Computational Inversion with Wasserstein Distances and Neural Network Induced Loss Functions

Wen Ding

Submitted in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy
under the Executive Committee
of the Graduate School of Arts and Sciences

COLUMBIA UNIVERSITY

2022

Abstract

Computational Inversion with Wasserstein Distances and Neural Network Induced Loss Functions

Wen Ding

This thesis presents a systematic computational investigation of loss functions in solving inverse problems of partial differential equations. The primary efforts are spent on understanding optimization-based computational inversion with loss functions defined with the Wasserstein metrics and with deep learning models. The scientific contributions of the thesis can be summarized in two directions.

In the first part of this thesis, we investigate the general impacts of different Wasserstein metrics and the properties of the approximate solutions to inverse problems obtained by minimizing loss functions based on such metrics. We contrast the results to those of classical computational inversion with loss functions based on the L^2 and $\mathcal{H}^{-1}$ metric. We identify critical parameters, both in the metrics and the inverse problems to be solved, that control the performance of the reconstruction algorithms. We highlight the frequency disparity in the reconstructions with the Wasserstein metrics as well as its consequences, for instance, the pre-conditioning effect, the robustness against high-frequency noise, and the loss of resolution when data used contain random noise. We examine the impact of mass unbalance and conduct a comparative study on the differences and important factors of various unbalanced Wasserstein metrics.

In the second part of the thesis, we propose loss functions formed on a novel offline-online computational strategy for coupling classical least-square computational inversion with modern deep learning approaches for full waveform inversion (FWI) to achieve advantages that can not be achieved with only one component. In a nutshell, we develop an offline learning strategy to construct a robust approximation to the inverse operator and utilize it to produce a viable initial guess and design a new loss function for the online inversion with a new dataset. We demonstrate through both theoretical analysis and numerical simulations that our neural network induced loss functions developed by the coupling strategy improve the loss landscape as well as computational

efficiency of FWI with reliable offline training on moderate computational resources in terms of both the size of the training dataset and the computational cost needed.

Table of Contents

List of Tables

List of Figures

Acknowledgements

First and foremost, I am incredibly grateful to my supervisors, Professor Qiang Du and Professor Kui Ren for their invaluable advice, unwavering support, and patience during my Ph.D. study. Their immense knowledge, plentiful experience, sharp insight and personal charisma have encouraged me all the time in my academic research and daily life. Their guidance benefited me all across my research career and writing of this thesis. I would also like to express my most profound appreciation to Professor Kyle Mandli, Professor Amir Sagiv, Professor Chris Wiggins, Professor Lu Zhang, and Professor Changxi Zheng for serving on my oral exam, thesis proposal and thesis defense committees. I have learned a lot from them during my time at Columbia. I will never forget the delightful discussions with Dr. Kuang Huang, Dr. Rachael Williams, and Professor Lu Zhang. It is their technical support and encouragement that inspired me. Some of the discussions of this thesis come from lecture materials from Prof. Kyle Mandli's Numerical Analysis class and Prof. Michael K. Tippett's Uncertainty Quantification class. I would like to take the chance to thank all the members of the Applied Math and Applied Physics department at Columbia University in the City of New York. Their kind help and support have made my study and life at Columbia University a wonderful time. Last but not least, I would like to thank my parents and my friends sincerely. This endeavor would have been impossible for me without their tremendous understanding and encouragement in the past few years.

The research in this work was partially supported by funding provided by the National Science Foundation through grants DMS-1620473, DMS-1913309, and EAR-2000850. Part of the numerical simulations presented in this thesis is performed on Columbia's Ginsburg HPC system (also partially supported by the National Science Foundation). These supports are greatly acknowledged.

Chapter 1: Introduction

In this introductory chapter, we provide some background on the main topics of the thesis. In particular, we give a brief review of classical loss functions commonly used in computational inverse problems. We highlight the fundamental properties of these loss functions and their impact on the solutions to the inverse problems.

1.1 Motivation

In the absence of analytical inversion formulas, most model-based inverse problems are solved by computational minimization algorithms that minimize the mismatch between model predictions and measured data. The choice of the objective function, often called the loss function, for the minimization problem is of critical importance. While an ideal loss function is a convex one, it is often impossible to have such loss functions for general nonlinear inverse problems. Loss functions with better convexity landscapes than others can significantly improve the optimization process when solving inverse problems.

The main objective of this thesis is to understand the effect of different loss functions on the reconstruction results for computational inverse problems, to propose numerical methods for reshaping the loss landscape, and thus improve the solutions of the inverse problems in several aspects. Our primary efforts will be the study of those loss functions based on the Wasserstein metrics that have attracted significant attention in the computational inverse problems community in recent years [3, 84, 74, 104, 40, 62, 85]. In addition, we will develop a deep-learning based approach to design problem-specific loss functions for inverse problems. This is a direction that has been extensively studied in recent years; see for instance [82, 97, 106, 69, 116] and references therein for more details.

1.2 Computational inverse problems

Numerical solution of linear and nonlinear inverse problems is a challenging task that is needed in many areas of science and engineering. To be specific, let us formulate an abstract inverse problem in the infinite-dimensional setting as the general model of the inverse coefficient problems we see in the rest of the thesis. Let $\Omega \subset \mathbb{R}^d$ be a bounded domain with smooth boundary $\partial\Omega$, S be a Banach space of real-valued functions defined on Ω, and $\theta : \Omega \mapsto \mathbb{R}$ be the target function on Ω that we are interested in reconstructing from some datum $\mathbf{g} \in Y$. We denote by $\mathbf{f}(\theta) : S \mapsto Y$ the forward model that takes θ to our measurement, that is

$$\mathbf{f}(\theta) = \mathbf{g}. \tag{1.1}$$

The objective of the inverse problem is to solve this equation to find θ. The property of the inverse problem depends on many factors: the property of the forward operator $\mathbf{f}$, the *a priori* property of the unknown θ, and the quality of the observed data $\mathbf{g}$ (for instance the noise in the data). The primary computational strategy, in the absence of explicit/semi-explicit reconstruction methods, is the one searching for the inverse solution by minimizing the mismatch between model predictions and observed data [139]:

$$\inf_{\theta \in S} L(\mathbf{f}(\theta), \mathbf{g}) + \beta \mathcal{R}(\theta)\,, \tag{1.2}$$

where the loss function $L : Y \times Y \mapsto \mathbb{R}_+$ is a non-negative function of evaluating the mismatch between the algorithm models and the dataset, and the term $\beta\mathcal{R}(\theta)$ is the regularization term that imposes some *a priori* constraints on the unknown θ to be reconstructed. The classical L^2 least-squares loss function performs the reconstruction by searching target quantity θ that minimizes the square difference $\|\mathbf{f}(\theta) - \mathbf{g}\|^2_{L^2(Y)}$. While the classical L^2 least-squares method has been extremely successful in many aspects, there have been great efforts in recent years to search for techniques to overcome some of its disadvantages in certain applications. An extremely successful development is the invention of methods that promote sparsity, in appropriate bases, in inverse

solutions for applications in signal and image processing where sparse structures are important; see for instance [54] and references therein for some recent overviews. The sparsity of solutions could be forced either through regularization strategies, such as methods based on total variation regularization in image processing [140], or through the mismatch functional, such as those based on L^1 and related functionals in compressive sensing [52, 45].

There are many important factors that decide the quality of this optimization based solution procedure for inverse problems, among which are the quality of the loss function, the regularization mechanism (i.e. the functional $\mathcal{R}(\theta)$), and the optimization algorithm used are three critical ones.

1.3 Loss landscape for inversion

The loss landscape associated with the inverse problem is the mismatch as a function of the target quantities θ

$$ls(\theta) = L(\mathbf{f}(\theta), \mathbf{g}) \,. \tag{1.3}$$

The loss landscape is an insightful way of visualizing the optimization problem and potentially offering more information about the inverse problem itself and why the loss function works. Readers can find more fascinating results of visualizing loss landscape in the context of deep learning in [83, 114, 131, 142, 50, 90, 123, 51, 56, 29, 55]. Let us emphasize that for most of the inverse problems we are interested in, the unknown θ is a function. Therefore it is impossible to visualize the loss function $ls(\theta)$ the same way as in the low-dimensional problems.

Ideally, we would prefer the loss landscape function to be convex. In such a case, every local minimum is global minimum, the optimal set is convex. The problem has at most one optimal point when the loss landscape is strictly convex. These powerful properties present valuable tools for developing accurate and efficient algorithms for solving optimization problem. Indeed, there are extremely fast algorithms specifically designed for convex problems; see [18] for a thorough overview in this direction.

In many applications, the forward operator and data have the particular forms:

$$\begin{aligned} \mathbf{f}(\theta) &= (f(\theta; h_i))_{i=1}^N \\ \mathbf{g} &= (g_i)_{i=1}^N \end{aligned} \tag{1.4}$$

where $f(\theta; h) : h \mapsto g$ is a model parameterized by θ. In this case, we have N pairs of input-output data for the model f: $\{(h_i, g_i)\}_{i=1}^N$, corresponding to N different observations of the model f. An example of such a forward model is a neural network that takes h as input and generate g as the output, with θ being the trainable parameters for the neural network. In this case, if we denote $l(\cdot, \cdot)$ as $L(\cdot, \cdot)$ operates on a single data point, then

$$L(\mathbf{f}(\theta), \mathbf{g}) = \sum_{i=1}^N l(f(\theta; h_i), g_i)\,. \tag{1.5}$$

This is the most common data framework for supervised learning problems in the machine learning community and is also the standard framework for inverse problems with multiple datasets.

1.4 Typical loss functions

We now briefly review several classical loss functions commonly used in computational inversion and learning. We explain those loss functions on linear inverse problems (in either finite- or infinite-dimensional settings). More specifically, we define

$$\mathbf{f}(\theta) = X\theta, \tag{1.6}$$

where $X \in \mathbb{R}^{n,d}$ is the forward operator and $\theta \in \mathbb{R}^d$ is the known. We assume that the data we have is polluted by random noise. The inverse problem is then to reconstruct θ in

$$\mathbf{g} = X\theta + \epsilon,$$

where ϵ is the noise. This is known as the linear regression problem. In this optimization framework, linear regression is essentially solving the following optimization problem:

$$\min_{\theta} L(X\theta, \mathbf{g})\,. \tag{1.7}$$

If we introduce the notations as in (1.4), that is, $X = \begin{pmatrix} x_1 \\ \dots \\ x_N \end{pmatrix} \in \mathbb{R}^{N,d}$, and $\mathbf{g} = \begin{pmatrix} g_1 \\ \dots \\ g_N \end{pmatrix} \in \mathbb{R}^N$, this problem can be rewritten into a typical linear regression form

$$\min_{\theta} \sum_{i=1}^{N} l(x_i\theta, g_i)\,. \tag{1.8}$$

1.4.1 The L^2 loss function

One of the most popular loss functions is the L^2 loss function given by [139]:

$$\ell_2^2(f, g) := \sum_{i=1}^{N} (f_i - g_i)^2, \quad f = (f_1, \cdots, f_N)^T, \quad g = (g_1, \cdots, g_N)^T\,, \tag{1.9}$$

in the finite-dimensional case.

The linear regression problem, in the absence of regularization, under the L^2 loss is therefore:

$$\min_{\theta \in \mathbb{R}^d} \|X\theta - \mathbf{g}\|_2^2\,. \tag{1.10}$$

This landscape function of least-square problem can be written more explicitly as

$$ls(\theta) = (X\theta - \mathbf{g}, X\theta - \mathbf{g}) = \theta^T X^T X\theta - 2\mathbf{g}^T X\theta + \mathbf{g}^T\mathbf{g}\,. \tag{1.11}$$

It is a quadratic function of θ, therefore convex. It is well-known that the solution of least square problem is $\widehat{\theta}_{LS} = (X^T X)^{-1} X^T \mathbf{g}$, where $(X^T X)^{-1}$ here represents the pseudo-inverse of $X^T X$. Let us

assume that the singular value decomposition of X is $X = USV^T$. We then have

$$\begin{aligned}\widehat{\theta}_{LS} &= (VSU^T USV^T)^{-1} VSU^T \mathbf{g} \\ &= VS^{-1}U^T \mathbf{g} \\ &= VS^{-1}U^T (X\theta + \epsilon) \\ &= \theta + VS^{-1}U^T \epsilon \,.\end{aligned}$$

Therefore, the mean square error of the least square estimate is

$$\begin{aligned}MSE(\widehat{\theta}_{LS}) &= \|\widehat{\theta} - \theta_{exact}\|_2 \\ &= US^{-2}V^T \|\epsilon\|_2 \\ &= \sum_{i=1}^{r} \frac{1}{s_i^2} u_i v_i^T \|\epsilon\|_2 \,.\end{aligned}$$

When $s_i \to 0_+$, $\dfrac{1}{s_i^2} \to +\infty$. Therefore intuitively speaking, the mean square error of the least square solution is large when X has small singular values $s_i \sim 0_+$.

1.4.2 The L^1 loss function

The L^1 loss function is another commonly used loss function. It is defined, in the finite-dimensional case, as

$$\ell_1(f, g) = \sum_{i=1}^{N} |f_i - g_i|, \quad f = (f_1, \cdots, f_N)^T, \quad g = (g_1, \cdots, g_N)^T \,, \tag{1.12}$$

The inversion based on the L^1 loss is often termed as the least absolute deviation(LAD) estimator. It is achieved by minimizing the landscape function

$$ls(\theta) = \|X\theta - \mathbf{g}\|_1 = \sum_{i=1}^{N} |x_i^T \theta - g_i| \,. \tag{1.13}$$

As a consequence, the loss landscape of LAD is the sum of absolute functions, which is convex, hence the landscape for least absolute deviation is also convex. Moreover, the loss landscape is also a piecewise linear function.

We can formally analyze the landscape function by looking at its derivative at a given $\widehat{\theta}$, assuming that $g_i - \sum_j X_{ij}\widehat{\theta}_j \neq 0$. This leads to [25],

$$\frac{\partial ls(\theta)}{\partial \theta_k}(\widehat{\theta}) = \sum_i \frac{g_i - \sum_j X_{ij}\widehat{\theta}_j}{|g_i - \sum_j X_{ij}\widehat{\theta}_j|}(-X_{ik}) \,. \tag{1.14}$$

In order to find the optimal values, we set $\frac{\partial ls(\theta)}{\partial \theta_k}(\widehat{\theta}) = 0$. With a little rearrangement of terms, we have

$$\sum_{i,j} \frac{X_{ij}X_{ik}\widehat{\theta}_j}{e_i(\widehat{\theta})} = \sum_i \frac{X_{ik}g_i}{e_i(\widehat{\theta})} \,, \tag{1.15}$$

where $e_i(\widehat{\theta}) = |g_i - \sum_j X_{ij}\widehat{\theta}_j|$ is the absolute deviation on data g_i. Therefore the solution of least absolute deviation satisfies

$$X^T E(\widehat{\theta}) X\widehat{\theta} = X^T E(\widehat{\theta})\mathbf{g} \tag{1.16}$$

where $E = diag^{-1}(e)$. Hence LAD can be viewed as a pre-conditioned least-square estimate, with the pre-conditioning matrix $X^T E(\widehat{\theta})$ depending on the current state $\widehat{\theta}$. This leads to a nonlinear, instead of linear, system of equations. Compared to the L^2 based least-square estimator, LAD puts a higher weight on items with smaller absolute deviation due to the nonlinear term E, which creates more "incentives" for small errors to decrease down to 0. In this way, it promotes sparsity in the solution of the problem.

1.4.3 Kernel loss functions

By kernel loss here we mean weighted L^2 loss functions of the form

$$L^2_K(f, g) := \sum_{i=1}^{N} (f_i - \sum_j K_{ij} g_j)^2 \tag{1.17}$$

where the positive semi-definite matrix K is the kernel matrix. When $K = I$ is the identity matrix, kernel losses reduce to the L^2 loss. We refer interested readers to [15] and references therein for more discussions on kernel losses.

The minimization problem for linear regression under the kernel loss is of the form

$$\min_{\theta \in \mathbb{R}^d} (X\theta - \mathbf{g}, K(X\theta - \mathbf{g})) \,. \tag{1.18}$$

Following the same procedure above, the landscape function for kernel loss can be written as

$$ls(\theta) = \theta^T X^T K X \theta - 2\mathbf{g}^T K X \theta + \mathbf{g}^T K \mathbf{g} \tag{1.19}$$

This landscape function is still quadratic. In fact, if we define V as the square root of the positive definite matrix K, that is $K = V^T V$, then the kernel loss estimate is the equivalent to solving the pre-conditioned least square problem:

$$\min_{\theta \in \mathbb{R}^d} \|VX\theta - V\mathbf{g}\|_2^2 \,. \tag{1.20}$$

Here V serves as the pre-conditioner. Depending on the property of K, the problem may behave differently from the classical least square problem. This form of weighted optimization is discussed extensively in the literature; see for instance [39] and references therein.

1.4.4 The $\mathcal{H}^{-1}$ loss function

The last example of the loss function we review here is closely related to the main topic of the thesis: loss functions based on the quadratic Wasserstein metrics. It is the $\mathcal{H}^{-1}$ loss function. As we will see later, it can also be viewed as a special case of kernel loss.

For any bounded smooth domain $\Omega \in \mathbb{R}^d$, we define the space of functions $\mathcal{H}^{-1}(\Omega)$ as the dual

of usual Hilbert space $\mathcal{H}_0^1(\Omega)$. The $\mathcal{H}^{-1}$ norm of a function f is defined as

$$\|f\|_{\mathcal{H}^{-1}(\Omega)} = \sup\{\langle f, u\rangle | u \in H_0^1(\Omega), \|u\|_{H_0^1(\Omega)} \leq 1\} \tag{1.21}$$

It can be shown that $\mathcal{H}^{-1}$ norm of f can be computed by solving a Laplacian equation [43]. More precisely, let

$$\|f\|_{\mathcal{H}^{-1}(\Omega)} = \|\nabla u\|_{L^2(\Omega)}, \quad \Delta u = f, \ \text{ in } \ \Omega \quad u = 0, \ \text{ on } \ \partial\Omega.$$

To illustrate the main properties of the $\mathcal{H}^{-1}$ loss, let us use the linear regression problem for a function θ in the one-dimensional case. The minimization problem can be written as

$$\begin{aligned} &\min_{\theta} \|u\|_2 \\ &L_{\Delta x} u = X\theta - \mathbf{g} \end{aligned} \tag{1.22}$$

where the matrix $L_{\Delta x}$ is the discretization of the one-dimensional Laplacian operator with zero boundary condition, that is

$$L_{\Delta x} = \frac{1}{\Delta x^2} \begin{pmatrix} 2 & -1 & 0 & \dots & 0 \\ -1 & 2 & -1 & \dots & 0 \\ \dots & \dots & \dots & \dots & \dots \\ 0 & \dots & -1 & 2 & -1 \\ 0 & 0 & \dots & -1 & 2 \end{pmatrix}_{n,n}$$

Let $\widehat{G}$ be the Green's function for the Laplacian operator in one dimension. Then $\widehat{G}$ reads [42]:

$$\widehat{G}(x, x_0) = \begin{cases} (1 - x_0)x, & for\ x \leq x_0 \\ x_0(1 - x), & for\ x > x_0 \end{cases} \tag{1.23}$$

Let us define $G_{ij} = \Delta x \widehat{G}(i\Delta x, j\Delta x)$, where $(n + 1)\Delta x = 1$. We can then show the following result straightforwardly.

Proposition 1.4.1. $GL_{\Delta x} = I$.

Proof. The result follows from direct algebraic calculations. We first verify that

$$2G_{i,j} - G_{i-1,j} - G_{i+1,j} = \tag{1.24}$$
$$\begin{cases} 2(1-j\Delta x)i\Delta x^2 - (1-j\Delta x)(i-1)\Delta x^2 - (1-j\Delta x)(i+1)\Delta x^2 = 0, 2 < i < j < n \\ 2(1-i\Delta x)i\Delta x^2 - (i-1)\Delta x^2(1-i\Delta x) - i\Delta x^2(1-(i+1)\Delta x) = \Delta x^2, 1 < i = j < n \\ 2(1-i\Delta x)j\Delta x^2 - (1-i\Delta x)(j-1)\Delta x^2 - (1-i\Delta x)(j+1)\Delta x^2 = 0, 1 < j < i < n \end{cases}$$

We then check:

$$2G_{1,i} - G_{2,i} = \begin{cases} 2\Delta x^2(1-\Delta x) - \Delta x^2(1-2\Delta x) = \Delta x^2, i = 1 \\ 2\Delta x^2(1-i\Delta x) - 2\Delta x^2(1-i\Delta x) = 0, i > 1 \end{cases} \tag{1.25}$$

In the same manner, we can verify that

$$2G_{n,i} - G_{n-1,i} = \begin{cases} 2n\Delta x^2(1-n\Delta x) - (n-1)\Delta x^2(1-n\Delta x) = \Delta x^2, i = n \\ = 2i\Delta x^2(1-n\Delta x) - i\Delta x^2(1-(n-1)\Delta x) = 0, i < n \end{cases} \tag{1.26}$$

Combining the above formulas together, we have $GL_{\Delta x} = I$. This finishes the proof. □

The above standard result shows that G is the inverse of $L_{\Delta x}$. Therefore, the linear regression with $\mathcal{H}^{-1}$ loss is equivalent to the minimization problem

$$\min_{\theta} \|GX\theta - G\mathbf{g}\|_2^2 \tag{1.27}$$

which is a pre-conditioned problem to the original least square problem $X\theta = \mathbf{g}$. In other words, using $\mathcal{H}^{-1}$ loss for linear regression is equivalent to using L^2 loss for the new linear regression problem:

$$GX\theta = G\mathbf{g} \tag{1.28}$$

This also coincides with using the kernel loss for the original linear problem with the kernel matrix

$K = G^2$.

It is also easy to verify that the L^2 estimate of the inverse problem is also the optimal solution of the $\mathcal{H}^{-1}$ estimate.

Proposition 1.4.2. *$\widehat{\theta}_{LS}$ is the solution of* (1.28)

Proof. This can be proved using pseudo-inverse. Denote X^+ as the pseudo-inverse of X, then $\widehat{\theta}_{LS} = X^+\mathbf{g}$. The solution for (1.28) is $\widehat{\theta}_{\mathcal{H}^{-1}} = (GX)^+G\mathbf{g} = X^+G^+G\mathbf{g} = X^+\mathbf{g} = \widehat{\theta}_{LS}$, since G has full rank. □

The spectral property of the matrix $L_{\Delta x}$ is well-known. We summarize this in the following proposition.

Proposition 1.4.3. *The matrix $L_{\Delta x}$ has eigenvalue $\lambda_p = \frac{2}{\Delta x^2}(1 - cos(p\pi\Delta x))$, $p = 1, \cdots, n$, with corresponding eigenvector $u^p := (sin(p\pi j\Delta x))_{j=1}^n$*

Proof. Straightforward algebraic calculations show that

$$\begin{aligned}
&\Delta x^2 L_{\Delta x} u_1^p = 2sin(p\pi\Delta x) - sin(p\pi 2\Delta x) \\
&= 2sin(p\pi\Delta x) - 2sin(p\pi\Delta x)cos(p\pi\Delta x) \\
&= 2sin(p\pi\Delta x)(1 - cos(p\pi\Delta x)) \\
&= (1 - cos(p\pi\Delta x))u_1^p
\end{aligned} \tag{1.29}$$

Moreover, we have

$$\begin{aligned}
&\Delta x^2 L_{\Delta x} u_n^p = 2sin(p\pi n\Delta x) - sin(p\pi(n-1)\Delta x) \\
&= 2sin(p\pi n\Delta x) - sin(p\pi(n-1)\Delta x) - sin(p\pi(n+1)\Delta x) \\
&= 2sin(pn\pi\Delta x) - 2sin(p\pi n\Delta x)cos(p\pi\Delta x) \\
&= 2sin(p\pi n\Delta x)(1 - cos(p\pi\Delta x)) \\
&= (1 - cos(p\pi\Delta x))u_n^p
\end{aligned} \tag{1.30}$$

The last step is to show that

$$\begin{aligned}
&\Delta x^2 L_{\Delta x} u_j^p = 2sin(p\pi j\Delta x) - sin(p\pi(j-1)\Delta x) - sin(p\pi(j+1)\Delta x) \\
&= 2sin(p\pi j\Delta x) - 2sin(p\pi j\Delta x)cos(p\pi\Delta x) \\
&= 2sin(p\pi j\Delta x)(1 - cos(p\pi\Delta x)) \\
&= (1 - cos(p\pi\Delta x))u_j^p
\end{aligned} \tag{1.31}$$

The conclusion therefore follows. □

The smallest eigenvalue of $L_{\Delta x}$ is

$$\begin{aligned}
&\lambda_1 = \tfrac{2}{\Delta x^2}(1 - cos(\pi\Delta x)) \\
&= \tfrac{2}{\Delta x^2}(\tfrac{1}{2}\pi^2\Delta x^2 + O(\Delta x^4)) \\
&= \pi^2 + O(\Delta x^2)
\end{aligned} \tag{1.32}$$

This is clearly bounded away from zero as $\Delta x \to 0$. Therefore the greatest eigenvalue of $G = L_{\Delta x}^{-1}$ is bounded, say $s(G) \leq \frac{1}{\pi^2 - 1}$ when n is sufficiently large.

One heuristic understanding of $L_{\Delta x}$ operator is that it takes the derivative twice, while the inverse operator G should behave similarly to integrating twice, increasing the smoothing property of data. The kernel G will therefore smooth out the noise in the data. In particular, the high-frequency components of the noise are damped more than the lower-frequency components. More specifically, suppose $\mathbf{g} = X\theta_{exact} + \epsilon$, where ϵ is noise, then the landscape functions for $\mathcal{H}^{-1}$ and least square estimate are respectively

$$ls(\theta) = (X\theta - \mathbf{g}, X\theta - \mathbf{g}) = (\theta_{exact} - \theta)^T X^T X(\theta_{exact} - \theta) - 2(\theta_{exact} - \theta)^T X^T \epsilon + \epsilon^T \epsilon\,, \tag{1.33}$$

and

$$ls(\theta) = (GX\theta - G\mathbf{g}, GX\theta - G\mathbf{g}) = (\theta_{exact} - \theta)^T (GX)^T GX(\theta_{exact} - \theta) - 2(\theta_{exact} - \theta)^T (GX)^T \epsilon + \widetilde{\epsilon}^T \widetilde{\epsilon}\,, \tag{1.34}$$

where $\widetilde{\epsilon} = G\epsilon$. The convexity of the two loss landscapes can be seen from the fact that the corresponding Hessians are both positive semidefinite:

$$\begin{aligned} Hessian(ls_{L^2}) &= X^T X\,, \\ Hessian(ls_{\mathcal{H}^{-1}}) &= X^T G^T G X\,. \end{aligned} \tag{1.35}$$

The following theorem implies that the singular value of $Hessian(ls_{\mathcal{H}^{-1}})$ is strictly less than $Hessian(ls_{L^2})$, i.e. the loss landscape of $ls_{\mathcal{H}^{-1}}$ is strictly "flatter" than that of ls_{L^2}.

Theorem 1.4.4. *Let $\sigma_i(X)$ be the i_{th} largest singular value for X. Then $\sigma_i(GX) < \frac{1}{\pi^2-1}\sigma_i(X)$ when n is sufficiently large.*

Proof. By (1.32), we have that $\sigma_1(G) < \frac{1}{\pi^2-1}$ when n is sufficiently large. The result then follows from the Min-Max principle for singular values:

$$\begin{aligned} \sigma_i(GX) &= \max_{U:dim(U)=i} \min_{u\in U,\|u\|_2=1} \|GXu\|_2 \\ &\leq \max_{U:dim(U)=i} \min_{u\in U,\|u\|_2=1} \|G\|_2\|Xu\|_2 \\ &= \sigma_1(G) \max_{U:dim(U)=i} \min_{u\in U,\|u\|_2=1} \|Xu\|_2 \\ &= \sigma_1(G)\sigma_i(X) \\ &< \frac{1}{\pi^2-1}\sigma_i(X)\,. \end{aligned}$$

□

Moreover, it is straightforward to see that the value of the loss function at the exact solution $\theta_{exact} = (X^T X)^{-1} X^T y$ are respectively:

$$\begin{aligned} ls_{L^2}(\theta_{exact}) &= \|\epsilon\|_2^2 \\ ls_{\mathcal{H}^{-1}}(\theta_{exact}) &= \|G\epsilon\|_2^2 = \|\tilde{\epsilon}\|_2^2\,. \end{aligned} \tag{1.36}$$

Here one expects $ls_{\mathcal{H}^{-1}}(\theta_{exact}) \ll ls_{L^2}(\theta_{exact})$ because the effect of G is similar to "taking average"

with respect to some moderate weight G. To be more specific, the following theorem shows that the impact of error under $\mathcal{H}^{-1}$ is approximately reduced to $O(\sqrt{\Delta x})\epsilon$, when noise is in effect.

Theorem 1.4.5. *Assume ϵ is i.i.d. with $E(\epsilon_i) = 0, Var(\epsilon_i) = \sigma^2$, and $G \in \mathbb{R}^{n\times n}, G_{ij} = \Delta x \widehat{G}(x_i, x_j)$ where $x_i = i\Delta x$. Let $\tilde{\epsilon} = G\epsilon$. Then $E\|\tilde{\epsilon}\|_2^2 = O(\Delta x)E\|\epsilon\|_2^2$*

Proof. This is the direct consequence of the following algebraic calculations.

$$
\begin{aligned}
& E(\|\tilde{\epsilon}\|_2^2) \\
&= E(\epsilon^T G^T G \epsilon) \\
&= E(\sum_{i,j,k} G_{ki} G_{kj} \epsilon_i \epsilon_j) \\
&= \sigma^2 \sum_{i,k} G_{ki} G_{ki} \\
&= \sigma^2 (\sum_{i=1}^{n} G_{ii} + 2 \sum_{1=i<k=n} G_{ik}) \\
&= \tfrac{\sigma^2}{(n+1)^4} (\sum_{i=1}^{n} i(n+1-i) + 2 \sum_{i=1}^{n-1} \sum_{k=i+1}^{n} i(n+1-k)) \\
&= \tfrac{n(2+n)}{12(1+n)^2} n\sigma^2 \\
&= \tfrac{(2+n)}{12(1+n)^2} E(\|\epsilon\|_2^2) \\
&= O(\tfrac{1}{n}) E(\|\epsilon\|_2^2) \\
&= O(\Delta x) E(\|\epsilon\|_2^2)
\end{aligned}
\tag{1.37}
$$

□

Remark 1.4.6. *Therefore, even though the exact solution of $\mathcal{H}^{-1}$ estimate is identical to that of the classical L^2 least-square solution, the effect of noise in the $\mathcal{H}^{-1}$ loss is $\tilde{\epsilon} \ll \epsilon$, which is much smaller than that of the L^2 loss (which is ϵ). This fact has significant consequences in the implementation of the optimization algorithm to solve the minimization problems. If we use the value of the loss function as the stopping criteria, for instance use some fixed positive value δ as the threshold, then for some ϵ, $G\epsilon$ could be very small such that $\|G\epsilon\|_2 < \delta < \|\epsilon\|_2$. This means that the same optimization algorithm will continue to run for the L^2 least-square, but will stop for the minimization problem based on the $\mathcal{H}^{-1}$ loss. Therefore inversion based on the $\mathcal{H}^{-1}$ loss stops*

prematurely, leading to less accurate final inversion result in this situation. We shall see this effect again in Chapter 3 with Wasserstein metrics.

While the rescaling of the problem by G in the above analysis seems artificial as the rescaling does not change the true solution of the inversion result in the absence of noise in the data used for inversion, it has a non-trivial impact in the case when the optimization is not solved perfectly, that is, when the loss function is not minimized to zero, or when noisy data are used in the inversion. This can be seen in a more precise analysis of the inversion process in the Fourier domain. Let us recall that the $\mathcal{H}^{-1}$ norm can be defined in the Fourier domain as [42]:

$$\|f\|_{\mathcal{H}^{-1}(\mathbb{R}^n)} = \left(\int_{\mathbb{R}^n} |\widehat{f}(\xi)|^2 (1+\xi^2)^{-1} d\xi\right)^{\frac{1}{2}}, \tag{1.38}$$

where $\widehat{f}$ is the Fourier transform of f. This definition shows that $\mathcal{H}^{-1}$ norm attaches a damping factor $(1+\xi^2)^{-1}$ to each Fourier mode $\widehat{f}(\xi)$. Therefore the high-frequency components are reduced by factor $\sim O(\xi^{-2})$. The higher the frequency is, the stronger the damping effect will be. We shall see this low frequency preserving and high-frequency damping effect in the numerical experiments as a benchmark to the Wasserstein metrics in the next chapter.

Theorem 1.4.7. *Let* $\epsilon = (\sum_{k=-\infty}^{\infty} A_k e^{j2\pi k x_i})_{i=1}^n$, *then* $G\epsilon = (\sum_{k=-\infty}^{\infty} O(\frac{1}{k^2}) A_k e^{j2\pi k x_i})_{i=1}^n$.

Proof. Let $T(k) = (e^{j2\pi k x_i})_{i=1}^n$, where $x_i = i\Delta x$, then we have

$$\begin{aligned}
(L_{\Delta x} T)_i &= \tfrac{1}{\Delta x^2} 2e^{j2\pi k x_i} - e^{j2\pi k(x_i+\Delta x)} - e^{j2\pi k(x_i-\Delta x)} \\
&= \tfrac{1}{\Delta x^2} e^{j2\pi k x_i}(2 - e^{j2\pi k\Delta x} - e^{-j2\pi k\Delta x}) \\
&= \tfrac{1}{\Delta x^2} e^{j2\pi k x_i}(2 - cos(2\pi k\Delta x)) \\
&= O(k^2) T_i
\end{aligned} \tag{1.39}$$

Therefore, we have $L_{\Delta x} T \sim O(k^2) T$. This then implies that $GT = O(\frac{1}{k^2}) T$. The result then follows directly. □

This result shows that the impact of the pre-conditioner G is to create a weight $O(\frac{1}{k^2})$ according

to frequency k of the Fourier mode of the noise. The factor $O(\frac{1}{k^2})$ makes the impact of high-frequency component of ϵ very small (but not zero). Therefore, the corresponding optimization process becomes more stable against high-frequency noise. If we take ϵ as the residual, that is, the mismatch between $X\theta$ and $\mathbf{g}$, it is clear that the minimization of the $\mathcal{H}^{-1}$ loss function will favor the low-frequency components of the unknown θ (because the high-frequency components of $X\theta - \mathbf{g}$ are suppressed with the factor $O(\frac{1}{k^2})$). Therefore, the inversion result, when the optimization is stopped at a given tolerance of the loss function value δ, looks smoother than the corresponding one from the regular L^2 least-square inversion. This phenomenon was analyzed in more detail in [38].

1.5 Contribution and outline of thesis

In the rest of the thesis, we study in detail two types of loss functions for computational inversion. The main contributions of the thesis are summarized in Chapter 3 and Chapter 4.

In Chapter 3, we performed a detailed computational study on inversion with loss functions constructed from various quadratic Wasserstein metrics. We analyze the property of the solutions to the inverse problem with such loss functions. In particular, we compare in detail the results with those of classical computational inversion with loss functions based on the L^2 and $\mathcal{H}^{-1}$ metrics. We demonstrate the frequency disparity in the reconstructions with the Wasserstein metrics as well as its consequences. Our main contributions lie in the systematic investigation of such loss functions in the setup of computational inversion where resolution and stability compete with each other. The result of is chapter is documented in [33].

In Chapter 4, we propose a loss function based on deep learning for computational inversion. In a nutshell, to invert $\mathbf{f}(\theta) = \mathbf{g}$, we train a neural network $\widehat{\mathbf{f}}_\alpha^{-1}$ (where α denoting the set of trainable parameters) to approximate the inverse operator $\mathbf{f}^{-1}$. This approximate inverse is trained to focus on the stable part of the inverse operator $\mathbf{f}$. We then use this trained approximation to form a new loss function for the inverse problem:

$$ls_{NN}(\theta) = \|\widehat{\mathbf{f}}_\alpha^{-1}(\mathbf{f}(\theta)) - \widehat{\mathbf{f}}_\alpha^{-1}(\mathbf{g})\|_2^2 . \tag{1.40}$$

The main benefit of this new loss function is that it has better convexity than its counterpart without the operator $\widehat{\mathbf{f}}_{\alpha}^{-1}$. We demonstrate with numerical simulations the feasibility of such a strategy in solving complex inverse problems such as the full waveform inversion problem in ultrasound and geophysical inversion. The result of this chapter is summarized in reference [34].

Chapter 2: Review of Wasserstein Distances

As a preparation for the studies in the next chapter, we briefly review Wasserstein distances and their properties in this chapter. Most of the introduction in this chapter can be found in standard literature on optimal transport theory, especially the monograph [135].

2.1 Basic definitions

Wasserstein distance of two probability measure μ, ν, defined on metric spaces X, Y respectively, is defined by Kantorovish's optimal transportation problem

$$\inf_{\pi \in \Pi(\nu,\mu)} \int_{X \times Y} c(x,y) d\pi(x,y)\,, \tag{2.1}$$

where $\Pi(\mu,\nu) = \{\pi \in P(X \times Y); \pi(A \times Y) = \mu(A), \pi(X \times B) = \nu(B)\}$

Intuitively, one can think of $\pi(x,y)$ as a transportation plan, moving mass from density μ to density ν. Then Kantorovish's optimal transportation problem is that under the cost of $c(x,y)$, what is the lowest cost plan.

In the case where $X = Y$ is a complete separable metric space with metric d, and $c = d^p$, the above definition reduces to

$$W_p(\mu,\nu) = \inf_{\pi \in \Pi(\nu,\mu)} \left(\int_{X \times Y} d^p(x,y) d\pi(x,y)\right)^{\frac{1}{p}}, 1 \leq p < \infty \tag{2.2}$$

and

$$W_\infty(\mu,\nu) = \inf_{\pi \in \Pi(\nu,\mu)} \sup_{(x,y) \in supp(\pi)} d(x,y)\,. \tag{2.3}$$

When $p = 1$, we call it W_1 distance, and when $p = 2$, we call it W_2 distance (which is often called the quadratic Wasserstein distance).

Monge's problem. Monge had another formulation of the optimal transport problem:

$$\inf_{\nu=T\#\mu} \int_X c(x, T(x)) d\mu(x) \tag{2.4}$$

where $\nu = T\#\mu$ if and only if $T : X \to Y$ is measurable and for any measureable set $A \subset X$, $\mu(A) = \nu(T(A))$, which is equivalent to

$$\int_X \phi(T(x)) d\mu(x) = \int_Y \phi(y) d\nu(y) \quad \forall \phi \in L^1(d\nu) \,. \tag{2.5}$$

Kantorovich's problem can be seen as a relaxed version of Monge's problem. If one restricts $\pi(x, y)$ in Kantorovich's problem to having a special form $\pi_T(x, y)$ such that $d\pi_T(x, y) = d\mu(x)\delta(y = T(x))$, then

$$\int_{X\times Y} c(x, y) d\pi_T(x, y) = \int_X c(x, T(x)) d\mu(x) \,. \tag{2.6}$$

Therefore, Kantorovich's problem reduces to Monge's problem [135].

In summary, Monge's problem is just the same as Kantorovich's except for one thing: it is additionally required that no mass be split. In other words, to each location x is associated with a unique destination $\mathbf{g}$, and T is the transportation plan. In most of the applications, important cases are the Wasserstein distance when $p > 1$. The strict convexity of c guarantees that if μ and ν are absolutely continuous with respect to the Lebesgue measure, then there is a unique solution to the Kantorovich problem, which turns out to be also the solution to the Monge problem [135]. That means Kantorovich's problem shares the same optimal transportation plan as Monge's problem. If the moving cost function $c(x, y)$ has some nice properties, we have the following conclusions. First one defines $c(x, y)$ to be strictly convex if $c(x, y) = d(x - y)$, and $c(x)$ is strictly convex on $\mathbb{R}^n$.

Theorem 2.1.1 (Optimal Transportation Theorem for a Strictly Convex Cost [135])**.** *Let c be a strictly convex, super-linear cost on $\mathbb{R}^n$, and let μ, ν be probability measures on $\mathbb{R}^n$, such that the total transportation cost from μ to ν is not always infinite. Assume moreover that μ is absolutely*

continuous with respect to the Lebesgue measure. Then, there exists a unique optimal transportation plan for the Kantorovich transportation problem, and it has the form $d\pi(x, y) = d\mu(x)\delta(y = T(x))$*, where* T *is uniquely determined* $d\mu$ *almost everywhere by the requirements that* $T\#\mu = \nu$*, and* $T(x) = x - \nabla c^*(\nabla\phi(x))$ *for some c-concave function* ϕ*, where* $c^*(y) = \sup_{x\in\mathbb{R}^n} x \cdot y - c(x)$ *is the Legendre transform.*

Monge's problem is well studied. It has various applications in engineering as well as computer vision. Suppose $d\mu(x) = \mathbf{f}(x)dx, d\nu(x) = \mathbf{g}(x)dx$. The change of variables formula of (2.5) formally leads to the requirement $\mathbf{f}(x) = \mathbf{g}(T(x))det(\nabla T(x))$. It can be proved that the optimality condition of T is equivalent to T being the gradient of some convex function $u(x)$.

Theorem 2.1.2 ([135])**.** *Suppose* $d\mu(x) = \mathbf{f}(x)dx, d\nu(x) = \mathbf{g}(x)dx$*. The squared Wasserstein metric is given by*

$$W_2^2(\mu, \nu) = \int_X \mathbf{f}(x)|x - \nabla u(x)|^2 dx \tag{2.7}$$

where u *is the solution of*

$$\begin{cases} det(D^2u(x)) = \frac{\mathbf{f}(x)}{\mathbf{g}(\nabla u(x))}, \ \ x \in X \\ u \text{ is convex}\,. \end{cases} \tag{2.8}$$

Benamou-Brenier minimization problem. Wasserstein distance can be defined from another point of view: the Benamou-Brenier formula. It offers us the flow dynamic approach to solve Kantorovich's problem. The core idea behind this is the Benamou-Brenier minimization problem:

$$\inf_{(\rho,\boldsymbol{\omega})\in V(\mathbf{f},\mathbf{g})} \frac{1}{T}\int_\Omega \int_0^T \rho(t, x)|\boldsymbol{\omega}(t, x)|^2 dtdx\,, \tag{2.9}$$

where $V(\mathbf{f}, \mathbf{g})$ is the set of all $(\rho, \boldsymbol{\omega})_{0\leq t\leq T}$ such that

$$\begin{aligned}
&\rho \in C([0,T]; w * -P_{ac}(\mathbb{R}^n)) \\
&\boldsymbol{\omega} \in \mathcal{L}^2(d\rho(\cdot, x)dt) \\
&\cup_{0\leq t\leq T} \ supp(\rho(t,\cdot)) \text{ is bounded} \\
&\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho\boldsymbol{\omega}) = 0 \text{ weakly(in the distributional sense)} \\
&\rho(t=0,x) = \mathbf{f}(x) \\
&\rho(t=T,x) = \mathbf{g}(x)\,.
\end{aligned}$$

Here the notation $w * -P_{ac}(\mathbb{R}^n))$ stands for the set of absolutely continuous probability measure $P_{ac}(\mathbb{R}^n)$, endowed with the weak-*topology. Then the Benamou-Brenier theorem combines optimal transport with the Benamou-Brenier minimization problem [135].

Theorem 2.1.3 (Benamou-Brenier formula [135])**.** *Let* $\mathbf{f}, \mathbf{g} \in P_{ac}(\mathbb{R}^n)$ *be compactly supported. Then*

$$W_2(\mathbf{f}, \mathbf{g}) = \inf_{(\rho,v)\in V(\mathbf{f},\mathbf{g})} \frac{1}{T} \int_\Omega \int_0^T \rho(t,x)|\boldsymbol{\omega}(t,x)|^2 dtdx\,. \tag{2.10}$$

2.2 Fundamental properties

We now summarize some of the fundamental properties of the Wasserstein metrics. We focus those that are related to our applications in the next two chapters.

2.2.1 Properties of W_2

We start with the quadratic Wasserstein metric. It has several properties that make it attractive as loss function for solving inverse problems. The following theorems are from [148]. Interested readers can find proofs and more applications in the original paper.

Theorem 2.2.1 (Convexity with respect to shift [148])**.** *Suppose* $\mathbf{f}$ *and* $\mathbf{g}$ *are probability density functions on X of bounded second moment. Let T be the optimal map that* $\mathbf{g}(T(A)) = \mathbf{f}(A), \forall A \subset X$.

If $\mathbf{f}_s(x) = \mathbf{f}(x - s\eta) \forall \eta \in \mathbb{R}^n$*, then the optimal map from* $\mathbf{f}_s(x)$ *to* $\mathbf{g}(y)$ *is* $T_s = T(x - s\eta)$*. Moreover,* $W_2^2(\mathbf{f}_s, \mathbf{g})$ *is convex with respect to the shift* s*.*

In fact, $W_2^2(\mathbf{f}_s, \mathbf{g}) = a + bs + cs^2$*, where* $a = W_2^2(\mathbf{f}, \mathbf{g}), b = 2\int_X \eta \cdot (x - T(x))\mathbf{f}(x)dx, c = |\eta|^2$*.*

Theorem 2.2.2 (Convexity with respect to dilation [148])**.** *Assume* $\mathbf{f}(x)$ *is a probability density function and* $\mathbf{g}(y) = \mathbf{f}(A^{-1}\mathbf{g})$ *where* A *is a symmetric positive definite matrix. Then* $W_2^2(\mathbf{f}, \mathbf{g}/\bar{\mathbf{g}})$ *is convex with respect to the eigenvalues* $\lambda_1, \cdots, \lambda_n$ *of* A*.*

In fact $W_2^2(\mathbf{f}, \mathbf{g}/ < \mathbf{g} >) = \int_X \mathbf{f}(Oz)z^T(I - \Lambda)^2 z dz$*, where* O *depends on* A*, and* Λ *is the diagonal matrix with diagonal element* $\lambda_1, \cdots, \lambda_n$*.*

The quadratic Wasserstein loss is also convex with respect to partial amplitude change. Consider the problem where a profile $\mathbf{f}$ is derived from $\mathbf{g}$, but with a decreased amplitude in part of the domain. That is, one supposes that the domain is decomposed into $\Omega = \Omega_1 \cup \Omega_2$, with $\Omega_1 \cap \Omega_2 = \emptyset$. For an amplitude loss parameter $0 \leq \beta \leq 1$, suppose that

$$\mathbf{f}_\beta(x) = \begin{cases} \beta\mathbf{g}(x), x \in \Omega_1 \\ \mathbf{g}(x), x \in \Omega_2 \end{cases} \tag{2.11}$$

Then following results hold.

Theorem 2.2.3 (Convexity of partial amplitude change [148])**.** $W_2^2(\mathbf{f}_\beta/\bar{\mathbf{f}_\beta}, \mathbf{g})$ *is convex function of* β*.*

Theorem 2.2.4 (Insensitivity with respect to oscillation [148])**.** *For* $k \geq 1$*, consider* $\mathbf{f}_k(x) = 1 + sin(2\pi kx)$ *on the line segment* $[0, 1]$*. Let* $d\mu_k(x) = \mathbf{f}_k(x)dx$*, and let* $d\nu(x) = dx$ *on* $[0, 1]$*, then* $W_p(\mu_k, \nu) = O(\frac{1}{k}), \forall p \geq 1$

Theorem 2.2.5 (Insensitivity to noise in 1-D [148])**.** *Let* $\mathbf{g}$ *be a positive probability density function on* $[0, 1]$ *and choose* $0 < c < \min \mathbf{g}$*. Let* $\mathbf{f}_N(x) = \mathbf{g}(x) + r^N(x)$*, which contains* N *piece piecewise constant noise* r_N *drawn from the uniform distribution* $U[-c, c]$*. Then* $\mathbb{E}W_2^2(\mathbf{f}_N/\bar{\mathbf{f}_N}, \mathbf{g}) = O(\frac{1}{N})$*, while* $\mathbb{E}L_2(\mathbf{f}_N/\bar{\mathbf{f}_N}, \mathbf{g}) = O(1)$

Let us remark that while the insensitivity result here is presented in the one-dimensional setting, it holds in higher-dimensional settings as well.

2.2.2 Dual formulation of W_1

Kantorovich's formulation of the optimal transport problem is a linear programming problem. In other words, one can solve Kantorovich's problem in the discrete case by linear programming.

Suppose X, Y are discrete spaces. Take $\mu = \sum_{i=1}^{n} \mu_i \delta_{x_i} = (\mu_i)_{1 \le i \le n}, \nu = \sum_{j=1}^{m} \nu_j \delta_{y_j} = (\nu_j)_{1 \le j \le m}$, where $\sum_{i=1}^{n} \mu = 1, \sum_{j=1}^{m} \nu_j = 1$. Any measure $\pi(x, y)$ can be represented by a nonnegative matrix $\pi = (\pi_{ij})$, where $\sum_i \pi_{ij} = \nu_j, \forall j$ and $\sum_j \pi_{ij} = \mu_i, \forall i$. And the cost is represented by matrix $c = (c_{ij})$. In this case, the Kantorovich problem reduces to the following linear programming problem:

$$\begin{aligned} &\inf_{\pi_{ij}} \sum_{ij} \pi_{ij} c_{ij} \\ &s.t. \pi_{ij} \ge 0 \forall i, j \\ &\sum_i \pi_{ij} = \nu_j, \forall j \\ &\sum_j \pi_{ij} = \mu_i, \forall i \end{aligned} \tag{2.12}$$

Linear programming has a perfect dual problem framework. Therefore the Kantorovich problem also has a duality framework.

Theorem 2.2.6 (Kantorovich Duality [135])**.** *Let X and Y be Polish spaces, let $\mu \in P(X)$ and $\nu \in P(Y)$, and let $c : X \times Y \to \mathbb{R}_+ \cup \{+\infty\}$ be a lower semi-continuous cost function. Whenever $\pi \in P(X \times Y)$ and $(\phi, \psi) \in \mathcal{L}^1(d\mu) \times \mathcal{L}^1(d\nu)$, we define*

$$I[\pi] = \int_{X \times Y} c(x, y) d\pi(x, y) J(\phi, \psi) = \int_X \phi d\mu + \int_Y \psi d\nu \tag{2.13}$$

Define $\Pi(\mu, \nu)$ to be the set of all Borel probability measures π on $X \times Y$ such that for all measureable subsets $A \subset X$ and $B \subset Y$, $\pi[A \times Y] = \mu[A]$, $\pi[X \times B] = \nu[B]$, and define Φ_c to be the set of all measurable functions $(\phi, \psi) \in \mathcal{L}^1(d\mu) \times \mathcal{L}^1(d\nu)$ satisfying

$$\phi(x) + \psi(y) \le c(x, y) \tag{2.14}$$

for $d\mu$-almost all $x \in X$, $d\nu$-almost all $\mathbf{g} \in Y$.

Then we have

$$\inf_{\Pi(\mu,\nu)} I[\pi] = \sup_{\Phi_c} J(\phi, \psi)\,. \tag{2.15}$$

Moreover, the infimum on the left-hand side of (2.15) *is attained. Furthermore, it does not change the value of the supremum on the right-hand side of* (2.15) *if one restricts the definition of Φ_c to those functions (ϕ, ψ) which are bounded and continuous.*

Basically the Kantorovich duality says that the Kantorovich problem is equivalent to the dual problem $\sup_{\Phi_c} J(\phi, \psi)$. This provides another way of interpreting the Wasserstein distance.

Dual formulation for W_1. When $c(x, y)$ is a metric $d(x, y)$, the greatest possible $\phi(x)$ given $\psi(y)$ is $\psi(y) = \sup_{\mathbf{g}}\{c(x, y) - \phi(x)\} = \phi^d(y)$, therefore $\sup_{\phi,\psi\in\Phi_c} (\mathbf{f}, \phi) + (\mathbf{g}, \psi) \le \sup_{\phi\in L^1(d\mu)} (\mathbf{f}, \phi) + (\mathbf{g}, \phi^d) \le \sup_{\phi\in L^1(d\mu)} (\mathbf{f}, \phi^{dd}) + (\mathbf{g}, \phi^d)$, where $\phi^{dd}(x) = \sup_{\mathbf{g}}\{d(x, y) - \phi^d(y)\}$.

Moreover, suppose $\phi(x_0)+\phi^d(y_0) = d(x_0, y_0)$, then $\forall x_1, \phi(x_1)+\phi^d(y_0) \le d(x_1, y_0)$. Therefore

$$\phi(x_1) - \phi(x_0) \le d(x_1, y_0) - d(x_0, y_0) \le d(x_1, x_0)\,. \tag{2.16}$$

Similarly $-d(x_1, x_0) \le \phi(x_1) - \phi(x_0)$, therefore $|\phi(x_0) - \phi(x_1)| \le d(x_0, x_1)$, thus it is reasonable to restrict $\|\phi\|_{Lip} \le 1$, where $\|\phi\|_{Lip} := \sup_{x_0,x_1} \frac{|\phi(x_0)-\phi(x_1)|}{d(x_0,x_1)}$.

Following the same argument, one has $\|\phi\|_{Lip} \le 1$, which implies that $-\phi^d(x) \le \inf_y\{d(x, y) - \phi^d(y)\} \le -\phi^d(x)$. That is to say $\phi^{dd}(x) = -\phi^d(x)$. Combining the two conclusions, one has

$$\begin{aligned}\sup_{\phi,\psi\in\Phi_c} (\mathbf{f}, \phi) + (\mathbf{g}, \psi) \le \sup_{\phi\in L^1(d\mu)} (\mathbf{f}, \phi^{dd}) + (\mathbf{g}, \phi^d) \le \sup_{\|\phi\|_{Lip}\le 1} (\mathbf{f}, \phi^{dd}) + (\mathbf{g}, \phi^d) = \\ \sup_{\|\phi\|_{Lip}\le 1} (\mathbf{f}, -\phi^d) + (\mathbf{g}, \phi^d) \le \sup_{\phi,\psi\in\Phi_c} (\mathbf{f}, \phi) + (\mathbf{g}, \psi)\,.\end{aligned} \tag{2.17}$$

Therefore, $\sup_{\|\phi\|_{Lip}\le 1} (\mathbf{f}, -\phi^d) + (\mathbf{g}, \phi^d) = \sup_{\phi,\psi\in\Phi_c} (\mathbf{f}, \phi) + (\mathbf{g}, \psi)$ when $c(x, y)$ is a metric. Furthermore, one can replace the optimization variable with ϕ^d, which can simplify the optimization even

further that

$$W_1(\mathbf{f}, \mathbf{g}) = \sup_{\phi: \|\phi\|_{Lip} \leq 1} (\mathbf{f}, \phi) - (\mathbf{g}, \phi) \,. \tag{2.18}$$

This is the dual norm of $\mathbf{f} - \mathbf{g}$ of the Lipschitz space. Comparing this formulation to the general duality framework, one knows in prior that $\psi = -\phi$ and one can restrain $\|\phi\|_{Lip} \leq 1$. This is summarized in the following result.

Theorem 2.2.7 (Kantorovich-Rubinstein theorem [135])**.** *when the cost of the optimal transportation for the cost $c(x, y) = d(x, y)$, then*

$$W_1(\mathbf{f}, \mathbf{g}) = \sup_{\|\phi\|_{Lip} \leq 1} (\mathbf{f} - \mathbf{g}, \phi) \tag{2.19}$$

where $\|\phi\|_{Lip} = \sup_{x,y} |\frac{\phi(x) - \phi(y)}{x - y}|$

Notice that

$$M : \{\phi : \|\phi\|_{Lip} \leq 1\} \to \{\phi : \|\phi\|_{Lip} \leq C\}, M(\phi) = C\phi \tag{2.20}$$

is a bijection. This gives us

$$\begin{aligned} &\sup_{\|\phi\|_{Lip} \leq C} (\mathbf{f} - \mathbf{g}, \phi) \\ &= \sup_{\|\phi\|_{Lip} \leq 1} (\mathbf{f} - \mathbf{g}, C\phi) \\ &= C \sup_{\|\phi\|_{Lip} \leq 1} (\mathbf{f} - \mathbf{g}, \phi) \\ &= C W_1(\mathbf{f}, \mathbf{g}) \,. \end{aligned} \tag{2.21}$$

That implies that changing the domain of ϕ to Lipschitz-C function will only result in a rescaling factor C, and, therefore will not change the optimization problem (and certainly not change the optimal ϕ). Fortunately, Lipschitz-C function is enough to deal with most functions one encounters in real applications.

Another useful way to rewrite the Kantorovich-Rubinstein theorem in $\mathbb{R}^n$, when the distance d is the standard Euclidean distance is converting $\|\phi\|_{Lip} \leq 1$ to $\|\nabla\phi\|_{L^\infty} \leq 1$. Suppose $\phi \in C^1$, then

$\|\phi\|_{Lip} \leq 1$ is equivalent to $\|\nabla\phi\| \leq 1$. Then the Lagrangian functional is

$$\sup_{\phi} \inf_{\lambda_1 \leq 0, \lambda_2 \geq 0} \int (\mathbf{f} - \mathbf{g})\phi + \lambda_1(\nabla\phi - 1) + \lambda_2(\nabla\phi + 1)\,. \tag{2.22}$$

The first-order optimality condition with respect to ϕ is $\mathbf{f} - \mathbf{g} - \nabla\lambda_1 - \nabla\lambda_2 = 0$ where $\lambda_1 \leq 0, \lambda_2 \geq 0$. Let $\lambda = \lambda_1 + \lambda_2$, then λ is a free variable, and $\lambda_1\lambda_2 = 0$. Therefore one can think $\lambda_1 = \lambda^-, \lambda_2 = \lambda^+$. After the substitution the optimality condition becomes $\nabla\lambda = \mathbf{f} - \mathbf{g}$.

The Lagrangian at the first-order optimality point is

$$\int \lambda_2 - \lambda_1 = \int |\lambda| = \|\lambda\|_1\,. \tag{2.23}$$

Therefore the dual problem becomes

$$\begin{aligned} &W_1(\mathbf{f}, \mathbf{g}) = \inf \|\lambda\|_1 \\ &s.t. \nabla \cdot \lambda = \mathbf{f} - \mathbf{g}\,. \end{aligned} \tag{2.24}$$

This shows that $W_1(\mathbf{f}, \mathbf{g})$ only depends on the difference $\mathbf{f} - \mathbf{g}$. By the complementary condition of Lagrangian multiplier, one has

$$\begin{cases} \lambda_1(\nabla\phi - 1) = 0, \\ \lambda_2(\nabla\phi + 1) = 0\,, \end{cases} \tag{2.25}$$

which means that

$$\begin{cases} \nabla\phi = 1 \;\; on \;\; \lambda > 0 \\ \nabla\phi = -1 \;\; on \;\; \lambda < 0\,. \end{cases} \tag{2.26}$$

Therefore the optimal ϕ is zig-zag function on $\lambda \neq 0$. ϕ only changes direction when λ changes sign, which means ϕ is approximately one integral smoother than λ. Since λ is one time smoother than $\mathbf{f} - \mathbf{g}$, we conclude that ϕ is also twice smoother than $\mathbf{f} - \mathbf{g}$.

This formulation is also called the dual of the dual formulation [135]. It also defines a metric.

Theorem 2.2.8 ([135]). *W_1 given in* (2.24) *defines a metric on space of probability density functions.*

Proof. Obviously $W_1(\mathbf{f},\mathbf{g}) \geq 0, W_1(\mathbf{f},\mathbf{g}) = 0 \leftrightarrow \mathbf{f} = \mathbf{g}$ and it is clear from definition that $W_1(\mathbf{f},\mathbf{g}) = W_1(\mathbf{g},\mathbf{f})$. It suffices to prove the triangle inequality.

Suppose λ_0, λ_1 are optimal solution for $(\mathbf{f},\mathbf{g}), (\mathbf{g},\mathbf{h})$ respectively. Then $\mathbf{f}-\mathbf{h} = \mathbf{f}-\mathbf{g}+\mathbf{g}-\mathbf{h} = \nabla\cdot\lambda_0 + \lambda_1$, namely, $\lambda_0+\lambda_1$ is a feasible solution for $(\mathbf{f},\mathbf{h})$. It follows that

$$W_1(\mathbf{f},h) \leq \|\lambda_0+\lambda_1\|_{L^1} \leq \|\lambda_0\|_{L^1} + \|\lambda_1\|_{L^1} = W_1(\mathbf{f},\mathbf{g}) + W_1(\mathbf{g},\mathbf{h})\,. \tag{2.27}$$

The proof is complete. □

Intuitively, since $\nabla\cdot\lambda = \mathbf{f}-\mathbf{g}$, we see that approximately $\lambda \sim \int \mathbf{f}-\mathbf{g}$, i.e., λ is one order of integration smoother than $\mathbf{f}-\mathbf{g}$. W_1 minimizes L_1 norm of λ. Therefore solving the inverse problem via W_1 dual of the dual formulation will give a smoother solution than L_2 minimization.

Let us recall that the $\mathcal{H}^{-1}$ norm of $\mathbf{f}-\mathbf{g}$ is defined as

$$\|\mathbf{f}-\mathbf{g}\|_{\mathcal{H}^{-1}(\Omega)} = \inf\{\|\nabla P\|_{\mathcal{L}^2(\Omega)} : -\nabla\cdot\nabla P = \mathbf{f}-\mathbf{g}, P|_{\partial\Omega} = 0\}\,. \tag{2.28}$$

If we think of ∇P as λ, we see the great similarity between loss functions $W_1(\mathbf{f},\mathbf{g})$ and $\|\mathbf{f}-\mathbf{g}\|_{\mathcal{H}^{-1}}$, assuming $\mathbf{f} = \mathbf{f}(\theta)$, in terms of their smoothing property.

A slight modification of the $\mathcal{H}^{-1}$ loss is directly connected to the W_2 loss.

2.3 Linearization of the W_2 distance

It is well-known that the linearization of the W_2 loss leads to a weighted $\mathcal{H}^{-1}$ loss, assuming that f and g have the same total mass. The rigorous derivation is documented in [135]. Here we provide a brief explanation of this connection.

Recall from Theorem 2.1.2 that the Monge-Ampere equation for the optimal transportation plan reads:

$$det(D^2u(x)) = \frac{\mathbf{f}(x)}{\mathbf{g}(\nabla u)}\,. \tag{2.29}$$

Assume that $\mathbf{f}$ is strictly positive, and $\mathbf{g}$ is very close to $\mathbf{f}$. This means that ∇u is very close to the identity. More precisely,

$$u(x) = \frac{|x|^2}{2} + \epsilon\psi + O(\epsilon^2), \qquad \mathbf{g} = (1 + \epsilon h + O(\epsilon^2))\mathbf{f}\,. \tag{2.30}$$

Plugging this assumption into (2.29) and keeping only first-order terms in ϵ, one has

$$-\Delta\psi + \nabla(-log f) \cdot \nabla\psi = h\,. \tag{2.31}$$

If we define $L = -\Delta + \nabla(-log f) \cdot \nabla$, then the linearized Monge-Ampere equation is $L\psi = h$. Here is a useful and easily checked lemma about the operator L.

Lemma 2.3.1 ([135])**.** *The operator $L = -\Delta + \nabla(-log f) \cdot \nabla$ satisfies the following integration by parts formula, $\forall f_1, f_2 \in H_0^2$,*

$$\int_X Lf_1 f_2 d\mu = \int_X f_1(Lf_2)d\mu = \int_X \nabla f_1 \cdot \nabla f_2 d\mu \tag{2.32}$$

One can introduce the weighted Sobolev Space norms by analogy with normal Sobolev space concept,

$$\begin{aligned} \|u\|^2_{\mathcal{L}^2(d\mu)} &= \textstyle\int_X u^2 d\mu \\ \|u\|^2_{\dot{\mathcal{H}}^1(d\mu)} &= \textstyle\int_X |\nabla u|^2 d\mu = \int_X uLu d\mu \\ \|u\|_{\dot{\mathcal{H}}^{-1}(d\mu)} &= \sup\{\textstyle\int_X uv d\mu : \|v\|_{\dot{\mathcal{H}}^1(d\mu)} = 1\} \end{aligned} \tag{2.33}$$

Then it is simple to check that the dual norm

$$\|u\|^2_{\dot{\mathcal{H}}^{-1}(d\mu)} = \int_X u(L^{-1}u)d\mu\,. \tag{2.34}$$

If u is the solution of (2.29), then the optimal transportation plan is $x - \nabla u = \epsilon\psi + O(\epsilon^2)$. Therefore

we have

$$\begin{aligned} W_2^2(\mu,\nu) &= \epsilon^2 \int_X |\nabla\psi|^2 d\mu + O(\epsilon^2) \\ &\approx \epsilon^2 \int_X \psi L\psi d\mu \\ &= \epsilon^2 \int_X h(L^{-1}h) d\mu \\ &= \|\epsilon h\|^2_{\dot{\mathcal{H}}^{-1}(d\mu)}, \end{aligned} \tag{2.35}$$

where $d\mu = \mathbf{f}(x)dx, d\nu = \mathbf{g}(x)dx$. Therefore, when μ and ν are close, the W_2 norm is weighted $\mathcal{H}^{-1}$, and the weight is μ.

The above calculation can be summarized as follows.

Theorem 2.3.2 (Weighted $\mathcal{H}^{-1}$ Norm is equivalent to W_2 Asymptotically [107])**.**

$$W_2(\mu, \mu + d\mu) = \|d\mu\|_{\dot{\mathcal{H}}^{-1}(\mu)} + o(d\mu)\,.$$

W_2 **based on dual formulation.** Using the duality framework of the original Kantorovich's problem of quadratic cost, one can show that W_2 is equivalent to the following unconstrained optimization problem

$$W_2^2(\mu,\nu) = M_2 - \inf_{\substack{\phi\in L^1(d\mu) \\ \phi \text{ is convex}}} \int_{\mathbb{R}^n} \phi d\mu + \int_{\mathbb{R}^n} \phi^* d\nu \tag{2.36}$$

where

$$M_2 = \int_{\mathbb{R}^n} \frac{|x|^2}{2} d\mu + \int_{\mathbb{R}^n} \frac{|y|^2}{2} d\nu \tag{2.37}$$

is the second moments of $\mathbf{f}$ and $\mathbf{g}$. Equivalently, one has

$$W_2^2(\mu,\nu) = M_2 - \inf_{(\phi,\psi)\in\Phi} \int_{\mathbb{R}^n} \phi d\mu + \int_{\mathbb{R}^n} \psi d\nu\,, \tag{2.38}$$

where

$$\Phi := \{(\phi,\psi) \in L^1(d\mu) \times L^1(d\mu); (x,y) \le \phi(x) + \psi(y)\}\,. \tag{2.39}$$

The systems (2.36) and (2.38) are simpler to work with than the Benamou-Brenier's formula since one only needs to store $\phi(x), \psi(y)$ (instead of $\rho(x,t), q(x,t)$) in these formulations.

By (2.36), one can rewrite $W_2^2(\mathbf{f},\mathbf{g}) = M_2 - \inf_{\phi \text{ is convex}} (\mathbf{f},\phi) + (\mathbf{g},\phi^*)$. This formulation gives a lower bound for W_2^2 by assigning ϕ. For example, taking $\phi(x) = \frac{|x|^2}{2}$, one can find that $W_2^2(\mathbf{f},\mathbf{g}) \geq M_2 - \int_{\mathbb{R}^n} \frac{|x|^2}{2} d\mu + \int_{\mathbb{R}^n} \frac{|y|^2}{2} d\nu = 0$. While the original formulation

$$W_2^2(\mathbf{f},\mathbf{g}) = \inf_{\pi(x,y)\in\Pi(\mu,\nu)} \int_{\mathbb{R}^{2n}} \frac{|x-y|^2}{2} d\pi(x,y) \tag{2.40}$$

gives an upper bound of W_2^2. One can take special ϕ to make general inequalities for W_2.

Let us define $J(\mathbf{f},\mathbf{g},\phi) = (\mathbf{f},\phi) + (\mathbf{g},\phi^*)$, where ϕ is convex. For fixed $\mathbf{f},\mathbf{g}$, one can find $\frac{\delta J}{\delta \phi}$.

$$\begin{aligned} (\mathbf{g},\phi^*) &= \int_{\mathbb{R}^n} \mathbf{g}(y)\phi^*(y)dy \\ &= \int_{\mathbb{R}^n, x=\arg\max xy-\phi(x)} \mathbf{g}(y)(xy-\phi(x))dy\,. \end{aligned} \tag{2.41}$$

Therefore, we have

$$\begin{aligned} \delta J &= (\mathbf{f},\delta\phi) + \int_{\mathbb{R}^n, x=\arg\max xy-\phi(x)} \mathbf{g}(y)(-\delta\phi(x))dy \\ &= (\mathbf{f},\delta\phi) + \int_{\mathbb{R}^n, \mathbf{g}=\partial\phi(x)} \mathbf{g}(y)(-\delta\phi(\nabla\phi^*(y)))dy \\ &= (\mathbf{f},\delta\phi) - \int_{\mathbb{R}^n} \mathbf{g}(\nabla\phi(z))\delta\phi(\nabla\phi^*(\nabla\phi(z)))\det(\nabla^2\phi(z))dz \\ &= \int_{\mathbb{R}^n} \mathbf{f}(x)\delta\phi(x)dx - \int_{\mathbb{R}^n} \mathbf{g}(\nabla\phi(x))\delta\phi(x)\det(\nabla^2\phi(x))dx \\ &= \int_{\mathbb{R}^n} (\mathbf{f}(x) - \mathbf{g}(\nabla\phi(x))\det(\nabla^2\phi(x)))\delta\phi(x)dx\,. \end{aligned} \tag{2.42}$$

Therefore $\frac{\delta J}{\delta\phi} = \mathbf{f}(x) - \mathbf{g}(\nabla\phi(x))\det(\nabla^2\phi(x))$. Thus when ϕ is the optimal solution, one should have $\mathbf{f}(x) = \mathbf{g}(\nabla\phi(x))\det(\nabla^2\phi(x))$, which is exactly the Monge-Ampere equation.

$\frac{\delta W_2^2(\mathbf{f},\mathbf{g})}{\delta \mathbf{f}}$ in n-dimensional space. This dual framework gives a universal variational form of W_2^2 in n-dimensional space($n \geq 1$) [23]. Suppose $\mathbf{g}$ is fixed, one would like to find how W_2^2 changes with

respect to $\mathbf{f}$, i.e. $\frac{\delta W_2^2(\mathbf{f},\mathbf{g})}{\delta \mathbf{f}}$. One can take the variational form of W_2^2:

$$\delta W_2^2(\mathbf{f},\mathbf{g}) = \int_{\mathbb{R}^n} \frac{|x|^2}{2}\delta\mathbf{f}(x)dx - \int_{\mathbb{R}^n} \phi(x)\delta\mathbf{f}(x)dx - (\int_{\mathbb{R}^n} \mathbf{f}(x)\delta\phi(x)dx + \int_{\mathbb{R}^n} \mathbf{g}(y)\delta\phi^*(y)dy)\,, \tag{2.43}$$

where $\phi(x) = \arg \inf_{\phi \text{ is convex}} (\mathbf{f},\phi) + (\mathbf{g},\phi^*)$, which implies $(\mathbf{f},\delta\phi) + (\mathbf{g},\delta\phi^*) = 0$, since ϕ is the global minimizer. Then

$$\delta W_2^2(\mathbf{f},\mathbf{g}) = \int_{\mathbb{R}^n} \frac{|x|^2}{2}\delta\mathbf{f}(x)dx - \int_{\mathbb{R}^n} \phi(x)\delta\mathbf{f}(x)dx\,, \tag{2.44}$$

which is equivalent to

$$\frac{\delta W_2^2(\mathbf{f},\mathbf{g})}{\delta \mathbf{f}} = \frac{|x|^2}{2} - \phi \tag{2.45}$$

for any dimensional space(not just one dimensional space we discussed before).

We can actually check that this formula reduces to the one obtained by the previous method in one dimension where

$$\begin{aligned} \tfrac{\delta W_2^2(\mathbf{f},\mathbf{g})}{\delta \mathbf{f}} &= \int_y^1 (T(x) - x)dx \\ &= \int_y^1 \phi'(x) - x dx \\ &= \tfrac{|y|^2}{2} - \phi(y)\,. \end{aligned} \tag{2.46}$$

Thus the gradient descent of W_2 distance boils down to finding ϕ, ϕ is the solution of the Monge-Ampere equation

$$\det(\nabla^2\phi(x)) = \frac{\mathbf{f}(x)}{\mathbf{g}(\nabla\phi(x))}\,, \tag{2.47}$$

or equivalently,

$$\phi(x) = \arg \inf_{\phi \text{ is convex}} (\mathbf{f},\phi) + (\mathbf{g},\phi^*)\,, \tag{2.48}$$

where ϕ is the optimal solution of the problem.

Equation for the derivative. If we denote by $F_{W_2^2} = \frac{\delta W_2^2(\mathbf{f},\mathbf{g})}{\delta \mathbf{f}}$, then $F_{W_2^2}$ satisfies

$$\mathcal{N} F_{W_2^2} = \frac{\mathbf{f}(x)}{\mathbf{g}(\nabla\phi(x))} - 1\,, \tag{2.49}$$

where $\mathcal{N}\mathbf{f} = \det(\nabla^2\mathbf{f})$. The structure of this equation is similar to that of the equation for the derivative $F_{\mathcal{H}^{-1}}$ of $\mathcal{H}^{-1}$ estimate:

$$\mathcal{L} F_{\mathcal{H}^{-1}} = \mathbf{f} - \mathbf{g}\,, \tag{2.50}$$

where $L\mathbf{f} = \Delta\mathbf{f}$. Therefore, the main difference of W_2^2 and $\mathcal{H}^{-1}$ is that the operator changes from Laplacian operator $L = tr(\nabla^2)$ to the $\det(\nabla^2)$ operator, and W_2 also uses relative difference $\frac{\mathbf{f}(x)}{\mathbf{g}(\nabla\phi(x))} - 1$ between two signals.

Chapter 3: Loss Functions Based on Wasserstin Distances

Wasserstein metrics based on optimal transport theory have been proposed as another type of mismatch measures in solving computational inverse problems; see for instance [19, 36, 37, 101, 100, 146, 147] and references therein for applications in seismic imaging, [121, 79] and references therein for mathematical imaging, and [125, 73, 58, 126] for computational inversion in machine learning.

Numerical experiments in the aforementioned references suggest that the Wasserstein metric has interesting properties that are attractive options for loss functions when considering solving inverse problems. Various works have been done on optimization landscape under Wasserstein metrics [133].

In this chapter, we study in detail the performance of such loss functions based on the Wasserstein metrics.

3.1 Loss function based on W_2 distance

Let us recall that we are interested in solving the inverse problem (1.1) by minimizing the mismatch between the model prediction $f(\theta)$ and the data g. That is, to minimize the functional

$$\Phi(\theta) := \frac{1}{2}\mathfrak{d}^2(\mathbf{f}(\theta), \mathbf{g}) + \frac{\gamma}{2}\mathcal{R}(\theta) \tag{3.1}$$

where $\mathfrak{d}^2(f, g)$ can be any of the loss functions we introduced in Chapter 1 or the Wasserstein distances we have introduced in Chapter 2. The functional $\mathcal{R}(\theta)$ represents the explicit regularization that imposes additional constraints on the known to be recovered, and γ is the parameter that controls the strength of the regularization term. We assume that the functional $\mathcal{R}$ is twice Fréchet differentiable.

When $f(\mathbf{x}) \geq 0$ and $g(\mathbf{x}) \geq 0$ are sufficiently regular and have the same total mass in the sense that $\int_\Omega f(\mathbf{x})d\mathbf{x} = \int_\Omega g(\mathbf{x})d\mathbf{x}$, the quadratic Wasserstein distance between f and g are given by, following the fluid dynamics formulation of Benamou and Brenier [14],

$$W_2^2(f,g) = \inf_{\rho,\boldsymbol{\omega}} \frac{1}{T}\int_0^T \int_\Omega \frac{1}{2}\rho|\boldsymbol{\omega}|^2 d\mathbf{x}dt \tag{3.2}$$

where $T > 0$ is given, $\rho(t,\mathbf{x})$ and $\boldsymbol{\omega}(t,\mathbf{x})$ satisfy, in the weak sense, the following over-determined transport equation:

$$\begin{aligned} \partial_t\rho + \nabla\cdot\rho\boldsymbol{\omega} &= 0, && \text{in } (0,T]\times\Omega \\ \rho(0,\mathbf{x}) &= f(\mathbf{x}), && \text{in } \Omega \\ \rho(T,\mathbf{x}) &= g(\mathbf{x}), && \text{in } \Omega \\ \mathbf{n}\cdot\boldsymbol{\omega} &= 0, && \text{on } (0,T]\times\partial\Omega\,. \end{aligned} \tag{3.3}$$

When $f(\mathbf{x}) = f(\sigma(\mathbf{x}))$, $W_2^2(f,g)$ can be viewed as a functional of σ. This functional is minimized when (1.1) is satisfied. Numerical experiments in [19, 36, 37, 101, 100, 146, 147, 121, 79] suggest that the quadratic Wasserstein metric W_2 has interesting properties that are attractive when considering solving inverse problems.

The objective function for the minimization problem based on W_2 loss, with additional regularization, now takes the form

$$\Phi(\theta) := \frac{1}{2}W_2^2(\mathbf{f}(\theta),\mathbf{g}) + \frac{\gamma}{2}\mathcal{R}(\theta)\,. \tag{3.4}$$

Rationale for constrained optimization approach. The functional $\Phi(\theta)$ in (3.1) can be minimized using an iterative scheme that starts with an initial guess θ_0 and keeps updating it until convergence. At any iteration k before convergence, we need to evaluate the distance between $\mathbf{f}(\theta_k)$ and $\mathbf{g}$, that is $\mathfrak{d}(\mathbf{f}(\theta_k),\mathbf{g})$, in order to evaluate the function value $\Phi(\theta_k)$ at the current step. This could be problematic when the standard quadratic Wasserstein metric W_2 is used here, because $W_2(\mathbf{f}(\theta_k),\mathbf{g})$ may not be defined when θ_k is not the solution of the inverse problem. In other words, during the iterations, $\mathbf{f}(\theta_k)$ and $\mathbf{g}$ might not have the same total mass. Therefore, $\Phi(\theta_k)$ might not

be finite if W_2 is the metric used in (3.1). This is why in most of the previous studies on using W_2 to solve inverse problems, $\mathbf{f}(\theta_k)$ has to be re-normalized at each iteration to have the same total mass as $\mathbf{g}$. This re-normalization process makes the analysis of the final inversion results very complicated.

The drawback of the above approach motivates us to use a constrained optimization approach to solve the minimization problem that avoids accurate evaluations of the metric $\mathfrak{d}(\mathbf{f}(\theta_k), \mathbf{g})$ during the iterations. We will formulate in Section 3.3 a constrained optimization approach to solve the minimization problem.

3.2 Wasserstein distances with mass unbalances

The Wasserstein distance $W_p(\mathbf{f}, \mathbf{g})$ from traditional optimal transport theory is only finite when two non-negative densities $\mathbf{f}$ and $\mathbf{g}$ have the same total mass on a domain $\Omega \subset \mathbb{R}^n$. This is an inconvenience for some applications where such a requirement is not satisfied. The distance has been generalized in several ways to deal with the situation that $\mathbf{f}$ and $\mathbf{g}$ do not have the same total mass. The optimal transport theory associated with the generalized metrics is often called unbalanced optimal transport. Here we recall several important generalizations for the purpose of comparison in our numerical investigation.

The first generalization is the Wasserstein-Fisher-Rao metric whose fluid dynamics formulation can be written as [108] [75][92, 86] [28, 27]:

$$W^2_{2,\mathrm{WFR}}(\mathbf{f}, \mathbf{g}) = \inf_{\rho,\boldsymbol{\omega},\zeta} \frac{1}{T} \int_0^T \int_\Omega \Big(\frac{1}{2}\rho|\boldsymbol{\omega}|^2 + \frac{1}{2}\rho\zeta^2 \Big) d\mathbf{x}dt \tag{3.5}$$

where

$$\begin{aligned} \partial_t \rho + \nabla \cdot \rho\boldsymbol{\omega} &= \rho\zeta(t, \mathbf{x}), && \text{in } (0, T] \times \Omega \\ \rho(0, \mathbf{x}) &= \mathbf{f}(\mathbf{x}), && \text{in } \Omega \\ \rho(T, \mathbf{x}) &= \mathbf{g}(\mathbf{x}), && \text{in } \Omega \\ \mathbf{n} \cdot \boldsymbol{\omega} &= 0, && \text{on } (0, T] \times \partial\Omega\,. \end{aligned} \tag{3.6}$$

The introduction of the additional dynamic source function ζ, which depends on both t and $\mathbf{x}$ variables, is used to absorb the mass difference between $\mathbf{f}$ and $\mathbf{g}$ such that $W_{2,\mathrm{WFR}}$ is finite even when $\mathbf{f}$ and $\mathbf{g}$ have different total mass.

The second generalization we will study is proposed in [53] as a variant of the Wasserstein-Fisher-Rao metric. Following [53], we call this metric $W_{2,\mathrm{UOT}}$. Its fluid dynamics formulation can be written as:

$$W^2_{2,\mathrm{UOT}}(\mathbf{f},\mathbf{g}) = \inf_{\rho,\boldsymbol{\omega},\zeta} \frac{1}{T}\int_0^T \int_\Omega \frac{1}{2}\rho|\boldsymbol{\omega}|^2 d\mathbf{x}dt + \int_0^T \frac{|\Omega|}{2\alpha}\zeta^2(t)dt \tag{3.7}$$

where α is a given parameter, and $(\rho,\boldsymbol{\omega})$ solves the following transport problem

$$\begin{aligned} \partial_t\rho + \nabla\cdot\rho\boldsymbol{\omega} &= \zeta(t), \quad \text{in } (0,T]\times\Omega \\ \rho(0,\mathbf{x}) &= \mathbf{f}(\mathbf{x}), \quad \text{in } \Omega \\ \rho(T,\mathbf{x}) &= \mathbf{g}(\mathbf{x}), \quad \text{in } \Omega \\ \mathbf{n}\cdot\boldsymbol{\omega} &= 0, \qquad \text{on } (0,T]\times\partial\Omega\,. \end{aligned} \tag{3.8}$$

The main difference between this formulation from the fluid dynamics formulation of $W_{2,\mathrm{WFR}}$ is that the dynamic source term ζ in the transport equation (3.8) depends only on the time variable t, not the $\mathbf{x}$ variable.

The third generalization is a slightly different version of $W_{2,\mathrm{WFR}}$, again, following [53], we call it generalized unnormalized optimal transport $W_{2,\mathrm{GUOT}}$

$$W^2_{2,\mathrm{GUOT}}(\mathbf{f},\mathbf{g}) = \inf_{\rho,\boldsymbol{\omega},\zeta} \frac{1}{T}\int_0^T \int_\Omega \frac{1}{2}\rho|\boldsymbol{\omega}|^2 d\mathbf{x}dt + \int_0^T \int_\Omega \frac{1}{2\alpha}\zeta^2(t,\mathbf{x})d\mathbf{x}dt \tag{3.9}$$

where α is a given parameter, and $(\rho, \boldsymbol{\omega})$ solves the following transport problem

$$\begin{aligned} \partial_t \rho + \nabla \cdot \rho\boldsymbol{\omega} &= \zeta(t,x), \quad \text{in } (0,T]\times\Omega \\ \rho(0,\mathbf{x}) &= \mathbf{f}(\mathbf{x}), \quad \text{in } \Omega \\ \rho(T,\mathbf{x}) &= \mathbf{g}(\mathbf{x}), \quad \text{in } \Omega \\ \mathbf{n}\cdot\boldsymbol{\omega} &= 0, \quad \text{on } (0,T]\times\partial\Omega\,. \end{aligned} \tag{3.10}$$

The difference from $W_{2,\mathrm{UOT}}$ is that the dynamic source term $\zeta(t,\mathbf{x})$ in the transport equation (3.10) depends on both $\mathbf{x}, t$.

The fourth generalization we consider is the mixed relaxed quadratic Wasserstein metric introduced by Benamou in [13]. The main idea here is to use a penalty term in the energy functional to soften the mass conservation requirement of the original balanced optimal transport. The dynamic formulation in this case is:

$$W^2_{2,\mathrm{Mixed}}(\mathbf{f},\mathbf{g}) = \inf_{\rho,\boldsymbol{\omega}} \frac{1}{T}\int_0^T \int_\Omega \frac{1}{2}\rho|\boldsymbol{\omega}|^2 d\mathbf{x}dt + \frac{\beta}{2}\int_\Omega |\rho(T,\mathbf{x}) - \mathbf{g}(\mathbf{x})|^2 d\mathbf{x} \tag{3.11}$$

where β is a given parameter, and

$$\begin{aligned} \partial_t \rho + \nabla \cdot \rho\boldsymbol{\omega} &= 0, \quad \text{in } (0,T]\times\Omega \\ \rho(0,\mathbf{x}) &= \mathbf{f}(\mathbf{x}), \quad \text{in } \Omega \\ \mathbf{n}\cdot\boldsymbol{\omega} &= 0, \quad \text{on } (0,T]\times\partial\Omega\,. \end{aligned} \tag{3.12}$$

The parameter β allows us to decide how strong we want to enforce the balance between $\mathbf{f}$ and $\mathbf{g}$. As β goes to $+\infty$, the metric returns to the original balanced W_2 metric.

The fifth generalization we consider is a slightly generalized version of W_1 given by[77]. They proved that W_1 could also be generalized to handle different total mass. The W_1 unnormalized Wasserstein metric is given by

$$UW_1(\mathbf{f}(\mathbf{x}), \mathbf{g}(\mathbf{x})) = \inf_{\rho(\mathbf{x}),\boldsymbol{\omega}(\mathbf{x}),\zeta(\mathbf{x})} \int_\Omega \rho(\mathbf{x})|\boldsymbol{\omega}(\mathbf{x})|d\mathbf{x} + \beta \int_\Omega |\zeta(\mathbf{x})|d\mathbf{x} \tag{3.13}$$

$$s.t. \nabla \cdot \rho(\mathbf{x})\boldsymbol{\omega}(\mathbf{x}) - \zeta(\mathbf{x}) = \mathbf{f}(\mathbf{x}) - \mathbf{g}(\mathbf{x}), x \in \Omega \tag{3.14}$$

Readers can refer to [77] for more comprehensive study of unnormalized Wasserstein metric. It is worth noting that UW_1 requires much less computational resources since it preserves the dimension of the original problems, while W_2 type of Wassersetin metrics usually requires one more dimension t to tackle the problem.

3.3 Constrained optimization algorithm

We reformulate the computational inversion problem as a PDE-constrained minimization problem. We now describe the procedure in the case where the metric $\mathfrak{d}$ is the $W_{2,\mathrm{WFR}}$ metric. We solve:

$$\min_{\rho,\boldsymbol{\omega},\zeta,\theta} \Phi_{W_{2,\mathrm{WFR}}}(\rho, \boldsymbol{\omega}, \zeta, \theta) \quad := \quad \frac{1}{T}\int_0^T \int_\Omega \frac{1}{2}\big(\rho|\boldsymbol{\omega}|^2 + \rho\zeta^2\big)d\mathbf{x}dt + \frac{\gamma}{2}\mathcal{R}(\theta)$$

subject to

$$\begin{aligned} \partial_t \rho + \nabla \cdot \rho\boldsymbol{\omega} &= \rho\zeta(t,\mathbf{x}), && \text{in } (0,T]\times\Omega \\ \rho(0,\mathbf{x}) &= \mathbf{f}(\theta(\mathbf{x})), && \text{in } \Omega \\ \rho(T,\mathbf{x}) &= \mathbf{g}(\mathbf{x}), && \text{in } \Omega \\ \mathbf{n}\cdot\boldsymbol{\omega} &= 0, && \text{on } (0,T]\times\partial\Omega\,. \end{aligned} \tag{3.15}$$

Formulations for the other metrics are done in exactly the same manner. For instance, if we remove the ζ variable, we would have the formulation for the case of the standard W_2 metric. Notice that the only free parameter is θ and we need to evaluate the forward operator $\mathbf{f}(\theta)$ in each iteration.

There has been tremendous progress in the past two decades on computational methods for PDE-constrained optimization problems such as (3.15); see for instance [5, 80] and references

therein for some recent overviews. This minimization problem is equivalent to the following saddle-point problem:

$$\inf_{\theta,\rho,\boldsymbol{\omega},\zeta} \sup_{\lambda} \mathcal{L}(\rho,\boldsymbol{\omega},\zeta,\theta,\lambda) \tag{3.16}$$

where the Lagrangian functional $\mathcal{L}$ is defined as

$$\begin{aligned}\mathcal{L}(\rho,\boldsymbol{\omega},\zeta,\theta,\lambda) &= \Phi_{W_{2,\mathrm{WFR}}}(\rho,\boldsymbol{\omega},\zeta,\theta) + \int_0^T \int_\Omega \lambda(\partial_t\rho + \nabla\cdot\rho\boldsymbol{\omega} - \rho\zeta)d\mathbf{x}dt \\ &+ \int_0^T \int_{\partial\Omega} \lambda\mathbf{n}\cdot\boldsymbol{\omega}dS(\mathbf{x})dt + \int_\Omega \lambda(0,x)(\rho(0,\mathbf{x}) - \mathbf{f}(\theta))d\mathbf{x} + \int_\Omega \lambda(T,x)(\rho(T,\mathbf{x}) - \mathbf{g})d\mathbf{x}\end{aligned} \tag{3.17}$$

with λ being the Lagrange multiplier for the constraints described by the transport equation and its initial, final and boundary conditions. Note that the implicit constraint $\rho \geq 0$ is not incorporated in this Lagrangian but will be imposed later in the numerical implementation.

The optimality conditions for the saddle point problem consist of three components: the forward transport problem (3.6) resulted from the variation of $\mathcal{L}$ with respect to λ, the adjoint transport problem resulted from the variation of $\mathcal{L}$ with respect to $(\rho,\boldsymbol{\omega})$:

$$\begin{aligned}\partial_t\lambda + \boldsymbol{\omega}\cdot\nabla\lambda + \zeta\lambda &= \frac{1}{2T}(|\boldsymbol{\omega}|^2 + |\zeta|^2), && \text{in } (0,T]\times\Omega \\ \boldsymbol{\omega} &= T\nabla\lambda, && \text{in } (0,T]\times\Omega \\ \zeta &= T\lambda, && \text{in } (0,T]\times\Omega \\ \lambda(T,\mathbf{x}) &= 0, && \text{in } \Omega \\ \lambda(t,\mathbf{x}) &= 0, && \text{on } (0,T]\times\partial\Omega\end{aligned} \tag{3.18}$$

and the control problem resulted from the variation of $\mathcal{L}$ with respect to θ:

$$\frac{\gamma}{2}\mathcal{R}'^*(\theta)[\chi_\Omega] + \mathbf{f}'^*(\theta)[\lambda(0,\mathbf{x})] = 0 \tag{3.19}$$

where $\mathbf{f}'^*(\theta)[\lambda(0,\mathbf{x})]$ is understood as the adjoint of the Fréchet derivative of $\mathbf{f}$ at θ in the direction

$\lambda(0, \mathbf{x})$.

As a side note, we observe that we can write down a closed equation for the adjoint variable λ by eliminating the variables $\boldsymbol{\omega}$ and ζ in the equation. This leads to the following Hamilton-Jacobi equation

$$\begin{aligned} \partial_t \lambda + \frac{1}{2T}|\nabla\lambda|^2 + \frac{1}{2T}\lambda^2 &= 0, \quad \text{in } (0,T]\times\Omega \\ \lambda(T,\mathbf{x}) &= 0, \quad \text{in } \Omega \\ \lambda(t,\mathbf{x}) &= 0, \quad \text{on } (0,T]\times\partial\Omega\,. \end{aligned} \tag{3.20}$$

We solve the nonlinear system of first-order optimality conditions (3.6), (3.18) and (3.19) with a variant of Newton's method. To present the method, we write the system abstractly into the form:

$$\boldsymbol{\mathcal{F}}(\rho, \boldsymbol{\omega}, \zeta, \theta, \lambda) = \mathbf{0}. \tag{3.21}$$

We use $\mathcal{F}^\mu$ to denote the μ ($\in \{\rho, \boldsymbol{\omega}, \zeta, \theta, \lambda\}$) component, including the equation for μ as well as the corresponding initial, final and boundary conditions, of $\boldsymbol{\mathcal{F}}$. Starting with an initial guess of the solution $(\rho_0, \boldsymbol{\omega}_0, \zeta_0, \theta_0, \lambda_0)$, Newton's method is characterized by the following iteration

$$(\rho_{k+1}, \boldsymbol{\omega}_{k+1}, \zeta_{k+1}, \theta_{k+1}, \lambda_{k+1}) = (\rho_k, \boldsymbol{\omega}_k, \zeta_k, \theta_k, \lambda_k) + \ell_k(\delta\rho, \delta\boldsymbol{\omega}, \delta\zeta, \delta\theta, \delta\lambda) \tag{3.22}$$

where$(\delta\rho, \delta\boldsymbol{\omega}, \delta\zeta, \delta\theta, \delta\lambda)$ is the update direction and ℓ_k is the step length in the update direction that will be determined with a line search process. The update direction is obtained by solving the system:

$$\begin{pmatrix} \mathcal{F}^\rho_\rho & \mathcal{F}^\rho_{\boldsymbol{\omega}} & \mathcal{F}^\rho_{\boldsymbol{\omega}} & \mathcal{F}^\rho_\theta & 0 \\ 0 & \mathcal{F}^{\boldsymbol{\omega}}_{\boldsymbol{\omega}} & 0 & 0 & \mathcal{F}^{\boldsymbol{\omega}}_\lambda \\ 0 & 0 & \mathcal{F}^\zeta_\zeta & 0 & \mathcal{F}^\zeta_\lambda \\ 0 & 0 & 0 & \mathcal{F}^\theta_\theta & \mathcal{F}^\theta_\lambda \\ 0 & \mathcal{F}^\lambda_{\boldsymbol{\omega}} & \mathcal{F}^\lambda_\zeta & 0 & \mathcal{F}^\lambda_\lambda \end{pmatrix} \begin{pmatrix} \delta\rho \\ \delta\boldsymbol{\omega} \\ \delta\zeta \\ \delta\theta \\ \delta\lambda \end{pmatrix} = - \begin{pmatrix} \mathcal{F}^\rho \\ \mathcal{F}^{\boldsymbol{\omega}} \\ \mathcal{F}^\zeta \\ \mathcal{F}^\theta \\ \mathcal{F}^\lambda \end{pmatrix} \tag{3.23}$$

where we used the notation $\mathcal{F}^\mu_\nu$ ($\mu, \nu \in \{\rho, \boldsymbol{\omega}, \zeta, \theta, \lambda\}$) to denote the Fréchet derivative of the $\mathcal{F}^\mu$ component of $\mathcal{F}$ with respect to the variable ν. The operators $\mathcal{F}^\mu_\nu$ as well as the functionals $\mathcal{F}^\mu$

are all evaluated at the current iteration $(\rho_k, \boldsymbol{\omega}_k, \zeta_k, \theta_k, \lambda_k)$. The exact forms of all the operators involved are summarized in the Appendix A.

3.4 Insights from linearization

The quadratic Wasserstein metrics we used in this chapter are all nonlinear metrics. This makes them not only computationally expensive to evaluate but also analytically challenging to understand. When using these metrics to solve inverse problems, we notice that the model prediction $\mathbf{f}(\theta)$ and the corresponding datum $\mathbf{g}$ are identical (or very close) to each other at the true solution θ. We now try to understand the behavior of the metrics in this situation following an informal linearization procedure.

3.4.1 Linearization of the Wasserstein metrics

We refer interested readers to [136, 119] for a rigorous treatment of the linearization of the W_2 metric, and [59, 98, 57] for applications of such linearization in the analysis of inversion results under the W_2 metric. Here we extend the analysis of [78] to the generalized metrics $W_{2,\mathrm{WFR}}$, $W_{2,\mathrm{UOT}}$ and $W_{2,\mathrm{Mixed}}$.

Linearization of W_2. We first determine the background state (that is, the case when $\mathbf{f} = \mathbf{g}$) of the transport equations. The ρ and $\boldsymbol{\omega}$ corresponding to this case is $\rho(t, \mathbf{x}) = \mathbf{f}$ (which implies that $\partial_t \rho(\mathbf{x}, t) = 0$) and $\boldsymbol{\omega} = \mathbf{0}$. When $\mathbf{g} - \mathbf{f}$ is small, we have that $\rho = \mathbf{f} + \delta\rho$ and $\boldsymbol{\omega} = \delta\boldsymbol{\omega}$ with $\delta\rho$ and $\delta\boldsymbol{\omega}$ small. The transport equation satisfied by $\delta\rho$ and $\delta\boldsymbol{\omega}$, to the first order, is:

$$\begin{aligned} \partial_t \delta\rho + \nabla \cdot \mathbf{f}\delta\boldsymbol{\omega} &= 0, && \text{in } (0,T] \times \Omega \\ \delta\rho(0, \mathbf{x}) &= 0, && \text{in } \Omega \\ \delta\rho(T, \mathbf{x}) &= \mathbf{g}(\mathbf{x}) - \mathbf{f}(\mathbf{x}), && \text{in } \Omega \\ \mathbf{n} \cdot \delta\boldsymbol{\omega} &= 0, && \text{on } (0,T] \times \partial\Omega\,. \end{aligned} \tag{3.24}$$

The square of the metric is then

$$W_2^2(\mathbf{f},\mathbf{g}) = \inf_{\delta\rho,\delta\boldsymbol{\omega}} \frac{1}{T}\int_0^T \int_\Omega \frac{1}{2}|\delta\boldsymbol{\omega}|^2\mathbf{f}(\mathbf{x})d\mathbf{x}dt + H.O.T \tag{3.25}$$

where $H.O.T$ stands for higher order terms.

The optimality condition for the system (3.24) and (3.25) gives that $\delta\boldsymbol{\omega}(t,\mathbf{x}) = \nabla\phi(\mathbf{x})$ with ϕ the solution to the elliptic equation:

$$-\nabla\cdot\mathbf{f}\nabla\phi = \frac{1}{T}(\mathbf{g}-\mathbf{f}), \ \text{ in } \ \Omega, \qquad \mathbf{n}\cdot\nabla\phi = 0, \ \text{ on } \ \partial\Omega. \tag{3.26}$$

This, in turn, means that (3.25) now becomes

$$W_2^2(\mathbf{f},\mathbf{g}) = \frac{1}{2}\int_\Omega |\nabla\phi|^2\mathbf{f}(\mathbf{x})d\mathbf{x} + H.O.T\,. \tag{3.27}$$

This simple calculation shows that when $\mathbf{g}-\mathbf{f}$ is sufficiently small, $W_2(\mathbf{f},\mathbf{g}) \approx \frac{1}{T}\|\mathbf{f}-\mathbf{g}\|_{\dot{\mathcal{H}}^{-1}_{(\mathbf{f}d\mathbf{x})}(\Omega)}$. This is the well-known asymptotic equivalence between W_2 and the $\dot{\mathcal{H}}^{-1}_{(\mathbf{f}d\mathbf{x})}$; see, for instance, [136] for more technical details on this equivalence.

Linearization of $W_{2,\mathrm{WFR}}$. We can linearize $W_{2,\mathrm{WFR}}$ in a similar way. The background solution to the transport equation in the case of $\mathbf{f}=\mathbf{g}$ is $(\rho,\boldsymbol{\omega},\zeta) = (\mathbf{f},\mathbf{0},0)$ for the $W_{2,\mathrm{WFR}}$ metric. When $\mathbf{g}$ is sufficiently close to $\mathbf{f}$, we have that $\rho = \mathbf{f}+\delta\rho$, $\boldsymbol{\omega} = \delta\boldsymbol{\omega}$ and $\zeta = \delta\zeta$, $(\delta\rho,\delta\boldsymbol{\omega},\delta\zeta)$ being sufficiently small, in an appropriate sense. The transport equation satisfied by $(\delta\rho,\delta\boldsymbol{\omega},\delta\zeta)$, to the first order, is then

$$\begin{aligned}
\partial_t\delta\rho + \nabla\cdot\mathbf{f}\delta\boldsymbol{\omega} &= \mathbf{f}\delta\zeta, && \text{in } (0,T]\times\Omega\\
\delta\rho(0,\mathbf{x}) &= 0, && \text{in } \Omega\\
\delta\rho(T,\mathbf{x}) &= \mathbf{g}(\mathbf{x})-\mathbf{f}(\mathbf{x}), && \text{in } \Omega\\
\mathbf{n}\cdot\delta\boldsymbol{\omega} &= 0, && \text{on } (0,T]\times\partial\Omega\,.
\end{aligned} \tag{3.28}$$

The square of the distance is given by

$$W_{2,\mathrm{WFR}}^2(\mathbf{f},\mathbf{g}) = \inf_{\delta\rho,\delta\boldsymbol{\omega}} \frac{1}{T}\int_0^T \int_\Omega \frac{1}{2}\Big(|\delta\boldsymbol{\omega}|^2 + (\delta\zeta)^2\Big)\mathbf{f}(\mathbf{x})d\mathbf{x}dt + H.O.T\,. \tag{3.29}$$

We can then check that the optimality conditions for the system (3.28) and (3.29) imply that $\delta\boldsymbol{\omega} = \nabla\phi(\mathbf{x})$ and $\delta\zeta = \phi$ with ϕ the solution to the following PDE:

$$-\nabla\cdot\mathbf{f}\nabla\phi + \mathbf{f}\phi = \frac{1}{T}(\mathbf{g}-\mathbf{f}), \;\text{ in }\; \Omega, \qquad \mathbf{n}\cdot\nabla\phi = 0, \;\text{ on }\; \partial\Omega. \tag{3.30}$$

The squared distance between $\mathbf{f}$ and $\mathbf{g}$ now becomes

$$W_{2,\mathrm{WFR}}^2(\mathbf{f},\mathbf{g}) = \frac{1}{2}\int_\Omega (|\nabla\phi|^2 + \phi^2)\mathbf{f}(\mathbf{x})d\mathbf{x} + H.O.T\,. \tag{3.31}$$

Therefore, the extra source term in the transport equation introduces an absorption mechanism in the linearized setting, that is, the term $\mathbf{f}\phi$ in (3.30). This absorption mechanism allows the densities $\mathbf{f}$ and $\mathbf{g}$ to have different total mass (in which case (3.30) still admits a unique bounded solution while (3.26) does not). This simple observation showcases that $W_{2,\mathrm{WFR}}$ is asymptotically equivalent to the $\mathcal{H}^{-1}_{(\mathbf{f}d\mathbf{x})}$ metric, not the $\dot{\mathcal{H}}^{-1}_{(\mathbf{f}d\mathbf{x})}$ metric.

Linearization of $W_{2,\mathrm{UOT}}$. Following the same procedure as in the $W_{2,\mathrm{WFR}}$ case, we have that $\rho = \mathbf{f} + \delta\rho$, $\boldsymbol{\omega} = \delta\boldsymbol{\omega}$ and $\zeta = \delta\zeta$ when $\mathbf{g}-\mathbf{f}$ is small. The only difference here is that $\delta\zeta$ depends only on t, not $\mathbf{x}$. The equation satisfied by $(\delta\rho, \delta\boldsymbol{\omega}, \delta\zeta)$, to the first order, is:

$$\begin{aligned}
\partial_t\delta\rho + \nabla\cdot\mathbf{f}\delta\boldsymbol{\omega} &= \delta\zeta(t), && \text{in } (0,T]\times\Omega\\
\delta\rho(0,\mathbf{x}) &= 0, && \text{in } \Omega\\
\delta\rho(T,\mathbf{x}) &= \mathbf{g}(\mathbf{x}) - \mathbf{f}(\mathbf{x}), && \text{in } \Omega\\
\mathbf{n}\cdot\delta\boldsymbol{\omega} &= 0, && \text{on } (0,T]\times\partial\Omega\,.
\end{aligned} \tag{3.32}$$

The square of the metric is

$$W_{2,\mathrm{UOT}}^2 = \inf_{\delta\rho,\delta\boldsymbol{\omega},\delta\zeta} \frac{1}{T}\int_0^T \int_\Omega \frac{1}{2}|\delta\boldsymbol{\omega}|^2 \mathbf{f}(\mathbf{x})d\mathbf{x}dt + \int_0^T \frac{1}{2}\frac{|\Omega|}{\alpha}(\delta\zeta(t))^2 dt + H.O.T\,. \tag{3.33}$$

We check that the optimality conditions of the problem (3.32) and (3.33) lead to $\delta\boldsymbol{\omega} = \nabla\phi(\mathbf{x})$ and $\delta\zeta = \frac{\alpha}{|\Omega|}\int_\Omega \phi(\mathbf{x})d\mathbf{x}$ with ϕ the solution to

$$-\nabla\cdot\mathbf{f}\nabla\phi + \frac{\alpha}{|\Omega|}\int_\Omega \phi(\mathbf{x})d\mathbf{x} = \frac{1}{T}(\mathbf{g}-\mathbf{f}), \;\; \text{in } \Omega, \qquad \mathbf{n}\cdot\nabla\phi = 0, \;\; \text{on } \partial\Omega. \tag{3.34}$$

Therefore, the square of the $W_{2,\mathrm{UOT}}$ distance between $\mathbf{f}$ and $\mathbf{g}$ is given as

$$W_{2,\mathrm{UOT}}^2(\mathbf{f},\mathbf{g}) = \frac{1}{2}\int_\Omega |\nabla\phi|^2\mathbf{f}(\mathbf{x})d\mathbf{x} + \frac{\alpha}{2|\Omega|}(\int_\Omega \phi(\mathbf{x})d\mathbf{x})^2 + H.O.T\,. \tag{3.35}$$

This calculation implies that when $\mathbf{f}$ and $\mathbf{g}$ have the same total mass, $\int_\Omega \phi(\mathbf{x})d\mathbf{x} = 0$. This can be easily seen by integrating (3.34) over the spatial domain Ω using the divergence theorem and the boundary condition. In this case, we can drop all the terms that involve $\int_\Omega \phi(\mathbf{x})d\mathbf{x} = 0$. Therefore, $W_{2,\mathrm{UOT}}$ reduces to W_2. This is not true for $W_{2,\mathrm{WFR}}$ which does not degenerate to W_2 in the case that $\mathbf{f}$ and $\mathbf{g}$ have the same total mass. Therefore, $W_{2,\mathrm{UOT}}$ is asymptotically equivalent to the $\mathcal{H}^{-1}_{(\mathbf{f}d\mathbf{x})}$ metric when $\mathbf{f}$ and $\mathbf{g}$ do not have the same total mass but is asymptotically equivalent to the $\dot{\mathcal{H}}^{-1}_{(\mathbf{f}d\mathbf{x})}$ metric if $\mathbf{f}$ and $\mathbf{g}$ do have the same total mass.

Linearization of $W_{2,\mathrm{GUOT}}$. Similarly as $W_{2,\mathrm{UOT}}$, we have that $\rho = \mathbf{f} + \delta\rho$, $\boldsymbol{\omega} = \delta\boldsymbol{\omega}$ and $\zeta = \delta\zeta$ when $\mathbf{g}-\mathbf{f}$ is small. The only difference here is that $\delta\zeta$ depends on both t and $\mathbf{x}$. The equation satisfied by $(\delta\rho, \delta\boldsymbol{\omega}, \delta\zeta)$, to the first order, is:

$$\begin{aligned}
\partial_t\delta\rho + \nabla\cdot\mathbf{f}\delta\boldsymbol{\omega} &= \delta\zeta(t,\mathbf{x}), && \text{in } (0,T]\times\Omega \\
\delta\rho(0,\mathbf{x}) &= 0, && \text{in } \Omega \\
\delta\rho(T,\mathbf{x}) &= \mathbf{g}(\mathbf{x}) - \mathbf{f}(\mathbf{x}), && \text{in } \Omega \\
\mathbf{n}\cdot\delta\boldsymbol{\omega} &= 0, && \text{on } (0,T]\times\partial\Omega\,.
\end{aligned} \tag{3.36}$$

The square of the metric is

$$W^2_{2,\mathrm{GUOT}} = \inf_{\delta\rho,\delta\boldsymbol{\omega},\delta\zeta} \frac{1}{T}\int_0^T \int_\Omega \frac{1}{2}|\delta\boldsymbol{\omega}|^2 \mathbf{f}(\mathbf{x}) d\mathbf{x}dt + \int_0^T \int_\Omega \frac{1}{2\alpha}(\delta\zeta(t,\mathbf{x}))^2 d\mathbf{x}dt + H.O.T\,. \quad (3.37)$$

We check that the optimality conditions of the problem (3.36) and (3.37) implies $\delta\boldsymbol{\omega} = \nabla\phi(\mathbf{x})$ and $\delta\zeta = \alpha\phi$ with ϕ satisfies

$$-\nabla\cdot\mathbf{f}\nabla\phi + \alpha\phi = \frac{1}{T}(\mathbf{g}-\mathbf{f}), \ \text{ in } \ \Omega, \qquad \mathbf{n}\cdot\nabla\phi = 0, \ \text{ on } \ \partial\Omega. \quad (3.38)$$

Therefore, the square of the $W_{2,\mathrm{GUOT}}$ distance between $\mathbf{f}$ and $\mathbf{g}$ can be written as follows

$$W^2_{2,\mathrm{GUOT}}(\mathbf{f},\mathbf{g}) = \frac{1}{2}\int_\Omega (|\nabla\phi|^2\mathbf{f}(\mathbf{x}) + \alpha\phi^2) d\mathbf{x} + H.O.T\,. \quad (3.39)$$

If we replace α with $\mathbf{f}(\mathbf{x})$ in (3.39) and (3.38), then it returns to the same formulation (3.31) and (3.30) for $W_{2,\mathrm{WFR}}$. Therefore comparing to $W_{2,\mathrm{WFR}}$, $W_{2,\mathrm{GUOT}}$ is asymptotically a mixed version of $\dot{\mathcal{H}}^{-1}_{(\mathbf{f}d\mathbf{x})}$ metric and $L^2_{(d\mathbf{x})}$ at finite α. On the other hand, $W_{2,\mathrm{GUOT}}$ is a generalization of $W_{2,\mathrm{UOT}}$ in the sense that it allows a non-uniform dynamic source term $\zeta(t,\mathbf{x})$.

Linearization of $W_{2,\mathrm{Mixed}}$. The background transport solution for the case of $\mathbf{f} = \mathbf{g}$ is again $(\rho,\boldsymbol{\omega}) = (\mathbf{f},\mathbf{0})$. When $\mathbf{g}-\mathbf{f}$ is sufficiently small, we have $\rho = \mathbf{f} + \delta\rho$ and $\boldsymbol{\omega} = \delta\boldsymbol{\omega}$, $(\delta\rho,\delta\boldsymbol{\omega})$ being sufficiently small. To the leading order, the transport equation satisfied by $(\delta\rho,\delta\boldsymbol{\omega})$ is

$$\begin{aligned} \partial_t\delta\rho + \nabla\cdot\mathbf{f}\delta\boldsymbol{\omega} &= 0, \quad \text{in } (0,T]\times\Omega \\ \delta\rho(0,\mathbf{x}) &= 0, \quad \text{in } \Omega \\ \mathbf{n}\cdot\delta\boldsymbol{\omega} &= 0, \quad \text{on } (0,T]\times\partial\Omega\,. \end{aligned} \quad (3.40)$$

The $W_{2,\mathrm{Mixed}}$ cost becomes

$$W^2_{2,\mathrm{Mixed}}(\mathbf{f},\mathbf{g}) = \inf_{\delta\rho,\delta\boldsymbol{\omega}} \frac{1}{T}\int_0^T \int_\Omega \frac{1}{2}\mathbf{f}|\delta\boldsymbol{\omega}|^2 d\mathbf{x}dt + \frac{\beta}{2}\int_\Omega (\delta\rho(T,\mathbf{x}) - (\mathbf{g}-\mathbf{f}))^2 d\mathbf{x} + H.O.T\,. \quad (3.41)$$

We then check that the optimality conditions of the problem (3.40) and (3.41) lead to that $\delta\omega(t, \mathbf{x}) = \nabla\phi(\mathbf{x})$ where ϕ solves

$$-\nabla \cdot \mathbf{f}\nabla\phi + \frac{1}{\beta T}\phi = \frac{1}{T}(\mathbf{g} - \mathbf{f}), \text{ in } \Omega, \qquad \mathbf{n} \cdot \nabla\phi = 0, \text{ on } \partial\Omega \tag{3.42}$$

The square of the metric now becomes

$$W^2_{2,\text{Mixed}}(\mathbf{f}, \mathbf{g}) = \frac{1}{2}\int_\Omega \left(|\nabla\phi|^2\mathbf{f}(\mathbf{x}) + \frac{1}{\beta}\phi^2\right)d\mathbf{x} + H.O.T\,. \tag{3.43}$$

This simple calculation also indicates that, in this asymptotic regime, $W_{2,\text{Mixed}}$ goes to W_2 as $\beta \to \infty$. However, at a finite β, $W_{2,\text{Mixed}}$ is asymptotically equivalent to the $\mathcal{H}^{-1}_{(\mathbf{f}d\mathbf{x})}$ metric, not the $\dot{\mathcal{H}}^{-1}_{(\mathbf{f}d\mathbf{x})}$ metric.

3.4.2 Linear inversion under the Wasserstein metrics

We briefly review the solution of a general linear inverse problem with the linearized Wasserstein metrics, following the presentation of [40]. The linear model, or the linearization of (1.2), takes the form:

$$A\theta = \mathbf{g}\,. \tag{3.44}$$

Here we focus on the effect of the metrics in the Fourier domain. Without loss of generality, we assume that $\Omega = (0, 2\pi)^d$, and the real-valued functions θ, $\mathbf{f}$, and $\mathbf{g}$, supported on Ω, all have boundary conditions of the form $\mathbf{n} \cdot \nabla\theta = 0$. We then have the following Fourier representations for those functions:

$$\theta(\mathbf{x}) = \sum_{\boldsymbol{\xi}\in\mathbb{Z}^d_+} \widehat{\theta}(\boldsymbol{\xi})\cos(\boldsymbol{\xi}\cdot\mathbf{x}), \quad \mathbf{f}(\mathbf{x}) = \sum_{\boldsymbol{\xi}\in\mathbb{Z}^d_+} \widehat{\mathbf{f}}(\boldsymbol{\xi})\cos(\boldsymbol{\xi}\cdot\mathbf{x}), \quad \mathbf{g}(\mathbf{x}) = \sum_{\boldsymbol{\xi}\in\mathbb{Z}^d_+} \widehat{\mathbf{g}}(\boldsymbol{\xi})\cos(\boldsymbol{\xi}\cdot\mathbf{x}). \tag{3.45}$$

To see the main effect of the metrics, we ignore the weight $\mathbf{f}$ in terms such as $-\nabla \cdot \mathbf{f}\nabla\phi$ and $\mathbf{f}\phi$ in the equation (3.26), (3.27) , (3.30), (3.31), (3.34), (3.35), (3.42), and (3.43). For reasonably nice $\mathbf{f}$, the factor $\mathbf{f}$ plays only a minor role in the Fourier domain as investigated in [107]. We also set $T = 1$.

We therefore have from (3.26) that

$$\phi(\mathbf{x}) = \sum_{\boldsymbol{\xi}\in\mathbb{Z}_+^d} \widehat{\phi}(\boldsymbol{\xi}) \cos(\boldsymbol{\xi}\cdot\mathbf{x}), \quad \text{with,} \quad \widehat{\phi}(\boldsymbol{\xi}) = \frac{1}{|\boldsymbol{\xi}|^2}\widehat{(\mathbf{g}-\mathbf{f})}(\boldsymbol{\xi}).$$

This leads to the following representation of the W_2^2 in the case of $\mathbf{g}-\mathbf{f}$ being small:

$$W_2^2(\mathbf{f},\mathbf{g}) \approx \frac{1}{2}\sum_{\boldsymbol{\xi}\in\mathbb{Z}_+^d} \frac{1}{|\boldsymbol{\xi}|^2}|\widehat{(\mathbf{g}-\mathbf{f})}(\boldsymbol{\xi})|^2 \tag{3.46}$$

where we have thrown away the higher-order terms. In a similar manner, we find that

$$W_{2,\mathrm{WFR}}^2(\mathbf{f},\mathbf{g}) \approx \frac{1}{2}\sum_{\boldsymbol{\xi}\in\mathbb{Z}_+^d} \frac{1}{1+|\boldsymbol{\xi}|^2}|\widehat{(\mathbf{g}-\mathbf{f})}(\boldsymbol{\xi})|^2, \tag{3.47}$$

$$W_{2,\mathrm{UOT}}^2(\mathbf{f},\mathbf{g}) \approx \frac{1}{2}\frac{\alpha}{\Omega}|\widehat{(\mathbf{g}-\mathbf{f})}(\mathbf{0})|^2 + \frac{1}{2}\sum_{\boldsymbol{\xi}\in\mathbb{Z}_+^d,\ \boldsymbol{\xi}\neq\mathbf{0}} \frac{1}{|\boldsymbol{\xi}|^2}|\widehat{(\mathbf{g}-\mathbf{f})}(\boldsymbol{\xi})|^2, \tag{3.48}$$

$$W_{2,\mathrm{GUOT}}^2(\mathbf{f},\mathbf{g}) \approx \frac{1}{2}\sum_{\boldsymbol{\xi}\in\mathbb{Z}_+^d} \frac{1}{\alpha+|\boldsymbol{\xi}|^2}|\widehat{(\mathbf{g}-\mathbf{f})}(\boldsymbol{\xi})|^2, \tag{3.49}$$

and

$$W_{2,\mathrm{Mixed}}^2(\mathbf{f},\mathbf{g}) \approx \frac{1}{2}\sum_{\boldsymbol{\xi}\in\mathbb{Z}_+^d} \frac{1}{\frac{1}{\beta}+|\boldsymbol{\xi}|^2}|\widehat{(\mathbf{g}-\mathbf{f})}(\boldsymbol{\xi})|^2. \tag{3.50}$$

To see the impact of the metrics on the inversion in the Fourier domain, we assume further that the operator A is diagonal in the Fourier domain, that is, the operator A has the representation

$$A\theta = \sum_{\boldsymbol{\xi}\in\mathbb{Z}_+} (\widehat{A}(\boldsymbol{\xi})\widehat{\theta}(\boldsymbol{\xi}))\cos(\boldsymbol{\xi}\cdot\mathbf{x}).$$

This assumption is not essential at all. It only simplifies the presentation below a little bit.

To invert the linear model (3.44) in the standard linearized W_2 metric, we seek the solution that

minimizes

$$\Phi_{W_2}(\theta) := W_2^2(\mathbf{f}(\theta), \mathbf{g}) = \frac{1}{2} \sum_{\boldsymbol{\xi} \in \mathbb{Z}_+^d} \frac{1}{|\boldsymbol{\xi}|^2} |\widehat{A}(\boldsymbol{\xi}) \widehat{\theta}(\boldsymbol{\xi}) - \widehat{\mathbf{g}}(\boldsymbol{\xi})|^2,$$

where we used the assumption that A is diagonal such that $\widehat{\mathbf{f}}(\theta)(\boldsymbol{\xi}) = \widehat{A\theta}(\boldsymbol{\xi}) = \widehat{A}(\boldsymbol{\xi}) \widehat{\theta}(\boldsymbol{\xi})$. The optimality condition resulted from the variation of $\Phi_{W_2}(\theta)$ with respect to $\widehat{\theta}(\boldsymbol{\xi})$ is

$$\widehat{A}^*(\boldsymbol{\xi}) \frac{1}{|\boldsymbol{\xi}|^2} \Big(\widehat{A}(\boldsymbol{\xi}) \widehat{\theta}(\boldsymbol{\xi}) - \widehat{\mathbf{g}}(\boldsymbol{\xi}) \Big) = 0,$$

which then leads to the solution

$$\widehat{\theta}_{W_2}(\boldsymbol{\xi}) = (\widehat{A}^* \frac{1}{|\boldsymbol{\xi}|^2} \widehat{A})^{-1} \widehat{A}^* \frac{1}{|\boldsymbol{\xi}|^2} \widehat{\mathbf{g}}(\boldsymbol{\xi}).$$

The same derivation procedure leads to the following inversion results from the $W_{2,\mathrm{WFR}}$, $W_{2,\mathrm{UOT}}$ and $W_{2,\mathrm{Mixed}}$ metrics. They are respectively:

$$\widehat{\theta}_{W_{2,\mathrm{WFR}}}(\boldsymbol{\xi}) = (\widehat{A}^* \frac{1}{1 + |\boldsymbol{\xi}|^2} \widehat{A})^{-1} \widehat{A}^* \frac{1}{1 + |\boldsymbol{\xi}|^2} \widehat{\mathbf{g}}(\boldsymbol{\xi}),$$

$$\widehat{\theta}_{W_{2,\mathrm{UOT}}}(\boldsymbol{\xi}) = \begin{cases} (\widehat{A}^* \widehat{A})^{-1} \widehat{A}^* \widehat{\mathbf{g}}(\boldsymbol{\xi}), & \text{when } \boldsymbol{\xi} = \mathbf{0} \\ (\widehat{A}^* \frac{1}{|\boldsymbol{\xi}|^2} \widehat{A})^{-1} \widehat{A}^* \frac{1}{|\boldsymbol{\xi}|^2} \widehat{\mathbf{g}}(\boldsymbol{\xi}), & \text{when } \boldsymbol{\xi} \neq \mathbf{0} \end{cases}$$

$$\widehat{\theta}_{W_{2,\mathrm{GUOT}}}(\boldsymbol{\xi}) = (\widehat{A}^* \frac{1}{\alpha + |\boldsymbol{\xi}|^2} \widehat{A})^{-1} \widehat{A}^* \frac{1}{\alpha + |\boldsymbol{\xi}|^2} \widehat{\mathbf{g}}(\boldsymbol{\xi}),$$

and

$$\widehat{\theta}_{W_{2,\mathrm{Mixed}}}(\boldsymbol{\xi}) = (\widehat{A}^* \frac{1}{\frac{1}{\beta} + |\boldsymbol{\xi}|^2} \widehat{A})^{-1} \widehat{A}^* \frac{1}{\frac{1}{\beta} + |\boldsymbol{\xi}|^2} \widehat{\mathbf{g}}(\boldsymbol{\xi}).$$

Let $-\Delta$ be the Laplacian operator on Ω with the homogeneous Neumann boundary condition. We can write the above solutions in the physical domain, respectively, as

$$\theta_{W_2}(\mathbf{x}) = \Big(A^* (-\Delta)^{-1} A \Big)^{-1} A^* (-\Delta)^{-1} \mathbf{g}(\mathbf{x}),$$

$$\theta_{W_{2,\mathrm{WFR}}}(\mathbf{x}) = \left(A^*(-\Delta + I)^{-1}A\right)^{-1} A^*(-\Delta + I)^{-1}\mathbf{g}(\mathbf{x}),$$

$$\theta_{W_{2,\mathrm{UOT}}}(\mathbf{x}) = \left(A^*(-\Delta + \int_\Omega)^{-1}A\right)^{-1} A^*(-\Delta + \int_\Omega)^{-1}\mathbf{g}(\mathbf{x})$$

$$\theta_{W_{2,\mathrm{GUOT}}}(\mathbf{x}) = \left(A^*(-\Delta + \alpha I)^{-1}A\right)^{-1} A^*(-\Delta + \alpha I)^{-1}\mathbf{g}(\mathbf{x}),$$

and

$$\theta_{W_{2,\mathrm{Mixed}}}(\mathbf{x}) = \left(A^*(-\Delta + \frac{1}{\beta}I)^{-1}A^{-1}\right)^{-1} A^*(-\Delta + \frac{1}{\beta}I)^{-1}\mathbf{g}(\mathbf{x}).$$

where I is the identity operator. Note again that $-\Delta + I$, $-\Delta + \frac{1}{\beta}I$ and $-\Delta + \int_\Omega$ are all invertible with the homogeneous Neumann boundary condition.

The simple calculations above reveal to us the following facts about linearized inversion with the W_2 metrics. First, in the ideal case when the original problem is uniquely invertible, the data used in the inversion contains no noise, inversion results under all the W_2 metrics are the same and degenerate to the classical L^2 based inversion results $\theta_{L^2}(\mathbf{x}) = (A^*A)^{-1}A^*\mathbf{g}(\mathbf{x})$. Second, inversion methods based on the W_2 metrics are "preconditioned" versions of the classical L^2 inversion. The operators $(-\Delta)^{-1}$, $(-\Delta+I)^{-1}$, $(-\Delta+\int_\Omega)^{-1}$, $(-\Delta+\alpha I)^{-1}$ and $(-\Delta+\frac{1}{\beta}I)^{-1}$ are all smoothing operators that damp the higher-frequency components of the data $\mathbf{g}$ before the inversion operation. The higher the frequency is, the stronger the damping effect will be. Therefore, when high-frequency noise present in the data, they are suppressed before the inversion.

3.5 Numerical implementations

We now present details on our implementation of the inversion methods outlined in the previous section. The algorithms are implemented in the `MATLAB` software with the source codes deposited at `github` [1]. In our numerical simulations in the next three sections, we display computational results for the one-dimensional domain $\Omega = [0, L]$ and the two-dimensional spatial domain $\Omega = [0, L_x] \times [0, L_y]$. The time interval for the transport equation is set as $\mathbb{T} = (0, T]$. The values of L, L_x, L_y and T will be given in later in specific examples.

[1] The github repository for our source codes is at https://github.com/wending1/.

3.5.1 Numerical discretizations

One common practice in the numerical discretization of the transport equation in the formulation of the fluid dynamics is to introduce the variables

$$\mathbf{m} = \rho\boldsymbol{\omega}, \quad \text{and,} \quad \zeta = \rho\zeta \tag{3.51}$$

so that the transport equations in the constraints of the minimization problem become linear. We follow this idea here in our presentation. We now detail the discretization of the Wasserstein-Fisher-Rao formulation $W_{2,\mathrm{WFR}}$. The same type of discretizations are used for the W_2, $W_{2,\mathrm{UOT}}$, $W_{2,\mathrm{GUOT}}$ and $W_{2,\mathrm{Mixed}}$ metrics. Details of numerical discretizations are given in the Appendix A.

3.5.2 Discretization of Fréchet derivatives.

We also need to discretize the Fréchet derivatives involved in (3.23). Those operators can be discretized using the same scheme as we just detailed. However, in our numerical implementation, we use a discretizing-then-optimizing approach for the optimization problem. This means that we formulate the optimization problem for each metric in the discrete setting using the discretization schemes for the transport equation and the cost functionals, that is, (A.3) and (A.13) in dimension one or (A.47) and (A.49) in dimension two. We then form the corresponding discrete version of the optimality condition (3.21). The discretized versions of the operators in the equation for the Newton update direction, (3.23), are then directly derived by taking gradients on the discrete level. To save space, we list the discrete version of all the operators involved in the Appendix. We also point out that in the limit when the sizes of the spatial-temporal grid Δx, $\Delta\mathbf{g}$, Δt go to 0, the discretized operators converge to their continuous correspondences given in the Appendix.

3.5.3 Newton's iteration

The Newton iteration (3.22) is implemented using `MATLAB` with a cubic line search scheme [105] to find the step length ℓ_k at iteration k. The standard Wolfe conditions [105] are imposed on the

line search scheme. More precisely, let $\Psi(\rho, \boldsymbol{\omega}, \zeta, \theta, \lambda) := \|\mathcal{F}\|_{L^2}^2$ be squared L^2 norm of the system (3.21), $p_k = (\rho_k, \boldsymbol{\omega}_k, \zeta_k, \theta_k, \lambda_k)$, and $\delta p_k = (\delta\rho_k, \delta\boldsymbol{\omega}_k, \delta\zeta_k, \delta\theta_k, \delta\lambda_k)$. We look for an ℓ_k that solves the one-dimensional minimization problem

$$\min_{\ell_k > 0} \Psi(p_k + \ell_k \delta p_k), \tag{3.52}$$

and satisfies the conditions:

$$\Psi(p_k + \ell_k \delta p_k) \leq \Psi(p_k) + c_1 \ell_k \nabla\Psi^T(p_k)\delta p_k, \tag{3.53}$$

$$\nabla\Psi^T(p_k + \ell_k \delta p_k)\delta p_k \geq c_2 \nabla\Psi^T(p_k)\delta p_k, \tag{3.54}$$

where c_1 and c_2 are two small positive constants. Following the suggestion in [105], we take $c_1 = 10^{-4}$ and $c_2 = 0.1$ in our numerical simulations in Section 3.8.

Initialization guesses. The Newton iteration needs to be started at a given initial guess $(\rho_0, \boldsymbol{\omega}_0, \zeta_0, \theta_0, \lambda_0)$. Since θ is the only intrinsic variable that we are interested in inverting for, we should only need to provide the initial guess θ_0. The initial guess for the other variables are constructed by solving the sub-optimization problem described by (3.5) and (3.6) with $\mathbf{f}(\theta)$ fixed at $\mathbf{f}(\theta_0)$. This way, the different components of the initial guess $(\rho_0, \boldsymbol{\omega}_0, \zeta_0, \theta_0, \lambda_0)$ for the nonlinear iteration are consistent with each other.

Stopping criteria. In the numerical simulations, we will compare results from the algorithms with different W_2 metrics. One of the major difficulties in making a fair comparison between different algorithms is to find a fair stopping criterion. In our simulations, we stop the iterations if either (i) the update is sufficiently small, that is, $\ell_k \|(\delta\rho_k, \delta\boldsymbol{\omega}_k, \delta\zeta_k, \delta\theta_k, \delta\lambda_k)\| \leq \varepsilon_1$ for a given tolerance ε_1, or (ii) the mismatch between the model prediction and the data has decayed sufficiently, that is $\Phi(\mathbf{f}(\theta_k), \mathbf{g}) \leq \varepsilon_2 \Phi(\mathbf{f}(\theta_0), \mathbf{g})$ (where θ_0 is the initial guess of the unknown function to be reconstructed) for some tolerance ε_2. The value of the parameters ε_1 and ε_2 will be provided later

when we present the simulation results.

3.5.4 General setup for simulations

We focus on three inverse problems where the data measured are non-negative so that we do not need to perform signal normalization, an operation that could have a significant impact on the inversion results, as is done in most of the previous studies [148, 40, 147, 146].

Inverting the Abel transform. The first inverse problem we consider is a linear problem described by the Abel equation. For a continuous function θ on $[0, 1]$, the Abel transform of θ, with parameter $0 < \alpha < 1$, is defined as [66]:

$$\mathbf{f}(s) \equiv A\theta := \int_0^s (s-t)^{-\alpha}\theta(t)dt, \qquad s \in [0, 1]. \tag{3.55}$$

Without loss of generality, we can assume that $\mathbf{f}(0) = 0$. In our numerical simulations later, we are interested in studying the performance of the Wasserstein metrics in inverting the Abel transform. The discretization scheme for the Abel transform is documented in the Appendix A.2. Interested readers are referred to [12] for more detailed discussions on equations of Abel type. Let us mention here the simple fact that the Abel transform can be analytically inverted to find $\theta(t)$(with $\mathbf{f}(0) = 0$):

$$\theta(t) = \frac{\sin(\pi\alpha)}{\pi}\Big(\int_0^t \frac{d\mathbf{f}(s)}{ds}\frac{1}{(t-s)^{1-\alpha}}ds\Big), \qquad t \in [0, 1]. \tag{3.56}$$

Inverse diffusion problem. The second example inverse problem we will computationally study is an inverse coefficient problem for the diffusion equation in a bounded domain $\Omega \subset \mathbb{R}^d$ $(d \geq 1)$ with smooth boundary $\partial\Omega$:

$$-\nabla \cdot \gamma\nabla u + \theta u = q, \ \text{ in } \Omega, \qquad u = b, \ \text{ on } \partial\Omega \tag{3.57}$$

where γ is the diffusion coefficient and θ is the absorption coefficient. When the coefficients and the boundary condition $\mathbf{g}$ are given, we can solve the diffusion equation to find its solution u. In the

inverse problem, we assume that we know everything else but not the absorption coefficient θ. We are interested in reconstructing θ from additional data of the form

$$\mathbf{f}(\theta) := \Lambda(\theta)u, \quad \in \Omega \tag{3.58}$$

where γ is the diffusion coefficient, and θ is the absorption coefficient. When the coefficients and the boundary condition $\mathbf{g}$ are given, we can solve the diffusion equation to find its solution u. In the inverse problem, we assume that we know everything else but not the absorption coefficient θ. We are interested in reconstructing θ from additional data of the form

Inverse wave propagation. The third inverse problem we consider is an inverse coefficient problem for the Helmholtz equation in a bounded domain:

$$\Delta u + k^2\big(1+n\big)u + ik\theta u = q, \quad \text{in } \Omega, \qquad u = b, \quad \text{on } \partial\Omega \tag{3.59}$$

where k is the wave number, n is the refractive index, and θ is the conductivity. We assume that wave number k, the coefficients, and the domain Ω are arranged in such a way that the Helmholtz equation (3.59) is uniquely solvable. In the inverse problem, we assume that we know n but not θ. We are interested in recovering θ from additional data of the form

$$\mathbf{f}(\theta) = \Lambda(\theta)|u|^2, \quad \mathbf{x} \in \Omega. \tag{3.60}$$

This is a simplified model for inverse problems in quantitative thermoacoustic tomography [8]. We discretize the Helmholtz model in a rectangular domain in our numerical simulations. The discretization scheme can be found in Appendix A.2.

Generation of synthetic data. The synthetic data we use in all the numerical simulations are generated from the forward models. We solve the forward model $\mathbf{g} = \mathbf{f}(\theta_{true})$ with the true coefficients to compute the data. We add additive random noise to the data computed from

the models to generate noisy data. More precisely, let $\mathbf{g}$ be the noise-free data. We generate multipliable noisy data $\mathbf{g}^\delta = \mathbf{g} + h$ where $h \sim \mathcal{N}(0, \delta\ diag(g))$ (δ being the variance of the noise). To generate high-frequency noise, we use Fourier transform $\mathcal{F}$ to filter out the low-frequency components of the noise. Let h be the random noise, we generate the high-frequency noise $\widetilde{h}$ by $\widetilde{h} := \mathcal{F}^{-1}\chi_{\{\xi||\xi|\geq\xi_0\}}\mathcal{F}(h)$ where $\mathcal{F}^{-1}$ denote the inverse Fourier transform, χ_B is the characteristic function of the set B and ξ_0 denotes the cutoff frequency.

Our numerical experiments show that the time interval size does not play a significant role in the solution process. In all the numerical experiments in the following subsections, we set $T = 1$ in the definition of the Wasserstein metrics for simplicity. We observe that T does not affect the reconstruction quality (although it does affect the overall computational cost of the reconstructions). *Unless stated otherwise, we set the regularization parameter $\gamma = 0$, introduced in* (3.15)*, in all the simulations as our objective in this study is to study the general behavior of the Wasserstein metrics in the inversion process, not to tune the parameters to get the best reconstruction result.*

The unknown in the following section are from one of the three options 3.1

Bell shape θ. The unknown is smooth. It tests the general ability of Wasserstein metrics to solve inverse problems.

$$\theta(x) = 1 + e^{-\frac{(2x-1)^2}{0.05}}, \quad x \in [0, 1] \tag{3.61}$$

Two scale θ. Unknown consists two different scales of modes, one high-frequency mode 60π and one low-frequency mode 2π. This example may be extreme, but it is a good indicator of the separation of frequencies by Wasserstein metrics.

$$\theta(x) = 1 + 0.5\sin(2\pi x) + 0.05\sin(30\pi x), \quad x \in [0, 1] \tag{3.62}$$

Discontinuous θ. The unknown is the simplest discontinuous function. It aims to test the performances for Wasserstein metrics in discontinuous settings.

$$\theta(x) = 1 + \mathcal{X}_{[\frac{1}{3},\frac{2}{3}]}, \quad x \in [0,1] \tag{3.63}$$

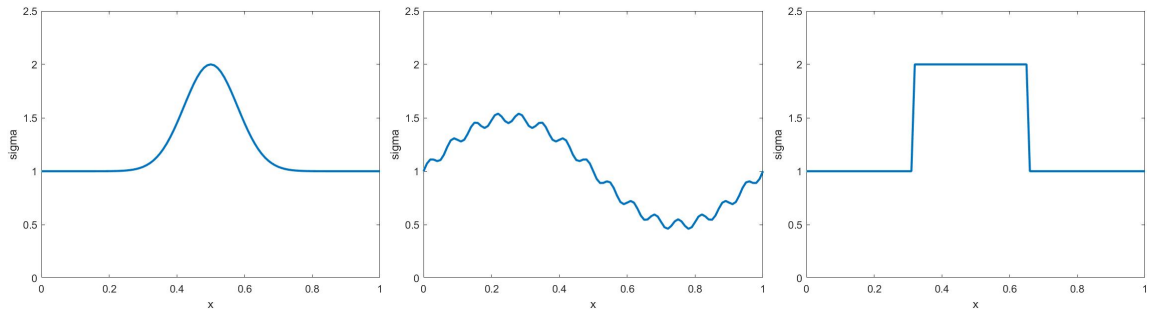


Figure 3.1: Exact unknown shape, from left to right: bell shape, two scale shape, discontinuous shape.

Display of optimization process. To fully display properties(especially smoothing effect and separation of frequencies) of Wasserstein metrics, it is vital that we display target quantities throughout the optimization process. Therefore, we organize the reconstruction results from different algorithms and setups by their loss value: the value of the current loss function, i.e., $loss = \Phi(\mathbf{f}(\theta_k), \mathbf{g})$ rather than simply showing the final reconstruction results. We regard the loss value as a fair comparison between different Wasserstein metrics.

Initial guess of numerics. Unless stated otherwise, we set the initial guess of independent variables of all the following numerical examples to be constant 1. Other related quantities are initialized such that they are consistent.

3.6 W_2 does NOT regularize.

The smoothing effect of the Wasserstein metrics we have just seen should not be confused with the smoothing effect introduced by variational regularization techniques such as Tikhonov

regularization, which is often introduced in L^2 based least-squares formulation. For the linear model (3.44), we minimize

$$\Phi_{L^2}(\theta) := \frac{1}{2}\sum_{\boldsymbol{\xi}\in\mathbb{Z}_+^d} |\widehat{A\theta}(\boldsymbol{\xi}) - \widehat{\mathbf{g}}(\boldsymbol{\xi})|^2 + \frac{\gamma}{2}\sum_{\boldsymbol{\xi}\in\mathbb{Z}_+^d}(1+|\boldsymbol{\xi}|^2)|\widehat{\theta}(\boldsymbol{\xi}) - \widehat{\theta}_0(\boldsymbol{\xi})|^2, \tag{3.64}$$

where θ_0 represents the *a priori* selected and γ controls of the strength of imposing the a priori.

It is straightforward to verify that the minimizing $\Phi_{L^2}(\theta)$ gives the solution

$$\widehat{\theta}_{L^2}(\boldsymbol{\xi}) = \left(\widehat{A}^*\widehat{A} + \gamma(1+|\boldsymbol{\xi}|^2)\right)^{-1}\widehat{A}^*\widehat{\mathbf{g}}(\boldsymbol{\xi}) \tag{3.65}$$

The insights we obtained in this section are analyzed. A nonlinear effect makes the Wasserstein distances different from the $\mathcal{H}^{-1}$ metrics [107].

3.7 Performance under noisy data

The pre-conditioning effect of the Wasserstein metrics on the inversion results has significant consequences for inverse problems with noisy data. This was discussed in [95, 40, 148, 37, 146] in the case of the regular W_2. The same analysis carries to $W_{2,\mathrm{WFR}}$, $W_{2,\mathrm{UOT}}$ and $W_{2,\mathrm{mixed}}$. Let us assume further that the linear operator A has the symbol

$$\widehat{A}(\boldsymbol{\xi}) = \langle\boldsymbol{\xi}\rangle^{-\theta}, \qquad \langle\boldsymbol{\xi}\rangle = \sqrt{1+|\boldsymbol{\xi}|^2}, \quad \theta > 0. \tag{3.66}$$

The assumption that $\theta > 0$ means that the operator is smoothing whose inverse amplifies the high-frequency component of the data. The inverse problem is, therefore, ill-conditioned (often called ill-posed in the literature).

We denote by $\mathbf{g}^\delta$ the data $\mathbf{g}$ polluted by random noise with δ the noise level in the data. More precisely, we denote by $\delta := \|\mathbf{g}^\delta - \mathbf{g}\|_{L^2(\Omega)}$. We assume that $\mathbf{g}^\delta$ and $\mathbf{g}$ have the same total mass so that we do not need to worry about the invertibility issue of W_2 inversion. We also denote by $R_{W_2}^{|\boldsymbol{\xi}|_c}$

the inversion operator under the W_2 metric truncated at frequency $|\boldsymbol{\xi}|_c$, that is,

$$R_{W_2}^{|\boldsymbol{\xi}|_c}\mathbf{g} := \sum_{\boldsymbol{\xi}\in\mathbb{Z}_+}^{|\boldsymbol{\xi}|\le|\boldsymbol{\xi}|_c} (\widehat{A}^*\frac{1}{|\boldsymbol{\xi}|^2}\widehat{A})^{-1}\widehat{A}^*\frac{1}{|\boldsymbol{\xi}|^2}\widehat{\mathbf{g}}(\boldsymbol{\xi})\cos(\boldsymbol{\xi}\cdot\mathbf{x})$$

Then the L^2 error in the reconstruction, defined as the L^2 difference between the reconstruction from data $\mathbf{g}^\delta$, $\widetilde{\theta}_{W_2}$, and the true solution θ, can be bounded as follows:

$$\begin{aligned}\|\widetilde{\theta}_{W_2}^\delta - \theta\|_{L^2(\Omega)} = \|\theta_{W_2}^\delta - \theta_{W_2} + \theta_{W_2} - \theta\|_{L^2(\Omega)} &\le \|\theta_{W_2}^\delta - \theta_{W_2}\|_{L^2(\Omega)} + \|\theta_{W_2} - \theta\|_{L^2(\Omega)} \\ &= \|R_{W_2}^{|\boldsymbol{\xi}|_c}(\mathbf{g}^\delta - \mathbf{g})\|_{L^2(\Omega)} + \|(R_{W_2}^{|\boldsymbol{\xi}|_c}A - I)\theta\|_{L^2(\Omega)}. \quad (3.67)\end{aligned}$$

On the other hand, it is straightforward to check that

$$\begin{aligned}&\|R_{W_2}^{|\boldsymbol{\xi}|_c}(\mathbf{g}^\delta - \mathbf{g})\|_{L^2(\Omega)} = \|\sum_{\boldsymbol{\xi}\in\mathbb{Z}_+}^{|\boldsymbol{\xi}|\le|\boldsymbol{\xi}|_c} (\widehat{A}^*\frac{1}{|\boldsymbol{\xi}|^2}\widehat{A})^{-1}\widehat{A}^*\frac{1}{|\boldsymbol{\xi}|^2}(\widehat{\mathbf{g}}^\delta - \widehat{\mathbf{g}})(\boldsymbol{\xi})\cos(\boldsymbol{\xi}\cdot\mathbf{x})\|_{L^2(\Omega)} \\ &= \|\sum_{\boldsymbol{\xi}\in\mathbb{Z}_+}^{|\boldsymbol{\xi}|\le|\boldsymbol{\xi}|_c} \langle\boldsymbol{\xi}\rangle^\theta|\boldsymbol{\xi}|\frac{(\widehat{\mathbf{g}}^\delta - \widehat{\mathbf{g}})(\boldsymbol{\xi})}{|\boldsymbol{\xi}|}\cos(\boldsymbol{\xi}\cdot\mathbf{x})\|_{L^2(\Omega)} \le |\boldsymbol{\xi}|_c^{\theta+1}\sum_{\boldsymbol{\xi}\in\mathbb{Z}_+}\frac{1}{|\boldsymbol{\xi}|^2}\widehat{|\mathbf{g}^\delta - \mathbf{g}|}^2 \le |\boldsymbol{\xi}|_c^{\theta+1}\|\mathbf{g}^\delta - \mathbf{g}\|_{L^2(\Omega)}\end{aligned}$$

where the last inequality is true with the assumption that $\mathbf{g}^\delta$ and $\mathbf{g}$ have the same total mass, and also that

$$\|(R_{W_2}^{|\boldsymbol{\xi}|_c}A - I)\theta\|_{L^2(\Omega)} = \|\sum_{\boldsymbol{\xi}\in\mathbb{Z}_+,|\boldsymbol{\xi}|\ge|\boldsymbol{\xi}|_c}\widehat{\theta}(\boldsymbol{\xi})\cos(\boldsymbol{\xi}\cdot\mathbf{x})\|_{L^2(\Omega)} = \sum_{\boldsymbol{\xi}\in\mathbb{Z}_+,|\boldsymbol{\xi}|\ge|\boldsymbol{\xi}|_c}|\widehat{\theta}(\boldsymbol{\xi})|^2.$$

Therefore, the reconstruction error bound (3.67) can be rewritten as

$$\|\widetilde{\theta}_{W_2}^\delta - \theta\|_{L^2(\Omega)} \le |\boldsymbol{\xi}|_c^{\theta+1}\delta + \sum_{\boldsymbol{\xi}\in\mathbb{Z}_+,|\boldsymbol{\xi}|\ge|\boldsymbol{\xi}|_c}|\widehat{\theta}(\boldsymbol{\xi})|^2. \quad (3.68)$$

This error bound is minimized if we take $|\boldsymbol{\xi}|_c \sim (\delta^{-1}|\widehat{\theta}(\boldsymbol{\xi})|)$.

Remark 3.7.1. *The simple analysis in this section: We emphasize that one of the main differences between $W_{2,\mathrm{WFR}}$, $W_{2,\mathrm{UOT}}$, and $W_{2,\mathrm{Mixed}}$ is that $W_{2,\mathrm{WFR}}$ has no free parameters in its definition. The*

mass imbalance is completely determined by $\mathbf{f}$ *and* $\mathbf{g}$*. In* $W_{2,\text{Mixed}}$*, however, the imbalance is not completely decided by the imbalance between* $\mathbf{f}$ *and* $\mathbf{g}$*. The parameter* β *allows the imposition of mass conservation. In the solution of inverse problems,* β *should go to* ∞ *to enforce that the final solution conserves the total mass. If the total mass of* $\mathbf{f}(\theta)$ *and* $\mathbf{g}$ *are not the same,* θ *is not the true solution to the inverse problem. We should set the sequence* β_k *to* ∞ *as* $k \to \infty$*.*

3.8 Numerical experiments

We now present a collection of numerical simulations to study the behavior of the Wasserstein metrics in solving inverse problems.

Before introducing any numerical results, we first plot the loss $\Phi(\theta)$ with respect to optimization iterations in figure 3.2. It decays exponentially regardless of the choice of algorithms, forward operators, and input θ shape.

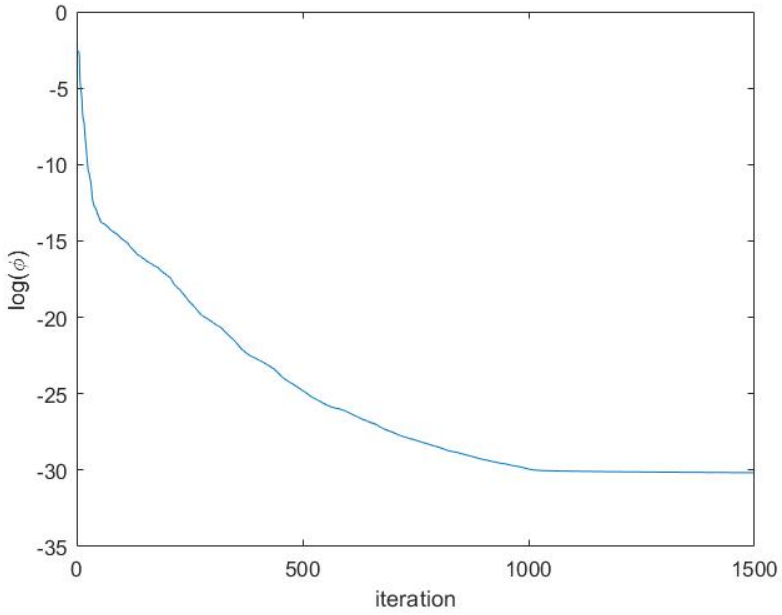


Figure 3.2: Optimization Loss $\log(\Phi(\theta))$.

3.8.1 The smoothing effect

In the first set of numerical experiments, we focus on the smoothing effect of Wasserstein metrics as analyzed in the linearization study of Section 3.4. The Wasserstein metrics are asymptotically equivalent to weighted $\dot{\mathcal{H}}^{-1}$ and $\mathcal{H}^{-1}$ metrics. These are weaker metrics compared to the L^2 metric. Therefore, inversion based on those metrics is more stable with respect to random noise presented in the data. This is evident in the inversion formulas given in linearization analysis 3.4.2 as high-frequency modes in the data are damped by the factor $\langle \boldsymbol{\xi} \rangle^{-1}$ before they pass through the inverse operators. Effectively the reconstruction consists of several low-frequency modes.

Experiment 1. In Figure 3.3, we exhibit reconstruction results for the inverse diffusion problem in the one-dimensional domain $\Omega = (0, 1)$ with data containing 10% of random Gaussian noise. The results show that Wasserstein metrics are less sensitive to noise than L^2 loss function.

3.8.2 Frequencies disparity

Experiment 2. In the following example, we showcase that while it is excellent that the Wasserstein metrics stabilize the reconstructions with noisy data, they also delay the reconstruction of the high-frequency component of the unknown. In Figure 3.4 and 3.5, we exhibit the reconstruction of the absorption coefficient 3.62 which has a high-frequency component that is far separated from its low-frequency component. The reconstructions show that Wasserstein metrics mainly delay the reconstruction of the high-frequency mode of the unknown. Because Wasserstein metrics recover low-frequency functions faster than high-frequency functions, therefore, if the function can be approximated with low-frequency components, the Wasserstein metric will recover it with low-frequency components as opposed to recovering with a high-frequency, noise-like function. Thus reducing the effect of high-freuqncy noises.

Moreover, we can plot the relative error of Fourier modes against optimization iterations, as in figure 3.6. Mode 2π decays to zero after iteration 30 and stays zero after that. While the relative error for mode 30π first increases and then decays to zero after iteration 500. This shows clearly

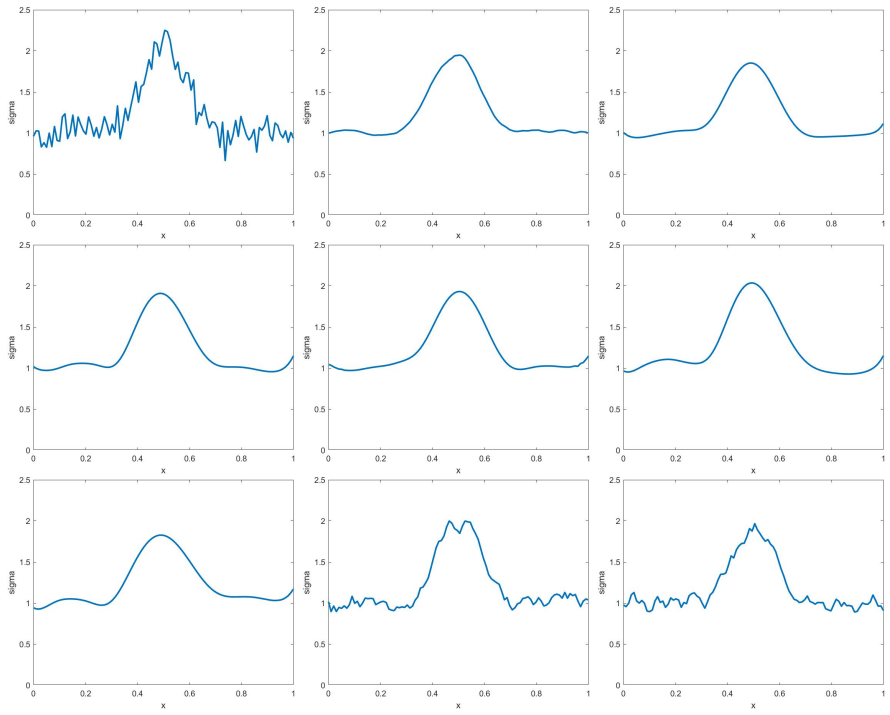


Figure 3.3: Reconstructing the bell shape absorption coefficient θ in 3.61 in the diffusion equation (3.57) in the one-dimensional domain $\Omega = (0, 1)$. First row from left to right : the exact L^2, $\mathcal{H}^{-1}$, W_2; Second row from left to right: $W_{2,\text{WFR}}$, $W_{2,\text{UOT}}(\frac{1}{\alpha} = 0.1)$, $W_{2,\text{GUOT}}(\frac{1}{\alpha} = 0.1)$; Third row from left to right: $W_{2,\text{Mixed}}(\beta = 0.1)$, W_1, $UW_1(\beta = 0.1)$. The synthetic data contains 10% of random noise, loss value is 10^{-5} for quadratic Wasserstein metrics, loss value is 10^{-3} for W_1 and UW_1 results.

that Wasserstein metrics delay the reconstruction of higher frequency components. Eventually, Wasserstein metrics still recover high-frequency components.

3.8.3 The effect of mass imbalance

Experiment 3. The rationale for using unbalanced optimal transport is that during the iterations before convergence, one should not expect that the model predictions match the measured data, that is, $\mathbf{f}$ does not have to have the same mass as $\mathbf{g}$. It's therefore more appropriate to use unbalanced optimal transport instead of balanced optimal transport. However, at the point of convergence, i.e. when θ is close to its true value, $\mathbf{f}$ and $\mathbf{g}$ should have the same mass. Therefore, the final results of unbalanced and balanced inversion are very close.

Among the unbalanced metrics, the $W_{2,\text{WFR}}$ metric will not degenerate to W_2 when $\mathbf{f}$ and $\mathbf{g}$ have

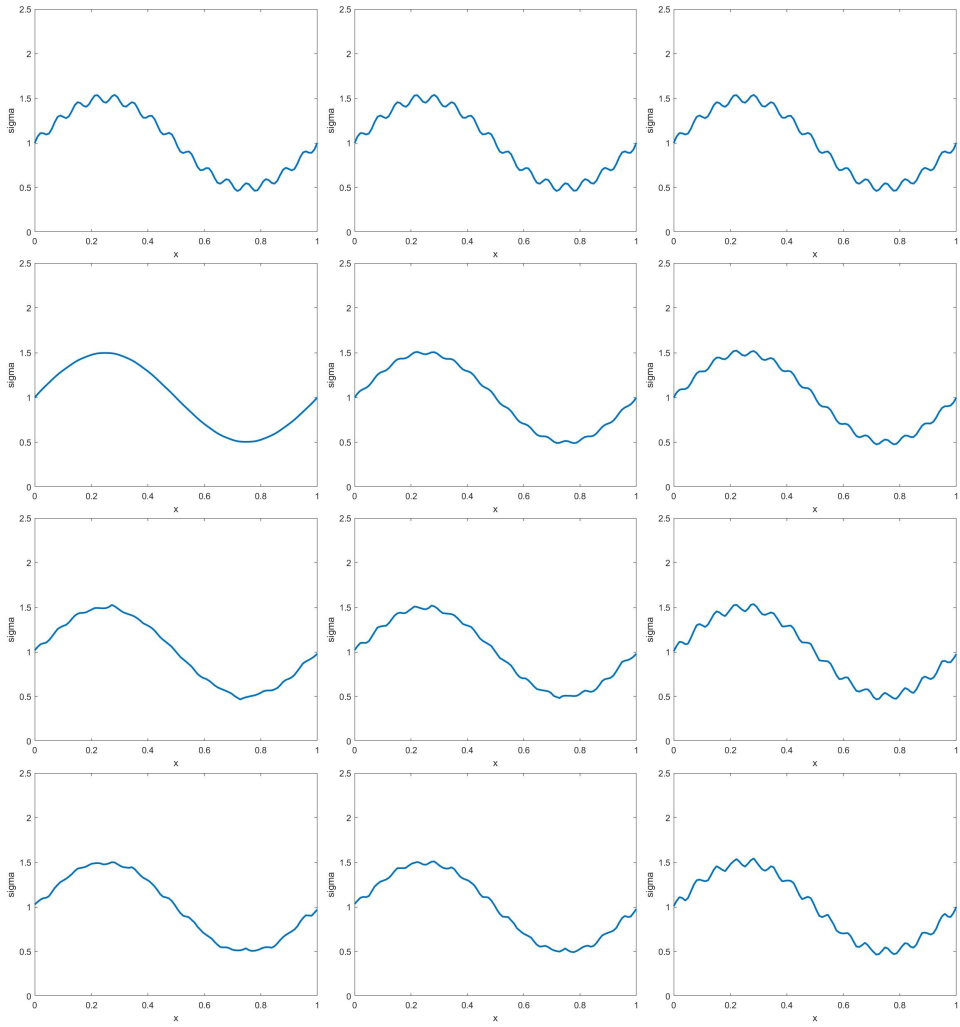


Figure 3.4: Reconstruction of the two scale coefficient θ given in (3.62) for Helmholtz model (3.59). The synthetic data contains no random noise. From top to bottom: $L^2, \mathcal{H}^{-1}, W_1, UW_1(\beta = 0.1)$. From left to right: loss value for $\mathcal{H}^{-1}$ is $5 * 10^{-6}, 2 * 10^{-6}, 10^{-6}$; loss value for L^2 is $10^{-1}, 10^{-2}, 10^{-3}$; loss value for W_1, UW_1 is $0.002, 0.0015, 5 * 10^{-4}$.

the same total mass, $W_{2,\text{UOT}}$ and $W_{2,\text{GUOT}}$ degenerate to W_2 when $\mathbf{f}$ and $\mathbf{g}$ have the same total mass, independent of the parameter α. The $W_{2,\text{Mixed}}$ metric will reduce to W_2 when β goes to 0. In our implementation, we take a sequence of parameter β_k (k being the iteration step of the algorithm) that goes to decays to 0 as k increases.

In our optimization framework, mass imbalance means the constraints have an empty feasible set, dealing with mass imbalance is equivalent to relaxing the constraints such that the feasible set is nonempty. The dealing of imbalance is embedded into the optimization procedure. As we can see

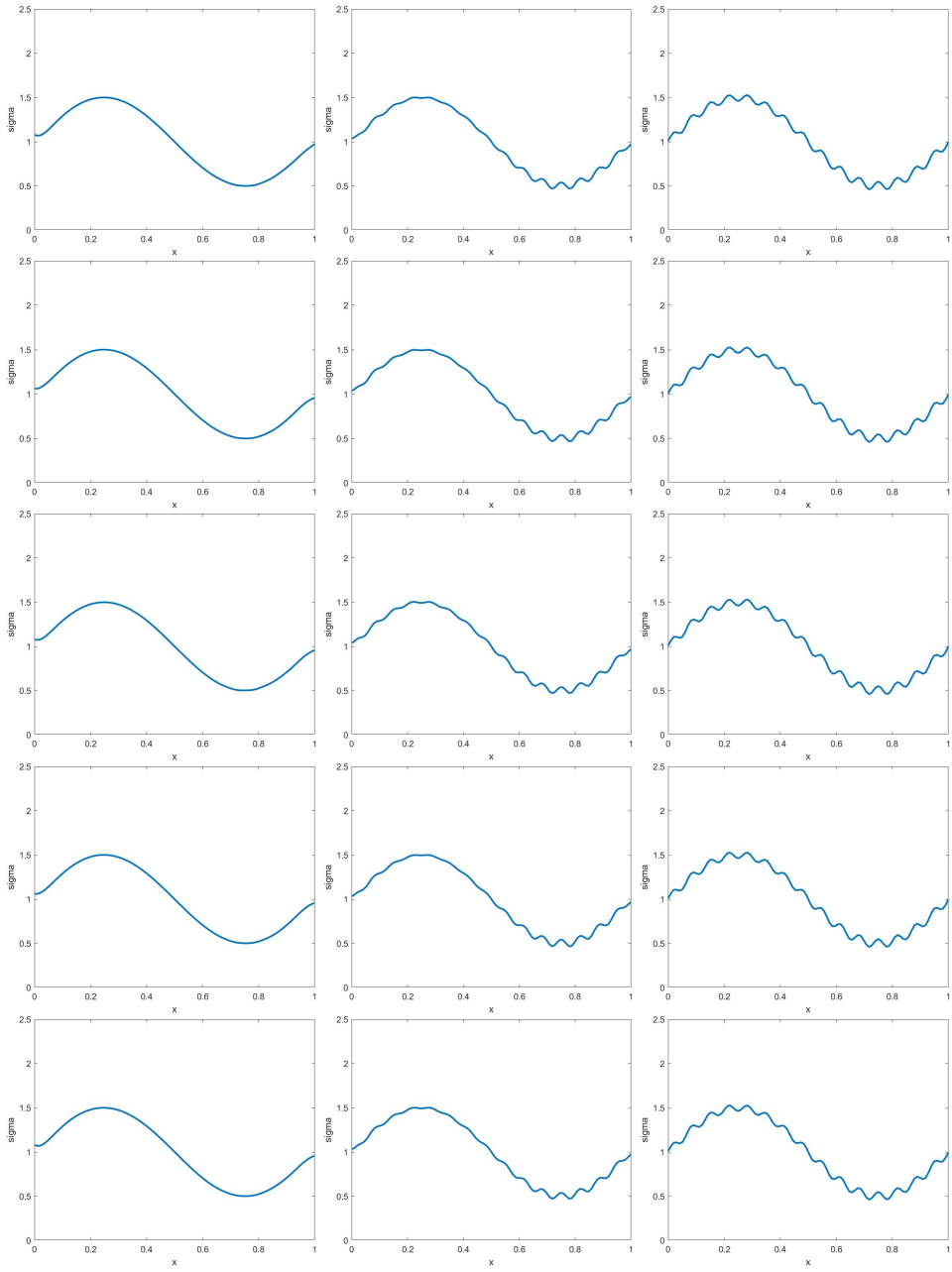


Figure 3.5: Reconstruction of the two scale coefficient θ given in (3.62) for Helmholtz model (3.59). The synthetic data contains no random noise. From top to bottom: $W_2, W_{2,\mathrm{WFR}}, W_{2,\mathrm{UOT}}(\frac{1}{\alpha} = 0.1), W_{2,\mathrm{GUOT}}(\frac{1}{\alpha} = 0.1), W_{2,\mathrm{Mixed}}(\beta = 0.1)$. From left to right: loss value for all W_2 types of metrics is $10^{-6}, 2 * 10^{-7}, 10^{-8}$.

from the results in Figure 3.7, we do not observe significant differences between reconstructions with $W_{2,\mathrm{WFR}}$, $W_{2,\mathrm{GUOT}}$, $W_{2,\mathrm{UOT}}$, $W_{2,\mathrm{Mixed}}$ and those with W_2, see figure 3.7. This is expected from the analysis in the asymptotic regime as the synthetic data we used in the inversion have the same

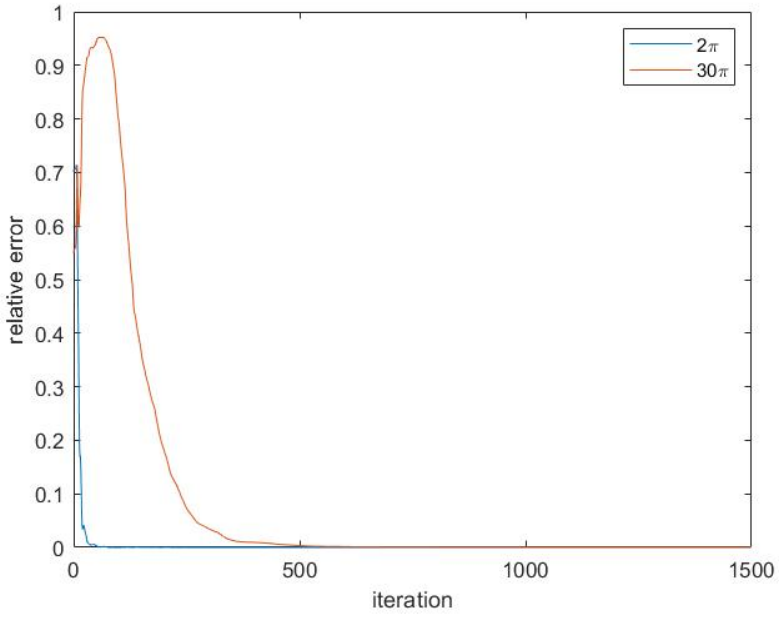


Figure 3.6: The same setup as 3.4 with $W_{2,\text{GUOT}}$ algorithm. Relative error for reconstructed θ coefficients i.e. $\frac{\int_0^1 |\theta(x)-\theta_{exact}(x)|\sin(n\pi x)dx}{\int_0^1 |\theta_{exact}(x)|\sin(n\pi x)dx}, n \in \{2, 30\}$.

total mass as the model predictions with the true coefficients.

3.8.4 Impact of penalty parameters

Experiment 4. The penalty coefficient is actually very important in $W_{2,\text{GUOT}}$, $W_{2,\text{UOT}}$, $W_{2,\text{Mixed}}$ and UW_1 when **f** and **g** does not have the same total mass. The penalty term controls the degree to which we enforce the dynamic source term to be as small as possible.

Intuitively speaking, the initial condition of the transport solution should start from **f** and end at **g**, the initial constraint ensures the accuracy of the reconstruction. The dynamic source term of $W_{2,\text{GUOT}}$, $W_{2,\text{UOT}}$ offers the flexibility of enforcing this constraint. A large penalty parameter puts a large weight on the dynamic source term, and smooths the reconstruction, making it more difficult to reconstruct high-frequency components; Moderate penalty parameter can recover both high and low-frequency components;. A small penalty parameter will weaken the enforcement of initial conditions, resulting in systematic reconstruction error, even in low Fourier modes, leading to totally wrong results. It is shown in figure 3.8 that as $\frac{1}{\alpha}$ decreases to 0, the ability to reconstruct

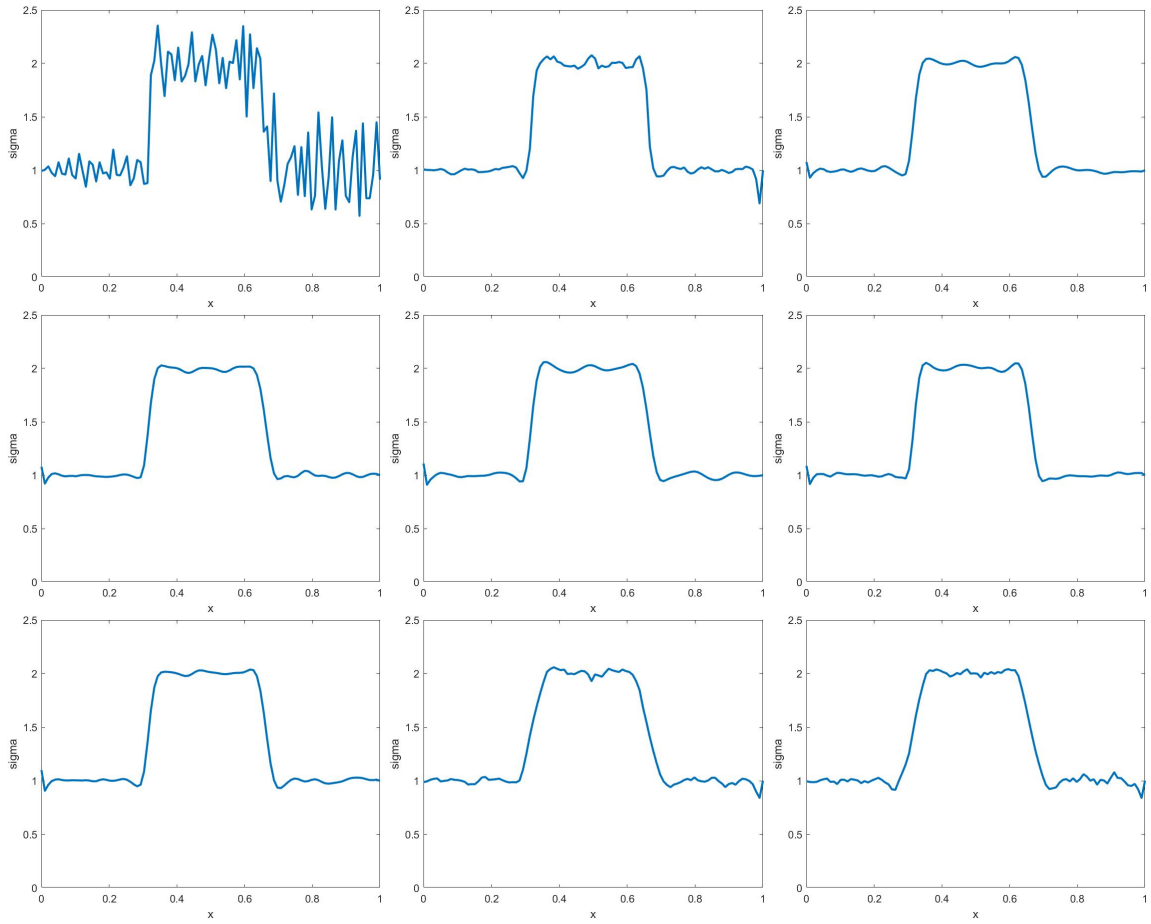


Figure 3.7: Reconstruction of the discontinuous absorption coefficient θ given in (3.63) for Abel transform problem (3.55). The synthetic data contains 1% random noise. First row: $L^2, \mathcal{H}^{-1}, W_2, W_{2,\mathrm{WFR}}$; Second row: $W_{2,\mathrm{UOT}}(\frac{1}{\alpha} = 0.1), W_{2,\mathrm{GUOT}}(\frac{1}{\alpha} = 0.1)$; Third row: $W_{2,\mathrm{Mixed}}(\beta = 0.1), W_1, UW_1(\beta = 0.1)$. All quadratic Wasserstein metrics results are plotted with loss value 10^{-7}; while W_1, UW_1 are plotted with loss value 10^{-3}.

high-frequency modes is stronger, but when $\frac{1}{\alpha}$ is too small, the reconstruction fails completely.

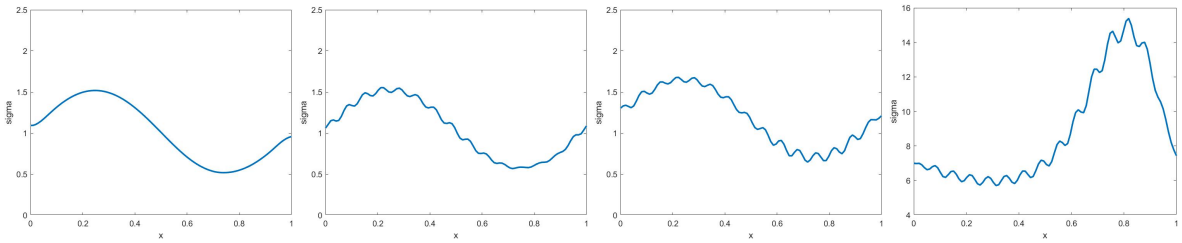


Figure 3.8: Reconstruction of the two scale coefficient θ given in (3.62) for Helmholtz model (3.59) with $W_{2,\mathrm{GUOT}}$. The synthetic data contains no random noise. From left to right: $\frac{1}{\alpha}$ is $10^{-3}, 10^{-4}, 10^{-5}, 10^{-8}$. All plotted with loss value 10^{-5}. Notice that when $\frac{1}{\alpha} = 10^{-8}$, the plot has a different scale in y axis from other plots.

This phenomenon is more evident in the case of UW_1, see figures 3.9. As the penalty parameter β decreases, the high-frequency components are more and more accurate, while the low-frequency components are compressed. Therefore the penalty parameter β in UW_1 can be used to adjust the reconstruction sensitivity to high or low-frequency components.

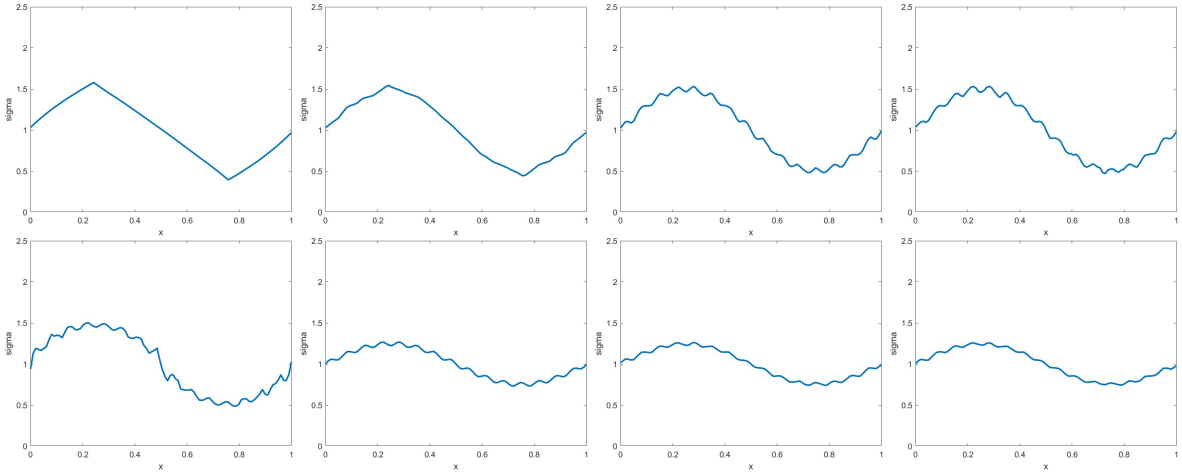


Figure 3.9: Reconstruction of the two scale coefficient θ given in (3.62) for Helmholtz equation (3.59) with UW_1. The synthetic data contains no random noise. First row: β is $1, 0.5, 0.1, 0.05$; Second row: β is $10^{-2}, 10^{-3}, 10^{-4}, 10^{-8}$. All plotted with relative loss value 10^{-3}.

3.8.5 Impact of initial guess

Experiment 5. It is impossible to get a fair comparison between the metrics on this since all the convergences are local. Our initial guesses have to be close enough to the true. In our numerical tests, we observe in general. However, the algorithms based on the W_2 metrics are not sensitive to initial guesses, especially when the initial guess has high-frequency components, for instance, piecewise constant initial guesses. This is in general our experiences in computational inverse problems; Figure 3.10 is one typical example of a computation process from different initial guesses. And 3.11 show the impact of different initial guesses on the final reconstruction of Wasserstein metrics. Although the path to reconstruction is different, the final results and the "speed" of reconstruction are almost the same for piecewise and constant initial guess.

Here we use two initial guess,constant initial guess: $\theta_{initial} = 1$ and piecewise initial guess

$$\theta_{initial}(x) = \begin{cases} 1 & x \in [0, \frac{1}{6}) \\ 1.5 & x \in [\frac{1}{6}, \frac{2}{6}) \\ 0.5 & x \in [\frac{2}{6}, \frac{3}{6}) \\ 1 & x \in [\frac{3}{6}, \frac{4}{6}) \\ 1.5 & x \in [\frac{4}{6}, \frac{5}{6}) \\ 0.5 & x \in [\frac{5}{6}, 1] \end{cases} \tag{3.69}$$

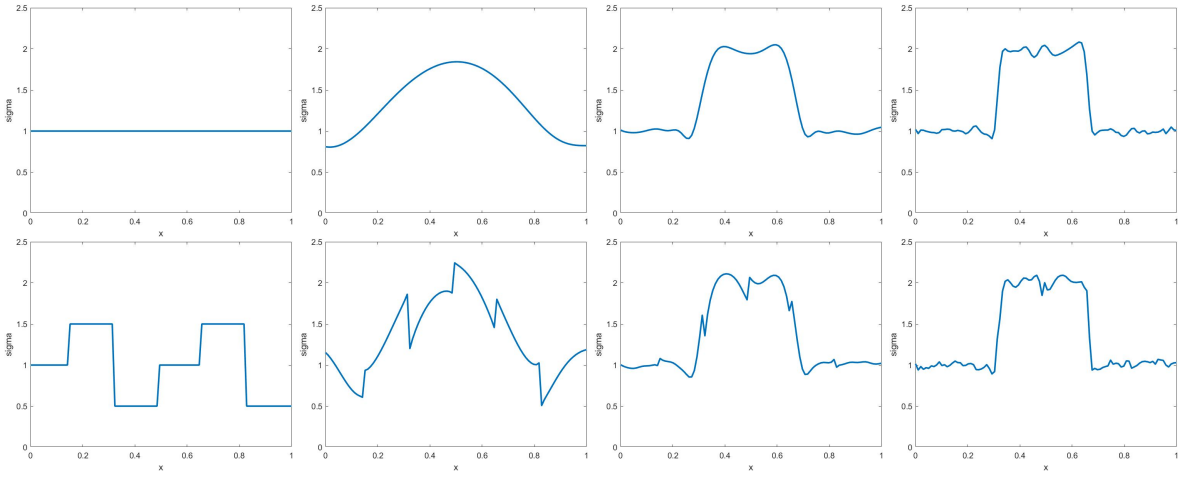


Figure 3.10: Reconstruction of the discontinuous coefficient θ given in (3.63) for Helmholtz model (3.59) with $W_{2,\mathrm{GUOT}}$. The synthetic data contains 5% random noise. First row: reconstruction process with constant initial guess; Second row: reconstruction with piecewise constant initial guess 3.69. From left to right: initial guess, β is $10^{-3}, 10^{-4}, 10^{-5}, 10^{-6}$.

3.8.6 Two dimensional simulations

Experiment 6. What we have observed in the previous numerical experiments also appears in problems in higher-dimensional spaces. This is demonstrated in Figure 3.14 3.13 3.15 3.16 where we show reconstruction results for the inverse diffusion problem in the two-dimensional domain $\Omega = [0, 1] \times [0, 1]$. The domain is covered with a uniform spatial mesh and the time interval

$t \in [0, T = 1]$ is covered with uniform mesh intervals. Our numerical experiences show that this phenomenon occurs independently of the exact form of the true coefficient σ. Exact discretization is in appendix A.12. We use the 2D diffusion equation as forwarding operators 3.57. Same as the 1D examples, we use similar three types of target quantities σ as follows 3.12

Continuous σ.

$$\sigma(x, y) = 1 - x(1 - x)sin(6\pi xy)sin(2\pi y)e^{x-y}, \quad (x, y) \in [0, 1] \times [0, 1] \tag{3.70}$$

Discontinuous σ.

$$\sigma(x, y) = 1 + \mathcal{X}_{(x,y)\in[\frac{1}{3},\frac{2}{3}]\times[\frac{1}{3},\frac{2}{3}]}, \quad (x, y) \in [0, 1] \times [0, 1] \tag{3.71}$$

Two scale σ.

$$\sigma(x, y) = 1 + 0.5\sin(2\pi x)\sin(2\pi y) + 0.1\sin(10\pi x)\sin(10\pi y), \quad (x, y) \in [0, 1] \times [0, 1] \tag{3.72}$$

Our numerical experiences show that two-dimensional Wasserstein metrics are similar to one-dimensional cases. We showcase the following three numerical examples for two-dimensional reconstruction: Discontinuous shape with noise 3.13, the continuous case 3.14 and the two scale cases for separation of frequencies 3.15 3.16.

3.8.7 Further discussions

Ill-conditioningness and regularity assumptions. As we have seen from the simplified analysis of linear inverse problems in the previous section, the quality of the inversion is determined by the combined effects of the Wasserstein metrics, the smoothing properties of the forward operators, and the regularity assumptions we impose on the unknown to be reconstructed. When the operator is very smooth or the unknown is assumed to be very smooth, the role of the Wasserstein is less prominent.

Disparity of frequencies. Due to the fact that high-frequency are damped, low-frequency components of the model prediction and data have a larger role in the objective function. Therefore, the gradient weighted more on the low-frequency part. The algorithm therefore matches low-frequency information first and then matches high-frequency information. This is very different from the L^2 case, where high-frequency information dominates the gradients, resulting in the algorithm trying to match high-frequency information first. At high frequency, the objective function has lots of local minimums. Therefore, the iteration is hard to converge.

Preservation of original solutions. Balanced mass Wasserstein distance does not change the original solution, to be more precise, when no noise is presented in the data, the original solution is the global minimizer for the corresponding optimization problem of the Wasserstein metric. This is observed by our numerical experiments that after sufficient iterations, the solution generated from Wasserstein metrics is the same as the solution generated by L^2 norm, see figure 3.3. If there is noise in the data, however, Wasserstein metrics are more robust against noise.

Robustness of W_2 against noise. What is that we gain: robustness against noise. The most important effect of W_2 is damping or delaying of high-frequency components in the reconstruction. So that we can control the effect of high-frequency modes as long as our numerical tests stop at an appropriate iteration. More than often, high-frequency means noises. Thus robustness against noise.

Loss of resolution with W_2. What is that we gain: robustness against noise. The most important effect of W_2 is damping or delaying of high-frequency components in the reconstruction. So that we can control the impact of high-frequency modes as long as our numerical tests stop at an appropriate iteration. More than often, high-frequency means noises. Thus robustness against noise.

The impact of mass imbalance. Balanced or imbalanced? On the one hand, unbalanced optimal transport should be used since we do not have the balance of mass until the last steps of the iterations. However, imbalanced optimal transport has a "regularization" parameter in the formulation that

requires training in practice. On the other hand, in the end, we need mass balance since otherwise the solution we obtained is not the true solution. Therefore, we should take the parameter β(or $\frac{1}{\alpha}$) to 0.

Optimization framework. We recommend combining Wasserstein formula and the forward problem into a single optimization framework: utilize the fluid dynamic formulation of the optimal transport, and treat the forward operator as the constraint. Let the optimization packages worry about the convergence and mass unbalance issue. Details of the discretization of different schemes are in the appendix.

Computational cost. Computationally, all the W_2 metrics increase the problem dimension by adding a new temporal dimension, which is more computationally expensive than the L^2, and $\mathcal{H}^{-1}$ approaches. The cost among different Wasserstein metrics looks similar. Considering that $\mathcal{H}^{-1}$ has similar reconstructing properties as W_2 type of metrics, it is recommended to apply $\mathcal{H}^{-1}$ approach.

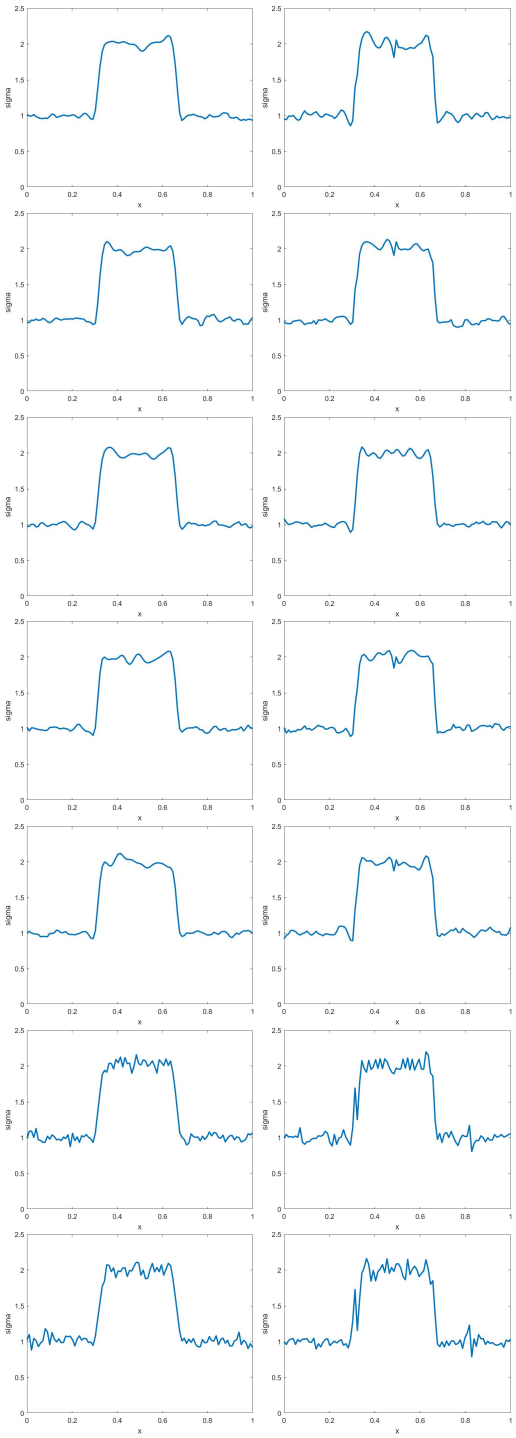


Figure 3.11: Reconstruction of the discontinuous coefficient θ given in (3.63) for Helmholtz model (3.59). The synthetic data contains 5% random noise. First column: reconstruction process with constant 1 initial guess; Second column: reconstruction with piecewise constant initial guess 3.69. From top to bottom: $W_2, W_{2,\text{WFR}}, W_{2,\text{UOT}}(\frac{1}{\alpha} = 0.1), W_{2,\text{GUOT}}(\frac{1}{\alpha} = 0.1), W_{2,\text{Mixed}}(\beta = 0.1), W_1, UW_1(\beta = 0.1)$. All results of quadratic Wasserstein metrics are plotted with loss value 10^{-6}; W_1, UW_1 results are plotted with loss value 10^{-3}

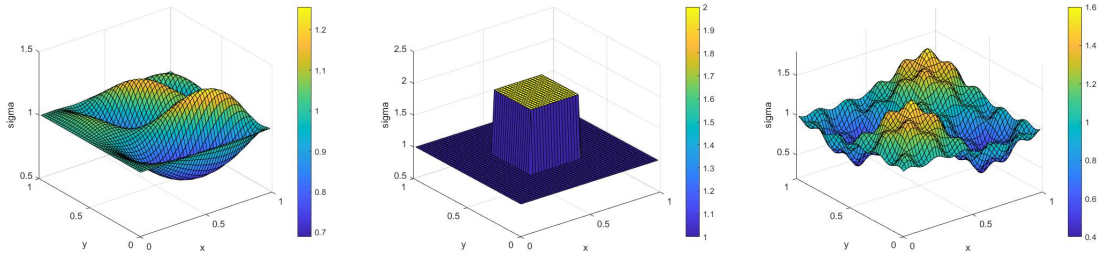


Figure 3.12: Exact shape of two dimensional unknowns. From left to right: continuous shape, discontinuous shape, two scale shape.

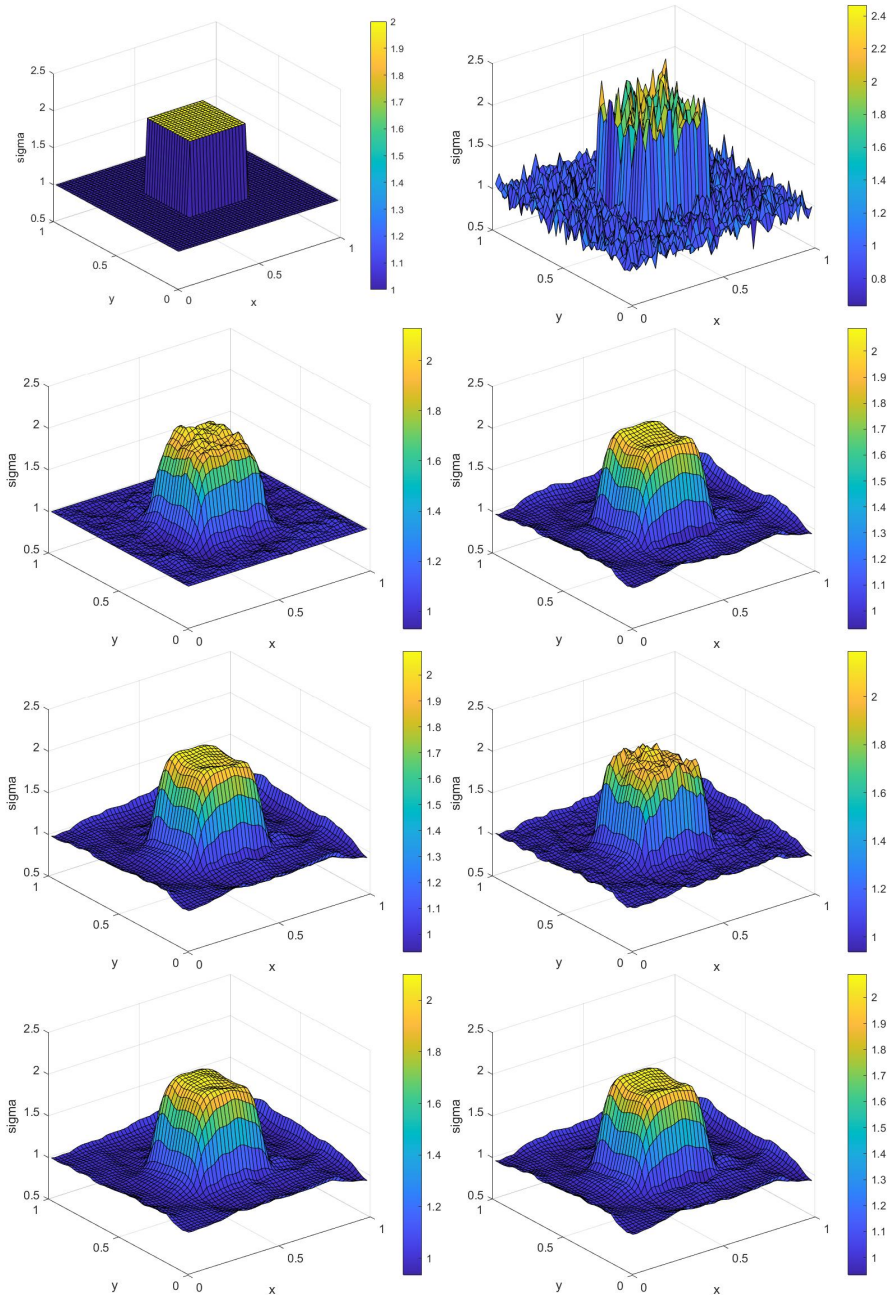


Figure 3.13: Reconstructing the discontinuous shape absorption coefficient σ in 3.70 for the 2D diffusion equation. First row from left to right : the exact σ, L^2; Second row from left to right: $\mathcal{H}^{-1}$, W_2; Third row from left to right: $W_{2,\text{WFR}}$, $W_{2,\text{UOT}}(\frac{1}{\alpha} = 0.1)$; fourth row: $W_{2,\text{GUOT}}(\frac{1}{\alpha} = 0.1)$, $W_{2,\text{Mixed}}(\beta = 0.1)$. The synthetic data contains 10% of random noise, loss value is $5 * 10^{-6}$ for all results.

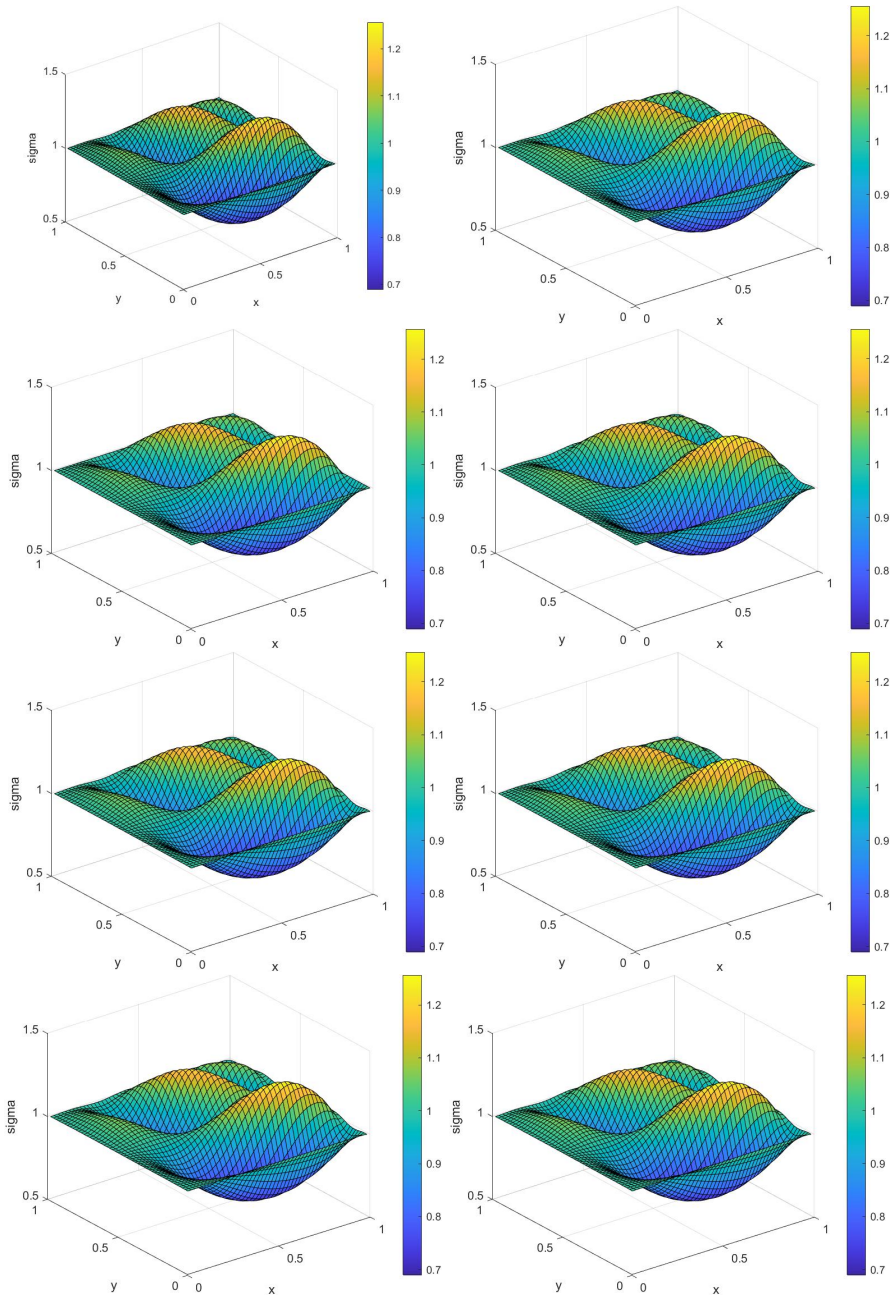


Figure 3.14: Reconstructing the continuous shape absorption coefficient σ in 3.70 for the 2D diffusion equation. First row from left to right : the exact σ, L^2; Second row from left to right: $\mathcal{H}^{-1}$, W_2; Third row from left to right: $W_{2,\text{WFR}}$, $W_{2,\text{UOT}}(\frac{1}{\alpha} = 0.1)$; fourth row: $W_{2,\text{GUOT}}(\frac{1}{\alpha} = 0.1)$, $W_{2,\text{Mixed}}(\beta = 0.1)$. The synthetic data contains no of random noise, loss value is 10^{-9} for all results.

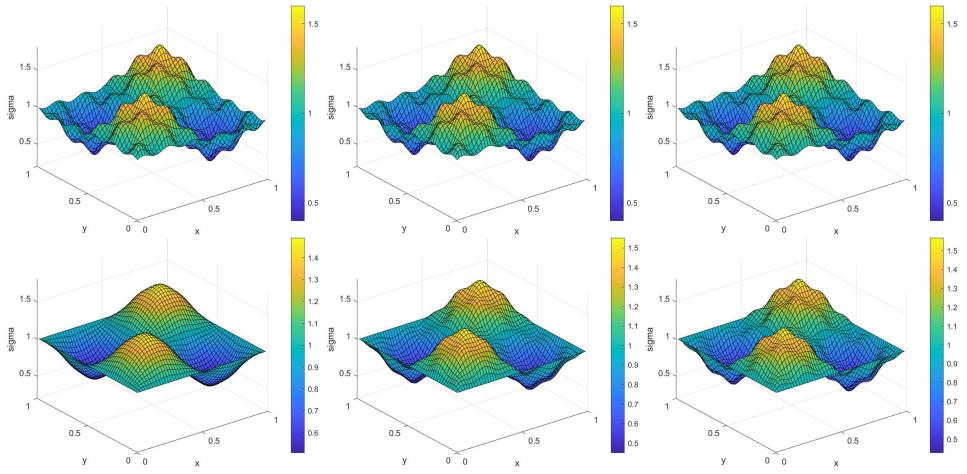


Figure 3.15: Reconstruction of the two scale coefficient σ given in (3.72) the 2D diffusion equation. The synthetic data contains no random noise. From left to right: loss value is $5 * 10^{-6}, 5 * 10^{-7}, 10^{-8}$ respectively. From top to bottom: $L^2, \mathcal{H}^{-1}$.

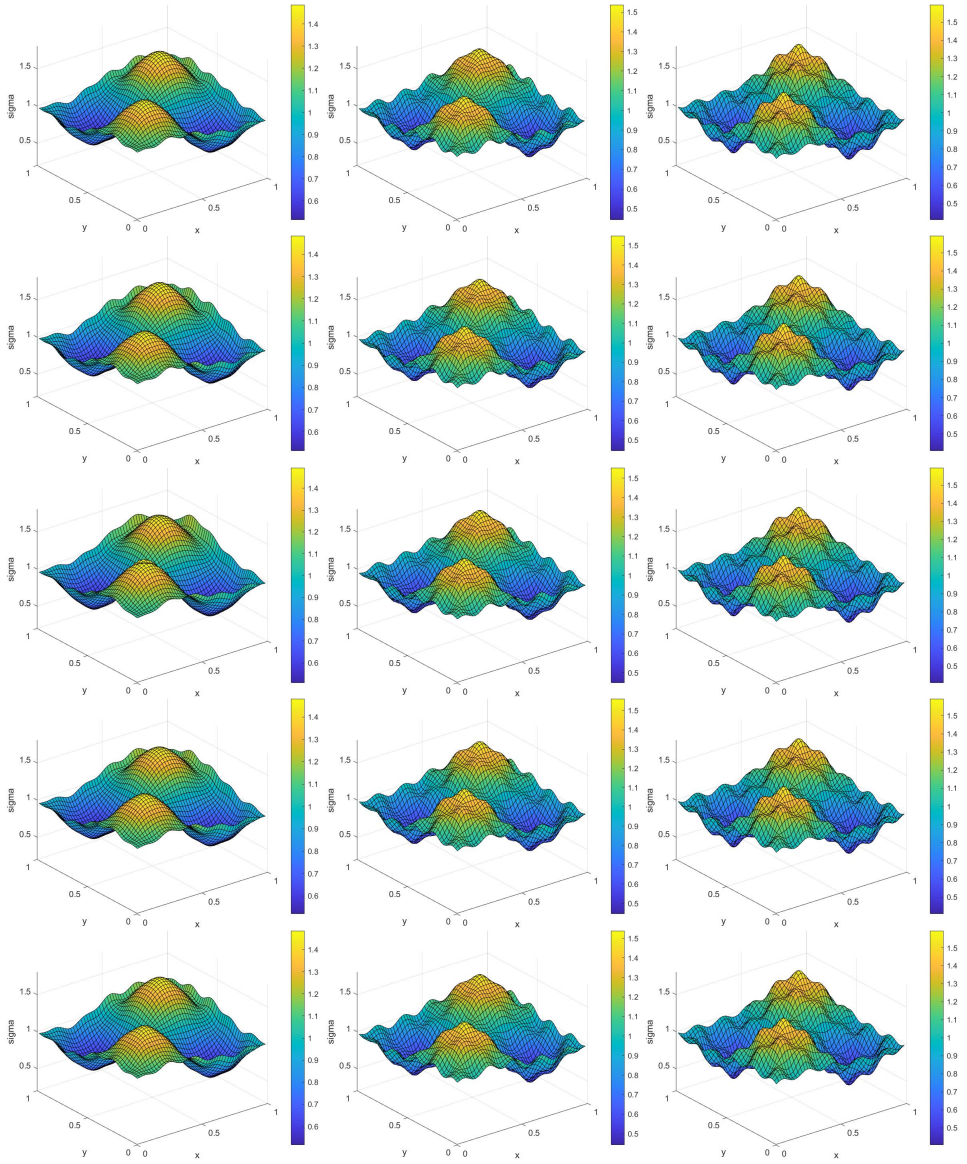


Figure 3.16: Reconstruction of the two scale coefficient σ given in (3.72) the 2D diffusion equation. The synthetic data contains no random noise. From left to right: loss value is $5 * 10^{-6}, 5 * 10^{-7}, 10^{-8}$ respectively. From top to bottom: $W_2, W_{2,\text{WFR}}, W_{2,\text{UOT}}(\frac{1}{\alpha} = 0.1), W_{2,\text{GUOT}}(\frac{1}{\alpha} = 0.1), W_{2,\text{Mixed}}(\beta = 0.1)$.

Chapter 4: Neural Network Induced Loss Function

Non-linear inverse problems are usually solved after discretization by optimization or variants thereof. While the gradient-based method converges well with an initial guess close to the true solution or the optimization loss landscape is convex, its convergence behavior can be erratic and the method can fail if the initial guess is far from the solution. In this chapter, we propose an approach to design loss functions induced by a well-trained neural network, applied directly to the non-linear operators and then obtained near convex landscapes, which are often simpler to solve.

The universal approximation theorem guarantees that neural networks can approximate any continuous function to arbitrary accuracy with no constraint on the width and depth of the hidden layers[31][63]. This offers a niche way of constructing landscapes in the scope of inverse problems. This chapter is based on [35]

4.1 Convexify loss landscape by neural network

To be consistent with the setups in this chapter, we change the notation of input as m instead of θ;

Suppose $m \in L^2(\mathbf{\Omega})$, and $g \in Y$, where $\mathbf{\Omega}$ is a smooth domain, and Y is some Banach space. We apply deep learning methods to train a neural network, denoted by $\widehat{\mathbf{f}}_\alpha^{-1}(\cdot)$, with α denoting the set of parameters of the neural networks, to approximate the inverse operator $\mathbf{f}^{-1}$. A training process based on the L^2 loss function can be formulated as:

$$\widehat{\alpha} = \arg\min_{\alpha\in\Theta} \mathfrak{L}(\alpha) \;\text{ with }\; \mathfrak{L}(\alpha) := \frac{1}{2N}\sum_{j=1}^{N} \|m_j - \widehat{\mathbf{f}}_\alpha^{-1}(\mathbf{g}_j)\|^2_{L^2(\Omega)}$$

where Θ represents the space of parameters of the network. The training dataset is generated synthetically: for each data point $(\mathbf{g}_j, m_j)$, $\mathbf{g}_j$ is generated by evaluating the forward operator

$\mathbf{f}(m_j)$.

The neural network induced loss function is

$$loss(a,b) = \|\widehat{\mathbf{f}}_{\widehat{\alpha}}^{-1}(a) - \widehat{\mathbf{f}}_{\widehat{\alpha}}^{-1}(b)\|_{L^2(\Omega)}^2, a, b \in \mathbf{\Omega} \tag{4.1}$$

Under the neural network induced loss function, we seek to solve for following pre-conditioned nonlinear inverse problem

$$\widehat{\mathbf{f}}_{\widehat{\alpha}}^{-1}\big(\mathbf{f}(m)\big) = \widehat{\mathbf{f}}_{\widehat{\alpha}}^{-1}(\mathbf{g}^{\delta}) \tag{4.2}$$

And the corresponding landscape function for neural network induced loss function is of the following form

$$ls_{NN}(m) = \|\widehat{\mathbf{f}}_{\widehat{\alpha}}^{-1}(\mathbf{f}(m)) - \widehat{\mathbf{f}}_{\widehat{\alpha}}^{-1}(\mathbf{g})\|_{L^2(\Omega)}^2 \tag{4.3}$$

When the neural network can perfectly approximate $\mathbf{f}^{-1}$, i.e. $\widehat{\mathbf{f}}_{\widehat{\alpha}}^{-1} = \mathbf{f}^{-1}$, we have

$$\begin{aligned} ls_{NN}(m) &= \|\mathbf{f}^{-1}(\mathbf{f}(m)) - \mathbf{f}^{-1}(\mathbf{g})\|_{L^2(\Omega)}^2 \\ &= \|m - \mathbf{f}^{-1}(\mathbf{g})\|_{L^2(\Omega)}^2 \end{aligned} \tag{4.4}$$

The loss landscape $ls_{NN}(m)$ is convex, in fact, a quadratic functional of m. When the learning is not perfect but sufficiently accurate, $ls_{NN}(m)$ still has an advantageous landscape. This is given in the following result.

Lemma 4.1.1. *Let* $\widehat{\mathbf{f}}_{\widehat{\alpha}}^{-1} : \mathbf{g} \in Y \mapsto m \in L^2(\Omega)$ *be an approximation to* $\mathbf{f}^{-1}$ *with Fréchet derivative at* $\mathbf{g}$ *given as* $d\widehat{\mathbf{f}}_{\widehat{\alpha}}^{-1}[\mathbf{g}]$. *Assume that*

$$\sup_{m} \|\widehat{\mathbf{f}}_{\widehat{\alpha}}^{-1}\big(\mathbf{f}(m)\big) - m\|_{L^2(\Omega)} \leq \epsilon \tag{4.5}$$

and

$$A := 1 + \sup_{\mathbf{g}} \|d\widehat{\mathbf{f}}_{\widehat{\alpha}}^{-1}[\mathbf{g}]\|_{\mathcal{L}(Y;L^2(\Omega))} < +\infty \tag{4.6}$$

for some $\epsilon > 0$ and $\mathbf{g}^{\boldsymbol{\delta}} = \mathbf{f}(m_0) + \boldsymbol{\delta}$ for some $\boldsymbol{\delta}$ with $\|\boldsymbol{\delta}\|_Y$ sufficiently small. Then we have that

$$\left| \|\widehat{\mathbf{f}}_{\widehat{\alpha}}^{-1}\big(\mathbf{f}(m)\big) - \widehat{\mathbf{f}}_{\widehat{\alpha}}^{-1}(\mathbf{g}^{\boldsymbol{\delta}})\|_{L^2(\Omega)} - \|m - m_0\|_{L^2(\Omega)} \right| \leq 2\epsilon + A\|\boldsymbol{\delta}\|_Y \,. \tag{4.7}$$

Proof. We denote by $r(m) = \widehat{\mathbf{f}}_{\widehat{\alpha}}^{-1}\big(\mathbf{f}(m)\big) - m$. We then have, by Taylor's theorem, that

$$\widehat{\mathbf{f}}_{\widehat{\alpha}}^{-1}(\mathbf{g}^{\boldsymbol{\delta}}) = \widehat{\mathbf{f}}_{\widehat{\alpha}}^{-1}(\mathbf{f}(m_0) + \boldsymbol{\delta}) = m_0 + r(m_0) + d\widehat{\mathbf{f}}_{\widehat{\alpha}}^{-1}[\mathbf{f}(m_0)]\big(\boldsymbol{\delta}\big) + o(\boldsymbol{\delta}),$$

where $\lim\limits_{\boldsymbol{\delta}\to 0} \frac{\|o(\boldsymbol{\delta})\|_{L^2(\Omega)}}{\|\boldsymbol{\delta}\|_Y} = 0$. We therefore have

$$\widehat{\mathbf{f}}_{\widehat{\alpha}}^{-1}\big(\mathbf{f}(m)\big) - \widehat{\mathbf{f}}_{\widehat{\alpha}}^{-1}(\mathbf{g}^{\boldsymbol{\delta}}) = m - m_0 + r(m) - r(m_0) - d\widehat{\mathbf{f}}_{\widehat{\alpha}}^{-1}[\mathbf{f}(m_0)]\big(\boldsymbol{\delta}\big) + o(\boldsymbol{\delta}) \,.$$

We can now use the triangle inequality to conclude that

$$\begin{aligned}
&|\|\widehat{\mathbf{f}}_{\widehat{\alpha}}^{-1}\big(\mathbf{f}(m)\big) - \widehat{\mathbf{f}}_{\widehat{\alpha}}^{-1}(\mathbf{g}^{\boldsymbol{\delta}})\|_{L^2(\Omega)} - \|m - m_0\|_{L^2(\Omega)}| \\
&\leq \|r(m) - r(m_0) - d\widehat{\mathbf{f}}_{\widehat{\alpha}}^{-1}[\mathbf{f}(m_0)]\big(\boldsymbol{\delta}\big) + o(\boldsymbol{\delta})\|_{L^2(\Omega)} \\
&\leq \|r(m)\|_{L^2(\Omega)} + \|r(m_0)\|_{L^2(\Omega)} + \|d\widehat{\mathbf{f}}_{\widehat{\alpha}}^{-1}[\mathbf{f}(m_0)]\big(\boldsymbol{\delta}\big)\|_{L^2(\Omega)} + \|o(\boldsymbol{\delta})\|_{L^2(\Omega)} \\
&\leq 2\epsilon + A\|\boldsymbol{\delta}\|_Y
\end{aligned} \tag{4.8}$$

where the last step comes from the assumptions in (4.5) and (4.6). The proof is complete. □

This result says that if the noise level δ in data $\mathbf{g}^{\delta}$ is sufficiently small and the Fréchet derivative of $\mathbf{f}_{\widehat{\alpha}}^{-1}$ is bounded, then the loss landscape $ls_{NN}(m)$ under neural network induced loss function behaves similarly to the quadratic functional $\|m - \widehat{\mathbf{f}}_{\widehat{\alpha}}^{-1}(\mathbf{g})\|_{L^2(\Omega)}^2$ provided that the training process for $\widehat{\mathbf{f}}_{\widehat{\alpha}}^{-1}$ is accurate enough. Therefore, the loss landscape under the neural network is approximately quadratic functional with error only depending on the neural network itself and the noise level in the dataset. It is clear that we can replace the strong assumption on the accuracy of $\widehat{\mathbf{f}}_{\widehat{\alpha}}^{-1}$, $\sup\limits_{m} \|\widehat{\mathbf{f}}_{\widehat{\alpha}}^{-1}\big(\mathbf{f}(m)\big) - m\|_{L^2(\Omega)} \leq \epsilon$, with the weaker assumption $\|\widehat{\mathbf{f}}_{\widehat{\alpha}}^{-1}\big(\mathbf{f}(m)\big) - m\|_{L^2(\Omega)} \leq \epsilon\|m\|_{L^2(\Omega)}$, in which case the 2ϵ term in the bound (4.7) will be replaced by $\epsilon(\|m\|_{L^2(\Omega)} + \|m_0\|_{L^2(\Omega)})$. The conclusion still holds.

Due to the smoothing property of the forward operator, the stability of the trained inverse operator, measured by the boundedness of its Fréchet derivative, is enough to ensure the accuracy of the neural network reconstruction. Therefore, if we could train network approximations with such stability properties, they have good generalization capabilities in the output space.

Lemma 4.1.2. *Let $m, m_0 \in C^2(\Omega) \cap [\underline{m}, \overline{m}]$ for some $0 < \underline{m} < \overline{m} < +\infty$. Then, when $\|m - m_0\|_{L^2(\Omega)}$ is sufficiently small, there exists a constant $\mathfrak{c}$ such that*

$$\|\widehat{\mathbf{f}}_{\widehat{\alpha}}^{-1}\big(\mathbf{f}(m)\big) - \widehat{\mathbf{f}}_{\widehat{\alpha}}^{-1}\big(\mathbf{f}(m_0)\big)\|_{L^2(\Omega)} \leq \mathfrak{c}\|m - m_0\|_{L^2(\Omega)} \tag{4.9}$$

Proof. The map $m \mapsto \mathbf{g} := \mathbf{f}(m)$ is Fréchet differentiable with the derivative at m in direction $\widetilde{m}$ denoted as $d\mathbf{f}[m](\widetilde{m})$. By Taylor's theorem, we have

$$\begin{aligned}\widehat{\mathbf{f}}_{\widehat{\alpha}}^{-1}\big(\mathbf{f}(m)\big) &= \widehat{\mathbf{f}}_{\widehat{\alpha}}^{-1}\big(\mathbf{f}(m_0) + d\mathbf{f}[m_0](m - m_0) + o(m - m_0)\big) \\ &= \widehat{\mathbf{f}}_{\widehat{\alpha}}^{-1}\big(\mathbf{f}(m_0)\big) + d\widehat{\mathbf{f}}_{\widehat{\alpha}}^{-1}[\mathbf{f}(m_0)]\big(d\mathbf{f}[m_0](m - m_0)\big) + \widetilde{o}(m - m_0),\end{aligned}$$

where $\lim_{m \to m_0} \frac{\|o(m-m_0)\|_Y}{\|m-m_0\|_{L^2(\Omega)}} = \lim_{m \to m_0} \frac{\|\widetilde{o}(m-m_0)\|_{L^2(\Omega)}}{\|m-m_0\|_{L^2(\Omega)}} = 0$. We therefore have

$$\widehat{\mathbf{f}}_{\widehat{\alpha}}^{-1}\big(\mathbf{f}(m)\big) - \widehat{\mathbf{f}}_{\widehat{\alpha}}^{-1}(\mathbf{f}(m_0)) = d\widehat{\mathbf{f}}_{\widehat{\alpha}}^{-1}[f(m_0)]\big(d\mathbf{f}[m_0](m - m_0)\big) + \widetilde{o}(m - m_0)\,.$$

The bound in (4.9) then follows from the assumption (4.6). □

When the class of input variable is sufficiently nice, for instance, when each m can be represented with a small number of Fourier coefficients in a narrow frequency band, one can hope that accurate training is achievable. When this is the case, Lemma 4.1.1 and Lemma 4.1.2 ensure that the learned model can be utilized to facilitate the new loss landscape.

4.2 Numerical methods

We list two numerical methods for solving the inverse problems with neural network induced loss functions.

4.2.1 Gradient descent iteration

Modern deep learning softwares such as pytorch and tensorflow have developed efficient auto differentiation algorithms [11]. With the help of those algorithms, we can compute gradient of inverse operator, i.e. $d\widehat{\mathbf{f}}_{\widehat{\alpha}}^{-1}[\mathbf{g}]$ both efficiently and accurately. Due to the chain rule, we have that

$$\frac{\partial ls_{NN}(m)}{\partial m} = (\widehat{\mathbf{f}}_{\widehat{\alpha}}^{-1}(\mathbf{f}(m)) - \widehat{\mathbf{f}}_{\widehat{\alpha}}^{-1}(\mathbf{g}))d\widehat{\mathbf{f}}_{\widehat{\alpha}}^{-1}[\mathbf{f}(m)]d\mathbf{f}[m] \tag{4.10}$$

where $d\mathbf{f}[m]$ represents the gradient of forward operator $\mathbf{f}$ at direction of m. We can solve the inverse problem by minimizing the landscape function with gradient descent or gradient-based algorithms such as ADAM [72]

4.2.2 Neumann series iteration

When an accurate training of $\mathbf{f}^{-1}$ is not available, we can train a good approximation to the inverse, that is, when the operator $\mathcal{I} - \widehat{\mathbf{f}}_{\widehat{\alpha}}^{-1} \circ \mathbf{f}$ is not zero but small in an appropriate operator norm, the inverse problem (4.2) can be solved by using Neumann series. More precisely, we can rewrite (4.2) as

$$m - K(m) = \widehat{\mathbf{f}}_{\widehat{\alpha}}^{-1}(\mathbf{g}^{\delta}), \quad K := \mathcal{I} - \widehat{\mathbf{f}}_{\widehat{\alpha}}^{-1} \circ \mathbf{f}$$

whose solution can be expressed in a Neumann series as

$$\widehat{m} = (\mathcal{I} - K)^{-1}(\widehat{\mathbf{f}}_{\widehat{\alpha}}^{-1}(\mathbf{g}^{\delta})) = \sum_{j=0}^{\infty} K^{j}\left(\widehat{\mathbf{f}}_{\widehat{\alpha}}^{-1}(\mathbf{g}^{\delta})\right). \tag{4.11}$$

The better the approximation $\widehat{\mathbf{f}}_{\widehat{\alpha}}^{-1}$ is to $\mathbf{f}^{-1}$, the faster the series converges.

4.3 Case study: full wave inversion

Full waveform inversion (FWI) refers to the process of extracting information on physical parameters of wave equations from data related to the solutions to the wave equations [4, 17, 19, 21, 26, 37, 41, 91, 99, 109, 110, 111, 122, 132, 134, 137]. In seismic imaging, this is manifested as the problem of reconstructing the speed distribution of seismic waves in the interior of the Earth from measured wave field data on the earth surface. The sources of the measured waves could come either from nature, such as earthquakes or from geophysical exploration activities by humankind, such as air guns and seismic vibrators. We refer interested readers to [22, 48, 103, 138] and references therein for overviews on the recent development in the field of FWI for geophysical applications. While the term FWI was mainly coined in the seismic imaging community, FWI also has a wide range of applications in other imaging applications, such as in medical ultrasound imaging [7, 16, 60, 67, 81, 93, 96, 141]. From the practical point of view, the main difference between geophysical and medical FWI is that the quality of the dataset collected in medical applications, both in terms of the variety of source-detector configurations can be arranged and in terms of the frequency contents of the incident sources, is much richer than that of the geophysical FWI dataset.

For the sake of concreteness, let us consider the simplest model of acoustic wave propagation in a heterogeneous medium Ω with wave speed field $m(\mathbf{x}) > 0$. The wave field u solves

$$\begin{aligned} \frac{1}{m^2(\mathbf{x})}\frac{\partial^2 u}{\partial^2 t} - \Delta u &= 0, && \text{in } (0,+\infty)\times\Omega \\ \frac{\partial u}{\partial \mathbf{n}} &= h(t,\mathbf{x}), && \text{on } (0,+\infty)\times\partial\Omega \end{aligned} \tag{4.12}$$

with an appropriate initial condition. Here, $\mathbf{n}$ is the unit outward normal vector of the domain boundary at $\mathbf{x} \in \partial\Omega$. The data that we measure is time traces of the solution to the wave equation (4.12) at a set of detector locations, say $\Gamma \subset \mathbb{R}^d$, for a period of time, say T, that is,

$$g := u(t,\mathbf{x})|_{(0,T]\times\Gamma}\,. \tag{4.13}$$

The objective of FWI in this setting is to recover the unknown wave speed field m in the wave equation (4.12) from the measured data $u(t,\mathbf{x})_{|(0,T]\times\Gamma}$ collected in a multi-source multi-detector configuration. This is a challenging inverse problem that has rich mathematical and computational content. The main computational strategy, due to the lack of explicit/semi-explicit reconstruction methods, in solving the FWI inverse problem as well as many other model-based inverse problems, is the classical L^2 least-squares formulations where we search for the inverse solution by minimizing the L^2 mismatch between model predictions and observed data. To formulate this more precisely, we assume that we collect data from N_s acoustic sources $\{h_s\}_{s=1}^{N_s}$, and we denote by $f(m;h_s)$ the forward model that takes m to the corresponding wave field data g_s ($1 \le s \le N_s$). Then the inverse problem of reconstructing m from measured data g_s^δ aims at solving the following operator equation:

$$\mathbf{f}(m) = \mathbf{g}^\delta \tag{4.14}$$

where

$$\mathbf{f}(m) := \begin{pmatrix} f(m;h_1) \\ \vdots \\ f(m;h_s) \\ \vdots \\ f(m;h_{N_s}) \end{pmatrix} \quad \text{and} \quad \mathbf{g}^\delta := \begin{pmatrix} g_1^\delta \\ \vdots \\ g_s^\delta \\ \vdots \\ g_{N_s}^\delta \end{pmatrix}.$$

The superscript δ denotes the fact that the datum g is polluted by measurement noise. The classical L^2 least-squares method performs the reconstruction by searching for m that minimizes the mismatch functional (with the possibility of adding a regularization term):

$$\Psi(m) := \frac{1}{2}\|\mathbf{f}(m) - \mathbf{g}^\delta\|^2_{[L^2((0,T]\times\Gamma)]^{N_s}}. \tag{4.15}$$

This challenging numerical optimization problem has been extensively studied in the past three decades. Many novel methods have been developed to address two of the main challenges: (i) the high computational cost needed to reconstruct high-resolution images of m, and (ii) the abundance

of local minimizers (due to the non-convexity of the least-squares functional) that trap iterative reconstruction algorithms; see for instance [22, 48, 120] for a detailed explanation of those challenges among others.

In recent years, there has been great interest in the FWI community to use deep learning techniques based on neural networks to replace the classical least-squares based inversion methods [1, 6, 30, 44, 46, 64, 68, 70, 89, 94, 117, 127, 128, 129, 130, 143, 145, 149, 150, 151]. Assume that we are given a set of sampled data

$$\{\mathbf{g}_j := (g_{j1}, \cdots, g_{js}, \cdots, g_{jN_s})^{\mathfrak{T}}, m_j\}_{j=1}^{N}, \tag{4.16}$$

where $\{m_j\}_{j=1}^{N}$ are a set of N velocity profiles sampled from a given distribution and $\{\mathbf{g}_j\}_{j=1}^{N}$ are the corresponding wave field predictions generated from N_s sources $\{h_s\}_{s=1}^{N_s}$ with the model $\mathbf{g} = \mathbf{f}(m)$. Deep learning methods try to train a neural network, denoted by $\mathbf{f}_\alpha^{-1}(\mathbf{g})$, with α denoting the set of parameters (that is, the weight matrices and the bias vectors) of the neural networks, that represents the inverse operator $\mathbf{f}^{-1}$. A training process based on the L^2 loss functional can be formulated as:

$$\widehat{\alpha} = \arg\min_{\alpha\in\Theta} \mathfrak{L}(\alpha) \quad \text{with} \quad \mathfrak{L}(\alpha) := \frac{1}{2N}\sum_{j=1}^{N} \|m_j - \mathbf{f}_\alpha^{-1}(\mathbf{g}_j)\|_{L^2(\Omega)}^2$$

where Θ represents the network's space of parameters, a regularization term can be added to the loss function $\mathfrak{L}(\alpha)$ to help stabilize the training process. The number of samples N needs to be large enough in order for $\mathfrak{L}(\alpha)$ to be a good approximation to the expectation of the mismatch over the distribution: $\mathbb{E}_m[\|m - \mathbf{f}_\alpha^{-1}(\mathbf{g}(m))\|_{L^2(\Omega)}^2]$. Many other loss functions can be used, but we will not dive into this direction. Note that since we know, the forward operator $\mathbf{f}$ and are only interested in learning its inverse operator, the datasets used in the training process are synthetic: for each data point $(\mathbf{g}_j, m_j)$, $\mathbf{g}_j$ is constructed by solving the wave equation (4.12) with the given speed field m_j and evaluate (4.13).

Numerical experiments, such as those documented in [6, 70, 89, 127, 143, 145, 149, 150, 151], showed that, with sufficiently large training datasets, it is possible to train highly accurate inverse

operators that can be used to directly map measured wave field data into the velocity field. This, together with the recent success in learning inverse operators for other inverse problems (see for instance [2, 10, 20, 47, 113, 124] for some examples) has led many to believe, probably overly optimistically, that one can completely replace classical computational inversion with offline deep learning.

Despite the tremendous success in deep learning for FWI, it is still computational challenging to train a once-for-all inverse machine $\mathbf{f}_\alpha^{-1}$. First, with the aim of reconstructing high-resolution images of the velocity field $m(\mathbf{x})$, the size of the neural networks to be constructed as a discrete representation of $\mathbf{f}_\alpha^{-1}$ is prohibitively large. Second, it is well known that $\mathbf{f} : m \mapsto \mathbf{g}$ is a smoothing operator (between appropriate spaces; see for instance [66] and references therein for more precise mathematical characterization of the statement). The inverse operator is therefore de-smoothing. Learning such operators requires the ability to capture precisely high-frequency information in the training data, and this is very hard to do in the training process as deep neural networks tend to capture low-frequency components of the data much more efficiently than the high-frequency components [112, 118, 144]. On top of the above, the inverse operator $\mathbf{f}_\alpha^{-1}$ we learned from model-generated data very often has limited generalization, making it challenging to apply the operator to new measured datasets.

In this chapter, we propose an offline-online computational strategy for coupling classical least-squares based computational inversion with deep learning based approaches for FWI to achieve advantages that can not be achieved with only one of the components. Roughly speaking, we utilize offline trained approximate inverse of the operator $\mathbf{f}$ to pre-condition online least-squares based numerical reconstructions. Instead of pursuing high-quality training of highly accurate inverse operators, we train neural networks that only capture the main features in the velocity field. This relaxes dramatically the requirement on both the size of the dataset and the computational resources needed in the training process, and the trained model is more generalizable to other classes of velocity models. Meanwhile, the offline trained approximate inverse is sufficient as a nonlinear pre-conditioner to improve the speed of convergence of the classical least-squares based FWI

numerical reconstruction in the online stage of the inversion.

The rest of this chapter is organized as follows. We first describe the proposed coupling strategy in Section 4.4 in the abstract setting. We then present some preliminary understanding of the training and reconstruction stage of the method in Section 4.5. In Section 4.6 we discuss the details of the implementation of the strategy. Extensive numerical simulations are presented in Section 4.7 to demonstrate the performance of the learning-inversion coupling.

4.4 Coupling learning with FWI

Our main objective here is to couple the deep learning based image reconstruction approach with the classical least-squares based image reconstruction method for FWI. More precisely, we utilize the approximate inverse we learned with neural networks to construct a new loss function for least-squares based FWI reconstruction from measured data.

4.4.1 Robust offline learning of main features

In the offline learning stage, we use deep learning to train an approximate inverse of the operator $\mathbf{f}$. As we outlined in the previous section, our main argument is that the learning process can only be performed reliably on a small number of dominant features of the velocity field. First, resolving all details of the velocity field requires over-sized neural networks that demand an exceedingly large amount of training data, not to mention that such networks are computationally formidable to train reliably. Second, large neural networks or large sizes display serious frequency bias in picking up frequency contents in the training datasets [144], making it inefficient to fit high-frequency components of the velocity field. Despite all the challenges in resolving high-frequency features, it has been shown in different scenarios that learning low-frequency components of the velocity profile can be done in a robust manner [88, 115, 129]. This means that if we take the Fourier representation, the lower Fourier modes of the inverse operator can be learned stably. This good low-frequency approximate inverse is our main interest in the learning stage (even though an accurate inverse itself would be better if one can realistically have it).

Let $\mathfrak{M}$ be the feature map we selected, and $\mathfrak{m}$ the corresponding feature vector, that is,

$$\mathfrak{M} : m(\mathbf{x}) \in \mathcal{M} \longmapsto \mathfrak{m} \in \mathbb{M},$$

where $\mathcal{M} \subseteq L^2(\Omega)$ is the class of velocity field that we are interested in and $\mathbb{M}$ is the space of the feature vectors. Motivated by the analysis of weighted optimization in [38, 39], we train a network, which we still denote as $\mathbf{f}_\alpha^{-1} : \mathbf{g} \mapsto \mathfrak{m}$, using the synthetic dataset (4.16), through the optimization problem

$$\widehat{\alpha} = \arg\min_{\alpha\in\Theta} \mathcal{L}(\alpha) \;\text{ with }\; \mathcal{L}(\alpha) := \frac{1}{2N}\sum_{j=1}^{N} \Big\| \boldsymbol{\mu} \circledast \Big(\mathfrak{m}_j - \mathbf{f}_\alpha^{-1}(\mathbf{g}_j)\Big)\Big\|_{\mathbb{M}}^2 \tag{4.17}$$

where the weight vector $\boldsymbol{\mu}$ is selected to weigh the loss heavily on the features we are interested in while damping the features that are hard to learn stably. The $\circledast$ is used to denote the componentwise product between the vectors involved. The selection of the feature vectors as well as the weighting vector $\boldsymbol{\mu}$ will be discussed in Section 4.7 in more detail. For the purpose of illustrating the main idea, let us point out that one example is to think of (4.17) as the equivalence of

$$\widehat{\alpha} = \arg\min_{\alpha\in\Theta} \frac{1}{2N}\sum_{j=1}^{N} \int_\Omega \Big(\int_\Omega \mu(\mathbf{x}-\mathbf{y})\big(m_j(\mathbf{y}) - \mathbf{f}_\alpha^{-1}(\mathbf{g}_j)(\mathbf{y})\big)d\mathbf{y}\Big)^2 d\mathbf{x}$$

in the Fourier domain, i.e. when the features we use are Fourier modes, with $\boldsymbol{\mu}$ being the Fourier transform of the kernel $\mu(\mathbf{x})$. If we take μ to be a smoothing kernel, such as a Gaussian kernel, $\boldsymbol{\mu}$ will decay fast with the increase of the frequency. In such a case, the learning problem (4.17) focuses on the lower Fourier modes of the velocity field m.

Weighted optimization schemes of the form (4.17) with weight $\boldsymbol{\mu}$ to emphasize dominant features in the learning problems have been extensively studied in the learning and inverse problems community; see [39] and references therein. When the feature we selected are Fourier basis, it has been shown that the correct selection of the weight $\boldsymbol{\mu}$ in the training scheme can lead to more robust learning results for a class of models $\mathbf{f}_\alpha^{-1}$ following certain distributions, sometimes at the expense of learning accuracy, with better generalization capabilities [39]. This is the main motivation for us

to adopt this strategy for our purpose in this research.

4.4.2 New loss function for online inversion

In the online reconstructions stage, we utilize the approximate inverse we trained to construct a new loss function for FWI image reconstruction from given noisy data $\mathbf{g}^\delta$. More precisely, instead of solving the problem (4.14), we aim to solve the modified model

$$\widehat{\mathbf{f}}_{\widehat{\alpha}}^{-1}\big(\mathbf{f}(m)\big) = \widehat{\mathbf{f}}_{\widehat{\alpha}}^{-1}(\mathbf{g}^\delta) \tag{4.18}$$

where

$$\widehat{\mathbf{f}}_{\widehat{\alpha}}^{-1} := \mathfrak{M}^{-1} \circ \mathbf{f}_{\widehat{\alpha}}^{-1} : \ \mathbf{g} \mapsto m$$

is the learned approximate to $\mathbf{f}^{-1}$ (while $\mathbf{f}_{\widehat{\alpha}}^{-1} : \ \mathbf{g} \mapsto \mathfrak{m}$ is the learned representation in $\mathfrak{M}$).

The least-squares formulation for the reconstruction problem now takes the form

$$\widehat{m} = \underset{m \in \mathcal{M}}{\arg\min}\, \Phi(m)\,, \tag{4.19}$$

with

$$\Phi(m) := \frac{1}{2}\|\widehat{\mathbf{f}}_{\widehat{\alpha}}^{-1}\Big(\mathbf{f}(m)\Big) - \widehat{\mathbf{f}}_{\widehat{\alpha}}^{-1}(\mathbf{g}^\delta)\|_{L^2(\Omega)}^2 + \frac{\gamma}{2}\|\mathfrak{M}^{-1}(\boldsymbol{\mu}^{-1} \circledast \mathfrak{m})\|_{L^2(\Omega)}^2\,. \tag{4.20}$$

The last term in the objective functional is a Tikhonov regularization functional that imposes a smoothness constraint on the target velocity field. This smoothness constraint is selected such that it is consistent with the training process. The natural initial guess for any iterative solution scheme for this minimization problem is $m_0 := \widehat{\mathbf{f}}_{\widehat{\alpha}}^{-1}(\mathbf{g}^\delta)$.

Let us emphasize that there is a significant difference between the L^2 loss function $\Phi(m)$ we introduced in (4.20), ignoring the regularization term, and the standard L^2 loss function $\Psi(m)$ defined in (4.15). Our loss function $\Phi(m)$ measures the mismatch between the approximations of the predicted velocity field and the true velocity field corresponding to the measured data, while the standard loss function $\Psi(m)$ measures the mismatch between predicted wave field data with

measured wave field data. In other words, our loss function works on the parameter space (also called the model space in the FWI literature, that is, the space of the velocity field) while the standard loss function is defined on the signal space (that is the space of wave field signals at the detectors). With reasonably-trained $\widehat{\mathbf{f}}_{\widehat{\alpha}}^{-1}$, the functional $\Phi(m)$ has advantageous landscape for optimization purpose as we will demonstrate in the numerical simulations in Section 4.7.

4.4.3 The benefits of the coupling approach

The offline-online coupling scheme we proposed allowed us to focus on training a robust approximate inverse instead of the exact inverse. This makes the learning process more stable and also requires less computational resources (in terms of the amount of data, the size of the network, and the computational cost for optimization) than training an accurate inverse. Moreover, the sacrifice in accuracy brings better generalizability for the learned approximate inverse. On the computational side, the trained approximate inverse serves as a "preconditioner" for the inversion process. It can not only provide a good initial guess for the reconstruction but also simplify the landscape of the optimization problem.

We finish this section with the following remark. In the ideal case when all the operators involved are invertible as they should be, the solution to (4.18) is identical to the solution to (4.14), assuming that $\mathbf{g}$ indeed lives in the range of $\mathbf{f}$. Therefore, our formulation does not change the true solution to the original inverse problem. However, as we will see, the new formulation utilizes the result of learning to facilitate the FWI reconstruction in terms of saving computational cost as well as making the optimization landscape more desirable.

4.5 Formal understanding of the coupling

We now attempt to gain a more systematic understanding of the coupling strategy. As we have argued in the previous sections, it is computationally challenging to train neural networks that are accurate approximations of the inverse operator and are very generalizable at the same time. However, there is certainly some dominant information in the inverse operator that we could extract

with learning and this is the approximate inverse that we are interested in constructing.

4.5.1 Elements of network training

Due to the fact that the training data we have are generated from exactly the same operator we are trying to represent with the neural network, the learning process we have is much more under control than those purely data-driven learning problems in applications. Here we highlight a few critical issues in the learning process without getting into the details of the implementation of the learning algorithm.

Sampling training data. To learn the inverse operator, we need to pay attention to both its input space and its output space. While our focus will be on learning the low-frequency component of the inverse operator, we want the training data to include as much high-frequency information as possible to gain generalization capability in the input space. Let $\mathbf{K}_{\text{out}}$ be the frequency range for the network output that we are interested in recovering and $\mathbf{K}_{\text{in}}$ the frequency range of the velocity fields that generated the wavefield data. We construct the training dataset as

$$\{m_j(\mathbf{x}), \mathbf{g} := \mathbf{f}(m_j(\mathbf{x}) + \widetilde{m}_j(\mathbf{x}))\}_{j=1}^{N}$$

where $\widetilde{m}_j(\mathbf{x})$ are selected such that $\mathcal{F}(\widetilde{m}_j)(\mathbf{k}) = 0\ \forall \mathbf{k} \in \mathbf{K}_{\text{out}}$, and $\mathcal{F}(\widetilde{m}_j)(\mathbf{k}) \neq 0\ \forall \mathbf{k} \in \mathbf{K}_{\text{in}} \backslash \mathbf{K}_{\text{out}}$ ($\mathcal{F}(m)$ denoting the Fourier transform of m). In other words, we train the network with input wavefield data having richer frequency content of the velocity field than the output velocity field. This construction enriches the frequency content of the input data but does not increase the computational cost of the training process.

The well-known result on the differentiability of the data $\mathbf{g}$ with respect to m, quoted in the proposition below, indicates that the input space of the inverse operator, i.e., the range of the forward operator, is quantitatively smoother than the output space (the velocity space) that we are working with. Therefore, a well-trained network approximation should have good interpolation ability in applications when the space of velocity field we are interested in working with is sufficiently

smooth.

Proposition 4.5.1 ([9, 32, 66]). *Let Ω be a smooth domain and $h(t, \mathbf{x})$ be the restriction of a C^1 function on $\partial\Omega$. Assume further that $m \in C^2(\bar{\Omega}) \cap [\underline{m}, \overline{m}]$ for some $0 < \underline{m} < \overline{m} < +\infty$. Then the map: $\mathbf{f}(m) : m \longmapsto \mathbf{g}$ is Fréchet differentiable at any $m \in C^2(\bar{\Omega}) \cap [\underline{m}, \overline{m}]$.*

The result is standard. We refer interested readers to [9, 32, 66] and references therein for more precise formulations of it in different scenarios. This result also ensures that if we can train a stable network, then the learning quality is guaranteed; see Lemma 4.1.2 below.

Network training error. Our main objective of this work is to focus the learning process on the low-frequency content of the output of the inverse operator. We do this with the weighted optimization scheme (4.17) by selecting weight $\boldsymbol{\mu}$ that heavily penalizes the low-frequency component of the mismatch of true data and the network prediction. The impact of such weighting schemes on the learning results have been analyzed extensively; see [38, 39] and reference therein. We illustrate this in an extremely simplified setting. Let $\mathbf{F} := (\mathbf{f}^{-1})'(m_0)$ be the linearization of $\mathbf{f}^{-1}$ at m_0 for a one-dimensional medium. Assume that the learning loss function $\mathcal{L}(\alpha)$ in (4.17) is minimized to the order of ε^2 in the training process. Then on the leading order, the trained $\mathbf{F}$ satisfies

$$\boldsymbol{\mu} \circledast (\mathbf{m} - \mathbf{FG}) \sim O(\varepsilon),$$

where $\mathbf{m} = [\mathbf{m}_1, \cdots, \mathbf{m}_N]$ is the matrix whose columns are vectors of the Fourier coefficients of the training velocity samples $\{m_j\}_{j=1}^N$, $\mathbf{G} = [\mathbf{g}_1, \cdots, \mathbf{g}_N]$ is a matrix whose columns are vectors of the input data, and $O(\varepsilon)$ is a diagonal matrix of size order ε. The trained linearized inverse operator, when applied to a new input data $\mathbf{g}^\delta$, gives the result

$$\mathbf{F}\mathbf{g}^\delta \sim \big(\mathbf{m} - \boldsymbol{\mu}^{-1} \circledast O(\varepsilon)\big)\mathbf{G}^{\mathfrak{T}}(\mathbf{G}\mathbf{G}^{\mathfrak{T}})^{-1}\mathbf{g}^\delta\,.$$

The nature of $\boldsymbol{\mu}$ indicates that the relative error in the learned output is more significant in the high-frequency Fourier modes.

4.5.2 Computational simplifications

The reconstruction stage of the coupling can be greatly simplified when the training of the neural network approximation is sufficiently accurate.

First, the coupling method will degenerate into a deep learning based method when we have confidence in our ability to train an accurate deep neural network representation of the inverse operator in FWI. Indeed, when $\widehat{\mathbf{f}}_{\widehat{\alpha}}^{-1} = \mathbf{f}^{-1}$, that is, $\widehat{\mathbf{f}}_{\widehat{\alpha}}^{-1}$ is exactly the inverse, the reconstruction step (4.19) simplifies to

$$\widehat{m} = \arg\min_{m\in\mathcal{M}} \frac{1}{2}\|m - \widehat{\mathbf{f}}_{\widehat{\alpha}}^{-1}(\mathbf{g}^{\delta})\|_{L^2(\Omega)}^2 + \frac{\gamma}{2}\|\nabla m\|_{L^2(\Omega)}^2 ,$$

assuming, only for the sake of simplifying the notation, that the weighting operator $\mu(\mathbf{x}-\mathbf{y})$ is taken as an integral operator such that $\boldsymbol{\mu}^{-1}(\mathbf{k}) = \mathbf{k}$. This gives a fast inversion for the new data and immediately leads to the optimal selection of the regularization parameter when the regularization term is not too complicated. In this case, we simply did a post-process on the deep learning reconstruction given by the operator $\widehat{\mathbf{f}}_{\widehat{\alpha}}^{-1}$. The solutions to this are explicitly given as

$$\widehat{m} = (\mathcal{I} + \gamma\Delta)^{-1}\widehat{\mathbf{f}}_{\widehat{\alpha}}^{-1}(\mathbf{g}^{\delta}) ,$$

where $\mathcal{I}$ is the identity and Δ is the Laplacian operator. Therefore, m is simply a smoothed version of the result produced by the trained neural network, $\widehat{\mathbf{f}}_{\widehat{\alpha}}^{-1}(\mathbf{g}^{\delta})$. The exact form of the smoothing effect depends on the selection of μ.

Second, training of inverse operator $\mathbf{f}^{-1}$ of FWI may be difficult due to computational or analytical reasons. Fortunately, even if we do not possess an accurate training of $\mathbf{f}^{-1}$, we can still gain something by using Neumann series, as we discussed above 4.2.2. However, Neumann series do require we can train a fairly good approximation to the inverse, that is, when the operator $\mathcal{I} - \widehat{\mathbf{f}}_{\widehat{\alpha}}^{-1} \circ \mathbf{f}$ is not zero but small in an appropriate operator norm. For the training we had, see more discussion in Section 4.7, a few terms of the Neumann series often provide sufficient accuracy for

the reconstruction.

Let us emphasize that by the informal analysis in Section 4.5.1, the error in the learning implies roughly that $|\mathcal{F}\big(m - K(m)\big)(\mathbf{k})| \sim \zeta(\mathbf{k})\|m\|_{L^2(\Omega)}$ with $\zeta(\mathbf{k})$ large for large $|\mathbf{k}|$. Due to the fact that the operator norm of $\mathcal{I} - K$ is bounded below by $\max_{\mathbf{k}} \zeta(\mathbf{k})$, this means that the convergence speed of the Neumann series is controlled by the worst training error in the (high-frequency) Fourier modes.

4.5.3 Utilizing learning outside of training domain

It is essential to point out that the weight $\boldsymbol{\mu}$ in the weighted training scheme (4.17) should be selected to emphasize the low-frequency components of the output and penalize the high-frequency components. It should not altogether remove the high-frequency components. If it does, then the high-frequency components of the velocity field in the reconstruction stage can not be recovered with the optimization problem (4.19). This is an obvious yet important observation that we summarize as a lemma to emphasize it.

Lemma 4.5.2. *Let* $\widehat{\mathbf{f}}_{\widehat{\alpha}}^{-1}$ *be such that for any* m, $\mathcal{F}[\widehat{\mathbf{f}}_{\widehat{\alpha}}^{-1}(\mathbf{f}(m))](\mathbf{k}) = \mathbf{0}\ \forall |\mathbf{k}| > k_0$, *and* $\widehat{m}$ *be reconstructed from* (4.19) *with a gradient-based iterative scheme or the Neumann series method in* (4.11). *Then* $\mathcal{F}[\widehat{m}](\mathbf{k}) = \mathbf{0}\ \forall |\mathbf{k}| > k_0$.

Proof. Under the assumption on $\widehat{\mathbf{f}}_{\widehat{\alpha}}^{-1}$, it is straightforward to check that $\mathcal{F}(m_0)(\mathbf{k}) = 0$ ($m_0 := \widehat{\mathbf{f}}_{\widehat{\alpha}}^{-1}(\mathbf{g}^\delta)$) $\forall |\mathbf{k}| > k_0$, and $\mathcal{F}(K^j m_0)(\mathbf{k}) = 0\ \forall |\mathbf{k}| > k_0$, for any $j \geq 1$. Therefore $\mathcal{F}(\widehat{m})(\mathbf{k}) = 0$ $\forall |\mathbf{k}| > k_0$. Let m_ℓ be the ℓ-th iteration of a gradient based iterative scheme, then $\mathcal{F}(r(m_\ell))(\mathbf{k}) = 0$ ($r(m) := \widehat{\mathbf{f}}_{\widehat{\alpha}}^{-1}\Big(\mathbf{f}(m)\Big) - \widehat{\mathbf{f}}_{\widehat{\alpha}}^{-1}(\mathbf{g}^\delta)$) $\forall |\mathbf{k}| > k_0$. This leads to the fact that $\mathcal{F}\Big(d\Phi[m_\ell](\delta m)\Big)(\mathbf{k}) = 0$ for any δm. Therefore, $\mathcal{F}(m_{\ell+1})(\mathbf{k}) = 0\ \forall |\mathbf{k}| > k_0$. The rest of the proof follows from an induction. □

For any velocity field that can be written as $m_b + \delta m$ with m_b the prediction of the trained neural network and δm outside of the range of the neural network but either has a small amplitude (compared to that of m) or has large amplitude by small support compared to the size of the domain (in which case δm is very localized), we can recover δm with an additional linearized reconstruction

step. We linearize the inverse problem around the network prediction $\mathbf{f}_{\widehat{\alpha}}^{-1}(\mathbf{g}^{\delta})$. The reconstruction can be performed with a classical migration scheme or equivalently by minimizing the following quadratic approximation to the functional (4.20):

$$\Psi_Q(m) = \frac{1}{2}\left\|\mathbf{f}(m_b) + d\mathbf{f}[m_b]\Big(m - m_b\Big) - \mathbf{g}^{\delta}\right\|^2_{[L^2((0,T]\times\Gamma)]^{N_s}} + \frac{\gamma}{2}\|\nabla m\|^2_{L^2(\Omega)}\,, \tag{4.21}$$

where $m_b := \mathbf{f}_{\widehat{\alpha}}^{-1}(\mathbf{g}^{\delta})$.

4.6 Computational implementation

We now provide some details on the implementation of the coupling framework we outlined in the previous section. For computational simplicity, we focus on the implementation in two spatial dimensions even though the methodology itself is independent of the dimension of the problem.

4.6.1 Computational setup

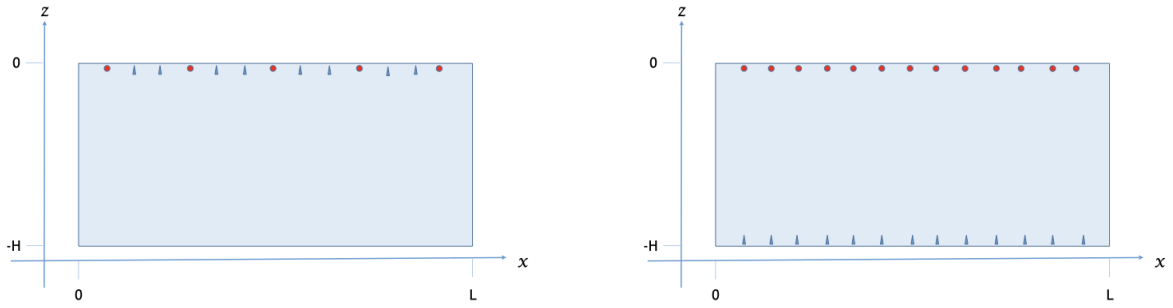


Figure 4.1: The two-dimensional computational domain $\Omega = (0, L) \times (-H, 0)$ for wave propagation. Periodic boundary conditions are imposed on the left and right boundaries. In geophysical applications, sources and detectors are placed on the top boundary (left) while in medical ultrasound applications, sources (red dots) and detectors (blue triangles) can be placed on both the top and the bottom boundaries (right).

For the purpose of being concrete, we first describe briefly the geometrical setting under which we implement the learning and reconstruction algorithms. Let $\mathbf{x} = (x, z)$. The computational domain of interests is $\Omega = (0, L) \times (-H, 0)$. We impose periodic boundary conditions on the left and right boundaries of the domain. Probing sources and detectors are placed on the top and bottom

boundaries $\Gamma_t = (0, L) \times \{0\}$ and $\Gamma_b = (0, L) \times \{-H\}$, depending on the exact applications we have in mind. In geophysical applications, source and detectors are both placed on the top boundary while in medical ultrasound type of applications, source and detectors could be placed on the opposite sides; see Figure 4.1 for an illustration. Under this setup, the wave equation (4.12) with a source $h(t, x)$ on the top boundary and a reflective bottom boundary takes the form

$$\begin{aligned} \frac{1}{m^2}\frac{\partial^2 u}{\partial^2 t} - \Delta u &= 0, && \text{in } (0, T] \times \Omega, \\ u(0, x, z) = \frac{\partial u}{\partial t}(0, x, z) &= 0, && (x, z) \in (0, L) \times (-H, 0), \\ u(t, 0, z) &= u(t, L, z), && (t, z) \in (0, T] \times (-H, 0), \\ \frac{\partial u}{\partial z}(t, x, -H) &= 0, && (t, x) \in (0, T] \times (0, L), \\ \frac{\partial u}{\partial z}(t, x, 0) &= h(t, x), && (t, x) \in (0, T] \times (0, L). \end{aligned} \tag{4.22}$$

Similar equations can be written down for other types of source-detector configurations.

4.6.2 The neural network for learning

With the above computational setup, we can generate the training dataset (4.16) by solving the wave equation (4.22) with given source functions. We will describe in detail how the training dataset is generated, including the spatial-temporal discretization of the wave equation (4.22).

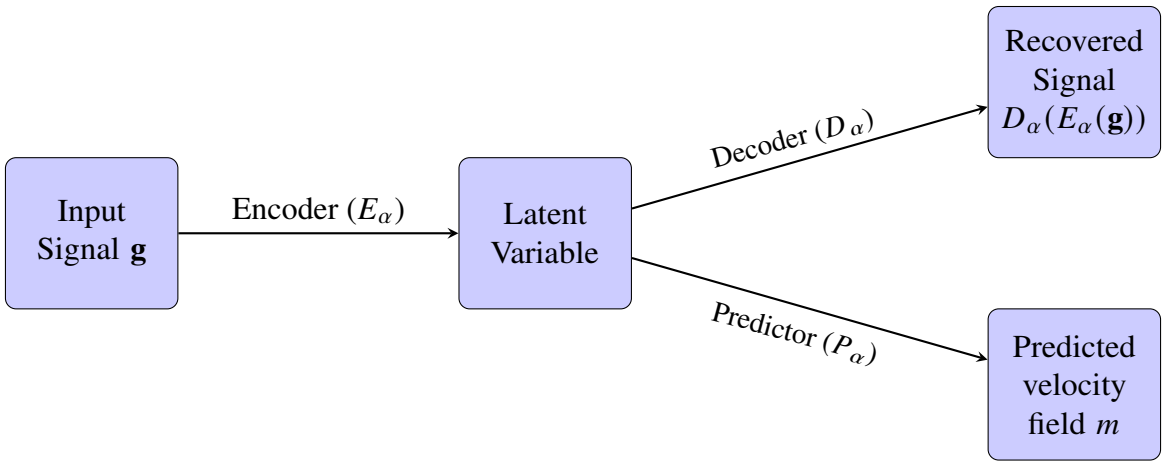


Figure 4.2: Network flow for learning the approximate inverse operator. Training objective is to select α such that $\mathbf{g} = D_\alpha(E_\alpha(\mathbf{g}))$ and $m = P_\alpha(E_\alpha(\mathbf{g}))$ for every datum pair $(\mathbf{g}, m)$.

We construct an autoencoder network scheme to represent the inverse operator. The learning architecture contains three significant substructures: an encoder network E_α, a decoder network D_α, and an additional predictor network P_α; see Figure 4.2 for an illustration of the network flow. More information on the training process and the construction of the encoder, the decoder and the predictor is documented in Appendix B. The encoder-decoder substructure is trained to regenerate the input data, while the predictor reads the latent variable to predict velocity field m. In terms of the input-output data, the network training aims at finding the network parameter α such that

$$\mathbf{g}_j = D_\alpha(E_\alpha(\mathbf{g}_j)) \;\text{ and }\; m_j = P_\alpha(E_\alpha(\mathbf{g}_j)),\;\; 1 \le j \le N\,. \tag{4.23}$$

This is done by a minimization algorithm that minimizes a combined ℓ^1-ℓ^2 loss function with the ℓ^1 loss for the encoder-decoder substructure while ℓ^2 for the encoder-predictor substructure. More precisely, we train the network by solving

$$\begin{gathered}\widehat{\alpha} = \arg\min_{\alpha\in\Theta} \widetilde{\mathcal{L}}(\alpha),\\ \text{with}\\ \widetilde{\mathcal{L}}(\alpha) := \frac{1}{N}\sum_{j=1}^{N} \|\mathbf{g}_j - D_\alpha(E_\alpha(\mathbf{g}_j))\|_{\ell^1} + \frac{1}{2N}\sum_{j=1}^{N} \|\boldsymbol{\mu} \circledast \left(\mathfrak{m}_j - P_\alpha(E_\alpha(\mathbf{g}_j))\right)\|_{\mathbb{M}}^2\,.\end{gathered} \tag{4.24}$$

While the ℓ^1 loss for the encoder-decoder substructure is standard in the learning literature, the second part of the loss function is simply what we introduced in (4.17). Once the training is performed, the approximated inverse is taken as

$$\mathbf{f}_{\widehat{\alpha}}^{-1} := P_{\widehat{\alpha}} \circ E_{\widehat{\alpha}}\,.$$

Let us emphasize that the primary motivation for us to adopt this autoencoder framework, instead of directly training a network for $\mathbf{f}_{\widehat{\alpha}}^{-1}$, is to take advantage of the commonly observed capability of autoencoders to identify lower dimension features from high-dimensional input data. That is,

very often, one can train the autoencoder such that the latent variable $E_\alpha(\mathbf{g})$ contains most of the valuable information in $\mathbf{g}$ but has a much lower dimension than $\mathbf{g}$. This lowers the dimension of the predictor network and therefore makes it easier to train the overall network. Moreover, the weighted optimization we used in the encoder-predictor substructure further stabilizes the learning process by focusing on matching the lower-frequency components of the output.

4.6.3 Learning-assisted FWI inversion

We tested two different algorithms to implement the pre-conditioned FWI reconstruction method, that is, the solution to the least-squares optimization problem (4.19).

Quasi-Newton method with adjoint state. We implemented a quasi-Newton method based on the BFGS gradient update rule [49] for the numerical reconstruction. This BFGS optimization algorithm itself is standard, so we will not describe it in detail here. The algorithm requires the gradient of the loss function $\Phi(m)$ defined in (4.20). We evaluate the gradient with a standard adjoint state method. The procedure is documented in Algor 1 . The main complication that the learning stage brings into the adjoint state calculation is that we will need the transpose of the gradient of the neural network with respect to its input. This imposes restrictive accuracy requirements on the training of the neural network in the sense that we need the network to learn not only the map from measurement to the velocity field but also the derivative of the operator.

Neumann series method. The Neumann series method based on (4.11) is more training friendly since it does not require the adjoint operator of the learned approximate inverse $\widehat{\mathbf{f}}_{\widehat{\alpha}}^{-1}$. We implemented a J-term truncated Neumann series approximation

$$\widehat{m} = \sum_{j=0}^{J-1} K^j \left(\widehat{\mathbf{f}}_{\widehat{\alpha}}^{-1}(\mathbf{g}^\delta) \right) . \tag{4.25}$$

The computational procedure is summarized in Algorithm 2 in Appendix B.

4.7 Numerical experiments

We now present some numerical simulations to illustrate some of the main characteristics of the proposed framework of coupling deep learning with model-based FWI reconstruction. We fix the computational domain to be $\Omega = [0, 1] \times [-1, 0]$, that is, $L = H = 1$. In this proof-of-concept study, we use acoustic source functions that can generate data at all frequencies. We leave it as future work to consider the situation where low-frequency wavefield data are impossible to measure in applications such as seismic imaging.

4.7.1 Velocity feature models

In this work, we consider two different feature models for the output velocity field of the neural network.

Generalized Fourier feature model. In the first model, we represent $m(\mathbf{x})$ as linear combinations of the Laplace-Neumann eigenfunctions on the computational domain Ω. To be precise, let $(\lambda_{\mathbf{k}}, \varphi_{\mathbf{k}})$ ($\mathbf{k} = (k_x, k_z) \in \mathbb{N}_0 \times \mathbb{N}_0$) be the eigenpair of the eigenvalue problem:

$$-\Delta\varphi = \lambda\varphi, \quad \text{in } \Omega, \qquad \mathbf{n} \cdot \nabla\varphi = 0, \quad \text{on } \partial\Omega\,.$$

where $\mathbf{n}(\mathbf{x})$ is the unit outward normal vector of the domain boundary at $\mathbf{x} \in \partial\Omega$. Then $\lambda_{\mathbf{k}} = (k_x\pi)^2 + (k_z\pi)^2$, and

$$\varphi_{\mathbf{k}}(x, z) = \cos(k_x\pi x)\cos(k_z\pi z)\,.$$

In our numerical simulations, we take

$$m(\mathbf{x}) = \sum_{k_x,k_z=0}^{M} \mathfrak{m}(\mathbf{k})\ \varphi_{\mathbf{k}}(x, z)\,, \tag{4.26}$$

for some given M. The generation of the random coefficients $\mathfrak{m}(\mathbf{k})$ will be described in detail in the next section.

Gaussian mixture model. The second feature model we take is the Gaussian mixture model. More precisely, we represent $m(\mathbf{x})$ as a superposition of Gaussian functions:

$$m(\mathbf{x}) = m_0 + \sum_{k=1}^{M} c_k e^{-\frac{1}{2}(\mathbf{x}-\mathbf{x}_0^k)^{\mathfrak{T}}\Sigma_k^{-1}(\mathbf{x}-\mathbf{x}_0^k)}. \tag{4.27}$$

With a small number of highly localized Gaussians, successful reconstruction of such a model could provide inside on source locating problems in seismic applications [24]. This is the primary motivation for us to consider this model.

4.7.2 Learning dataset generation

To generate training data, we generate a set of velocity fields and then solve the wave equation model (4.22) with source functions $\{h_s\}_{s=1}^{N_s}$ to get the corresponding wave field data at the detectors.

Generating velocity fields. We first construct a set of N random velocity fields $\{m_j\}_{j=1}^{N}$ using the representation (4.26) or (4.27). We do this by randomly choosing the coefficients $\{\mathfrak{m}(\mathbf{k})\}_{\mathbf{k}\in\mathbb{N}_0\times\mathbb{N}_0}$ from the uniform distribution $\mathcal{U}[-0.5, 0.5]$ when considering the model (4.26) and the coefficients c_k from $\mathcal{U}[0, 5]$, $\mathbf{x}_0^k$ from $\mathcal{U}(-H, 0) \times \mathcal{U}(0, L)$, $(\Sigma_k)_{ij}$ from $\mathcal{U}[0, 0.2] + 0.1$ and $m_0 = 10$ when using the model (4.27). To mimic the frequency content of realistic velocity fields, we force the coefficient $\mathfrak{m}(\mathbf{k})$ in the random Fourier model (4.26) to decay asymptotically as

$$\mathfrak{m}(\mathbf{k}) \sim \mathfrak{m}(\mathbf{k})[(k_x + 1)(k_z + 1)]^{-\beta}, \quad \text{for large } |\mathbf{k}| = \sqrt{k_x^2 + k_z^2} \tag{4.28}$$

with $\beta \geq 0$ given in the concrete examples later.

To make sure that the velocity fields we generated are physically meaningful, we rescale them so that the velocity lives in a range $[\underline{m}, \overline{m}]$ $(0 < \underline{m} < \overline{m} < +\infty)$. The linear rescaling is done through the operation

$$m(\mathbf{x}) \leftarrow \frac{\overline{m} - \underline{m}}{m^* - m_*} m(\mathbf{x}) + \frac{\underline{m} m^* - \overline{m} m_*}{m^* - m_*}, \tag{4.29}$$

where $m^* := \max_{\mathbf{x}} m(\mathbf{x})$ and $m_* := \min_{\mathbf{x}} m(\mathbf{x})$.

In Figure 4.3 we show typical samples of the velocity field generated from the aforementioned process. The top panel of Figure 4.3 shows the surface plots of 4 different randomly generated velocity fields using the model (4.26) with $M = 4$. The bottom panel presents the surface plots of 4 random realizations of the velocity field given by the model (4.27) with $M = 2$. Random noise at different levels will be added to the sampled velocity fields to study the generalization of the learning scheme we have. The exact level of noise will be given later in concrete examples.

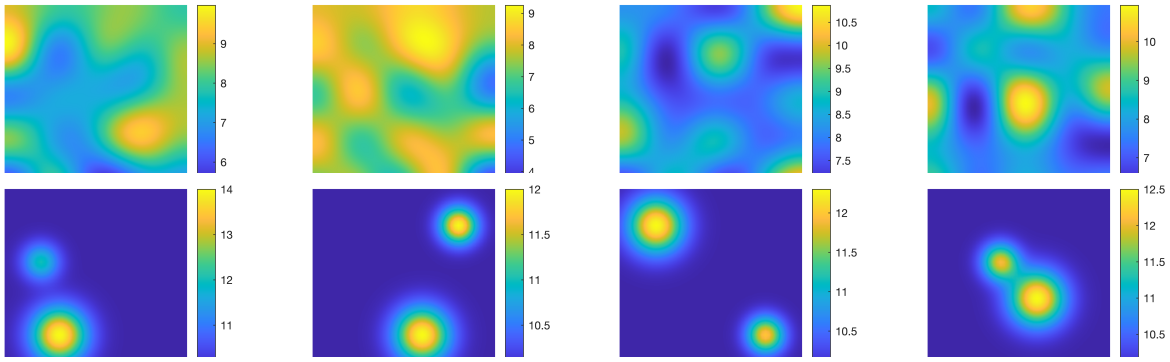


Figure 4.3: Random samples of the velocity field for training of the neural networks. Top row: velocity fields generated from (4.27) with $M = 4$; bottom row: velocity fields generated from (4.26) with $M = 2$.

Finite difference scheme for the wave equation. We use the time-domain stagger-grid finite difference scheme that adopts a second-order in both the time and the spatial directions to solve the wave equation (4.22). Precisely, the discretization is performed with elements over the Cartesian grids formed by $(x_k, z_l) = (k\Delta x, l\Delta z), k, l = 0, 1, ..., K$ with $\Delta x = L/K$ and $\Delta z = H/K$. The receivers are equally placed at the bottom surface, coinciding with the grid points, as documented in the right panel of Figure 4.1, namely, there are $K + 1$ receivers for each velocity model. We then record the wave signal starting at time t_0 and take another shot every $j\Delta t$ until the termination time T, here, j is a positive integer and Δt is the uniform time step size for the forward wave solver. As an example for illustration, we take

$$h(t, x) = e^{\frac{-(x-0.6)^2}{0.01}} - e^{\frac{-(x-0.3)^2}{0.01}} \tag{4.30}$$

L	H	K	Δt	t_0	j	T
1	1	50	0.0005	0	20	0.5

Table 4.1: Values of parameters in the spatial and temporal discretization of the wave equation and the time node of the recorded wave signal.

to be the top source in (4.22) and present the recorded time series wave signals in Figure 4.4. Table 4.1 summarizes the parameters we used to generate these wavefield signals.

Figure 4.4, from the left panel to the right panel, shows the time series wave signals at the bottom surface generated from the velocity model satisfying (4.27) with $M = 2$, and the velocity model satisfying (4.26) with $M = 4$, respectively; from the top panel to the bottom panel are the wave signals without noise, with 10% multiplication Gaussian noise, and with 10% additive Gaussian noise, respectively.

Last, we note that to obtain a reliable learning dataset, one needs to guarantee the stability of the time integrator when solving (4.22). Recall that the second order time-domain stagger-grid finite difference forward wave solver is stable under the following CFL condition

$$\Delta t \leq \frac{\min\{\Delta x, \Delta z\}}{\sqrt{2}\max_{\mathbf{x}}\{m(\mathbf{x})\}}. \tag{4.31}$$

To guarantee the stability of the forward solver for all velocity samples, we force

$$\Delta t = \Delta t^* < \frac{\min\{\Delta x, \Delta z\}}{\sqrt{2}\max_{\mathbf{x}}\{\overline{m}(\mathbf{x})\}},$$

where $\overline{m}$ is used in the scaling (4.29), for the data generation of the offline training stage. In this work, we set $\Delta t^* = 0.0005$ as shown in Table 4.1 based on our setting.

4.7.3 Training and testing performance

We now present a systematic numerical exploration of the training and testing performance of the offline training stage. Given that the training and application of the Gaussian mixture velocity model (4.27) with a small amount of Gaussians functions is exceptionally successful (due to the

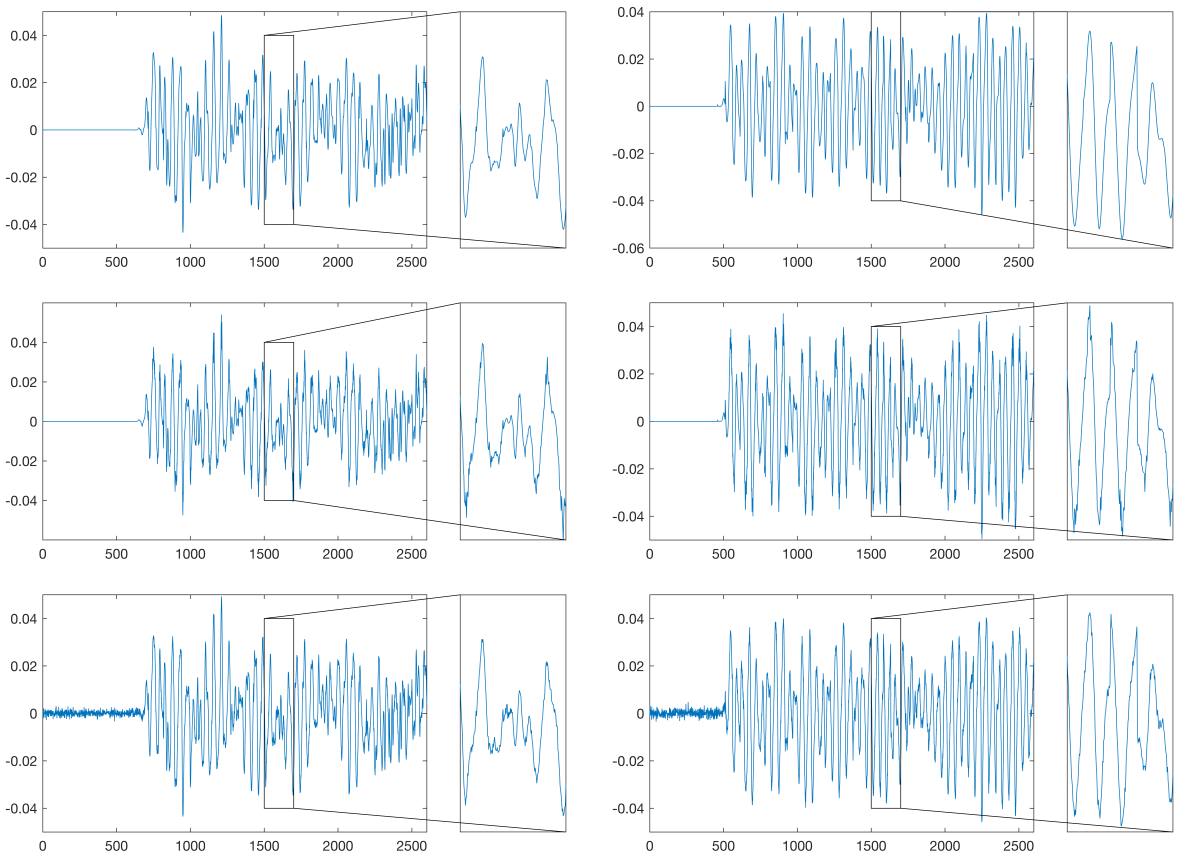


Figure 4.4: The left panel presents time series wave signals at the bottom surface generated from a velocity model satisfying (4.27) with $M = 4$, while the right panel shows time series wave signals at the bottom surface generated from a velocity model constructed by (4.26) with $M = 2$. From the top to the bottom are time series wave signals without noise, with 10% multiplication Gaussian noise and with 10% additive Gaussian noise, respectively.

smallness of the parameter space) according to our numerical experience, we will focus on the training of the generalized Fourier velocity model (4.26).

Training dataset size. We first emphasize that the training results we show in this section are obtained on a very small dataset in the following sense. The number of data points in the artificial dataset $\{\mathbf{g}_j, m_j\}_{j=1}^N$ is small with $N = 10^6$. Moreover, for each m_j, we collect the wavefield from $N_s = 3$ illumination sources and $N_d = 51$ detectors. Those source-detector pairs are a subset of the source-detector pairs for the dataset we used in the reconstruction step. Moreover, at each detector, we use only data at 51 time steps out of the 1000 time steps in the numerical solutions. This small dataset is used so that we can handle the computational cost of the training process with our limited

computing resources. It is also intentionally done to demonstrate that one can train reasonable approximate inverse with a significantly smaller dataset if one is willing to sacrifice a little of the training accuracy.

Training-testing dataset split. We perform a standard training validation cycle on the neural network approximate inverse. Before the training process starts, we randomly split the artificial dataset of $N = 10^6$ data points into a training dataset and a testing dataset. The training dataset takes 80% of the original dataset, while the test dataset takes the rest 20% of the data points. The training dataset and the validation dataset have no intersection, namely, no data points in the validation set are present in the training dataset.

Random Fourier velocity model: case of non-decaying coefficients

We start with the most challenging scenario where we train the neural network to approximate the inverse operator for the velocity model (4.26) with randomly generated Fourier coefficients without any decay requirement on the coefficients, that is, we set the decay rate $\beta = 0$ in (4.28). This is a highly challenging case because the effective parameter space of this class of velocity models grows exponentially with respect to the number of Fourier models we have in the model. Ideally, one would need an exponentially large training dataset in order to have reasonable training results. However, due to the smooth property of the map $\mathbf{f} : m \mapsto \mathbf{g}$, we demonstrate below that with a relatively small dataset and a very limited number of source-detector pairs and time shots, our training result is fairly encouraging.

In Figure 4.5, we show three randomly selected velocity fields (m) from the testing dataset, the corresponding neural network predictions ($\widetilde{m} = \widehat{\mathbf{f}}_{\widehat{\alpha}}^{-1}(\mathbf{f}(m))$), and the error in the prediction ($m - \widetilde{m}$). The largest number of Fourier modes allowed in these learning processes is 10, meaning that $0 \leq k_x, k_z \leq 9$ in the velocity model (4.26). The training output is a 10×10 matrix containing the content of $\mathfrak{m}(\mathbf{k})$ in (4.26). The output space is therefore 100-dimensional. A naive visual inspection of the results in Figure 4.5 shows that the training process is quite successful as the

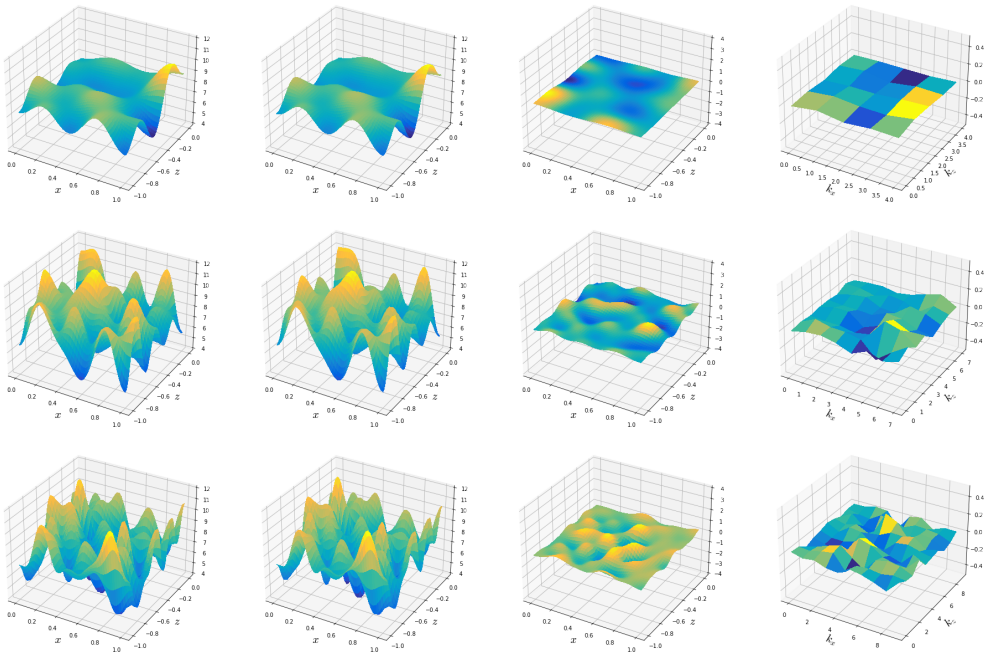


Figure 4.5: Three randomly selected velocity fields from the testing dataset: 5×5 coefficients Fourier model, 8×8 coefficients Fourier model, 10×10 coefficients Fourier model. All true cases have decay rate $\beta = 0$ (column 1), the corresponding predictions by the trained neural network (column 2), the error of the prediction (column 3), and the error in the neural network prediction ($\widetilde{m}(\mathbf{x})$) in the Fourier domain ($\mathfrak{m}(\mathbf{k}) - \widetilde{\mathfrak{m}}(\mathbf{k})$) (column 4).

testing errors seem to be pretty reasonable, especially given that our training dataset is fairly small (0.8×10^6 data points to be precise). While it is expected that when the number of Fourier modes allowed in the velocity model is huge, the validation error will be sufficiently large if we keep the training sample size, we do observe that validation error is quite small in general for cases when less than 10×10 Fourier modes are pursued in the learning process. Increasing computational power would certainly improve training quality.

Let us remark that our training results indeed show that we have better accuracy in learning the low-frequency components of the inverse operator, as we discussed in the previous sections of the work. In the right column of Figure 4.5, we provide the Fourier coefficients of the errors in the network prediction. In all velocity fields, we see larger errors in the higher-frequency components of the network velocity recovery. This is a universal phenomenon that we observed over the testing dataset.

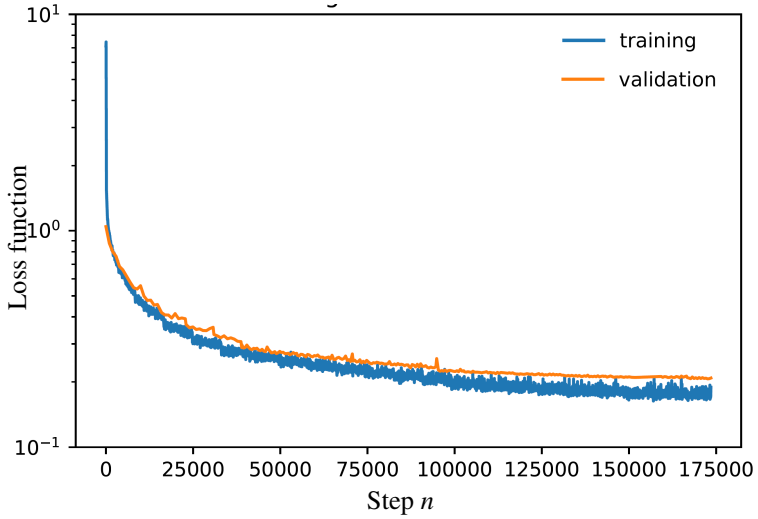


Figure 4.6: Training and validation loss curves for a typical learning experiment. Very similar curves are observed for each of the learning experiments we performed.

To dive a little more into the training quality and the optimization landscape after applying our neural network pre-conditioner, we offer in Figure 4.6 the training-validation loss curves for a typical learning experiment. We observe very similar curves for training and validating with the velocity model (4.26) with different total numbers of Fourier modes. We measure the training accuracy quantitatively with the size of the operator $\mathcal{I} - \widehat{\mathbf{f}}_{\widehat{\alpha}}^{-1} \circ \mathbf{f}$. More precisely, we evaluate the three main quantities for a data point $(\mathbf{g}, m)$ in the testing dataset:

(i) The error in the network prediction of Fourier modes of m:

$$\Delta\mathfrak{m}(\mathbf{k}) := \mathfrak{m}(\mathbf{k}) - \mathbf{f}_{\widehat{\alpha}}^{-1} \circ \mathbf{f}(m)\,.$$

(ii) The landscape of the classical functional $\Psi(m)$ evaluated along a line in the direction of a given Fourier mode of m, $\varphi_{\mathbf{k}}$, passing through two different points $m = \mathbf{f}^{-1}(\mathbf{g})$ and $m_{\text{net}} = \widehat{\mathbf{f}}_{\widehat{\alpha}}^{-1}(\mathbf{g})$:

$$\Psi_{\mathcal{O}}(h;\mathbf{k}) := \|\mathbf{g} - \mathbf{f}(m_0 + h\varphi_{\mathbf{k}})\|^2_{[L^2((0,T]\times\Gamma)]^{N_s}}, \quad m_0 = \mathcal{O}(m), \quad \mathcal{O} \in \{\mathcal{I}, \widehat{\mathbf{f}}_{\widehat{\alpha}}^{-1} \circ \mathbf{f}\}\,.$$

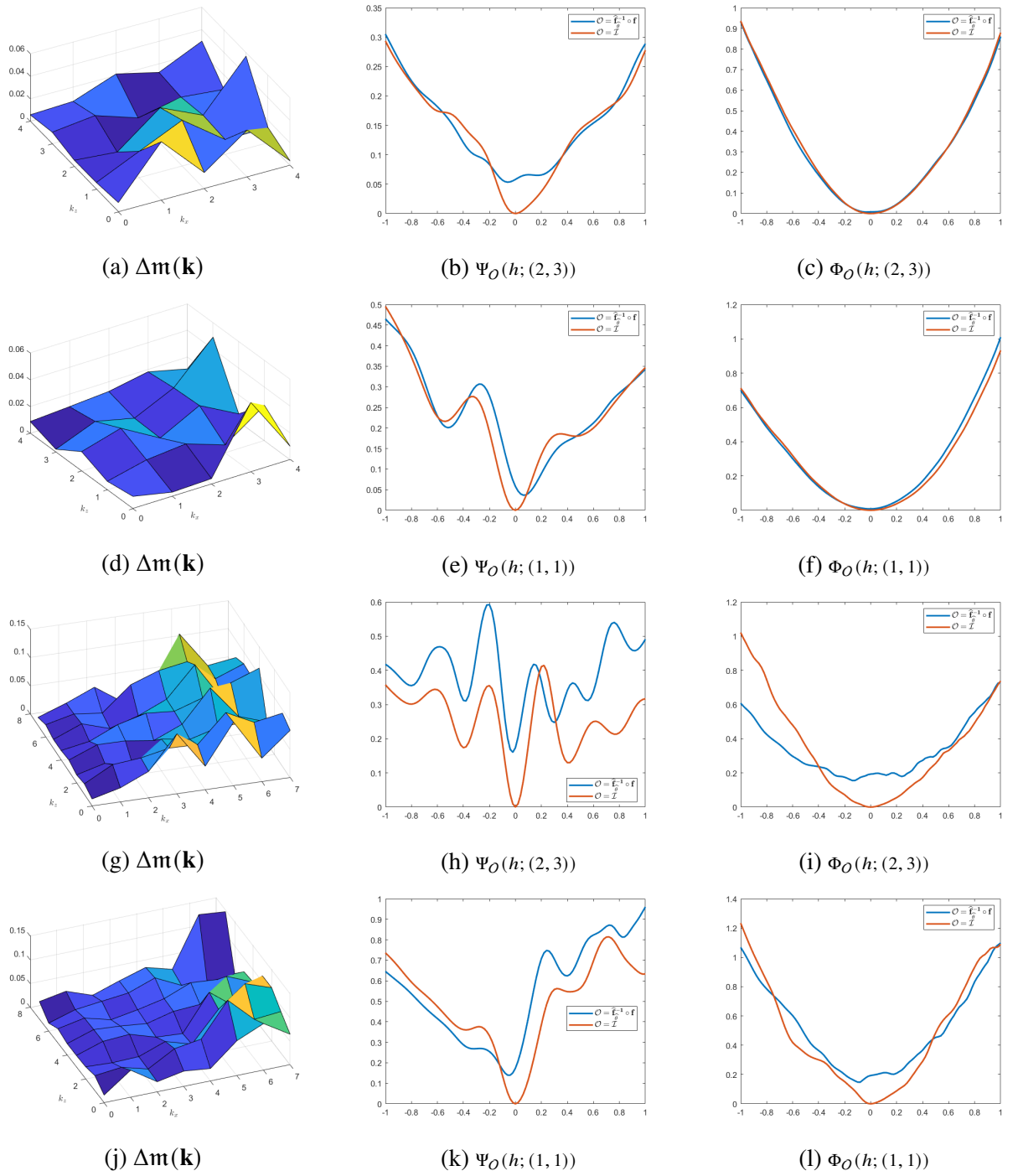
(iii) The landscape under our pre-conditioner, the new mismatch function $\Phi(m)$ evaluated as in (ii):

$$\Phi_O(h;\mathbf{k}) := \|\widehat{\mathbf{f}}_{\widehat{\alpha}}^{-1}(\mathbf{g}) - \widehat{\mathbf{f}}_{\widehat{\alpha}}^{-1} \circ \mathbf{f}(m_0 + h\varphi_{\mathbf{k}})\|^2_{L^2(\Omega)}, \quad m_0 = O(m), \quad O \in \{\mathcal{I}, \widehat{\mathbf{f}}_{\widehat{\alpha}}^{-1} \circ \mathbf{f}\}\,.$$

When a perfect learning is performed, we would have $\Delta\mathfrak{m}(\mathbf{k}) = \mathbb{5}$, $\Psi_{\mathcal{I}}(h;\mathbf{k}) = \Psi_{\widehat{\mathbf{f}}_{\widehat{\alpha}}^{-1}\circ\mathbf{f}}(h;\mathbf{k})$, and $\Phi_{\mathcal{I}}(h;\mathbf{k}) = \Phi_{\widehat{\mathbf{f}}_{\widehat{\alpha}}^{-1}\circ\mathbf{f}}(h;\mathbf{k})$ for any $(\mathbf{g}, m)$ in the training dataset, and $\Delta\mathfrak{m}(\mathbf{k})$ small, $\Psi_{\mathcal{I}}(h;\mathbf{k}) \approx \Psi_{\widehat{\mathbf{f}}_{\widehat{\alpha}}^{-1}\circ\mathbf{f}}(h;\mathbf{k})$, and $\Phi_{\mathcal{I}}(h;\mathbf{k}) \approx \Phi_{\widehat{\mathbf{f}}_{\widehat{\alpha}}^{-1}\circ\mathbf{f}}(h;\mathbf{k})$ for any $(\mathbf{g}, m)$ in the testing dataset.

In Figure 4.7, we show plots of $\Delta\mathfrak{m}(\mathbf{k})$ (left column), $\Psi_{\mathcal{I}}(h;\mathbf{k})$ (red line) and $\Psi_{\widehat{\mathbf{f}}_{\widehat{\alpha}}^{-1}\circ\mathbf{f}}(\mathbf{k})$ (blue line) (middle column), and $\Phi_{\mathcal{I}}(h;\mathbf{k})$ (red line) and $\Phi_{\widehat{\mathbf{f}}_{\widehat{\alpha}}^{-1}\circ\mathbf{f}}(\mathbf{k})$ (blue line) (right column), for four randomly selected $(\mathbf{g}, m)$ pairs in the testing dataset. Shown are results for $\mathbf{k} = (2, 3)$ and $\mathbf{k} = (1, 1)$. Very similar behavior are observed along other coordinates $\varphi_{\mathbf{k}}$.

The plots in Figure 4.7 provide a quantitative description of the accuracy of the trained network. They clearly indicates that the trained $\widehat{\mathbf{f}}_{\widehat{\alpha}}^{-1}$ is indeed a good approximation to $\mathbf{f}^{-1}$. Moreover, a comparison of the second column and the third column gives the impression that along with the coordinates we plotted, the new objective functional Φ in (4.20) has a much better landscape than the classical Ψ in (4.15). This is what we observed in other coordinates that are not shown here as well. Therefore, the trained neural network $\widehat{\mathbf{f}}_{\widehat{\alpha}}^{-1}$ can work as a nonlinear pre-conditioner to improve convexify of the optimization landscape. Moreover, the plots provided a good indication that the trained network is fairly generalizable in the following sense. The Fourier coefficients (including $\mathfrak{m}_{(2,3)}$ and $\mathfrak{m}_{(1,1)}$ shown in the plots) in the training dataset are all randomly drawn in the interval $[-0.5, 0.5]$. Here in the plots, we consider the coefficient values in the range $[-1, 1]$. The agreement of the red and blue lines outside of the training value range $[-0.5, 0.5]$, that is, in the range $[-1, -0.5) \cup (0.5, 1]$, suggests that the trained neural network can be used in a region of coefficient values that is far larger than its training domain.



(a) $\Delta\mathfrak{m}(\mathbf{k})$ (b) $\Psi_{\mathcal{O}}(h;(2,3))$ (c) $\Phi_{\mathcal{O}}(h;(2,3))$

(d) $\Delta\mathfrak{m}(\mathbf{k})$ (e) $\Psi_{\mathcal{O}}(h;(1,1))$ (f) $\Phi_{\mathcal{O}}(h;(1,1))$

(g) $\Delta\mathfrak{m}(\mathbf{k})$ (h) $\Psi_{\mathcal{O}}(h;(2,3))$ (i) $\Phi_{\mathcal{O}}(h;(2,3))$

(j) $\Delta\mathfrak{m}(\mathbf{k})$ (k) $\Psi_{\mathcal{O}}(h;(1,1))$ (l) $\Phi_{\mathcal{O}}(h;(1,1))$

Figure 4.7: Plots of $\Delta\mathfrak{m}(\mathbf{k})$ (first column), $\Psi_{\mathcal{O}}(h;\mathbf{k})$ (second column), and $\Phi_{\mathcal{O}}(h;\mathbf{k})$ (third column) for four different $(\mathbf{g}, m)$ pairs in the testing dataset. The velocity model for rows 1-2 has $M = 4$ and that for the plots in rows 3-4 has $M = 7$.

Random Fourier velocity model: case of decaying coefficients

While training the neural network for an approximate inverse $\widehat{\mathbf{f}}_{\widehat{\alpha}}^{-1}$ on a large space of velocity field is extremely useful for generalization purposes, it also poses significant challenges when the number of Fourier modes included in the model gets very large. Not only will we need an exponentially larger training dataset, but also the training process takes exponentially growing computational power. This is what we observed in our numerical experiments. In this section, we show some training-validation results for the velocity model (4.26) with decaying Fourier coefficients following the pattern we imposed in (4.28). We present results from two different cases: the slow decay case with $\beta = 1/2$ and the fast decay case with $\beta = 1$.

In Figure 4.8, we show typical validation results on five randomly selected velocity profiles in the testing dataset. The top two rows are the results for the training of the velocity model with $\beta = 0$, the third row is the case of $\beta = 1/2$ while the bottom two rows are for the case of $\beta = 1$. In both cases, the training is successful, as can be seen from the relatively small errors in the predictions. Plots of the functionals $\Psi_{\mathcal{I}}$ and $\Psi_{\widehat{\mathbf{f}}_{\widehat{\alpha}}^{-1}\circ\mathbf{f}}$ show similar patterns as those in Figure 4.7. We omit those to space. Moreover, prediction errors in the Fourier domain display very similar behavior as observed in the previous subsection: the error is higher for high-frequency components and lower for low-frequency components.

To study the generalization capability of the learned network, we validate the learning with on dataset generated from a different velocity model, which is considered the case where training and testing data samples are from different classes. In Figure 4.9, we train a neural network to recover the first 10×10 Fourier coefficient of the velocity field and validate the trained neural network on a dataset generated from velocity models that contain 20×20 random Fourier modes. The decay rate in this particular case is $\beta = 1$ but similar results are observed for $\beta = 1/2$ as well. The validation results demonstrate that the trained network is reasonably generalizable in our considered setting.

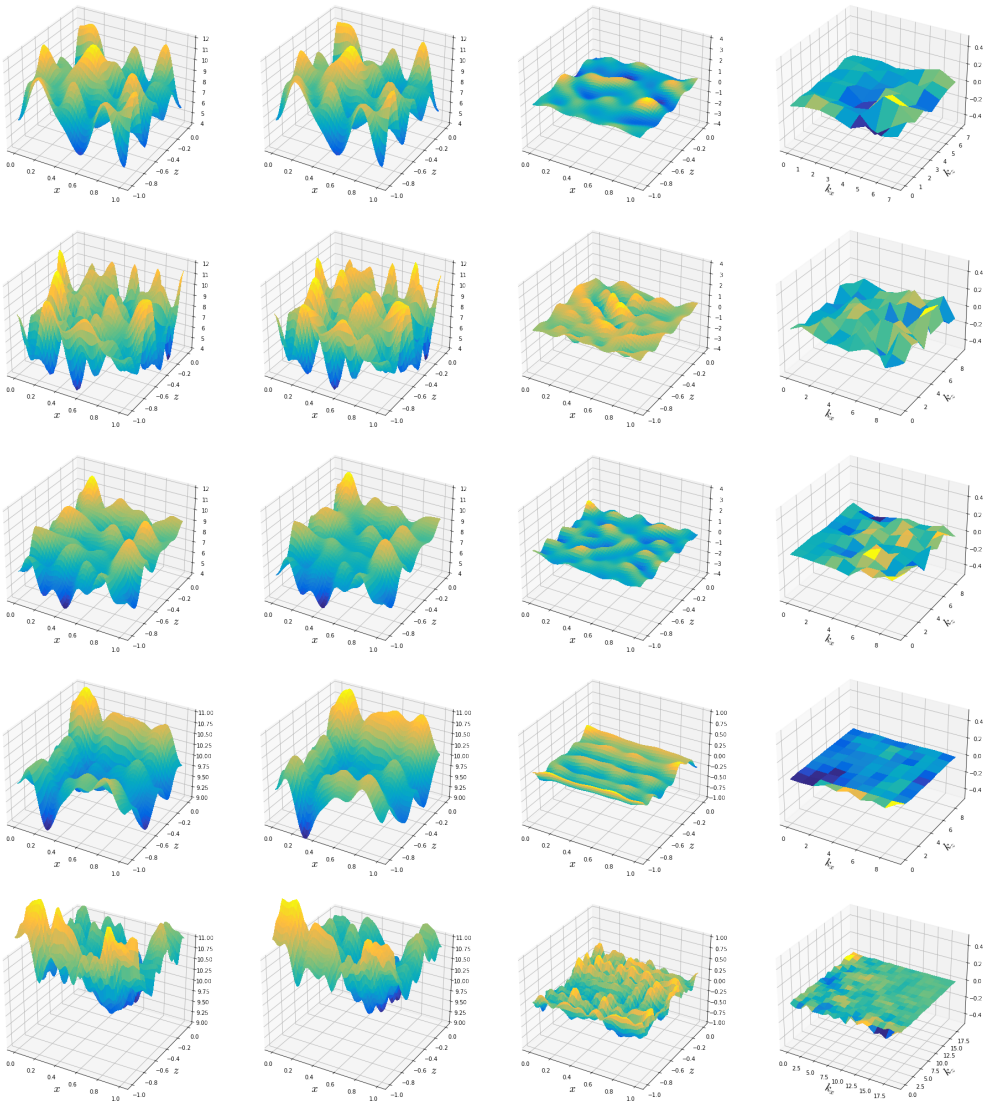


Figure 4.8: Validation results on four specific velocity fields in the testing dataset. Row 1 is the results for 8 Fourier velocity model with $\beta = 0$, Row 2 is the results for 10 Fourier velocity model with $\beta = 0$, row 3 is the results for 10 Fourier velocity model with $\beta = 1/2$, row 4 is the results for 10 Fourier velocity model with $\beta = 1$ while row 5 is are results for 20 Fourier velocity model with $\beta = 1$. From left to right are: the true velocity field, neural network prediction, the error of the prediction, and the error of the prediction in the Fourier domain.

Mesh-based velocity model

In the last training-validation numerical experiment, we demonstrate that the phenomena observed in the previous subsections are not particularly due to the Fourier parameterization of the velocity field in (4.26) that we used. Indeed, the results are more related to our method of training.

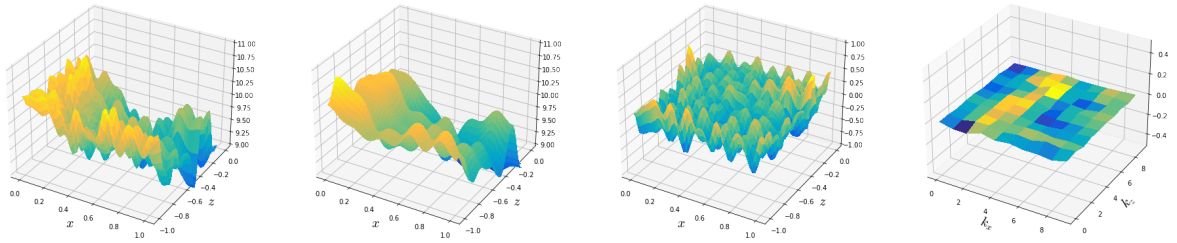


Figure 4.9: The instance of validation of learning results in a different class of velocity models for the case of $\beta = 1$. Shown from left to right are: the true velocity field, the neural network prediction, the error in the prediction, and the error in the Fourier domain.

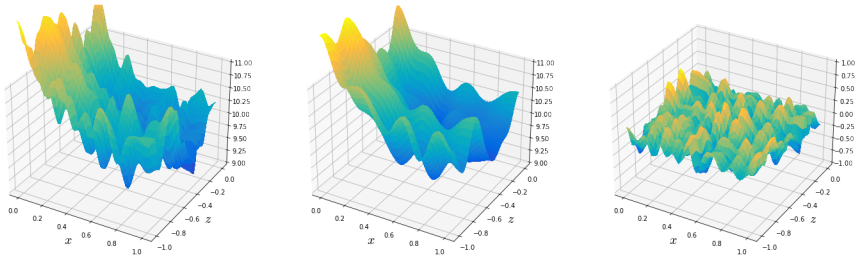


Figure 4.10: Out-of-domain validation of a training result with mesh-based velocity representation. Shown from left to right are: the true velocity field, the network prediction, and the error in the prediction.

Here we perform the same type of training on a neural network whose output is the velocity field represented on a 51×51 uniform mesh on the domain Ω. The output space is therefore much larger compared to the training in the case of the random Fourier velocity model. However, after projecting into the Fourier space, the training result has almost identical properties to what we observed in the random Fourier model. In Figure 4.10, we show the out-of-domain validation result for the training. The velocity fields that generated the training dataset have 10×10 Fourier modes while the velocity fields in the validation dataset have 20×20 Fourier modes (but represented on a 51×51 uniform mesh), both generated with $\beta = 1$. The relatively small validation errors indicate that the training is fairly successful and reasonably generalizable. The computational cost, in this case, is much larger than those in the previous subsections since the neural network has a larger size due to the increased size of the network output.

4.7.4 Learning-assisted FWI reconstruction

In this section, we present inversion results for some simulated datasets to verify the efficiency and stability of the proposed coupling method. All simulations on the inversion stage are conducted on a quadcore Intel Core i7 with 16 GB RAM.

Convexity of the new loss function

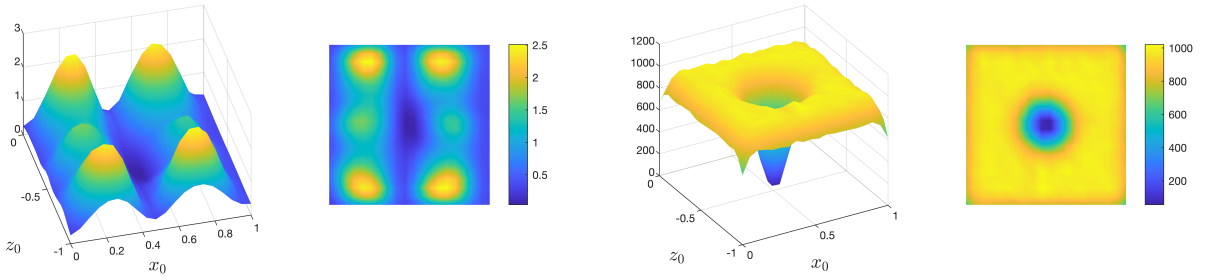
Lemma 4.1.1 indicates that if we have relatively accurate training, the new loss function for our coupled reconstruction scheme behaves similarly to the functional $\|m - m_0\|^2_{L^2(\Omega)}$, m_0 being the true solution. Figure 4.7 provided some evidence of this in the training of the random Fourier model. In the one coefficient case, plots in Figure 4.7 clearly show that the new loss function is almost convex. We now present some numerical evidence in the case of the Gaussian mixture velocity model. In particular, we are interested in seeking convexity with respect to the location of a Gaussian perturbation. More precisely, the velocity field $m(\mathbf{x})$ is set to be a single Gaussian model with $M = 1$ in (4.27), that is,

$$\begin{aligned} &m(\mathbf{x}) = m_0 + c_1 e^{-\frac{1}{2}(\mathbf{x}-\mathbf{x}_0^1)^{\mathfrak{T}}\Sigma_1^{-1}(\mathbf{x}-\mathbf{x}_0^1)}, \\ &\mathbf{x}_0^1 = (x_0^1, z_0^1), \\ &\Sigma_1 = \begin{pmatrix} \sigma_1^2 & 0 \\ 0 & \sigma_1^2 \end{pmatrix} \end{aligned} \tag{4.32}$$

where the background velocity m_0, the amplitude c_1, and the variance σ_1 are fixed to be $(m_0, c_1, \sigma_1) = (10, 5, 0.1)$. We then present the objective functions $\Psi(m)$ and $\Phi(m)$ ($\gamma = 0$) with respect to the location (x_0^1, z_0^1) in Figure 4.11. The setting of the offline training stage for generating $\widehat{\mathbf{f}}^{-1}_{\widehat{\alpha}}$ is the same as those in Section 4.7.2.

Figure 4.11 presents the landscapes of objective functions $\Psi(m)$ and $\Phi(m)$ ($\gamma = 0$) with fixed (m_0, c_1, σ_1). In particular, we set $(x_0^1, z_0^1) = (0.5, -0.5)$ as the ground true velocity model which generates the wave signal $\mathbf{g}$. From Figure 4.11, we observe that, (i) the classical loss function $\Psi(m)$

is not a convex function, and its landscape shows that the optimization can be easily trapped into a local minimum if the initial model is not carefully chosen; (ii) the new loss function $\Phi(m)$ ($\gamma = 0$) for the proposed coupling method becomes more convex which is consistent with Lemma 4.1.1. In addition, we note that when the initial model is close enough to the exact model (located at the convex region of the misfit function), the global minimum is guaranteed and one can also expect a fast convergence. In fact, a good initial model under the setting of the proposed coupling scheme can be easily obtained by adding a small perturbation to $\widehat{\mathbf{f}}_{\widehat{\alpha}}^{-1}(\mathbf{g})$ as indicated by Neumann series (4.11).



(a) Landscape of $\Psi(m)$ (b) Landscape of $\Phi(m)$

Figure 4.11: The landscape of the classical (left) and new (right) objective functions for the location of a Gaussian perturbation of the velocity field.

Inversion for the velocity model (4.27) **with** $M = 2$

The first inversion example is performed to recover the following mixed Gaussian velocity model (4.27) with $M = 2$ and $m_0 = 10$,

$$m(\mathbf{x}) = 10 + \sum_{k=1}^{2} c_k e^{-\frac{1}{2}(\mathbf{x}-\mathbf{x}_0^k)^T \Sigma_k^{-1}(\mathbf{x}-\mathbf{x}_0^k)}, \quad \mathbf{x}_0^k = (x_0^k, z_0^k), \quad \Sigma_k = \begin{pmatrix} \sigma_k^2 & 0 \\ 0 & \sigma_k^2 \end{pmatrix}. \tag{4.33}$$

We use the same offline training settings as those for the mixed Gaussian wave signal generation in Section 4.7.2 to construct $\widehat{\mathbf{f}}_{\widehat{\alpha}}^{-1}$ for the online inversion stage. However, to generate versatile wave signals at the bottom surface to recover the features $\{c_1, c_2, \sigma_1, \sigma_2, x_0^1, x_0^2, z_0^1, z_0^2\}$, we enforce three

different top sources $h_i(x), i = 1, 2, 3$ with

$$h_1(x) = e^{-\frac{(x-0.8)^2}{0.01}} - e^{-\frac{(x-0.2)^2}{0.01}}, \quad h_2(x) = e^{-\frac{(x-0.4)^2}{0.01}} - e^{-\frac{(x-0.7)^2}{0.01}},$$

and

$$h_3(x) = e^{-\frac{(x-0.6)^2}{0.01}} - e^{-\frac{(x-0.3)^2}{0.01}},$$

rather than one single external top source in Section 4.7.2.

For the inversion stage, we implement a J-term truncated Neumann series approximation (4.25) to obtain the reconstructed velocity image. Note that $J = 1$ corresponds to the reconstructed velocity image from the offline training stage. We also add the Gaussian noise with zeros mean and 10% standard derivation to test the stability of the proposed coupling scheme. Figure 4.12 presents the reconstructed images. Precisely, the first three columns show the surface plots of the exact velocity field, the neural network prediction velocity field from the offline training stage, and the reconstructed velocity field with $J = 20$ from the online inversion stage, while the last column displays the difference between the exact velocity field (first column) and the reconstructed velocity field (third column). From the top row to the bottom row of Figure 4.12, we present the results from the noise-free wave signal, the wave signal with 10% multiplication Gaussian noise, and the wave signal with 10% additive Gaussian noise, respectively. We see that the online inversion stage improves the accuracy of the reconstructions for all cases. Table 4.2 lists the L^2/L^∞ errors on the velocity field for the entire computational domain, as well as the CPU time for various implementations with different values of J. There, we note that for the wave signals without noise and with 10% multiplication Gaussian noise, both L^2 and L^∞ reconstruction errors dropped by a factor $\sim 10^4$ within 30 seconds; for the wave signal with 10% additive Gaussian noise, it seems that there is no improvement to add more Neumann terms in (4.25) once the L^2 error reduces to 5.89×10^{-3} and L^∞ error reduces to 4.28×10^{-2}; for this type of the situation, we can use the reconstruction from adding Neumann terms as an initial guess for a gradient-based optimization scheme to further improve the resolution of the reconstruction, see Section 4.6.3.

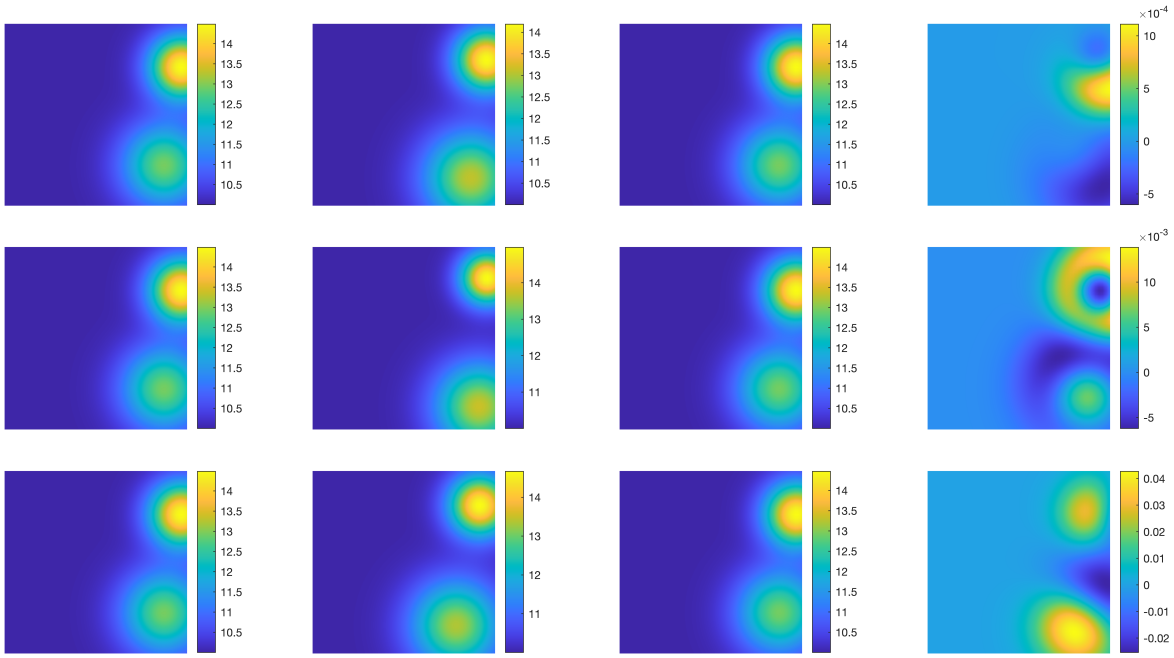


Figure 4.12: The reconstructed velocity images for the mixed Gaussian (4.33). From left to right are the ground true velocity field, the reconstructed velocity field with $J = 1$, the reconstructed velocity field with $J = 20$, and the difference between the ground true velocity field and the reconstructed velocity field with $J = 20$ (first column - third column). From top to bottom are the results from the noise-free wave signal, the wave signal with 10% multiplication Gaussian noise, and the wave signal with 10% additive Gaussian noise.

Inversion for the velocity model (4.26) **with** $M = 4$

For the second inversion example, we work on reconstructing the features of the following velocity model

$$m(\mathbf{x}) = \sum_{k_x,k_z=0}^{4} \mathfrak{m}(\mathbf{k}) \cos(k_x \pi x) \cos(k_z \pi z), \quad \mathbf{k} = (k_x, k_z) \tag{4.34}$$

with $\beta = 0$ in (4.28), namely, we don't consider any decay on the coefficients for this example. In addition, we use the same training settings as those in Section 4.7.2 for the Fourier wave signal generation. But for the external top sources $h_i(x)$, we choose them to be the same as the sources in Section 4.7.4 to generate resourceful training samples for the construction of $\widehat{\mathbf{f}}_{\widehat{\alpha}}^{-1}$.

For the online inversion stage, we also implement a J-term truncated Neumann series approximation (4.25) to recover the velocity model. To test the stability of the proposed coupled scheme,

	no noise			10% multiplicative noise			10% additive noise		
J	L^2 error	L^∞ error	CPU time(s)	L^2 error	L^∞ error	CPU time(s)	L^2 error	L^∞ error	CPU time(s)
1	1.48e-01	1.12e-00	0	2.54e-01	1.98e-00	0	2.60e-01	1.60e-00	0
20	1.10e-04	1.11e-03	6.83	1.71e-03	1.39e-02	6.71	5.79e-03	4.27e-02	6.89
40	3.21e-06	2.60e-05	13.80	4.84e-05	5.48e-04	13.60	5.88e-03	4.28e-02	13.38
60	3.09e-06	2.41e-05	20.34	5.78e-06	4.38e-05	20.97	5.89e-03	4.28e-02	20.51
80	2.86e-06	2.66e-05	27.74	4.63e-06	4.36e-05	28.52	5.89e-03	4.28e-02	27.56

Table 4.2: L^2/L^∞ reconstruction errors, and the CPU time for the inversion stage with different J-term truncated Neumann series approximation, as well as different noise level/form for the reconstruction of the mixed Gaussian (4.33).

	no noise			10% multiplicative noise			10% additive noise		
J	L^2 error	L^∞ error	CPU time(s)	L^2 error	L^∞ error	CPU time(s)	L^2 error	L^∞ error	CPU time(s)
1	1.78e-01	8.46e-01	0	1.79e-00	7.25e-00	0	2.64e-01	1.16e-00	0
20	8.52e-04	4.90e-03	6.08	1.23e-02	6.97e-02	5.81	7.13e-03	3.70e-02	5.87
40	1.49e-05	8.41e-05	11.15	4.93e-03	2.76e-02	11.24	1.69e-03	9.26e-03	11.73
60	2.60e-07	1.53e-06	17.97	3.06e-03	1.71e-02	17.42	6.81e-04	3.77e-03	17.67
80	2.16e-08	1.34e-07	22.74	2.19e-03	1.22e-02	23.25	2.05e-04	1.12e-03	23.13

Table 4.3: L^2/L^∞ reconstruction errors, and the CPU time for the inversion stage with different J-term truncated Neumann series approximation, as well as different noise level/form for the reconstruction of the Fourier model (4.34).

as in Section 4.7.4, we add the Gaussian noise with zeros mean and 10% standard derivation to the wave signals. Figure 4.13 presents the surface plots of the reconstructed velocity images with $J = 20$, as well as the surface plots for the difference between the reconstructed image and the ground true velocity model. The layout of Figure 4.13 is the same as the one in Figure 4.12. We observe that the training prediction is stable with respect to the noise, see the second column of Figure 4.13 and L^2/L^∞ errors when $J = 1$ in Table 4.3. In addition, we note that the inversion stage can significantly improve the accuracy of the reconstruction. For the data without noise, the errors dropped by a factor $\sim 10^7$ within 30 seconds; even for the data with 10% Gaussian noise, the errors also dropped by a factor $\sim 10^3$ within 30 seconds.

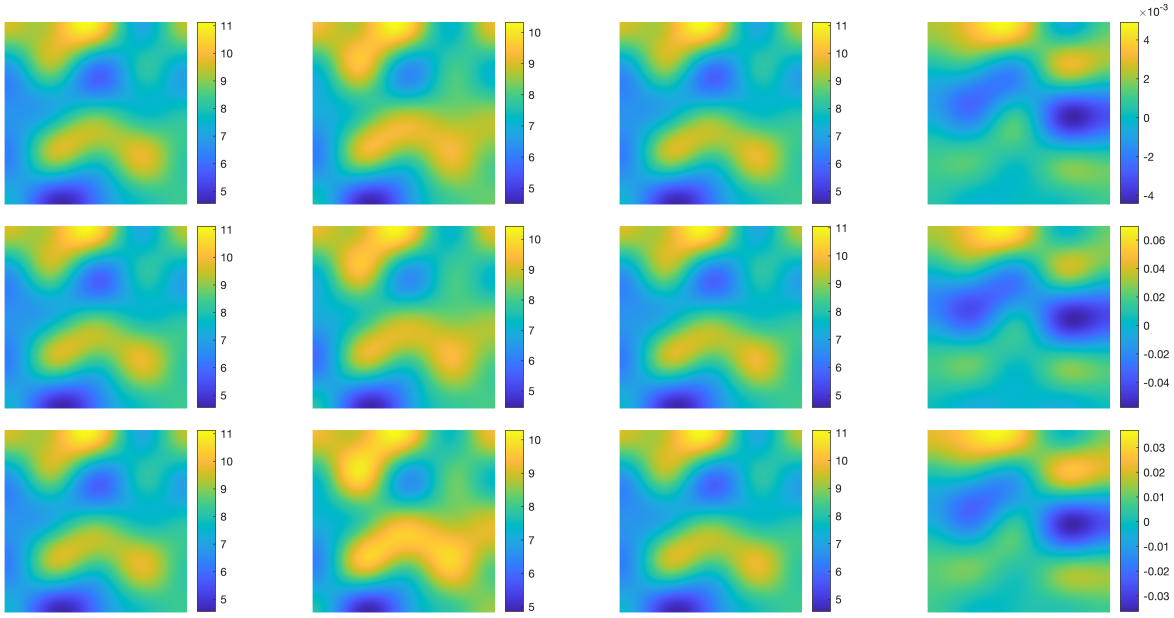


Figure 4.13: The reconstructed velocity images for the general Fourier type (4.26) with $M = 4$. From top to bottom are for the velocity reconstruction without noise, with 10% multiplication Gaussian noise, with 10% additive Gaussian noise, respectively. While from left to right are the ground true velocity field, the reconstructed velocity field from the neural network in the offline training stage, and the reconstructed velocity image with $J = 20$, error for the reconstructed velocity image with $J = 20$, respectively.

Inversion for the velocity model (4.26) **with** $M = 7$

For the third example, we consider a velocity model which contains 8 Fourier modes in each direction, namely,

$$m(x,z) = \sum_{k_x,k_z=0}^{7} \mathfrak{m}(\mathbf{k}) \cos(k_x \pi x) \cos(k_z \pi z). \tag{4.35}$$

The spatial and temporal discretization, as well as the rules for data generation, the choice of the top source $h_i(x)$ are the same as the example in Section 4.7.4.

For the inversion stage, we again implement a J-term truncated Neumann series approximation (4.25) to obtain the reconstructed velocity image. Figure 4.14 presents the surface plots of the reconstructed velocity images with various values of J in the online inversion stage. Precisely, each row of Figure 4.14 corresponds to one velocity model; from left to right are the ground true velocity field, the reconstructed velocity image with $J = 1$, the reconstructed velocity image with $J = 20$, and the reconstructed velocity image with $J = 50$, respectively. We note that the online inversion

stage improves the accuracy of the reconstruction for all cases, which verifies the effectiveness of the proposed coupling scheme.

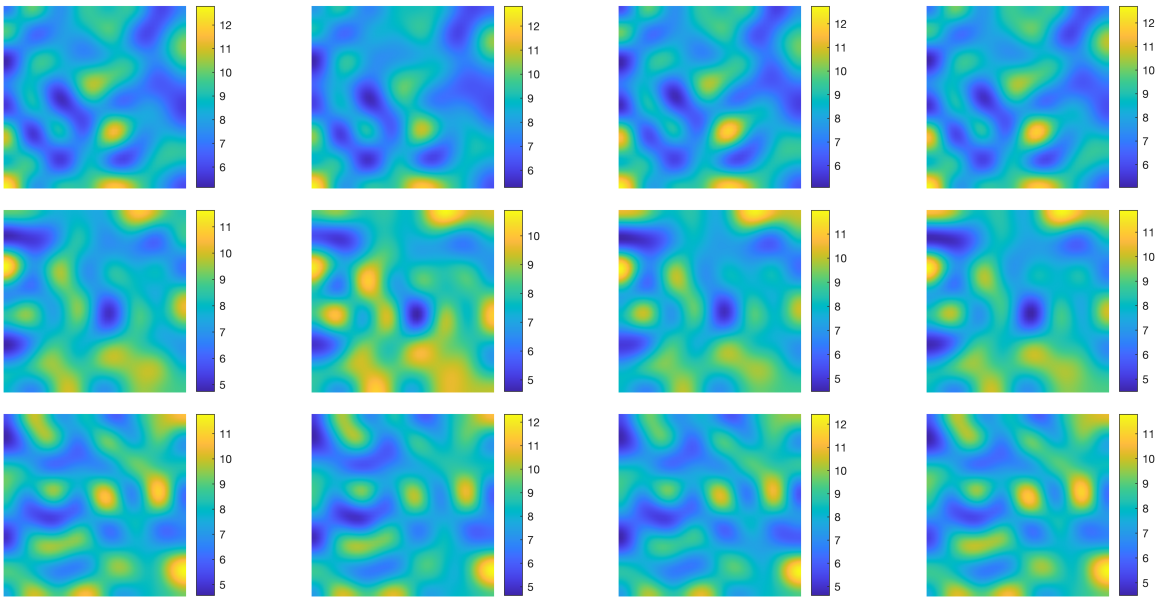


Figure 4.14: The reconstructed velocity images for the Fourier model (4.35). Each row corresponds to the reconstruction of one velocity field. From left to right are the ground true velocity field, the reconstructed velocity field with $J = 1$, the reconstructed velocity field with $J = 20$, and the reconstructed velocity field with $J = 50$, respectively.

Inversion for the velocity model outside of the training domain

We test the proposed coupling scheme on a velocity model outside the training domain in the last example. Precisely, the design of the offline training stage is the same as the one in Section 4.7.4, namely, we focus on learning the first 5 Fourier modes in each direction during the training. However, our goal in this example is to reconstruct the following velocity model,

$$m(x,z) = \begin{cases} 8.4, & (x,z) \in [0.22, 0.74] \times [-0.52, -0.5], \\ 7.6, & \text{otherwise}, \end{cases}, \quad (x,z) \in [0,1] \times [-1,0], \tag{4.36}$$

which is apparently outside of the training domain containing many high-frequency components. To reconstruct (4.36), we first implement the J-term truncated Neumann series approximation (4.11) with $J = 20$ to obtain the low frequency part of the velocity model (4.36), then use it as the initial

guess of a quasi-Newton algorithm based on the BFGS gradient update rule to minimize (4.15). In addition, to recover the high-frequency components of the velocity field, except the 51 receivers at the bottom surface in the training stage, we place another 51 receivers at the top surface when minimizing (4.15), and enforce 7 different top sources $h_i(x), i = 1, 2 \cdots, 7$ with h_1, h_2, h_3 being the same as the top sources in the training stage, and

$$h_4(x) = e^{-\frac{(x-0.7)^2}{0.01}} - e^{-\frac{(x-0.2)^2}{0.01}}, \quad h_5(x) = e^{-\frac{(x-0.3)^2}{0.01}} - e^{-\frac{(x-0.9)^2}{0.01}},$$

$$h_6(x) = e^{-\frac{(x-0.2)^2}{0.01}} - e^{-\frac{(x-0.5)^2}{0.01}}, \quad h_7(x) = e^{-\frac{(x-0.1)^2}{0.01}} - e^{-\frac{(x-0.6)^2}{0.01}}.$$

Figure 4.15 presents the surface plots of the reconstructed velocity images with both noise-free data and the data with Gaussian noises. Precisely, the top row shows the reconstructed velocity from noise-free data, the middle row displays the reconstructed velocity from the data with 10% multiplication Gaussian noise, and the bottom row presents the reconstructed velocity from the data with 10% additive Gaussian noise; while from the left to the right columns are the ground true velocity field, the reconstructed velocity image with $J = 1$, the reconstructed velocity image with $J = 20$ (initial guess), and the reconstructed velocity image by minimizing (4.15), respectively. We note that adding several terms to the Neumann series approximation can lead to a relatively good reconstruction for the low-frequency components of the velocity field (4.36) for all cases (noise-free data and the data with Gaussian noise) by comparing the reconstruction results in column 2 and column 3. Then solving an extra classical minimization problem as documented in Section 4.5.3 helps grab the high-frequency components of the velocity field as shown in the last column.

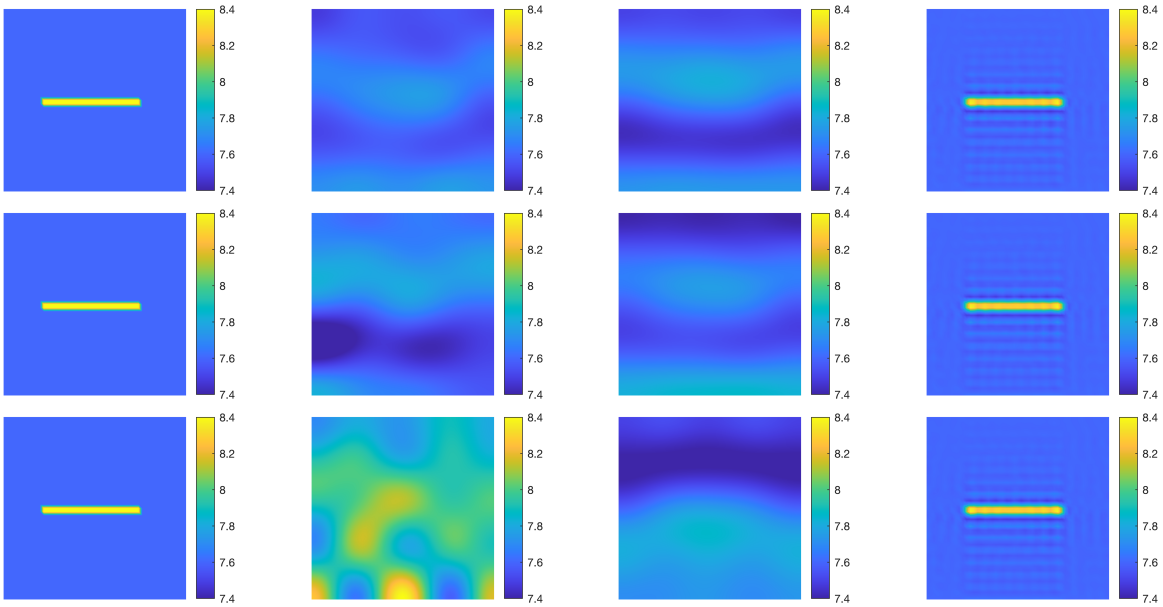


Figure 4.15: The reconstructed velocity images for the velocity model (4.36). From top to bottom are the reconstruction with noise-free signal, the signal with 10% multiplication Gaussian noise, and the signal with 10% additive Gaussian noise, respectively. From left to right are the ground true velocity field, the reconstructed velocity field with $J = 1$, the reconstructed velocity field with $J = 20$, and the reconstructed velocity field by minimizing (4.15), respectively.

Chapter 5: Concluding Remarks

In this thesis, we analyze different loss functions for solving inverse problems and demonstrate via numerical simulations that improvement can be achieved by modifying some traditional loss functions.

In the third chapter, we performed a systematic computational study on the performance of different Wasserstein metrics as loss functions in the computational solution of linear and nonlinear inverse problems. We highlighted the similarities and differences between the inverse solutions from those metrics and compared them to inversion results from the classical L^2 least-squares formulation. While it is true that in general inversions based on the Wasserstein metrics are computationally more expensive, they do offer some advantages. These inversions are much more robust to the random noise, especially high-frequency noise, in the data used for the inversions. This is true for the Wasserstein metrics in both balanced and unbalanced optimal transport theory. Secondly, these inversions are more robust to the initial guess we choose to start the iterative inversion algorithms.

In the fourth chapter, we presented an approach to design loss functions from an offline-online computational strategy for coupling deep learning methods with classical model-based iterative reconstruction schemes for the FWI problem. The main advantage of the coupling lies in two aspects. First, the coupling requires much less rigorous training for the learning part than a purely learning based approach. This makes learning the approximate inverse operator much more realistic with limited computational resources. Second, offline learning can still significantly reduce the online reconstruction with new datasets when used as a nonlinear pre-conditioner. The numerical simulations we performed demonstrated the feasibility of such a coupled approach. Moreover, the loss landscape under neural network-induced loss function is indeed more convex compared to the classical L^2 loss landscape. There are many essential issues in the current direction that need to be

more rigorously investigated. One particular aspect is to develop a mathematical characterization of the training error in the learning process and study its impact on the reconstruction step. A second aspect is to improve the learning algorithm to learn more features in the inverse operator. As we reasoned in this chapter, it is incredibly challenging to learn all the details in the inverse operator. However, we believe one could do much better than the numerical experiments in this chapter, where we pursue only a very small number of features in the learning process. Searching for better feature models for the velocity field and the time traces of the wave field is also an essential task with the potential to significantly improve the performance of the learning procedure.

This thesis only touches a small portion of what we believe to be an important area of computational inversion and learning: to construct loss functions to improve the computational efficiency as well as computational results of the inversion and learning process. We mainly focus on the formulation of ideas and computational demonstration of the effectiveness of the ideas. We have only limited mathematical understanding on them. Our future goal is to develop more systematical mathematical characterizations on the properties of the loss functions we proposed in this thesis.

References

[1] A. Adler, M. Araya-Polo, and T. Poggio, "Deep learning for seismic inverse problems: Toward the acceleration of geophysical analysis workflows," *IEEE Signal Processing Magazine*, vol. 38, pp. 89–119, 2021.

[2] J. Adler and O. Öktem, "Solving ill-posed inverse problems using iterative deep neural networks," *Inverse Problems*, vol. 33, 2017, 124007.

[3] J. Adler, A. Ringh, O. Öktem, and J. Karlsson, "Learning to solve inverse problems using wasserstein loss," *arXiv preprint arXiv:1710.10898*, 2017.

[4] V. Akçelik, G. Biros, and O. Ghattas, "Parallel multiscale Gauss-Newton-Krylov methods for inverse wave propagation," in *Proceedings of the 2002 ACM/IEEE Conference on Supercomputing*, 2002, pp. 1–15.

[5] H. Antil, D. P. Kouri, M.-D. Lacasse, and D. Ridzal, Eds., *Frontiers in PDE-Constrained Optimization*. Springer, 2018.

[6] M. Araya-Polo, A. Adler, S. Farris, and J. Jennings, "Fast and accurate seismic tomography via deep learning," in *Deep Learning: Algorithms and Applications*, Springer, 2020, pp. 129–156.

[7] E. Bachmann and J. Tromp, "Source encoding for viscoacoustic ultrasound computed tomography," *J. Acoust. Soc. Am.*, vol. 147, pp. 3221–3235, 2020.

[8] G. Bal, K. Ren, G. Uhlmann, and T. Zhou, "Quantitative thermo-acoustics and related problems," *Inverse Problems*, vol. 27, 2011, 055007.

[9] G. Bao and W. W. Symes, "On the sensitivity of hyperbolic equation to the coefficient," *Comm. in P.D.E.*, vol. 21, pp. 395–422, 1996.

[10] G. Bao, X. Ye, Y. Zang, and H. Zhou, "Numerical solution of inverse problems by weak adversarial networks," *Inverse Problems*, vol. 36, no. 11, 2020, 115003.

[11] A. G. Baydin, B. A. Pearlmutter, A. A. Radul, and J. M. Siskind, "Automatic differentiation in machine learning: A survey," *Journal of Marchine Learning Research*, vol. 18, pp. 1–43, 2018.

[12] R. Beerends, "An introduction to the abel transform," in *Miniconference on Harmonic Analysis*, vol. 15, Australian National University, Mathematical Sciences Institute, 1987, pp. 21–34.

[13] J.-D. Benamou, "Numerical resolution of an "unbalanced" mass transport problem," *ESAIM: Math. Model. Numer. Anal.*, vol. 37, pp. 851–868, 2003.

[14] J. D. Benamou and Y. Brenier, "A computational fluid mechanics solution to the Monge-Kantorovich mass transfer problem," *Numer. Math.*, vol. 84, pp. 375–393, 2000.

[15] A. Berlinet and C. Thomas-Agnan, "Reproducing kernel hilbert spaces in probability and statistics," 2004.

[16] S. Bernard, V. Monteiller, D. Komatitsch, and P. Lasaygues, "Ultrasonic computed tomography based on full-waveform inversion for bone quantitative imaging," *Phys. Med. Bio.*, vol. 62, pp. 7011–7035, 2017.

[17] L. Borcea, V. Druskin, A. Mamonov, and M. Zaslavsky, "Untangling the nonlinearity in inverse scattering with data-driven reduced order models," *Inverse Problems*, vol. 34, 2018, 065008.

[18] S. Boyd and L. Vandenberghe, *Convex optimization*. Cambridge university press, 2004.

[19] R. Brossier, S. Operto, and J. Virieux, "Which data residual norm for robust elastic frequency-domain full waveform inversion," *Geophysics*, vol. 75, R37–R46, 2010.

[20] T. A. Bubba, M. Galinier, M. Lassas, M. Prato, L. Ratti, and S. Siltanen, "Deep neural networks for inverse problems with pseudodifferential operators: An application to limited-angle tomography," *SIAM J. Imaging Sci.*, vol. 14, pp. 470–505, 2021.

[21] C. Bunks, F. M. Saleck, S. Zaleski, and G. Chavent, "Multiscale seismic waveform inversion," *Geophysics*, vol. 50, pp. 1457–1473, 1995.

[22] C. Burstedde and O. Ghattas, "Algorithmic strategies for full waveform inversion: 1D experiments," *Geophysics*, vol. 74, pp. 37–46, 2009.

[23] R. Chartrand, B. Wohlberg, K. Vixie, and E. Bollt, "A gradient descent solution to the monge-kantorovich problem," *Applied Mathematical Sciences*, vol. 3, no. 22, pp. 1071–1080, 2009.

[24] J. Chen, Y. Chen, H. Wu, and D. Yang, "The quadratic Wasserstein metric for earthquake location," *J. Comput. Phys.*, vol. 373, pp. 188–209, 2018.

[25] K. Chen, Z. Ying, H. Zhang, and L. Zhao, "Analysis of least absolute deviation," *Biometrika*, vol. 95, no. 1, pp. 107–122, 2008.

[26] K. Chen and M. D. Sacchi, "Time-domain elastic Gauss-Newton full-waveform inversion: A matrix-free approach," *Geophys. J. Int.*, vol. 223, pp. 1007–1039, 2020.

[27] L. Chizat, G. Peyré, B. Schmitzer, and F. X. Vialard, "An interpolating distance between optimal transport and Fisher-Rao metrics," *Found. Comput. Math.*, vol. 18, pp. 1–44, 2016.

[28] ——, "Unbalanced optimal transport: Dynamic and Kantorovich formulations," *J. Funct. Anal.*, vol. 274, pp. 3090–3123, 2018.

[29] Y. Cooper, "The loss landscape of overparameterized neural networks," *arXiv preprint arXiv:1804.10200*, 2018.

[30] G. Côrte, J. Dramsch, H. Amini, and C. MacBeth, "Deep neural network application for 4D seismic inversion to changes in pressure and saturation: Optimizing the use of synthetic training datasets," *Geophysical Prospecting*, vol. 68, pp. 2164–2185, 2020.

[31] G. Cybenko, "Approximation by superpositions of a sigmoidal function," *Mathematics of Control, Signals, and Systems (MCSS)*, vol. 2, no. 4, pp. 303–314, Dec. 1989.

[32] T. Dierkes, O. Dorn, F. Natterer, V. Palamodov, and H. Sieschott, "Fréchet derivatives for some bilinear inverse problems," *SIAM J. Appl. Math.*, vol. 62, pp. 2092–2113, 2002.

[33] W. Ding, Q. Du, and K. Ren, "Computational inverse problems with quadratic wasserstein metrics," *Preprint*, 2022.

[34] W. Ding, K. Ren, and L. Zhang, "Coupling deep learning with full waveform inversion," *preprint*, 2022, arXiv.

[35] W. Ding, K. Ren, and L. Zhang, "Coupling deep learning with full waveform inversion," *arXiv preprint arXiv:2203.01799*, 2022.

[36] B. Engquist and B. D. Froese, "Application of the Wasserstein metric to seismic signals," *Commun. Math. Sci.*, vol. 12, pp. 979–988, 2014.

[37] B. Engquist, B. D. Froese, and Y. Yang, "Optimal transport for seismic full waveform inversion," *Commun. Math. Sci.*, vol. 14, pp. 2309–2330. 2016.

[38] B. Engquist, K. Ren, and Y. Yang, "The quadratic Wasserstein metric for inverse data matching," *Inverse Problems*, vol. 36, p. 055 001, 2020, arXiv:1911.06911.

[39] ——, "A generalized weighted optimization method for computational learning and inversion," *ICLR 2022*, 2022, arXiv:2201.09223.

[40] B. Engquist, K. Ren, and Y. Yang, "The quadratic wasserstein metric for inverse data matching," *Inverse Problems*, vol. 36, no. 5, p. 055 001, 2020.

[41] I. Epanomeritakis, V. Akcelik, O. Ghattas, and J. Bielak, "A Newton-CG method for large-scale three-dimensional elastic full-waveform seismic inversion," *Inverse Problems*, vol. 24, 2008, 034015.

[42] L. C. Evans, *Partial Differential Equations*. Providence, RI: American Mathematical Society, 2010.

[43] L. C. Evans, *Partial differential equations*. American mathematical society, 2022, vol. 19.

[44] J. Fang *et al.*, "Data-driven low-frequency signal recovery using deep-learning predictions in full-waveform inversion," *Geophysics*, vol. 85, A37–A43, 2020.

[45] A. C. Fannjiang, T. Strohmer, and P. Yan, "Compressed remote sensing of sparse objects," *SIAM J. Imag. Sci.*, vol. 3, pp. 595–618, 2010.

[46] S. Farris, M. Araya-Polo, J. Jennings, B. Clapp, and B. Biondi, "Tomography: A deep learning vs full-waveform inversion comparison," in *Proceedings, First EAGE Workshop on High Performance Computing for Upstream in Latin America*, vol. 2018, 2018, pp. 1–5.

[47] J. Feliu-Fabà, Y. Fan, and L. Ying, "Meta-learning pseudo-differential operators with deep neural networks," *J. Comput. Phys.*, vol. 404, 2020, 109309.

[48] A. Fichtner, *Full Seismic Waveform Modelling and Inversion*. Berlin: Springer-Verlag, 2011.

[49] R. Fletcher, *Practical methods of optimization*. John Wiley & Sons, 2013.

[50] S. Fort, H. Hu, and B. Lakshminarayanan, "Deep ensembles: A loss landscape perspective," *arXiv preprint arXiv:1912.02757*, 2019.

[51] S. Fort and S. Jastrzebski, "Large scale structure of neural network loss landscapes," *Advances in Neural Information Processing Systems*, vol. 32, 2019.

[52] S. Foucart and H. Rauhut, *A Mathematical Introduction to Compressive Sensing*. Birkhäuser, 2013.

[53] W. Gangbo, W. Li, S. Osher, and M. Puthawala, "Unnormalized optimal transport," *arXiv:1902.03367v1*, 2019.

[54] H. Gao, H. Yu, S. Osher, and G. Wang, "Multi-energy CT based on a prior rank, intensity and sparsity model (PRISM)," *Inverse Problems*, vol. 27, 2011, 115012.

[55] T. Garipov, P. Izmailov, D. Podoprikhin, D. P. Vetrov, and A. G. Wilson, "Loss surfaces, mode connectivity, and fast ensembling of dnns," *Advances in neural information processing systems*, vol. 31, 2018.

[56] B. Ghorbani, Y. Xiao, and S. Krishnan, "The effect of network depth on the optimization landscape," 2019.

[57] M. Goldman, M. Huesmann, and F. Otto, "Quantitative linearization results for the monge-ampère equation," *Communications on Pure and Applied Mathematics*, vol. 74, no. 12, pp. 2483–2560, 2021.

[58] A. Gramfort, G. Peyré, and M. Cuturi, "Fast optimal transport averaging of neuroimaging data," in *International Conference on Information Processing in Medical Imaging*, Springer, 2015, pp. 261–272.

[59] P. Greengard, J. G. Hoskins, N. F. Marshall, and A. Singer, "On a linearization of quadratic wasserstein distance," *arXiv preprint arXiv:2201.13386*, 2022.

[60] L. Guasch, O. Calderón Agudo, M. X. Tang, P. Nachev, and M. Warner, "Full-waveform inversion imaging of the human brain," *Digit. Med.*, vol. 3, 2000, 28.

[61] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2016.

[62] H. Heaton, S. W. Fung, A. T. Lin, S. Osher, and W. Yin, "Wasserstein-based projections with applications to inverse problems," *SIAM Journal on Mathematics of Data Science*, vol. 4, no. 2, pp. 581–603, 2022.

[63] K. Hornik, M. Stinchcombe, and H. White, "Multilayer feedforward networks are universal approximators," *Neural Networks*, vol. 2, no. 5, pp. 359–366, 1989.

[64] W. Hu, Y. Jin, X. Wu, and J. Chen, "Progressive transfer learning for low-frequency data prediction in full-waveform inversion," *Geophysics*, vol. 86, R369–R382, 2021.

[65] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *Proceedings of the 32nd International Conference on Machine Learning*, 2015.

[66] V. Isakov, *Inverse Problems for Partial Differential Equations*, second. New York: Springer-Verlag, 2006.

[67] A. Javaherian, F. Lucka, and B. T. Cox, "Refraction-corrected ray-based inversion for three-dimensional ultrasound tomography of the breast," *Inverse Problems*, vol. 36, 2020, 125010.

[68] Z. Jia, R. Guo, M. Li, G. Wang, Z. Liu, and Y. Shao, "3-d model-based inversion using supervised descent method for aspect-limited microwave data of metallic targets," *IEEE Trans. Geosci. Remote Sens.*, pp. 1–10, 2021.

[69] S. Kamyab, Z. Azimifar, R. Sabzi, and P. Fieguth, "Deep learning methods for inverse problems," *PeerJ Computer Science*, vol. 8, e951, 2022.

[70] V. Kazei, O. Ovcharenko, P. Plotnitskii, D. Peter, X. Zhang, and T. Alkhalifah, "Mapping full seismic waveforms to vertical velocity profiles by deep learning," *Geophysics*, vol. 86, pp. 1–50, 2021.

[71] D. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv:1412.6980*, 2014.

[72] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[73] S Kolouri, S. R. Park, M. Thorpe, D. Slepcev, and G. K. Rohde, "Optimal mass transport: Signal processing and machine-learning applications," *IEEE Signal Processing Magazine*, vol. 34, pp. 43–59, 2017.

[74] S. Kolouri, G. K. Rohde, and H. Hoffmann, "Sliced wasserstein distance for learning gaussian mixture models," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 3427–3436.

[75] S. Kondratyev, L. Monsaingeon, and D. Vorotnikov, "A new optimal transport distance on the space of finite Radon measures," *Adv. Differential Equations*, vol. 21, pp. 1117–1164, 2016.

[76] Y. Lecun, P. Haffner, L. Bottou, and Y. Bengio, "Object recognition with gradient-based learning," in *Feature grouping*, D. Forsyth, Ed. Springer, 1999.

[77] W. Lee, R. Lai, W. Li, and S. Osher, "Generalized Unnormalized Optimal Transport and its fast algorithms," *Journal of Computational Physics*, vol. 436, p. 110 041, Jul. 2021, arXiv: 2001.11530.

[78] ——, "Generalized unnormalized optimal transport and its fast algorithms," vol. 436, 2021, 110041.

[79] J. Lellmann, D. Lorenz, C. Schönlieb, and T. Valkonen, "Imaging with Kantorovich-Rubinstein discrepancy," *SIAM J. Imag. Sci.*, vol. 7, pp. 2833–2859, 2014.

[80] G. Leugering *et al.*, Eds., *Trends in PDE-Constrained Optimization*. Springer, 2014.

[81] F. Li, U. Villa, S. Park, and M. A. Anastasio, "Three-dimensional stochastic numerical breast phantoms for enabling virtual imaging trials of ultrasound computed tomography," *arXiv:2106.02744*, 2021.

[82] H. Li, J. Schwab, S. Antholzer, and M. Haltmeier, "NETT: Solving inverse problems with deep neural networks," *Inverse Problems*, 2020.

[83] H. Li, Z. Xu, G. Taylor, C. Studer, and T. Goldstein, "Visualizing the loss landscape of neural nets," *Advances in neural information processing systems*, vol. 31, 2018.

[84] L. Li, A. Vidard, F.-X. Le Dimet, and J. Ma, "Topological data assimilation using wasserstein distance," *Inverse Problems*, vol. 35, no. 1, p. 015 006, 2018.

[85] Z. Li, Y. Tang, J. Chen, and H. Wu, "The quadratic wasserstein metric with squaring scaling for seismic velocity inversion," *arXiv preprint arXiv:2201.11305*, 2022.

[86] M. Liero, A. Mielke, and G. Savaré, "Optimal entropy-transport problems and a new Hellinger-Kantorovich distance between positive measures," *Inventiones Mathematicae*, vol. 211, pp. 969–1117, 2018.

[87] M. Lin, Q. Chen, and S. Yan, *Network in network*, 2014. arXiv: 1312.4400 [cs.NE].

[88] Z. Lin *et al.*, "Low-frequency data prediction with iterative learning for highly nonlinear inverse scattering problems," *IEEE Trans. Microw. Theory Tech.*, 2021.

[89] B. Liu, S. Yang, Y. Ren, X. Xu, P. Jiang, and Y. Chen, "Deep-learning seismic full-waveform inversion for realistic structural models," *Geophysics*, vol. 86, R31–R44, 2021.

[90] B. Liu, "Understanding the loss landscape of one-hidden-layer relu networks," *Knowledge-Based Systems*, vol. 220, p. 106 923, 2021.

[91] Q. Liu, S. Beller, W. Lei, D. Peter, and J. Tromp, "Preconditioned BFGS-based uncertainty quantification in elastic full waveform inversion," *arXiv:2009.12663*, 2020.

[92] D. Lombardi and E.Maitre, "Eulerian models and algorithms for unbalanced optimal transport," *ESAIM: Math. Model. Numer. Anal.*, vol. 49, pp. 1717–1744, 2015.

[93] F. Lucka, M. Pérez-Liva, B. E. Treeby, and B. T. Cox, "High resolution 3D ultrasonic breast imaging by time-domain full waveform inversion," *arXiv:2102.00755*, 2021.

[94] S. Mache, P. K. Pokala, K. Rajendran, and C. S. Seelamantula, "DuRIN: A deep-unfolded sparse seismic reflectivity inversion network," *arXiv:2104.04704*, 2021.

[95] S. Mahankali, "Velocity inversion using the quadratic wasserstein metric," *arXiv e-prints*, arXiv–2009, 2020.

[96] T. P. Matthews, J. Poudel, L. Li, L. V. Wang, and M. A. Anastasio, "Parameterized joint reconstruction of the initial pressure and sound speed distributions for photoacoustic computed tomography," *SIAM J. Imaging Sci.*, vol. 11, pp. 1560–1588, 2018.

[97] M. T. McCann, K. H. Jin, and M. Unser, "A review of convolutional neural networks for inverse problems in imaging," *arXiv preprint arXiv:1710.04011*, 2017.

[98] Q. Mérigot, A. Delalande, and F. Chazal, "Quantitative stability of optimal transport maps and linearization of the 2-wasserstein space," in *International Conference on Artificial Intelligence and Statistics*, PMLR, 2020, pp. 3186–3196.

[99] L. Métivier, A. Allain, R. Brossier, Q. Mérigot, E. Oudet, and J. Virieux, "Optimal transport for mitigating cycle skipping in full waveform inversion: A graph space transform approach," *Geophysics*, vol. 83, R515–R540, 2018.

[100] L. Métivier, R. Brossier, Q. Mérigot, E. Oudet, and J. Virieux, "An optimal transport approach for seismic tomography: Application to 3D full waveform inversion," *Inverse Probl.*, vol. 32, 2016, 115008.

[101] ——, "Measuring the misfit between seismograms using an optimal transport distance: Application to full waveform inversion," *Geophys. J. Int.*, vol. 205, pp. 345–377, 2016.

[102] D. Misra, *Mish: A self regularized non-monotonic activation function*, 2020. arXiv: 1908.08681 [cs.LG].

[103] R. Modrak and J. Tromp, "Seismic waveform inversion best practices: Regional, global and exploration test cases," *Geophys. J. Int.*, vol. 206, pp. 1864–1889, 2016.

[104] M. Motamed and D. Appelo, "Wasserstein metric-driven bayesian inversion with applications to signal processing," *International Journal for Uncertainty Quantification*, vol. 9, no. 4, 2019.

[105] J. Nocedal and S. J. Wright, *Numerical Optimization*. New York: Springer-Verlag, 2006.

[106] G. Ongie, A. Jalal, C. A. Metzler, R. G. Baraniuk, A. G. Dimakis, and R. Willett, "Deep learning techniques for inverse problems in imaging," *IEEE Journal on Selected Areas in Information Theory*, vol. 1, no. 1, pp. 39–56, 2020.

[107] R. Peyre, "Comparison between w2 distance and h-1 norm, and localization of wasserstein distance," *ESAIM: Control, Optimisation and Calculus of Variations*, vol. 24, no. 4, pp. 1489–1501, 2018.

[108] B. Piccoli and F. Rossi, "Generalized wasserstein distance and its application to transport equations with source," *Arch. Rational Mech. Anal.*, vol. 211, pp. 335–358, 2014.

[109] R.-E. Plessix, "A review of the adjoint-state method for computing the gradient of a functional with geophysical applications," *Geophys. J. Int.*, vol. 167, pp. 495–503, 2006.

[110] R. G. Pratt, "Seismic waveform inversion in the frequency domain, Part 1: Theory and verification in a physical scale model," *Geophysics*, vol. 64, pp. 888–901, 1999.

[111] R. G. Pratt, C. Shin, and G. J. Hicks, "Gauss-Newton and full Newton methods in frequency-space seismic waveform inversion," *Geophys. J. Int.*, vol. 133, pp. 341–362, 1998.

[112] N. Rahaman *et al.*, "On the spectral bias of neural networks," in *Proceedings of the 36th International Conference on Machine Learning*, vol. 97, 2019, pp. 5301–5310.

[113] M. Raissi, P. Perdikaris, and G. E. Karniadakis, "Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations," *J. Comput. Phys.*, vol. 378, pp. 686–707, 2019.

[114] A. Rangamani *et al.*, "Loss landscapes and generalization in neural networks: Theory and applications," Ph.D. dissertation, Johns Hopkins University, 2020.

[115] Y. Ren, L. Nie, S. Yang, P. Jiang, and Y. Chen, "Building complex seismic velocity models for deep learning inversion," *IEEE Access*, vol. 9, pp. 63 767–63 778, 2021.

[116] J. A. Rivera, D. Pardo, and E. Alberdi, "Design of loss functions for solving inverse problems using deep learning," in *International Conference on Computational Science*, Springer, 2020, pp. 158–171.

[117] R. Rojas-Gomez, J. Yang, Y. Lin, J. Theiler, and B. Wohlberg, "Physics-consistent data-driven waveform inversion with adaptive data augmentation," *IEEE Geosci. Remote. Sens. Lett.*, pp. 1–5, 2020.

[118] B. Ronen, D. Jacobs, Y. Kasten, and S. Kritchman, "The convergence rate of neural networks for learned functions of different frequencies," in *Advances in Neural Information Processing Systems*, vol. 32, 2019.

[119] F. Santambrogio, *Optimal Transport for Applied Mathematicians: Calculus of Variations, PDEs, and Modeling*. Birkhäuser, 2015.

[120] F. Santosa and W. W. Symes, *Analysis of Least-squares velocity inversion*. Society of Exploration Geophysicists, 1989.

[121] B. Schmitzer, "A sparse multiscale algorithm for dense optimal transport," *J. Math. Imag. Vision*, vol. 56, pp. 238–259, 2016.

[122] L. Sirgue and R. G. Pratt, "Efficient waveform inversion and imaging: A strategy for selecting temporal frequencies," *Geophysics*, vol. 69, pp. 231–248, 2004.

[123] I. Skorokhodov and M. Burtsev, "Loss landscape sightseeing with multi-point optimization," *arXiv preprint arXiv:1910.03867*, 2019.

[124] J. D. Smith, K. Azizzadenesheli, and Z. E. Ross, "Eikonet: Solving the eikonal equation with deep neural networks," *arXiv:2004.00361*, 2020.

[125] J. Solomon, R. M. Rustamov, L. Guibas, and A. Butscher, "Wasserstein propagation for semi-supervised learning," in *Proceedings of the 31st International Conference on Machine Learning*, E. P. Xing and T. Jebara, Eds., 2014, pp. 306–314.

[126] J. Solomon *et al.*, "Convolutional Wasserstein distances," *ACM Trans. Graphics*, vol. 34, pp. 1–11, 2015.

[127] C. Song and T. Alkhalifah, "Wavefield reconstruction inversion via physics-informed neural networks," *arXiv:2104.06897*, 2021.

[128] B. Sun and T. Alkhalifah, "ML-misfit: Learn a robust misfit function for full-waveform inversion using machine learning," *arXiv:2002.03163v2*, 2020.

[129] H. Sun and L. Demanet, "Extrapolated full-waveform inversion with deep learning," *Geophysics*, vol. 85, R275–R288, 2020.

[130] J. Sun, K. A. Innanen, and C. Huang, "Physics-guided deep learning for seismic inversion with hybrid training and uncertainty analysis," *Geophysics*, vol. 86, R303–R317, 2021.

[131] R. Sun, T. Fang, and A. Schwing, "Towards a better global loss landscape of gans," *Advances in Neural Information Processing Systems*, vol. 33, pp. 10 186–10 198, 2020.

[132] W. W. Symes, "Migration velocity analysis and waveform inversion," *Geophysical Prospecting*, vol. 56, pp. 765–790, 2008.

[133] Y. Tang, Y. Zheng, and N. Li, "Analysis of the Optimization Landscape of Linear Quadratic Gaussian (LQG) Control," p. 12,

[134] J. Tromp, C. Tape, and Q. Liu, "Seismic tomography, adjoint methods, time reversal and banana-doughnut kernels," *Geophys. J. Int.*, vol. 160, pp. 195–216, 2005.

[135] C. Villani, *Topics in Optimal Transportation*. Providence, RI.: American Mathematical Society, 2003.

[136] ——, *Optimal Transport: Old and New*. Springer Science & Business Media, 2008.

[137] J. Virieux, A. Asnaashari, R. Brossier, L. Métivier, A. Ribodetti, and W. Zhou, "An introduction to full waveform inversion," in *Encyclopedia of Exploration Geophysics*, 2014, R1–1–R1–40.

[138] J. Virieux and S. Operto, "An overview of full-waveform inversion in exploration geophysics," *Geophysics*, vol. 74, WCC1–WCC26, 2009.

[139] C. R. Vogel, *Computational Methods for Inverse Problems*, ser. Frontiers in Applied Mathematics. Philadelphia: SIAM, 2002.

[140] C. R. Vogel and M. E. Oman, "Fast, robust total variation-based reconstruction of noisy, blurred images," *IEEE transactions on image processing*, vol. 7, no. 6, pp. 813–824, 1998.

[141] J. Wiskin, B. Malik, D. Borup, N. Pirshafiey, and J. Klock, "Full wave 3D inverse scattering transmission ultrasound tomography in the presence of high contrast," *Scientific Reports*, vol. 10, 2020, 20166.

[142] D. Wu, Y. Wang, and S.-t. Xia, "Revisiting loss landscape for adversarial robustness," *arXiv preprint arXiv:2004.05884*, 2020.

[143] Y. Wu and Y. Lin, "InversionNet: An efficient and accurate data-driven full waveform inversion," *IEEE Transactions on Computational Imaging*, vol. 6, pp. 419–433, 2020.

[144] Z. J. Xu, Y. Zhang, and Y. Xiao, "Training behavior of deep neural network in frequency domain," *arXiv:1807.01251v3*, 2018.

[145] F. Yang and J. Ma, "Deep-learning inversion: A next-generation seismic velocity model building method," *Geophysics*, vol. 84, R583–R599, 2019.

[146] Y. Yang and B. Engquist, "Analysis of optimal transport and related misfit functions in full-waveform inversion," *Geophysics*, vol. 83, A7–A12, 2018.

[147] Y. Yang, B. Engquist, J. Sun, and B. D. Froese, "Application of optimal transport and the quadratic Wasserstein metric to full waveform inversion," *Geophysics*, vol. 83, R43–R62, 2018.

[148] Y. Yang, B. Engquist, J. Sun, and B. F. Hamfeldt, "Application of optimal transport and the quadratic wasserstein metric to full-waveform inversion," *Geophysics*, vol. 83, no. 1, R43–R62, 2018.

[149] S. Yu and J. Ma, "Data-driven geophysics: From dictionary learning to deep learning," *arXiv:2007.06183*, 2020.

[150] W. Zhang and J. Gao, "Deep-learning full-waveform inversion using seismic migration images," *IEEE Trans. Geosci. Remote Sens.*, pp. 1–18, 2021.

[151] Z. Zhang and Y. Lin, "Data-driven seismic waveform inversion: A study on the robustness and generalization," *IEEE Trans. Geosci. Remote Sens.*, vol. 58, pp. 6900–6913, 2020.

Appendix A: Discretization of Wasserstein Distances

In this appendix, we document the schemes we used in the discretization of the problems we discussed in Chapter 3 on computaitonal inversion with Wasserstein metrics.

A.1 Mesh Discretization in one spatial dimension.

In one-dimensional case, $\Omega = (0, L)$. We use uniform spatial and temporal grids with $N_x = 100$ and $N_t = 100$ respectively:

$$0 = x_0 < x_1 < \cdots < x_{N_x} = L, \;\; x_j = j\Delta x$$

$$0 = t_0 < t_1 < \cdots < t_{N_t} = T, \;\; t_j = j\Delta t$$

where $\Delta x = \frac{L}{N_x}$ and $\Delta t = \frac{T}{N_t}$. Let X be the density variable ρ or the source variable ζ, and Y be the $m(= \rho\omega)$ variable. We use staggered grids for X and Y. We define:

$$X^n_{j+\frac{1}{2}} = X(\frac{x_j + x_{j+1}}{2}, t_n), \qquad \text{and} \qquad Y^{n+\frac{1}{2}}_j = Y(x_j, \frac{t_n + t_{n+1}}{2}).$$

Discretize ρ and m as

$$\rho^n_{j+\frac{1}{2}} = \rho(\frac{x_j + x_{j+1}}{2}, t_n) \tag{A.1}$$

$$m^{n+\frac{1}{2}}_j = m(x_j, \frac{t_n + t_{n+1}}{2}) \tag{A.2}$$

We discretize the transport problem as

$$\begin{aligned} \frac{\rho_{j+\frac{1}{2}}^{n+1} - \rho_{j+\frac{1}{2}}^{n}}{\Delta t} + \frac{m_{j+1}^{n+\frac{1}{2}} - m_{j}^{n+\frac{1}{2}}}{\Delta x} &= \zeta_{j+\frac{1}{2}}^{n+\frac{1}{2}}, \\ \rho_{j+\frac{1}{2}}^{0} &= f_{j+\frac{1}{2}}, \\ \rho_{j+\frac{1}{2}}^{N_t} &= g_{j+\frac{1}{2}}, \\ m_{0}^{n+\frac{1}{2}} = 0 \qquad m_{N_x}^{n+\frac{1}{2}} &= 0\,, \end{aligned} \tag{A.3}$$

A.2 Discretization of the forward problems

When solving inverse problem, the initial data $\mathbf{f}$ is computed by evaluating the forward model $\mathbf{f}(\theta)$. The discretization of the forward model $\mathbf{f}$ will be discussed in this section when we specify the exact form of $\mathbf{f}$.

A.2.1 Abel Transform

The following integral transform relationship, known as the Abel transform, exists between two functions $\mathbf{f}(x)$ and $\theta(t)$ for $0 < \alpha < 1$,

$$\begin{aligned} \mathbf{f}(x) &= \int_0^x \frac{\theta(t)}{(x-t)^\alpha} dt \\ \theta(t) &= \frac{\sin(\pi\alpha)}{\pi} \left(\int_a^t \frac{d\mathbf{f}(s)}{ds} \frac{1}{(t-s)^{1-\alpha}} ds + \frac{\mathbf{f}(0)}{t^{1-\alpha}} \right) \end{aligned}$$

The measure is $\mathbf{f}$. Given $\mathbf{f}$, one would like to find θ such that $\mathbf{f} = Abel(\theta)$.

Let $f_i = \mathbf{f}(\frac{x_i + x_{i+1}}{2})$, $\theta_i = \theta(x_i)$, then

$$\mathbf{f}_j = (\Delta x)^{1-\alpha} \sum_{k=0}^{j-1} \frac{\theta_k}{(j - (k + \frac{1}{2}))^\alpha}$$

Here we take $\alpha = 0.9$

A.2.2 Helmholtz Equation

$$\Delta u + k^2(1 + n(x))u + ik\theta(x)u = q, \ in \ \Omega \tag{A.4}$$

$$u(x) = b(x), \ on \ \partial\Omega \tag{A.5}$$

The measure is $\mathbf{f}(\theta) = \Lambda(\theta(x))|u(x)|^2, \ in \ \Omega$, where $\Lambda(\theta(x)) = 1 \ or \ \theta$. The inverse problem can be formulated as given $k, n(x), q(x), b(x), \mathbf{g}$, one seeks $\theta(x)$ such that $\mathbf{g} = \Lambda(\theta(x))|u(x)|^2$

Let $u_i = u(\frac{x_i + x_{i+1}}{2}), \theta_i = \theta(x_i), n_i = n(x_i), q_i = q(x_i), b_0 = b(x_0), b_{N-1} = b(x_{N_x})$, then the discrete Helmholtz equation is

$$\frac{u_{j+1} - 2u_j + u_{j-1}}{\Delta x^2} + k^2(1 + n_j)u_j + ik\theta_j u_j = q_j \tag{A.6}$$

$$for \ j \in \{1, \cdots, N_x - 1\}$$

$$u_0 = b_0, u_{N_x - 1} = b_{N-1} \tag{A.7}$$

Notice that u_i is complex.

Simulation parameters $q(x) = xe^x + xsin(2\pi x)i - 1i, b_0 = 1 + 2i, b_{N-1} = 2 + 1i, k = 1, n = 1$

A.2.3 Diffusion Equation

$$-\nabla \cdot \gamma(x)\nabla u(x) + \theta(x)u(x) = q(x), \ in \ \Omega \tag{A.8}$$

$$u(x) = b(x), \ on \ \partial\Omega \tag{A.9}$$

The measure is $\mathbf{f}(\theta) = \Lambda(\theta(x))u(x), \ in \ \Omega$, where $\Lambda(\theta(x)) = 1 \ or \ \theta$.

Now, for simplicity, we assume $\gamma(x)$ is constant within the region Ω. Therefore the inverse problem can be formulated as given $\gamma(x) = \gamma, b(x), \mathbf{g}$, one seeks $\theta(x)$ such that $\mathbf{g} = \Lambda(\theta(x))u(x)$

Let $u_i = u(\frac{x_i+x_{i+1}}{2}), q_i = q(x_i), b_0 = b(x_0), b_{N-1} = b(x_{N_x})$, then the discrete Diffusion equation is

$$-\gamma\frac{u_{j+1} - 2u_j + u_{j-1}}{\Delta x^2} + \theta_j u_j = q_j \tag{A.10}$$

$$for\ j \in \{1, \cdots, N_x - 2\}$$

$$u_0 = b_0, u_{N_x-1} = b_{N-1} \tag{A.11}$$

Simulation parameters $q(x) = 3xe^x + 1, \gamma = 0.1, b_0 = 1, b_{N-1} = 2$

A.3 Solving Inverse Problem Using Wasserstein-Fisher-Rao Metric

A.3.1 Discretization of Wasserstein-Fisher-Rao Metric

The continuous version of the Wasserstein-Fisher-Rao problem will be

$$W^2_{2,WFR}(\rho_0, \rho_1) = \begin{cases} \inf\limits_{\rho,m,\zeta} \frac{1}{T}\int_0^T \int_\Omega \frac{m^2}{2\rho} + \frac{\zeta^2}{2\rho} dxdt \\ s.t. \rho_t + \nabla \cdot m = \zeta \\ \rho(0, x) = \rho_0(x) \\ \rho(T, x) = \rho_1(x) \\ \rho \geq 0 \\ m \cdot n = 0\ on\ \partial\Omega \end{cases} \tag{A.12}$$

When needed, the cost functional is evaluated by a trapezoidal scheme for the integration. More

precisely, the functional $\Phi_{W_{2,\mathrm{WFR}}}$ is discretized as

$$\Phi_{W_{2,\mathrm{WFR}}}(\rho, m, \zeta, \theta) = \frac{\Delta x \Delta t}{2T} \sum_{j=0}^{N_x-1} \sum_{n=0}^{N_t-1} \frac{(m_j^{n+\frac{1}{2}})^2 + (m_{j+1}^{n+\frac{1}{2}})^2}{\rho^n_{j+\frac{1}{2}} + \rho^{n+1}_{j+\frac{1}{2}}} + \frac{2(\zeta^{n+\frac{1}{2}}_{j+\frac{1}{2}})^2}{\rho^n_{j+\frac{1}{2}} + \rho^{n+1}_{j+\frac{1}{2}}} \tag{A.13}$$

Discretized problem formulation

$$\begin{cases} \min\limits_{m,\rho,\zeta} \Phi_{W_{2,\mathrm{WFR}}\Delta x,\Delta t}(\rho, m, \zeta) \\ s.t. \ \ \frac{\rho^{n+1}_{j+\frac{1}{2}} - \rho^n_{j+\frac{1}{2}}}{\Delta t} + \frac{m^{n+\frac{1}{2}}_{j+1} - m^{n+\frac{1}{2}}_{j}}{\Delta x} = \zeta^{n+\frac{1}{2}}_{j+\frac{1}{2}} \\ \rho^0_{j+\frac{1}{2}} = \rho_0(x_{j+\frac{1}{2}}) \\ \rho^{N_t}_{j+\frac{1}{2}} = \rho_1(x_{j+\frac{1}{2}}) \\ \rho^n_{j+\frac{1}{2}} \geq 0 \\ m_0^{n+\frac{1}{2}} = 0 \\ m_{N_x}^{n+\frac{1}{2}} = 0 \end{cases} \tag{A.14}$$

We emphasize that the source term $\zeta^{n+\frac{1}{2}}_{j+\frac{1}{2}}$ now represents $\rho^{n+\frac{1}{2}}_{j+\frac{1}{2}} \zeta^{n+\frac{1}{2}}_{j+\frac{1}{2}}$ due to the change of variable introduced in (3.51). The implicit constraint $\rho^n_{j+\frac{1}{2}} > 0$ will be imposed in the optimization procedure.

A.3.2 Solving Inverse Abel Transform with Wasserstein-Fisher-Rao Metric

Notice that **g** represents the data.

$$
\begin{cases}
\min\limits_{m,\rho,\zeta,\theta,\mathbf{f}} \Phi_{W_{2,\mathrm{WFR}}\Delta x,\Delta t}(\rho, m, \zeta) \\
s.t. \ \frac{\rho^{n+1}_{j+\frac{1}{2}} - \rho^{n}_{j+\frac{1}{2}}}{\Delta t} + \frac{m^{n+\frac{1}{2}}_{j+1} - m^{n+\frac{1}{2}}_{j}}{\Delta x} = \zeta^{n+\frac{1}{2}}_{j+\frac{1}{2}} \\
\rho^{0}_{j+\frac{1}{2}} = \mathbf{f}_j \\
\rho^{N_t}_{j+\frac{1}{2}} = \mathbf{g}_j \\
\rho^{n}_{j+\frac{1}{2}} \geq 0 \\
m^{n+\frac{1}{2}}_{0} = 0 \\
m^{n+\frac{1}{2}}_{N_x} = 0 \\
\mathbf{f}_j = (\Delta x)^{1-\alpha} \sum\limits_{k=0}^{j-1} \frac{\theta_k}{(j-(k+\frac{1}{2}))^{\alpha}}
\end{cases}
\tag{A.15}
$$

A.3.3 Solving Inverse Helmholtz Equation with Wasserstein-Fisher-Rao Metric

$$
\begin{cases}
\min\limits_{m,\rho,\zeta,\theta,u,\mathbf{f}} \Phi_{W_{2,\mathrm{WFR}}\Delta x,\Delta t}(\rho, m, \zeta) \\
s.t. \ \frac{\rho^{n+1}_{j+\frac{1}{2}} - \rho^{n}_{j+\frac{1}{2}}}{\Delta t} + \frac{m^{n+\frac{1}{2}}_{j+1} - m^{n+\frac{1}{2}}_{j}}{\Delta x} = \zeta^{n+\frac{1}{2}}_{j+\frac{1}{2}} \\
\rho^{0}_{j+\frac{1}{2}} = \mathbf{f}_j \\
\rho^{N_t}_{j+\frac{1}{2}} = \mathbf{g}_j \\
\rho^{n}_{j+\frac{1}{2}} \geq 0 \\
m^{n+\frac{1}{2}}_{0} = 0 \\
m^{n+\frac{1}{2}}_{N_x} = 0 \\
\frac{u_{j+1} - 2u_j + u_{j-1}}{\Delta x^2} + k^2(1+n_j)u_j + ik\theta_j u_j = q_j \\
u_0 = b_0, u_{N_x-1} = b_{N-1} \\
\mathbf{f}_j = (Re(u_j)^2 + Im(u_j)^2)\theta_j
\end{cases}
\tag{A.16}
$$

Or

$$
\begin{cases}
\min\limits_{m,\rho,\zeta,\theta,u,\mathbf{f}} \Phi_{W_{2,\mathrm{WFR}}\Delta x,\Delta t}(\rho, m, \zeta) \\
s.t.\ \ \frac{\rho_{j+\frac{1}{2}}^{n+1}-\rho_{j+\frac{1}{2}}^{n}}{\Delta t} + \frac{m_{j+1}^{n+\frac{1}{2}}-m_{j}^{n+\frac{1}{2}}}{\Delta x} = \zeta_{j+\frac{1}{2}}^{n+\frac{1}{2}} \\
\rho_{j+\frac{1}{2}}^{0} = \mathbf{f}_j \\
\rho_{j+\frac{1}{2}}^{N_t} = \mathbf{g}_j \\
\rho_{j+\frac{1}{2}}^{n} \geq 0 \\
m_{0}^{n+\frac{1}{2}} = 0 \\
m_{N_x}^{n+\frac{1}{2}} = 0 \\
\frac{Re(u_{j+1})-2Re(u_j)+Re(u_{j-1})}{\Delta x^2} + k^2(1+n_j)Re(u_j) - k\theta_j Im(u_j) \\
= Re(q_j) \\
\frac{Im(u_{j+1})-2Im(u_j)+Im(u_{j-1})}{\Delta x^2} + k^2(1+n_j)Im(u_j) + k\theta_j Re(u_j) \\
= Im(q_j) \\
u_0 = b_0, u_{N_x-1} = b_{N-1} \\
\mathbf{f}_j = (Re(u_j)^2 + Im(u_j)^2)\theta_j
\end{cases}
\tag{A.17}
$$

A.3.4 Solving Inverse Diffusion Equation with Wasserstein-Fisher-Rao Metric

$$
\begin{cases}
\min\limits_{m,\rho,\zeta,\theta,u,\mathbf{f}} \Phi_{W_{2,\mathrm{WFR}}\Delta x,\Delta t}(\rho, m, \zeta) \\
s.t. \ \frac{\rho^{n+1}_{j+\frac{1}{2}} - \rho^{n}_{j+\frac{1}{2}}}{\Delta t} + \frac{m^{n+\frac{1}{2}}_{j+1} - m^{n+\frac{1}{2}}_{j}}{\Delta x} = \zeta^{n+\frac{1}{2}}_{j+\frac{1}{2}} \\
\rho^{0}_{j+\frac{1}{2}} = \mathbf{f}_j \\
\rho^{N_t}_{j+\frac{1}{2}} = \mathbf{g}_j \\
\rho^{n}_{j+\frac{1}{2}} \geq 0 \\
m^{n+\frac{1}{2}}_{0} = 0 \\
m^{n+\frac{1}{2}}_{N_x} = 0 \\
-\gamma \frac{u_{j+1} - 2u_j + u_{j-1}}{\Delta x^2} + \theta_j u_j = q_j \\
u_0 = b_0, u_{N_x - 1} = b_{N-1} \\
\mathbf{f}_j = u_j \theta_j
\end{cases}
\tag{A.18}
$$

A.4 Wasserstein-UOT Metric

For two non-negative densities ρ_0, ρ_1 on a domain $\Omega \subset \mathbb{R}^n$, and given parameter α, the Wasserstein-UOT is obtained by optimizing

$$
\begin{cases}
W^2_{2,UOT}(\rho_0, \rho_1) = \inf\limits_{\rho,m,\zeta} \frac{1}{T} \int_0^T \int_\Omega \frac{m^2}{2\rho} + \frac{|\Omega|\zeta^2}{2\alpha} dt \\
s.t. \rho_t + \nabla \cdot m = \zeta \\
\rho(0, x) = \rho_0(x) \\
\rho(T, x) = \rho_1(x) \\
\rho \geq 0 \\
m \cdot n = 0 \ on \ \partial\Omega
\end{cases}
\tag{A.19}
$$

A.4.1 Discretization of Wasserstein-UOT Metric

Define

$$\Phi_{W_{2,\mathrm{UOT}}}\Delta x, \Delta t(\rho, m, \zeta) = \frac{1}{2T}\Delta x \Delta t \sum_{j=0}^{N_x-1}\sum_{n=0}^{N_t-1}\frac{(m_j^{n+\frac{1}{2}})^2 + (m_{j+1}^{n+\frac{1}{2}})^2}{\rho_{j+\frac{1}{2}}^n + \rho_{j+\frac{1}{2}}^{n+\frac{1}{2}}} + \Delta t \sum_{n=0}^{N_t-1}\frac{|\Omega|(\zeta^{n+\frac{1}{2}})^2}{2\alpha} \tag{A.20}$$

Then the discretized version of 1D Wasserstein-UOT Metric is

$$\begin{cases} \min\limits_{m,\rho,\zeta} \Phi_{W_{2,\mathrm{UOT}}\Delta x,\Delta t}(\rho, m, \zeta) \\ s.t.\ \ \frac{\rho_{j+\frac{1}{2}}^{n+1} - \rho_{j+\frac{1}{2}}^{n}}{\Delta t} + \frac{m_{j+1}^{n+\frac{1}{2}} - m_j^{n+\frac{1}{2}}}{\Delta x} = \zeta^{n+\frac{1}{2}} \\ \rho_{j+\frac{1}{2}}^0 = \rho_0(x_{j+\frac{1}{2}}) \\ \rho_{j+\frac{1}{2}}^{N_t} = \rho_1(x_{j+\frac{1}{2}}) \\ \rho_{j+\frac{1}{2}}^n \geq 0 \\ m_0^{n+\frac{1}{2}} = 0 \\ m_{N_x}^{n+\frac{1}{2}} = 0 \end{cases} \tag{A.21}$$

A.5 Wasserstein-GUOT Metric

For two non-negative densities ρ_0, ρ_1 on a domain $\Omega \subset \mathbb{R}^n$, and given parameter α, the Wasserstein-GUOT is obtained by optimizing

$$\begin{cases} W_{2,GUOT}^2(\rho_0, \rho_1) = \inf\limits_{\rho,m,\zeta} \frac{1}{T}\int_0^T \int_\Omega \frac{m^2}{2\rho} + \frac{\zeta^2}{2\alpha} dx dt \\ s.t. \rho_t + \nabla \cdot m = \zeta \\ \rho(0, x) = \rho_0(x) \\ \rho(T, x) = \rho_1(x) \\ \rho \geq 0 \\ m \cdot n = 0 \ on \ \partial\Omega \end{cases} \tag{A.22}$$

A.5.1 Discretization of Wasserstein-GUOT Metric

Define

$$\Phi_{W_{2,\mathrm{GUOT}}}\Delta x, \Delta t(\rho, m, \zeta) = \frac{1}{2T}\Delta x \Delta t \sum_{j=0}^{N_x-1}\sum_{n=0}^{N_t-1} \frac{(m_j^{n+\frac{1}{2}})^2 + (m_{j+1}^{n+\frac{1}{2}})^2}{\rho_{j+\frac{1}{2}}^n + \rho_{j+\frac{1}{2}}^{n+\frac{1}{2}}} + \frac{(\zeta_{j+\frac{1}{2}}^{n+\frac{1}{2}})^2}{\alpha} \tag{A.23}$$

Then the discretized version of 1D Wasserstein-GUOT Metric is

$$\begin{cases} \min\limits_{m,\rho,\zeta} \Phi_{W_{2,\mathrm{GUOT}}\Delta x,\Delta t}(\rho, m, \zeta) \\ s.t. \ \ \frac{\rho_{j+\frac{1}{2}}^{n+1} - \rho_{j+\frac{1}{2}}^{n}}{\Delta t} + \frac{m_{j+1}^{n+\frac{1}{2}} - m_j^{n+\frac{1}{2}}}{\Delta x} = \zeta_{j+\frac{1}{2}}^{n+\frac{1}{2}} \\ \rho_{j+\frac{1}{2}}^{0} = \rho_0(x_{j+\frac{1}{2}}) \\ \rho_{j+\frac{1}{2}}^{N_t} = \rho_1(x_{j+\frac{1}{2}}) \\ \rho_{j+\frac{1}{2}}^{n} \geq 0 \\ m_0^{n+\frac{1}{2}} = 0 \\ m_{N_x}^{n+\frac{1}{2}} = 0 \end{cases} \tag{A.24}$$

A.5.2 Solving Inverse Abel Transform with Wasserstein-GUOT Metric

Discretized problem formulation for Abel transform, given $\mathbf{g}$ as the data, try to find θ, such that $\mathbf{g} = Abel(\theta)$

$$\begin{cases} \min\limits_{m,\rho,\zeta,\theta,\mathbf{f}} \Phi_{W_{2,\text{GUOT}}\Delta x,\Delta t}(\rho, m, \zeta) \\ s.t. \ \frac{\rho^{n+1}_{j+\frac{1}{2}} - \rho^{n}_{j+\frac{1}{2}}}{\Delta t} + \frac{m^{n+\frac{1}{2}}_{j+1} - m^{n+\frac{1}{2}}_{j}}{\Delta x} = \zeta^{n+\frac{1}{2}}_{j+\frac{1}{2}} \\ \rho^{0}_{j+\frac{1}{2}} = \mathbf{f}_j \\ \rho^{N_t}_{j+\frac{1}{2}} = \mathbf{g}_j \\ \rho^{n}_{j+\frac{1}{2}} \geq 0 \\ m^{n+\frac{1}{2}}_{0} = 0 \\ m^{n+\frac{1}{2}}_{N_x} = 0 \\ \mathbf{f}_j = (\Delta x)^{1-\alpha} \sum\limits_{k=0}^{j-1} \frac{\theta_k}{(j-(k+\frac{1}{2}))^{\alpha}} \end{cases} \tag{A.25}$$

A.5.3 Solving Inverse Helmholtz Equation with Wasserstein-GUOT Metric

Discretized formulation for Helmholtz Equation

$$
\begin{cases}
\min\limits_{m,\rho,\zeta,\theta,u,\mathbf{f}} \Phi_{W_{2,\mathrm{GUOT}}\Delta x,\Delta t}(\rho, m, \zeta) \\
s.t.\ \ \frac{\rho_{j+\frac{1}{2}}^{n+1} - \rho_{j+\frac{1}{2}}^{n}}{\Delta t} + \frac{m_{j+1}^{n+\frac{1}{2}} - m_{j}^{n+\frac{1}{2}}}{\Delta x} = \zeta_{j+\frac{1}{2}}^{n+\frac{1}{2}} \\
\rho_{j+\frac{1}{2}}^{0} = \mathbf{f}_j \\
\rho_{j+\frac{1}{2}}^{N_t} = \mathbf{g}_j \\
\rho_{j+\frac{1}{2}}^{n} \geq 0 \\
m_{0}^{n+\frac{1}{2}} = 0 \\
m_{N_x}^{n+\frac{1}{2}} = 0 \\
\frac{Re(u_{j+1}) - 2Re(u_j) + Re(u_{j-1})}{\Delta x^2} + k^2(1+n_j)Re(u_j) - k\theta_j Im(u_j) \\
= Re(q_j) \\
\frac{Im(u_{j+1}) - 2Im(u_j) + Im(u_{j-1})}{\Delta x^2} + k^2(1+n_j)Im(u_j) + k\theta_j Re(u_j) \\
= Im(q_j) \\
u_0 = b_0, u_{N_x-1} = b_{N-1} \\
\mathbf{f}_j = (Re(u_j)^2 + Im(u_j)^2)\theta_j
\end{cases}
\tag{A.26}
$$

A.5.4 Solving Inverse Diffusion Equation with Wasserstein-GUOT Metric

The following is the discretized formulation for Diffusion Equation

$$
\begin{cases}
\min\limits_{m,\rho,\zeta,\theta,u,\mathbf{f}} \Phi_{W_{2,\mathrm{GUOT}}\Delta x,\Delta t}(\rho, m, \zeta) \\
s.t.\ \ \frac{\rho^{n+1}_{j+\frac{1}{2}} - \rho^{n}_{j+\frac{1}{2}}}{\Delta t} + \frac{m^{n+\frac{1}{2}}_{j+1} - m^{n+\frac{1}{2}}_{j}}{\Delta x} = \zeta^{n+\frac{1}{2}}_{j+\frac{1}{2}} \\
\rho^{0}_{j+\frac{1}{2}} = \mathbf{f}_j \\
\rho^{N_t}_{j+\frac{1}{2}} = \mathbf{g}_j \\
\rho^{n}_{j+\frac{1}{2}} \geq 0 \\
m^{n+\frac{1}{2}}_{0} = 0 \\
m^{n+\frac{1}{2}}_{N_x} = 0 \\
-\gamma \frac{u_{j+1} - 2u_j + u_{j-1}}{\Delta x^2} + \theta_j u_j = q_j \\
u_0 = b_0, u_{N_x - 1} = b_{N-1} \\
\mathbf{f}_j = u_j \theta_j
\end{cases}
\tag{A.27}
$$

A.6 Solving Inverse Problem Using Balanced Wasserstein Distance

A.6.1 Discretization of Balanced Wasserstein Metric

The continuous version of the problem will be

$$
W_2^2(\rho_0, \rho_1) = \begin{cases}
\inf\limits_{\rho,m,\theta,u} \frac{1}{T} \int_0^T \int_\Omega \frac{m^2}{2\rho} dx dt \\
s.t. \rho_t + \nabla \cdot m = 0 \\
\rho(0, x) = \rho_0(x) \\
\rho(T, x) = \rho_1(x) \\
\rho \geq 0 \\
m \cdot n = 0 \ on \ \partial\Omega
\end{cases}
\tag{A.28}
$$

Discretized problem formulation

$$
\begin{cases}
\min\limits_{m,\rho} \Phi_{W_2 \Delta x, \Delta t}(\rho, m) \\
s.t. \ \ \frac{\rho^{n+1}_{j+\frac{1}{2}} - \rho^{n}_{j+\frac{1}{2}}}{\Delta t} + \frac{m^{n+\frac{1}{2}}_{j+1} - m^{n+\frac{1}{2}}_{j}}{\Delta x} = 0 \\
\rho^{0}_{j+\frac{1}{2}} = \rho_0(x_{j+\frac{1}{2}}) \\
\rho^{N_t}_{j+\frac{1}{2}} = \rho_1(x_{j+\frac{1}{2}}) \\
\rho^{n}_{j+\frac{1}{2}} \geq 0 \\
m^{n+\frac{1}{2}}_{0} = 0 \\
m^{n+\frac{1}{2}}_{N_x} = 0
\end{cases}
\tag{A.29}
$$

where

$$
\Phi_{W_2 \Delta x, \Delta t}(\rho, m) = \frac{1}{2T} \Delta x \Delta t \sum_{j=0}^{N_x - 1} \sum_{n=0}^{N_t - 1} \frac{(m^n_j)^2 + (m^n_{j+1})^2}{\rho^n_{j+\frac{1}{2}} + \rho^{n+1}_{j+\frac{1}{2}}}
\tag{A.30}
$$

A.6.2 Solving Inverse Abel Transform with Balanced Optimal Transport

$$
\begin{cases}
\min\limits_{m,\rho,\theta,\mathbf{f}} \Phi_{W_2 \Delta x, \Delta t}(\rho, m) \\
s.t. \frac{\rho^{n+1}_{j+\frac{1}{2}} - \rho^{n}_{j+\frac{1}{2}}}{\Delta t} + \frac{m^{n+\frac{1}{2}}_{j+1} - m^{n+\frac{1}{2}}_{j}}{\Delta x} = 0 \\
\rho^{0}_{j+\frac{1}{2}} = \mathbf{f}_j \\
\rho^{N_t}_{j+\frac{1}{2}} = \mathbf{g}_j \\
\rho^{n}_{j+\frac{1}{2}} \geq 0 \\
m^{n+\frac{1}{2}}_{0} = 0 \\
m^{n+\frac{1}{2}}_{N_x} = 0 \\
\mathbf{f}_j = (\Delta x)^{1-\alpha} \sum\limits_{k=0}^{j-1} \frac{\theta_k}{(j-(k+\frac{1}{2}))^{\alpha}}
\end{cases}
\tag{A.31}
$$

A.6.3 Solving Inverse Helmholtz Equation with Balanced Optimal Transport

$$
\begin{cases}
\min\limits_{m,\rho,\theta,u,\mathbf{f}} \Phi_{W_2 \Delta x, \Delta t}(\rho, m) \\
s.t. \ \frac{\rho_{j+\frac{1}{2}}^{n+1} - \rho_{j+\frac{1}{2}}^{n}}{\Delta t} + \frac{m_{j+1}^{n+\frac{1}{2}} - m_{j}^{n+\frac{1}{2}}}{\Delta x} = 0 \\
\rho_{j+\frac{1}{2}}^{0} = \mathbf{f}_j \\
\rho_{j+\frac{1}{2}}^{N_t} = \mathbf{g}_j \\
\rho_{j+\frac{1}{2}}^{n} \geq 0 \\
m_{0}^{n+\frac{1}{2}} = 0 \\
m_{N_x}^{n+\frac{1}{2}} = 0 \\
\frac{Re(u_{j+1}) - 2Re(u_j) + Re(u_{j-1})}{\Delta x^2} + k^2(1+n_j)Re(u_j) - k\theta_j Im(u_j) \\
= Re(q_j) \\
\frac{Im(u_{j+1}) - 2Im(u_j) + Im(u_{j-1})}{\Delta x^2} + k^2(1+n_j)Im(u_j) + k\theta_j Re(u_j) \\
= Im(q_j) \\
u_0 = b_0, u_{N_x - 1} = b_{N-1} \\
\mathbf{f}_j = (Re(u_j)^2 + Im(u_j)^2)\theta_j
\end{cases}
\tag{A.32}
$$

A.6.4 Solving Inverse Diffusion Equation with Balanced Optimal Transport

$$
\begin{cases}
\min\limits_{m,\rho,\theta,u,\mathbf{f}} \Phi_{W_2 \Delta x, \Delta t}(\rho, m) \\
s.t. \ \ \frac{\rho^{n+1}_{j+\frac{1}{2}} - \rho^{n}_{j+\frac{1}{2}}}{\Delta t} + \frac{m^{n+\frac{1}{2}}_{j+1} - m^{n+\frac{1}{2}}_{j}}{\Delta x} = 0 \\
\rho^{0}_{j+\frac{1}{2}} = \mathbf{f}_j \\
\rho^{N_t}_{j+\frac{1}{2}} = \mathbf{g}_j \\
\rho^{n}_{j+\frac{1}{2}} \geq 0 \\
m^{n+\frac{1}{2}}_{0} = 0 \\
m^{n+\frac{1}{2}}_{N_x} = 0 \\
-\gamma \frac{u_{j+1} - 2u_j + u_{j-1}}{\Delta x^2} + \theta_j u_j = q_j \\
u_0 = b_0, u_{N_x - 1} = b_{N-1} \\
\mathbf{f}_j = u_j \theta_j
\end{cases}
\tag{A.33}
$$

A.7 Solving Inverse Problem Using Relaxed Quadratic Wasserstein Metric

A.7.1 Discretization of Relaxed Quadratic Wasserstein Metric

The continuous version of the relaxed quadratic Wasserstein metric is

$$
W^2_{2,Mixed}(\rho_0, \rho_1) = \begin{cases}
\inf\limits_{m,\rho} \{ \frac{1}{T} \int_\Omega \int_0^T \frac{|m|^2}{2\rho} dt dx + \frac{\beta}{2} \int |\rho(T,x) - \rho_1(x)|^2 dx \} \\
s.t. \rho_t + \nabla \cdot m = 0 \\
\rho(0,x) = \rho_0(x) \\
\rho \geq 0 \\
m \cdot n = 0 \ on \ \partial\Omega
\end{cases}
\tag{A.34}
$$

Discretized problem formulation

$$
\begin{cases}
\min\limits_{m,\rho} \Phi_{W_{2,\text{mixed}}\Delta x,\Delta t}(\rho, m, \rho_1) \\
s.t.\ \ \frac{\rho^{n+1}_{j+\frac{1}{2}} - \rho^{n}_{j+\frac{1}{2}}}{\Delta t} + \frac{m^{n+\frac{1}{2}}_{j+1} - m^{n+\frac{1}{2}}_{j}}{\Delta x} = 0 \\
\rho^{0}_{j+\frac{1}{2}} = \rho_0(x_{j+\frac{1}{2}}) \\
\rho^{n}_{j+\frac{1}{2}} \geq 0 \\
m^{n+\frac{1}{2}}_{0} = 0 \\
m^{n+\frac{1}{2}}_{N_x} = 0
\end{cases}
\tag{A.35}
$$

where

$$
\Phi_{W_{2,\text{mixed}}\Delta x,\Delta t}(\rho, m, \mathbf{f}) = \frac{1}{2T}\Delta x \Delta t \sum_{j=0}^{N_x-1}\sum_{n=0}^{N_t-1} \frac{(m^{n+\frac{1}{2}}_{j})^2 + (m^{n+\frac{1}{2}}_{j+1})^2}{\rho^{n}_{j+\frac{1}{2}} + \rho^{n+1}_{j+\frac{1}{2}}} + \frac{\beta}{2}\Delta x \sum_{j=0}^{N_x-1} (\rho^{N_t}_{j+\frac{1}{2}} - f_j)^2
$$

A.7.2 Solving Inverse Abel Transform with Relaxed Quadratic Wasserstein Metric

$$
\begin{cases}
\min\limits_{m,\rho,\theta,\mathbf{f}} \Phi_{W_{2,\text{mixed}}\Delta x,\Delta t}(\rho, m, \mathbf{f}) \\
s.t.\ \ \frac{\rho^{n+1}_{j+\frac{1}{2}} - \rho^{n}_{j+\frac{1}{2}}}{\Delta t} + \frac{m^{n+\frac{1}{2}}_{j+1} - m^{n+\frac{1}{2}}_{j}}{\Delta x} = 0 \\
\rho^{0}_{j+\frac{1}{2}} = \mathbf{g}_j \\
\rho^{n}_{j+\frac{1}{2}} \geq 0 \\
m^{n+\frac{1}{2}}_{0} = 0 \\
m^{n+\frac{1}{2}}_{N_x} = 0 \\
\mathbf{f}_j = (\Delta x)^{1-\alpha} \sum\limits_{k=0}^{j-1} \frac{\theta_k}{(j-(k+\frac{1}{2}))^{\alpha}}
\end{cases}
\tag{A.36}
$$

A.7.3 Solving Inverse Helmholtz Equation with Relaxed Quadratic Wasserstein Metric

$$
\begin{cases}
\min\limits_{m,\rho,\theta,u,\mathbf{f}} \Phi_{W_{2,\text{mixed}}\Delta x,\Delta t}(\rho, m, \mathbf{f}) \\
s.t.\ \ \frac{\rho_{j+\frac{1}{2}}^{n+1}-\rho_{j+\frac{1}{2}}^{n}}{\Delta t} + \frac{m_{j+1}^{n+\frac{1}{2}}-m_{j}^{n+\frac{1}{2}}}{\Delta x} = 0 \\
\rho_{j+\frac{1}{2}}^{0} = \mathbf{g}_j \\
\rho_{j+\frac{1}{2}}^{n} \geq 0 \\
m_{0}^{n+\frac{1}{2}} = 0 \\
m_{N_x}^{n+\frac{1}{2}} = 0 \\
\frac{Re(u_{j+1})-2Re(u_j)+Re(u_{j-1})}{\Delta x^2} + k^2(1+n_j)Re(u_j) - k\theta_j Im(u_j) \\
= Re(q_j) \\
\frac{Im(u_{j+1})-2Im(u_j)+Im(u_{j-1})}{\Delta x^2} + k^2(1+n_j)Im(u_j) + k\theta_j Re(u_j) \\
= Im(q_j) \\
u_0 = b_0, u_{N_x-1} = b_{N-1} \\
\mathbf{f}_j = (Re(u_j)^2 + Im(u_j)^2)\theta_j
\end{cases}
\tag{A.37}
$$

A.7.4 Solving Inverse Diffusion Equation with Relaxed Quadratic Wasserstein Metric

$$
\begin{cases}
\min\limits_{m,\rho,\theta,u,\mathbf{f}} \Phi_{W_{2,\text{mixed}}\Delta x,\Delta t}(\rho, m, \mathbf{f}) \\
s.t.\ \ \frac{\rho_{j+\frac{1}{2}}^{n+1}-\rho_{j+\frac{1}{2}}^{n}}{\Delta t} + \frac{m_{j+1}^{n+\frac{1}{2}}-m_{j}^{n+\frac{1}{2}}}{\Delta x} = 0 \\
\rho_{j+\frac{1}{2}}^{0} = \mathbf{g}_j \\
\rho_{j+\frac{1}{2}}^{n} \geq 0 \\
m_{0}^{n+\frac{1}{2}} = 0 \\
m_{N_x}^{n+\frac{1}{2}} = 0 \\
-\gamma\frac{u_{j+1}-2u_j+u_{j-1}}{\Delta x^2} + \theta_j u_j = q_j \\
u_0 = b_0, u_{N_x-1} = b_{N-1} \\
\mathbf{f}_j = u_j\theta_j
\end{cases}
\tag{A.38}
$$

A.8 Solving Inverse Problem Using W_1 Wasserstein Metric

A.8.1 Discretization of W_1 Wasserstein Metric

The continuous version of the relaxed quadratic Wasserstein metric is

$$W_1(\mathbf{f},\mathbf{g}) = \begin{cases} \inf\limits_{\mathbf{m}} \int_\Omega \|\mathbf{m}(x)\| dx \\ s.t. \nabla \cdot \mathbf{m} = \mathbf{f} - \mathbf{g} \end{cases} \tag{A.39}$$

Discretized problem formulation

$$\begin{aligned} &\min_{m,\rho} \Phi_{W_1 \Delta x}(m) \\ &s.t. \frac{m_{j+1} - m_j}{\Delta x} = \mathbf{f}_j - \mathbf{g}_j \end{aligned} \tag{A.40}$$

where

$$\Phi_{W_1 \Delta x}(m) = \Delta x \sum_{j=0}^{N_x - 1} |m_j| \tag{A.41}$$

A.9 Solving Inverse Problem Using UW_1 Wasserstein Metric

A.9.1 Discretization of UW_1 Wasserstein Metric

The continuous version of the relaxed quadratic Wasserstein metric is

$$UW_1(\mathbf{f},\mathbf{g}) = \begin{cases} \inf\limits_{\mathbf{m}} \int_\Omega \|\mathbf{m}(x)\| + \beta \|\zeta\| dx \\ s.t. \nabla \cdot \mathbf{m} - \zeta = \mathbf{f} - \mathbf{g} \end{cases} \tag{A.42}$$

Discretized problem formulation

$$\begin{aligned} &\min_{m,\rho} \Phi_{UW_1 \Delta x}(m, \zeta) \\ &s.t. \frac{m_{j+1} - m_j}{\Delta x} - \zeta_{j+\frac{1}{2}} = \mathbf{f}_j - \mathbf{g}_j \end{aligned} \tag{A.43}$$

where

$$\Phi_{UW_1\Delta x}(m,\zeta) = \Delta x \sum_{j=0}^{N_x-1} |m_j| + \beta|\zeta_{j+\frac{1}{2}}| \tag{A.44}$$

A.10 Solving Inverse Problem Using L^2 Norm

$$\begin{cases} \min\limits_{u} \frac{1}{2}\Delta x \sum\limits_{j=0}^{Nx-1} (f_j - \mathbf{g}_j)^2 \\ s.t.\ \ Discrete\ forward\ problem \end{cases} \tag{A.45}$$

A.11 Solving Inverse Problem Using $\mathcal{H}^{-1}$ Norm

$$\begin{cases} \min\limits_{u} \frac{1}{2\Delta x} \sum\limits_{j=0}^{Nx-1} (P_{j+1} - P_j)^2 \\ s.t.\ \ -\frac{P_{j+1}-2P_j+P_{j-1}}{\Delta x^2} = \mathbf{g}_j - \mathbf{f}_j \\ P_0 = 0 \\ P_{N_x} = 0 \\ s.t.\ \ Discrete\ forward\ problem \end{cases} \tag{A.46}$$

A.12 Discretization in two spatial dimensions.

In the two-dimensional case with $\Omega = (0, L_x) \times (0, L_y)$, we also use a uniform structured temporal-spatial grid. Let $N_t = 10$, $N_x = 50$ and $N_y = 50$ be the number of intervals in the t, x and y variables respectively, then the grid (t_i, x_j, y_k) is defined as

$$t_i = i\Delta t,\ \ 0 \le i \le N_t; \quad x_j = j\Delta x,\ \ 0 \le j \le N_x; \qquad y_k = k\Delta y,\ \ 0 \le k \le N_y$$

where $\Delta t = \frac{T}{N_t}$, $\Delta x = \frac{L_x}{N_x}$, and $\Delta y = \frac{L_y}{N_y}$.

We again use staggered grids for (ρ, ζ) and $\mathbf{m} = (m_x, m_y)$ to discretize them as

$$\rho^n_{i+\frac{1}{2},j+\frac{1}{2}} = \rho(\frac{x_i + x_{i+1}}{2}, \frac{y_j + y_{j+1}}{2}, t_n), \quad \zeta^{n+\frac{1}{2}}_{i+\frac{1}{2},j+\frac{1}{2}} = \zeta(\frac{x_i + x_{i+1}}{2}, \frac{y_j + y_{j+1}}{2}, \frac{t_n + t_{n+1}}{2})$$
$$(m_x)^{n+\frac{1}{2}}_{i,j+\frac{1}{2}} = m_x(x_i, \frac{y_j + y_{j+1}}{2}, \frac{t_n + t_{n+1}}{2})$$
$$(m_y)^{n+\frac{1}{2}}_{i+\frac{1}{2},j} = m_y(\frac{x_i + x_{i+1}}{2}, y_j, \frac{t_n + t_{n+1}}{2}) .$$

This gives the following discretization scheme for the transport equation:

$$\begin{aligned}
\frac{\rho^{n+1}_{i+\frac{1}{2},j+\frac{1}{2}} - \rho^n_{i+\frac{1}{2},j+\frac{1}{2}}}{\Delta t} + \frac{(m_x)^{n+\frac{1}{2}}_{i+1,j+\frac{1}{2}} - (m_x)^{n+\frac{1}{2}}_{i,j+\frac{1}{2}}}{\Delta x} + \frac{(m_y)^{n+\frac{1}{2}}_{i+\frac{1}{2},j+1} - (m_y)^{n+\frac{1}{2}}_{i+\frac{1}{2},j}}{\Delta y} &= \zeta^{n+\frac{1}{2}}_{i+\frac{1}{2},j+\frac{1}{2}} \\
\rho^0_{i+\frac{1}{2},j+\frac{1}{2}} &= f_{i+\frac{1}{2},j+\frac{1}{2}} \\
\rho^{N_t}_{i+\frac{1}{2},j+\frac{1}{2}} &= g_{i+\frac{1}{2},j+\frac{1}{2}} \\
(m_x)^{n+\frac{1}{2}}_{0,j+\frac{1}{2}} = 0 \qquad (m_x)^{n+\frac{1}{2}}_{N_x,j+\frac{1}{2}} &= 0 \\
(m_y)^{n+\frac{1}{2}}_{i+\frac{1}{2},0} = 0 \qquad (m_y)^{n+\frac{1}{2}}_{i+\frac{1}{2},N_y} &= 0 .
\end{aligned} \tag{A.47}$$

A.12.1 Diffusion equation in 2D

In 2D simulation, we mainly use diffusion equation as an example.

$$\begin{aligned}
&-\gamma(\frac{u_{i-1,j} + u_{i+1,j} - 2u_{i,j}}{\Delta x^2} + \frac{u_{i,j-1} + u_{i,j+1} - 2u_{i,j}}{\Delta y^2}) + \theta_{i,j} u_{i,j} = q_{i,j} \\
&u_{0,j} = (b_{x_0})_j \\
&u_{N_x,j} = (b_{N_x})_j \\
&u_{i,0} = (b_{y_0})_i \\
&u_{i,N_y} = (b_{N_y})_i \\
&f_{i,j} = u_{i,j}\theta_{i,j}
\end{aligned} \tag{A.48}$$

Where $q(x, y) = 1 + 3xe^{xy}$, and other parameters $b_{x_0} = b_{N_x} = b_{y_0} = b_{N_y} = 1, \gamma = 0.1$

A.12.2 Solving Inverse Diffusion Equation with Mixed Relaxed Quadratic Wasserstein Metric in 2D

The discrete version of the cost functional in this case, following the trapezoidal rule for spatial and temporal integration, takes the form

$$\begin{aligned}\Phi_{2D,W_{2,\text{mixed}}\Delta x,\Delta y,\Delta t}(\rho,m,\mathbf{f}) = \frac{1}{2T}\Delta x\Delta y\Delta t \sum_{n=0}^{N_t-1}\sum_{i=0}^{N_x-1}\sum_{j=0}^{N_y-1} \\ \frac{((m_x)^{n+\frac{1}{2}}_{i,j+\frac{1}{2}})^2+((m_x)^{n+\frac{1}{2}}_{i+1,j+\frac{1}{2}})^2+((m_y)^{n+\frac{1}{2}}_{i+\frac{1}{2},j})^2+((m_y)^{n+\frac{1}{2}}_{i+\frac{1}{2},j+1})^2}{\rho^n_{i+\frac{1}{2},j+\frac{1}{2}}+\rho^{n+1}_{i+\frac{1}{2},j+\frac{1}{2}}}+ \\ \frac{\beta}{2}\Delta x\Delta y\sum_{i=0}^{N_x-1}\sum_{j=0}^{N_y-1}(\rho^{N_t}_{i+\frac{1}{2},j+\frac{1}{2}}-\mathbf{f}_{i,j})^2\end{aligned} \tag{A.49}$$

The discrete problem for inverse diffusion equation would be

$$
\begin{cases}
\min\limits_{m,\rho,\theta,\mathbf{f},u} \Phi_{2D,W_{2,\text{mixed}}\Delta x,\Delta y,\Delta t}(\rho, m, \mathbf{f}) \\
s.t. \\
\frac{\rho^{n+1}_{i+\frac{1}{2},j+\frac{1}{2}} - \rho^{n}_{i+\frac{1}{2},j+\frac{1}{2}}}{\Delta t} + \frac{(m_x)^{n+\frac{1}{2}}_{i+1,j+\frac{1}{2}} - (m_x)^{n+\frac{1}{2}}_{i,j+\frac{1}{2}}}{\Delta x} + \frac{(m_y)^{n+\frac{1}{2}}_{i+\frac{1}{2},j+1} - (m_y)^{n+\frac{1}{2}}_{i+\frac{1}{2},j}}{\Delta y} = 0 \\
\rho^{0}_{i+\frac{1}{2},j+\frac{1}{2}} = \mathbf{g}_{i,j} \\
\rho^{n}_{j+\frac{1}{2}} \geq 0 \\
(m_x)^{n+\frac{1}{2}}_{0,j+\frac{1}{2}} = 0 \\
(m_x)^{n+\frac{1}{2}}_{N_x,j+\frac{1}{2}} = 0 \\
(m_y)^{n+\frac{1}{2}}_{i+\frac{1}{2},0} = 0 \\
(m_y)^{n+\frac{1}{2}}_{i+\frac{1}{2},N_y} = 0 \\
-\gamma(\frac{u_{i-1,j}+u_{i+1,j}-2u_{i,j}}{\Delta x^2} + \frac{u_{i,j-1}+u_{i,j+1}-2u_{i,j}}{\Delta y^2}) + \theta_{i,j}u_{i,j} = q_{i,j} \\
u_{0,j} = (b_{x0})_j \\
u_{N_x,j} = (b_{Nx})_j \\
u_{i,0} = (b_{y0})_i \\
u_{i,N_y} = (b_{Ny})_i \\
f_{i,j} = u_{i,j}\theta_{i,j}
\end{cases}
\tag{A.50}
$$

A.12.3 Solving Inverse Diffusion Equation with Wasserstein-Fisher-Rao Metric in 2D

The discrete version of the cost functional for $W_{2,\text{WFR}}$, following the trapezoidal rule for spatial and temporal integration, takes the form

$$
\begin{aligned}
&\Phi_{2D,W_{2,WFR}\Delta x,\Delta y,\Delta t}(\rho, m, \zeta) = \frac{1}{2T}\Delta x\Delta y\Delta t \sum_{n=0}^{N_t-1}\sum_{i=0}^{N_x-1}\sum_{j=0}^{N_y-1} \\
&\frac{((m_x)^{n+\frac{1}{2}}_{i,j+\frac{1}{2}})^2+((m_x)^{n+\frac{1}{2}}_{i+1,j+\frac{1}{2}})^2+((m_y)^{n+\frac{1}{2}}_{i+\frac{1}{2},j})^2+((m_y)^{n+\frac{1}{2}}_{i+\frac{1}{2},j+1})^2}{\rho^{n}_{i+\frac{1}{2},j+\frac{1}{2}}+\rho^{n+1}_{i+\frac{1}{2},j+\frac{1}{2}}} + \\
&\frac{1}{2}\Delta x\Delta y\Delta t \sum_{n=0}^{N_t-1}\sum_{i=0}^{N_x-1}\sum_{j=0}^{N_y-1} \frac{2(\zeta^{n+\frac{1}{2}}_{i+\frac{1}{2},j+\frac{1}{2}})^2}{\rho^{n}_{i+\frac{1}{2},j+\frac{1}{2}}+\rho^{n+1}_{i+\frac{1}{2},j+\frac{1}{2}}}
\end{aligned}
\tag{A.51}
$$

The discrete problem for inverse diffusion equation would be adding the 2D diffusion equation into the constraints.

$$
\begin{cases}
\min\limits_{m,\rho,\theta,\mathbf{f},u,\zeta} \Phi_{2D,W_{2,\mathrm{WFR}}\Delta x,\Delta y,\Delta t}(\rho, m, \zeta) \\
s.t. \frac{\rho^{n+1}_{i+\frac{1}{2},j+\frac{1}{2}} - \rho^{n}_{i+\frac{1}{2},j+\frac{1}{2}}}{\Delta t} + \frac{(m_x)^{n+\frac{1}{2}}_{i+1,j+\frac{1}{2}} - (m_x)^{n+\frac{1}{2}}_{i,j+\frac{1}{2}}}{\Delta x} + \frac{(m_y)^{n+\frac{1}{2}}_{i+\frac{1}{2},j+1} - (m_y)^{n+\frac{1}{2}}_{i+\frac{1}{2},j}}{\Delta y} = \zeta^{n+\frac{1}{2}}_{i+\frac{1}{2},j+\frac{1}{2}} \\
\rho^{0}_{i+\frac{1}{2},j+\frac{1}{2}} = \mathbf{g}_{i,j} \\
\rho^{N_t+1}_{i+\frac{1}{2},j+\frac{1}{2}} = \mathbf{f}_{i,j} \\
\rho^{n}_{j+\frac{1}{2}} \geq 0 \\
(m_x)^{n+\frac{1}{2}}_{0,j+\frac{1}{2}} = 0 \\
(m_x)^{n+\frac{1}{2}}_{N_x,j+\frac{1}{2}} = 0 \\
(m_y)^{n+\frac{1}{2}}_{i+\frac{1}{2},0} = 0 \\
(m_y)^{n+\frac{1}{2}}_{i+\frac{1}{2},N_y} = 0 \\
-\gamma(\frac{u_{i-1,j}+u_{i+1,j}-2u_{i,j}}{\Delta x^2} + \frac{u_{i,j-1}+u_{i,j+1}-2u_{i,j}}{\Delta y^2}) + \theta_{i,j}u_{i,j} = q_{i,j} \\
u_{0,j} = (b_{x0})_j \\
u_{N_x,j} = (b_{Nx})_j \\
u_{i,0} = (b_{y0})_i \\
u_{i,N_y} = (b_{Ny})_i \\
f_{i,j} = u_{i,j}\theta_{i,j}
\end{cases}
\tag{A.52}
$$

A.12.4 Solving Inverse Diffusion Equation with Wasserstein-GUOT Metric in 2D

The discrete version of the cost functional for $W_{2,\mathrm{GUOT}}$, following the trapezoidal rule for spatial and temporal integration, takes the form

$$\Phi_{2D,W_{2,\mathrm{GUOT}}\Delta x,\Delta y,\Delta t}(\rho,m,\zeta)=\frac{1}{2T}\Delta x\Delta y\Delta t\sum_{n=0}^{N_t-1}\sum_{i=0}^{N_x-1}\sum_{j=0}^{N_y-1}\frac{((m_x)^{n+\frac{1}{2}}_{i,j+\frac{1}{2}})^2+((m_x)^{n+\frac{1}{2}}_{i+1,j+\frac{1}{2}})^2+((m_y)^{n+\frac{1}{2}}_{i+\frac{1}{2},j})^2+((m_y)^{n+\frac{1}{2}}_{i+\frac{1}{2},j+1})^2}{\rho^{n}_{i+\frac{1}{2},j+\frac{1}{2}}+\rho^{n+1}_{i+\frac{1}{2},j+\frac{1}{2}}}+\frac{1}{2\alpha}\Delta x\Delta y\Delta t\sum_{n=0}^{N_t-1}\sum_{i=0}^{N_x-1}\sum_{j=0}^{N_y-1}(\zeta^{n+\frac{1}{2}}_{i+\frac{1}{2},j+\frac{1}{2}})^2 \tag{A.53}$$

The discrete problem for inverse diffusion equation would be adding the 2D diffusion equation into the constraints.

$$\begin{cases}\min\limits_{m,\rho,\theta,\mathbf{f},u,\zeta}\Phi_{2D,W_{2,\mathrm{GUOT}}\Delta x,\Delta y,\Delta t}(\rho,m,\zeta)\\ s.t.\ \frac{\rho^{n+1}_{i+\frac{1}{2},j+\frac{1}{2}}-\rho^{n}_{i+\frac{1}{2},j+\frac{1}{2}}}{\Delta t}+\frac{(m_x)^{n+\frac{1}{2}}_{i+1,j+\frac{1}{2}}-(m_x)^{n+\frac{1}{2}}_{i,j+\frac{1}{2}}}{\Delta x}+\frac{(m_y)^{n+\frac{1}{2}}_{i+\frac{1}{2},j+1}-(m_y)^{n+\frac{1}{2}}_{i+\frac{1}{2},j}}{\Delta y}=\zeta^{n+\frac{1}{2}}_{i+\frac{1}{2},j+\frac{1}{2}}\\ \rho^{0}_{i+\frac{1}{2},j+\frac{1}{2}}=\mathbf{g}_{i,j}\\ \rho^{N_t+1}_{i+\frac{1}{2},j+\frac{1}{2}}=\mathbf{f}_{i,j}\\ \rho^{n}_{j+\frac{1}{2}}\geq 0\\ (m_x)^{n+\frac{1}{2}}_{0,j+\frac{1}{2}}=0\\ (m_x)^{n+\frac{1}{2}}_{N_x,j+\frac{1}{2}}=0\\ (m_y)^{n+\frac{1}{2}}_{i+\frac{1}{2},0}=0\\ (m_y)^{n+\frac{1}{2}}_{i+\frac{1}{2},N_y}=0\\ -\gamma(\frac{u_{i-1,j}+u_{i+1,j}-2u_{i,j}}{\Delta x^2}+\frac{u_{i,j-1}+u_{i,j+1}-2u_{i,j}}{\Delta y^2})+\theta_{i,j}u_{i,j}=q_{i,j}\\ u_{0,j}=(b_{x0})_j\\ u_{N_x,j}=(b_{Nx})_j\\ u_{i,0}=(b_{y0})_i\\ u_{i,N_y}=(b_{Ny})_i\\ f_{i,j}=u_{i,j}\theta_{i,j}\end{cases} \tag{A.54}$$

A.12.5 Solving Inverse Diffusion Equation with Wasserstein-UOT Metric in 2D

The discrete version of the cost functional for $W_{2,\mathrm{UOT}}$, following the trapezoidal rule for spatial and temporal integration, takes the form

$$\Phi_{2D,W_{2,\mathrm{UOT}}\Delta x,\Delta y,\Delta t}(\rho,m,\zeta)=\frac{1}{2T}\Delta x\Delta y\Delta t\sum_{n=0}^{N_t-1}\sum_{i=0}^{N_x-1}\sum_{j=0}^{N_y-1}\frac{((m_x)^{n+\frac{1}{2}}_{i,j+\frac{1}{2}})^2+((m_x)^{n+\frac{1}{2}}_{i+1,j+\frac{1}{2}})^2+((m_y)^{n+\frac{1}{2}}_{i+\frac{1}{2},j})^2+((m_y)^{n+\frac{1}{2}}_{i+\frac{1}{2},j+1})^2}{\rho^n_{i+\frac{1}{2},j+\frac{1}{2}}+\rho^{n+1}_{i+\frac{1}{2},j+\frac{1}{2}}}+\frac{1}{2\alpha}\Delta t\sum_{n=0}^{N_t-1}(\zeta^{n+\frac{1}{2}})^2 \tag{A.55}$$

The discrete problem for inverse diffusion equation would be adding the 2D diffusion equation into the constraints.

$$
\begin{cases}
\min\limits_{m,\rho,\theta,\mathbf{f},u,\zeta} \Phi_{2D,W_{2,\mathrm{UOT}}\Delta x,\Delta y,\Delta t}(\rho, m, \zeta) \\
s.t. \frac{\rho^{n+1}_{i+\frac{1}{2},j+\frac{1}{2}} - \rho^{n}_{i+\frac{1}{2},j+\frac{1}{2}}}{\Delta t} + \frac{(m_x)^{n+\frac{1}{2}}_{i+1,j+\frac{1}{2}} - (m_x)^{n+\frac{1}{2}}_{i,j+\frac{1}{2}}}{\Delta x} + \frac{(m_y)^{n+\frac{1}{2}}_{i+\frac{1}{2},j+1} - (m_y)^{n+\frac{1}{2}}_{i+\frac{1}{2},j}}{\Delta y} = \zeta^{n+\frac{1}{2}} \\
\rho^{0}_{i+\frac{1}{2},j+\frac{1}{2}} = \mathbf{g}_{i,j} \\
\rho^{N_t+1}_{i+\frac{1}{2},j+\frac{1}{2}} = \mathbf{f}_{i,j} \\
\rho^{n}_{j+\frac{1}{2}} \geq 0 \\
(m_x)^{n+\frac{1}{2}}_{0,j+\frac{1}{2}} = 0 \\
(m_x)^{n+\frac{1}{2}}_{N_x,j+\frac{1}{2}} = 0 \\
(m_y)^{n+\frac{1}{2}}_{i+\frac{1}{2},0} = 0 \\
(m_y)^{n+\frac{1}{2}}_{i+\frac{1}{2},N_y} = 0 \\
-\gamma(\frac{u_{i-1,j}+u_{i+1,j}-2u_{i,j}}{\Delta x^2} + \frac{u_{i,j-1}+u_{i,j+1}-2u_{i,j}}{\Delta y^2}) + \theta_{i,j}u_{i,j} = q_{i,j} \\
u_{0,j} = (b_{x0})_j \\
u_{N_x,j} = (b_{Nx})_j \\
u_{i,0} = (b_{y0})_i \\
u_{i,N_y} = (b_{Ny})_i \\
f_{i,j} = u_{i,j}\theta_{i,j}
\end{cases}
\tag{A.56}
$$

A.12.6 Solving Inverse Diffusion Equation with Balanced Wasserstein Metric in 2D

The discrete version of the cost functional for W_2, following the trapezoidal rule for spatial and temporal integration, takes the form

$$
\Phi_{2D,W_2\Delta x,\Delta y,\Delta t}(\rho, m) = \frac{1}{2T}\Delta x\Delta y\Delta t \sum_{n=0}^{N_t-1}\sum_{i=0}^{N_x-1}\sum_{j=0}^{N_y-1} \frac{((m_x)^{n+\frac{1}{2}}_{i,j+\frac{1}{2}})^2+((m_x)^{n+\frac{1}{2}}_{i+1,j+\frac{1}{2}})^2+((m_y)^{n+\frac{1}{2}}_{i+\frac{1}{2},j})^2+((m_y)^{n+\frac{1}{2}}_{i+\frac{1}{2},j+1})^2}{\rho^{n}_{i+\frac{1}{2},j+\frac{1}{2}}+\rho^{n+1}_{i+\frac{1}{2},j+\frac{1}{2}}}
\tag{A.57}
$$

The discrete problem for inverse diffusion equation would be adding the 2D diffusion equation into the constraints.

$$
\begin{cases}
\min\limits_{m,\rho,\theta,\mathbf{f},u} \Phi_{2D,W_2\Delta x,\Delta y,\Delta t}(\rho, m) \\
s.t. \frac{\rho^{n+1}_{i+\frac{1}{2},j+\frac{1}{2}} - \rho^{n}_{i+\frac{1}{2},j+\frac{1}{2}}}{\Delta t} + \frac{(m_x)^{n+\frac{1}{2}}_{i+1,j+\frac{1}{2}} - (m_x)^{n+\frac{1}{2}}_{i,j+\frac{1}{2}}}{\Delta x} + \frac{(m_y)^{n+\frac{1}{2}}_{i+\frac{1}{2},j+1} - (m_y)^{n+\frac{1}{2}}_{i+\frac{1}{2},j}}{\Delta y} = 0 \\
\rho^{0}_{i+\frac{1}{2},j+\frac{1}{2}} = \mathbf{g}_{i,j} \\
\rho^{N_t+1}_{i+\frac{1}{2},j+\frac{1}{2}} = \mathbf{f}_{i,j} \\
\rho^{n}_{j+\frac{1}{2}} \geq 0 \\
(m_x)^{n+\frac{1}{2}}_{0,j+\frac{1}{2}} = 0 \\
(m_x)^{n+\frac{1}{2}}_{N_x,j+\frac{1}{2}} = 0 \\
(m_y)^{n+\frac{1}{2}}_{i+\frac{1}{2},0} = 0 \\
(m_y)^{n+\frac{1}{2}}_{i+\frac{1}{2},N_y} = 0 \\
-\gamma(\frac{u_{i-1,j}+u_{i+1,j}-2u_{i,j}}{\Delta x^2} + \frac{u_{i,j-1}+u_{i,j+1}-2u_{i,j}}{\Delta y^2}) + \theta_{i,j}u_{i,j} = q_{i,j} \\
u_{0,j} = (b_{x0})_j \\
u_{N_x,j} = (b_{Nx})_j \\
u_{i,0} = (b_{y0})_i \\
u_{i,N_y} = (b_{Ny})_i \\
f_{i,j} = u_{i,j}\theta_{i,j}
\end{cases}
\tag{A.58}
$$

A.12.7 Solving Inverse Diffusion Equation with UW_1 Metric in 2D

The discrete version of the cost functional for UW_1, following the trapezoidal rule for spatial and temporal integration, takes the form

$$
\begin{aligned}
&\Phi_{2D,UW_1\Delta x,\Delta y}(m, \zeta) = \Delta x \Delta y \sum_{i=0}^{N_x-1} \sum_{j=0}^{N_y-1} \\
&\frac{1}{2}(|(m_x)_{i,j+\frac{1}{2}}| + |(m_x)_{i+1,j+\frac{1}{2}}| + |(m_y)_{i+\frac{1}{2},j}| + |(m_y)_{i+\frac{1}{2},j+1}|) + \beta|\zeta_{i+\frac{1}{2},j+\frac{1}{2}}|
\end{aligned}
\tag{A.59}
$$

The discrete problem for inverse diffusion equation would be adding the 2D diffusion equation into the constraints.

$$\begin{cases} \min\limits_{m,\theta,\mathbf{f},u,\zeta} \Phi_{2D,UW_1\Delta x,\Delta y}(m,\zeta) \\ s.t. \frac{(m_x)_{i+1,j+\frac{1}{2}}-(m_x)_{i,j+\frac{1}{2}}}{\Delta x} + \frac{(m_y)_{i+\frac{1}{2},j+1}-(m_y)_{i+\frac{1}{2},j}}{\Delta y} - \zeta_{i+\frac{1}{2},j+\frac{1}{2}} = \mathbf{f}_{i,j} - \mathbf{g}_{i,j} \end{cases} \tag{A.60}$$

Appendix B: Neural Network structure and Training

B.1 Network structure

For the sake of reproducibility of our research, we provide here the structures of the encoder, decoder and predictor networks we used in the encoder-decoder-predictor training framework described in Section 4.6.2; see Figure B.1.

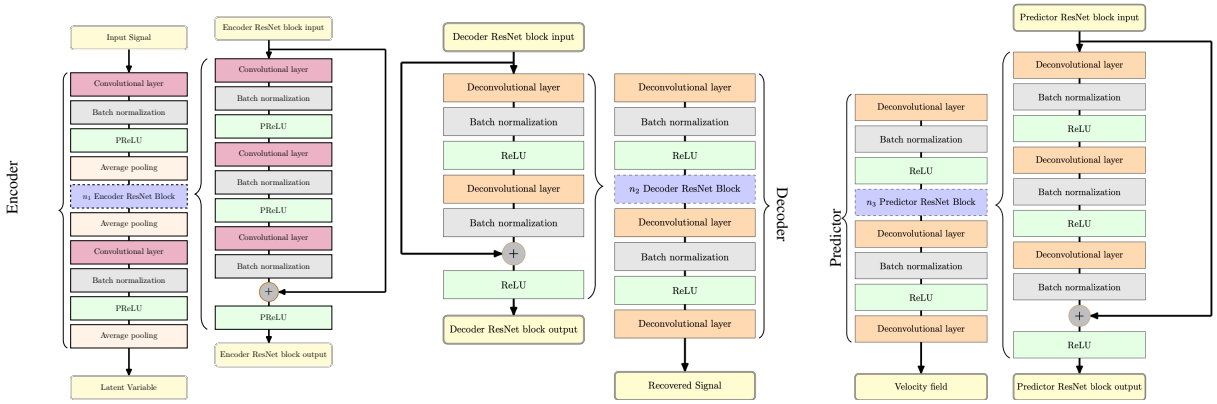


Figure B.1: Network structures of the encoder, decoder and predictor networks.

The different layers of the networks are all standard as indicated by their names. In our implementation, the input of the neural network is the a $N_t \times N_d \times N_s$ tensor representing the solution of (4.22) for N_s sources, at N_d detector points $\{\mathbf{x}_d\}_{d=1}^{N_d}$, and on N_t time instances $\{t_j\}_{j=1}^{N_t}$: $u(t_i, \mathbf{x}_j; h_s)$, $i = 1, \cdots, N_t$, $j = 1, \cdots, N_d$, and $s = 1, \cdots, N_s$. The network outputs the recovered input (from the decoder and the reconstructed velocity field from the predictor. When the output velocity field is represented with the Fourier basis, the output of the predictor is an $M \times M$ matrix whose ij-element is $\mathfrak{m}(k_i, k_j)$ $(0 \le k_i, k_j \le M)$.

Besides the sizes of the network input (that is, the input of the encoder) and the network output (that is, the output of the predictor), the key parameters of the overall network are: (i) the size of the latent variables, and (ii) the number of ResNet blocks in each of the sub-networks (n_1, n_2

and n_3). In our implementation, we tested the network structure with different numbers of ResNet blocks. The training results are not sensitive to the selection of such numbers (which controls the size of the overall network). In the numerical simulations presented in this chapter, we use $n_1 = 10$, $n_2 = 5$ and $n_3 = 10$. The computational code we used for the numerical simulations in this chapter, implemented using Python, are deposited at https://github.com/wending1/FWI_Deep_Learning.

B.2 Neural Network Training

The network training is achieved with the Adam optimizer [71]. The learning rate is initially set to be 5×10^{-4}, and decays by a factor of 1.2 for every 5 epoch. The batch size is chosen to be 128. We stop the training after 50 epoch.

The detail structures of the encoder (left), decoder (middle) and predictor (right) are illustrated in Figure B.1. The encoder includes convolutional layers [76], batch normalization layers [65], Mish activation layers [102], average pooling layers [87], and n_1 = encoder ResNet blocks [61]. See left diagram of Figure B.1 for the detailed structure. The decoder contains de-convolutional layers, batch normalization layers, Mish layers, average pooling layers, and n_2 decoder ResNet blocks. The predictor includes convolutional layers, batch normalization layers, Mish activation layers, average pooling layers, and n_3 predictor ResNet blocks as well as fully connected layer to make velocity field prediction. The middle and right diagrams of Figure B.1 present the detailed structures of the decoder and the predictor.

B.3 Loss function for training neural networks

Given training data set $\{g_i\}_{i=1}^N$ corresponding velocity $\{m_i\}_{i=1}^N$ and trainable parameters set α. Denote the encoder, decoder, predictor as $E_\alpha, D_\alpha, P_\alpha$. The neural network utilizes different loss functions (l_1, l_2) for encoder-decoderand velocity prediction. The total training loss is the sum of

two individual losses

$$\text{Loss}(\alpha) := \frac{1}{N}\sum_{i=1}^{N} l_1(D_\alpha(E_\alpha(g_i)), g_i) + l_2(P_\alpha(E_\alpha(g_i))), m_i) \tag{B.1}$$

- g_i is the input datum; $D_\alpha(E_\alpha(g_i))$ is the recovered datum.
- m_i is the true velocity field that generated g_i by 4.22.
- $P_\alpha(E_\alpha(g_i))$ is the predicted velocity field; $P \circ E$ is the pre-conditioner $\widehat{f}_{\widehat{\alpha}}^{-1}$.
- $l_1(x, \tilde{x}) = \|x - \tilde{x}\|_1$, is the L_2 loss between input datum and recovered datum.
- $l_2(x, \tilde{x}) = \|x - \tilde{x}\|_2$ is the L_2 loss between predicted and actual velocity field/parameters.

B.4 Adjoint state gradient calculation

We summarize here the calculation of the Fréchet derive of the objective function $\Phi(m)$ defined in (4.20) with respect to the velocity field m.

Following Proposition 4.5.1, the Fréchet differentiability of the map $\mathbf{f}(m)$ with respect to m is well-established under reasonable assumptions on the smoothness of the domain, the regularity of the incident wave source h and the regularity of the velocity field m. With the assumption we have on the differentiability of the trained network $\widehat{\mathbf{f}}_{\widehat{\alpha}}^{-1}$, the Fréchet differentiability of $\Phi(m)$ in (4.20) is ensured.

To simplify the notation, we denote by

$$r(m) := \widehat{\mathbf{f}}_{\widehat{\alpha}}^{-1}(\mathbf{f}(m)) - \widehat{\mathbf{f}}_{\widehat{\alpha}}^{-1}(\mathbf{g}^\delta) \tag{B.2}$$

the data residual and $\Gamma \subset \Omega$ the subset of the boundary of the domain where acoustic data are measured. To be concrete, we take the regularization functional to be the $\mathcal{H}^1$ semi-norm of the unknown m. We also assume that the velocity is known on the boundary of the domain so that

the perturbation $\delta m|_{\partial\Omega} = 0$. This assumption simplified the calculations below but is by no means essential.

Taking the derivative of $\Phi(m)$ with respect to m in the direction δm, we have

$$\Phi'(m)[\delta m] = \int_\Omega r(m)\big(\widehat{\mathbf{f}}_{\widehat{\alpha}}^{-1}\big)'(\mathbf{f}(m))\Big[\mathbf{f}'(m)[\delta m]\Big]d\mathbf{x} + \gamma\int_\Omega \nabla m\cdot\nabla\delta m\, d\mathbf{x}\,. \tag{B.3}$$

Let $\big(\widehat{\mathbf{f}}_{\widehat{\alpha}}^{-1}\big)'^{*} : L^2(\Omega) \mapsto L^2((0,T]\times\Gamma)$ be the adjoint of the operator $\big(\widehat{\mathbf{f}}_{\widehat{\alpha}}^{-1}\big)'(\mathbf{f}(m))$. Using the assumption that $\delta m|_{\partial\Omega} = 0$, we can then write the above result as

$$\Phi'(m)[\delta m] = \int_0^T\int_\Gamma \big(\widehat{f}_{\widehat{\alpha}}^{-1}\big)'^{*}[r(m)]\mathbf{f}'(m)[\delta m]\, dS(\mathbf{x})dt - \gamma\int_\Omega \Big(\Delta m\Big)\delta m\, d\mathbf{x}\,. \tag{B.4}$$

This can be written into the following form with the adjoint operator of $\mathbf{f}'(m)$, $\mathbf{f}'^{*} : L^2((0,T]\times\Gamma) \mapsto L^2(\Omega)$:

$$\Phi'(m)[\delta m] = \int_\Omega \mathbf{f}'^{*}\Big[\big(\widehat{\mathbf{f}}_{\widehat{\alpha}}^{-1}\big)'^{*}[r(m)]\Big]\delta m\, d\mathbf{x} - \gamma\int_\Omega \Big(\Delta m\Big)\delta m\, d\mathbf{x}\,. \tag{B.5}$$

The adjoint operator $\mathbf{f}'^{*}$ can be found in the standard way. We document the calculation for the specific two-dimensional setup we have as follows.

For the wave equation (4.22), we can formally differentiate u with respect to m to have that u' solves

$$\begin{aligned}
\frac{1}{m^2}\frac{\partial^2 u'}{\partial^2 t} - \Delta u' &= 2\frac{\delta m}{m^3}\frac{\partial^2 u}{\partial^2 t}, && \text{in } (0,T]\times\Omega\\
u'(0,x,z) = \frac{\partial u'}{\partial t}(0,x,z) &= 0, && (x,z)\in(0,L)\times(-H,0)\\
u'(t,0,z) &= u'(t,L,z), && (t,z)\in(0,T]\times(-H,0)\\
\frac{\partial u'}{\partial z}(t,x,-H) &= 0, && (t,x)\in(0,T]\times(0,L)\\
\frac{\partial u'}{\partial z}(t,x,0) &= 0, && (t,x)\in(0,T]\times(0,L)
\end{aligned} \tag{B.6}$$

Let us define the adjoint problem

$$\begin{aligned}
\frac{1}{m^2}\frac{\partial^2 w}{\partial^2 t} - \Delta w &= 0, && \text{in } (0,T]\times\Omega \\
w(T,x,z) = \frac{\partial w}{\partial t}(T,x,z) &= 0, && (x,z)\in(0,L)\times(-H,0) \\
w(t,0,z) = w(t,L,z) &= 0, && (t,z)\in(0,T]\times(-H,0) \\
\frac{\partial w}{\partial x}(t,0,z) + \frac{\partial w}{\partial x}(t,L,z) &= 0, && (t,z)\in(0,T]\times(-H,0) \\
\frac{\partial w}{\partial z}(t,x,-H) &= 0, && (t,x)\in(0,T]\times(0,L) \\
\frac{\partial w}{\partial z}(t,x,0) &= \big(\widehat{\mathbf{f}}_{\widehat{\alpha}}^{-1}\big)'^{*}[r(m)], && (t,x)\in(0,T]\times(0,L)
\end{aligned} \tag{B.7}$$

We can then multiply the equation for u' by w and the equation for w by u' and use integration by part to show that

$$\Phi'(m)[\delta m] = -\int_\Omega \frac{2}{m^3}\Big(\int_0^T \frac{\partial w}{\partial t}\frac{\partial u}{\partial t}dt\Big)\delta m\, d\mathbf{x} - \gamma\int_\Omega \Big(\Delta m\Big)\delta m\, d\mathbf{x}\,. \tag{B.8}$$

When the data in the inversion are collected from N_s different incoming sources $\{h_s\}_{s=1}^{N_s}$, the forward map $\mathbf{f}(m)$ and the data $\mathbf{g}^\delta$ defined in (4.14). Let u_s $(1\le s\le N_s)$ be solution to (4.22) with source h_s, and w_s be the solution to the adjoint equation (B.7) with the s-th component of $\big(\widehat{\mathbf{f}}_{\widehat{\alpha}}^{-1}\big)'^{*}[\mathbf{r}(m)]$, here

$$\mathbf{r}(m) = \widehat{\mathbf{f}}_{\widehat{\alpha}}^{-1}(\mathbf{f}(m)) - \widehat{\mathbf{f}}_{\widehat{\alpha}}^{-1}(\mathbf{g}^\delta), \tag{B.9}$$

then derivative of $\Phi(m)$ can be computed as

$$\Phi'(m)[\delta m] = -\int_\Omega \frac{2}{m^3}\Big(\sum_{s=1}^{N_s}\int_0^T \frac{\partial w_s}{\partial t}\frac{\partial u_s}{\partial t}dt\Big)\delta m\, d\mathbf{x} - \gamma\int_\Omega \Big(\Delta m\Big)\delta m\, d\mathbf{x}\,. \tag{B.10}$$

The computational procedure is summarized in Algorithm 1. The main difference between the calculation here and the adjoint calculation for a standard FWI gradient calculation is that we need to use the network $\widehat{\mathbf{f}}_{\widehat{\alpha}}^{-1}$ to backpropagate the data into the velocity field in Line 5 of Algorithm 1 to compute the residual, and then use the adjoint of the network operator (transpose of the gradient in the discrete case), $\big(\widehat{\mathbf{f}}_{\widehat{\alpha}}^{-1}\big)'^{*}$, to map the residual $\mathbf{r}(m)$ to the source of the adjoint wave equation in

Algorithm 1 Gradient Calculation with Adjoint State

1: **for** $s = 1$ to N_s **do**
2: Solve (4.22) with h_s for u_s
3: Evaluate the $\mathbf{f}(m; h_s)$ component of $\mathbf{f}(m)$
4: Evaluate $\mathbf{r}(m)$ according to (B.9) with the network $\widehat{\mathbf{f}}_{\widehat{\alpha}}^{-1}$
5: Evaluate $\left(\widehat{\mathbf{f}}_{\widehat{\alpha}}^{-1}\right)'^{*}[\mathbf{r}(m)]$ with the neural network
6: **for** $s = 1$ to N_s **do**
7: Solve (B.7) with the s-th component of $\left(\widehat{\mathbf{f}}_{\widehat{\alpha}}^{-1}\right)'^{*}[\mathbf{r}(m)]$ as the source term for w_s
8: Evaluate $\Phi'(m)$ according to (B.10)

Line 6.

B.5 Inversion with truncated Neumann series

The truncated Neumann series reconstruction (4.11) can be implemented with only the forward wave simulation and the learned neural network (without the need for the gradient of the learned operator). Let us define

$$m_0 := \widehat{\mathbf{f}}_{\widehat{\alpha}}^{-1}(\mathbf{g}^{\delta}), \quad R_J := \sum_{j=0}^{J-1} K^j\big(m_0\big)\,,$$

with K defined in (4.11). We can then verify that

$$R_{J+1}(m_0) = (I + K\sum_{j=0}^{J-1} K^j)\big(m_0\big) = m_0 + KR_J(m_0) = m_0 + R_J(m_0) - \widehat{\mathbf{f}}_{\widehat{\alpha}}^{-1}(\mathbf{f}(R_J(m_0)))\,. \quad \text{(B.11)}$$

This leads to the computational procedure summarized in Algorithm 2. The main difference between

Algorithm 2 Reconstruction with J-Term Truncated Neumann Series

1: Evaluate $m_0 := \widehat{\mathbf{f}}_{\widehat{\alpha}}^{-1}(\mathbf{g}^{\delta})$ with the learned neural network
2: Set $m \leftarrow m_0$;
3: **for** $j = 1$ to $j = J - 1$ **do**
4: **for** $s = 1$ to N_s **do**
5: Solve (4.22) with (m, h_s) for u_s
6: Evaluate the $\mathbf{f}(m; h_s)$ component of $\mathbf{f}(m)$
7: Update $m \leftarrow m_0 + m - \widehat{\mathbf{f}}_{\widehat{\alpha}}^{-1}(\mathbf{f}(m))$

the calculation here and the adjoint calculation for a standard FWI gradient calculation is that we

only need to evaluate the network $\widehat{\mathbf{f}}_{\widehat{\alpha}}^{-1}$ to project the data back into the velocity field to compute the residual $m_0 - \widehat{\mathbf{f}}_{\widehat{\alpha}}^{-1}(\mathbf{f}(m))$, and update the current result m.