications without needing to collect long-term data samples.

Finally, we develop a data-driven approach for observable discovery which aids a Mori-Zwanzig based reduced-order modeling approach. The Mori-Zwanzig formalism provides a mathematical structure for evolving *observables* according to a general Langevin dynamics equation (GLE). The GLE incorporates memory effects into the dynamics via memory-kernels and a Markov term, and these kernels can be learned with appropriate data. However, appropriately choosing an observable set to evolve is not well understood. We establish a data-driven joint learning problem which simultaneously determines the observable set along with the evolution kernels, and show improved performance on coarse-grain turbulence simulations, even over long trajectories beyond six eddy-turnover timescales. Moreover, by construction our memory kernels satisfy the generalized fluctuation-dissipation relationship which is imposed by the Mori-Zwanzig theory and distinguishes our process from other memory-based data-driven models.

In future work, we would like to extend the MHDM process to multi-channel images and consider this approach on more general image decomposition or segmentation tasks. The theory for recovering blurred images with an MHDM approach can also be extended. There are several directions for additional exploration for the optical flow project. We would like to test additional features to improve the flow and explore non-local modifications to the optical flow constraint equations. We would very much like to interpret the discovered observables used within the Mori-Zwanzig kernels of Chapter 4. Because we use simple projection-based kernels, understanding how the observables are related to the initial states (or generating set) would give a physical subgrid closure model for turbulent fluid simulation. This would have many impactful ramifications for the fluid modeling sciences. Additionally, applying this observable discovery process to other dynamical systems (including turbulence in three-dimensions) should be considered.

While each of these projects are distinct, we close with a remark on the overlap they share and possible future directions, where ideas from each may inspire advancements for one another. At their core, each of these are inverse problems. Since the multiscale process within the image recovery problem naturally gives rise to structure-texture decompositions of images, MHDM based decompositions could be used to form additional features for the Texture-Flow process in Chapter 3. Moreover, the optical flow project is intrinsically related to coarse-grain fluid simulation since the remote-sensing data represents snapshots from a drastically under-resolved (on the order of kilometers/pixel) dynamical fluid system. Thus, given an accurate coarse-grain velocity field for the fluid motion, we cannot expect it to transport the water vapor accurately unless subgrid corrections are incorporated into the forward model. An ideal fidelity term for the optical flow constraint equation, necessarily, would incorporate these subgrid effects. Here, there is an opportunity for determining such a fidelity term by applying the data-driven observable discovery approach to the optical flow problem. The Mori-Zwanzig kernels, which define an evolution equation for the velocity field (along with additional components of the observable set), can be understood as an optical flow constraint equation. Inverting this equation for velocity reveals the desired wind velocity field. This would hinge on understanding the observable generating function and the kernel actions.

REFERENCES

[AA08] Gilles Aubert and Jean-Franois Aujol. “A Variational Approach to Removing Multiplicative Noise.” *SIAM Journal on Applied Mathematics*, **68**(4):925–946, 2008.

[ABH93] Patrick Anandan, James R Bergen, Keith J Hanna, and Rajesh Hingorani. “Hierarchical model-based motion estimation.” In *Motion analysis and image sequence processing*, pp. 1–22. Springer, 1993.

[AD90] Luigi Ambrosio and Giani Dal Maso. “A general chain rule for distributional derivatives.” *Proceedings of the American Mathematical Society*, **108**:691–702, 1990.

[Adr05] Ronald J Adrian. “Twenty years of particle image velocimetry.” *Experiments in Fluids*, **39**(2):159–169, 2005.

[AV94] Robert Acar and Curtis R Vogel. “Analysis of bounded variation penalty methods for ill-posed problems.” *Inverse problems*, **10**(6):1217, 1994.

[BBF97] Albert Bijaoui, Yves Bobichon, Yanling Fang, and Frederic Rue. “Multiscale Methods Applied to the Analysis of Synthetic Aperture Radar Images.” *Traitement du Signal*, **14**:179–194, 1997.

[BBP04] Thomas Brox, Andrés Bruhn, Nils Papenberg, and Joachim Weickert. “High accuracy optical flow estimation based on a theory for warping.” In *European conference on computer vision*, pp. 25–36. Springer, 2004.

[BBP16] Steven L Brunton, Bingni W Brunton, Joshua L Proctor, and J Nathan Kutz. “Koopman invariant subspaces and finite linear representations of nonlinear dynamical systems for control.” *PloS one*, **11**(2):e0150171, 2016.

[BBV23a] Joel Barnett, Andrea Bertozzi, Luminita Vese, and Igor Yanovsky. “Incorporating Texture Features into Optical Flow for Atmospheric Wind Velocity Estimation.” In *IGARSS 2023 - 2023 IEEE International Geoscience and Remote Sensing Symposium*, pp. 3784–3787. IEEE, 2023.

[BBV23b] Joel Barnett, Andrea Bertozzi, Luminita A. Vese, and Igor Yanovsky. “Texture-based optical flow for wind velocity estimation from water vapor data.” In Weilin Hou and Linda J. Mullen, editors, *Ocean Sensing and Monitoring XV*, volume 12543, pp. 164–175. International Society for Optics and Photonics, SPIE, 2023.

[BD06] Wayne Bresky and Jaime Daniels. “The feasibility of an optical flow algorithm for estimating atmospheric motion.” In *Proceedings of the Eighth Int. Winds Workshop, Beijing, China*, pp. 24–28. Citeseer, 2006.

[BD18] Durga Prasad Bavirisetti and Ravindra Dhuli. "Multifocus image fusion using multiscale image decomposition and saliency detection." *Ain Shams Engineering Journal*, **9**(4):1103–1117, 2018.

[BLR23] Joel Barnett, Wen Li, Elena Resmerita, and Luminita Vese. "Multiscale hierarchical decomposition methods for images corrupted by multiplicative noise.", 2023.

[Bur78] Christoph B Burckhardt. "Speckle in ultrasound B-mode scans." *IEEE Transactions on Sonics and ultrasonics*, **25**:1–6, 1978.

[BWS05] Andrés Bruhn, Joachim Weickert, and Christoph Schnörr. "Lucas/Kanade meets Horn/Schunck: Combining local and global optic flow methods." *International journal of computer vision*, **61**:211–231, 2005.

[CBK15] Chi Wai Chow, Serge Belongie, and Jan Kleissl. "Cloud motion and stability estimation for intra-hour solar forecasting." *Solar Energy*, **115**:645–655, 2015.

[CCN09] Antonin Chambolle, Vicent Caselles, Matteo Novaga, Daniel Cremers, and Thomas Pock. "An introduction to Total Variation for Image Analysis." working paper or preprint, November 2009.

[CD09] Antonin Chambolle and Jerome Darbon. "On Total Variation Minimization and Surface Evolution Using Parametric Maximum Flows." *International Journal of Computer Vision*, **84**:288–307, 2009.

[CDI23] Salvatore Cuomo, Mariapia De Rosa, Stefano Izzo, Francesco Piccialli, and Monica Pragliola. "Speckle noise removal via learned variational models." *Applied Numerical Mathematics*, 2023.

[CFX15] Guangmang Cui, Huajun Feng, Zhihai Xu, Qi Li, and Yueting Chen. "Detail preserved fusion of visible and infrared images using regional saliency extraction and multi-scale image decomposition." *Optics Communications*, **341**:199–209, 2015.

[Cha04] Antonin Chambolle. "An algorithm for total variation minimization and applications." *Journal of Mathematical imaging and vision*, **20**(1):89–97, 2004.

[CHA06] Thomas Corpetti, Dominique Heitz, Georges Arroyo, Etienne Mémin, and Alina Santa-Cruz. "Fluid experimental flow estimation based on an optical-flow scheme." *Experiments in fluids*, **40**(1):80–97, 2006.

[Cha23] Antonin Chambolle. "Total variation minimization for image reconstruction: The multiplicative noise case." *preprint*, 2023.

[CHK02] Alexandre J Chorin, Ole H Hald, and Raz Kupferman. “Optimal prediction with memory.” *Physica D: Nonlinear Phenomena*, **166**(3-4):239–257, 2002.

[CL95] Antonin Chambolle and Pierre-Louis Lions. “Image restoration by constrained total variation minimization and variants.” volume 2567, 1995. SPIE Electronic Imaging Proceedings.

[CL97] Antonin Chambolle and Pierre-Louis Lions. “Image recovery via total variation minimization and related problems.” *Numerische Mathematik*, **76**:167–188, 1997.

[CMM00] Tony Chan, Antonio Marquina, and Pep Mulet. “High-order total variation-based image restoration.” *SIAM Journal on Scientific Computing*, **22**(2):503–516, 2000.

[CV99] Tony Chan and Luminita Vese. “An active contour model without edges.” In *International conference on scale-space theories in computer vision*, pp. 141–151. Springer, 1999.

[CV00] Tony F Chan and Luminita A Vese. “An efficient variational multiphase motion for the Mumford-Shah segmentation model.” In *Conference Record of the Thirty-Fourth Asilomar Conference on Signals, Systems and Computers (Cat. No. 00CH37154)*, volume 1, pp. 490–494. IEEE, 2000.

[CV01] Tony F Chan and Luminita A Vese. “Active contours without edges.” *IEEE Transactions on image processing*, **10**(2):266–277, 2001.

[CZS15] Xu Chen, Pascal Zillé, Liang Shao, and Thomas Corpetti. “Optical flow for incompressible turbulence motion estimation.” *Experiments in fluids*, **56**(1):1–14, 2015.

[DLV23] Noémie Debroux, Carole Le Guyader, and Luminita A. Vese. “A Multiscale Deformation Representation.” *SIAM Journal on Imaging Sciences*, **16**:802–841, 2023.

[DMR22] Jerome Darbon, Tingwei Meng, and Elena Resmerita. “On Hamilton?Jacobi PDEs and image denoising models with certain nonadditive noise.” *J Math Imaging Vis*, **64**:408–441, 2022.

[DSK09] Eric Darve, Jose Solomon, and Amirali Kia. “Computing generalized Langevin equations and generalized Fokker–Planck equations.” *Proceedings of the National Academy of Sciences*, **106**(27):10884–10889, 2009.

[DZ13] Yiqiu Dong and Tieyong Zeng. “A Convex Variational Model for Restoring Blurred Images with Multiplicative Noise.” *SIAM Journal on Imaging Sciences*, **6**(3):1598–1625, 2013.

[Goo76] Joseph W Goodman. "Some fundamental properties of speckle." *JOSA*, **66**(11):1145–1150, 1976.

[GSS23] Priyam Gupta, Peter J Schmid, Denis Sipp, Taraneh Sayadi, and Georgios Rigas. "Mori-Zwanzig latent space Koopman closure for nonlinear autoencoder." *arXiv preprint arXiv:2310.10745*, 2023.

[HHM08] Dominique Heitz, Patrick Héas, Etienne Mémin, and Johan Carlier. "Dynamic consistent correlation-variational approach for robust optical flow estimation." *Experiments in Fluids*, **45**(4):595–608, 2008.

[HJL21] John Harlim, Shixiao W Jiang, Senwei Liang, and Haizhao Yang. "Machine learning for prediction with missing dynamics." *Journal of Computational Physics*, **428**:109922, 2021.

[HMS10] Dominique Heitz, Etienne Mémin, and Christoph Schnörr. "Variational fluid flow measurements from image sequences: synopsis and perspectives." *Experiments in fluids*, **48**(3):369–393, 2010.

[HNW09] Yu-Mei Huang, Michael K. Ng, and You-Wei Wen. "A New Total Variation Method for Multiplicative Noise Removal." *SIAM J. Imaging Sci.*, **2**:20–40, 2009.

[HS81] Berthold K.P. Horn and Brian G Schunck. "Determining optical flow." *Artificial intelligence*, **17**(1-3):185–203, 1981.

[Hun73] Bobby R. Hunt. "The Application of Constrained Least Squares Estimation to Image Restoration by Digital Computer." *IEEE Transactions on Computers*, **C-22**(9):805–812, 1973.

[JWM22] Zhengmeng Jin, Jie Wang, Lihua Min, and Minling Zheng. "An adaptive total generalized variational model for speckle reduction in ultrasound images." *J. Franklin Inst.*, **359**:8377–8394, 2022.

[JY10] Zhengmeng Jin and Xiaoping Yang. "Analysis of a new variational model for multiplicative noise removal." *Journal of Mathematical Analysis and Applications*, **362**:415–426, 2010.

[Koo31] Bernard Osgood Koopman. "Hamiltonian Systems and Transformation in Hilbert Space." *Proceedings of the National Academy of Sciences*, **17**(5):315–318, 1931.

[KRW23] Stefan Kindermann, Elena Resmerita, and Tobias Wolf. "Multiscale hierarchical decomposition methods for ill-posed problems." *arXiv:2304.08332*, 2023.

[KSA21] Dmitrii Kochkov, Jamie A Smith, Ayya Alieva, Qing Wang, Michael P Brenner, and Stephan Hoyer. "Machine learning–accelerated computational fluid dynamics." *Proceedings of the National Academy of Sciences*, **118**(21):e2101784118, 2021.

[LF16] Min Liu and Qibin Fan. "A modified convex variational model for multiplicative noise removal." *Journal of Visual Communication and Image Representation*, **36**:187–198, 2016.

[LK81] Bruce D Lucas, Takeo Kanade, et al. *An iterative image registration technique with an application to stereo vision*, volume 81. Vancouver, 1981.

[LKB18] Bethany Lusch, J Nathan Kutz, and Steven L Brunton. "Deep learning for universal linear embeddings of nonlinear dynamics." *Nature communications*, **9**(1):4950, 2018.

[LL21] Kevin K Lin and Fei Lu. "Data-driven model reduction, Wiener projections, and the Koopman-Mori-Zwanzig formalism." *Journal of Computational Physics*, **424**:109864, 2021.

[LMM15] Tianshu Liu, Ali Merat, M. H. M. Makhmalbaf, Claudia Fajardo, and Parviz Merati. "Comparison between optical flow and cross-correlation methods for extraction of velocity fields from particle images." *Experiments in Fluids*, **56**(8):166, 2015.

[LRV21] Wen Li, Elena Resmerita, and Luminita Vese. "Multiscale Hierarchical Image Decomposition and Refinements: Qualitative and Quantitative Results." *SIAM J. Imaging Sci.*, **14**(2):844–877, 2021.

[LSF20] Tianshu Liu, David M Salazar, Hassan Fagehi, Hassan Ghazwani, Javier Montefort, and Parviz Merati. "Hybrid optical-flow-cross-correlation method for particle image velocimetry." *Journal of Fluids Engineering*, **142**(5), 2020.

[LTL21] Yen Ting Lin, Yifeng Tian, Daniel Livescu, and Marian Anghel. "Data-Driven Learning for the Mori–Zwanzig Formalism: A Generalization of the Koopman Learning Framework." *SIAM Journal on Applied Dynamical Systems*, **20**(4):2558–2601, 2021.

[LTP23] Yen Ting Lin, Yifeng Tian, Danny Perez, and Daniel Livescu. "Regression-based projection for learning Mori–Zwanzig operators." *SIAM Journal on Applied Dynamical Systems*, **22**(4):2890–2926, 2023.

[LV05] Triet M Le and Luminita A Vese. "Image decomposition using total variation and div (BMO)." *Multiscale Modeling & Simulation*, **4**(2):390–423, 2005.

[LV08] Carole Le Guyader and Luminita A Vese. "Self-repelling snakes for topology-preserving segmentation models." *IEEE Transactions on Image Processing*, **17**(5):767–779, 2008.

[LYZ21] Jin Lu, Hua Yang, Qinghu Zhang, and Zhouping Yin. “An accurate optical flow estimation of PIV using fluid velocity decomposition.” *Experiments in Fluids*, **62**(4):1–16, 2021.

[Mey01] Yves Meyer. *Oscillating patterns in image processing and nonlinear evolution equations: the fifteenth Dean Jacqueline B. Lewis memorial lectures*, volume 22. American Mathematical Soc., 2001.

[Mez05] Igor Mezić. “Spectral properties of dynamical systems, model reduction and decompositions.” *Nonlinear Dynamics*, **41**:309–325, 2005.

[MM16] Alexandre Mauroy and Igor Mezić. “Global stability analysis using the eigenfunctions of the Koopman operator.” *IEEE Transactions on Automatic Control*, **61**(11):3356–3369, 2016.

[MNR19] Klas Modin, Adrian Nachman, and Luca Rondi. “A multiscale theory for image registration and nonlinear inverse problems.” *Advances in Mathematics*, **346**:1009–1066, 2019.

[MO07] Hazime Mori and Makoto Okamura. “Dynamic structures of the time correlation functions of chaotic nonequilibrium fluctuations.” *Physical Review E*, **76**(6):061104, 2007.

[Mor65] Hazime Mori. “Transport, collective motion, and Brownian motion.” *Progress of theoretical physics*, **33**(3):423–455, 1965.

[MS85] David Mumford and Jayant Shah. “Boundary detection by minimizing functionals.” In *IEEE Conference on Computer Vision and Pattern Recognition*, volume 17, pp. 137–154. San Francisco, 1985.

[MS89] David Bryant Mumford and Jayant Shah. “Optimal approximations by piecewise smooth functions and associated variational problems.” *Communications on pure and applied mathematics*, 1989.

[MS13] Enric Meinhardt-Llopis and Javier Sánchez. “Horn-schunck optical flow with a multi-scale strategy.” *Image Processing on line*, 2013.

[MW18] Chao Ma, Jianchun Wang, et al. “Model reduction with memory and the machine learning of dynamical systems.” *arXiv preprint arXiv:1808.04258*, 2018.

[MW20] Shinya Maeyama and Tomo-Hiko Watanabe. “Extracting and modeling the effects of small-scale fluctuations on large-scale fluctuations by Mori–Zwanzig projection operator method.” *Journal of the Physical Society of Japan*, **89**(2):024401, 2020.

[NW22] Siwei Nao and Yan Wang. "Speckle noise removal model based on diffusion equation and convolutional neural network." *Computational Intelligence and Neuroscience*, 2022.

[OR19] Samuel E Otto and Clarence W Rowley. "Linearly recurrent autoencoder networks for learning dynamics." *SIAM Journal on Applied Dynamical Systems*, **18**(1):558–593, 2019.

[OSV03] Stanley Osher, Andrés Solé, and Luminita Vese. "Image decomposition and restoration using total variation minimization and the H^{-1} norm." *Multiscale Modeling & Simulation*, **1**(3):349–370, 2003.

[OZW21] Amir Ouyed, Xubin Zeng, Longtao Wu, Derek Posselt, and Hui Su. "Two-Stage Artificial Intelligence Algorithm for Calculating Moisture-Tracking Atmospheric Motion Vectors." *Journal of Applied Meteorology and Climatology*, **60**(12):1671 – 1684, 2021.

[PD17] Eric J Parish and Karthik Duraisamy. "Non-Markovian closure models for large eddy simulations using the Mori-Zwanzig formalism." *Physical Review Fluids*, **2**(1):014604, 2017.

[PLS06] Dana Paquin, Doron Levy, Eduard Schreibmann, and Lei Xing. "Multiscale image registration." *Math. Biosci. Eng.*, **3**(2):389–418, 2006.

[PWM19] Derek J. Posselt, Longtao Wu, Kevin Mueller, Lei Huang, Fredrick W. Irion, Shannon Brown, Hui Su, David Santek, and Christopher S. Velden. "Quantitative Assessment of State-Dependent Atmospheric Motion Vector Uncertainties." *Journal of Applied Meteorology and Climatology*, **58**(11):2479–2495, 2019.

[RB96] Frédéric Rué and Albert Bijaoui. "A multiscale vision model applied to astronomical images." *Vistas in Astronomy*, **40**:495–502, 1996.

[Ric22] Lewis F Richardson. *Weather prediction by numerical process.* University Press, 1922.

[RLO03] Leonid Rudin, Pierre-Louis Lions, and Stanley Osher. "Multiplicative Denoising and Deblurring: Theory and Algorithms." In *Geometric Level Set Methods in Imaging Vision and Graphics*, pp. 201–213, New York, 2003. Springer.

[RO94a] Leonid Rudin and Stanley Osher. "Total variation based image restoration with free local constraints." volume I, pp. 31–35, 1994. Proc. IEEE ICIP, Austin (Texas) USA.

[RO94b] Leonid I Rudin and Stanley Osher. "Total variation based image restoration with free local constraints." In *Proceedings of 1st international conference on image processing*, volume 1, pp. 31–35. IEEE, 1994.

[ROF92] Leonid I. Rudin, Stanley Osher, and Emad Fatemi. "Nonlinear total variation based noise removal algorithms." *Physica D: Nonlinear Phenomena*, **60**(1):259–268, 1992.

[Sch10] Peter J Schmid. "Dynamic mode decomposition of numerical and experimental data." *Journal of fluid mechanics*, **656**:5–28, 2010.

[SI] University of Southern California Signal and Image Processing Institute. "The USC-SIPI Image Database." https://sipi.usc.edu/database/database.php. Accessed: 2023-04-24.

[SKD08] William C Skamarock, Joseph B Klemp, Jimy Dudhia, David O Gill, Dale M Barker, Michael Duda, Xiang-Yu Huang, Wei Wang, and Jordan G Powers. "A Description of the Advanced Research WRF Version 3." U.S. National Center for Atmospheric Research, 2008.

[SMF13] Javier Sánchez Pérez, Enric Meinhardt-Llopis, and Gabriele Facciolo. "TV-L1 Optical Flow Estimation." *Image Processing On Line*, **3**:137–150, 2013. `https://doi.org/10.5201/ipol.2013.26`.

[SO08] Jianing Shi and Stanley Osher. "A Nonlinear Inverse Scale Space Method for a Convex Multiplicative Noise Model." *SIAM Journal on Imaging Sciences*, **1**(3):294–321, 2008.

[SPC09] Frank Steinbrücker, Thomas Pock, and Daniel Cremers. "Large displacement optical flow computation withoutwarping." In *2009 IEEE 12th International Conference on Computer Vision*, pp. 1609–1614. IEEE, 2009.

[ST10] Gabriele. Steidl and Tanja Teuber. "Removing multiplicative noise by Douglas-Rachford splitting methods." *J. Math. Imaging Vision*, **36**:168–184, 2010.

[TA09] Eitan Tadmor and Prashant Athavale. "Multiscale image representation using novel integro-differential equations." *Inverse Problems & Imaging*, **3**(4):693, 2009.

[TH16] Liming Tang and Chuanjiang He. "Multiscale variational decomposition and its application for image hierarchical restoration." *Computers & Electrical Engineering*, **54**:354–369, 2016.

[TLA21] Yifeng Tian, Yen Ting Lin, Marian Anghel, and Daniel Livescu. "Data-driven learning of Mori–Zwanzig operators for isotropic turbulence." *Physics of Fluids*, **33**(12), 2021.

[TNV04] Eitan Tadmor, Suzanne Nezzar, and Luminita Vese. "A multiscale image representation using hierarchical (BV, L^2) decompositions." *Multiscale Modeling & Simulation*, **2**(4):554–579, 2004.

[TNV08] Eitan Tadmor, Suzanne Nezzar, and Luminita Vese. "Multiscale hierarchical decomposition of images with applications to deblurring, denoising, and segmentation." *Commun. Math. Sci.*, **6**(2):281–307, 2008.

[UCK17] Asmat Ullah, Wen Chen, Mushtaq Ahmad Khan, and HongGuang Sun. "A new variational approach for multiplicative noise and blur removal." *PloS ONE*, **12**(1):e0161787, 2017.

[VC02] Luminita A Vese and Tony F Chan. "A multiphase level set framework for image segmentation using the Mumford and Shah model." *International journal of computer vision*, **50**:271–293, 2002.

[Ves01] Luminita Vese. "A study in the bv space of a denoisingdeblurring variational problem." *Applied Mathematics and optimization*, **44**:131–161, 2001.

[VL16] Luminita A Vese and Carole Le Guyader. *Variational methods in image processing.* CRC Press Boca Raton, FL, 2016.

[Vol67] Aizik Isaakovich Vol'pert. "The spaces BV and quasilinear equations." *Matematicheskii Sbornik*, **115**(2):255–302, 1967.

[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.

[WLL20] Tingting Wu, Wei Li, Lihua Li, and Tieyong Zeng. "A convex variational approach for image deblurring with multiplicative structured noise." *IEEE Access*, **8**:37790–37807, 2020.

[WPZ09] Andreas Wedel, Thomas Pock, Christopher Zach, Horst Bischof, and Daniel Cremers. "An improved algorithm for tv-l 1 optical flow." In *Statistical and geometrical approaches to visual motion analysis*, pp. 23–45. Springer, 2009.

[WSN14] Stfan van der Walt, Johannes L. Schnberger, Juan Nunez-Iglesias, Franois Boulogne, Joshua D. Warner, Neil Yager, Emmanuelle Gouillart, Tony Yu, and the scikit-image contributors. "scikit-image: image processing in Python." *PeerJ*, **2**:e453, jun 2014.

[WW17] Wang-chun Woo and Wai-kin Wong. "Operational Application of Optical Flow Techniques to Radar-Based Rainfall Nowcasting." *Atmosphere*, **8**(3), 2017.

[WYN21] Wei Wang, Mingjia Yao, and Michael K Ng. "Color image multiplicative noise and blur removal by saturation-value total variation." *Applied Mathematical Modelling*, **90**:240–264, 2021.

[XAN14] Robert Xu, Prashant Athavale, Adrian Nachman, and Graham A Wright. "Multiscale registration of real-time and prior MRI data for image-guided cardiac interventions." *IEEE Transactions on Biomedical Engineering*, **61**(10):2621–2632, 2014.

[YKH19] Enoch Yeung, Soumya Kundu, and Nathan Hodas. "Learning deep neural network representations for Koopman operators of nonlinear dynamical systems." In *2019 American Control Conference (ACC)*, pp. 4832–4839. IEEE, 2019.

[YPW23] Igor Yanovsky, Derek Posselt, Longtao Wu, Svetla Hristova-Veleva, Hai Nguyen, Bjorn Lambrigtsen, and Xubin Zeng. "Atmospheric Motion Vector Retrieval Using the Total Variation-Based Optical Flow Method." In *IGARSS 2023 - 2023 IEEE International Geoscience and Remote Sensing Symposium*, pp. 3780–3783, 2023.

[YSP22] Igor Yanovsky, Hui Su, Derek J. Posselt, Svetla Hristova-Veleva, Longtao Wu, Bjorn Lambrigtsen, and Hai Nguyen. "Atmospheric Motion Vector Retrieval using the Optical Flow Method." In *25th Conference on Satellite Meteorology, Oceanography, and Climatology, Joint 2022 NOAA Satellite Conference.* AMS Collective Madison Meeting, 2022.

[ZLG22] Yu Zhang, Songsong Li, Zhichang Guo, Boying Wu, and Shan Du. "Image Multiplicative Denoising Using Adaptive Euler's Elastica as the Regularization." *Journal of Scientific Computing*, **90**(2):1–34, 2022.

[ZPB07] Christopher Zach, Thomas Pock, and Horst Bischof. "A duality based approach for realtime tv-l 1 optical flow." In *Joint pattern recognition symposium*, pp. 214–223. Springer, 2007.

[Zwa73] Robert Zwanzig. "Nonlinear generalized Langevin equations." *Journal of Statistical Physics*, **9**(3):215–220, 1973.

[Zwa01] Robert Zwanzig. *Nonequilibrium statistical mechanics.* Oxford university press, 2001.