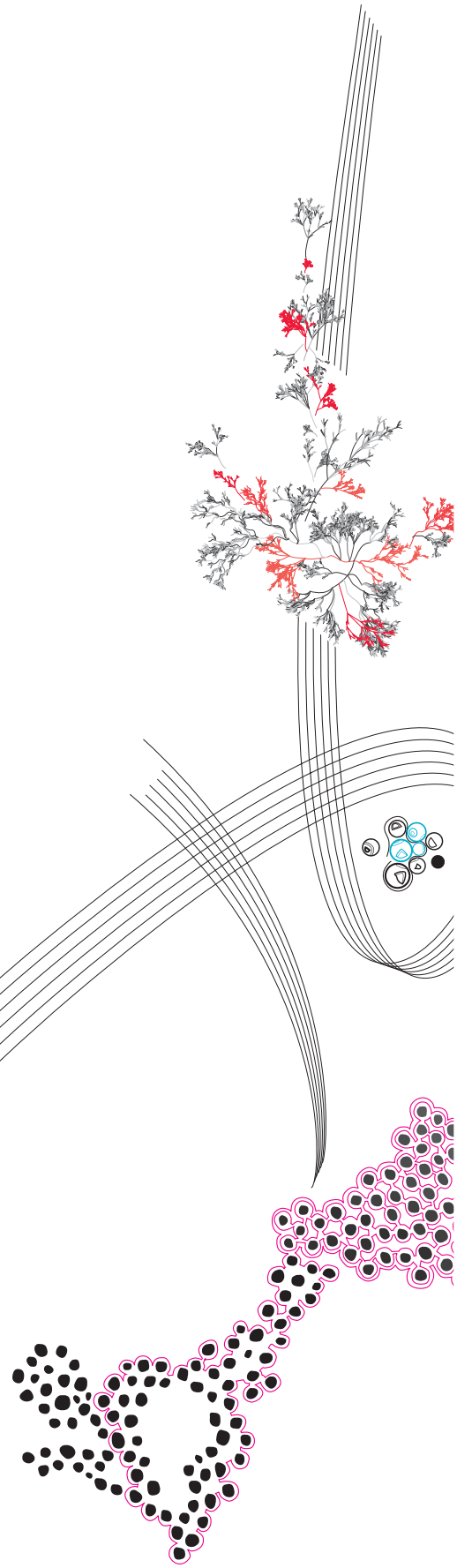


MSc Thesis Applied Mathematics

Learning Distributionally Robust Solutions for Inverse Problems using the Wasserstein Distance

Floor van Maarschalkerwaart

Graduation Committee:
Prof. Dr. Christoph Brune
Dr. Marcello Carioni
Dr. Matthias Schlottbom

July 23, 2024

Department of Applied Mathematics
Faculty of Electrical Engineering,
Mathematics and Computer Science

UNIVERSITY OF TWENTE.

Acknowledgement

Before you lies my Master Thesis, a project I have dedicated the past six and a half months to. This thesis combines a topic I am already quite familiar with (inverse problems) and subjects that were entirely new to me (distributionally robust optimization and optimal transport). Throughout this project, I have significantly expanded my theoretical knowledge in these areas and deepened my understanding of foundational topics such as measure theory and duality. Unlike my previous projects, which were predominantly numerical, this thesis is heavily theoretical. This theoretical emphasis has been both the most rewarding and the most challenging aspect of my work. Most importantly, I have thoroughly enjoyed the opportunity to dive so deeply into a topic and conducting my own research.

I would like to express my sincere gratitude to my entire graduation committee for their supervision and help over these past months. Your unwavering confidence in me and my project often provided the encouragement I needed to regain my own confidence in my research during challenging times. I am very proud of the final result, which I could not have achieved without your guidance.

Marcello, thank you for being accessible and willing to meet on short notice. Your explanations and insights were invaluable, often helping me to finalize the proofs in this thesis. Christoph, I am grateful for your frequent availability despite your busy schedule. Your ability to draw connections to other research projects and maintain a high-level view of the research and its context was immensely helpful. Lastly, I want to thank Matthias for joining my graduation committee, taking the time to read my report and providing critical questions and feedback.

I hope you enjoy reading my thesis.

Floor van Maarschalkerwaart
July 23, 2024

Learning Distributionally Robust Solutions for Inverse Problems using the Wasserstein Distance

Floor van Maarschalkerwaart

July 23, 2024

Abstract

This thesis proposes a novel data-driven framework that integrates Wasserstein robustness into inverse problem modeling. It aims to bridge the research fields of inverse problems and Wasserstein robustness, providing insights into the relationship between regularization and robustness, which are critical for developing stable and reliable solutions in various applications. We introduce a general framework to find distributionally robust solutions and prove a new strong duality result that can be modified to be applied in many fields. For an academic impact case, the dual representation of an inverse problem robust to Gaussian noise in the measurement space is further explored and reduced to a convex, finite-dimensional problem, making it computationally tractable. The framework is validated through numerical simulations, demonstrating that it can learn solutions for inverse problems that are robust to perturbations in the Wasserstein distance. This work expands the theoretical foundations of both distributionally robust optimization and inverse problems and is applicable to various other types of problems. The developed framework holds promise for practical applications in diverse fields, including more complex inverse problems and higher-dimensional applications.

Keywords: Wasserstein distance, robustness, distributionally robust optimization, optimization, duality, inverse problems, regularization, probability measures, conditional shifts, proximal operator

Contents

1 Notation and definitions

$\mathbb{R}_+ := \{x \in \mathbb{R} : x \geq 0\}$	Set of non-negative real numbers
$\bar{\mathbb{R}} := \mathbb{R} \cup \{\infty\}$	Extended real number line
$\bar{\mathbb{R}}_+ := \mathbb{R}_+ \cup \{\infty\}$	Set of non-negative real numbers including infinity
$M(\Omega)$	Space of finite Borel measures on Ω
$C_b(\Omega)$	Space of continuous bounded functions on Ω
$P(\Omega)$	Space of probability measures on Ω
$\mathcal{P}(\Omega)$	Power set of Ω
$\mathcal{B}(\Omega)$	Borel algebra of Ω
$m_{\mathcal{U}}(\Omega; \mathbb{R})$	Collection of measurable functions $\phi : (\Omega, \mathcal{U}(\Omega)) \to (\bar{\mathbb{R}}, \mathcal{B}(\bar{\mathbb{R}}))$, where $\mathcal{U}(\Omega) = \cap_{\mu \in P(\Omega)} \mathcal{B}_\mu(S)$ is the universal σ-algebra
$[f, A]$	Epigraph of a function f over a set A
$\Pi(\mu_1, \mu_2)$	Set of joint probability measures with first marginal $\mu_1 \in P(\Omega_1)$ and second marginal $\mu_2 \in P(\Omega_2)$
μ^*	Upper script on a probability measure denotes a fixed probability measure

2 Introduction

Inverse problems are present across all fields where there is a necessity to reconstruct data from (ill-posed) measurements. Examples include engineering (e.g. electromagnetic waves, elastodynamics), medical imaging (e.g. CT, MRI, X-ray), wavefield imaging (e.g. ultrasound) and data assimilation (e.g. numerical weather prediction) [1]. Solving inverse problems is often challenging due to noisy measurements and the ill-posed nature of the problems (unstable, no or several solutions). There is well-established classical theory on inverse problems guaranteeing mathematically that a solution obtained by classical methods actually solves the inverse problem. For more detailed notes on inverse problem, we refer to [2]. Unfortunately these methods still have some limitations. The forward models used are approximations of reality, accurate models have a (too) high numerical complexity and the inputs do not cover the full model parameter space [3]. Consequently, these models struggle to capture the highly specific structures in data [3], leading to suboptimal results across different applications.

Recently, data-driven methods have demonstrated significant success in solving inverse problems. These are generic models that learn solutions for specific problems through training data, thus being easily applicable to many different datasets from different fields [3]. However, a major drawback of data-driven approaches is their lack of interpretability, explainability and general theoretical guarantees. This thesis seeks to bridge this gap by relating data-driven methods for inverse problems with classical approaches, thereby enhancing their explainability and interpretability. Specifically, we will explore Wasserstein robustness in inverse problems and aim to draw insights into its relationship with classical regularization methods.

Inverse problems Usually, we work with a model that transforms certain inputs into outputs through a well-defined process. This model could be anything: a medical imaging device, a weather prediction model or something as simple as a matrix operating on the input. In mathematical terms, we describe this process as a forward problem. Here, we know the input $x \in X$ and the model, represented by a forward operator $H : X \to Y$, processes this input to produce an output $y = Hx \in Y$. The real challenge arises when you face an inverse problem. Instead of a known input, you begin with the output, which is often noisy:

$$y = Hx + \mathrm{e} \in Y.$$

In this equation, e represents the noise that corrupts the measurement. The goal is to work backward to reconstruct the unknown input x. For example, in medical imaging, we obtain a CT-scan of a patient, which gives us a measurement y. However, our objective is to reconstruct the image of the person, represented by x.

Inverse problems are tricky for several reasons. First, the noise can corrupt the measurement. Second, these problems are often ill-posed, meaning that small changes in the noisy output can lead to significant variations in the reconstructed input and there might be multiple (or no) possible inputs that could lead to the same output. To combat these difficulties, there are various classical regularization techniques. These methods help to stabilize the solutions, allowing us to extract meaningful information from noisy, incomplete, or indirect measurements. Examples are truncated SVD [4], total variation (TV) [5] and Tikhonov [6] regularization. Solutions to inverse problems play a crucial role in various scientific and engineering applications such as medical imaging techniques, enhancing signal processing, or refining data analysis.

With the rise of deep learning models, data-driven approaches to solving inverse problems have gained prominence. One such approach is the Bayesian framework, which provides a method for addressing the uncertainties inherent in inverse problems. In this framework the possible input x and its corresponding measurement y are viewed as realizations of X- and Y-valued random variables $\mathbf{x}$ and $\mathbf{y}$ respectively. The objective of Bayesian inversion is to characterize the full posterior distribution p_{post} of $\mathbf{x}$ given $\mathbf{y}$ by using Bayes' theorem:

$$p_{\mathrm{post}}(x|y) = \frac{1}{Z(y)} p_{\mathrm{w}}(y - Hx) p_0(x).$$

Here, p_0 is the prior probability density on $\mathbf{x}$, reflecting any prior knowledge or assumptions about the input. $Z(y)$ is a normalizing constant ensuring that the posterior distribution is properly scaled. The data likelihood is specified through the noise distribution p_{w} and the forward operator H which together describe how the input x is transformed into the measurement y and how noise affects this process. Bayesian inversion allows for a probabilistic interpretation of the solution. This probabilistic framework can be further refined and made robust. One such technique involves assuming that the true distribution is close to the perturbed measure in terms of the Wasserstein distance. This allows for the establishment of a robust estimation framework against perturbations on probability measures defined over the joint input and measurement spaces. This Bayesian perspective can be incorporated into inverse problems by setting up a Distributionally Robust Optimization framework.

Distributionally Robust Optimization and the Wasserstein distance Numerous problems across various fields are impacted by uncertain parameters, which can only be estimated indirectly through sample observations. Data-driven methods aim to learn a solution from a finite number of training samples but an often-occurring problem is that it is difficult to find a solution that will perform well on unseen test samples. A Distributionally Robust Optimization (DRO) [7] problem minimizes the expected risk of a problem under uncertainty where the true distribution of the data is usually unknown but can be approximated by an observed distribution. This observed distribution however, often leads to a low out-of-sample performance. Therefore, in a DRO we assume that the real distribution of the data must be 'close' to the observed distribution and choose an ambiguity set of distributions 'close' to the observed distribution. We then find the solution with minimal risk under the worst cases allowed within the ambiguity set. In other words, it finds the 'best worst-case' scenario. This robust solution then has guaranteed low out-of-sample error: it generalizes well to new data. In order to formulate a DRO problem, a metric is needed that measures the distance between probability measures - to determine which measures are 'close' to the observed one. For this, the Wasserstein distance is often chosen as a suitable metric. The Wasserstein distance is well-defined for any pair of probability measures, regardless of any mutual singularity - unlike ϕ-divergences - and it is sensitive to the relative position of the supports of singular distributions being compared, thus being less sensitive to vanishing gradient problems - unlike integral probability measures (IPMs) [8].

Aim of thesis In the previous paragraph, it is mentioned that robust solutions generalize well to new data. This generalizing effect is similar to a regularizing effect we have seen in machine learning and inverse problems. Thus the question arises, is there more than just similarity to these methods? What happens when we want to robustify an inverse problem? In specific cases, could the robust and regularized problems be equivalent? The overarching goal of this thesis is to address these questions. It aims to bridge the topics introduced and provide insights into the relationship between robustness and regularization for inverse problems. Can we learn solutions for inverse problems that are robust with respect to perturbations in the Wasserstein distance (measurement space, ground-truth space or both)? How well does this learned solution perform (with respect to classical regularization methods)? What do we gain with respect to classical regularization methods? The fields of Wasserstein robustness and data-driven approaches for inverse problems have both seen considerable growth in recent years. The challenge in this thesis is how to connect the two topics and to determine what we gain with the Wasserstein robust optimization framework with respect to classical regularization methods.

2.1 Related works

Inspiration Recently, Daniel Kuhn has made significant contributions to the field of Wasserstein-DRO. Notably, Kuhn et al. [7] present a comprehensive overview of Wasserstein-DRO with applications in machine learning, while Carioni et al. [8] provide a review on using Wasserstein-DRO as a data-driven method for solving inverse problems, including an extensive overview of the mathematical results underlying the mentioned methods. Both works serve as the main inspiration for this thesis.

Data-driven approaches for inverse problems The literature on data-driven approaches for inverse problems has seen considerable growth in recent years. For surveys on this subject see [3, 9–11]. The work in [12] compares the performance of two classical and two data-driven approaches. The increased use of neural networks (in combination with a classical regularization term) as (adversarial)

regularizers for inverse problems is highlighted in works such as [13–18]. For a survey and a review on neural networks as regularizers, see [10] and [11] respectively. Additionally, neural networks can be used to learn the regularization parameters [19] or an auxiliary network to generate adversarial examples can be used to improve the robustness of deep-learning based approaches for inverse problems [20]. Another data-driven approach for inverse problems, proposed in [21] is to find an optimal low-rank regularized inverse matrix by using the training data in a Bayesian risk minimization framework. The results in [22] build upon this by combining data-driven and model-based methods in Bayesian inverse problems in a learned-SVD regularization method, showing a connection between Tikhonov and learned-SVD regularization.

(Wasserstein) DRO The DRO framework is used in many applications, for a review on DRO in general see [23]. We are mainly interested in its applications in machine learning and inverse problems. The DRO framework is used in [24] to learn models that are robust to perturbations in the data distribution. Bayesian and minimax approaches to solve the minimum MSE estimation problem for inverse problems are combined by setting them in the Wasserstein DRO framework in [25] while [26] uses a spectral decomposition method to solve the Wasserstein DRO problem for Gaussian process regression and Bayesian linear inverse problems. The Wasserstein distance itself finds applications in various aspects of inverse problems, serving as a regularization term [27, 28], appearing in a proximal gradient method [29] or as a loss function [30].

DRO and regularization Connections between DRO and regularization methods have been explored extensively. According to Shafieezadeh-Abadeh et al. [31], the first connection between robustification and regularization is found in [32]. Specifically, they show that the DRO problem with a *Frobenius norm*-uncertainty set is equivalent to Tikhonov regularization. In [33], an overview of the conditions under which robustification and regularization are equivalent can be found, see Table 2 on page 15 for a summary. Specific connections between Wasserstein DRO and regularization have been found in [31, 34, 35]. Wasserstein DRO as a regularization technique for linear regression is introduced in [31], guaranteeing an upper confidence bound on the loss on test data and proving the regularized learning problems are tractable. They show that classical regularized learning models are special cases of their proposed framework. A new concept, the 'variation of a function' is introduced in [34] as a new form of regularization that generalizes Total-Variation, Lipschitz and gradient regularization to provide a general connection between Wasserstein DRO and regularization. Their results show that Wasserstein DRO is closely related to a 'variation regularization' problem.

Connections It is evident that the fields of Wasserstein DRO and data-driven approaches for inverse problems are fast-growing and there are already some connections between the two. Notable approaches connecting machine learning and DRO are Distributionally Robust classification, Distributionally Robust regression, Distributionally Robust Maximum Likelihood Estimation, Distributionally Robust Minimum Mean Square Error Estimation [7], Wasserstein Generative Adversarial Networks (W-GANs) and adversarial regularization [8]. Specifically, Wasserstein robustness has been incorporated in inverse problems before [26], numerically showing that the method holds promise. Ultimately, there are currently not many theoretical and general results on the subject. Connections between (Wasserstein-)DRO and regularization have been established in linear regression problems [31–33, 35] and in general [34], but not fully incorporating the possible ill-posedness and non-linearity of an inverse problem and often only looking at robustness in either the input- or output-distributions instead of considering perturbations in the joint or conditional distribution. This thesis aims to fill this gap and extend the theory on Wasserstein robustness as a regularization method to inverse problems.

2.2 Approach and main contributions

To connect the two main topics of this thesis (inverse problems on the one hand and Wasserstein robustness on the other), we propose a framework to find distributionally robust solutions for inverse problems by building on an existing idea for a framework and combining concepts from optimal transport and inverse problems in a novel way. We formulate a Bayesian (i.e. conditional) framework for inverse problems as a DRO-problem. We explore the characteristics of this framework by presenting a dual representation for the problem and a general strong duality result which allows us to see some

connections to regularization. Subsequently, a specific inverse problem case is chosen, assuming a conditional setting with Gaussian noise in the measurement process. We continue our exploration of the proposed framework by exploring the dual problem for this specific scenario and reducing it to a convex, finite-dimensional problem in order to verify and validate the model through numerical simulations which suggests the Wasserstein-DRO solution coincides with the least-squares solution. This research presents a significant advancement in distributionally robust optimization and inverse problems by integrating Wasserstein robustness in inverse problem modeling. Through thorough analysis, this work not only expands theoretical foundations but also holds promise to significantly enhance the reliability of solutions to inverse problems across various fields in practice.

3 Theoretical framework

This section outlines the theoretical framework for this thesis, mainly stemming from two key research areas: distributionally robust optimization and inverse problems. To ensure the thesis is self-contained, we begin by establishing foundational concepts related to (probability) measures. Following this, we delve into the theory of Wasserstein distance and optimal transport, before exploring the specifics of distributionally robust optimization. Finally, we address the theory of inverse problems.

3.1 Measure theory

Measure theory extends concepts such as the length of an interval on the real line to more complex and abstract subsets, enabling us to apply various notions of length to suit specific problems. Before we define what a measure is, we will introduce some essential concepts that form the foundation for defining measures. For a comprehensive introduction to measure theory, we refer to [36].

Definition 3.1 (Power set)**.** The power set of a set Ω, denoted by $\mathcal{P}(\Omega)$, is the set of all subsets of Ω, including the empty set and Ω.

Definition 3.2 (σ-algebra)**.** We define a σ-algebra on Ω as a special set of subsets of Ω. The set $\mathcal{F} \subseteq \mathcal{P}(\Omega)$ is called a σ- algebra if it satisfies the following properties:

(a) Empty and full set: $\emptyset, \Omega \in \mathcal{F}$.

(b) Closed under complement: if $A \in \mathcal{F}$, then $A^c := \Omega \backslash A \in \mathcal{F}$.

(c) Closed under countable unions: if $A_i \in \mathcal{F}, i \in \mathbb{N}$ then $\cup_{i=1}^{\infty} A_i \in \mathcal{F}$.

It follows from these properties that any σ-algebra is also closed under intersection.

Definition 3.3 (Measurable set)**.** Let $\mathcal{F}$ be any σ-algebra on Ω. Then any $A \in \mathcal{F}$ is called a $\mathcal{F}$-measurable set.

Definition 3.4 (Generated σ-algebra)**.** Let $B \subseteq \mathcal{P}(\Omega)$ be any collection of subsets of a set Ω. We define the σ-algebra generated by B as the smallest σ-algebra that contains B:

$$\sigma(B) := \cap\{\mathcal{F} \subset \mathcal{P}(\Omega) : B \subset \mathcal{F}, \mathcal{F} \text{ is a } \sigma\text{-algebra}\}.$$

This set is nonempty as $\mathcal{P}(\Omega)$ is a σ-algebra that contains B and an intersection of σ-algebras is itself a σ-algebra.

Definition 3.5 (Borel σ-algebra)**.** Let (Ω, τ) be a topological space. We define the Borel σ-algebra of Ω as the set generated by the collection τ of open set on Ω:

$$\mathcal{B}(\Omega) := \sigma(\tau).$$

In other words, it is the smallest σ-algebra that contains all open sets of Ω.

Finally, we have all ingredients to give the definition of a measure. We define measures on σ-algebras as we want measures to satisfy certain properties that cannot in general be satisfied on the whole power set, but they can be fulfilled on a (Borel) σ-algebra.

Definition 3.6 (Measure and measure space)**.** Let $\mathcal{F}$ be a σ-algebra of Ω and $(\Omega, \mathcal{F})$ a measurable space. A map $\mu : \mathcal{F} \to [0, \infty]$ is called a measure if it satisfies the following properties:

(a) The empty set maps to zero: $\mu(\emptyset) = 0$.

(b) Non-negativity: $\mu(A) \geq 0 \quad \forall A \in \mathcal{F}$.

(c) Countable additivity or σ-additive: if $A_1, A_2, ... \in \mathcal{F}$ are all disjoint (i.e. $A_i \cap A_j = \emptyset$ whenever $i \neq j$ for all $A_i \in \mathcal{F}$), then $\mu\left(\cup_{i=1}^{\infty} A_i\right) = \sum_{i=1}^{\infty} \mu(A_i)$.

If μ is a measure on $(\Omega, \mathcal{F})$, then we call $(\Omega, \mathcal{F}, \mu)$ a measure space.

In the special case that $\mu(\Omega) = 1$, we call μ a probability measure and $(\Omega, \mathcal{F}, \mu)$ a probability space. This brings us to the next section on probability measures.

3.2 Probability measures

This section expands the theory on measures to probability measures specifically. The contents of this section are heavily informed by [8].

Definition 3.7 (Probability space)**.** A probability space consists of the triplet $(\Omega, \mathcal{F}, \mu)$ providing a formal model of a random process. It contains:

- The sample space Ω, the space of all possible outcomes. We will assume Ω to be a Polish space which is characterized by being a complete metric space with an underlying metric $d : \Omega \times \Omega \to [0, \infty]$ and possessing a countable dense subset.
- An event space $\mathcal{F}$ which is a σ-algebra consisting of subsets of Ω, a collection of all events we want to include where an event is a set of outcomes.
- A probability measure $\mu : \mathcal{F} \to [0, 1]$ which assigns a probability to each event in the event space. Let $P(\Omega)$ denote the set of all possible probability measures on the measurable space $(\Omega, \mathcal{F})$.

Definition 3.8 (Probability law)**.** For any $\mathbb{R}^d$-valued random variable $\mathbf{x}$ on a probability space, the corresponding probability law is defined as the following probability measure $\mu_{\mathbf{x}}$ on $(\mathbb{R}^d, \mathcal{B}(\mathbb{R}^d))$, where $\mathcal{B}(\mathbb{R}^d)$ denotes the Borel σ-algebra of $\mathbb{R}^d$:

$$\mu_{\mathbf{x}}(\mathscr{B}) := \mu(\mathbf{x}^{-1}(\mathscr{B})) \quad \forall \mathscr{B} \in \mathcal{B}(\mathbb{R}^d).$$

Definition 3.9 (Probability density function)**.** We define the probability density function of $\mathbf{x}$ by the nonnegative function $p_{\mathbf{x}} : \mathbb{R}^d \to [0, +\infty]$ such that $\mu_{\mathbf{x}}(\mathscr{B}) = \int_{\mathscr{B}} p_{\mathbf{x}} d\lambda \quad \forall \mathscr{B} \in \mathcal{B}(\mathbb{R}^d)$ where λ is the Lebesgue measure.

The existence of $p_{\mathbf{x}}$ is guaranteed by the Radon-Nikodym theorem if $p_{\mathbf{x}}$ is absolutely continuous with respect to λ, so if $\mu_{\mathbf{x}}(\mathscr{B}) = 0$ whenever $\lambda(\mathscr{B}) = 0$ for any $\mathscr{B} \in \mathcal{B}(\mathbb{R}^d)$. If this is the case, it is unique λ-almost everywhere.

Definition 3.10 (Expected value of a random variable)**.** The expected value of a random variable $\mathbf{x}$, denoted as $\mathbb{E}_{\mathbf{x}\sim\mu_{\mathbf{x}}}[\mathbf{x}]$ or $\mathbb{E}[\mathbf{x}]$ is defined as

$$\mathbb{E}_{\mathbf{x}\sim\mu_{\mathbf{x}}}[\mathbf{x}] := \int_{\Omega} X(\omega) d\mu(\omega) = \int_{\mathbb{R}^d} x d\mu_{\mathbf{x}}(x).$$

If $\mathbf{x}$ has a density $p_{\mathbf{x}}$, the expectation can equivalently be written as

$$\mathbb{E}_{\mathbf{x}\sim\mu_{\mathbf{x}}}[\mathbf{x}] := \int_{\mathbb{R}^d} x p_{\mathbf{x}} d\lambda(x).$$

Definition 3.11 (Measurable function)**.** Any mapping $T : (\Omega_1, \mathcal{F}_1) \to (\Omega_2, \mathcal{F}_2)$ between two measurable spaces with the property that $T^{-1}(A) \in \mathcal{F}_1 \quad \forall A \in \mathcal{F}_2$ is said to be a measurable function from $(\Omega_1, \mathcal{F}_1)$ to $(\Omega_2, \mathcal{F}_2)$.

Definition 3.12 (Push-forward measure)**.** Given any any measurable function $T : (\Omega_1, \mathcal{F}_1) \to (\Omega_2, \mathcal{F}_2)$ between two measurable spaces and a probability measure μ on $(\Omega_1, \mathcal{F}_1)$. The push-forward measure of μ through T, denoted as $T_{\#\mu}$ is defined as a probability measure on $(\Omega_2, \mathcal{F}_2)$ such that

$$T_{\#\mu}(A) = \mu(T^{-1}(A)) = \mu\{x \in \Omega_1 : T(x) \in A\} \quad \forall A \in \mathcal{F}_2.$$

For a measurable function $g : (\mathbb{R}^d, \mathcal{B}(\mathbb{R}^d)) \to (\mathbb{R}^m, \mathcal{B}(\mathbb{R}^m))$, let $\mu_{\mathbf{y}} := g_{\#}\mu_{\mathbf{x}}$. The expected value of $\mathbf{y} = g(\mathbf{x})$ can be written as:

$$\mathbb{E}_{\mathbf{x}\sim\mu_{\mathbf{x}}}[g(\mathbf{x})] := \int_{\mathbb{R}^d} g(x) d\mu_{\mathbf{x}}(x) = \int_{\mathbb{R}^m} y d\mu_{\mathbf{y}}(y).$$

Definition 3.13 (Joint probability distribution)**.** We denote by $\mu(\mu_1, \mu_2)$ the set of joint probability distributions with marginals $\mu_1 \in P(\Omega_1)$ and $\mu_2 \in P(\Omega_2)$, which is defined as follows. For every $\mu \in \mu(\mu_1, \mu_2)$,

$$\mu(A \times \Omega_2) = \mu_1(A) \text{ for every Borel subset } A \subset \Omega_1 \text{ and}$$
$$\mu(\Omega_1 \times B) = \mu_2(B) \text{ for every Borel subset } B \subset \Omega_2.$$

This is equivalent to stating that for every $\phi_1 \in C(\Omega_1)$ and $\phi_2 \in C(\Omega_2)$:

$$\int_{\Omega_1 \times \Omega_2} \phi_1(x) d\mu(x, u) = \int_{\Omega_1} \phi_1(x) d\mu_1(x) \text{ and}$$
$$\int_{\Omega_1 \times \Omega_2} \phi_2(u) d\mu(x, u) = \int_{\Omega_2} \phi_2(u) d\mu_2(u).$$

Now that we have established the definitions of probability spaces, (probability) measures, random variables and their expected value, we proceed to the topics this thesis aims to connect: distributionally robust optimization and inverse problems.

3.3 Distributionally Robust Optimization (DRO)

In a distributional optimization problem, we want to minimize the expected risk of a decision problem under uncertainty. In other words, each decision has an uncertain loss, where the true distribution is usually unknown but may be indirectly observed through data samples. One could use the observed distribution, but this will always be different from the true distribution and the decision problem will inherit (often even amplify) estimation errors in the observed distribution. Furthermore, it can generally be shown that even when the input parameters of a decision problem are unbiased in their distribution, the optimization outcomes are often optimistically biased, leading to worse than expected out-of-sample results. This phenomenon is sometimes called the optimizer's curse or optimization bias. For a deeper understanding of this phenomenon, we refer to [7], which heavily informs the contents presented in this section. Additionally, the content of this section draws extensively from another relevant work, namely [8]. The optimizer's curse shows that a distributional optimization problem is not robust, as we do not have a guarantee of low out-of-sample risk. In order to have this guarantee, it is essential to quantify the sensitivity of the risk with respect to the unknown distribution. To compare distributions, we require a distance measure between them. In the next section, we will introduce the Wasserstein distance and some of its properties in order to formulate a general DRO problem.

3.3.1 Wasserstein distance and optimal transport

As mentioned, the Wasserstein distance is a metric to describe how 'close' probability measures are to each other. It is sometimes called the 'earth movers distance' as it gives the minimal amount of work (= cost) needed to transform one pile of sand (= probability measure) to another. The Wasserstein distance finds the transport plan that can transform one pile into another for the least amount of work. The smaller this amount of work is, the more similar the two measures are. Figure 1 illustrates this concept by showing to piles or probability measure to be transformed into each other.

The Wasserstein distance is a suitable metric for probability measures as it is well-defined for any pair of probability measures, regardless of any mutual singularity (i.e. it does not matter if their support overlaps), unlike ϕ-divergences. The latter are only well-defined if the distributions have overlapping support, which is often not the case when data is approximated on low-dimensional manifolds. This leads ϕ-divergences to be unstable, thus suffering from vanishing or exploding gradients. Moreover, the Wasserstein distance is sensitive to the relative position of the supports and to the geometry of distributions that are being compared, unlike integral probability measures (IPMs). Additionally, the Wasserstein distance has an intuitive interpretation as the minimum 'cost' required to transform one distribution into another. For examples of the mentioned metrics, see [8, p. 5-6].

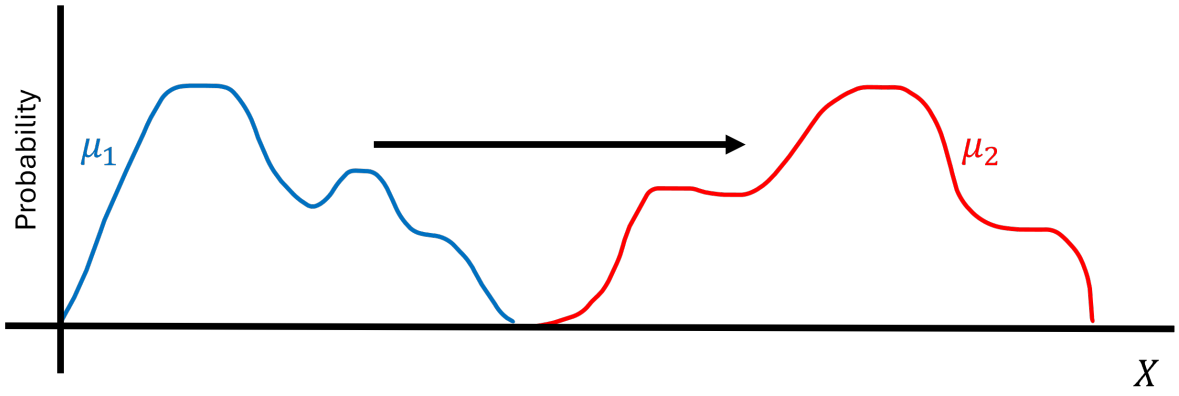


Figure 1: Two probability measures, or 'piles of sand' to be transformed into one another.

Definition 3.14 (Wasserstein distance)**.** Given a cost function $c : \Omega \times \Omega \to [0, \infty)$ satisfying the properties of a distance metric the p-Wasserstein distance ($1 \leq p < \infty$) between two Borel probability measures $\mu_1, \mu_2 \in P(\Omega)$ is defined as

$$W_p(\mu_1, \mu_2) := \left(\min_{\pi \in \Pi(\mu_1, \mu_2)} \int_{\Omega \times \Omega} c^p(x, u) d\pi(x, u) \right)^{1/p}, \quad \mu_1, \mu_2 \in P(\Omega)$$

It can be shown that the Wasserstein distance is a distance metric on the space of probability measures $P(\Omega)$, see [37, p. 106] for the proof and more properties of the Wasserstein distance. The optimization problem in (3.14) represents an infinite-dimensional linear program over the transportation plan π. This linear program admits a strong dual, offering an alternative definition of the Wasserstein distance which can be further simplified if $p = 1$.

Theorem 3.1 (Dual Kantorovich Problem)**.** *For any $p \in [1, \infty)$, the p^{th} power of the p-Wasserstein distance between π_1 and π_2 admits the dual representation*

$$W_p^p(\mu_1, \mu_2) = \sup \int_\Omega \psi(u) d\mu_2(u) - \int_\Omega \phi(x) d\mu_1(x)$$

s.t. ϕ and ψ are bounded continuous functions on Ω with

$$\psi(x) - \phi(u) \leq c^p(x, u) \quad \forall x, u \in \Omega.$$

Theorem 3.2 (Kantorovich-Rubinstein theorem)**.** *The type-1 Wasserstein distance between μ_1 and μ_2 admits the dual representation*

$$W_1(\mu_1, \mu_2) = \sup_{Lip(\phi) \leq 1} \int_\Omega \phi(x) d\mu_1(dx) - \int_{\mathbb{R}^m} \phi(u) d\mu_2(u)$$

where $Lip(\phi) = \sup_{x \neq u} \frac{|\phi(x) - \phi(u|}{c(x,u)}$ is the Lipschitz modulus of an extended real-valued function ϕ on Ω with respect to the cost $c(x, u)$.

For the proof of each theorem we refer to [38, Ch. 5] and [38, Remark 6.5], respectively.

3.3.2 DRO problem

Definition 3.15 (Wasserstein-DRO)**.** Finally, we can define the Wasserstein-distributionally robust optimization (DRO) problem as

$$\inf_{g \in \Sigma} \sup_{\mu : W_p(\mu, \mu^*) \leq \epsilon} \int_\Omega l(x; g) d\mu(x) \tag{1}$$

where we parameterize the loss function with g in some set Σ, $\int_\Omega l(x; g) d\mu$ denotes the expected total loss under the distribution μ and μ^* is the unknown true distribution.

By defining the Wasserstein ball, we can slightly rewrite (1).

Definition 3.16 (Wasserstein Ball)**.** We define the p-Wasserstein ball as

$$B_{\epsilon,p}(\mu^*) := \{\mu \in P(\Omega) : W_p(\mu, \mu^*) \leq \epsilon\}.$$

This gives an alternative formulation of the Wasserstein-DRO:

$$\inf_{g\in\Sigma} \sup_{\mu\in B_{\epsilon,p}(\mu^*)} \int_\Omega l(x; g) d\mu(x). \tag{2}$$

In [7, Th. 18-23], it is shown that for some specific cases the worst-case risk provides an upper confidence bound on the true risk and that the best worst-case (optimal value of DRO) provides an upper confidence bound on the out-of-sample performance of its optimizers. This motivates the idea that the (Wasserstein-)DRO can beat the optimizer's curse.

3.4 Inverse problems

The contents of this section draw heavily from the theory in [8, 39]. As mentioned before, inverse problems aim to approximate the unknown true solution $x^* \in X$ from its noisy and indirect measurement $y^\delta \in Y$, usually given by an additive noise model

$$y^\delta = Hx^* + e$$

where $H : X \to Y$ is a linear operator and X and Y are Banach spaces endowed by the norms $\|\cdot\|_X$ and $\|\cdot\|_Y$ respectively. Finally, e represents the combined measuring and modeling error with $\|e\|_Y \leq \delta$. Note that an inverse problem can also be modeled by $y^\delta = H^\delta x^*$ where the noise (or corruption) is not additive but incorporated in the operator $H^\delta : X \to Y$. Poisson noise for example is not additive.

Inverse problems often suffer from ill-posedness due to modeling and measurement errors. We call an inverse problem ill-posed if it is not well-posed.

Definition 3.17 (Well-posed inverse problem)**.** An equation $Hx = y$ is well-posed if all three of the following criteria hold:

- **Existence**: there exists at least one x for which $Hx = y$.
- **Uniqueness**: there is exactly one x for which $Hx = y$.
- **Stability**: the solution depends continuously on the data, i.e. there is a constant $C < \infty$ s.t. $\|x - x'\| \leq C\|y - y'\|$ where $Hx = y$ and $Hx' = y'$.

If we find an x^δ for which $Hx^\delta = y^\delta$, we can ask ourselves how big the backward error $\left\|x^\delta - x^*\right\|_X$ is with respect to the forward error $\left\|Hx^\delta - Hx^*\right\|_Y = \|e\|_Y$. In practice, we call a problem ill-posed if a small error in the data can lead to a large error in the reconstruction.

Ill-posedness causes problems in applicability of inverse problems. In the physical world, there is only one reality. If our model gives us one or multiple solutions this would yield an obvious problem: in reality there is exactly one solution representing the physical reality. How do we know which one to take? How do we make sure there even exists a solution? In many applications, well-posedness is generally not satisfied, for example in matrix inversion but also in sensitive fields such as computed tomography, where accuracy is crucial to the application of medical imaging.

To solve an inverse problem, the first elements to address are the existence and uniqueness of our solution. We cannot guarantee to recover the true solution x^* for all measurements so we introduce the concept of a generalized solution: a solution that is closest to the measured data with respect to a term measuring data discrepancy (often called the loss function) $f : Y \times Y \to \mathbb{R}_+$ such as the squared normed difference $f(Hx, y) = \|Hx - y\|_Y^2$. We look for a solution $\tilde{x}$ minimizing this data discrepancy:

$$\tilde{x} \in \tilde{X} := \arg\min_{x\in X} f(Hx, y)$$

This addresses the existence of a solution, but the operator H can still have non-trivial null-space, causing $\tilde{x}$ not to be unique. To address that problem, we choose the solution with the smallest norm. Thus we define the minimum norm solution

$$\hat{x} := \arg\min_{x \in \tilde{X}} \|x\|_X.$$

When f and $\|\cdot\|_X$ are both given by the squared L^2-norm, we call this the **least-squares minimum-norm solution**.

Now that we have established existence and uniqueness, the stability issue becomes the next challenge to address, presenting a more involved problem to resolve. If we consider inverse problems as a system of linear equations $Hx = y$ with $H \in \mathbb{R}^{m \times n}$ a given matrix of rank $k \leq \min\{m, n\}$ and $y \in \mathbb{R}^m$ the data, we can apply different notions of inverses. The generalized inverse $H^g \in \mathbb{R}^{n \times m}$ of a matrix H is any matrix satisfying $HH^gH = H$, see [40] for a book on theory and applications of generalized inverses. When a matrix has a 'regular' left-inverse (i.e. it is non-singular), this is its unique generalized inverse. Notable types of generalized inverses include the 'regular' left inverse, the right inverse H^R (s.t. $HH^R = I$) , the Drazin inverse [41] and the Moore-Penrose pseudo-inverse [42, 43]. The Drazin inverse can be applied for square matrices and is the unique matrix H^D such that $H^DHH^D = H^D$, $HH^D = H^DH$ and $H^{k+1}H^D = H^k$ with k the smallest non-negative integer such that rank(H^{k+1}) =rank(H^k). The most notable one however, is the Moore-Penrose pseudo-inverse which is defined in terms of the singular value decomposition:

$$H^\dagger := V_k \Sigma_k^{-1} U_k^T$$

where $V_k = (v_1, v_2, ..., v_k)$ and $U_k = (u_1, u_2, ..., u_k)$ contain the first k right and left singular vectors respectively and the diagonal of Σ_k contains the k largest singular values. The solution $x^\dagger := H^\dagger y$ coincides with the least-squares minimum-norm solution. This means we can guarantee the existence and uniqueness of the solution but unfortunately this solution may still be unstable. The condition number $\kappa(H) = \left\|H^\dagger\right\|\|H\|$ characterises if a matrix is ill-conditioned since

$$\frac{\left\|x^* - x^\delta\right\|_X}{\|x^*\|_X} \leq \kappa(H) \frac{\left\|y^* - y^\delta\right\|_Y}{\|y^*\|_Y}$$

with $y^* = Hx^*$ the true solution and $y^\delta = Hx^\delta$ the noisy measurement. Thus, the larger the condition number, the more ill-posed the problem is. We now have a unique solution $x^\dagger$, but the condition number $\|H\|_2\left\|H^\dagger\right\|_2 = \sigma_1/\sigma_k$ may still be large.

Alternatively, we can express the solution as

$$x^\dagger = V_k \Sigma_k^{-1} U_k^T y = \sum_{i=1}^{k} \frac{\langle u_i, y \rangle}{\sigma_i} v_i.$$

This allows us to see that the component in y corresponding to v_i is amplified by σ_i^{-1}. The discrete Picard condition is satisfied by the vector $y \in \mathbb{R}^m$ if $|\langle u_i, y \rangle|$ decays faster with i than the singular values σ_i, in which case the stability is usually not a problem.

The pseudo-inverse of a matrix and the discrete Picard condition can be extended to operators in function spaces. Assume the forward operator $H : X \to Y$ to be a bounded linear operator and that the spaces X and Y are Hilbert spaces. Denote the restriction of H to the orthogonal complement of its null space $\mathcal{N}(H)^\perp$ mapping to its range as $\tilde{H} : \mathcal{N}(H)^\perp \to \mathcal{R}(H)$. The Moore-Penrose pseudo-inverse $H^\dagger : \mathcal{R}(H) \cup \mathcal{R}(H)^\perp \to \mathcal{N}(H)^\perp$ is then defined as the unique linear extension of $\tilde{H}^{-1}$ with $\mathcal{N}(H^\dagger) = \mathcal{R}(H)^\perp$ and $\mathcal{R}(H^\dagger) = \mathcal{N}(H)^\perp$. It can be shown [39, Ch. 3.2] that if y is an element of the domain of $H^\dagger$, this pseudo-inverse gives the unique minimum-norm solution $x^\dagger = H^\dagger y$. This again gives existence and uniqueness, but no guarantees of stability (continuity).

For a *compact* operator H its pseudo-inverse can be expressed as $H^\dagger y = \sum_{i=1}^{\infty} \frac{\langle u_i, y \rangle_Y}{\sigma_i} v_i$ with σ_i the i-th singular value and u_i, v_i the i-th left and right singular vectors. The Picard condition is fulfilled if for every $y \in \mathcal{D}(H^\dagger)$,

$$\sum_{j=1}^{\infty} \frac{\left|\langle y, u_j\rangle_Y\right|^2}{\sigma_j^2} < \infty.$$

This condition tells us how stable or unstable our unique solution is, so how can we modify our solution to make sure it is stable? This is where **regularization** comes in. When applying regularization, we basically force a stable solution. For compact operators, we can regularize the pseudo-inverse to force it to be bounded (and continuous). Define the regularized pseudo-inverse as

$$H_\alpha^\dagger y = \sum_{i=0}^{\infty} g_\alpha(\sigma_i) \langle y, u_i\rangle v_i$$

where g_α describes the type of regularization that is used. For example, Tikhonov regularization uses $g_\alpha(\sigma) = \frac{\sigma}{\sigma^2+\alpha}$ and truncated SVD uses

$$g_\alpha(\sigma) = \begin{cases} \sigma^{-1}, & \text{if } \sigma > \alpha \\ 0, & \text{otherwise.} \end{cases}$$

The regularization term should provide that $H_\alpha^\dagger$ is a bounded linear operator for $\alpha > 0$ and converges pointwise to $H^\dagger$ as $\alpha \to 0$, for $y \in \mathcal{D}(H^\dagger)$.

The most popular regularization methods fall under the umbrella of variational regularization. This approach seeks to construct a family of reconstruction operators $G^\lambda : Y \to X$, parameterized by λ, such that $G^\lambda(y)$ provides a satisfactory approximation of x. The idea is to incorporate prior knowledge on the solution x into the problem. This method establishes these reconstruction maps as the solution to a variational minimization problem:

$$G^\lambda(y) \in \arg\min_{x \in X} f(Hx, y) + R_\lambda(x)$$

where $f : Y \times Y \to \mathbb{R}_+$ represents the data fidelity and $R_\lambda : X \to \mathbb{R}$ is a regularization term parameterized by λ incorporating prior information on the input. Often, the regularizer is constructed as $R_\lambda = \lambda R(x)$ with R a fixed regularizer and $\lambda \in \mathbb{R}_+$ a penalty parameter balancing data fidelity and regularization. With small λ, the emphasis lies on fitting the data and with a large λ the emphasis lies on regularization. Most classical reconstruction operators are convergent regularization schemes, where the regularization term is chosen such that $G^\lambda(y)$ varies continuously in y and there exists a parameter selection rule $\lambda : \delta \mapsto \lambda(\delta)$ such that as the noise level $\delta \to 0$, $G^{\lambda(\delta)}(y)$ converges to a generalized solution of the noiseless operator equation $y^* = Hx^*$, where y^* denotes the noise-free measurement. Popular choices are Total-Variation (TV) regularization with $R_\lambda(x) = \lambda\|\nabla x\|_1$ and generalized Tikhonov regularization with $R_\lambda(x) = \lambda\|Bx\|_Z^2$ where $B : X \to Z$ is a bounded linear operator.

4 Fundamental theorems

Although the theoretical framework has been established, there are still numerous preliminary results that we will build upon, mostly related to the DRO problem and its dual formulation. In this thesis, we will draw on findings from other notable researchers across the various fields we are integrating. This section begins by presenting a dual representation for a problem analogous to ours, though simpler, which yields a representation similar to Tikhonov regularization. We will provide a detailed proof of the strong duality of this representation, as it serves as a significant source of inspiration. Additionally, an interesting alternative proof of strong duality is given. Following this, we will give a convex, finite-dimensional reduction of the Wasserstein problem to make it computable.

4.1 Dual representation

In [35], Blanchet et al. construct a DRO formulation for the setting of linear and logistic regression models where the goal is to find the best fitting parameter vector $\beta \in \mathbb{R}^d$ relating the given data points $\{(x_i, y_i) : i = 1, ..., n\}$ with predictor variables $x_i \in \mathbb{R}^d$ and responses $y_i \in \mathbb{R}$ $\forall i$. The goal is to find the β that best fits all parameterized models relating each data-point. They show that when considering a square loss and the 2-Wasserstein distance with l_q-norm as cost function, the DRO problem can be reformulated as a problem very similar to Tikhonov-regularization. In order to discuss their results in detail, their DRO formulation, Proposition 1 and 2 and Theorem 1 [35, p. 8-11] will be repeated here along with their proofs [44] [35, p.28-29], with some changes in notation to facilitate better connection with our context in a later stage.

Blanchet et al. use a slightly more general formulation of the Wasserstein-distance, which they define as follows.

Definition 4.1 (Optimal transport cost)**.** Let $c : \mathbb{R}^m \times \mathbb{R}^m \to [0, \infty]$ be any lower semi-continuous function such that $c(u, u) = 0 \quad \forall u \in \mathbb{R}^m$. Given two probability distributions $\mu_1, \mu_2 \in P(\mathbb{R}^m)$, the optimal transport cost is defined as

$$\mathcal{D}_c(\mu_1, \mu_2) = \inf_{\pi \in \Pi(\mu_1, \mu_2)} \int_{\mathbb{R}^m \times \mathbb{R}^m} c(x, u) d\pi(x, u),$$

Blanchet et al. do not apply any other assumptions on the cost function $c(\cdot)$, but if we choose $c^{1/p}(u, w) = \|u - w\|_q$ for some $p, q \geq 1$, then $\mathcal{D}_c^{1/p}(\cdot)$ coincides with the p-Wasserstein distance.

Using this generalization, their DRO formulation of this problem is defined as:

$$\inf_{\beta \in \mathbb{R}^d} \sup_{\mu : \mathcal{D}_c(\mu, \mu^*) \leq \varepsilon} \int_{\mathbb{R}^{d+1}} l(x, y; \beta) d\mu(x, y). \tag{3}$$

Here,

- $\mu^*(x, y) := \frac{1}{n} \sum_{i=1}^n \delta_{(x_i, y_i)}(x, y)$ is the empirical distribution,
- $\int_{\mathbb{R}^{d+1}} l(x, y; \beta) d\mu(x, y)$ denotes the expected loss under the distribution μ,
- and the loss function $l(x, y; \beta)$ evaluates the fit of the regression coefficient vector β for data point (x, y).

Proposition 4.1. *Let $c : \mathbb{R}^{d+1} \times \mathbb{R}^{d+1}$ be a lower semi-continuous cost funtion satisfying $c((x, y), (u, v)) = 0$ whenever $(x, y) = (u, v)$. For $\gamma \geq 0$ and loss functions $l(x, y; \beta)$ that are upper semi-continuous in (x, y) for each β, define*

$$\phi_\gamma(x, y; \beta) := \sup_{u \in \mathbb{R}^d, v \in \mathbb{R}} \{l(u, v; \beta) - \gamma c((u, v), (x, y))\}.$$

Then

$$\sup_{\mu : \mathcal{D}_c(\mu, \mu^*) \leq \varepsilon} \int_{\mathbb{R}^{d+1}} l(x, y; \beta) d\mu(x, y) = \min_{\gamma \geq 0} \left\{ \gamma \varepsilon + \int_{\mathbb{R}^{d+1}} \phi_\gamma(x, y; \beta) d\mu^*(x, y) \right\}.$$

The DR regression problem (3) *then reduces to*

$$\inf_{\beta\in\mathbb{R}^d}\sup_{\mu:\mathcal{D}_c(\mu,\mu^*)\le\varepsilon}\int_{\mathbb{R}^{d+1}} l(x,y;\beta)d\mu(x,y) = \inf_{\beta\in\mathbb{R}^d}\min_{\gamma\ge 0}\left\{\gamma\varepsilon + \int_{\mathbb{R}^{d+1}}\phi_\gamma(x,y;\beta)d\mu^*(x,y)\right\}. \tag{4}$$

For the proof of this proposition, see Section 4.2.

Proposition 4.2. *Fix $q\in[1,\infty]$ and let $\bar\beta = (-\beta,1)$ for brevity. Consider $l(x,y;\beta) = (y-\beta^T x)^2$ and $c((x,y),(u,v)) = \|(x,y)-(u,v)\|_q^2$. Then,*

$$\inf_{\beta\in\mathbb{R}^d}\sup_{\mu:\mathcal{D}_c(\mu,\mu^*)\le\varepsilon}\int_{\mathbb{R}^{d+1}} l(x,y;\beta)d\mu(x,y) = \inf_{\beta\in\mathbb{R}^d}\left(\sqrt{MSE_n(\beta)} + \sqrt{\varepsilon}\left\|\bar\beta\right\|_p\right)^2,$$

where $MSE_n(\beta) = \int(y-\beta^T x)^2 d\mu^*(x,y) = \frac{1}{n}\sum_{i=1}^N (y_i-\beta^T x_i)^2$ as the assumed distribution is discrete, and p is such that $1/p+1/q=1$.

Proof. For brevity, let $\bar x = (x,y)$ and $\bar\beta = (-\beta,1)$. The loss function is then $l(x,y;\beta) = (\bar\beta^T\bar x)^2$. We will start by evaluating $\phi_\gamma(x,y;\beta)$ as it is defined in Proposition (4.1) and use this expression to evaluate the right hand side of (4) in Proposition (4.1).

$$\phi_\gamma(x,y;\beta) = \sup_{\bar u\in\mathbb{R}^{d+1}}\left\{(\bar\beta^T\bar u)^2 - \gamma\|\bar x-\bar u\|_q^2\right\}.$$

Apply a change of variables $\Delta = \bar u - \bar x$ and use that by Hölders inequality $\left|\bar\beta^T\Delta\right| \le \left\|\bar\beta\right\|_p\|\Delta\|_q$ with $1/p+1/q=1$, where the equality holds for some $\Delta\in\mathbb{R}^{d+1}$:

$$\begin{aligned}
\phi_\gamma(\bar x;\beta) &= \sup_{\Delta\in\mathbb{R}^{d+1}}\left\{(\bar\beta^T\bar x + \bar\beta^T\Delta)^2 - \gamma\|\Delta\|_q^2\right\}\\
&= \sup_{\Delta\in\mathbb{R}^{d+1}}\left\{(\bar\beta^T\bar x + sign(\bar\beta^T\bar x)\left|\bar\beta^T\Delta\right|)^2 - \gamma\|\Delta\|_q^2\right\}\\
&= \sup_{\Delta\in\mathbb{R}^{d+1}}\left\{(\bar\beta^T\bar x + sign(\bar\beta^T\bar x)\left\|\bar\beta\right\|_p\|\Delta\|_q)^2 - \gamma\|\Delta\|_q^2\right\}\\
&= (\bar\beta^T\bar x)^2 + \sup_{\Delta\in\mathbb{R}^{d+1}}\left\{-(\gamma-\|\beta\|_p^2)\|\Delta\|_q^2 + 2\left|\bar\beta^T\bar x_i\right|\left\|\bar\beta\right\|_p\|\Delta\|_q\right\}\\
&= \begin{cases}(\bar\beta^T\bar x)^2\frac{\gamma}{\gamma-\left\|\bar\beta\right\|_p^2} & \text{if } \gamma > \|\beta\|_p^2,\\ \infty & \text{else.}\end{cases}
\end{aligned}$$

We can use this to evaluate

$$\begin{aligned}
\sup_{\mu:\mathcal{D}_c(\mu,\mu^*)\le\varepsilon}\int_{\mathbb{R}^{d+1}} l(x,y;\beta)d\mu(x,y) &= \inf_{\gamma\ge 0}\left\{\gamma\varepsilon + \int_{\mathbb{R}^{d+1}}\phi_\gamma(x,y;\beta)d\mu^*(x,y)\right\}\\
&= \inf_{\gamma\ge\|\beta\|_p^2}\left\{\gamma\varepsilon + \frac{\gamma}{\gamma-\left\|\bar\beta\right\|_p^2}\int_{\mathbb{R}^{d+1}}(\bar\beta^T\bar x)^2 d\mu^*(\bar x)\right\}.
\end{aligned} \tag{5}$$

Since Blanchet et al. assume a discrete setting, we have $\int(\bar\beta^T\bar x)^2 d\mu^*(\bar x) = \frac{1}{n}\sum_{i=1}^n(\bar\beta^T\bar x)^2 =: MSE_n(\beta)$ which we recognize as the mean square error.

The right-hand side of (5) is a convex function growing to ∞ (when $\gamma\to\infty$ or $\gamma\to\left\|\bar\beta\right\|_p^2$) so the global minimizer of this function can be characterized uniquely by first optimality condition. After some calculations, we finally have

$$\sup_{\mu:\mathcal{D}_c(\mu,\mu^*)\le\varepsilon}\int_{\mathbb{R}^{d+1}} l(x,y;\beta)d\mu(x,y) = \left(\sqrt{MSE_n(\beta)} + \sqrt{\varepsilon}\left\|\bar\beta\right\|_p^2\right)^2.$$

Combining this with the duality result from Proposition (4.1), we have

$$\begin{aligned}\inf_{\beta\in\mathbb{R}^d}\sup_{\mu:\mathcal{D}_c(\mu,\mu^*)\leq\varepsilon}\int_{\mathbb{R}^{d+1}} l(x,y;\beta)d\mu(x,y) &= \inf_{\beta\in\mathbb{R}^d}\inf_{\gamma\geq 0}\left\{\gamma\varepsilon+\int_{\mathbb{R}^{d+1}}\phi_\gamma(x_i,y_i;\beta)d\mu^*\right\}\\ &= \inf_{\beta\in\mathbb{R}^d}\left(\sqrt{MSE_n(\beta)}+\sqrt{\varepsilon}\left\|\bar{\beta}\right\|_p\right)^2\end{aligned}$$

□

Finally, Blanchet et al. modify the cost such that $c((x,y),(u,v)) = (N_q((x,y),(u,v)))^\rho$ with $\rho = 2$ and

$$N_q((x,y),(u,v)) = \begin{cases}\|x-u\|_q, & \text{if } y=v,\\ \infty, & \text{otherwise.}\end{cases}$$

This modified cost function assigns infinite cost when $y \neq v$ so the supremum in (3) is only over joint distributions that do not alter the marginal distribution of y, thus only admitting distributional ambiguities with respect to the predictor variables x in the ambiguity set.

Theorem 4.1. *Consider $l(x,y;\beta) = (y-\beta^T x)^2$ and $c((x,y),(u,v)) = (N_q((x,y),(u,v)))^\rho$ with $\rho = 2$. Then,*

$$\inf_{\beta\in\mathbb{R}^d}\sup_{\mu:\mathcal{D}_c(\mu,\mu^*)\leq\varepsilon}\int_{\mathbb{R}^{d+1}} l(x,y;\beta)d\mu(x,y) = \inf_{\beta\in\mathbb{R}^d}\left(\sqrt{MSE_n(\beta)}+\sqrt{\varepsilon}\|\beta\|_p\right)^2$$

where $MSE_n(\beta) := \int(y-\beta^T x)^2 d\mu^*(x,y) = \frac{1}{n}\overset{N}{i=1}(y_i-\beta^T x_i)^2$ as the assumed distribution is discrete, and p is such that $1/p+1/q=1$.

Thus recovering an l_p-norm regularized regression.

Proof. Again, start by seeing that

$$\begin{aligned}\phi_\gamma(x_i,y_i;\beta) &= \sup_{u\in\mathbb{R}^d,v\in\in\mathbb{R}}\left\{(v^T-\beta^T x)^2-\gamma N_q((u,v),(x_i,y_i))\right\}\\ &= \sup_{u\in\mathbb{R}^d}\left\{(y_i-\beta^T x)^2-\gamma N_q((u,y_i),(x_i,y_i))\right\}\\ &= \sup_{u\in\mathbb{R}^d}\left\{(y_i-\beta^T x)^2-\gamma\|u-x_i\|_q^2\right\}\\ &= \begin{cases}(y_i-\beta^T x)^2\frac{\gamma}{\gamma-\|\beta\|_p^2} & \text{if } \gamma>\|\beta\|_p^2,\\ \infty & \text{else.}\end{cases}\end{aligned}$$

The second equality follows from the fact that $N_q((u,v),(x_i,y_i)) = \infty$ whenever $v \neq y_i$ so the supremum is effectively only over (u,v) such that $v = y_i$. The last equality follows the same lines of reasoning as in the proof of Proposition 4.2 and the rest of the proof for Theorem 4.1 is exactly the same as that of Proposition 4.2.

□

4.2 Proof of Proposition 4.1

As mentioned before, the proof of Proposition 4.1 is an application of Theorem 1 from [44]. The assumptions and definitions will be summarized here in order to repeat the Theorem with some changes in notation with respect to Blanchet et al.

Assumption 1 (A1) $c : S\times S\to\mathbb{R}_+$ is a non-negative lower semi-continuous function satisfying $c(x,u) = 0$ if and only if $x = u$.

Assumption 2 (A2) $f \in L^1(d\mu^*)$ is upper semi-continuous.

4.2.1 The primal problem

The primal problem then is to evaluate

$$I := \sup\left\{\int l d\nu : \mathcal{D}_c(\mu^*, \nu) \leq \varepsilon\right\}.$$

As the infimum of the definition of the optimal transport cost $\mathcal{D}_c$ is attained for any given non-negative semi-continuous cost function c, this can be rewritten as follows:

$$I = \sup\left\{\int l(u) d\pi(x,u) : \pi \in \bigcup_{\nu \in P(S)} \Pi(\mu,\nu), \int c d\pi \leq \varepsilon\right\}.$$

If we let

$$I(\pi) := \int l(u) d\pi(x,u) \quad \text{and} \quad \Phi_{\mu^*,\varepsilon} := \left\{\bigcup_{\nu \in P(S)} \Pi(\mu,\nu) : \int c d\pi \leq \varepsilon\right\}$$

then

$$I = \sup\{I(\pi) : \pi \in \Phi_{\mu^*,\varepsilon}\} \tag{6}$$

is the primal problem.

4.2.2 The dual problem and weak duality

Use $m_{\mathcal{U}}(S;\mathbb{R})$ to denote the collection of measurable functions $\phi : (S,\mathcal{U}(S)) \to (\bar{\mathbb{R}}, \mathcal{B}(\bar{\mathbb{R}}))$ where $\mathcal{U}(S) = \cap_{\mu \in P(S)} \mathcal{B}_\mu(S)$ is the universal σ-algebra. Since $\mathcal{U}(S) \subseteq \mathcal{B}_\mu(S)$ for every $\mu \in P(S)$, any $\phi \in m_{\mathcal{U}}(S;\mathbb{R})$ is also measurable when S and $\bar{\mathbb{R}}$ are equipped with the σ-algebras $\mathcal{B}_\mu(S)$ and $\mathcal{B}(\bar{\mathbb{R}})$ respectively. This leads the integral $\int \phi d\mu$ to be well-defined for any non-negative $\phi \in m_{\mathcal{U}}(S;\mathbb{R})$.

Define $\Delta_{c,l}$ to be the collection of all pairs (λ, ϕ) such that λ is a non-negative real number, $\phi \in m_{\mathcal{U}}(S;\mathbb{R})$ and

$$\phi(x) + \lambda c(x,u) \geq l(u), \quad \forall x, u.$$

In other words,

$$\Delta_{c,l} := \{(\lambda,\phi) : \lambda \in \mathbb{R}_+, \phi \in m_{\mathcal{U}}(S;\mathbb{R}), \phi(x) \geq l(u) - \lambda c(x,u) \quad \forall x,u \in S\}.$$

For every such $(\lambda,\phi) \in \Delta_{c,l}$ consider the dual problem

$$J(\lambda,\phi) := \lambda\varepsilon + \int_S \phi(x) d\mu^*(x). \tag{7}$$

Theorem 4.2 (Weak duality)**.** *Assume (A1) and (A2) hold. We have $J \geq I$ whenever I and J are defined as in* (6) *and* (7)*, respectively.*

Proof. For any $\pi \in \Phi_{\mu^*,\varepsilon}$ and $(\lambda,\phi) \in \Lambda_{c,l}$, we have

$$\begin{aligned}
J(\lambda,\phi) &= \lambda\varepsilon + \int \phi(x) d\pi(x,u) \\
&\geq \lambda\varepsilon + \int (l(u) - \lambda c(x,u)) d\pi(x,u) \\
&= \int l(u) d\pi(x,y) + \lambda\left(\varepsilon - \int c(x,u) d\pi(x,u)\right) \\
&\geq \int l(u) d\pi(x,u) \\
&= I(\pi).
\end{aligned}$$

Thus,

$$J := \inf\{J(\lambda,\phi) : (\lambda,\phi) \in \Delta_{c,l}\} \geq I.$$

□

4.2.3 Strong duality

Theorem 4.3. *Under the Assumptions (A1) and (A2),*

(a) I=J. In other words,

$$\sup\{I(\pi) : \pi \in \Phi_{\mu^*,\varepsilon}\} = \inf\{J(\lambda,\phi) : (\lambda,\phi) \in \Delta_{c,l}\}.$$

(b) For any $\lambda \geq 0$, define $\phi_\lambda : S \to \mathbb{R}_+$ as follows:

$$\phi_\lambda(x) := \sup_{u\in S}\{l(u) - \lambda c(x,u)\}.$$

There exists a dual optimizer of the form (λ, ϕ_λ) for some $\lambda \geq 0$. In addition, any feasible $\phi^* \in \Phi_{\mu^*,\varepsilon}$ and $(\lambda^*, \phi_{\lambda^*}) \in \Delta_{c,l}$ are primal and dual optimizers, satisfying $I(\pi^*) = J(\lambda^*, \phi_{\lambda^*})$ if and only if

(i)

$$l(u) - \lambda^* c(x,u) = \sup_{z\in S}\{l(z) - \lambda^* c(x,z)\} \quad \pi^* a.s., \text{ and}$$

(ii)

$$\lambda^* \left(\int c(x,u) d\pi^*(x,u) - \varepsilon \right) = 0.$$

Outline of a Proof. For the full proof we refer the reader to [35, Ch.4], here we will summarize the outline of the proof for (a) with the additional assumptions that S is a compact Polish space and $c : S \times S \to \mathbb{R}_+$ is continuous. This is part of the proof has been the most significant source for our own strong duality proof later on.

Let $X = C_b(S \times S)$ and identify its topological dual $X^* = M(S \times S)$ which represent the vector space of bounded continuous functions equipped with the supremum norm and finite Borel measures on $S \times S$ equipped with the total variation norm, respectively.

Define C and D as the sets of functions

$$C := \{g \in X : \exists \phi \in C_b(S), \lambda \geq 0 \text{ s.t. } g(x,u) = \phi(x) + \lambda c(x,u) \quad \forall x,u \in S\}$$

and

$$D := \{g \in X : g(x,u) \geq l(u) \quad \forall x,u \in S\}.$$

Every g in the convex subset C is defined by the pair (λ, ϕ), which in turn, can be uniquely identified by,

$$\phi(x) = g(x,x) \quad \text{and} \quad \lambda = \frac{g(x,y) - \phi(x)}{c(x,y)},$$

for some $(x,y) \in S$ such that $c(x,y) \neq 0$. With this invertible relationship in mind, define the functionals $\Theta : C \to \mathbb{R}$ and $\Gamma : D \to \mathbb{R}$ as:

$$\Theta(g) := \lambda\varepsilon + \int \phi d\mu \quad \text{and} \quad \Gamma(g) := 0.$$

The functional Θ is convex, Γ is concave, and we are interested in

$$\inf_{g\in C\cap D}\{\Theta(g) - \Gamma(g)\} = \inf\{J(\lambda,\phi) : \lambda \geq 0, \phi \in C_b(S), \phi(x) + \lambda c(x,u) \geq l(u) \quad \forall x,u\}.$$

Next, we want to identify the conjugate functionals $\Theta^* : C^* \to \mathbb{R}$ and $\Gamma^* : D^* \to \mathbb{R}$ and their respective domains C^* and D^*. By definition of the conjugate functional,

$$C^* = \left\{ \pi \in X^* : \sup_{g \in C} \left\{ \int g d\pi - \Theta(g) \right\} < \infty \right\} \quad \text{and} \quad D^* = \left\{ \pi \in X^* : \inf_{g \in D} \int g d\pi < -\infty \right\},$$
$$\Theta^*(\pi) := \sup_{g \in C} \left\{ \int g d\pi - \Theta(g) \right\} \quad \text{and} \quad \Gamma^*(\pi) := \inf_{g \in D} \int g d\pi.$$

To determine C^* and Θ^*, see that $\forall \pi \in M(S \times S)$,

$$\begin{aligned}
\Theta^*(\pi) &= \sup_{g \in C} \left\{ \int g d\pi - \Theta(g) \right\} \\
&= \sup_{(\lambda, \phi) \in \mathbb{R}_+ \times C_B(S)} \left\{ \int_{S \times S} (\phi(x) + \lambda c(x,u)) d\pi(x,u) - \left(\lambda \varepsilon + \int_S \phi(x) d\mu^*(x) \right) \right\} \\
&= \sup_{(\lambda, \phi) \in \mathbb{R}_+ \times C_B(S)} \left\{ \lambda \left(\int_{S \times S} c(x,u) d\pi(x,u) - \varepsilon \right) - \int_S \phi(x) \left(d\pi(x,u) - d\mu^*(x) \right) \right\} \\
&= \begin{cases} 0 & \text{if } \int c d\pi \leq \varepsilon \text{ and } \pi(Q \times S) = \mu^*(Q) \quad \forall Q \in \mathcal{B}(S), \\ \infty & \text{otherwise.} \end{cases}
\end{aligned}$$

Thus, we have

$$C^* = \left\{ \pi \in M(S \times S) : \int c d\pi \leq \varepsilon, \quad \pi(Q \times S) = \mu^* \quad \forall Q \in \mathcal{B}(S) \right\}$$

and $\Theta^* = 0$.

To determine D^*, we use Lemma 15 in appendix B from [44] which states that $\inf_{g \in D} \int g d\pi < -\infty$ whenever $\pi \in M(S \times S)$ is not non-negative. If it is non-negative, then

$$\inf \left\{ \int g(x,u) d\pi(x,u) : g(x,u) \geq l(u), \quad \forall x, u \right\} = \int l(u) d\pi(x,u)$$

as l is upper semi-continuous and bounded from above. Thus it can be approximated pointwise by a monotonically decreasing sequence of continuous functions (if $d(\cdot, \cdot)$ is a function that metrizes S then $f_n(x) = \sup_{u \in S} \{ f(u) - n d(x,u) \}$ is continuous and satisfies $f_n \downarrow f$ pointwise) and the equality follows by the monotone convergence theorem. Thus we have

$$D^* = \left\{ \pi \in M_+(S \times S) : \int l d\pi > -\infty \right\} \text{ and } \Gamma^*(\pi) = \int l d\pi.$$

Then

$$\begin{aligned}
\Gamma^*(\pi) - \Theta^*(\pi) &= \int l d\pi \\
\text{on } C^* \cap D^* &= \left\{ \pi \in \cup_{\nu \in K} \Pi(\mu^*, \nu) : \int c d\pi \leq \varepsilon, \int l d\pi > -\infty \right\}.
\end{aligned}$$

Since I is defined to equal $\sup \left\{ \int l d\mu : \mathcal{D}_c(\mu, \mu^*) \leq \varepsilon, \int l d\mu > -\infty \right\}$, it follows that

$$\sup_{\pi \in C^* \cap D^*} \{ \Gamma^*(\pi) - \Theta^*(\pi) \} = I.$$

$$\sup \{ \Gamma^*(\pi) - \Phi^*(\pi) : \pi \in C^* \cap D^* \} = I.$$

The set $C \cap D$ contains points in the relative interiors of C and D and the epigraph of the function Γ has non-empty interior. Thus it follows from Fenchel's duality theorem [45, p.201] (see Section 4.4) that

$$\inf_{g\in C\cap D}\{\Theta(g)-\Gamma(g)\}=\sup\{\Gamma^*(\pi)-\Phi^*(\pi):\pi\in C^*\cap D^*\}.$$

where the supremum on the right hand side is achieved by some $\pi^*\in\Phi_{\mu^*,\varepsilon}$. We can rewrite,

$$\inf\{J(\lambda,\phi):\lambda\geq 0,\phi\in C_b(S),\phi(x)+\lambda c(x,u)\geq l(u)\quad\forall x,u\}=\max\{I(\pi):\pi\in\Phi_{\mu,\varepsilon}\}=:I.$$

Since $C_b(S)\subseteq m_{\mathcal{U}}(S;\bar{R})$,

$$J\leq\inf\{J(\lambda,\phi):\lambda\geq 0,\phi\in C_b(S),\phi(x)+\lambda c(x,u)\geq l(u)\quad\forall x,u\}=I.$$

Due to weak duality we have $J\geq I$, therefore, $J=I$ and we have strong duality.

□

4.2.4 Dual optimizer

Blanchet's proof that (λ,ϕ_λ) with $\phi_\lambda(x):=\sup_{u\in S}\{l(u)-\lambda c(x,u)\}$ optimizes the dual problem is quite involved and part of a larger proof including some parts not relevant to us, and releases some assumptions that we wish to maintain. Therefore, we present our own small Lemma below to show (λ,ϕ_λ) is an optimizer.

Lemma 4.4. *Define $\phi_\lambda(x):=\sup\{\phi(x)+\lambda c(x,u)\}$. We have*

$$\inf_{(\lambda,\phi)\in\Delta_{c,l}}J(\lambda,\phi)=\inf_{\lambda\geq 0}J(\lambda,\phi_\lambda).$$

Proof. We want to prove

$$\inf_{(\lambda,\phi)\in\Delta_{c,l}}\left\{\lambda\delta+\int\phi(x)d\mu^*\right\}=\inf_{\lambda\geq 0}\left\{\lambda\delta+\int\sup_u\{l(u)-\lambda c(x,u)\}d\mu^*\right\}.$$

Recall that for every $(\lambda,\phi)\in\Delta_{c,l}$:

$$\phi(x)\geq l(u)-\lambda c(x,u)\quad\forall x,u$$

then also

$$\phi(x)\geq\sup_u\{l(u)-\lambda c(x,u)\}\quad\forall x.$$

which leads us to our first inequality,

$$\inf_{(\lambda,\phi)\in\Delta c,l}\left\{\lambda\delta+\int\phi(x)d\mu^*\right\}\geq\inf_{\lambda\geq 0}\left\{\lambda\delta+\int\sup_u\{l(u)-\lambda c(x,u)\}d\mu^*\right\}. \tag{8}$$

Secondly, let $\phi_\lambda(x):=\sup\{\phi(x)+\lambda c(x,u)\}$ and see that

$$\inf_{(\lambda,\phi)\in\Delta c,l}\left\{\lambda\delta+\int\phi(x)d\mu^*\right\}\leq\inf_{(\lambda,\phi_\lambda)\in\Delta c,l}\left\{\lambda\delta+\int\phi_\lambda(x)d\mu^*\right\} \tag{9}$$

$$=\inf_{\lambda\geq 0}\left\{\lambda\delta+\int\sup\{\phi(x)+\lambda c(x,u)\}d\mu^*\right\} \tag{10}$$

because in (9) the infimum on the left-hand side is taken over a bigger set than the infimum on the right-hand side. Combining (8) and (10), we get an equality. This completes the proof. □

4.3 Evaluation of dual representation using the proximal operator

This section, we will explore some characteristics of the dual optimizer, which is a new contribution to the best of our knowledge. We take again $\phi_\lambda(\bar{x};\beta):=\sup_{\bar{u}\in S}\{l(\bar{u};\beta)-\lambda c(\bar{x},\bar{u})\}$ and we choose cost function $c(\bar{x},\bar{u}):=\|\bar{u}-\bar{x}\|_2^2$. Notice that

$$\phi_\lambda(\bar{x};\beta)=\sup_{\bar{u}\in S}\{l(\bar{u};\beta)-\lambda\|\bar{u}-\bar{x}\|_2^2\}=-\lambda\inf_{\bar{u}\in S}\left\{-\frac{1}{\lambda}l(\bar{u};\beta)+\|\bar{u}-\bar{x}\|_2^2\right\}.$$

Here we can recognize the proximal operator since

$$\arg\min_{\bar{u}\in S}\left\{-\frac{1}{\lambda}l(\bar{u};\beta)+\|\bar{u}-\bar{x}\|_2^2\right\}=:\operatorname{prox}_{-\frac{1}{\lambda}l(\bar{u};g)}(\bar{x}).$$

Dependent on our choice for the loss function, this allows for an explicit expression of ϕ_λ. To the best of our knowledge, this is a new contribution and could help us find an explicit expression for higher-dimensional problems than linear regression.

We will use the following properties for proximal operators [46]:

1. **Quadratic function:** for any quadratic function $f(x)=\frac{1}{2}x^TAx+b^Tx+c$ with A a semi-definite square matrix we have $\operatorname{prox}_{tf}(x)=(I+tA)^{-1}(x-tb)$.
2. **Composition with affine mapping:** If $f(x)=g(Ax+b)$ with $AA^T=(1/\alpha)I$, then $\operatorname{prox}_f(x)=(I-\alpha A^TA)x+\alpha A^T(\operatorname{prox}_{\alpha^{-1}g}(Ax+b)-b)$.
3. **Moreau decomposition:** for $\lambda>0$, $x=\operatorname{prox}_{\lambda f}(x)+\lambda\operatorname{prox}_{\lambda^{-1}f^*}(x/\lambda)\quad\forall x$.

4.3.1 Quadratic loss function

To take on the linear regressional setting from Blanchet et al. (Section 4.1) we modify the cost

$$c((x,y),(u,v))=\begin{cases}\|u-x\|_2^2, & \text{if } v=y\\ \infty, & \text{else}\end{cases}$$

to only admit ambiguities in the x-space to simplify the problem. This allows us to write x and u without the bars. We take $u,x,\beta\in\mathbb{R}^d$, $y=v\in\mathbb{R}$ and a square loss function $l(u,v;g)=(\beta^Tu-v)^2$. We rewrite the loss:

$$\begin{aligned}l(u,v;\beta)&=(\beta^Tu-v)^2=(\beta^Tu-v)(\beta^Tx-v)^T\\&=(\beta^Tu)^T(\beta^Tu)-v\beta^Tu-\beta^Tuv+v^2\\&=u^T\beta\beta^Tu-2v\beta^Tu+v^2\\&=\frac{1}{2}u^T2\beta\beta^Tu-2v\beta^Tu+v^2\\&=\frac{1}{2}u^TAu+b^Tu+c\end{aligned}$$

where we let $A=2\beta\beta^T$, $b=-2v\beta, c=v^2$ to recognize a quadratic function. Finally letting $t=-\frac{1}{\lambda}$ we can use property 1 for quadratic functions to find an explicit expression for the proximal operator:

$$\operatorname{prox}_{tl}(x)=(I+tA)^{-1}(x-tb)=(I-\frac{2}{\lambda}\beta\beta^T)^{-1}(x+\frac{2}{\lambda}v\beta)=(I-\frac{2}{\lambda}\beta\beta^T)^{-1}(x+\frac{2}{\lambda}y\beta)=:\hat{u}.$$

since we only consider $v=y$. This means that $\phi_\lambda(x,y;\beta)=(y-\beta^Tx)-\lambda\|\hat{u}-x\|_2^2$.

4.3.2 Norm loss

Let $g(u)=Gu$ where $G\in\mathbb{R}^{n\times m}$, $n,m>0$ s.t. $GG^T=\frac{1}{\alpha}I$ and $l(u;g)=\|g(u)-y\|=\|Gu-y\|$ for any norm $\|\cdot\|$. To simplify notation, let $f(u)=-\frac{1}{\lambda}l(u)$. Then, we are looking for $\operatorname{prox}_f(x)$. Additionally, let $k(x)=-\frac{1}{\lambda}\|x\|^2$, then $f(u)=k(Gu-y)$. Using property 2, we see that

$$\operatorname{prox}_f(x)=x-\alpha G^T\left(Gx-y-\operatorname{prox}_{\alpha^{-1}k}(Gx-y)\right). \tag{11}$$

If we let $h(x)=\|x\|$, then $\alpha^{-1}k(x)=-(\alpha\lambda)^{-1}h(x)$. Assuming $-(\alpha\lambda)^{-1}>0$, use the Moreau decomposition to see that

$$\begin{aligned}\operatorname{prox}_{\alpha^{-1}k}(Gx-y) &= \operatorname{prox}_{-(\alpha\lambda)^{-1}h}(Gx-y)\\ &= Gx-y-(-(\alpha\lambda)^{-1})\operatorname{prox}_{((\alpha\lambda)^{-1}h^*}(-\alpha\lambda)(Gx-y)) \\ &= Gx-y+(\alpha\lambda^{-1}\operatorname{prox}_{-\alpha\lambda h^*}(-\alpha\lambda(Gx-y)).\end{aligned} \tag{12}$$

Notice that in the dual problem, we optimize over $\lambda \geq 0$, which means we require $\alpha < 0$.

For $h(\cdot) = \|\cdot\|$ any norm we have $h^*(y) = \delta_{\mathcal{B}_{\|\cdot\|_*}}(y)$ with $\mathcal{B}_{\|\cdot\|_*} := \{y \mid \|y\|_* \leq 1\}$, which is the indicator function of the dual norm ball. Thus for γ any scalar we know $\operatorname{prox}_{\gamma h^*}(x) = \operatorname{proj}_{\gamma\mathcal{B}_{\|\cdot\|_*}}(x)$ where $\gamma\mathcal{B}_{\|\cdot\|_*} := \{y \mid \|y\|_* \leq \gamma\}$ and $\operatorname{proj}_{\gamma\mathcal{B}_{\|\cdot\|_*}}$ is the orthogonal projection onto the scaled dual norm ball [46]. Thus,

$$\operatorname{prox}_{-\lambda\alpha h^*}(-\alpha\lambda(Gx-y)) = \operatorname{proj}_{\mathcal{B}_{\|\cdot\|_*}}(-\lambda\alpha(Gx-y)) \tag{13}$$

Putting (11), (12) and (13) together we finally have

$$\begin{aligned}\operatorname{prox}_f(x) &= x-\alpha G^T\Big(Gx-y-\Big(Gx-y+(\alpha\lambda)^{-1}\operatorname{proj}_{\mathcal{B}\|\cdot\|_*}(-\alpha\lambda(Gx-y)\Big)\Big)\\ &= x-\alpha G^T\Big(Gx-y-Gx+y-\operatorname{proj}_{\mathcal{B}\|\cdot\|_*}\big((\alpha\lambda)^{-1}(Gx-y)\big)\Big)\\ &= x+\lambda^{-1}G^T\operatorname{proj}_{\mathcal{B}\|\cdot\|_*}\big((\alpha\lambda)^{-1}(Gx-y)\big).\end{aligned} \tag{14}$$

To evaluate the value of the proximal operator further, we can choose the L_2 norm as this has an explicit expression for the proximal operator [46]. Since the L_2-norm is self-dual, we can use that the projection onto the unit ball is given by

$$\operatorname{proj}_{\mathcal{B}_{\|\cdot\|_2}}(x) = \begin{cases}\frac{x}{\|x\|_2} & \text{if } \|x\|_2 > 1\\ x & \text{if } \|x\|_2 \leq 1.\end{cases}$$

Thus,

$$\operatorname{proj}_{\mathcal{B}_{\|\cdot\|_2}}(-\alpha\lambda(Gx-y)) = \begin{cases}-\frac{Gx-y}{\|Gx-y\|_2}, & \text{if } \|Gx-y\|_2 \geq 1\\ -\alpha\lambda(Gx-y) & \text{else}\end{cases}$$

Plugging this into (14), we get

$$\begin{aligned}\operatorname{prox}_f(x) &= x+\lambda^{-1}G^T\cdot\begin{cases}-\frac{Gx-y}{\|Gx-y\|_2}, & \text{if } \|Gx-y\|_2 \geq 1\\ -\alpha\lambda(Gx-y), & \text{else}\end{cases}\\ &= x-G^T(Gx-y)\cdot\begin{cases}\frac{1}{\lambda\|Gx-y\|_2}, & \text{if } \|Gx-y\|_2 \geq 1\\ \alpha, & \text{else.}\end{cases}\end{aligned}$$

In this thesis, we will focus on achieving a duality result and some numerical simulations but the insights from this section could be useful for future research wanting to explicitly evaluate higher-dimensional DRO-problems.

4.4 Fenchel duality theorem

The Fenchel Duality Theorem [45, p.201] is necessary for Blanchet et al.'s proof of strong duality, and will also be necessary for our strong duality proof later in this thesis. The theorem is as follows.

Theorem 4.5 (Fenchel Duality Theorem)**.** *Assume that f and g are, respectively, convex and concave functionals on the convex sets C and D in a normed space X. Assume that $C \cap D$ contains points*

in the relative interior of both C and D and that either $[f, C]$ or $[g, D]$ has nonempty interior, where $[f, C]$ denotes the epigraph of f over C. Lastly, suppose $\inf_{x\in C\cap D}\{f(x) - g(x)\}$ is finite. Then

$$\inf_{x\in C\cap D}\{f(x) - g(x)\} = \max_{x^*\in C^*\cap D^*}\{g^*(x^*) - f^*(x^*)\}$$

where the maximum on the right-hand side is achieved by some $x_0^* \in C^* \cap D^*$. If the infimum on the left is achieved by some $x_0 \in C \cap D$, then

$$\max_{x\in C}[\langle x, x_0^*\rangle - f(x)] = \langle x, x_0^*\rangle - f(x_0)$$

and

$$\max_{x\in D}[\langle x, x_0^*\rangle - g(x)] = \langle x, x_0^*\rangle - g(x_0).$$

For the proof we refer the reader to [45, p.201].

4.5 Alternative proof for strong duality

In [47], Chen et al. provide an interesting alternative proof of strong duality of the Wasserstein-DRO problem using the Lagrangian dual and push-forward measures. Ultimately, a proof similar to this one does not allow us the freedom to manipulate the definitions to suit our framework (which we will introduce in the next section). However, it provides an interesting alternative perspective on the proof/problem. We define the primal problem similarly, as

$$I := \sup_{\pi\in\Phi} I(\pi) \tag{15}$$

where $I(\pi) := \int l(u)d\pi(x, u)$ and $\Phi := \left\{\pi \in P(S \times S), \pi \in \cup_{\nu\in P(S)}\Pi(\mu^*, \nu) : W(\mu^*, \nu) \leq \varepsilon\right\}$.

We define the dual problem as

$$J := \inf_{\lambda\geq 0} J(\lambda) \tag{16}$$

where $J(\lambda) := \lambda\varepsilon + \int \sup_{u\in S}\{l(u) - \lambda c(x, u)\}\, d\mu^*(x)$.

Note that in this method, we choose a specific map inside the integral right away, instead of leaving it general as in the proof by Blanchet et al. [44].

Definition 4.2 (Growth Rate)**.** Define the growth rate of the loss function $l(u)$ given an unbounded set X and a fixed $x \in X$ as

$$\mathrm{GR}_l := \limsup_{c(x,u)\to\infty} \frac{|h(u) - h(x)|}{c(x, u)}.$$

In [47, Th.3.1.1.], Chen et al. show that if the growth rate of the loss function is infinite, the primal problem will not have a finite optimal value and that strong duality fails to hold.

Theorem 4.6 (Weak duality)**.** *Suppose the loss function is upper semi-continuous and has finite growth rate $GR_l < \infty$. Then weak duality holds: $I \leq J$, where I and J are defined as in* (15) *and* (16) *respectively.*

Chen et al. use a slightly different proof for their weak duality theorem than what we will show below. As the expressions are similar, a weak duality proof similar to the one employed by Blanchet et al. in Section 4.2 suffices here as well.

Proof. For any $\pi \in \Phi$:

$$
\begin{aligned}
J(\lambda) &:= \lambda\varepsilon + \int \sup_{u\in S} \{l(u) - \lambda c(x,u)\}\, d\mu^*(x) \\
&\geq \lambda\varepsilon + \int l(u) - \lambda c(x,u) d\pi(x,u) \\
&= \int l(u) d\pi + \lambda \left(\varepsilon - \int c(x,u) d\pi(x,u)\right) \\
&\geq \int l(u) d\pi(x,u) \\
&:= I(\pi)
\end{aligned}
$$

So $J := \inf_{\lambda\geq 0} J(\lambda) \geq \sup_{\pi\in\Phi} I(\pi) = I$ and we have weak duality. □

4.5.1 Strong duality

Theorem 4.7 (Strong duality)**.** *Suppose the loss function is upper semi-continuous and has finite growth rate $GR_l < \infty$. Then the dual problem* (16) *always admits a minimizer λ^* and strong duality holds: $I = J < \infty$, where I and J are defined as in* (15) *and* (16) *respectively.*

Proof. Let $\phi_\lambda := \inf_{u\in S}\{\lambda c(x,u) - l(u)\}$. Construct a measure ν (candidate optimizer) as a convex combination of two measures, each of which is a perturbation of μ^* :

$$
\nu = qT_{\#\mu^*} + (1-q)\hat{T}_{\#\mu^*} \tag{17}
$$

where $T, \hat{T} : S \to S$ produce a minimizer to ϕ_{λ^*} where λ^* is the optimal solution to the dual problem. In other words,

$$
T(x), \hat{T}(x) \in \{u \in S : \lambda^* c(x,u) - l(u) = \phi_{\lambda^*}(x)\}
$$

The q in (17) is chosen such that $q \in [0,1]$ and

$$
q\int_S c(T(x),x)d\mu^*(x) + (1-q)\int_S c(\hat{T}(x),x)d\mu^*(x) = \varepsilon. \tag{18}
$$

To ensure the existence of such a q, choose $T, \hat{T}$ to satisfy

$$
\int_S c(T(\cdot),\cdot)d\mu^* \leq \varepsilon \text{ and } \int_S c(\hat{T}(\cdot),\cdot)d\mu^* \geq \varepsilon.
$$

First, see that

$$
\begin{aligned}
W(\nu,\mu^*) &= \sup_{f,g}\left\{\int_S f(u)d\nu(u) + \int_S g(x)d\mu^*(x) : f(u) \leq \inf_{x\in S}\{c(x,u) - g(x)\}, \quad \forall u \in S\right\} \\
&= \sup_{f,g}\left\{q\int_S f(u)d\mu^*(T^{-1}(u)) + (1-q)\int_S f(u)d\mu^*(\hat{T}^{-1}(x))\right. \\
&\qquad\qquad \left. + \int_S g(x)d\mu^*(x) : f(u) \leq \inf_{x\in S}\{c(x,u) - g(x)\}, \quad \forall u \in S\right\} \\
&\leq \sup_{g}\left\{q\int_S (c(T(x),x) - g(x))\, d\mu^*(x) + (1-q)\int_S \left(c(\hat{T}(x),x) - g(x)\right) d\mu^*(x)\right. \\
&\qquad\qquad \left. + \int_S g(x)d\mu^*(x)\right\} \\
&= q\int_S c(T(x),x)d\mu^*(x) + (1-q)\int_S c(\hat{T}(x),x)d\mu^*(x) \\
&= \varepsilon
\end{aligned}
$$

where the first equality follows from Kantorovich duality (see Theorem 3.2), the second step uses the structure of ν defined in (17), the third step replaces $f(u)$ with its upper bound and the last step uses the definition of q in (18). This means that ν is in the Wasserstein-ball, in other words it is primal feasible.

Now we will establish the optimality of ν by showing its objective value is equal to the optimal dual value.

$$\begin{aligned}
\int_S l(u)d\nu(u) &= q\int_S l(u)d\mu^*(T^{-1}(u)) + (1-q)\int_S l(u)d\mu^*(\hat{T}^{-1}(u)) \\
&= q\int_S \left(\lambda^* c(T(x),x) - \phi_{\lambda^*}(x)\right) d\mu^*(x) + (1-q)\int_S \left(\lambda^* c(\hat{T}(x),x) - \phi_{\lambda^*}(x)\right) d\mu^*(x) \\
&= q\lambda^* \int_S c(T(x),x)d\mu^*(x) - \int_S \phi_{\lambda^*}(x) + (1-q)\lambda^* \int_S c(\hat{T}(x),x)d\mu^*(x) \\
&= \lambda^*\varepsilon - \int_S \phi_{\lambda^*}(x)d\mu^*(x) \\
&= J.
\end{aligned}$$

where the second equality follows from the definition of T and $\hat{T}$, the fourth quality comes from the definition of q and the final equality follows from the optimality of λ^*. Then $I = \sup_{\pi\in\Phi} \int l(u)d\nu(u) \geq J$ and since $I \leq J$ we have strong duality: $I = J$. □

4.6 Convex reduction of Wasserstein-DRO problem

The sections above have focused on the dual representation and a finite-dimensional reduction for a linear regressional setting. In general, the Wasserstein-DRO problem in (2) is an infinite-dimensional optimization problem and thus intractable. Esfahani and Kuhn [48, Section 4.1] have shown that the Wasserstein-DRO problem can be expressed as a finite-dimensional convex program, as will be stated in the following Theorem which corresponds to Theorem 4.2 in [48, p.129]. We repeat the theorem here with the following changes of notation, where the symbol before the arrow is the symbol used by Esfahani and Kuhn: $\Xi \to S$, $\xi \to x$, $\mathbb{Q} \to \mu$, $\hat{\mathbb{P}}_N \to \mu^*$.

Theorem 4.8 (Convex reduction.)**.** *Let $B_\varepsilon(\mu^*) := \{\mu \in P(S) : W_1(\mu,\mu^*) \leq \varepsilon\}$, $l_k(x) := \max_{k\leq K} l_k(x)$ with $l_k : \mathbb{R}^m \to \bar{\mathbb{R}}, k \leq K$. Assume that $S \subseteq \mathbb{R}^m$ is convex and closed, that the functions $-l_k$ are proper, convex and lower semi-continuous for all $k \leq K$ and that l_k does not always equal $-\infty$ on S for all $k \leq K$.*

With these conventions and assumptions, we have

$$\sup_{\mu\in B_\varepsilon(\mu^*)} \int_S l(x)d\mu = \begin{cases} \inf\limits_{\lambda,s_i,z_{ik},v_{ik}} \lambda\varepsilon + \dfrac{1}{N}\sum\limits_{i=1}^N s_i & \\ \quad s.t. [-l_k]^*(z_{ik}-v_{ik}) - \sigma_S(v_{ik}) - \langle z_{ik}, \hat{x_i}\rangle \leq s_i & \forall i \leq N, \forall k \leq K \\ \quad \|z_{ik}\|_* & \forall i \leq N, \forall k \leq K \end{cases} \tag{19}$$

with $\chi_S(x) := \begin{cases} 0 & \text{if } x \in S \\ \infty & \text{else} \end{cases}$ *the characteristic function of S and $\sigma_S(z) := \sup_{x\in S}\langle z, x\rangle$ its conjugate which is the support function of S. Lastly, $[-l_k]^*(z_{ik}-v_{ik})$ denotes the conjugate of $-l_k$ evaluated at $z_{ik}-v_{ik}$ and $\|z_{ik}\|_*$ denotes the dual norm of z_{ik}.*

Proof. Using the definition of the Wasserstein-distance, we rewrite

$$\begin{aligned}
\sup_{\mu\in B_\varepsilon(\mu^*)} \int_S l(x)d\mu &= \begin{cases} \sup_{\pi,\mu} & \int_S l(x)d\mu \\ \text{s.t.} & \int_{S\times S} \|x-u\|d\pi(x,u) \leq \varepsilon \\ & \pi \in \Pi(\mu,\mu^*) \end{cases} \\
&= \begin{cases} \sup_{\mu_i\in P(S)} & \frac{1}{N}\sum_{i=1}^N \int_S l(x)d\mu_i(x) \\ \text{s.t.} & \frac{1}{N}\sum_{i=1}^N \int_S \|x-\hat{x_i}\|d\mu_i(x) \leq \varepsilon. \end{cases}
\end{aligned} \tag{20}$$

where the second equality follows from using using Bayes law to see that $\pi(x,u)$ can be constructed from the marginal distribution $\mu^*(u)$ and the conditional distributions $\mu_i(x|u=\hat{x}_i)$, $i \leq N$, in other words $\pi = \frac{1}{N}\sum_{i=1}^N \delta_{\hat{x_i}}(u) \otimes \mu_i(x)$. By duality we can rewrite (20):

$$\begin{aligned}
&\sup_{\mu_i \in P(S)} \inf_{\lambda \geq 0} \frac{1}{N}\sum_{i=1}^N \int_S l(x) d\mu_i(x) + \lambda\left(\varepsilon - \frac{1}{N}\sum_{i=1}^N \int_S \|x-\hat{x_i}\| d\mu_i(x)\right) \\
&\leq \inf_{\lambda \geq 0} \sup_{\mu_i \in P(S)} \lambda\varepsilon + \frac{1}{N}\sum_{i=1}^N \int_S \left(l(x) - \lambda\|x-\hat{x_i}\|\right) d\mu_i(x) \\
&= \inf_{\lambda \geq 0} \lambda\varepsilon + \frac{1}{N}\sum_{i=1}^N \sup_{x \in S} \left(l(x) - \lambda\|x-\hat{x_i}\|\right) \qquad (21)
\end{aligned}$$

where the first (in-)equality follows from the max-min inequality and the second follows from the fact that $P(S)$ contains the dirac distributions supported on S. Introducing the auxiliary variables s_i, $i \leq N$ we rewrite (21):

$$\begin{aligned}
&\begin{cases} \inf_{\lambda, s_i} & \lambda\varepsilon + \frac{1}{N}\sum_{i=1}^N s_i \\ \text{s.t.} & \sup_{x\in S}\left(l(x) - \lambda\|x-\hat{x_i}\|\right) \leq s_i, \quad \forall i \leq N \\ & \lambda \geq 0 \end{cases} \\
=&\begin{cases} \inf_{\lambda, s_i} & \lambda\varepsilon + \frac{1}{N}\sum_{i=1}^N s_i \\ \text{s.t.} & \sup_{x\in S}\left(l_k(x) - \max_{\|z_ik\|_* \leq \lambda}\langle z_{ik}, x-\hat{x_i}\rangle\right) \leq s_i, \quad \forall i \leq N, \forall k \leq K \\ & \lambda \geq 0 \end{cases} \\
\leq&\begin{cases} \inf_{\lambda, s_i} & \lambda\varepsilon + \frac{1}{N}\sum_{i=1}^N s_i \\ \text{s.t.} & \min_{\|z_ik\|_* \leq \lambda}\sup_{x\in S}\left(l_k(x) - \langle z_{ik}, x-\hat{x_i}\rangle\right) \leq s_i, \quad \forall i \leq N, \forall k \leq K \\ & \lambda \geq 0 \end{cases}
\end{aligned}$$

where the first equality uses the definition of the dual norm $\|x\|_* := \sup_{\|z\|\leq 1}\langle z, x\rangle$ and decomposes $l(x)$ while the second inequality follows from interchanging the maximization over z_{ik} with the minus sign and once again using the max-min inequality. We can move the minimum over z_{ik} to a new constraint:

$$\begin{aligned}
&\begin{cases} \inf\limits_{\lambda, s_i, z_{ik}} & \lambda\varepsilon + \frac{1}{N}\sum_{i=1}^N s_i \\ & \text{s.t.} \sup\limits_{x\in S}\left(l_k(x) - \langle z_{ik}, x\rangle\right) + \langle z_{ik}, \hat{x_i}\rangle \leq s_i, \quad \forall i \leq N, \forall k \leq K \\ & \quad \|z_{ik}\|_* \leq \lambda \qquad\qquad \forall i \leq N, \forall k \leq K \end{cases} \\
=&\begin{cases} \inf\limits_{\lambda, s_i, z_{ik}} & \lambda\varepsilon + \frac{1}{N}\sum_{i=1}^N s_i \\ & \text{s.t.} [-l_k + \chi_S]^*(z_{ik}) - \langle z_{ik}, \hat{x_i}\rangle \leq s_i, \quad \forall i \leq N, \forall k \leq K \\ & \quad \|z_{ik}\|_* \leq \lambda \qquad\qquad \forall i \leq N, \forall k \leq K \end{cases} \qquad (22)
\end{aligned}$$

where the equality follows from the definitions of the conjugate function and the characteristic function and the substitution of z_{ik} with $-z_{ik}$. In [48], they show that using the assumption of upper semi-continuity on the loss functions l_k, the inequalities in the proof above actually become equalities, which means the optimal values of (2) and (22) are the same and

$$\begin{aligned}
[-l_k + \chi_S]^*(z_{ik}) &= \inf_{v_{ik}}\left([-l_k]^*(z_{ik} - v_{ik}) + [\chi_S]^*(v_{ik})\right) \\
&= \mathrm{cl}\left[\inf_{v_{ik}}\left([-l_k]^*(z_{ik} - v_{ik}) + \sigma_S(v_{ik})\right)\right],
\end{aligned}$$

where cl[·] denotes the closure operator mapping a function to its larges lower semi-continuous minorant. Finally, by seeing that $\mathrm{cl}[l(x)] \leq 0$ if and only if $l(x) \leq 0$, we have that (2) is equal to (19).

□

5 Wasserstein robustness for Bayesian estimation

In this section, we introduce a new framework for Wasserstein robustness in Bayesian estimation. The remainder of this thesis is dedicated to analyzing the properties of this framework and conducting simulations to support our findings.

Let $A \subset \mathbb{R}^n$ and $B \subset \mathbb{R}^m$ be two measurable sets for $n, m \in \mathbb{N}$. We assume that μ^*_{AB} is a fixed probability measure on the space $A \times B$ and that the true measure μ_{AB} is 'not far' from the fixed distribution μ^*_{AB} in terms of the Wasserstein distance. For a bounded loss $L : A \times B \to \mathbb{R}$ we can define the 'worst-case scenario' as:

$$\sup_{\mu_{AB} \in K : W_p(\mu_{AB}, \mu^*_{AB}) \leq \varepsilon} \int_{A \times B} L(a, b) d\mu_{AB}$$

where:

- $\varepsilon > 0$ is a positive constant,
- $1 \leq p < \infty$,
- and $K \subseteq P(A \times B)$ is weakly* closed, where $P(A \times B)$ represents the set of joint probability measures on A and B.

To compute a robust estimator we seek an unknown estimator g giving us the 'best worst-case scenario' for a bounded loss $l : A \times B \to \mathbb{R}$ parameterized by g, which is found through the following variational objective

$$R_{K,\varepsilon,p} := \inf_{g \in \Sigma} \sup_{\mu_{AB} \in K : W_p(\mu_{AB}, \mu^*_{AB}) \leq \varepsilon} \int_{A \times B} l(a, b; g) d\mu_{AB} \tag{23}$$

where Σ is a finite-dimensional set. This set is often chosen as a parameterized family of functions of given complexity. For example, it could be the family of linear or affine maps or neural networks of a fixed architecture. Analogous to the alternative formulation of the Wasserstein-DRO using the Wasserstein ball, we give the following definition.

Definition 5.1 (Constrained Wasserstein Ball)**.** We define the p-Wasserstein ball constrained to the set K as

$$B_{K,\varepsilon,p}(\mu^*) := \{\mu \in K : W_p(\mu, \mu^*) \leq \varepsilon\}.$$

With this concept we have the final definition for our problem framework.

Definition 5.2 (Constrained Wasserstein-DRO Problem)**.** We define the Constrained Wasserstein-DRO Problem as finding the $g \in \Sigma$ that minimizes the following objective function with a bounded loss function $l : A \times B \to \mathbb{R}$ parameterized by g

$$R_{K,\varepsilon,p} = \inf_{g \in \Sigma} \sup_{\mu_{AB} \in B_{K,\varepsilon,p}(\mu^*_{AB})} \int_{A \times B} l(a, b; g) d\mu_{AB}. \tag{24}$$

Note that equations (23) and (24) are equivalent and can be interchanged.

5.1 Problem variants

By making choices for A, B, Σ, and K in (24) we can recover robust reconstruction frameworks for different types of problems. In [35], the DRO-problem Blanchet et al. consider is basically a specific version of our framework. We can make choices for the spaces we work with to recover their linear regressional setting. Consider the input x_i (or a in our case) as a real vector, the output or measurements y_i (b in our case) as a real scalar and the parameter β (g in our case) as a vector. In other words, in (24), choose $A = \mathbb{R}^d$, $B = \mathbb{R}$ and $\Sigma = \mathbb{R}^d$ and we would have the same input, output, and parameter spaces. Second, they assume a discrete observed probability distribution $\hat{\mathbb{P}}_n$ (corresponding to μ^*_{AB} in our case) which we leave more general and continuous. The last and most

important difference is that for the non-fixed marginal of π, they consider the whole space of joint probability measures to optimize over, while we restrict to an arbitrary subset $K \subseteq P(A \times B)$ which makes the problem generalizable to other settings, like inverse problems. In addition to the choices for the spaces, we can recover a **linear regression problem** by choosing $K = P(A \times B)$. By making the same choices but $B = \{-1, +1\}$ for the output space, and seeing β as a weight vector we can recover a **binary classification problem**.

5.1.1 Inverse problems

In an inverse problem with the input space X and output space Y, given an observation likelihood $\mu^*_{Y|X} : X \to P(Y)$ and a ground-truth distribution $\mu^*_X \in P(X)$, we want to obtain a reconstruction of the ground-truth. We may want a reconstruction that is robust with respect to the whole joint probability space or a reconstruction robust with respect to the noisy distribution of either $\mu^*_{Y|X}$ or μ^*_X. For ease of notation, let $\mu^*_{XY} := \mu^*_{Y|X} \otimes \mu^*_X$. In (24), choose $A = X$, $B = Y$. In an unsupervised setting, so if only μ^*_Y is known instead of $\mu^*_{Y|X}$, the Bayesian framework introduced in the previous section cannot be applied anymore so we consider a supervised setting. In practice, this means the input- and output-data must be paired.

Definition 5.3 (Wasserstein-DRO for inverse problems)**.** If we want to obtain a reconstruction that is **robust with respect to corrupted ground-truth**, we choose $K := \{\mu_X \otimes \mu^*_{Y|X} : \mu_X \in P(X)\}$ and we obtain

$$R^X_{K,\varepsilon,p} := \inf_{g \in \Sigma} \sup_{\mu_X \otimes \mu^*_{Y|X} \in B_{K,\varepsilon,p}(\mu^*_{XY})} \int_X \int_Y l(x, y; g) d\mu^*_{Y|X}(y|x) d\mu_X(x).$$

If on the other hand a reconstruction that is **robust with respect to corrupted measurements** is desired, we choose $K := \{\mu^*_X \otimes \mu_{Y|X} : \mu_{Y|X}$ conditional probabilities$\}$ and we have

$$R^Y_{K,\varepsilon,p} := \inf_{g \in \Sigma} \sup_{\mu^*_X \otimes \mu_{Y|X} \in B_{K,\varepsilon,p}(\mu^*_{XY})} \int_X \int_Y l(x, y; g) d\mu_{Y|X}(y|x) d\mu^*_X(x).$$

Suppose that given data $\{x_1, ..., x_n\}$, we want to choose deterministic distributions. We can take empirical distributions $\mu^*_{Y|X}(x_i) := \delta_{Hx_i}$ where $H : X \to Y$ is some forward operator and $\mu^*_X = \frac{1}{N} \sum_{i=1}^N \delta_{x_i}$. Then $\mu^*_{X,Y} = \mu^*_X \otimes \mu^*_{Y|X} = \frac{1}{N} \sum_{i=1}^n \delta_{(x_i, Hx_i)}$. If we want to model measurements corrupted with additive Gaussian noise, we can choose deterministic $\mu_{Y|X}$ by choosing $K = \{\mu^*_X \otimes \mu_{Y|X} : \mu_{Y|X} \sim N(Hx, \sigma^2), \sigma \in \bar{\mathbb{R}}\}$ with $H : X \to Y$ an operator describing the forward model. For dual representations later on, it will become clear that K must be weakly* closed so it is necessary to include δ_{Hx} in K. For simplicity, we will disregard whether K is closed or not in this section.

We can refine the set K even further if we know the type of noise present in the inverse problem. Various choices can be made based on the specific noise present in the problem. Let x^* be the unknown noise-less ground-truth, $H : X \to Y$ some forward operator and e noise with some distribution.

- **Additive noise.** We model an inverse problem with additive noise in the measurement or input as $y^\delta = Hx^* + e$ or $y^\delta = H(x^* + e)$ respectively. A common choice is to model additive white Gaussian noise (AWGN) on the measurement. We can recover this in the Wasserstein-DRO for inverse problems by choosing $K := \{\mu^*_X \otimes \mu_{Y|X} : \mu_{Y|X} \sim \mathrm{N}(Hx, \sigma^2), \sigma \in \mathbb{R}\}$ or $K := \{\mu_X \otimes \mu^*_{Y|X} : \mu_X \sim \mathrm{N}(\tilde{x}, \sigma^2), \tilde{x}, \sigma \in \mathbb{R}\}$. A simple problem would be to assume our measurement is a noisy image and we want to include only ambiguities in the measurement-space. Then we choose $H = I$ the identity operator and $K := \{\mu^*_X \otimes \mu_{Y|X} : \mu_{Y|X} \sim \mathrm{N}(x, \sigma^2), \sigma \in \mathbb{R}\}$. Instead of Gaussian we can also assume a Laplace-, uniform- or Poisson (with constant parameter) distribution for the noise [39].
- **Multiplicative noise.** In the case of multiplicative noise, we model the inverse problem as $y^\delta = Hx^* \cdot e$ or $y^\delta = H(x^* \cdot e)$ if the noise is applied to the measurement or input, respectively. If we have noise on the measurement given by Gamma distributed random variables we can choose $K := \{\mu^*_X \otimes \mu_{Y|X} : \mu_{Y|X} \sim \mathrm{Gamma}(\alpha, \frac{\beta}{Hx}), \alpha, \beta \in \mathbb{R}\}$, with shape parameter α and scale parameter β. In the case of noise on the input, we choose $K := \{\mu_X \otimes \mu^*_{Y|X} : \mu_X \sim \mathrm{Gamma}(\alpha, \beta), \alpha, \beta \in \mathbb{R}\}$.

- **Noise with general dependency on input or measurement.** Generally, we model an inverse problem as $y^\delta = H^\delta x^*$ with $H^\delta : X \to Y$ a forward operator including the noise dependency. Other than additive or multiplicative, the noise could also be neither of those but dependent the input and/or measurement. If the measurement is done with noisy input we model it as $H^\delta x = H(\delta(x))$ where $\delta : X \to X$ is some noise operator. Or the other way around, with $H^\delta x = \delta(Hx)$, $\delta : Y \to Y$. Take for example Poisson noise, often used to model errors in photon counting which finds application in various fields, including tomography, microscopy and CCD sensors of digital cameras or astronomy systems [39]. The noise is applied via a Poisson process. We view the measurements y_i^δ as stochastic variables having a Poisson distribution with mean $(Hx)_i$ so we model the problem as $y \sim \text{Pois}(Hx)$. In this case we choose $K := \{\mu_X^* \otimes \mu_{Y|X} : \mu_{Y|X} \sim \text{Pois}(Hx)\}$.

Other than K, we can refine the set Σ and the loss function to specify the type of inverse problem we are modeling, as this set can basically be seen as the 'candidate inverses'. For example:

- **Signal processing and deblurring**. When we are trying to recover a sharp image (signal) x^* from a (possibly noisy) blurred image (filtered or distorted signal) y^δ this can be modeled as $y^\delta = H * x^*$ with H a convolution kernel or point spread function and $*$ representing a convolution. To recover this type of problem, one may choose Σ as (a subset of) (de-)convolution operators that work in the frequency domain.

- **Rootfinding**. Given a continuous function $H : \mathbb{R} \to \mathbb{R}$ and $y \in \mathbb{R}$ we want to find x such that $H(x) = y$. This problem is ill-posed when the derivative of H is small near the root. For these types of problem we choose Σ as the continuous functions.

- **Matrix inversion**. If a square matrix H has large condition number $\left\|H^{-1}\right\|\|H\|$, the inversion problem is ill-posed. If H is non-singular we may choose Σ as square matrices with full rank to find the inverse of H. If we have more knowledge on the operator H we can incorporate this into Σ. For example, if H is diagonal, we can take the subset of matrices in Σ that are diagonal. If $H \in \mathbb{R}^{m\times n}$ is singular, we may choose $\Sigma \subseteq \mathbb{R}^{n\times m}$. Knowledge about the sparse elements in the forward operator can help us find a suitable subset of Σ.

- **Wavefield imaging**. Wavefield imaging uses acoustic or electromagnetic waves emitted or reflected by an object of interest to find information about that object such as its location or size which is used in radar, ultrasound or seismic imaging. The simplest forms of wavefield imaging problems have measurements $y_i(t) = v(t, x_i), i \in \{1, 2, ..., n\}$ where the underlying physics are described be some wave equation. An inverse source problem tries to reconstruct the source term $q(t, x)$ from the wavefield $v(t, x)$ and assumes a known and constant $c(x) = c_0$ whereas an inverse medium problem tries to reconstruct the speed of propagation $c(x)$. An example would be to consider the operator as a convolution with the Green function in which case we could choose Σ again as (a subset of) (de-)convolution operators in the frequency domain.

- **X-ray tomography**. The Radon transform describes all possible X-ray measurements of a two-dimensional image where the measurement (a sinogram) is calculated by taking line-integrals along straight lines at various angles and shifts. To compute its inverse precisely, we would need a large amount of angles and shifts which is undesirable in practice. In this case, we can again choose Σ as a specific subset of (de-)convolution operators in the frequency domain.

5.1.2 Other problem variants

The framework introduced in this thesis can be used to model all kinds of problems so this chapter is in no way exhaustive. We list two more potential problems which could incorporate our proposed framework.

Neural network classifiers. Let $B = \{1, ..., N\}$ be labels, and $A \subset \mathbb{R}^n$ a given dataset. Let $\mu_A^* \in P(A)$ be a ground-truth data distribution and $\mu_{B|A} : A \to \mathbb{P}(B)$ a stochastic classification method. If we let g be parameterized by θ, we have $g_\theta : A \to B$ parameterized by a neural network with weights in Σ. We can make choices for K similar to the previous section to obtain a framework robust to corruptions in either the ground-truth or the classification method distribution.

Distributionally Robust Maximum Likelihood Estimation In Maximum Likelihood Estimation (MLE), we are looking to estimate parameters of a probability distribution from data. Suppose we have independent training samples $\hat{x}_i \in X, i \in [N]$ and are looking to estimate the mean vector $\mu \in \mathbb{R}^m$ and the precision matrix X, which is the inverse of the covariance matrix $\Delta \in \mathbb{S}_+^m$ of a random vector $x \in \mathbb{R}^m$, where $\mathbb{S}_+^m$ denotes the space of positive-definite matrices of size $m \times m$. [7]. We can recover this setting by choosing $\Sigma = \mathbb{R}^m \times \mathbb{S}_+^m$ and $K = P(X)$.

6 Dual representation

In the previous section, we have introduced the Wasserstein-DRO for inverse problems. However, before delving into specific applications, we show a general strong duality result for the constrained Wasserstein-DRO problem, which will subsequently be adapted for use in inverse problems.

Let I represent the inner supremum in (24), define $\mu_{XY} := \mu_X^* \otimes \mu_{Y|X}$ for simpler notation, choose the 1-Wasserstein distance and suppress the subscript of p in the Wasserstein ball, i.e.

$$I := \sup_{\mu_{XY} \in B_{K,\varepsilon}(\mu_{XY}^*)} \int_{X \times Y} l(x, y; g) d\mu_{XY}. \tag{25}$$

We will continue to choose the 1-Wasserstein distance and suppress the subscript of p in the Wasserstein ball in the rest of this thesis, unless stated otherwise.

In this section, we focus on the quantity I and refer to it as the primal problem, as it represents an infinite-dimensional optimization problem. Thus, we seek a dual problem that admits a finite-dimensional reformulation that facilitates easier evaluation. Recall that Blanchet et al. [35] have a similar but less general problem formulation. In [44], they prove a strong duality result for the inner supremum which considers general space S (corresponding to $X \times Y$ in our case) and loss function $f \in L^1(d\mu)$ (corresponding to $l \in L^1(d\mu_{XY}^*)$ in our case). We will follow a similar approach as Blanchet et al in [44] to get to a duality result. The contents of this section are thus highly inspired by their methods but altered to fit our (more general) problem framework. As we ignore the outer infimum of (24) for now, the only difference with our case remains the set K: we restrict ourselves to an unspecified subset of probabilities measures on $X \times Y$ and Blanchet et al. consider all joint probabilities. This complicates the road to strong duality as we do not consider the entire joint probability space but a subset of it.

6.1 Primal problem

To simplify our notation, let $S := X \times Y$, $\bar{x} = (x, y)$, $\bar{u} = (u, v)$ where $x, u \in X$, $y, v \in Y$. We define two ambiguity sets as

$$\Phi_{\mu_{XY}^*, K} := \left\{ \pi \in P(S \times S) : \pi \in \bigcup_{\nu \in K} \Pi(\mu_{XY}^*, \nu) \right\}$$

and

$$\Phi_{\mu_{XY}^*, K, \varepsilon} := \left\{ \pi \in P(S \times S) : \pi \in \bigcup_{\nu \in K} \Pi(\mu_{XY}^*, \nu), \int_{S \times S} c(\bar{x}, \bar{u}) d\pi(\bar{x}, \bar{u}) \leq \varepsilon \right\}$$

where we leave K an unspecified subset of $P(S)$. Recognize the first set as all joint probabilities in $S \times S$ that have μ_{XY}^* as their first marginal and any $\nu \in K$ as their second marginal. The second set can then be recognized as a subset of the first, including only those that are contained within the Wasserstein ball.
We use the ambiguity set to define the objective function as

$$I(\pi) := \int_{S \times S} l(\bar{u}) d\pi(\bar{x}, \bar{u})$$

where we ignore the g-parameter in the loss function for the time being. This lets us reformulate our **primal problem** as

$$I = \sup_{\pi \in \Phi_{\mu_{XY}^*, K, \varepsilon}} I(\pi). \tag{26}$$

Finally, assume:

- **(A1)** $c : S \times S \to \mathbb{R}_+$ to be non-negative lower semi-continuous s.t. $c(x, y) = 0$ if and only if $x = y$ and
- **(A2)** $l \in L^1(d\mu_{XY}^*)$ is upper semi-continuous.

6.2 Dual problem and weak duality

For the same reasons as in Section 4.2.2, let $m_{\mathcal{U}}(S;\mathbb{R})$ denote the collection of measurable functions $\phi : (S,\mathcal{U}(S)) \to (\bar{\mathbb{R}},\mathcal{B}(\bar{\mathbb{R}}))$ where $\mathcal{U}(S) = \cap_{\mu\in P(S)}\mathcal{B}_\mu(S)$ is the universal σ-algebra. Define the set

$$\Lambda_{c,l} := \Bigg\{(\lambda,\phi,\psi) : \lambda \in \mathbb{R}_+, \phi,\psi \in m_{\mathcal{U}}(S;\mathbb{R}), \int_{S\times S}(\phi(\bar{x})+\psi(\bar{u}))d\pi(\bar{x},\bar{u}) \\ \geq \int_{S\times S}(l(\bar{u})-\lambda c(\bar{x},\bar{u}))d\pi(\bar{x},\bar{u}) \quad \forall \pi \in \Phi_{\mu^*_{XY},K}\Bigg\}.$$

For such $(\lambda,\phi,\psi) \in \Lambda_{c,l}$, consider

$$J(\lambda,\phi,\psi) := \lambda\varepsilon + \sup_{\pi\in\Phi_{\mu^*_{XY},K}}\int_{S\times S}(\phi(\bar{x})+\psi(\bar{u}))d\pi(\bar{x},\bar{u}).$$

Finally, let

$$J := \inf_{(\lambda,\phi,\psi)\in\Lambda_{c,l}} J(\lambda,\phi,\psi) \tag{27}$$

which we refer to as the **dual problem**.

Theorem 6.1 (Weak duality)**.** *Assume (A1) and (A2) hold and S is a compact, Polish space. We have $J \geq I$ where I and J are defined as in* (26) *and* (27)*, respectively.*

Proof. We have that $\int l d\mu^*_{XY}$ is finite and $\int(\phi+\psi)d\pi \geq \int l d\pi$ for every $(\lambda,\phi,\psi) \in \Lambda_{c,l}, \pi \in \Phi_{\mu^*_{XY},K}$. Thus, the integral in the definition of $J(\lambda,\phi,\psi)$ avoids ambiguities such as $\infty-\infty$ for any $(\lambda,\phi,\psi) \in \Lambda_{c,l}, \pi \in \Phi_{\mu^*_{XY},K}$.

For any $\pi \in \Phi_{\mu^*_{XY},K,\varepsilon}$ and $(\lambda,\phi,\psi) \in \Lambda_{c,l}$:

$$\begin{aligned}\lambda\varepsilon + \int_{S\times S}(\phi(\bar{x})+\psi(\bar{u}))d\pi(\bar{x},\bar{u}) &\geq \lambda\varepsilon + \int_{S\times S}(l(\bar{u})-\lambda c(\bar{x},\bar{u}))d\pi(\bar{x},\bar{u}) \\ &= \int_{S\times S} l(\bar{u})d\pi(\bar{x},\bar{u}) + \lambda\left(\varepsilon - \int_{S\times S} c(\bar{x},\bar{u})d\pi(\bar{x},\bar{u})\right) \\ &\geq \int_{S\times S} l(\bar{u})d\pi(\bar{x},\bar{u}) \\ &= I(\pi)\end{aligned}$$

where the second equality follows from the assumption that $(\lambda,\phi,\psi) \in \Lambda_{c,l}$ and $\pi \in \Phi_{\mu^*_{XY},K,\varepsilon}$ and the fifth inequality follows from the fact that $\int c d\pi \leq \varepsilon$, $\forall \pi \in \Phi_{\mu^*_{XY},K,\varepsilon}$.

Taking the supremum with respect to π we obtain

$$\begin{aligned}J(\lambda,\phi,\psi) &= \sup_{\pi\in\Phi_{\mu^*_{XY}}} \lambda\varepsilon + \int_{S\times S}(\phi(\bar{x})+\psi(\bar{u}))d\pi(\bar{x},\bar{u}) \\ &\geq \sup_{\pi\in\Phi_{\mu^*_{XY},K,\varepsilon}} \lambda\varepsilon + \int_{S\times S}(\phi(\bar{x})+\psi(\bar{u}))d\pi(\bar{x},\bar{u}) \\ &\geq \sup_{\pi\in\Phi_{\mu^*_{XY},K,\varepsilon}} I(\pi)\end{aligned}$$

So, taking now the infimum in $\Lambda_{c,l}$ we conclude that

$$J := \inf_{(\lambda,\phi,\psi)\in\Lambda_{c,l}} J(\lambda,\phi,\psi) \geq \sup_{\pi\in\Phi_{\mu^*_{XY},K,\varepsilon}} I(\pi) = I.$$

□

We have weak duality and will refer to J as the dual problem. The question now remains if we have strong duality, i.e. $J = I$?

6.3 Strong duality

Before presenting the strong duality result, we need to define certain sets and functionals, verify their convexity and introduce a lemma, all in preparation for the application of the Fenchel Duality Theorem [45, p. 201].

Define the sets of functions

$$C := \Bigg\{ g \in X : \exists \phi, \psi \in C_b(S), \lambda \geq 0 \text{ s.t.} \\ \int_{S\times S} g(\bar{x},\bar{u}) d\pi(\bar{x},\bar{u}) = \int_{S\times S} (\phi(\bar{x}) + \psi(\bar{u}) + \lambda c(x,u)) d\pi(\bar{x},\bar{u}) \quad \forall \pi \in \Phi_{\mu^*_{XY},K} \Bigg\} \tag{28}$$

and

$$D := \left\{ g \in X : \int_{S\times S} g(\bar{x},\bar{u}) d\pi(\bar{x},\bar{u}) \geq \int_{S\times S} l(\bar{u}) d\pi(\bar{x},\bar{u}), \quad \forall \pi \in \Phi_{\mu^*_{XY},K} \right\}. \tag{29}$$

Notice that both of these sets are convex. To see this, let $g_1, g_2 \in C$, $\gamma \in [0,1]$ and define $g_3 := \gamma g_1 + (1-\gamma) g_2$ and see that for every $\pi \in \Phi_{\mu^*_{XY},K}$:

$$\begin{aligned} \int g_3 d\pi &= \int \gamma g_1 d\pi + \int (1-\gamma) g_2 d\pi \\ &= \int \left([\gamma\phi_1 + (1-\gamma)\phi_2] + [\gamma\psi_1 + (1-\gamma)\psi_2] + c[\gamma\lambda_1 + (1-\gamma)\lambda_2] \right) d\pi \\ &= \int (\phi_3 + \psi_3 + \lambda_3 c) d\pi \end{aligned}$$

and thus $g_3 \in C$. Define g_3, γ analogous for D, but now let $g_1, g_2 \in D$. Then for every $\pi \in \Phi_{\mu^*,K}$:

$$\int g_3 d\pi \geq \gamma \int l d\pi + (1-\gamma) \int l d\pi = \int l d\pi$$

so $g_3 \in D$. Thus, both C and D are convex.

Define the functionals $\Theta : C \to \mathbb{R}$ and $\Gamma : D \to \mathbb{R}$ respectively as

$$\Theta(g) := \inf_{(\lambda,\phi,\psi)\in A(g)} \left\{ \lambda\varepsilon + \sup_{\pi \in \Phi_{\mu^*_{XY},K}} \int_{S\times S} (\phi(\bar{x}) + \psi(\bar{u})) d\pi(\bar{x},\bar{u}) \right\} \quad \text{and} \tag{30}$$

$$\Gamma(g) := 0. \tag{31}$$

with

$$A(g) := \Bigg\{ (\lambda,\phi,\psi) : \ \phi,\psi \in C_b(S), \lambda \geq 0 \\ \int_{S\times S} g(\bar{x},\bar{u}) d\pi(\bar{x},\bar{u}) = \int_{S\times S} (\phi(\bar{x}) + \psi(\bar{u}) + \lambda c(x,u)) d\pi(\bar{x},\bar{u}) \quad \forall \pi \in \Phi_{\mu^*_{XY},K} \Bigg\}$$

It is simple to see that A is convex, since $A(tg_1 + (1-t)g_2) = tA(g_1) + (1-t)A(g_2)$ for $g_1, g_2 \in C$ and $t \in [0,1]$. To show Θ is convex, recall

$$J(\lambda,\phi,\psi) := \lambda\varepsilon + \sup_{\pi \in \Phi_{\mu^*_{XY},K}} \int_{S\times S} (\phi(\bar{x}) + \psi(\bar{u})) d\pi(\bar{x},\bar{u})$$

which we recognize to be convex: $J(t(\lambda_1,\phi_1,\psi_1)+(1-t)(\lambda_2,\phi_2,\psi_2)) \leq tJ(\lambda_1,\phi_1,\psi_1)+(t-1)J(\lambda_2,\phi_2,\psi_2)$. We can rewrite

$$\Theta(g) := \inf_{(\lambda,\phi,\psi)\in A(g)} J(\lambda,\phi,\psi).$$

Now let $(\lambda_1, \phi_1, \psi_1)$ and $(\lambda_2, \phi_2, \psi_2)$ realize the infima for $g_1, g_2 \in C$ respectively, i.e.

$$\Theta(g_1) := \inf_{(\lambda,\phi,\psi)\in A(g_1)} J(\lambda, \phi, \psi) = J(\lambda_1, \phi_1, \psi_1),$$
$$\Theta(g_2) := \inf_{(\lambda,\phi,\psi)\in A(g_2)} J(\lambda, \phi, \psi) = J(\lambda_2, \phi_2, \psi_2).$$

Now consider

$$\Theta(tg_1 + (1-t)g_2) = \inf_{(\lambda,\phi,\psi)\in A(tg_1+(1-t)g_2)} J(\lambda, \phi, \psi).$$

Since $A(g)$ is convex we can construct a point

$$(\lambda_3, \phi_3, \psi_3) := t(\lambda_1, \phi_1, \psi_1) + (1-t)(\lambda_2, \phi_2, \psi_2) \in A(tg_1 + (1-t)g_2).$$

By convexity of J,

$$J(\lambda_3, \phi_3, \psi_3) = tJ(\lambda_1, \phi_1, \psi_1) + (t-1)J(\lambda_2, \phi_2, \psi_2)$$

and thus

$$\begin{aligned}\Theta(tg_1 + (1-t)g_2) &= \inf_{(\lambda,\phi,\psi)\in A(tg_1+(1-t)g_2)} J(\lambda, \phi, \psi) \\ &\leq J(\lambda_3, \phi_3, \psi_3) \\ &= tJ(\lambda_1, \phi_1, \psi_1) + (t-1)J(\lambda_2, \phi_2, \psi_2) \\ &= t\Theta(g_1) + (1-t)\Theta(g_2)\end{aligned}$$

and thus we have shown that Θ is convex. Finally, we remark that Γ is concave.

Lemma 6.2. *Let $S \times S$ be a compact Polish space, and assume (A1) and (A2) hold. Define C and D as in* (28) *and* (29) *respectively. Their intersection $C \cap D$ contains at least one point in the relative interior of C and D. Finally, the epigraph of Γ over D has non-empty interior.*

Proof. Denote by $ri(A)$ the relative interior of a set A. We will show there exists an $h \in C \cap D$ such that $h \in ri(C) \cap ri(D)$. We take $h(x,u) = c(x,u) + \sup_{x\in S}\{l(x)\}$. We can choose $\phi(x) := c(x,x) + \sup_{x\in S}\{l(x)\}$, $\psi(u) := 0$ and $\lambda = \frac{c(x,u)-c(x,x)}{c(x,u)}$ then $\phi(x) + \psi(u) + \lambda c(x,u) = h(x,u)$ and clearly $\int \phi + \psi + \lambda c d\pi = \int h d\pi \quad \forall \pi \in \Phi_{\mu^*_{XY},K}$ so $h \in C$. Clearly $h \in D$ since $h(x,u) \geq l(u)$ for every u and thus $\int h d\pi \geq \int l d\pi \quad \forall \pi \in \Phi_{\mu^*_{XY},K}$. Consequently, $h \in C \cap D$ and we will now show it is an element of the relative interior of each set. Since both C and D are convex, we must show that for every $g \in C$ there exists a $\gamma > 1$ such that $\gamma h + (1-\gamma)g \in C$, and analogous for D.

Let $\gamma > 1$ and $g \in C$ arbitrary. Define $f(x,u) := \gamma h(x,u) + (1-\gamma)g(x,u)$. Then for every $\pi \in \Phi_{\mu^*,K}$,

$$\begin{aligned}\int f(x,u)d\pi &= \int \gamma(c(x,u) + \sup_{x\in S}\{l(x)\})d\pi + \int (1-\gamma)g(x,u)d\pi \\ &= \int \gamma(c(x,u) + \sup_{x\in S}\{l(x)\})d\pi + \int (1-\gamma)(\phi(x) + \psi(u) + \lambda c(x,u))d\pi \\ &= \int [(1-\gamma)\phi(x) + \psi(u) + \gamma \sup_{x\in S}\{l(x)\} + (\gamma - \gamma\lambda - \lambda)c(x,u)]d\pi \\ &= \int [\phi_2(x) + \psi_2(u) + \lambda_2 c(x,u)]d\pi\end{aligned}$$

where we define $\phi_2(x) := (1-\gamma)\phi(x) + \sup_{x\in S}\{l(x)\}$, $\psi_2(u) := (1-\gamma)\psi(u)$ and $\lambda_2 := \gamma - \gamma\lambda + \lambda$. Thus $f \in C$ which means $h \in ri(C)$.

Now let $\gamma > 1$ and $g \in D$ arbitrary. Define similarly $f(x,u) := \gamma h(x,u) + (1-\gamma)g(x,u)$. Then for every $\pi \in \Phi_{\mu^*,K}$,

$$\begin{aligned}\int f(x,u)d\pi &= \int \gamma(c(x,u) + \sup_{x\in S}\{l(x)\})d\pi + \int (1-\gamma)g(x,u)d\pi \\ &\geq \int (\gamma(c(x,u) + l(u))d\pi + \int (1-\gamma)l(u)d\pi \\ &= \int (\gamma c(x,u) + l(u))d\pi \geq \int l(u)d\pi\end{aligned}$$

and thus $f \in D$ which means $h \in ri(D)$.

To show $[\Gamma, D]$ has non-empty interior, we need only show that D has non-empty interior. Let $g \in D$, $\varepsilon > 0$ and let $h = g + \varepsilon$ which is also in D since $\int h d\pi = \int g + \varepsilon d\pi \geq \int l d\pi + \varepsilon \geq \int l d\pi$ where we use that π is a probability measure so $\int \varepsilon d\pi = \varepsilon$. Take $\eta \in X$ such that $\|h - \eta\|_\infty \leq \varepsilon/2$. We have

$$\varepsilon/2 > \|h-\eta\|_\infty = \sup_{x,u\in S} |h(x,u) - \eta(x,u)| = |g(x,u) + \varepsilon - \eta(x,u)| \geq g(x,u) + \varepsilon - \eta(x,u) \quad \forall x,u.$$

Then we have $\eta(x,u) > g(x,u) + \varepsilon/2$ and finally

$$\int \eta d\pi > \int (g + \varepsilon/2)d\pi \geq \int l d\pi + \epsilon/2 > \int l d\pi$$

and thus $\eta \in int(D)$, so D has non-empty interior. □

Lemma 6.3. *Let $S \times S$ be a compact Polish space, $l : S \to \mathbb{R}$ upper semi-continuous and D defined as in* (29)*. It holds that*

$$\inf_{g\in D} \int g d\pi = -\infty$$

if and only if π is not non-negative.

Proof. Clearly if π is non-negative then $\inf_{g\in D} \int g d\pi > 0 > -\infty$.

Now suppose π is not non-negative, then we can use the Jordan decomposition $\pi = \pi^+ - \pi^-$ of a positive measure π^+ and a negative measure π^- such that $\pi^+(A) = 0 < \pi^-(A) < \infty$ for some $A \in \mathcal{B}(S \times S)$. Any Borel measure on a Polish space is regular which means that for any finite measure μ on $\mathcal{B}(S \times S)$,

$$\begin{aligned}\mu(A) &= \sup\{\mu(C) \mid C \subseteq A, C \text{ compact}\} \\ \text{and } \mu(A) &= \inf\{\mu(O) \mid A \subseteq O, O \text{ open}\}.\end{aligned}$$

Thus given $\delta > 0$, there exists a compact set $C_\delta \subseteq A$ and an open set $O_\delta \supseteq A$ such that $\pi^-(O_\delta) - \delta \leq \pi^-(A) \leq \pi^-(C_\delta) + \delta$. Moreover, since $0 = \pi^+(A) \geq \pi^+(O) + \delta$, we have $\pi^+(O_\delta) \leq \delta$.

Since $S \times S$ is compact, we can use Urysohn's lemma [49, Th.10.8] to see that there exists a continuous function $h : S \times S \to [0,1]$ such that $h(x,y) = 1$ for all $x \in C_\delta$ and $h(x,y) = 0$ for all $x \notin O_\delta$. Note that since l is upper semi-continuous and S is compact, we have $\sup_{x\in S} l(x) < \infty$. Also, by choosing $\delta \leq \pi^-(A)/2$, we have

$$\int h d\pi = \int h d\pi^+ - \int h d\pi^- \leq \pi^+(O_\delta) - \pi^-(C_\delta) \leq 2\delta - \pi^-(A) < 0$$

where the second (in-)equality we use that $\int h d\pi^-$ is bounded from below by $\pi^-(C)$ and $\int h d\pi^+$ is bounded from above by $\pi^+(O)$. Combining these facts, we have that $\inf_{n\geq 1} \int g_n d\pi = -\infty$ for the sequence of continuous functions $g_n(x,y) = nh(x,y) + \sup_{x\in S} l(x)$. As $g_n(x,y) \geq l(y)$ for all $x, y \in S$, also $\int g_n d\pi \geq \int l d\pi \ \forall \pi \in \Phi_{\mu^*_{XY},K}$. It follows that $\inf_{g\in D} \int g d\pi \leq \inf_{n\geq 1} g_n d\pi = -\infty$. □

Finally, we have all the ingredients to state our strong duality result.

Theorem 6.4. *Assume (A1) and (A2) hold, S is a compact Polish space and $c : S \times S \to \mathbb{R}_+$ is continuous. Finally, assume K is weakly* closed.*

If the assumptions hold, then strong duality holds: $J = I < \infty$, where I and J are defined as in (26) *and* (27)*, respectively.*

Proof. To prepare for the application of the Fenchel duality theorem, we will define the functionals and spaces of interest and identify their conjugates. Define the sets of functions C and D as in (28) and (29) respectively. Define the functionals $\Theta : C \to \mathbb{R}$ and $\Gamma : D \to \mathbb{R}$ as in (30) and (31), respectively. We let $X = C_b(S \times S)$ and recognize its topological dual $X^* = M(S \times S)$ which represent the vector space of bounded continuous functions equipped with the supremum norm and finite Borel measures on $S \times S$ equipped with the total variation norm, respectively.

First, note that the infimum for Θ guarantees an invertible relationship between every $g \in C$ and the triple (λ, ϕ, ψ), i.e. they uniquely define each other. We are interested in

$$\inf_{g \in C \cap D} \{\Theta(g) - \Gamma(g)\} = \inf_{\substack{g \in C \\ (\lambda,\phi,\psi) \in A(g)}} \left\{ J(\lambda, \phi, \psi) : \int_{S \times S} (\phi(\bar{x}) + \psi(\bar{u}) + \lambda c(x, u)) d\pi(\bar{x}, \bar{u}) \right.$$
$$\left. \geq \int_{S \times S} l(\bar{u}) d\pi(\bar{x}, \bar{u}) \quad \forall \pi \in \Phi_{\mu^*_{XY}, K} \right\}.$$

Next, we want to identify the conjugate functionals $\Theta^* : C^* \to \mathbb{R}$ and $\Gamma^* : D^* \to \mathbb{R}$ and their respective domains C^* and D^*. By definition of the conjugate functional,

$$C^* = \left\{ \pi \in X^* : \sup_{g \in C} \left\{ \int g d\pi - \Theta(g) \right\} < \infty \right\} \quad \text{and} \quad D^* = \left\{ \pi \in X^* : \inf_{g \in D} \int g d\pi < -\infty \right\},$$
$$\Theta^*(\pi) := \sup_{g \in C} \left\{ \int g d\pi - \Theta(g) \right\} \quad \text{and} \quad \Gamma^*(\pi) := \inf_{g \in D} \int g d\pi.$$

To determine C^* and Θ^*, see that $\forall \pi \in M(S \times S)$,

$$
\begin{aligned}
\Theta^*(\pi) &= \sup_{g\in C}\left\{\int g d\pi - \Theta(g)\right\} \\
&= \sup_{\substack{g\in C \\ (\lambda,\phi,\psi)\in A(g)}} \left\{ \int_{S\times S} (\phi(\bar{x})+\psi(\bar{u})+\lambda c(\bar{x},\bar{u}))d\pi(\bar{x},\bar{u}) \right. \\
&\qquad \left. - \inf_{\substack{g\in C \\ (\lambda,\phi,\psi)\in A(g)}} \left\{ \lambda\varepsilon + \sup_{\tilde{\pi}\in\Phi_{\mu^*_{XY},K}} \int_{S\times S} (\phi(\bar{x})+\psi(\bar{u}))d\tilde{\pi}(\bar{x},\bar{u}) \right\} \right\} \\
&= \sup_{\substack{g\in C \\ (\lambda,\phi,\psi)\in A(g)}} \left\{ \int_{S\times S} (\phi(\bar{x})+\psi(\bar{u})+\lambda c(\bar{x},\bar{u}))d\pi(\bar{x},\bar{u}) \right. \\
&\qquad \left. + \sup_{\substack{g\in C \\ (\lambda,\phi,\psi)\in A(g)}} \left\{ -\lambda\varepsilon - \sup_{\pi\in\Phi_{\mu^*_{XY},K}} \int_{S\times S} (\phi(\bar{x})+\psi(\bar{u}))d\pi(\bar{x},\bar{u}) \right\} \right\} \\
&= \sup_{\substack{g\in C \\ (\lambda,\phi,\psi)\in A(g)}} \left\{ \int_{S\times S} (\phi(\bar{x})+\psi(\bar{u})+\lambda c(\bar{x},\bar{u}))d\pi(\bar{x},\bar{u}) \right. \\
&\qquad \left. - \lambda\varepsilon - \sup_{\tilde{\pi}\in\Phi_{\mu^*_{XY},K}} \int_{S\times S} (\phi(\bar{x})+\psi(\bar{u}))d\tilde{\pi}(\bar{x},\bar{u}) \right\} \\
&= \sup_{\substack{g\in C \\ (\lambda,\phi,\psi)\in A(g)}} \left\{ \lambda\left(\int_{S\times S} c(\bar{x},\bar{u})d\pi(\bar{x},\bar{u}) - \varepsilon\right) \right. \\
&\qquad \left. - \left(\int_{S\times S} (\phi(\bar{x})+\psi(\bar{u}))d\pi(\bar{x},\bar{u}) - \sup_{\tilde{\pi}\in\Phi_{\mu^*_{XY},K}} \int_{S\times S} (\phi(\bar{x})+\psi(\bar{u}))d\tilde{\pi}(\bar{x},\bar{u})\right) \right\} \\
&= \sup_{\substack{g\in C \\ (\lambda,\phi,\psi)\in A(g)}} \left\{ \lambda\left(\int_{S\times S} c(\bar{x},\bar{u})d\pi(\bar{x},\bar{u}) - \varepsilon\right) \right. \\
&\qquad \left. - \left(\int_{S\times S} (\phi(\bar{x})+\psi(\bar{u}))d\pi(\bar{x},\bar{u}) - \sup_{\tilde{\mu}\in K} \int_{S\times S} (\phi(\bar{x})+\psi(\bar{u}))d\mu^*_{XY}(\bar{x})d\tilde{\mu}(\bar{u})\right) \right\} \\
&= \begin{cases} 0 & \text{if } \int cd\pi \le \varepsilon, \pi(Q\times S) = \mu^*_{XY}(Q)\ \forall Q\in\mathcal{B}(S) \text{ and } \pi(S\times R) = \mu_{XY}\ \forall R\in\mathcal{B}(S) \\ & \quad \text{for some } \mu_{XY}(R)\in K \\ \infty & \text{otherwise} \end{cases}
\end{aligned}
$$

where the third equality follows from changing the infimum to a supremum and the fourth equality removes the second supremum as it is redundant. The fifth equality uses that we take the supremum over $\tilde{\pi} \in \Phi_{\mu^*_{XY},K}$ which means the first marginal of $\tilde{\pi}$ is μ^*_{XY}. Thus, we are left with taking the supremum over $\tilde{\mu} \in K$. The last equality uses that for the second term to be 0, we need π and $\tilde{\pi}$ to have the same marginals. As K is assumed to be weakly* closed, the second marginal is contained within K. Therefore, π has μ^*_{XY} and some $\mu_{XY} \in K$ as its marginals.

Thus, we have

$$
\begin{aligned}
C^* = \Bigg\{ \pi \in M(S\times S) : \int cd\pi \le \varepsilon,\ \pi(Q\times S) = \mu^*_{XY}\ \forall Q\in\mathcal{B}(S) \\
\text{and } \pi(S\times R) = \mu_{XY}\ \forall R\in\mathcal{B}(S) \text{ for some } \mu_{XY}(R)\in K \Bigg\}
\end{aligned}
$$

and $\Theta^* = 0$.

To determine D^*, we use Lemma 6.3, which states that $\inf_{g\in D}\int g d\pi = -\infty$ whenever $\pi \in M(S\times S)$ is not non-negative. If it is non-negative, then

$$\inf_{g\in D}\left\{\int g d\pi : \int g d\pi \geq \int l d\pi, \quad \forall \pi \in \Phi_{\mu^*_{XY},K}\right\} = \int l d\pi$$

as l is upper semi-continuous and bounded from above. Thus it can be approximated pointwise by a monotonically decreasing sequence of continuous functions, then the equality follows by the monotone convergence theorem. Thus we have

$$D^* = \left\{\pi \in M_+(S\times S) : \int l d\pi > -\infty\right\} \text{ and } \Gamma^*(\pi) = \int l d\pi.$$

Then

$$\Gamma^*(\pi) - \Theta^*(\pi) = \int l d\pi$$
$$\text{on } C^* \cap D^* = \left\{\pi \in \cup_{\nu\in K}\Pi(\mu^*_{XY},\nu) : \int c d\pi \leq \varepsilon, \int l d\pi > -\infty\right\}.$$

Since I is defined to equal $\sup\left\{\int l d\mu_{XY} : W(\mu_{XY},\mu^*_{XY}) \leq \varepsilon, \int l d\mu_{XY} > -\infty\right\}$, it follows that

$$\sup_{\pi\in C^*\cap D^*}\{\Gamma^*(\pi) - \Theta^*(\pi)\} = I.$$

By Lemma 6.2, C and D are convex, the set $C\cap D$ contains points in the relative interiors of C and D and the epigraph of the function Γ over D has non-empty interior. Thus, we can apply the Fenchel duality Theorem [45, p.201]. By consequence of the mentioned theorem,

$$\inf_{g\in C\cap D}\{\Theta(g) - \Gamma(g)\} = \sup\{\Gamma^*(\pi) - \Theta^*(\pi) : \pi \in C^*\cap D^*\}$$

where the supremum on the right hand side is achieved by some $\pi^* \in \Phi_{\mu^*_{XY},K,\varepsilon}$. We can rewrite,

$$\inf_{(\lambda,\phi,\psi)\in A}\left\{J(\lambda,\phi,\psi) : \int(\phi+\psi+\lambda c)d\pi \geq \int l d\pi \quad \forall \pi \in \Phi_{\mu^*_{XY},K,\varepsilon}\right\} = \max_{\pi\in\Phi_{\mu^*_{XY},K,\varepsilon}} I(\pi) =: I.$$

Since $C_b(S) \subseteq m_{\mathcal{U}}(S;\bar{R})$,

$$J \leq \inf_{(\lambda,\phi,\psi)\in A}\left\{J(\lambda,\phi.\psi) : \int(\phi+\psi+\lambda c)d\pi \geq \int l d\pi \quad \forall \Phi_{\mu^*_{XY},K,\varepsilon}\right\} = I.$$

Due to weak duality we have $J \geq I$, therefore, $J = I$ and we have strong duality.

□

6.4 Finite-dimensional reduction of dual

Currently, our dual formulation optimizes over $\phi,\psi \in m_{\mathcal{U}}(S;\mathbb{R})$ and is thus not yet finite-dimensional. Ideally, we would like to find an explicit expression for ϕ and ψ, parameterized by λ. Recall that in [44], Blanchet et al. optimize (λ,ϕ) over the collection of pairs

$$\Delta_{c,l} := \{(\lambda,\phi) : \lambda \in \mathbb{R}_+, \phi \in m_{\mathcal{U}}(S;\mathbb{R}), \phi(x) \geq l(u) - \lambda c(x,u) \quad \forall x,u \in S\}$$

and they find $\inf_{(\lambda,\phi)\in\Delta_{c,l}} J(\lambda,\phi) = \inf_{\lambda\geq 0} J(\lambda,\phi_\lambda)$ with $\phi_\lambda := \sup_u\{l(u)-\lambda c(x,u)\}$, a finite-dimensional problem. In order to evaluate our dual problem $J := \inf_{(\lambda,\phi,\psi)\in\Lambda_{c,l}} J(\lambda,\phi,\psi)$, we would like to find similar expressions for ϕ and ψ. However with K unspecified, there exists little hope to find a general finite-dimensional expression for J. We can demonstrate however, that when $K = P(S)$, the optimizers are $\phi = \sup_u\{l(u) - \lambda c(x,u)\}$ and $\psi = 0$. Thus verifying that for $K = P(S)$ our problem is equivalent to the one presented in [44], i.e. an unconstrained Wasserstein-DRO. In the next section, we will explore an inverse problem case with a constrained K.

Theorem 6.5 (Finite-dimensional reduction for $K = P(S)$)**.** *Let $K = P(S)$ and $\phi_\lambda := \sup_u\{l(u) - \lambda c(x,u)\}$. Then,*

$$\inf_{(\lambda,\phi,\psi)\in\Lambda_{c,l}} J(\lambda,\phi,\psi) = \inf_{\lambda\geq 0} J(\lambda,\phi_\lambda,0).$$

Proof. We want to show

$$\begin{aligned}\inf_{(\lambda,\phi,\psi)\in\Lambda_{c,l}} &\left\{\lambda\varepsilon + \sup_{\pi\in\Phi_{\mu^*_{XY},K}} \int (\phi(x)+\psi(u))d\pi(x,u)\right\} \\ &= \inf_{\lambda\in\mathbb{R}_+}\left\{\lambda\varepsilon + \int \sup_u\{l(u)-\lambda c(x,u)\}d\mu^*\right\}.\end{aligned}$$

First, see that

$$\inf_{(\lambda,\phi)\in\Lambda_{c,l}} \left\{\lambda\varepsilon + \sup_{\pi\in\Phi_{\mu^*_{XY},K}} \int (\phi(x)+\psi(u))d\pi(x,u)\right\} \tag{32}$$

$$\leq \inf_{(\lambda,\phi=\phi_\lambda,\psi=0)\in\Lambda c,l}\left\{\lambda\varepsilon + \int(\phi_\lambda(x)+\psi(u))d\mu^*\right\} \tag{33}$$

$$= \inf_{\lambda\in\mathbb{R}_+}\left\{\lambda\varepsilon + \int \sup_{u\in S}\{(u)+\lambda c(x,u)\}d\mu^*\right\}$$

because the infimum in (32) is taken over a bigger set than the infimum in (33).

Secondly,

$$\begin{aligned}\inf_{(\lambda,\phi,\psi)\in\Lambda_{c,l}} &\left\{\lambda\varepsilon + \sup_{\pi\in\Phi_{\mu^*_{XY},K}} \int (\phi(x)+\psi(u))d\pi(x,u)\right\} \\ &= \inf_{(\lambda,\phi,\psi=0)\in\Lambda_{c,l}}\left\{\lambda\varepsilon + \sup_{\pi\in\Phi_{\mu^*_{XY},K}} \int (\phi(x)+\psi(u))d\pi(x,u)\right\} \\ &\geq \inf_{\lambda\in\mathbb{R}_+}\left\{\lambda\varepsilon + \sup_{\pi\in\Phi_{\mu^*_{XY},K}} \int (l(u)-\lambda c(x,u))d\pi(x,u)\right\} \\ &= \inf_{\lambda\in\mathbb{R}_+}\left\{\lambda\varepsilon + \int_S \sup_{\nu\in K}\left\{\int_S (l(u)-\lambda c(x,u))d\nu(u)\right\}d\mu^*_{XY}(x)\right\} \\ &\geq \inf_{\lambda\in\mathbb{R}_+}\left\{\lambda\varepsilon + \int_S \sup_{\hat{u}\in S}\left\{l(\hat{u})-\lambda c(x,\hat{u})\right\}d\mu^*_{XY}(x)\right\} \\ &= \inf_{\lambda\in\mathbb{R}_+}\left\{\lambda\varepsilon + \int \phi_\lambda(x)d\mu^*_{XY}\right\}\end{aligned}$$

where the second inequality uses that whenever $(\lambda,\phi,\psi)\in\Lambda_{c,l}$, $\int\phi+\psi d\pi \geq \int(l-\lambda c)d\pi$ for all $\pi\in\Phi_{\mu^*_{XY},K}$, the third equality applies both marginals of π since $\pi\in\Phi_{\mu^*_{XY},K}$, the fourth equality applies the δ-measure over u as all δ-measures on S are included in $K = P(S)$. This completes the proof. □

This Theorem verifies that when $K = P(S)$, our framework is equivalent to an unconstrained Wasserstein-DRO problem.

In the linear regressional setting, with the assumptions from Theorem 4.1 this would give us the same expression of an l_p-norm regularized regression problem.

7 Inverse problem with Guassian noise in measurement space

In this chapter we study a particular case, namely an inverse problem where we assume y is given by an operation on x and we desire robustness to additive Gaussian noise in the measurement space. For this problem, we will verify the assumptions of the strong duality theorem and reduce the problem to a finite-dimensional one to make it computationally tractable. We validate our framework with numerical simulations of this case in the next section.

We let $\mu^*_{Y|X} := \delta_{Hx}$ and $\mu^*_X := \frac{1}{N}\sum_{i=1}^N \delta_{x_i}$ and assume additive Gaussian noise in the measurement space so

$$\mu^* = \frac{1}{N}\sum_{i=1}^N \delta_{(x_i,Hx_i)} \text{ and } K = \left\{\mu^*_X \otimes \mu_{Y|X} \in P(S) : \mu_{Y|X} = N(Hx,\sigma),\ \sigma \in [0,m]\right\} \cup \{\delta_{Hx}\}$$

where $N(Hx,\sigma)$ represents the normal distribution with mean Hx and variance σ and $m \in \bar{\mathbb{R}}_+$. Finally, we choose as cost inside the Wasserstein-distance $c((x,y),(u,v)) = \|(x,y)-(u,v)\|^2$ any squared norm and as loss function we choose $l(x,y;g) = \|x-g(y)\|_2^2$.

Notice that the assumptions for the strong duality theorem are satisfied: by choosing a closed interval for σ and including the delta-measure of Hx in our definition of K, we have made sure that K is weakly* closed.

7.1 Dual representation

The assumptions for strong duality are verified and thus we can simply apply Theorem 6.4 on our primal problem to find a dual representation.

$$\begin{aligned}
I &= \sup_{\mu \in B_{K,\varepsilon}(\mu^*_{XY})} \int_S l(x,y;g)d\mu(x,y) \\
&= \inf_{(\lambda,\phi,\psi)\in\Lambda_{c,l}} \lambda\varepsilon + \sup_{\pi\in\Phi_{\mu^*_{XY},K}} \int_{S\times S} (\phi(x,y)+\psi(u,v))d\pi((x,y),(u,v)) \\
&= \inf_{(\lambda,\phi,\psi)\in\Lambda_{c,l}} \lambda\varepsilon + \sup_{\pi\in\Phi_{\mu^*_{XY},K}} \frac{1}{N}\sum_{i=1}^N \int_{X\times Y} (\phi(x,y)+\psi(\hat{x}_i,H\hat{x}_i))dN(H\hat{x}_i,\sigma)(x,y) \\
&= \inf_{(\lambda,\phi,\psi)\in\Lambda_{c,l}} \lambda\varepsilon + \sup_{\sigma\in[0,m]} \frac{1}{N\sqrt{2\pi\sigma^2}^n} \sum_{i=1}^N \int_Y \mathrm{e}^{-\frac{1}{2\sigma^2}\|H\hat{x}_i-y\|^2_{\mathbb{R}^2}} (\phi(\hat{x}_i,y)+\psi(\hat{x}_i,H\hat{x}_i))dy \qquad (34)
\end{aligned}$$

where the third equality follows from applying the first marginal μ^* over (u,v) and applying μ^*_X over x. We are left with the Gaussian measure which is applied in the fourth equality, leaving us with a Lebesgue-integral over y. Here, n is the dimension of the data. Note that this is still an infinite-dimensional problem as we optimize over $\phi,\psi \in m_{\mathcal{U}}(S;\mathbb{R})$. Ideally, we would like to find an expression for ϕ and ψ, likely in terms of λ and the cost and loss functions, similar to Blanchet et al. (see Section 4.2.4). The next section will implement an alternative approach to make the problem finite-dimensional.

7.2 Finite-dimensional reduction

In order to make our problem computationally tractable, we will continue with (25) (i.e. only the inner supremum of the Wasserstein-DRO) and reduce this problem to a finite-dimensional convex problem for the case considered in this chapter. Inspired by the convex reduction of the Wasserstein-problem by Esfahani and Kuhn [48] summarized in Section 4.6, we show a similar convex reduction for our problem. We start with a short lemma on the convexity of the ambiguity set(s) to prepare for the Theorem showing the convex reduction of (25) with K and μ^* chosen as in this chapter. We finish with a simple algorithm to solve the final convex optimization problem.

Lemma 7.1. *Suppose K is convex, then $\Phi_{\mu^*_{XY},K}$ and $\Phi_{\mu^*_{XY},K,\varepsilon}$ are both convex.*

Proof. Let $\pi_1, \pi_2 \in \Phi_{\mu^*_{XY},K}$ and define $\pi_3 := \lambda\pi_1 + (1-\lambda)\pi_2 \in P(S \times S)$ where $\lambda \in [0,1]$. Let $\phi \in C(S \times S)$ be any test function.

First, since both π_1 and π_2 have μ^*_{XY} as a marginal we have $\int \phi d\pi_3 = \lambda \int \phi d\mu^*_{XY} + (1-\lambda)\int \phi d\mu^*_{XY} = \int \phi d\mu^*_{XY}$ so π_3 has μ^*_{XY} as marginal as well.

Second, we know $\int \phi d\pi_1 = \int \phi d\nu_1$ and $\int \phi d\pi_2 = \int \phi d\nu_2$ for some $\nu_1, \nu_2 \in K$. Thus, $\int \phi d\pi_3 = \int \phi d(\lambda\nu_1 + (1-\lambda)\nu_2) = \int \phi d\nu_3$ where $\nu_3 \in K$ by convexity of K. Thus, $\pi_3 \in \bigcup_{\nu \in K} \Pi(\mu^*_{XY}, \nu)$ which means $\pi_3 \in \Phi_{\mu^*_{XY},K}$. This concludes the convexity of $\Phi_{\mu^*_{XY},K}$.

Now suppose that we restrict π_1 and π_2 to be inside the Wasserstein ball, in other words $\pi_1, \pi_2 \in \Phi_{\mu^*_{XY},K,\varepsilon}$. Then $\int c d\pi_3 = \lambda \int c d\pi_1 + (1-\lambda)\int c d\pi_2 \leq \lambda\varepsilon + (1-\lambda)\varepsilon = \varepsilon$ so π_3 is also inside the Wasserstein ball and thus $\pi_3 \in \Phi_{\mu^*_{XY},K,\varepsilon}$. This concludes the convexity of $\Phi_{\mu^*_{XY},K,\varepsilon}$ and the proof is complete. □

Theorem 7.2 (Finite-dimensional convex reduction for an Inverse Problem with additive Gaussian Noise)**.** *We let $\mu^*_{Y|X} := \delta_{Hx}$ and $\mu^*_X := \frac{1}{N}\sum_{i=1}^N \delta_{x_i}$ and assume additive Gaussian noise in the measurement space so*

$$\mu^*_{XY} = \frac{1}{N}\sum_{i=1}^N \delta_{(x_i, Hx_i)} \text{ and } K = \left\{\mu^*_X \otimes \mu_{Y|X} \in P(S) : \mu_{Y|X} = N(Hx, \sigma),\ \sigma \in [0, m]\right\} \cup \{\delta_{Hx}\}$$

where $N(Hx, \sigma)$ represents the normal distribution with mean Hx and variance σ and $m \in \bar{\mathbb{R}}_+$. For the cost inside the Wasserstein distance, choose $c((x,y),(u,v)) := \|(x,y)-(u,v)\|^2$ any squared norm and let the loss $l(x,y;g)$ be upper semi-continuous, proper and concave such that it is not $-\infty$ everywhere on $S \times S$. Then our primal problem is equal to

$$\begin{aligned} I &= \sup_{\pi \in \Phi_{\mu^*_{XY},K,\epsilon}} \int_{S\times S} l(u,v;g) d\pi((x,y),(u,v)) \\ &= \inf_{\lambda \in \mathbb{R}_+} \sup_{\sigma \in [0,m]} \lambda\epsilon + \frac{1}{N\sqrt{2\pi\sigma^2}^n} \sum_{i=1}^N \int_Y e^{-\frac{1}{2\sigma^2}\|H\hat{x}_i - y\|^2_{\mathbb{R}^n}} \left(l(\hat{x}_i, y; g) - \lambda\|H\hat{x}_i - y\|^2\right) dy. \end{aligned} \tag{35}$$

Proof. First, we note that K is convex. To see this let $\mu_3 = \lambda\mu_1 + (1-\lambda)\mu_2$ where $\lambda \in [0,1]$ and $\mu_1, \mu_2 \in K$ so $\mu_1 = \mu^*_X \otimes N(Hx, \sigma_1)$ and $\mu_2 = \mu^*_X \otimes N(Hx, \sigma_2)$ for some $\sigma_1, \sigma_2 \in \bar{\mathbb{R}}_+$. Then

$$\begin{aligned} \mu_3 &= \mu^*_X \otimes (\lambda N(Hx, \sigma_1) + (1-\lambda)N(Hx, \sigma_2)) = \mu^*_X \otimes (N(\lambda Hx, \lambda^2\sigma_1) + N((1-\lambda)Hx, (1-\lambda)^2\sigma_2)) \\ &= \mu^*_X \otimes N(Hx, \lambda^2\sigma_1 + (1-\lambda)^2\sigma_2) = \mu^*_X \otimes N(Hx, \sigma_3) \end{aligned}$$

where $\sigma_3 \in \bar{\mathbb{R}}_+$ since $\sigma_1, \sigma_2 \in \mathbb{R}_+$ and $\lambda, (1-\lambda) \geq 0$ so $\mu_3 \in K$. By Lemma 7.1, the ambiguity set $\Phi_{\mu^*_{XY},K,\epsilon}$ is convex as well.

Recall $B_{K,\epsilon}(\mu^*_{XY}) := \{\mu \in K : W_1(\mu, \mu^*_{XY}) \leq \epsilon\}$ and that our problem can equivalently be written as

$$\begin{aligned} I &= \sup_{\pi \in \Phi_{\mu^*_{XY},K,\epsilon}} \int_{S\times S} l(u,v;g) d\pi((x,y),(u,v)) \\ &= \sup_{\mu \in B_{\epsilon,K}(\mu^*_{XY})} \int_S l(x,y;g) d\mu(x,y). \end{aligned}$$

Using the definition of the Wasserstein-distance, we rewrite

$$\sup_{\mu\in B_{K,\epsilon}(\mu^*_{XY})}\int_S l(x)d\mu = \begin{cases}\sup_{\pi,\mu} & \int_S l(x,y;g)d\mu(x,y)\\ \text{s.t.} & \int_{S\times S}\|(x,y)-(u,v)\|^2 d\pi((x,y),(u,v))\le\epsilon\\ & \pi\in\Pi(\mu,\mu^*_{XY})\\ & \mu\in K\end{cases}$$

$$= \begin{cases}\sup_{\mu_i\in K} & \frac{1}{N}\sum_{i=1}^N\int_S l(x,y;g)d\mu_i(x,y)\\ \text{s.t.} & \frac{1}{N}\sum_{i=1}^N\int_S\|(x,y)-(\hat{x}_i,H\hat{x}_i)\|^2 d\mu_i(x,y)\le\epsilon.\end{cases} \tag{36}$$

Above, the second equality follows from Bayes law, which states that any joint distribution π of (x,y) and (u,v) can be constructed from the marginal distribution μ^*_{XY} of (u,v) and the conditional distribution μ_i of (x,y) given $(u,v)=(\hat{x}_i,H\hat{x}_i)$, $i\le N$. In other words, $\pi=\frac{1}{N}\sum_{i=1}^N\delta_{(\hat{x}_i,H\hat{x}_i)}(u,v)\otimes\mu_i(x,y)=$ with $\mu_i\in K$. By duality we can rewrite (36):

$$\sup_{\mu_i\in K}\inf_{\lambda\in\mathbb{R}_+}\frac{1}{N}\sum_{i=1}^N\int_S l(x,y;g)d\mu_i(x,y)+\lambda\left(\epsilon-\frac{1}{N}\sum_{i=1}^N\int_S\|(x,y)-(\hat{x}_i,H\hat{x}_i)\|^2 d\mu_i(x,y)\right)$$

$$\le\inf_{\lambda\in\mathbb{R}_+}\sup_{\mu_i\in K}\lambda\epsilon+\frac{1}{N}\sum_{i=1}^N\int_S\left(l(x,y;g)-\lambda\|(x,y)-(\hat{x}_i,H\hat{x}_i)\|^2\right)d\mu_i(x,y) \tag{37}$$

$$=\inf_{\lambda\in\mathbb{R}_+}\sup_{\sigma\in[0,m]}\lambda\epsilon+\frac{1}{N}\sum_{i=1}^N\int_Y\left(l(\hat{x}_i,y;g)-\lambda\|(\hat{x}_i,y)-(\hat{x}_i,H\hat{x}_i)\|^2\right)dN(H\hat{x}_i,\sigma)(y) \tag{38}$$

$$=\inf_{\lambda\in\mathbb{R}_+}\sup_{\sigma\in[0,m]}\lambda\epsilon+\frac{1}{N}\sum_{i=1}^N\int_Y\left(l(\hat{x}_i,y;g)-\lambda\|H\hat{x}_i-y\|^2\right)dN(H\hat{x}_i,\sigma)(y) \tag{39}$$

$$=\inf_{\lambda\in\mathbb{R}_+}\sup_{\sigma\in[0,m]}\lambda\epsilon+\frac{1}{N\sqrt{2\pi\sigma^2}^n}\sum_{i=1}^N\int_Y \mathrm{e}^{-\frac{1}{2\sigma^2}\|H\hat{x}_i-y\|^2_{\mathbb{R}^n}}\left(l(\hat{x}_i,y;g)-\lambda\|H\hat{x}_i-y\|^2\right)dy \tag{40}$$

where n is the dimension of the data (e.g. $y\in\mathbb{R}^n$). The first (in-)equality follows from the max-min inequality. From (37) to (38) we apply the δ measure over the x-variable and we are left with the normal distribution over y. From (39) to (40) we apply the normal measure to transform the integral to an integral over y. We have a linear problem with convex constraints and since l is upper semi-continuous (thus $-l$ is lower semi-continuous), we can apply a strong duality result [50, Proposition 3.4]. Thus, the inequality in (37) actually is an equality and we are finished with a finite-dimensional optimization problem. □

We see that the finite-dimensional reduction in (35) is very similar to the dual representation of the problem in (34). In fact, if we find ϕ,ψ such that $\phi(\hat{x}_i,y)+\psi(\hat{x}_i,H\hat{x}_i)=l(\hat{x}_i,y;g)-\lambda c((\hat{x}_i,y),(\hat{x}_i,H\hat{x}_i))=l(x,y;g)-\lambda\|H\hat{x}_i-y\|^2$ we have the exact same expression. For this particular case, we can obtain (35) by choosing $\phi(\hat{x}_i,y)=l(\hat{x}_i,y;g)-\lambda\|H\hat{x}_i-y\|^2$ and $\psi=0$ in (34).

We now have a finite-dimensional, convex problem in (35) which means we can compute the optimal value of our problem using any convex optimization scheme. The focus of this thesis lies on the theory so for the numerical examples we adopt a simple optimization scheme. We approximate the optimal value of (35) by alternating a few steps of gradient descent for λ and a few steps of gradient ascent for σ. We define

$$G(\lambda,\sigma;g):=\lambda\epsilon+\frac{1}{N\sqrt{2\pi\sigma^2}^n}\sum_{i=1}^N\int_Y \mathrm{e}^{-\frac{1}{2\sigma^2}\|H\hat{x}_i-y\|^2_{\mathbb{R}^n}}\left(l(\hat{x}_i,y;g)-\lambda\|H\hat{x}_i-y\|^2\right)dy$$

so that (35) is equivalent to

$$\inf_{\lambda\in\mathbb{R}_+}\sup_{\sigma\in[0,m]}G(\lambda,\sigma;g).$$

The partial derivatives of $G(\lambda, \sigma; g)$ with respect to λ and σ are respectively

$$\frac{dG}{d\lambda}(\sigma) = \epsilon - \frac{1}{N\sqrt{2\pi\sigma^2}^n} \sum_{i=1}^{N} \int_Y \mathrm{e}^{-\frac{1}{2\sigma^2}\|H\hat{x}_i - y\|_{\mathbb{R}^n}^2} \|H\hat{x}_i - y\|^2 dy$$

and

$$\frac{dG}{d\sigma}(\lambda, \sigma; g) = \frac{1}{N\sqrt{2\pi\sigma^2}^n} \sum_{i=1}^{N} \int_Y \mathrm{e}^{-\frac{1}{2\sigma^2}\|H\hat{x}_i - y\|_{\mathbb{R}^n}^2} \left(l(\hat{x}_i, y; g) - \lambda\|H\hat{x}_i - y\|^2\right) \left(\frac{1}{\sigma^3}\|H\hat{x}_i - y\|_{\mathbb{R}^n}^2 - \frac{2}{n\sigma}\right) dy.$$

Letting γ_1, γ_2 be two learning rates, m_1 the number of iterations, m_2 the number of sub-iterations and λ_0, σ_0 our initial values our algorithm will look as in Algorithm 1.

Algorithm 1 Alternating Gradient Descent/Ascent to Find Optimizers for Discrete Worst-Case Problem

Require: $\lambda_0, \sigma_0 \geq 0$, $\gamma_1, \gamma_2, m_1, m_2 \in \mathbb{N}, g \in \Sigma$
Ensure: λ, σ
  $\lambda \leftarrow \lambda_0$
  $\sigma \leftarrow \sigma_0$
  for $k \leftarrow 1$ to m_1 **do**
    for $i \leftarrow 1$ to m_2 **do**
      $\lambda \leftarrow \max\{\lambda - \gamma_1 \times \frac{dG}{d\lambda}(\sigma; g), 0\}$
    end for
    for $j \leftarrow 1$ to m_2 **do**
      $\sigma \leftarrow \max\{\sigma + \gamma_2 \times \frac{dG}{d\sigma}(\lambda, \sigma; g), 0\}$
    end for
    if λ and σ converge **OR** objective G converges **then**
      break
    end if
  end for

This algorithm gives us the values λ^*, σ^* with which we approximate the optimal value of $G(\lambda, \sigma; g)$ is optimal i.e. $G(\lambda^*, \sigma^*; g) \approx \inf_{\lambda \in \mathbb{R}_+} \sup_{\sigma \in \mathbb{R}_+} G(\lambda, \sigma; g)$. Finally, recall that our full problem is to find the value of g that minimizes the maximum value of $G(\lambda, \sigma; g)$. Our original full full problem is

$$\inf_{g \in \Sigma} \sup_{\mu \in B_{\epsilon,K}(\mu^*_{XY})} \int_{S \times S} l(x, y; g) d\mu$$

which we approximate with

$$\inf_{g \in \Sigma} G(\lambda^*, \sigma^*; g).$$

We solve the latter by employing Algorithm 2 which finds the worst-case value for each g in a pre-specified set Σ with Algorithm 1 and then finds the g with the smallest value for $G(\lambda, \sigma; g)$. This returns the g yielding the best worst-case.

Algorithm 2 Algorithm to Solve Discrete Best Worst-Case Problem

Require: Σ ▷ Set of candidates
Ensure: g^*, G^* ▷ Optimal candidate and its worst-case value
  Initialize worst-cases as an empty list
  for $i, g \in$ enumerate(Σ) **do**
    $\lambda[i], \sigma[i] \leftarrow$ Algorithm 1(g)
    worst-cases[i] $\leftarrow G(\lambda[i], \sigma[i]; g)$
  end for
  $i^* \leftarrow \arg\min$(worst-cases)
  $g^* \leftarrow \Sigma[i^*]$
  $G^* \leftarrow$ worst-cases$[i^*]$
  return g^*, G^*

In the next section, we will verify that the optimization runs smoothly, giving motivation to the idea that a decoupled optimization like this gives similar results to a coupled one. Decoupled meaning we first optimize over λ, σ and then over g. To give some initial examples and simulations, we will choose a simple set for Σ dependent on the simulated problem.

8 Numerical examples

This section shows some simple examples using Algorithms 1 and 2 to illustrate the robustness and performance of the Wasserstein-DRO framework for inverse problems. We continue with the inverse problem with additive Gaussian uncertainties in the measurement space, so we adopt again the assumptions in Theorem 7.2. We take $H \in \mathbb{R}^{n\times n}$, $\Sigma \subseteq \mathbb{R}^{n\times n}$ and $X, Y = \mathbb{R}^n$ to model the inverse problem $y^\delta = Hx^* + \delta$ where we assume the noise δ is normally distributed. In order to make some simple calculations, we will manually choose Σ as a small set containing the (approximate) inverse (left-inverse for non-singular cases, pseudo-inverses for singular cases) and a few random candidates. We will refer to these 'candidate inverses' as 'map index i' with $i = 0, 1, 2, 3$ referring to the index they have in the set Σ. For every example of a non-singular matrix, the last map in the set (map 3) is the real inverse. Ideally, other methods (i.e. deep learning methods) would be used to include a larger set Σ but as this thesis has a theoretical focus, we leave that up to future research. We randomly generate the data $\hat{x}_i$, $i = 1, 2, ..., 12$ with values between 0 and 10. We take twelve data-points because that is the number of simulatenous processes we can implement. For the remainder of this section, denote the learning rates for σ and λ by γ_σ and γ_λ, respectively and their starting values by σ_0 and λ_0 respectively. Unless stated otherwise, we take as number of sub-iterations 3 and number of total iterations 300. When the difference between the current risk value and the previous risk value is less than 0.0001 for two consecutive iterations we conclude the objective value has converged and we terminate the iteration process. The same holds for the parameters: if for **both** parameters holds that the difference between its current value and its previous value is less than 0.0001 for two consecutive iterations, we terminate the iteration process.

For each iteration, we calculate the derivatives of λ and σ three times each and the objective value once. As each of these contains a multi-dimensional integral that is calculated for each data-point, we calculate $7 \cdot 12$ multi-dimensional integrals per iteration. We usually have four candidate inverses in our set Σ, which means we calculate 100.800 of these integrals per map. To improve efficiency, we use multi-processing to calculate the integral for each data-point in parallel and we set the bounds of the integral at the smallest value where the integral seems to converge. During test simulations, we settled on bounds of ± 30. For future research, a more efficient numerical approach should be considered.

For simplicity, we refer to the value the objective function $G(\lambda, \sigma; g)$ takes for a certain triple $(\lambda, \sigma; g)$ as the 'risk'. For each example, we give plots illustrating the decoupled optimization processes:

- **The worst-case optimization process**: for each candidate inverse we create a separate figure which plots for every iteration the current value of σ on the left (blue) and λ on the right (red) y-axis, against the risk for that pair on the x-axis. Ideally, these plots would all show the convergence to a supremum of the risk.
- **The best-worst case optimization process**: in one figure, we plot the final 'worst-case' risk value on the y-axis for each candidate inverse on the x-axis. This is for each map the risk value their 'worst-case' optimization algorithm finished on.

As the examples focus on a low-dimensional ($\mathbb{R}^{2\times 2}$) forward operator and data for computational efficiency, we first illustrate the Picard condition for a higher-dimensional operator before exploring these examples.

The code for the Wasserstein-DRO examples can be found in Appendix A.1 and the code for the Picard condition can be found in Appendix A.2.

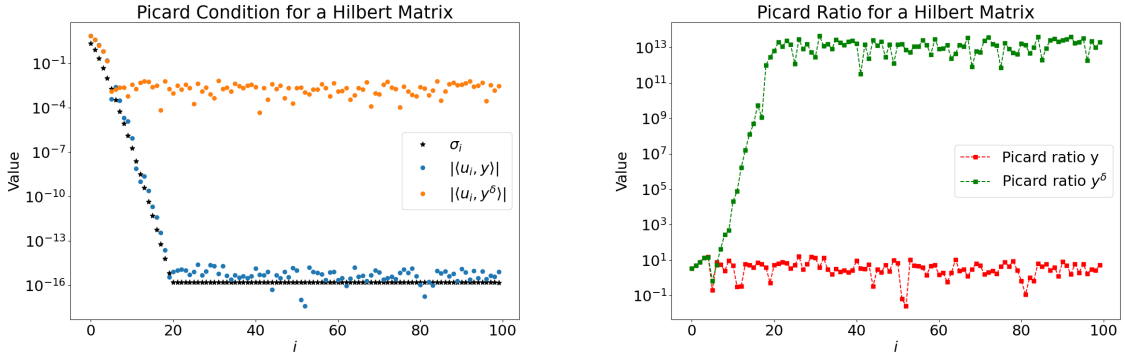
8.1 Picard condition for a high-dimensional operator

To illustrate how the Picard condition is applied to a high-dimensional operator, we take as forward operator H a 100×100 Hilbert matrix, a square matrix whose elements are given by $H_{ij} = \frac{1}{i+j-1}$. For example, this is the 3×3 Hilbert matrix:

$$\begin{bmatrix} 1 & \frac{1}{2} & \frac{1}{3} \\ \frac{1}{2} & \frac{1}{3} & \frac{1}{4} \\ \frac{1}{3} & \frac{1}{4} & \frac{1}{5} \end{bmatrix}$$

which has condition number approximately 524. The Hilbert matrix is a well-known ill-conditioned matrix, as its condition number grows rapidly as its size increases. The 100×100 Hilbert matrix has condition number 1.075×10^{19}, meaning that inverting a Hilbert matrix will be numerically unstable. Recall that the Moore-Penrose pseudo-inverse applied to a measurement $y \in \mathbb{R}^n$ is given by $H^\dagger y = \sum_{i=1}^{n} \frac{\langle u_i, y \rangle_Y}{\sigma_i} v_i$ with σ_i the i-th singular value and u_i, v_i the i-th left and right singular vectors. Thus a vector $y \in \mathbb{R}^n$ satisfies the Picard condition if the projections $|\langle u_i, y \rangle|$ decay faster than the singular values σ_i. Recall that for a non-singular matrix, the real inverse is equal to the Moore-Penrose inverse, which is the case here.

We build the 100×100 Hilbert matrix, generate random data x and noiseless and noisy measurements $y = Hx$ and $y^\delta = y + \delta$ respectively, where δ is normally distributed with mean zero and variance $\sigma^2 = 0.01^2$. We calculate the singular value decomposition and the projections of y and y^δ with the left singular vectors. Figure 2a depicts the decay of the singular values against the decay of the projections of noisy and noiseless data with the left singular vectors while Figure 2b depicts the Picard ratio for noisy and noiseless data, both are semi-log plots.



(a) Decay of σ_i against $|\langle u_i, y \rangle|$ and $\left|\langle u_i, y^\delta \rangle\right|$

(b) Picard ratio of y and y^δ

Figure 2: Picard Condition for the 100×100 Hilbert Matrix

The noiseless data decays in a similar exponential fashion to the singular values, but the noisy data does not decay as the singular values do and thus does not satisfy the discrete Picard condition and is ill-posed. This is further illustrated by the Picard ratio growing exponentially for the noisy data but not for the noiseless data. We conclude that the matrix is near-singular and very sensitive to noise in the data and the inverse might be difficult to work with numerically when we have noisy data. Note that for any singular matrix, we have at least one singular value equal to 0, leading the Picard ratio to grow to infinity. For the types of matrices mentioned in this section (unstable and/or singular matrices), it would thus be very useful to obtain a robust inverse.

8.2 Inverse problem with measurement in $\mathbb{R}^n$ and Gaussian noise

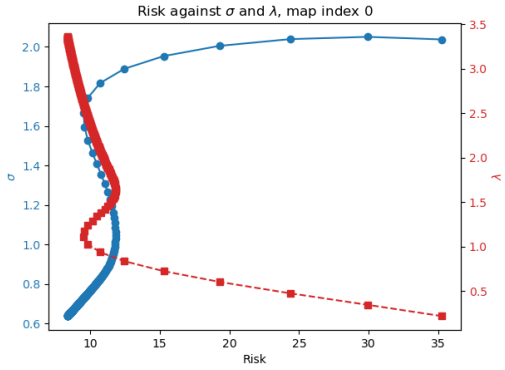
We take input and output spaces $X, Y = \mathbb{R}^n$, specifically $n = 2$. For this problem we model a non-singular **stable** and a non-singular **unstable** forward operator to see if the Wasserstein-DRO gives the real inverse. Next to that, we model two singular matrices to investigate whether the Wasserstein-DRO gives a robust inverse, which we can compare to other inverses.

8.2.1 Non-singular, stable forward operator

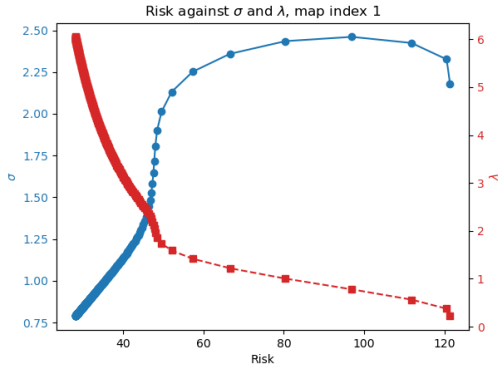
We choose the forward operator H as $H = \begin{bmatrix} 2 & 0 \\ 0 & 2 \end{bmatrix}$ which has real left inverse $H^{-1} = \begin{bmatrix} 1/2 & 0 \\ 0 & 1/2 \end{bmatrix}$. We simulate the Wasserstein-DRO against two versions of Σ, each contains the real left inverse but Σ_1 contains candidate inverses 'close' to the left inverse and Σ_2 contains candidates less 'close' to the left inverse. For Σ_1 we randomly add values between [-0.5, 0.5] to the diagonal of the left inverse while for Σ_2 we randomly add values between [-5, 5]. Each set contains four candidates, including the left inverse which has map index 3. For Σ_1 we use $\gamma_\lambda = 0.001, \gamma_\sigma = 0.001, \lambda_0 = 0.1, \sigma_0 = 2$ and for Σ_2

we use $\gamma_\lambda = 0.0001, \gamma_\sigma = 0.00001, \lambda_0 = 0.1, \sigma_0 = 2$. Notice that we have chosen smaller learning rates for Σ_2 as the gradients for maps further away from the real inverse are larger and could cause steps that were too big. This is undesirable since we don't want to skip over a maximum and don't want negative values for both parameters.

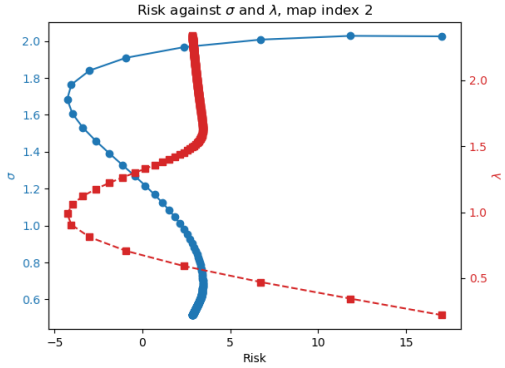
Figure 3 shows for each map in Σ_1 the worst-case optimization process. The process for all maps starts at $\lambda_0 = 0.1$, $\sigma_0 = 2$ and for each iteration gives the current value of λ, σ on right and left y-axis respectively, with the corresponding risk on the x-axis. Map 3 is the only one that has converged and is also the one with the smallest worst-case risk value, as can be seen in Figure 4. This Figure shows the worst-case risk value for each map index, showing map 3 indeed has the smallest worst-case risk. This means the algorithm has selected the real left-inverse as the most robust one. It is interesting to note that map 2 has an optimization process that looks the most similar to the one from map 3 (parabolic shapes that converge to a line) and is also the one with closest 'worst-case' risk value.



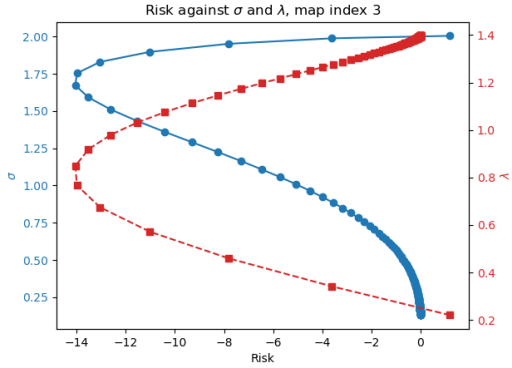
(a) Map index 0



(b) Map index 1



(c) Map index 2



(d) Map index 3

Figure 3: Worst-case optimization for non-singular, stable forward operator. Maps $\in \Sigma_1$.

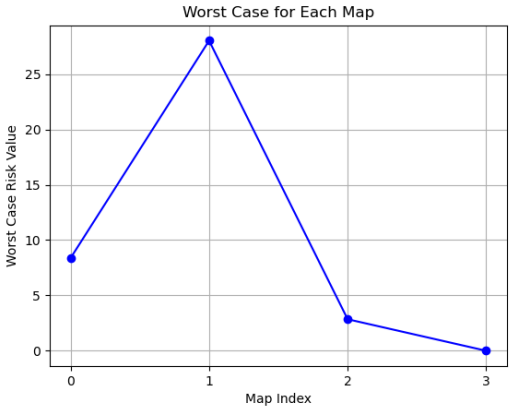
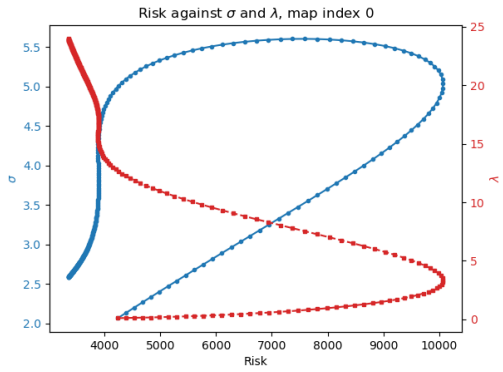
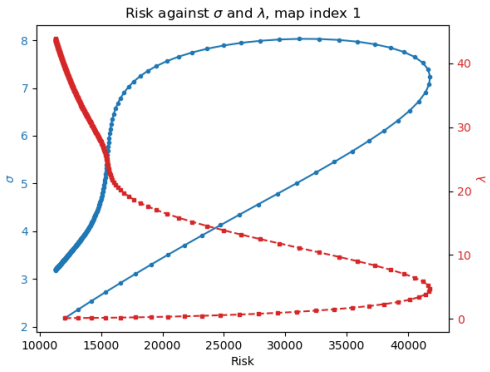


Figure 4: Best worst-case optimization for non-singular, stable forward operator. Maps $\in \Sigma_1$.

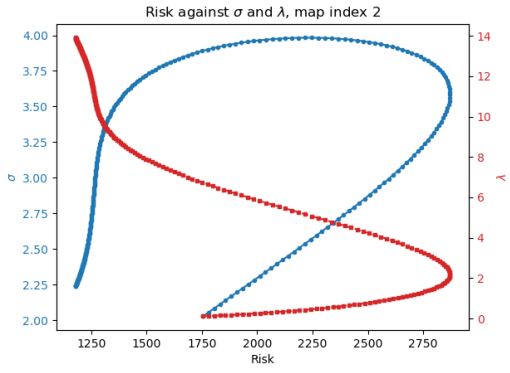
Figure 5 shows the worst-case optimization process for each map in Σ_2. Smaller learning rates result in smaller steps between iterations, causing map 3 to not converge. Larger learning rates or more iterations might improve this. The optimization process for maps in Σ_2 differs significantly from those in Σ_1 and the real inverse, as they are 'further away' from the real inverse. Figure 6 displays the worst-case risk values for each map, which are higher due to the maps being less close to the real inverse. Map index 3 has the smallest worst-case again, indicating the algorithm selected the real left-inverse in both examples for a non-singular, well-posed operator.



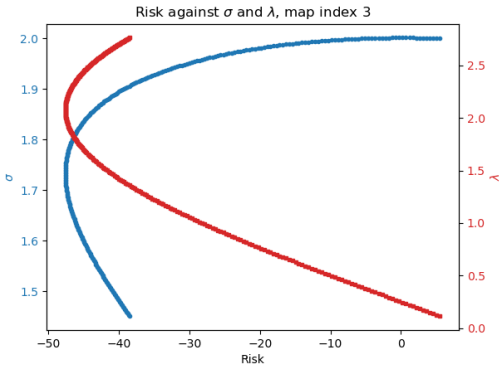
(a) Map index 0



(b) Map index 1



(c) Map index 2



(d) Map index 3

Figure 5: Worst-case optimization for non-singular, stable forward operator. Maps $\in \Sigma_2$.

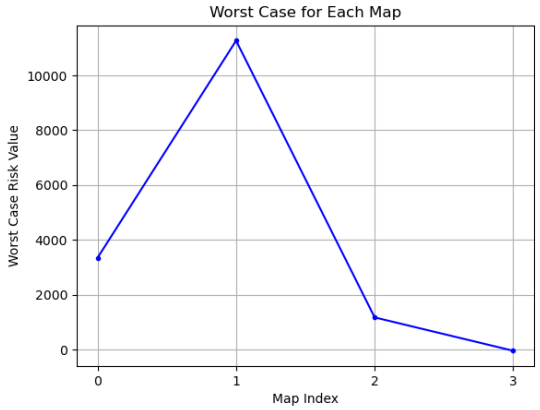
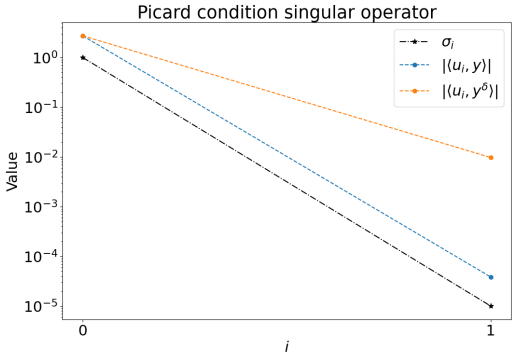


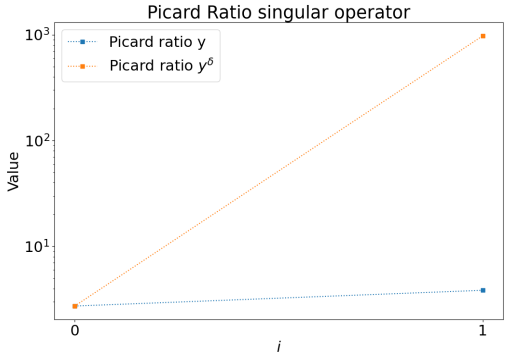
Figure 6: Best worst-case optimization for non-singular, stable forward operator. Maps $\in \Sigma_2$.

8.2.2 Non-singular, unstable forward operator

As an unstable forward operator, we choose $H = \begin{bmatrix} 1 & 0 \\ 0 & 1e-5 \end{bmatrix}$. For diagonal matrices, its condition number is simply the ratio between the largest diagonal element and the smallest on the diagonal. Thus we can calculate the condition number $1/(1e-5) = 100000$ and its left-inverse $H^{-1} = \begin{bmatrix} 1 & 0 \\ 0 & 1e5 \end{bmatrix}$. We illustrate the ill-posedness of this matrix with the Picard condition in Figure 7 in a similar manner as in the beginning of this chapter. In Figure 7a we see that while $|\langle u_i, y\rangle|$ decays similarly to the singular values, this is not the case for $\left|\langle u_i, y^\delta\rangle\right|$ so it does not satisfy the Picard condition. This is confirmed by Figure 7b, illustrating that the Picard ratio of noisy data grows much faster than noiseless data. Thus, we have an ill-posed problem.



(a) Decay of σ_i against $|\langle u_i, y\rangle|$ and $\left|\langle u_i, y^\delta\rangle\right|$

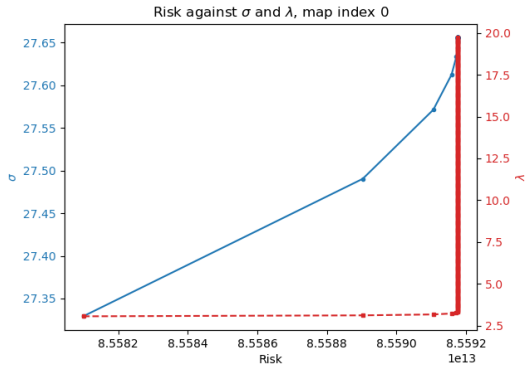


(b) Picard ratio of y and y^δ

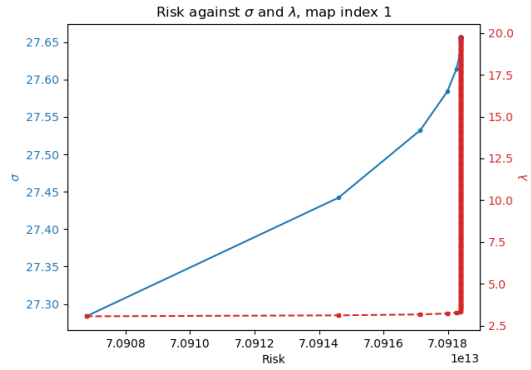
Figure 7: Picard Condition for an ill-posed forward operator

We again simulate against two versions of Σ similar to the previous section. For Σ_1 we randomly offset the diagonal of the real inverse by $\pm 10\%$. For Σ_2 we randomly generate diagonal candidates with first diagonal value between $[0, 2]$ and the second between $[10^3, 10^7]$. Each set again contains four candidates, including the left inverse (with index 3) and we use $\gamma_\lambda = 10^{-6}, \gamma_\sigma = 10^{-12}, \lambda_0 = 3, \sigma_0 = 27$.

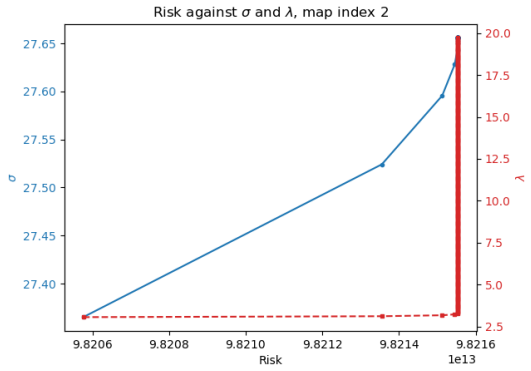
Figure 8 shows the worst-case optimization process for Σ_1 which shows a similar process for each map. The optimization of σ starts on the left, makes some very big steps and then quickly converges while λ also starts on the left but keeps growing steadily, and with it the risk as well. As the scales on these plots are so large it seems like the risk converges, but this is not the case for any of the maps. More iterations (800) give the same results. Figure 9 shows that map 1 has the smallest worst-case risk value which is not the real inverse. Since none of these optimization processes seem to converge it is likely



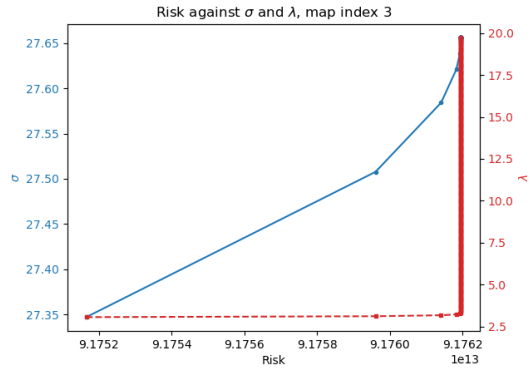
(a) Map index 0



(b) Map index 1



(c) Map index 2



(d) Map index 3

Figure 8: Worst-case optimization for non-singular, stable forward operator. Maps $\in \Sigma_1$.

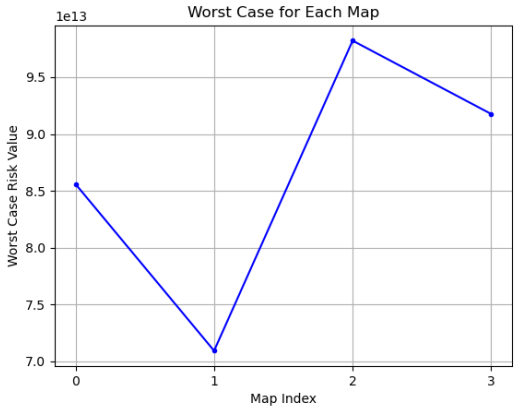
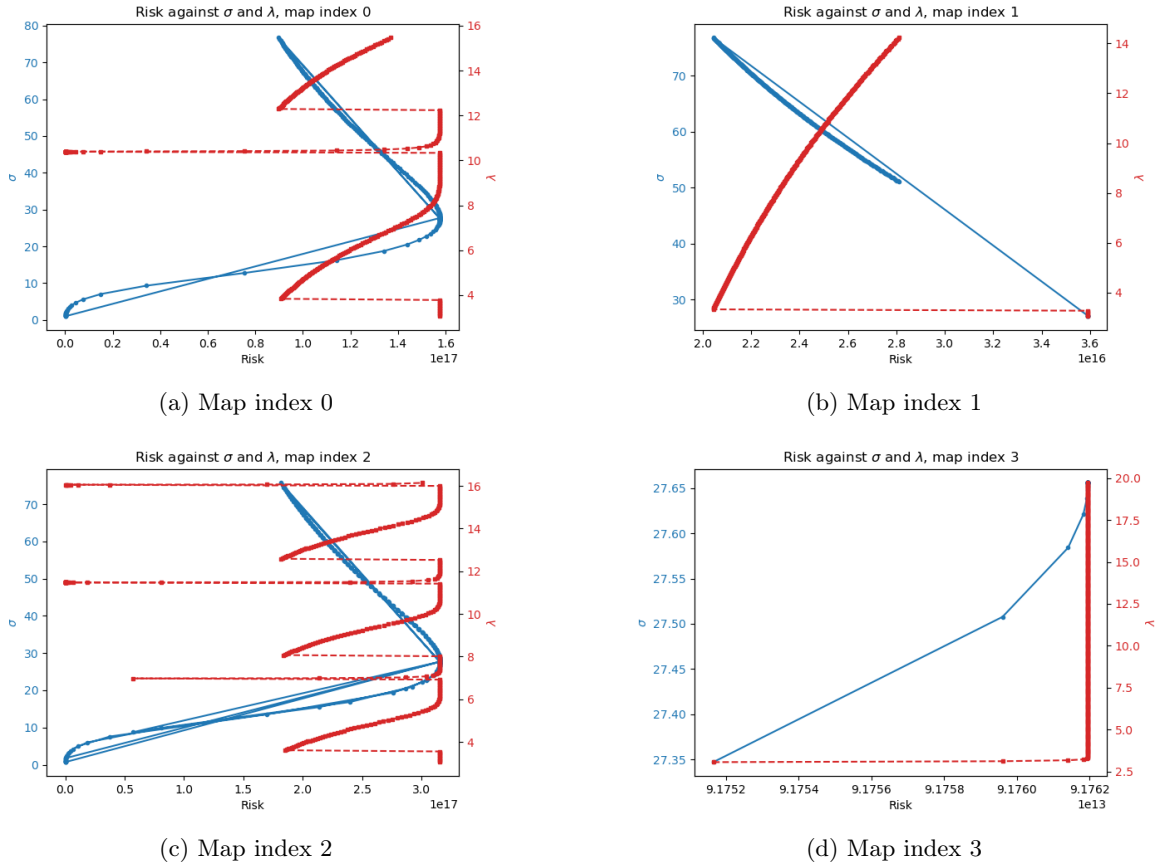


Figure 9: Best worst-case optimization for non-singular, stable forward operator. Maps $\in \Sigma_1$.

that the numerical implementation needs some improvements, for example better (adaptive) learning rates or a restriction on the step size.

Figure 10 depicts the worst-case optimization process for Σ_2. There is some jumping around in the optimization processes of the maps further away from the real inverse and the optimization is not very smooth. For maps 0 and 2, it seems like σ is converging to a maximum but then jumps away from it only to converge toward it again. On the other hand, λ also makes jumps and then keeps growing, similarly to the previous example. These jumps in the optimization process are likely a result



(a) Map index 0

(b) Map index 1

(c) Map index 2

(d) Map index 3

Figure 10: Worst-case optimization for non-singular, unstable forward operator. Maps $\in \Sigma_2$ (less close to real inverse).

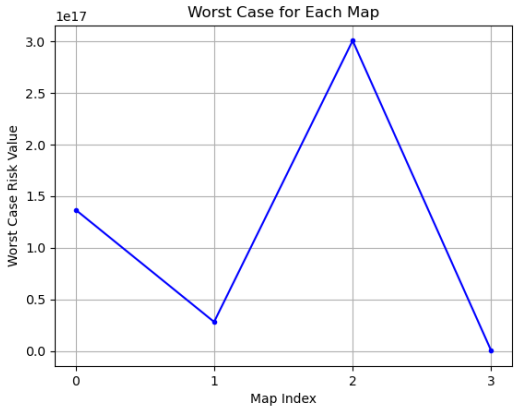


Figure 11: Best worst-case optimization for non-singular, unstable forward operator. Maps $\in \Sigma_2$ (less close to real inverse).

of non-ideal learning rates, which should be improved for better performance. Figure 11 shows that this time the real inverse, map 3, is selected by the algorithm as the robust solution.

A difficulty in the implementation of this case was caused by some iterations bringing σ below zero while the variance is a non-negative parameter. Technically, it should be allowed to be zero (or infinity) as this is needed for our set K to be closed. However, this gives computational problems as

the objective function and both its partial derivatives contain a division by σ. To combat this, we tried projecting σ to a small value close to zero but this gave unsatisfactory results as at some point while σ would converge towards the (local) maximum after the 'jump', it would jump again to a negative value which was projected to 10^{-5}. After this, the optimization got 'stuck' in this point with a very low risk while at this point we are still maximizing the risk (find the worst-case for this candidate inverse). For an example of this, see Figure 12. As an alternative approach we chose to use a smaller learning rate ($\hat{\gamma}_\sigma = \gamma_\sigma \cdot 0.0001$) whenever the original learning rate bring σ below zero. If it was still brought below zero, we left σ unchanged but this was never necessary as the smaller learning rate solved that issue. In the next iteration, a new value of λ could cause σ to change again. Again, this case could likely benefit from better (adaptive) learning rates or a restriction on the step size to diminish the 'jumping' to numbers that are large in magnitude (positive or negative). As the implementation needs improvements, the results for the non-singular unstable operator are not to be taken at face-value.

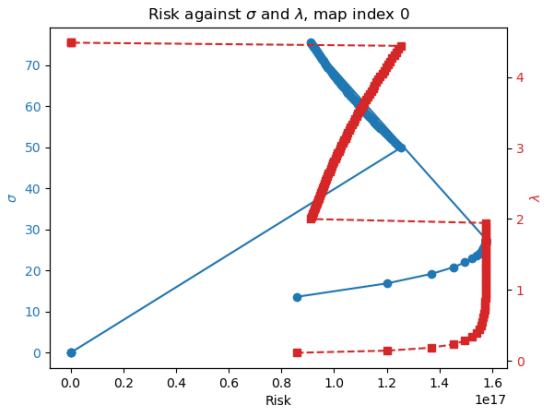
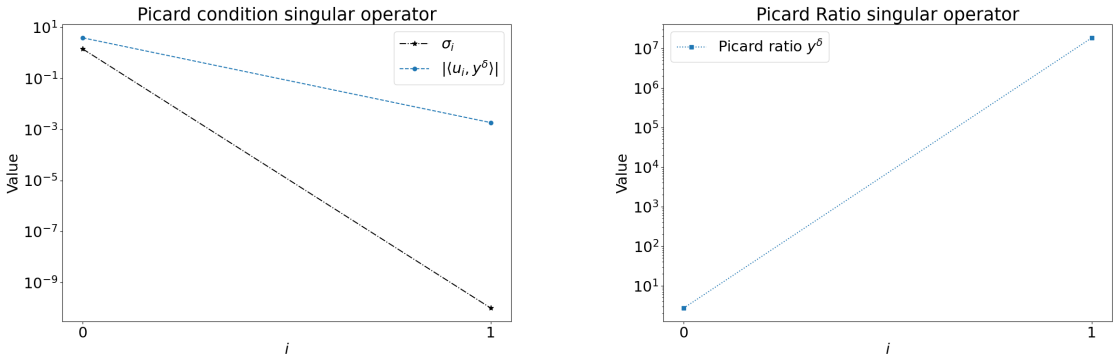


Figure 12: Projection σ whenever it gets below 0. (Worst-case optimization for non-singular, unstable forward operator. Σ less close to real inverse.)

8.2.3 Singular forward operator 1

As a singular operator we choose $H = \begin{bmatrix} 1 & 0 \\ 1 & 0 \end{bmatrix}$ which is neither injective (it has non-trivial null-space) nor surjective (it has linearly dependent rows), meaning a left- or right-inverse does not exist. The left singular vectors are $u_1 \approx [-0.707, -0.707]^T$ and $u_2 \approx [-0.707, 0.707]^T$. If we let $x = [x_1, x_2]^T$, we have $y = Hx = [x_1, x_1]^T$ so $|\langle u_1, y\rangle| = 1.414x_1$ and $|\langle u_2, y\rangle| = 0$ for any pair (x, y). As one of the singular values is equal to 0, and the other is 1.414 we have Picard ratio $x_1 + \frac{0}{0}$ which is undefined and thus makes no sense to plot. For the noisy data however this is not the case, as $y^\delta = [x_1, x_1] + \delta$ with δ normally distributed with mean 0 and variance $\sigma^2 = 0.01^2$. As one of the singular values is equal to 0, we replace this by 10^{-10} in order to plot it on the semi-log plot. Thus the 10^7 in Figure 13b would in reality be infinity and this operator clearly does not satisfy the Picard condition.

The Moore-Penrose pseudo-inverse can be computed using the SVD: $H^{\text{MP}} = \begin{bmatrix} 1/2 & 1/2 \\ 0 & 0 \end{bmatrix}$. Notice that HH^{MP} and $H^{\text{MP}}H$ are Hermitian, $HH^{\text{MP}}H = H$ and $H^{\text{MP}}HH^{\text{MP}} = H^{\text{MP}}$, thus satisfying the properties of a pseudo-inverse. For this matrix, any matrix where the elements of the first row sum to 1 is a generalized inverse, independent of the second row. Since $H^2 = H$, the Drazin inverse of H is equal to H itself: $H^D = H$. This time we simulate against one version of Σ, containing the Moore-Penrose and Drazin inverses and two randomly generated generalized inverses: one that has zeros on the second row (which we denote by G1) and one that has completely random values on the second row (G2). Map 0 is the Moore-Penrose inverse, map 1 is the Drazin inverse, map 2 is G1 and map 3 is G2. We use $\gamma_\lambda = 10^{-4}, \gamma_\sigma = 10^{-4}, \lambda_0 = 6, \sigma_0 = 4$. As this case converged slower, we used 600 iterations.

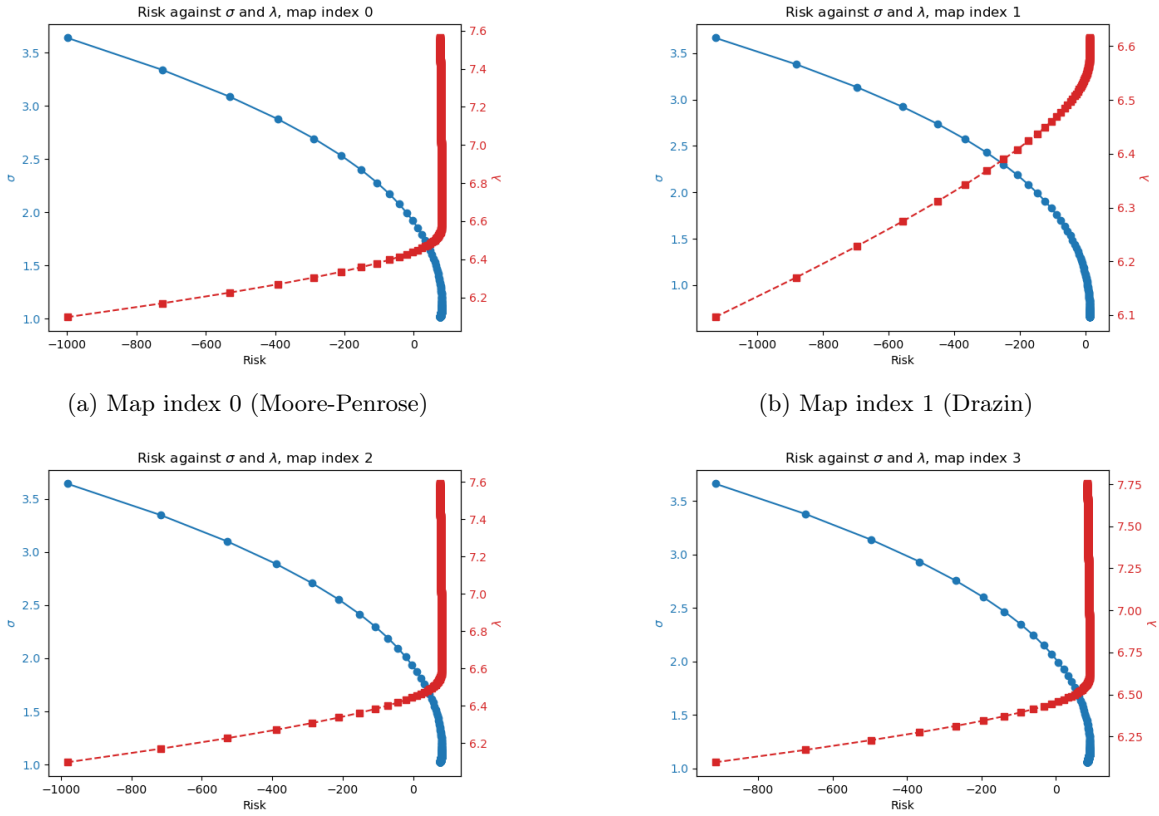


(a) Decay of σ_i against $\left|\left\langle u_i, y^\delta \right\rangle\right|$

(b) Picard ratio y^δ

Figure 13: Picard Condition for a singular forward operator

Figure 14 shows that the worst-case optimization for each map starts with a large negative risk and converges to a positive value. The optimization for the Moore-Penrose inverse converges to 76.81, for the Drazin inverse it converges to 14.87, G1 converges to 14.87 and G2 converges to 84.64. This means the algorithm selects the Drazin as the robust solution and that the Moore-Penrose and G1 actually give the same result. The decoupled optimization is smooth which makes the results plausible.



(a) Map index 0 (Moore-Penrose)

(b) Map index 1 (Drazin)

(c) Map index 2 (G1)

(d) Map index 3 (G2)

Figure 14: Worst-case optimization of risk against σ, λ for a singular forward operator.

To validate our findings, we test the performance of each inverse by investigating the reconstruction error for each inverse. To do this, we randomly generate 100 input data-vectors where x is between

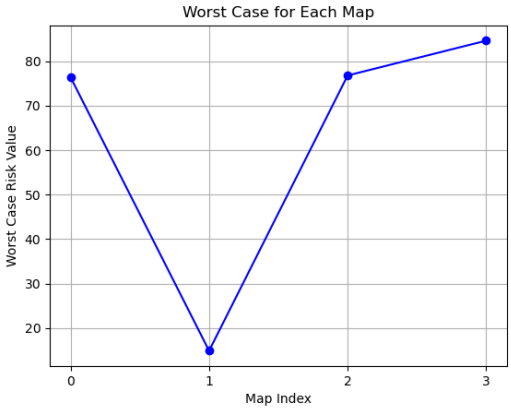
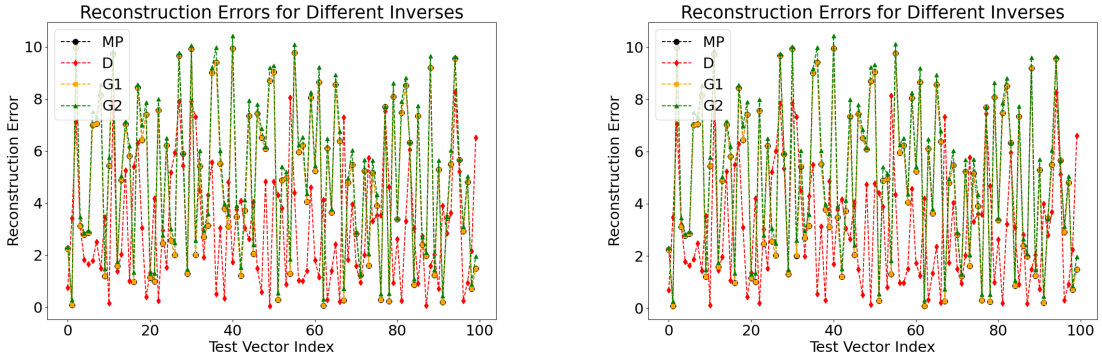


Figure 15: Best worst-case optimization for a singular forward operator.

0 and 10 and generate noiseless measurements $y = Hx$ and noisy measurements $y^\delta = y + \delta$ where δ is normally distributed with variance $\sigma^2 = 0.1^2$. For each data-point (x, y) we calculate the reconstruction error $\left\|x - H^\dagger y\right\|_2$ where $H^\dagger$ denotes the candidate inverse (map) being considered. We do this for every candidate. Figure 24 shows the reconstruction errors for noiseless and noisy data where MP denotes the Moore-Penrose inverse and D denotes the Drazin inverse. The noiseless data gives average reconstruction errors approximately $4.806, 3.123, 4.806, 5.107$ respectively, the noisy data gives approximately $4.807, 3.128, 4.807, 5.107$. The Drazin inverse actually gives the smallest average reconstruction error while the Moore-Penrose and the first generalized inverse perform very similarly. Figure 16 shows that the latter two overlap everywhere. The average reconstruction error is equal for the noiseless data and the difference is approximately 0.0001 for the noisy data. The Drazin inverse gives the smallest average reconstruction error for both noiseless and noisy data which coincides with the result from the Wasserstein-DRO algorithm.



(a) Noiseless data

(b) Noisy data

Figure 16: Reconstruction error for different inverses of singular forward operator.

This particular forward operator is quite tricky as it 'throws away' any knowledge about the second element of the data. Say we have input vector $x = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$. This gives measurement $y = Hx = \begin{bmatrix} x_1 \\ x_1 \end{bmatrix}$. The inverses give the following reconstructions for noiseless data $y = Hx$:

$$H^{\mathrm{MP}} = \begin{bmatrix} x_1 \\ 0 \end{bmatrix}, H^{\mathrm{D}} = \begin{bmatrix} x_1 \\ x_1 \end{bmatrix}, H^{\mathrm{G1}} = \begin{bmatrix} x_1 \\ 0 \end{bmatrix}, H^{\mathrm{G2}} = \begin{bmatrix} x_1 \\ Cx_1 \end{bmatrix}.$$

The forward operator projects the input to a line $y_1 = y_2$ in the measurement space. The MP inverse and G1 project this back into the input space on a line $x_2 = 0$ while the Drazin inverse projects it on a line $x_1 = x_2$ and G2 projects it on a line $x_2 = C \cdot x_1$. In any case, it explains why the performance of

the MP inverse and G1 is equivalent for noiseless data. The performance of each inverse is extremely sensitive to patterns between the two elements in x. If in our dataset the second element is often close to 0, the MP inverse or G1 inverse will perform better and when the second element is often close to the first element, the Drazin or G2 inverse will perform better. Simulations with data specifically generated to reflect these two cases confirm these expectations.

Figure 17 shows the worst-case value for each map when then data has been chosen so the second element is randomly generated between 0 and 0.05. The optimization process for each map has converged with the risk for MP, D, G1 and G2 being respectively $-0.0106, 18.377, -0.0108, -0.0043$. We see now that the MP or G1 inverse would be selected as robust inverse, which is confirmed by plotting the reconstruction errors of each inverse as we did in Figure 18a with noisy data generated similarly as in Figure 17. This yields average reconstruction errors $0.060, 5.356, 0.059, 0.345$, giving the same solution as our Wasserstein-DRO problem: for this data MP and G1 are the best inverses. In Figure 18b we have generated noisy data randomly with elements of x between -10 and 10 to depict more realistic data. This gives average reconstruction errors $5.060, 6.313, 5.060, 5.113$. In this case we see as well that the MP and G1 inverses perform the best. It is likely that if we repeat the Wasserstein-DRO with this data, we would get the same solution. We conclude with the observation that this problem is very sensitive to the nature of the data.

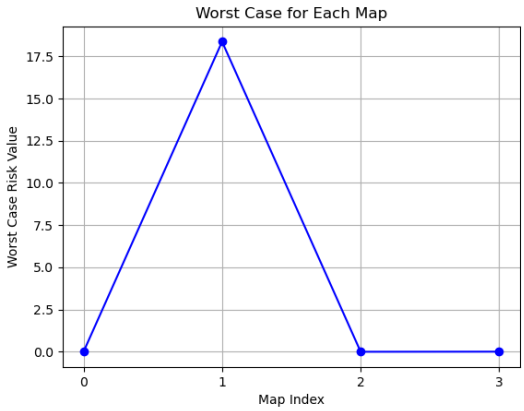
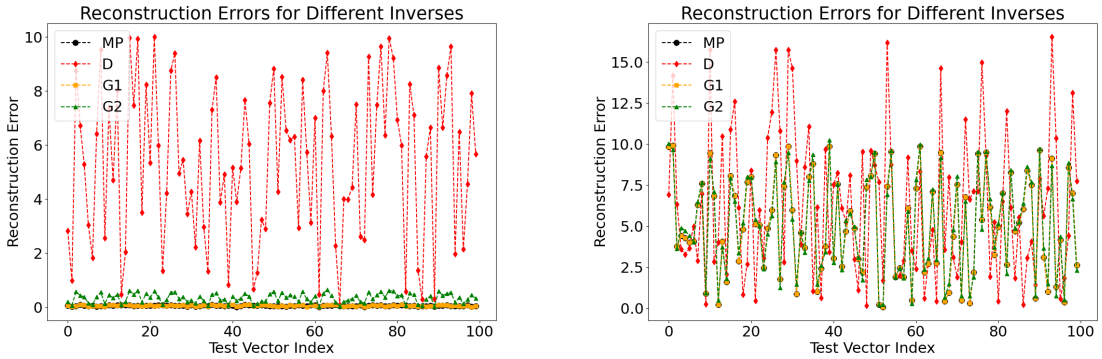


Figure 17: Best worst-case optimization for a singular forward operator, alternative data.

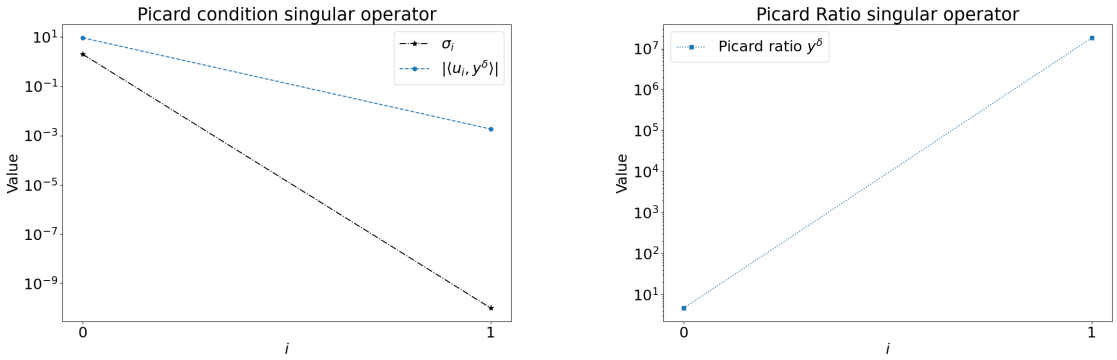


(a) Data with second element of x close to 0 (noisy) (b) Data with values of x between -10 and 10 (noisy)

Figure 18: Reconstruction error for different inverses of singular forward operator, for two datasets

8.2.4 Singular forward operator 2

To better investigate the performance of the Wasserstein-DRO we choose another singular operator that does not 'throw away' the second element of the input data. We choose $H = \begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix}$ which is neither injective nor surjective and has one singular value equal to zero. In a similar way as the previous example, we see through Figure 19 that the Picard condition is not satisfied and this is an ill-posed problem.



(a) Decay of σ_i against $\left|\left\langle u_i, y^\delta \right\rangle\right|$

(b) Picard ratio of y^δ

Figure 19: Picard Condition for a singular forward operator

The Moore-Penrose and Drazin inverse are equal and given by $H^{\text{MP}} = H^D = \begin{bmatrix} 1/4 & 1/4 \\ 1/4 & 1/4 \end{bmatrix}$. Any matrix such that all its elements sum to one is a generalized inverse for this forward operator. We simulate against a Σ containing the Moore-Penrose/Drazin inverse and two randomly generated generalized inverses. We use $\gamma_\lambda = 10^{-4}, \gamma_\sigma = 10^{-4}, \lambda_0 = 6, \sigma_0 = 1$. We use again 600 iterations.

Figure 20 shows the map with smallest worst-case risk is G1. Figure 21 shows the worst-case optimization process. Notably, all optimizations have converged and they all look smooth. This simulation used randomly generated data between 0 and 10. We ran another simulation with data between -50 and 50, which gave very different results, of which none converged. These results can be seen in Figures 22 and 23 which shows the Moore-Penrose/Drazin inverse is selected by the algorithm as the robust solution for this dataset. The optimization processes are still smooth but have not (yet) converged.

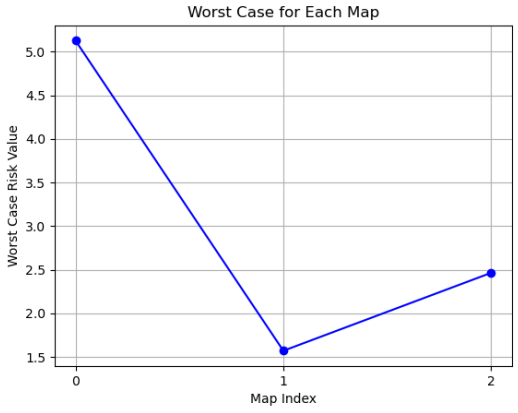
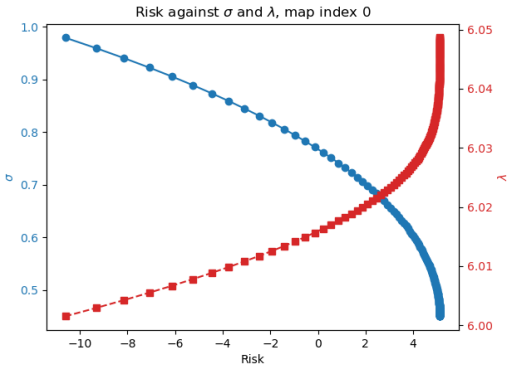
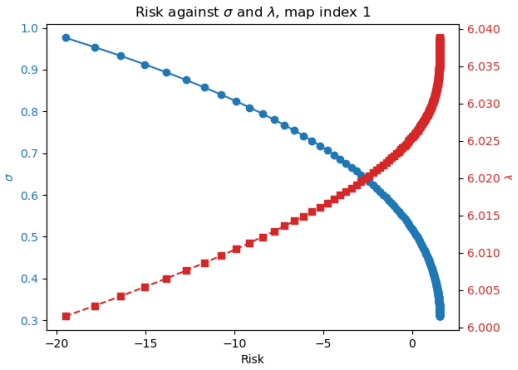


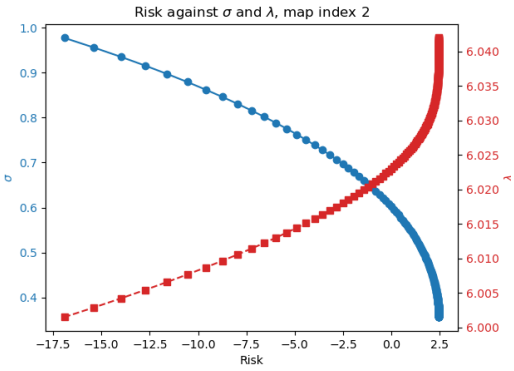
Figure 20: Best worst-case optimization for a singular forward operator.



(a) Map index 0 (Moore-Penrose/Drazin)



(b) Map index 1 (G1)



(c) Map index 2 (G2)

Figure 21: Worst-case optimization for a singular forward operator.

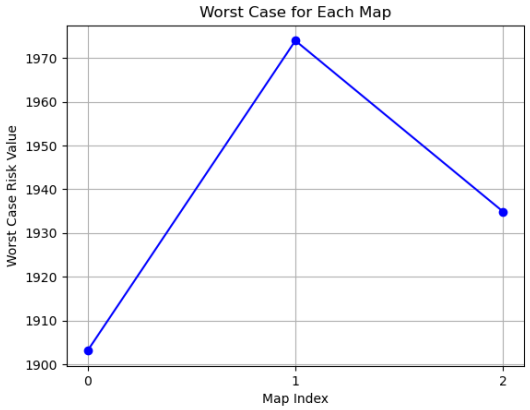
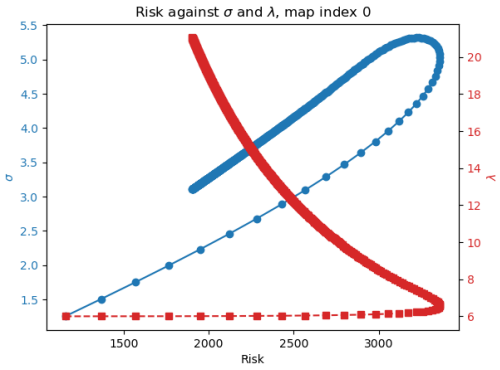
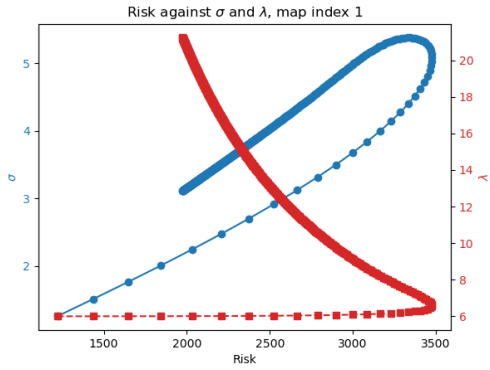


Figure 22: Best worst-case optimization for a singular forward operator, alternative data.

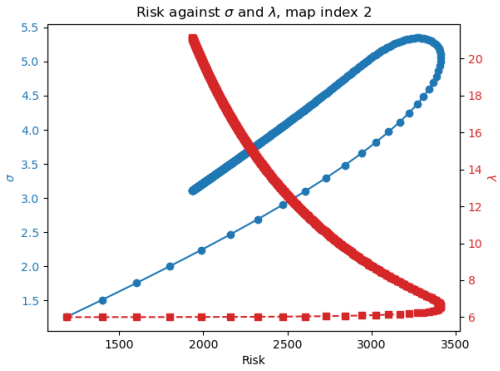
We calculate again the reconstruction errors for each candidate inverse and plot them in Figure 24 with data generated between 0 and 10. The average reconstruction error of the MP/Drazin inverse and the two generalized inverses are respectively $2.231, 2.880, 2.449$ for the noiseless data and $2.232, 2.886, 2.451$ for the noisy data. The average reconstruction error of data generated between -10 and 10 gives similar results. The solution chosen by the Wasserstein-DRO for data between -10 and 10 is the one with the smallest reconstruction error.



(a) Map index 0 (Moore-Penrose/Drazin)

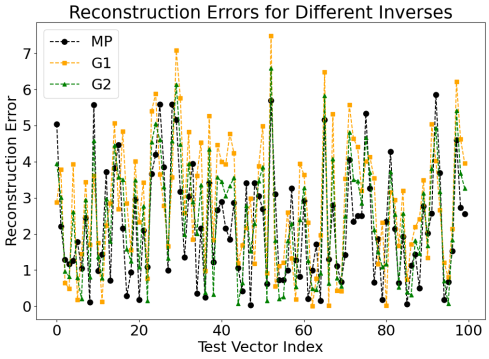


(b) Map index 1 (G1)

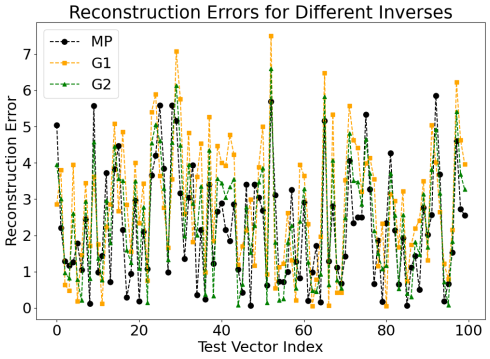


(c) Map index 2 (G2)

Figure 23: Worst-case optimization for a singular forward operator, alternative data.



(a) Noiseless data



(b) Noisy data

Figure 24: Reconstruction error for different inverses of singular forward operator

8.3 Conclusions on numerical results

We have seen that for a non-singular stable forward operator, the Wasserstein-DRO algorithm selects the real inverse as the solution for both cases of Σ and the optimization goes smoothly. For the two singular operators we have looked at, it depends on the data what type of pseudo-inverse is chosen as the solution. However, with data chosen carefully to represent a realistic problem, it seems the Moore-Penrose inverse is often chosen as the most robust one. As we know, the Moore-Penrose inverse corresponds to the least-squares solution. This case used a squared 2-norm as loss function which means the robust solution coincides with the least-squares solution. For the unstable forward operator,

the case where Σ was not necessarily close to the inverse, the real inverse was selected as the solution but for the case of Σ close to the real inverse this was not the case. However, this optimization is not smooth and shows some irregular behavior, indicating that the implementation needs improvement and the decoupled optimization might not be the best approach. Consequently, the results should be interpreted with caution. Especially a better implementation of the learning rate should be considered, for example an adaptive learning rate and/or a restriction on the step size in any way. The handling of parameters being brought below zero could also be improved upon in future implementations. In the objective function $G(\lambda, \sigma; g)$ and both its partial derivatives, there is a division by σ. Since K must be closed in order to apply any of the theorems in this thesis, theoretically 0 (corresponding to the delta-measure) and ∞ should be included as possible values for σ but this is not implemented into the algorithm in any way.

The decoupled optimization for singular and non-singular operators goes smoothly, perhaps with some fine-tuning it will work smoothly for the unstable operator as well. This smooth behavior suggests that the decoupled approach could be a fine approximation.

In any case, the algorithm proposed in this thesis is very slow so future research should consider a more efficient algorithm. There are many hyper-parameters to tune (starting values, learning rates, number of iterations) which makes the implementation tedious. To compute the integrals, we used scipy.integrate, which likely has some faster alternatives, using GPU for example.

While this thesis contains no numerical performance guarantees, the results of the implementation suggest that there lies promise in the constrained Wasserstein-DRO problem. The implementation of the constrained Wasserstein-DRO problem for ill-posed forward operators need careful consideration of learning rates and step sizes while singular forward operators need careful consideration of the data. Still, the optimization process works smoothly and results suggest that the solution chosen as the robust solution to the constrained Wasserstein-DRO problem corresponds with the least-squares solution.

9 Conclusion and outlook

The primary objective of this thesis was to apply Wasserstein robustness to inverse problems and to explore the relationship between robustness and regularization. While the focus was predominantly on the first goal, significant strides were also made towards achieving the second objective.

This thesis encompasses a wide range of topics, including (probability) measure theory, optimization and duality, robustness, optimal transport, inverse problems and linear regression but the most significant contribution of this work is the novel combination of optimal transport and inverse problems as presented herein. We proposed a novel framework integrating Wasserstein robustness within inverse problem modelling using a Bayesian (conditional) approach. The integration of Wasserstein robustness in inverse problems opens new ways to improve the reliability and stability of solutions in various applied mathematics and engineering fields. The approach presented in this thesis has the potential to significantly enhance the robustness of inverse problem-solving methods against noise and ill-posed problems in general.

A thorough analysis of the relevant mathematical concepts was conducted, yielding new general strong duality results. Building on the results by Blanchet et al. [44] for linear regression, we derived a dual representation in a more general sense. By making specific choices for spaces and ambiguity set(s), we can obtain Blanchet's linear regression problem - which allows us to see a connection to regularization - as well as various other types of inverse and forward problems that we wish to be robust to any type of noise.

The theoretical framework we presented was applied to a specific case involving an inverse problem with Gaussian additive noise. Analytical results for this particular case were derived, and a finite-dimensional reduction was presented to make the framework computable. Numerical results for simple forward operators were provided to verify and validate the proposed model. We have left numerical performance guarantees to future research but have shown that while there is room for improvement, we are able to learn a solution to an inverse problem that is robust in the measurement space, for multiple types of simple forward operators (non-singular well-posed, non-singular ill-posed and singular matrices). In the case of a squared 2-norm loss, the robust solution to the constrained Wasserstein-DRO coincides with the least squares solution. The numerical results of the Wasserstein-DRO were further validated by numerical comparisons of the performance of the 'candidate inverses', confirming the solution to the Wasserstein-DRO has the smallest average reconstruction error. Future research could directly compare the results of regularized inverse problems and Wasserstein-DRO problems to further establish a connection between regularization and robustification. The unstable forward matrices need improvement in the implementation as they require more careful consideration of learning rates and step sizes. These matrices have a large Picard ratio, which makes them difficult to solve even with Tikhonov regularization. If we were to improve our implementation it could mean that Wasserstein-DRO handles the ill-posedness better than regularization. It is simple to rewrite the considered problem to an inverse problem with robustness in the input space so fair to assume that numerical simulations for this case will also be successful.

The academic impact case of addressing a general inverse problem with Gaussian noise is significant, as this scenario is encountered in many fields. Thus, it is an interesting case that holds importance for both practical and research purposes. For future research, it would be interesting to complexify the problem by using a multivariate Gaussian distribution in the definition of K, where we have a covariance matrix that would be able to capture the nature of our data. This would likely make the Wasserstein-DRO for inverse problems more powerful. Other than the case presented in this thesis, there are many more interesting cases to explore: different types of inverse problems (i.e. non-linear problems such as convolutions), robustness to noise on the input space instead of the measurement space (or both), and robustness to other types of noise (i.e. applied Poisson noise). See Section 5.1.1 for an exploration of noise and problem types.

The general framework in this thesis can help us find a dual representation for many types of problems that we wish to robustify to any type of noise, not just specifically applied to inverse problems. While this thesis has laid a solid foundation, the general framework only goes so far. To evaluate the dual

representation and/or reduce the problem to a finite-dimensional one, choices need to be made with regards to the spaces and ambiguity set(s) to represent a more specific problem. This thesis has made some insights into the relationship between robustness and regularization but the general framework presented within this thesis and its alternative representations can be a great starting point to continue the exploration of this relationship.

This thesis has a strong focus on mathematical theory and analysis of the concepts presented. It is only natural that future research should expand on the numerical work done in this thesis. In order to make the proposed framework useful in practical applications, research must be done to increase the computational efficiency in order to work with higher-dimensional data, more complicated sets of 'candidate inverses' (Σ) and more complicated forward operators. The computational efficiency could be increased by using the Sinkhorn distance (a type of entropic regularization [51]) and employing a more powerful algorithm than the simple alternating gradient descent/ascent used in this thesis. Perhaps incorporating neural networks could help to incorporate a larger set of candidate inverses. In any case, the algorithm would be more powerful if the inner min-max problem was coupled to the outer min-problem, as our current approach is de-coupled.

If the numerical methods can be improved upon, the proposed framework has promise to be useful in many fields such as signal processing, medical imaging, wavefield imaging, data assimilation and engineering, where inverse problems are prevalent and robustness is crucial.

In conclusion, this thesis represents a significant advancement in the fields of distributionally robust optimization and inverse problems by introducing a novel framework that integrates Wasserstein robustness into inverse problem modeling. We have shown that we are able to learn solutions to inverse problems through data, yielding a robust inverse operator. The idea is that the regularization is done inherently by the robustification process. Through a thorough analysis of mathematical concepts and the development of analytical and computational methods, this research makes some significant steps to gain valuable insights into the relationship between robustness and regularization. The analytical results show an inherent connection between robustness and regularization. The innovative combination of optimal transport and inverse problems presented herein not only expands the theoretical foundations but also holds promise for practical applications in diverse fields. By addressing the limitations of existing approaches and proposing new avenues for future research, this thesis sets a solid foundation for further exploration and advancement in robust inverse problem-solving methods. With its interdisciplinary approach and thorough analysis, this research holds significant value in enhancing the reliability and stability of solutions to inverse problems across various disciplines.

Bibliography

[1] Fatih Yaman, Valery G Yakhno, and Roland Potthast. A Survey on Inverse Problems for Applied Sciences. *Mathematical Problems in Engineering*, 2013:976837, 2013. ISSN 1024-123X. doi: 10.1155/2013/976837. URL https://doi.org/10.1155/2013/976837.

[2] Matthias J. Ehrhardt and Lukas F. Lang. Lecture Notes "Inverse Problems in Imaging", University of Cambridge, 2018.

[3] Simon Arridge, Peter Maass, Carola-Bibiane Schönlieb, and Ozan Öktem. Solving inverse problems using data-driven models. *Acta Numerica*, 28:1–174, 2019. ISSN 0962-4929. doi: DOI:10.1017/S0962492919000059. URL https://www.cambridge.org/core/product/CE5B3725869AEAF46E04874115B0AB15.

[4] Per Christian Hansen. Truncated Singular Value Decomposition Solutions to Discrete Ill-Posed Problems with Ill-Determined Numerical Rank. *SIAM Journal on Scientific and Statistical Computing*, 11(3):503–518, 1990. ISSN 0196-5204. doi: 10.1137/0911028.

[5] Leonid I Rudin, Stanley Osher, and Emad Fatemi. Nonlinear total variation based noise removal algorithms. *Physica D: Nonlinear Phenomena*, 60(1):259–268, 1992. ISSN 0167-2789. doi: https://doi.org/10.1016/0167-2789(92)90242-F. URL https://www.sciencedirect.com/science/article/pii/016727899290242F.

[6] A. N. Tikhonov, A. V. Goncharsky, V. V. Stepanov, and A. G. Yagola. *Numerical Methods for the Solution of Ill-Posed Problems*. Springer Netherlands, Dordrecht, 1 edition, 1995. ISBN 978-90-481-4583-6. doi: 10.1007/978-94-015-8480-7.

[7] Daniel Kuhn, Peyman Mohajerin Esfahani, Viet Anh Nguyen, and Soroosh Shafieezadeh-Abadeh. Wasserstein Distributionally Robust Optimization: Theory and Applications in Machine Learning. In *Operations research & management science in the age of analytics*, pages 130–166. Informs, 2019.

[8] Marcello Carioni, Subhadip Mukherjee, Hong Ye Tan, and Junqi Tang. Unsupervised approaches based on optimal transport and convex analysis for inverse problems in imaging. *arXiv preprint*, 2023.

[9] Andreas Hauptmann, Subhadip Mukherjee, Carola-Bibiane Schönlieb, and Ferdia Sherry. Convergent regularization in inverse problems and linear plug-and-play denoisers. *arXiv preprint*, 2023.

[10] Gregory Ongie, Ajil Jalal, Christopher A Metzler, Richard G Baraniuk, Alexandros G Dimakis, and Rebecca Willett. Deep Learning Techniques for Inverse Problems in Imaging. *IEEE Journal on Selected Areas in Information Theory*, 1(1):38–56, 2020. doi: 10.1109/JSAIT.2020.2991563. URL http://www.ieee.org/publications.

[11] Markus Haltmeier and Linh Nguyen. Regularization of Inverse Problems by Neural Networks. In Ke Chen, Carola-Bibiane Schönlieb, Xue-Cheng Tai, and Laurent Younes, editors, *Handbook of Mathematical Models and Algorithms in Computer Vision and Imaging: Mathematical Imaging and Vision*, pages 1065–1093. Springer International Publishing, Cham, 2023. ISBN 978-3-030-98661-2. URL https://doi.org/10.1007/978-3-030-98661-2_81.

[12] Peijie Qiu. Data-Driven Approaches to Solve Inverse Problems. *Master of Science (MSc) Thesis, Washington University, St. Louis*, 2021. doi: 10.7936/vsqs-hc87.

[13] Sebastian Lunz, Ozan Öktem, and Carola-Bibiane Schönlieb. Adversarial Regularizers in Inverse Problems. *arXiv preprint*, 2019.

[14] M A G Duff, N D F Campbell, and M J Ehrhardt. Regularising Inverse Problems with Generative Machine Learning Models. *Journal of Mathematical Imaging and Vision*, 66(1):37–56, 2024. ISSN 1573-7683. doi: 10.1007/s10851-023-01162-x. URL https://doi.org/10.1007/s10851-023-01162-x.

[15] Subhadip Mukherjee, Marcello Carioni, Ozan Öktem, and Carola-Bibiane Schönlieb. End-to-end reconstruction meets data-driven regularization for inverse problems. *arXiv preprint*, 2021.

[16] Subhadip Mukherjee, Sören Dittmer, Zakhar Shumaylov, Sebastian Lunz, Ozan Öktem, and Carola-Bibiane Schönlieb. Learned convex regularizers for inverse problems. 2021.

[17] Giovanni S. Alberti, Ernesto De Vito, Matti Lassas, Luca Ratti, and Matteo Santacesaria. Learning the optimal Tikhonov regularizer for inverse problems. *Advances in Neural Information Processing Systems*, 34:25205–25216, 2021.

[18] Erich Kobler, Alexander Effland, Karl Kunisch, and Thomas Pock. Total Deep Variation: A Stable Regularization Method for Inverse Problems. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(12):9163–9180, 2022. ISSN 19393539. doi: 10.1109/TPAMI.2021.3124086.

[19] Babak Maboudi Afkham, Julianne Chung, and Matthias Chung. Learning regularization parameters of inverse problems via deep neural networks. *Inverse Problems*, 37(10):105017, 2021. ISSN 0266-5611. doi: 10.1088/1361-6420/ac245d. URL `https://iopscience.iop.org/article/10.1088/1361-6420/ac245d`.

[20] Ankit Raj, Yoram Bresler, and Bo Li. Improving Robustness of Deep-Learning-Based Image Reconstruction. In Hal Daumé III and Aarti Singh, editors, *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 7932–7942. PMLR, 2020. URL `https://proceedings.mlr.press/v119/raj20a.html`.

[21] Julianne Chung and Matthias Chung. An efficient approach for computing optimal low-rank regularized inverse matrices. *Inverse Problems*, 30(11):114009, 2014. ISSN 0266-5611. doi: 10.1088/0266-5611/30/11/114009. URL `https://iopscience.iop.org/article/10.1088/0266-5611/30/11/114009`.

[22] Yoeri E. Boink and Christoph Brune. Learned SVD: solving inverse problems via hybrid autoencoding. 2019. URL `https://arxiv.org/abs/1912.10840v3`.

[23] Hamed Rahimian and Sanjay Mehrotra. Distributionally Robust Optimization: A Review. *arXiv preprint*, 2019. doi: 10.5802/ojmo.15.

[24] John Duchi and Hongseok Namkoong. Learning Models with Uniform Performance via Distributionally Robust Optimization. 2020.

[25] Viet Anh Nguyen, Soroosh Shafieezadeh-Abadeh, Daniel Kuhn, and Peyman Mohajerin Esfahani. Bridging Bayesian and Minimax Mean Square Error Estimation via Wasserstein Distributionally Robust Optimization. *arXiv preprint*, 2021.

[26] Xuhui Zhang, Jose Blanchet, Youssef Marzouk, Viet Anh Nguyen, and Sven Wang. Distributionally Robust Gaussian Process Regression and Bayesian Inverse Problems. *arXiv preprint*, 2022.

[27] Yiming Gao. A Wasserstein distance and total variation regularized model for image reconstruction problems. *Inverse Problems and Imaging*, 0(0):0–0, 2023. ISSN 1930-8337. doi: 10.3934/ipi.2023045.

[28] Kristian Bredies, Marcello Carioni, Silvio Fanzon, and Francisco Romero. A Generalized Conditional Gradient Method for Dynamic Inverse Problems with Optimal Transport Regularization. *Foundations of Computational Mathematics*, 23(3):833–898, 2023. ISSN 1615-3383. doi: 10.1007/s10208-022-09561-z. URL `https://doi.org/10.1007/s10208-022-09561-z`.

[29] Howard Heaton, Samy Wu Fung, Alex Tong Lin, Stanley Osher, and Wotao Yin. Wasserstein-Based Projections with Applications to Inverse Problems. *SIAM Journal on Mathematics of Data Science*, 4(2):581–603, 2022. ISSN 2577-0187. doi: 10.1137/20M1376790.

[30] Jonas Adler, Axel Ringh, Ozan Öktem, and Johan Karlsson. Learning to solve inverse problems using Wasserstein loss. 2017.

[31] Soroosh Shafieezadeh-Abadeh, Daniel Kuhn, and Peyman Mohajerin Esfahani. Regularization via Mass Transportation. *Journal of Machine Learning Research*, 20(103), 2019.

[32] Laurent El Ghaoui and Hervé Lebret. Robust Solutions to Least-Squares Problems with Uncertain Data. *Society for Industrial and Applied Mathematics*, 18(4):15, 1997. URL `http://www.siam.org/journals/simax/18-4/29813.html`.

[33] Dimitris Bertsimas and Martin S. Copenhaver. Characterization of the equivalence of robustification and regularization in linear and matrix regression. *European Journal of Operational Research*, 270(3):931–942, 2018. ISSN 0377-2217. doi: 10.1016/J.EJOR.2017.03.051.

[34] Rui Gao, Xi Chen, and Anton J Kleywegt. Wasserstein Distributionally Robust Optimization and Variation Regularization. *Operations Research*, 2022. ISSN 0030-364X. doi: 10.1287/opre.2022.2383. URL `https://doi.org/10.1287/opre.2022.2383`.

[35] Jose Blanchet, Yang Kang, and Karthyek Murthy. Robust Wasserstein Profile Inference and Applications to Machine Learning. 2020. doi: 10.1017/jpr.2019.49.

[36] Terrence Tao. An Introduction To Measure Theory. 2011. URL `https://api.semanticscholar.org/CorpusID:117492913`.

[37] Cédric Villani. *Optimal Transport*, volume 338 of *Grundlehren der mathematischen Wissenschaften*. Springer, Berlin, Heidelberg, 2009. ISBN 978-3-540-71049-3. doi: 10.1007/978-3-540-71050-9. URL `http://link.springer.com/10.1007/978-3-540-71050-9`.

[38] Cédric Villani. *Optimal transport, old and new*. Springer, Berlin Heidelberg New York Hong Kong London Milan Paris Tokyo, 2008.

[39] Tristan van Leeuwen and Christoph Brune. 10 Lectures on Inverse Problems and Imaging, 2023. URL `https://tristanvanleeuwen.github.io/IP_and_Im_Lectures/intro.html`.

[40] Adi Ben-Israel and Thomas N.E. Greville. *Generalized inverses: theory and applications*, volume 15. Springer Science & Business Media, 2003.

[41] M. P. Drazin. Pseudo-Inverses in Associative Rings and Semigroups. *The American Mathematical Monthly*, 65(7):506–514, 1958. ISSN 00029890. doi: 10.2307/2308576.

[42] E.H. Moore. On the reciprocal of the general algebraic matrix. *Bulletin of the American Mathematical Society*, 26(9):394–395, 1920.

[43] Roger Penrose. A generalized inverse for matrices. *Proceedings of the Cambridge Philosophical Society*, 51(3):406–413, 1955.

[44] Jose Blanchet and Karthyek Murthy. Quantifying Distributional Model Risk via Optimal Transport. *https://doi.org/10.1287/moor.2018.0936*, 44(2):565–600, 2019. ISSN 15265471. doi: 10.1287/MOOR.2018.0936. URL `https://pubsonline.informs.org/doi/abs/10.1287/moor.2018.0936`.

[45] David G. Luenberger. Optimization by vector space methods. page 326, 1968. URL `https://www.wiley.com/en-us/Optimization+by+Vector+Space+Methods-p-9780471181170`.

[46] Lieve Vandenberghe. Proximal Mapping Lecture Notes University of California, Los Angeles Spring, 2022.

[47] Ruidi Chen and Ioannis Ch. Paschalidis. Distributionally Robust Learning. *Foundations and Trends in Optimization*, 4(1-2):1–243, 2020. ISSN 2167-3888. doi: 10.1561/2400000026.

[48] Peyman Mohajerin Esfahani and Daniel Kuhn. Data-driven distributionally robust optimization using the Wasserstein metric: performance guarantees and tractable reformulations. *Mathematical Programming*, 171(1-2):115–166, 2018. ISSN 0025-5610. doi: 10.1007/s10107-017-1172-1.

[49] Charalambos D. Aliprantis and Owen Burkinshaw. *Principles of Real Analysis*. Academic Press, 3rd edition, 1998.

[50] Alexander Shapiro. On Duality Theory of Conic Linear Problems. In *Semi-Infinite Programming: Recent Advances*, pages 135–165. Springer US, Boston, MA, 2001. ISBN 978-1-4757-3403-4. URL https://doi.org/10.1007/978-1-4757-3403-4_7.

[51] Marco Cuturi. Sinkhorn distances: lightspeed computation of optimal transport. In *Proceedings of the 26th International Conference on Neural Information Processing Systems - Volume 2*, NIPS'13, pages 2292–2300, Red Hook, NY, USA, 2013. Curran Associates Inc.

A Appendix

A.1 Python code for constrained Wasserstein-DRO

This appendix contains the code used to implement the Wasserstein-DRO from section 8.

```
import numpy as np
from scipy.integrate import quad, nquad
import matplotlib.pyplot as plt
import os
from multiprocessing import Pool

######################################################################
# This code implements the Constrained Wasserstein-DRO problem for matrix inversion
# with Gaussian additive noise as introduced in the Master Thesis
# "Learning Distributionally Robust Solutions for Inverse Problems using the
    Wasserstein Distance"
# by Floor van Maarschalkerwaart.
######################################################################


def vector_norm(vector):
    ##### Calulcates 2-norm of a vector
    return np.linalg.norm(vector)


def loss(xi, y, g):
    ##### Calculates the loss function as squared 2-norm for
    # xi: x-value
    # y: y-value
    # g: candidate inverse

    global n
    g = np.squeeze(g)

    if measurement == 'matrix':
        y = y.reshape((n, n))

    xhat = g @ y

    if not np.isscalar(y):
        xi = xi.ravel()
        xhat = xhat.ravel()

    return np.linalg.norm(xi - xhat)**2


def seperate_integrals(xi, integrand, lamb, sig, g, H):
    #### Calculates a single integrand
    # xi: x-value
    # integrand: integrand to be evaluated
    # lamb: lambda-value
    # sig: sigma-value
    # g: candidate inverse
    # H: forward map

    global measurement, nn, ab
    if measurement == 'scalar':
        si = quad(integrand, -np.inf, np.inf, args=xi)[0]
    else:
        integrand_fixed_xi = lambda *y: integrand(*y, xi=xi, lamb=lamb, sig=sig, g=g,
    H=H)
        lower_bounds = np.ones(nn) * (-ab)
        upper_bounds = np.ones(nn) * ab
        bounds = [(lower_bounds[i], upper_bounds[i]) for i in range(nn)]

        si = nquad(integrand_fixed_xi, bounds)[0]
    return si


def integrate_sum(integrand, lamb, sig, g, H):
```

```
    #### Calculates four integrals in parallel
    # integrand: integrand to be evaluated
    # lamb: lambda-value
    # sig: sigma-value
    # g: candidate inverse
    # H: forward map

    global eps, n, nn, N, x, measurement, ab
    frac = (1 / (N * np.power(2 * np.pi * (sig**2), 1/nn)))  # fraction before
    integral

    pool_args = [(xi, integrand, lamb, sig, g, H) for xi in x]
    with Pool() as pool:
        results = pool.starmap(seperate_integrals, pool_args)
        pool.close()
        pool.join()

    s = sum(results)
    funcval = frac * s

    return funcval


def compute_common_terms(y, xi, sig, H):
    #### Separately compute common terms for efficiency
    # y: y-value
    # xi: x-value
    # sig: sigma-value
    # H: forward operator

    if measurement != 'scalar':
        y = np.array(y).ravel()

    yi = (H @ xi).ravel() if measurement != 'scalar' else H @ xi

    norm_yi_y = np.linalg.norm(yi - y)  # norm of Hx - y
    exponent = np.exp(-norm_yi_y**2 / (2 * sig**2))  # exponent in beginning of
    integrand

    return yi, norm_yi_y, exponent


def integrand_G(*y, xi, lamb, sig, g, H):
    #### Integrand in G(lambda, sigma; g)
    yi, norm_yi_y, exponent = compute_common_terms(y, xi, sig, H)

    term = loss(xi, y, g) - lamb*norm_yi_y**2
    funcval = exponent * term

    return funcval


def integrand_lamb(*y, xi, lamb, sig, g, H):
    #### Integrand in dG/dlambda (sigma; g)
    yi, norm_yi_y, exponent = compute_common_terms(y, xi, sig, H)
    funcval = exponent * norm_yi_y**2

    return funcval


def integrand_sig(*y, xi, lamb, sig, g, H):
    #### Integrand in dG/dsigma (lambda, sigma; g)
    global nn
    yi, norm_yi_y, exponent = compute_common_terms(y, xi, sig, H)
    term = (loss(xi, y, g) - lamb*norm_yi_y**2)*(sig**(-3)*norm_yi_y**2 - 2/(nn*sig))
    funcval = exponent * term
    return funcval


def G(lamb, sig, g):
    global eps, H
```

```
    funcval = lamb * eps + integrate_sum(integrand_G, lamb, sig, g, H)
    return funcval


def dGdlamb(lamb, sig, g):
    global eps, H
    funcval = eps - integrate_sum(integrand_lamb, lamb, sig, g, H)
    return funcval


def dGdsig(lamb, sig, g):
    global H
    funcval = integrate_sum(integrand_sig, lamb, sig, g, H)
    return funcval


def gradient_descent(lamb0, sig0, g):
    #### Gradient descent algorithm for lambda
    # lamb0: starting value of lambda
    # sig0: starting value of sigma
    # g: candidate inverse
    # Returns: lambda

    global lr1, sub_iterations, lambs, H
    lamb = lamb0
    sig = sig0

    for i in range(sub_iterations):
        # make sure lambda >= 0
        new_lamb = max(lamb - lr1*dGdlamb(lamb, sig, g), 0)
        if abs(new_lamb - lamb) > 50:
            # if jump in lambda is too large, adapt to a smaller learning rate
            new_lamb = lamb - (lr1*1e-3) * dGdlamb(lamb, sig, g)
        lamb = new_lamb
    return lamb


def gradient_ascent(lamb0, sig0, g):
    #### Gradient ascent algorithm for sigma
    # lamb0: starting value of lambda
    # sig0: starting value of sigma
    # g: candidate inverse
    # Returns: sigma

    global lr2, sub_iterations, sigs, H
    lamb = lamb0
    sig = sig0

    for i in range(sub_iterations):
        new_sig = sig + lr2 * dGdsig(lamb, sig, g)
        if abs(new_sig - sig) > 50:
            # if jump in sigma is too large, adapt to a smaller learning rate
            new_sig = sig + (lr2*1e-4) * dGdsig(lamb, sig, g)
        if new_sig < 0:
            # if sigma < 0, adapt to a smaller learning rate
            new_sig = sig + (lr2*1e-4) * dGdsig(lamb, sig, g)
        if new_sig > 0:
            # make sure sigma > 0
            sig = new_sig
        else:
            print('sigma below zero so unchanged')

    return sig


def alternating_grad(lamb0, sig0, g):
    #### Alternating gradient descent/ascent algorithm for lambda/sigma
    # lamb0: starting value of lambda
    # sig0: starting value of sigma
    # g: candidate inverse
    # Returns: optimal lambda, sigma and objective
```

```
    global total_iterations, sub_iterations, lr1, lr2, err, lambs, sigs, H, sigs,
    lambs

    lamb = lamb0
    sig = sig0

    sigs = []
    lambs = []
    opts = []

    for i in range(total_iterations):
        lamb = gradient_descent(lamb, sig, g)
        lambs.append(lamb)

        sig = gradient_ascent(lamb, sig, g)
        sigs.append(sig)

        opt = G(lamb, sig, g)
        opts.append(opt)

        # Print information every 10 iterations
        if i % 10 == 0:
            print(f'iteration {i}')
            print(f'current lambda = {lamb}')
            print(f'current sigma = {sig}')
        if i > 1:
            # Quit algorithm if parameters have converged
            if abs(sigs[i] - sigs[i - 1]) < err and abs(sigs[i - 1] - sigs[i - 2]) <
    err and abs(lambs[i] - lambs[i - 1]) < err and abs(lambs[i - 1] - lambs[i - 2]) <
    err:
                print(f'converged parameters at it = {i}')
                break
            # Quit algorithm if objective has converged
            if abs(opts[i] - opts[i - 1]) < err and abs(opts[i - 1] - opts[i - 2]) <
    err:
                print(f'converged objective  at it = {i}')
                break
    return lamb, sig, opts


def minmax(g):
    #### Calculates the worst-case value for a particular candidate inverse 'g'
    # Returns: optimal value, optimal lambda, optimal sigma
    # and a vector 'opts' containing the objective value for each iteration

    global total_iterations, sub_iterations, lr1, lr2, lamb_start, sig_start, H

    lamb0 = lamb_start
    sig0 = sig_start

    lamb_opt, sig_opt, opts = alternating_grad(lamb0, sig0, g)
    opt = G(lamb_opt, sig_opt, g)

    return opt, lamb_opt, sig_opt, opts


def worstcase(g, i):
    #### Plots the vector containing the objective value for each iteration
    # against sigma and lambda and saves it to a file
    # g: candidate inverse
    # i: index of candidate inverse
    # Returns: optimal values of lambda, sigma and the objective function

    global basepath, total_iterations, lamb0, sig0, sub_iterations, lr1, lr2, lambs,
    sigs, H

    opt, lamb_opt, sig_opt, opts = minmax(g)
    print(f'Total risk = {opt}')
    print(f'Optimal lambda = {lamb_opt}')
    print(f'Optimal sigma = {sig_opt}')

```

```
    fig, ax1 = plt.subplots()

    # Plot worst cases against sigs
    color = 'tab:blue'
    ax1.set_xlabel('Risk')
    ax1.set_ylabel(r'$\sigma$', color=color)
    ax1.plot(opts, sigs, 'o-', color=color, markersize=3)
    ax1.tick_params(axis='y', labelcolor=color)

    # Create a secondary y-axis to plot lambs
    ax2 = ax1.twinx()
    color = 'tab:red'
    ax2.set_ylabel(r'$\lambda$', color=color)
    ax2.plot(opts, lambs, 's--', color=color, markersize=3)
    ax2.tick_params(axis='y', labelcolor=color)

    plt.title(fr'Risk against $\sigma$ and $\lambda$, map index {i}')
    plt.plot()
    # plt.show()

    # Save file
    file_path = os.path.join(basepath, f'optimization_map_{i}.png')
    plt.savefig(file_path)
    plt.close()

    return opt, lamb_opt, sig_opt


def best_worstcase(gs):
    #### Calculates the best-worst case g and the corresponding objective value and plots the worst-case for each g
    # and saves plot to a file as well as a .txt file containing the results and some initial values
    # gs: list of candidate inverses to optimize over

    global basepath, total_iterations, H, lamb_start, sig_start

    worstcases = []
    lamb_opts = []
    sig_opts = []

    # For each candidate inverse (or 'map'), calculate the worst-case lambda, sigma and objective
    for i, g in enumerate(gs):
        print(f'testing map index {i}')
        opt, lamb_opt, sig_opt = worstcase(g, i)
        worstcases.append(opt)
        lamb_opts.append(lamb_opt)
        sig_opts.append(sig_opt)

    # Plot worst-case for each g
    x_values = list(range(0, len(worstcases)))
    plt.plot(x_values, worstcases, marker='o', linestyle='-', color='b', markersize=3)
    plt.title('Worst Case for Each Map')
    plt.xlabel('Map Index')
    plt.ylabel('Worst Case Risk Value')
    plt.grid(True)
    plt.xticks(x_values)
    # plt.show()

    # Save file
    path = os.path.join(basepath, 'best_worstcase'+'.png')
    plt.savefig(path)

    # Find best-worst case and corresponding lambda, sigma, g
    funcval = min(worstcases)
    best_worstcase_idx = worstcases.index(funcval)
    best_worstcase_lamb = lamb_opts[best_worstcase_idx]
    best_worstcase_sig = sig_opts[best_worstcase_idx]
    best_worstcase_g = gs[best_worstcase_idx]

```

```
    # Save results to a .txt file
    output = [
        f'Best worst case g = {best_worstcase_g}, idx = {best_worstcase_idx}\n',
        f'Best worst case lambda = {best_worstcase_lamb}\n',
        f'Best worst case sigma = {best_worstcase_sig}\n',
        f'Best worst case risk = {funcval}\n',
        f'Sigma = {gs}\n',
        f'forward map H = {H}\n',
        f'starting value lamb = {lamb_start}, sig = {sig_start}'
    ]

    result_filename = os.path.join(basepath, 'results.txt')

    with open(result_filename, 'w') as f:
        f.writelines(output)

    print(f'Results saved to {result_filename}')


def generate_data(measurement, nr_gs, n, N, matrix):
    #### Generate data
    # measurement: 'scalar', 'vector' or 'matrix', decides what type of problem we are considering (shape of y)
    # nr_gs: number of 'candidate inverses' to consider
    # n: data-dimension
    # N: number of data-points
    # matrix: 'wellposed', 'illposed', 'singular1' or 'singular2', decides the type of forward operator

    # Returns:
    # x: list of x datapoints
    # y: list of y datapoitns
    # H: forward operator
    # gs: list of candidate inverses

    global pm

    if measurement not in ['scalar', 'vector', 'matrix']:
        print('Wrong problem type')
        return

    # Generate forward operator
    a = 2
    if matrix == 'wellposed':
        D = np.ones(n)*a
        H = D if measurement == 'scalar' else np.diag(D)
    elif matrix == 'illposed':
        D = np.ones(n)
        D[n-1] = 1e-5
        H = D if measurement == 'scalar' else np.diag(D)
    elif matrix == 'singular1':
        H = np.array([[1, 0], [1, 0]])
    elif matrix == 'singular2':
        H = np.array([[1, 1], [1, 1]])
    else:
        print('Wrong forward operator type!')
        return

    x = []
    y = []
    gs = []

    # Generate data x, y
    for _ in range(N):
        xi_shape = (n, n) if measurement == 'matrix' else (n, 1)
        xi = np.squeeze(np.random.uniform(1e-19, 10, size=xi_shape))
        x.append(xi)
        y.append(H @ xi)

    # Generate list of candidate inverses
    if matrix == 'wellposed':
```

```
        # Include the real inverse and perturbations of it
        diag = np.ones(n) * (1 / a)
        inv = diag if measurement == 'scalar' else np.diag(diag)

        for _ in range(nr_gs):
            g_shape = (n, 1) if measurement == 'scalar' else n
            g = np.random.uniform(-pm, pm, size=g_shape)
            g = g + np.ones(g_shape)*(1/a)
            g = np.diag(g) if measurement != 'scalar' else g
            gs.append(g)
        gs.append(inv)

    elif matrix == 'illposed':
        # Include the real inverse and perturbations of it
        diag = np.ones(n)
        diag[n-1] = 1e5
        inv = diag if measurement == 'scalar' else np.diag(diag)

        for _ in range(nr_gs):
            if pm == 'custom':
                g = np.array([np.random.uniform(0, 2),  np.random.uniform(1e3, 1e7)])
                g = np.diag(g)
                gs.append(g)
            else:
                g = np.array([np.random.uniform(-pm*1, pm*1), np.random.uniform(-pm*1
    e5, pm*1e5)])
                g = np.diag(g) + inv
                gs.append(g)
        gs.append(inv)

    elif matrix == 'singular1':
        # Include Moore-Penrose, Drazin inverse and two random generalized inverses
        MP = np.array([[0.5, 0.5], [0, 0]])
        gs.append(MP)

        Drazin = H
        gs.append(Drazin)

        # Two generalized inverses
        alpha = np.random.uniform(0, 1)
        g = np.array([[alpha, 1 - alpha], [0, 0]])  # with 0's on second row
        gs.append(g)
        beta = np.random.uniform(0, 1)
        g = np.array([[beta, 1 - beta], [np.random.uniform(-1, 1), np.random.uniform
    (-1, 1)]])  # anything on second row
        gs.append(g)

    elif matrix == 'singular2':
        # Include Moore-Penrose = Drazin inverse and two random generalized inverses
        MP = np.array([[0.25, 0.25], [0.25, 0.25]])
        gs.append(MP)

        # Two generalized inverses
        g = np.random.rand(2, 2)
        g = g / np.sum(g)
        gs.append(g)
        g = np.random.rand(2, 2)
        g = g / np.sum(g)
        gs.append(g)

    return x, H, y, gs


# Initialize global lists
lambs = []
sigs = []

# Initial values
eps = 0.001  # epsilon
lamb_start = 0.4  # starting value lambda
sig_start = 1.7  # starting value sigma
```

```
lr1 = 0.001  # learning rate lambda
lr2 = 0.001  # learning rate sigma
err = 0.0001  # convergence tolerance (when we consider parameters or objective '
    converged')
nr_gs = 3  # number of candidate inverses in Sigma
sub_iterations = 3  # number of iterations for gradient descent/ascent on lambda/sigma
total_iterations = 800  # total number of iterations of alternating gradient descent/
    ascent
N = 12  # nr of datapoints
n = 2  # data-dimension
pm = 0.2  # perturbation of real inverse
# pm = 'custom'

# Set type of measurement (shape of y)
# measurement = 'scalar'
measurement = 'vector'
# measurement = 'matrix'

# Set type of forward operator
matrix = 'wellposed'
# matrix = 'illposed'
# matrix= 'singular1'
# matrix = 'singular2'

# Set right dimensions
if measurement == 'matrix':
    nn = n*n
else:
    nn = n

# Set random seed for reproducibility
rs = 348
np.random.seed(rs)

# Bounds for integrals
ab = 30

# Include important information in folder name
info = fr'H={matrix} ab={ab}_it={total_iterations} pm={pm} lr1={lr1} lr2={lr2}'
basepath = os.path.join(fr'C:\Users\floor\Documents\TW\Master\Afstuderen\Figures',
    measurement, info)

# Create folder
if not os.path.exists(basepath):
    os.makedirs(basepath)

if __name__ == "__main__":
    # Generate data
    x, H, y, gs = generate_data(measurement, nr_gs, n, N, matrix)

    # Find best-worstcase g
    best_worstcase(gs)
```

A.2 Python code for Picard condition

This appendix contains the code used to compute the Picard condition in section 8.

```
    import numpy as np
import matplotlib.pyplot as plt

######################################################################
# Calculates Picard condition for ill-posed and two singular matrices
# and plots reconstruction error of different pseudo-inverses for the singular
    matrices
######################################################################

n = 2
ii = [0, 1]

# Construct forward operator
```

```
# Hilbert matrix
# n = 100
# ii = np.linspace(0, 100, n)
# H = hilbert(n)

# ill-posed
# D = np.ones(n)
# D[n - 1] = 1e-5
# H = np.diag(D)

# singular
# H = np.array([[1, 0], [1, 0]])  # s1
H = np.array([[1, 1], [1, 1]])  # s2

# Calculate SVD
U, s, Vh = np.linalg.svd(H, full_matrices=True)

N = 12  # nr of datapoints
sigma = 1e-2  # variance of noise

# Random seed for reproducibility
rs = 348
np.random.seed(rs)

x = []
y = []
y_delta = []

# Generate data
for _ in range(N):
    xi_shape = (n, 1)
    xi = np.squeeze(np.random.uniform(1e-19, 10, size=xi_shape))
    x.append(xi)
    y.append(H @ xi)

# Choose pseudo-inverses to include
# for s1
# gs = []
# MP = np.array([[0.5, 0.5], [0, 0]])
# gs.append(MP)
# Drazin = H
# gs.append(Drazin)
# alpha = np.random.uniform(0, 1)
# g1 = np.array([[alpha, 1 - alpha], [0, 0]])
# gs.append(g1)
# beta = np.random.uniform(0, 1)
# g2 = np.array([[beta, 1 - beta], [np.random.uniform(-1, 1), np.random.uniform(-1, 1)
    ]])
# gs.append(g2)

# for s2
gs = []
MP = np.ones((n, n))*0.25
gs.append(MP)
g = np.random.rand(2, 2)
g1 = g / np.sum(g)
gs.append(g1)
g = np.random.rand(2, 2)
g2 = g / np.sum(g)
gs.append(g2)

# Add noise to data
for i in range(N):
    noise = np.random.rand(n)
    y_delta.append(y[i] + sigma * noise)

idx = 0  # which datapoint to consider

# Add elements from null-space of singular operators (if wanted)
# xi = np.array([1, -1])  # s2
# xi = np.array([0, -1])  # s1
```

```
# yi = H @ xi
# yi_delta = yi + sigma*np.random.rand(n)
# x.append(xi)
# y.append(yi)
# y_delta.append(yi_delta)
# idx = N

# Calculate projections of data with singular values
projections_f = np.abs(U.T @ y[idx])
projections_f_delta = np.abs(U.T @ y_delta[idx])

# For singular matrices, replace 0-singular values with a very small value
small_value = 1e-10  # Small value to replace zeros
s_with_small_value = np.where(s == 0, small_value, s)

# Calculate picard ratio
picard_ratios_f = projections_f / s_with_small_value
picard_ratios_f_delta = projections_f_delta / s_with_small_value

# Plot projections and singular values to inspect Picard condition
font = {'size'   : 22}
plt.rc('font', **font)
fig1, ax = plt.subplots(1, 1, figsize=(12, 8))
ax.semilogy(s + small_value, '*-.', label=r'$\sigma_i$', ms=8, color='black')
ax.semilogy(projections_f + small_value, 'o--', label=r'$|\langle u_i, y\rangle|$')
ax.semilogy(projections_f_delta + small_value, 'o--', label=r'$|\langle u_i, y^{\delta
    }\rangle|$')
ax.semilogy(projections_f + np.sqrt(2/np.pi)*sigma,'k--', label=r'upper bound $|\
    langle u_i, f^{\delta}\rangle|$')
ax.set_xticks(ii)
ax.set_xlabel(r'$i$')
ax.set_ylabel('Value')
ax.set_title('Picard condition')
ax.legend()
plt.show()
plt.close()

# Plot Picard ratio
fig2, ax = plt.subplots(1, 1, figsize=(12, 8))
ax.semilogy(picard_ratios_f + small_value, 's:', label=r'Picard ratio y')
ax.semilogy(picard_ratios_f_delta + small_value, 's:', label=r'Picard ratio $y^{\delta
    }$')
ax.set_xticks(ii)
ax.set_xlabel(r'$i$')
ax.set_ylabel('Value')
ax.set_title('Picard Ratio')
ax.legend()
plt.show()
plt.close()


# Below, calculate reconstruction error for each pseudo-inverse


def reconstruction_error(H_pseudo, f, x):
    # Calculate reconstruction error
    # H_pseudo: pseudo-inverse to consider
    # f: measurement
    # x: ground-truth
    x_reconstructed = H_pseudo @ f
    error = np.linalg.norm(x_reconstructed - x)
    return error


N = 100  # nr of data-points to consider
x = []
y = []
y_delta = []

# Generate new data
for _ in range(N):
```

```
    xi_shape = (n, 1)
    xi = np.squeeze(np.random.uniform(-10, 10, size=xi_shape))
    # xi = np.array([np.random.uniform(0, 10), np.random.uniform(0, 0.05)])
    x.append(xi)
    y.append(H @ xi)

sigma = 0.1  # variance of noise
for i in range(N):
    noise = np.random.rand(n)
    y_delta.append(y[i] + sigma * noise)
y = y_delta

# Calculate errors for each test vector and each pseudo-inverse
errors1 = [reconstruction_error(MP, f, u) for f, u in zip(y, x)]
# errorsD = [reconstruction_error(Drazin, f, u) for f, u in zip(y, x)]
errors2 = [reconstruction_error(g1, f, u) for f, u in zip(y, x)]
errors3 = [reconstruction_error(g2, f, u) for f, u in zip(y, x)]

# Aggregate errors
avg_error1 = np.mean(errors1)
# avg_errorD = np.mean(errorsD)
avg_error2 = np.mean(errors2)
avg_error3 = np.mean(errors3)

print(f"Average reconstruction error for MP: {avg_error1}")
# print(f"Average reconstruction error for Drazin: {avg_errorD}")
print(f"Average reconstruction error for g1: {avg_error2}")
print(f"Average reconstruction error for g2: {avg_error3}")

# Plot errors for visual comparison
plt.figure(figsize=(12, 8))
plt.plot(errors1, 'o--', label='MP', markersize=8, color='black')
# plt.plot(errorsD, 'd--', label='D', color='red')
plt.plot(errors2, 's--', label='G1', color='orange')
plt.plot(errors3, '^--', label='G2', color='green')
plt.xlabel('Test Vector Index')
plt.ylabel('Reconstruction Error')
plt.title('Reconstruction Errors for Different Inverses')
plt.legend(loc=2)
plt.show()
plt.close()
```