Contributions to optimal transport for Machine Learning : Ground Metric and Generalized Framework

Tanguy Kerdoncuff

Numéro d'ordre NNT : 2021LYSES047

THESE de DOCTORAT DE L'UNIVERSITE DE LYON

opérée au sein de :

Univ Lyon, UJM-Saint-Etienne, CNRS, Institut d'Optique Graduate School,
Laboratoire Hubert Curien UMR 5516, F-42023, SAINT-ETIENNE, France.

École Doctorale ED488 Sciences, Ingénierie, Santé

Spécialité : **Informatique**

Soutenue publiquement le 09/12/2021, par:

Tanguy Kerdoncuff

Contributions to Optimal Transport for Machine Learning: Ground Metric and Generalized Framework

---

Devant le jury composé de :

Élisa FROMONT	Professeure, Université de Rennes 1	Présidente
Marianne CLAUSEL	Professeure, Université de Lorraine	Rapporteuse
Nicolas COURTY	Professeur, Université Bretagne Sud	Rapporteur
Marc SEBBAN	Professeur, Université de Saint-Étienne	Directeur
Rémi EMONET	Maître de conférences, Université de Saint-Étienne	Co-encadrant

Acknowledgements

First, I would like to thank my two thesis's referees Marianne Clausel, Nicolas Courty and the president of my thesis defense Elisa Fromont for their proofreading of my manuscript and their advice and research discussions.

My deepest thanks go to my supervisors, Marc Sebban who was always present to supervise my research and guide me into the research world and Rémi Emonet which always helped me in the mathematical or computational problems I encountered in my everyday life as a PhD student.

It was a great pleasure to collaborate with Sofien Dhouib and Ievgen Redko for their impressive mathematical knowledge, with Michael Perrot for its rigor and perspicacity and also with Rémi Viola and Thibaud Leteno on this fascinating subject of fraud detection.

Life in the laboratory was awesome with notably all the PhD students and postdocs: Emmanuel, Valentina, Guillaume Metzler, Nam, Kevin, Jordan, Léo, Rémi Viola, Rémi Vaudaine, Guillaume Muller, Jules, Paul, Manvy, Michael, Raphael, Édouardo, Volodimir.

I would like also to thank Thomas Gautrais for his precious help concerning the cluster and all my technical computer issues.

Finally, I want to thank my partner, Florence, for her patience and support during the difficult periods of my thesis.

I dedicate this thesis to my grandmother, who left too soon.

Contents

Introduction **1**

List of Publications **5**

List of Notations **7**

1 Background **9**

1.1 Optimal Transport (OT) . . . 9

1.1.1 Mathematical settings . . . 9

1.1.2 Optimal Transport solvers . . . 13

1.1.3 Gromov Wasserstein . . . 18

1.1.4 OT barycenter . . . 20

1.2 Machine Learning . . . 21

1.2.1 Machine Learning setting . . . 22

1.2.2 Domain Adaptation . . . 22

1.2.3 OT-based distances versus Kullback-Leiber divergence in Machine Learning 24

1.3 First order optimization tools for OT problems . . . 25

1.3.1 Frank-Wolfe algorithm . . . 26

1.3.2 Projected Mirror Descent . . . 29

1.4 Conclusion . . . 32

2 Metric Learning for Optimal Transport **33**

2.1 Introduction . . . 33

2.2 OTDA and Metric Learning . . . 35

2.2.1 OTDA . . . 35

2.2.2 Metric Learning . . . 36

2.3 Theoretical analysis of Domain Adaptation with OT . . . 37

2.3.1 PCA and Wasserstein Distance . . . 37

2.3.2 Upper Bound on the Target Risk . . . 38

2.4 MLOT: Metric Learning in OT for DA . . . 40

2.5 Experiments . . . 41

2.5.1 Datasets . . . 41

2.5.2 Setup and Cross-validation . . . 41
2.5.3 Analysis of the Results . . . 42
2.6 Conclusion . . . 44

3 A Swiss Army Knife for Minimax Optimal Transport 47
3.1 Introduction . . . 47
3.2 Preliminary Knowledge . . . 48
3.3 Robust Optimal Transport with a convex set of cost Matrices . . . 49
3.3.1 Problem formulation . . . 50
3.3.2 Choice of $\mathcal{C}$ . . . 50
3.3.3 Proposed optimization strategy . . . 53
3.3.4 Variations for different choices of $\mathcal{C}$ . . . 57
3.3.5 Towards the notion of stability of cost matrices . . . 57
3.4 Experiments . . . 58
3.4.1 Convergence and execution time . . . 59
3.4.2 Comparison to SRW . . . 59
3.4.3 Stability and noise sensitivity . . . 60
3.4.4 Color transfer . . . 61
3.5 Conclusion . . . 62

4 Sampled Gromov Wasserstein 65
4.1 Introduction . . . 65
4.2 Background on Gromov Wasserstein (GW) . . . 67
4.3 Approaches to solve GW . . . 67
4.4 Scalable GW optimization . . . 69
4.4.1 Sampled Gromov Wasserstein (SaGroW) . . . 69
4.4.2 Convergence analysis . . . 71
4.4.3 Particular case: Pointwise GW . . . 73
4.4.4 A KL regularization-based variant . . . 75
4.4.5 Efficient computation the GW distance from a transport plan . . . 76
4.5 Experiments . . . 78
4.5.1 General setup and methods . . . 78
4.5.2 Speed and accuracy of the GW estimate . . . 81
4.5.3 Hyperparameters analysis . . . 81
4.5.4 Graph classification . . . 83
4.5.5 Continuous Gromov Wasserstein . . . 84
4.6 Conclusion . . . 85

5 Optimal Tensor Transport 87
5.1 Introduction . . . 87

5.2 Preliminary knowledge . . . 89
5.3 Optimal Tensor Transport (OTT) . . . 90
5.4 Algorithm to solve OTT . . . 92
5.5 Theoretical results . . . 93
5.6 Experiments . . . 94
5.6.1 Domain Adaptation (DA) . . . 95
5.6.2 Comparison based clustering using OTT barycenters . . . 97
5.7 Conclusion . . . 99

A Appendix of Metric Learning for Optimal Transport 105
A.1 Domain Adaptation experiments . . . 105
A.1.1 Complete table of the main DA experiment . . . 105
A.1.2 Metric Learning and OTDA separately . . . 105

B Appendix of Swiss-Army Knife for Optimal Transport 109
B.1 Proofs from Section 3.3.2 . . . 109
B.2 Proofs from Section 3.3.3 . . . 111
B.3 Proofs from Section 3.3.4 . . . 114
B.4 Experimental evaluations . . . 117

C Appendix of Sampled Gromov Wasserstein 121
C.1 Scalable GW optimization . . . 121
C.1.1 Detailed derivations for the convergence . . . 121
C.1.2 A KL regularization-based variant . . . 128
C.1.3 Approximating the Gromov Wasserstein distance . . . 128
C.2 Experiments . . . 129
C.2.1 General setup and methods . . . 129
C.2.2 Speed and accuracy of the GW estimate . . . 130
C.2.3 Hyperparameter analysis . . . 131
C.2.4 Small experiment on SaGroW without the KL regularization . . . 133
C.2.5 Small experiment on the entropy parameter . . . 134
C.2.6 Small experiment on the α parameter . . . 134
C.2.7 Graph classification . . . 135

D Appendix of Optimal Tensor Transport 139
D.1 Illustration of the difference between OTT and Co-OT . . . 139
D.2 Mirror Descent . . . 140
D.3 Complexity of the gradient computation of OTT . . . 140
D.4 Theoretical results . . . 142
D.5 Experiments . . . 148
D.5.1 The number of samples M . . . 148

D.5.2 Domain Adaptation (DA) 149
D.5.3 Comparison based clustering using OTT barycenter 165

E Unfinished research related to the contributions of this thesis 169
E.1 Pointwise Wasserstein and Sliced Wasserstein 169
E.2 Optimal Slide Transition 170
E.2.1 Algorithm details 170
E.2.2 Practical details on how to use OST 171
E.3 Gromov Wasserstein and Wasserstein GAN 171

F Résumé en Français 175

List of Figures 194

List of Tables 196

List of Algorithms 197

Bibliography 199

Abstract 213

Introduction

Machine learning is a subfield of Artificial Intelligence which aims at providing algorithms that simulate human intelligence to solve complex tasks. A peculiarity of a Machine Learning process is that it does not require to be explicitly programmed and relies instead on training examples. These latter are described either directly with features in a vector space (e.g., weight, age, price,...) or with a structured representation (e.g., picture, text, sound,...). In some applications, the samples are represented only relatively to each other, for instance in graphs made of pairwise connections between examples, like in social networks or in molecules. The idea behind Machine Learning is to design an algorithm that solves a desired task by learning a function from the available dataset and will generalize well on unseen examples. One of the most common tasks is supervised binary classification, where the goal is to distinguish between two classes. For instance, given a training set of labelled pictures of cats and dogs, a classification algorithm learns to discriminate between the two categories. It will be correct if it predicts well the class of unseen pictures. This generalization capacity is typically estimated from a so-called test set. In the last decade, Machine Learning has received a large amount of attention with various target applications, in computer vision (semantic segmentation, object tracking), arts (image or music generation), fraud detection, to cite a few.

In many machine learning scenarios, comparing two empirical probability measures is of great interest. This can occur when evaluating the similarity between two images represented by point clouds in the RGB space, between two texts encoded through word embeddings or between two sets of cells whose genome/transcriptome is numerically described in a feature space.

One possible direction to address this task is to use the Optimal Transport (OT) theory, originally introduced by Gaspard Monge (Monge, 1781), aiming at answering the following question: How to move resources from some locations to satisfy requirements at others locations with the least effort, in the sense that the global cost of moving the resources is minimized. For example, the resources can be soldiers coming from different military bases that should be sent to different locations of the front line, and the military commander wants to minimize the average travel cost of the soldiers to avoid unnecessary efforts. As originally introduced, the Monge problem prevented the possibility to split the soldiers coming from the same military base, leading to an ill-posed problem with no guarantee of uniqueness and existence of the solution. More than 200 years later, Leonid Kantorovitch (Kantorovich, 1942) proposed a relaxed mathematical formulation of the transportation problem which notably defines a real distance

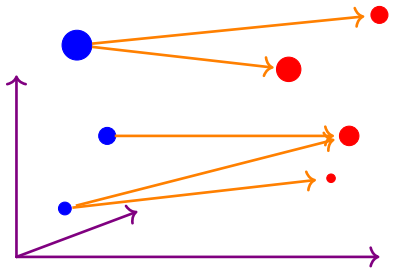


Figure 1: *Illustration of the Optimal Transport problem between two discrete distributions.*
Cited on page [1]

between probability distributions: the Wasserstein distance. This latter has received a great attention during the past few years by the Machine Learning community as a powerful tool to compare probability measures. An example of the Optimal Transport solution is displayed on Figure 1, where the entire mass of the blue distribution is sent on the top of the red distribution by the orange arrows. The Wasserstein distance between the two distributions is the sum of all the orange distances weighted by the corresponding mass transported on each arrow. It turns out that a crucial element of Optimal Transport is the function used as distance between the samples of the two point clouds. In such a discrete setting, the OT problem is entirely defined by a cost matrix, which contains all the pairwise distances between the points of the two probability measures. This cost matrix is usually referred in OT to as *ground metric*, *ground function* or *ground cost*. The most common and natural ground distance is the Euclidean distance, raised to a certain power p, which leads to the so-called p-Wasserstein distance.

The ground metric is at the core of several contributions of this manuscript. In particular, we exploit the metric learning framework to learn a ground metric in the form of a Mahalanobis distance used to address Domain Adaptation tasks, i.e. when we observe a shift of distribution between the training and testing data. We also study, in the first part of this thesis, from a theoretical perspective how to learn a stable ground metric.

A second series of contributions relies on the scalability of OT. Indeed, some variants aim at comparing probability measures that do not necessarily lie in the same feature space. This is the case of the Gromov Wasserstein distance whose algorithmic complexity prevents its use on large datasets. We propose in this thesis a fast algorithm for this problem based on sampling strategies. Building upon this algorithm, we further introduce an extension generalizing both the Wasserstein and the Gromov Wasserstein problems, which allows us to apply OT on high dimensional tensors.

Funding and context of this thesis. This thesis is part of the TADALoT Project[1], funded by the region Auvergne-Rhône-Alpes (France) with the Pack Ambition Recherche (2017, 17 011047 01). This thesis was carried out in the Data Intelligence team of the Hubert Curien laboratory. This laboratory is a research unit (UMR 5516) between the Jean Monnet University

[1] https://twitwi.github.io/tadalot/

of Saint-Étienne, the CNRS and the Institut d'Optique Graduate School.

Organization The manuscript is composed of 5 chapters.

- The first chapter provides the necessary background for the rest of the document. It introduces in details the Optimal Transport theory, some solvers and some extensions, notably the Gromov-Wasserstein problem. Then, some background knowledge is presented about the standard Machine Learning setting and the Domain Adaptation framework. This chapter also presents the Frank-Wolfe and the Mirror Descent algorithms as key optimization elements for the extended OT formulations.

- The second chapter explores the capacity of learning the ground metric of the Optimal Transport problem for an unsupervised Domain Adaptation task. This latter aims at deploying on a target distribution a model learned from labelled source data. The choice of the euclidean distance to compare points might not be the best choice given the task at hand. Instead, we propose to **learn a Mahalanobis distance** using a Metric Learning approach. The advantage of the Mahalanobis distance comes from its ability to reduce the dimensionality of the feature space by learning a low rank matrix. A generalization bound on the target error is derived to guide the design of our algorithm, called Metric Learning for Optimal Transport (MLOT).

- The third chapter theoretically focuses on the stability of OT in the worst case scenario consisting in selecting **the worst possible cost matrix**. This leads to a min-max problem aiming at reducing the distance while looking for the cost matrix, in a possibly infinite set, that increases this distance the most. The proposed solver, based on an elegant cutting plane method, finds the saddle points of this min-max problem. Some experiments highlight the interest of the algorithm compared to the state of the art in terms of time complexity and noise reduction. Building upon this theoretical framework, we define a notion of *local stability* for the cost matrices in order to select the most stable one. We show that this stability is highly correlated with a noise sensitivity notion. We present some experiments to select a stable cost matrix in a color transfer task.

- The fourth chapter is dedicated to an extension of the Optimal Transport theory to incomparable spaces, known as the Gromov-Wasserstein problem. The existing solvers rely on specific loss functions to compute the gradient efficiently and are not suitable in a general setting. We propose a **fast method to solve the Gromov-Wasserstein problem** for any loss function, which relies on a stochastic approximation of the gradient by resorting to a sampling scheme. We also analyze the convergence to a stationary point of the proposed algorithm which notably includes a proof of convergence for existing Gromov-Wasserstein solvers in the concave case. Lastly, we show that an even better speed improvement can be achieved when the stochastic approximation is based on only one example allowing us to make use of the fast one dimensional Optimal Transport solver.

- The fifth chapter is devoted to the definition of a **new distance between tensors of arbitrary orders**. In the classical Optimal Transport setting, the cost function compares vectors that come from points in a vector space; in the Gromov-Wasserstein problem, one compares vectors (often of size 1) that come from matrices. We propose the Optimal Tensor Transport (OTT) problem, where the loss can compare vectors coming from tensors of any orders. To do so, we propose a framework that allows different transport plans to be used along the different dimensions instead of relying on only one transport plan. With the introduction of a new notion of barycenter for OTT, we show competitive results in a comparison-based clustering task. Additionally, some experiments in Domain Adaptation are conducted on datasets represented as tensors, which highlight the interest of such a new formulation compared to the state of the art.

For the sake of clarity of the manuscript, some figures, experiments as well as some long proofs are relegated in the appendices. Two interesting works, related to this thesis, but not explored to their full potential are available in Appendix E.1 and E.3. Additionally, as a technical contribution, Appendix E.2 presents a software developed during this thesis which creates transitions between the slides of a presentation. Those transitions are optimal, in the sense that they minimize the cost of moving from one slide to another.

List of Publications

Publications

KERDONCUFF Tanguy, EMONET Rémi and SEBBAN Marc. Metric Learning in Optimal Transport for Domain Adaptation. In International Joint Conferences on Artificial Intelligence (IJCAI). 2020. p. 2162-2168. (Kerdoncuff et al., 2020).

DHOUIB Sofien, REDKO Ievgen, KERDONCUFF Tanguy, EMONET, Rémi, and SEBBAN Marc. A Swiss Army Knife for Minimax Optimal Transport. In International Conference on Machine Learning (ICML). 2021. p. 2504-2513. (Dhouib et al., 2020).

KERDONCUFF Tanguy, EMONET Rémi, and SEBBAN Marc. Sampled Gromov Wasserstein. In Machine Learning Journal (MLJ). 2021. p. 2151–2186. (Kerdoncuff et al., 2021).

KERDONCUFF Tanguy, PERROT Michael, EMONET Rémi, and SEBBAN Marc. Optimal Tensor Transport. In Association for the Advancement of Artificial Intelligence (AAAI). 2022. (Kerdoncuff et al., 2022)

List of Notations

Linear algebra

$\mathbb{N}$	The set of integers
$\mathbb{R}$	The set of real numbers
$\mathbb{R}_+$	The set of positive numbers
$[\![1, N]\!]$	The set of all integer between 1 and N
$\mathbb{R}^{I \times K}$	The set of real value matrices of size $I \times K$
$\mathbb{I}_K$	The identity matrix of size $K \times K$
$\boldsymbol{M}_{i\bullet}$	The vector of size K extracted from the matrix $\boldsymbol{M} \in \mathbb{R}^{I \times K}$
$\mathcal{T}_{I_1 \times ... \times I_D}$	The set of real value D-order tensors of size $I_1 \times ... \times I_D$
$\mathcal{T}_{I^D}$	The set of real value cubic D-order tensors of size $I \times ... \times I$
$\boldsymbol{L}_{\bullet j \bullet l}$	The matrix of size $I_1 \times I_3$ extracted from the tensor $\boldsymbol{L} \in \mathcal{T}_{I_1 \times I_2 \times I_3 \times I_4}$
	Matrices and tensors will be represented with a bold uppercase letters
	Vectors will be represented with bold lowercase letters
$\|.\|$	A norm, depending on the context
$\|.\|_F$	The Frobenius norm of a matrix or tensor
$\|.\|_p$	The l_p norm for vectors or Schatten-p norm for matrices
$\langle ., . \rangle$	Scalar product, depending on the context
$\langle ., . \rangle_F$	Frobenius dot product between matrices or tensors
$\boldsymbol{M}^\top$	The transpose of any matrix $\boldsymbol{M}$: $\boldsymbol{M}_{ij}^\top = \boldsymbol{M}_{ji}$
$\mathrm{Tr}(\boldsymbol{M})$	The trace of a square matrix $\boldsymbol{M}$: $\mathrm{Tr}(\boldsymbol{M}) = \sum_i \boldsymbol{M}_{ii}$
$\nabla_x f$ or $\nabla f(x)$	The gradient of the function f evaluated on x
Δ_R	The $(R-1)$ standard simplex $\left\{(c_r)_{r \in [\![1,R]\!]} \in \mathbb{R}_+^R \middle\| \sum_{r=1}^R c_r = 1\right\}$

Probability theory and Optimal Transport

$\mathcal{P}(\mathcal{X})$	The set of probability distributions with support included on the set $\mathcal{X}$
$\delta_{\boldsymbol{x}}$	The Dirac distribution at the point $\boldsymbol{x}$
$\boldsymbol{x} \sim \mu$	Sample of the distribution μ
$\mathbb{E}(C)$	Expectation of a random variable C
$\mathbb{P}(A)$	Probability of the event A
$\mathbb{P}_{x\sim\mu}(A(x))$	Probability of the event $A(x)$
$B(\mathcal{X})$	The set of Borel on $\mathcal{X}$
$g\#\mu$	The push forward operator between a function g and a distribution μ: $\forall \mu \in P(\mathcal{X}), g : \mathcal{X} \rightarrow, B \in B(\mathcal{X})\mathcal{X},\ g\#\mu(B) = \mu(g^{-1}(B))$
$\mathbb{1}_I$	The $(I \times 1)$ column vector full of 1
$\Pi_{\mu\nu}$	The collection of all joint probability measures on $\mathcal{X} \times \mathcal{Y}$ with marginals $\mu \in P(\mathcal{X})$ and $\nu \in P(\mathcal{Y})$
$\mathcal{U}_{\boldsymbol{ab}}$	The set of discrete transport plan with marginals $\boldsymbol{a}$ and $\boldsymbol{b}$: $\{\boldsymbol{T} \in \mathbb{R}_+^{I\times K} \vert \boldsymbol{T}\mathbb{1}_K = \boldsymbol{a}, \boldsymbol{T}^\top \mathbb{1}_I = \boldsymbol{b}\}$
$\mathcal{N}(\boldsymbol{m}, \boldsymbol{V})$	The Gaussian distribution with mean $\boldsymbol{m}$ and covariance $\boldsymbol{V}$

Chapter 1

Background

Abstract

This chapter contains most of the necessary background for the manuscript. It starts with a complete description of the Optimal Transport (OT) theory, as well as a presentation of the most used solvers and some variants of the original OT problem. Then, the second section provides some background about Machine Learning, in particular the classification and Domain Adaptation settings that will be tackled in this manuscript. The last section shortly explains the Frank-Wolfe and Mirror Descent algorithms which will be useful to understand the last two chapters of this manuscript.

1.1 Optimal Transport (OT)

When using the expression Optimal Transport (OT), we generally refer to the related theory. But note that OT can be associated to other terms: the OT *problem* that should be solved; the OT *plan* as the solution of this latter problem; the associated OT *solver*; the OT *barycenters*; and even the OT *distance*. An illustration of an OT barycenter, as formally defined later in Section 1.1.4, is given in Figure 1.1.

1.1.1 Mathematical settings

In its original form, a transport plan allows to map an absolutely continuous probability distribution $\mu \in \mathcal{P}(\mathcal{X})$ onto another one $\nu \in \mathcal{P}(\mathcal{Y})$ for two spaces $\mathcal{X}$ and $\mathcal{Y}$ included in possibly different vector spaces. The admissible mappings are the joint probability distributions in $\mathcal{X} \times \mathcal{Y}$ with marginals μ and ν. More formally, the set of all possible transport plans is defined as,

$$\begin{aligned} \Pi_{\mu\nu} &= \left\{\pi \in \mathcal{P}(\mathcal{X} \times \mathcal{Y}) \,|\, P_x\#\pi = \mu, P_y\#\pi = \nu\right\} \\ &\text{with } P_x : (\mathcal{X} \times \mathcal{Y}) \to \mathcal{X} \text{ and } P_y : (\mathcal{X} \times \mathcal{Y}) \to \mathcal{Y}, \\ &\qquad (x, y) \mapsto x \qquad\qquad (x, y) \mapsto y \end{aligned} \tag{1.1}$$

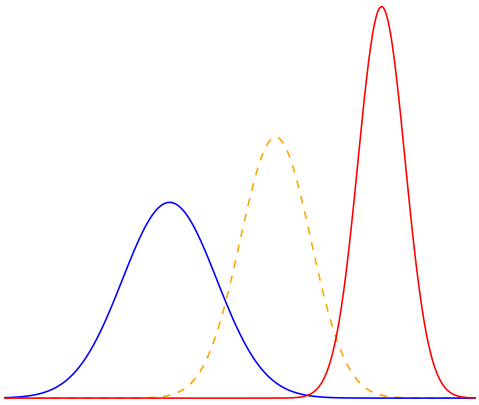


Figure 1.1: *Illustration of the Optimal Transport barycenter (dashed orange line) between two continuous distributions (blue and red).*

Cited on page [9]

where the # symbol corresponds to the push forward operator which intuitively applies a function on every point of the distribution, $\forall A \in B(\mathcal{X}), \mu \in P(\mathcal{X}), g : \mathcal{X} \to \mathcal{X}$, $g\#\mu(A) = \mu(g^{-1}(A))$. This set of transport plans is not empty as the "independent" coupling $\mu \times \nu$ is always in $\Pi_{\mu\nu}$. This coupling distribution will be called the uniform distribution as it is the distribution closest to the uniform one which respects the marginal constraints.

The goal of the OT problem is to find the optimal transport plan, in the sense that it should reduce the global cost of moving μ on the top of ν. To quantify the cost between two points $(x, y) \in (\mathcal{X}, \mathcal{Y})$, a continuous cost function is defined, $c : (\mathcal{X}, \mathcal{Y}) \to \mathbb{R}$. This cost is typically a distance, a pseudo distance or a divergence but can be any continuous function. While the two spaces $\mathcal{X}$ and $\mathcal{Y}$ can in theory be different, they are very often the same to define a meaningful cost c. This cost function is often called the ground function or ground cost. The Optimal Transport solvers seek to solve the Kantorovich problem (Kantorovich, 1942), defined as,

$$\min_{\pi \in \Pi_{\mu\nu}} \mathop{\mathbb{E}}_{x,y\sim\pi} [c(x, y)] = \min_{\pi \in \Pi_{\mu\nu}} \int_{\mathcal{X}\times\mathcal{Y}} c(x, y) d\pi(x, y). \tag{1.2}$$

The existence of such a minimum is guaranteed for particular cost functions (Villani, 2008, Theorem 4.1). Notably, for any continuous cost function, the minimum exists.

The Optimal Transport problem defines a distance between distributions as long as the cost c is a distance (Villani, 2008, theorem 7.3). In fact, the OT problem has the same property as its ground cost. For pseudo-metric costs, the OT problem also leads to a pseudo metric. Some other cost functions can define a real metric between distributions, such as a distance to the power p. Indeed for all $p \geq 1$, the p-Wasserstein (Kantorovich, 1942) distance, for any pair of distributions with finite p-order moments, is defined as,

$$\mathcal{W}_p(\mu, \nu) = \left(\min_{\pi \in \Pi_{\mu\nu}} \int_{\mathcal{X}\times\mathcal{Y}} c(\boldsymbol{x}, \boldsymbol{y})^p d\pi(\boldsymbol{x}, \boldsymbol{y}) \right)^{\frac{1}{p}}. \tag{1.3}$$

In most of the applications, the 2-Wasserstein distance with the Euclidean distance is used, which notably allows to get rid of the square root of the Euclidean distance. Moreover, the p-

Wasserstein distance at the power p, denoted as $\mathcal{W}_p^p(\mu,\nu)$, is often used to compare distributions, as the power function is a strictly increasing function.

The original Monge formulation (Monge, 1781) is also sometimes used for describing the OT problem but, as mentioned in the introduction, this formulation have no solution for some pairs of discrete or semi-discrete distributions. The idea is to find a *function* $f_T : \mathcal{X} \to \mathcal{Y}$ that maps the distribution μ to the distribution ν while minimizing the global cost,

$$\min_{f_T \# \mu = \nu} \int_{\mathcal{X}} c(\boldsymbol{x}, f_T(\boldsymbol{x})) d\mu(\boldsymbol{x}). \tag{1.4}$$

This formulation cannot handle well the discrete case as a point $\boldsymbol{x}$ cannot be sent at different locations. This function f_T is called the Monge mapping. The Kantorovich formulation solves this problem as the distribution π can be seen as some kind of function with stochastic output $(f_T(\boldsymbol{x}) \leftrightarrow \pi_{|\boldsymbol{x}})$.

In Machine Learning, we often have access to empirical distributions $\hat{\mu}$ and $\hat{\nu}$ composed of sums of Diracs that come from real distributions μ and ν. For this reason, the discrete formulation based on finite discrete distributions is presented bellow. First, the $(R-1)$ standard simplex is denoted as $\Delta_R = \left\{(c_r)_{r\in[\![1,R]\!]} \in \mathbb{R}_+^R | \sum_{r=1}^R c_r = 1\right\}$. This allows to define a discrete distribution $\hat{\mu} = \sum_{i=1}^I \boldsymbol{a}_i \delta_{\boldsymbol{x}_i}$ supported by $I \in \mathbb{N}$ points $(\boldsymbol{x}_i)_{i\in[\![1,I]\!]}$ in $\mathcal{X}$ and the associated probability vector $\boldsymbol{a} \in \Delta_I$. In the same way, $\hat{\nu} = \sum_{k=1}^K \boldsymbol{b}_k \delta_{\boldsymbol{y}_k}$ is supported by $K \in \mathbb{N}$ points $(\boldsymbol{y}_k)_{k\in[\![1,K]\!]}$ in $\mathcal{Y}$ associated with the probability vector $\boldsymbol{b} \in \Delta_K$. A transport plan can now be represented as a $(I \times K)$ matrix and the set of all transport plans defined as,

$$\mathcal{U}_{\boldsymbol{ab}} = \left\{\boldsymbol{T} \in \mathbb{R}_+^{I\times K} | \boldsymbol{T}\mathbb{1}_K = \boldsymbol{a}, \boldsymbol{T}^\top \mathbb{1}_I = \boldsymbol{b}\right\}. \tag{1.5}$$

This set is a convex polytope defined by $I \times K + I + K - 1$ linear constraints (Brualdi, 2006). The first $I \times K$ constraints come from the positivity of each element in the matrix; the $I + K$ constraints come from the marginal constraints, and the minus one comes from the redundancy of the marginals which both sum to 1. The Optimal Transport problem can now be written as,

$$\min_{\boldsymbol{T}\in\mathcal{U}_{\boldsymbol{ab}}} \sum_{i=1}^I \sum_{k=1}^K c(\boldsymbol{x}_i, \boldsymbol{y}_k)\boldsymbol{T}_{ik} = \min_{\boldsymbol{T}\in\mathcal{U}_{\boldsymbol{ab}}} \sum_{i=1}^I \sum_{k=1}^K \boldsymbol{C}_{ik}\boldsymbol{T}_{ik} = \min_{\boldsymbol{T}\in\mathcal{U}_{\boldsymbol{ab}}} \langle \boldsymbol{C}, \boldsymbol{T}\rangle, \tag{1.6}$$

with $\boldsymbol{C}$ the cost matrix associated with the cost function c: $\forall (i,k) \in [\![1,I]\!] \times [\![1,K]\!]$ $\boldsymbol{C}_{ik} = c(\boldsymbol{x}_i, \boldsymbol{y}_k)$. For two fixed distributions, this matrix contains all the necessary information of the function c. Notice that the OT problem only has to find the best matrix $\boldsymbol{T}$ instead of the distribution π as $\boldsymbol{T}$ contains all the necessary information to describe π. We will often confuse π and $\boldsymbol{T}$, for example using the slightly abusive notation: $\boldsymbol{T} \in \Pi_{\mu\nu}$. The $\hat{\mu}$ notation will be used essentially to emphasize the fact that the distribution comes from a sampling of a distribution μ. Thus in the rest of the manuscript, μ can describe both a discrete or continuous distribution, depending on the context.

Note that for two continuous distributions in the same vector space, if the cost is the squared Euclidean distance there is an unique solution (Brenier, 1991). Similarly, most of the time in the discrete case, there is an unique optimal transport plan. However, one can construct cases

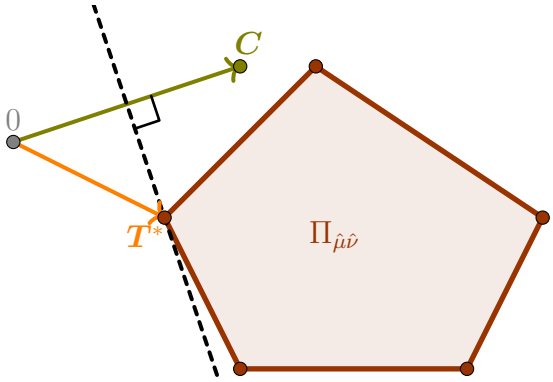


Figure 1.2: *Representation in two dimensions of the Optimal Transport problem. The* $\boldsymbol{C}$ *vector represents the position of the cost matrix in the space. The best transport plan* $\boldsymbol{T}^*$ *is the one which is the "most" perpendicular to the vector* $\boldsymbol{C}$ *because the scalar product has to be minimized.* *Cited on pages [12,12]*

where two transport plans reach the minimum. An example will be given in the next section in Figure 1.5. This will be an important point in this manuscript as a learned cost can have a special form for which one can no longer suppose the uniqueness of the optimal transport plan. This will notably be the case in the Chapter 3 where a min-max formulation will naturally lead to multiple optimal transport plans.

An illustration of the OT problem, as defined in Equation 1.6, is provided in Figure 1.2. The case where the solution is not unique occurred only when an entire face of the polytope is included in the perpendicular hyperplane drawn in dashed line. As the problem is linear in T, at least one of the minima is at the extremal point of the polytope (Bertsimas and Tsitsiklis, 1997). Moreover, this extremal point is sparse i.e., the matrix $\boldsymbol{T}$ has at most $I + K - 1$ non-null values (Peyré et al., 2019). This sparse solution is at the edge (or extremal point) of the polytope in the sense that it cannot be written as a convex combination of the other transport plans (Peyré et al., 2019). The (brown) limit of the polytope $\Pi_{\hat{\mu}\hat{\nu}}$ represents the non-negative constraints on the elements of the transport plan. The marginal constraints are not represented in this two dimensional plot. This notion is illustrated intuitively in the Figure 1.3; the brown polytope is in a $I \times K - (I + K - 1)$ dimensional subspace, represented as a 2D polytope on the figure, into the $I \times K$ dimensional matrix space, represented as 3D.[1] In addition to those marginal equality constraints, the set of all transport plans is bounded by the non-negativity of its elements which are again represented by the edges of the brown polytope in Figure 1.3. While Figures 1.2 and 1.3 try to give as much intuition as possible, there is still some missing information: the edge of the polytope should be on the axis to represent the non-negativity.

If we suppose that the two probability vectors are uniform and $I = K$, we will often use the notation N instead of I and K in such a situation. In this case, the extremal points are simply the permutation matrices (Birkhoff, 1946).

[1] Best viewed in 3D: https://www.geogebra.org/3d/k4neb7tw

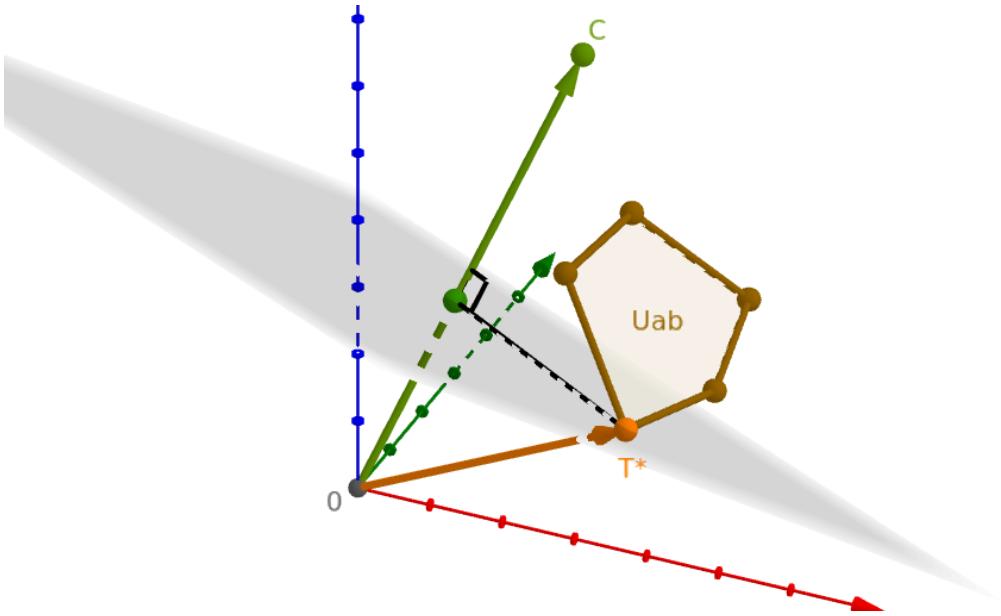


Figure 1.3: *Representation in three dimensions of the Optimal Transport problem. The polytope of the possible transport plans is now represented in a subspace of the matrix space.*
Cited on pages [12,12,12]

1.1.2 Optimal Transport solvers

In this section, we will discuss about the solvers that address the Optimal Transport problem. Note that, there is a large literature aiming at improving the computation of the OT problem in particular cases. For instance, when the dimension of the spaces $\mathcal{X}$ and $\mathcal{Y}$ is small (2 or 3) and for the 1-Wasserstein distance (or EMD) Shirdhonkar and Jacobs (2008) approximate the OT distance in a time complexity of $O(N)$. Three different methods will be used in this manuscript: (i) an exact solver, (ii) one dealing with the very special case of 1-dimensional OT and (iii) the Sinkhorn algorithm.

Direct solutions

The 1-Wasserstein distance is also called the Earth Mover's Distance (EMD), mostly in the computer vision domain. There exist various solvers for the exact optimal transport problem (Rubner et al., 1997; Bonneel et al., 2011). In this manuscript, we will use essentially the EMD algorithm of the POT library (Flamary et al., 2021) originally implemented by (Bonneel et al., 2011). The main idea is to interpret the OT problem as a particular case of the min cost flow problem (Ahuja et al., 1988) in a bipartite graph and use the network simplex algorithm to solve it. The most important element is that the time complexity of this simplex algorithm is $O\left(N^3 log(N)\right)$ (Tarjan, 1997) (see also (Peyré et al., 2019, Section 3.5)) in the worst case, however in practice it is often very fast for a reasonable number of points N.

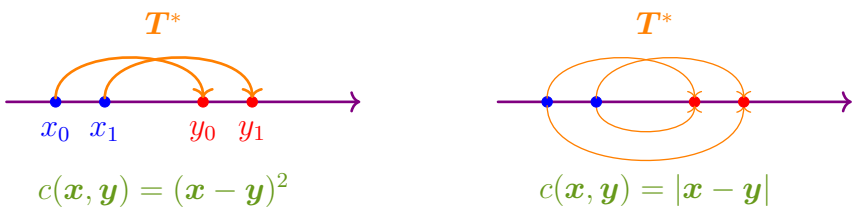


Figure 1.4: *1D Optimal Transport (**Left**) Optimal transport plan with any strictly convex cost function. (**Right**) Any transport plan is optimal with the absolute cost, which is less intuitive.*
Cited on pages [13,13]

1-dimensional Optimal Transport problem

When $\mathcal{X} = \mathcal{Y} = \mathbb{R}$, the OT problem has a solution in closed form which can be computed very quickly as the time complexity is drastically reduced. In addition, let us suppose that the ground function can be written as $c(x, y) = l(x - y)$ where l is a strictly convex function. In the 1-dimensional continuous case, with F_μ and F_ν the two cumulative distribution functions, the Monge mapping is $F_\nu^{-1}(F_\mu(x)) : \mathbb{R} \to \mathbb{R}$. In the discrete case, with the same assumption on c, the OT plan can be computed by sorting the two lists of samples and then matching them together. This is easy to understand and implement when the two distributions have the same number of points and uniform weights: the elements of each sorted list are matched one by one (see the left part of Figure 1.4). In the general case, the weights of the marginal have to be carefully respected. An efficient implementation is available in the POT library. Note that if the function c is only convex but not strictly convex, there are potentially many optimal transport plans as illustrated on the right of Figure 1.4. This is another intuitive justification of the interest of the p-Wasserstein distance ($p > 1$), as the Euclidean distance is the absolute function in 1D which is not strictly convex. As the optimal transport plan is the same for any strictly convex cost function, this latter has an impact only on the value of the distance. Note that for the concave 1D case, Caracciolo et al. (2020) propose an upper-bound on the average cost of the optimal matching but no fast solver is available.

Sinkhorn algorithm

Cuturi (2013) proposes to solve a regularized version of the Optimal Transport problem. The idea is to add an entropy regularization term to ensure that the OT problem becomes strictly convex and then solve it efficiently. One can define the entropy of a distribution represented with a matrix $\boldsymbol{T}$ as,

$$\mathcal{H}(T) = -\sum_{i=1}^{I}\sum_{k=1}^{K} \boldsymbol{T}_{ik} log(\boldsymbol{T}_{ik}) \tag{1.7}$$

The entropy quantifies the uncertainty of a distribution. For example, a Dirac is a distribution with no uncertainty with a 0 entropy. The entropy criterion is often used when no enough information to describe a distribution is available (Jaynes, 1957). For instance, in a compact set

included in a vector space, the uniform distribution maximizes the entropy. Another example is the Gaussian distribution, which maximizes the entropy for a fixed mean and variance. In the OT setting, the uniform transport plan $\mu \times \nu \in \Pi_{\mu\nu}$ will be often used as a default choice that maximizes the entropy.

Cuturi (2013) proposes to add to the Equation (1.6) an entropy regularization term weighted by a regularization parameter $\epsilon \in \mathbb{R}_+^*$,

$$\min_{\boldsymbol{T} \in \mathcal{U}_{\boldsymbol{ab}}} \sum_{i=1}^{I} \sum_{k=1}^{K} \boldsymbol{C}_{ik} \boldsymbol{T}_{ik} - \epsilon \mathcal{H}(\boldsymbol{T}). \tag{1.8}$$

Because the entropy is maximized, the solution of Equation (1.8) will no longer be sparse. This regularization acts as a barrier function and does not allow any null value in the transport plan. The regularization does not diverge when an element of $\boldsymbol{T}$ tends to 0 but the slope tends to infinity while the first term is always linear in $\boldsymbol{T}$. Thus, close to an edge point of the polytope, a small modification of $\boldsymbol{T}$ will highly change the entropy term but not really the scalar product. The optimal transport plan for Equation (1.8) is not anymore an extremal point of the set of all transport plans. Since Equation (1.8) is strictly convex in $\boldsymbol{T}$, the solution cannot be on the extremal point of the polytope, and thus the solution is unique leading to *the* optimal transport plan.

An equivalent formulation is obtained when using a Kullback-Leiber (KL) (Kullback and Leibler, 1951) regularization instead of the entropy regularization. The KL between two distributions represented as matrices $\boldsymbol{T}$ and $\boldsymbol{T}'$ can be defined as,

$$KL(\boldsymbol{T}||\boldsymbol{T}') = \sum_{i,k=1}^{I,K} \boldsymbol{T}_{ik} log\left(\frac{\boldsymbol{T}_{ik}}{\boldsymbol{T}'_{ik}}\right) \tag{1.9}$$

In fact, the entropy regularization is equivalent to a KL divergence between the transport plan and the uniform distribution $\mu \times \nu$. As mentioned earlier, this is a natural choice as there is no particular prior on the optimal transport plan. Replacing the entropy with the KL in Equation (1.8), gives the following equality,

$$\min_{\boldsymbol{T} \in \mathcal{U}_{\boldsymbol{ab}}} \sum_{i,k=1}^{I,K} \boldsymbol{C}_{ik} \boldsymbol{T}_{ik} + \epsilon KL(\boldsymbol{T}||\mu \times \nu) \tag{1.10}$$

$$= \min_{\boldsymbol{T} \in \mathcal{U}_{\boldsymbol{ab}}} \sum_{i,k=1}^{I,K} \boldsymbol{C}_{ik} \boldsymbol{T}_{ik} + \epsilon \left(\sum_{i,k=1}^{I,K} \boldsymbol{T}_{ik} log\left(\boldsymbol{T}_{ik}\right) - \boldsymbol{T}_{ik} log\left(\boldsymbol{a}_i \boldsymbol{b}_k\right) \right) \tag{1.11}$$

$$= \min_{\boldsymbol{T} \in \mathcal{U}_{\boldsymbol{ab}}} \sum_{i,k=1}^{I,K} \boldsymbol{C}_{ik} \boldsymbol{T}_{ik} + \epsilon \left(-\mathcal{H}(T) - \sum_{i,k=1}^{I,K} \boldsymbol{T}_{ik} \left(log(\boldsymbol{a}_i) + log(\boldsymbol{b}_k)\right) \right) \tag{1.12}$$

$$= \min_{\boldsymbol{T} \in \mathcal{U}_{\boldsymbol{ab}}} \sum_{i,k=1}^{I,K} \boldsymbol{C}_{ik} \boldsymbol{T}_{ik} - \epsilon \mathcal{H}(\boldsymbol{T}) - \epsilon \sum_{i=1}^{I} \boldsymbol{a}_i log(\boldsymbol{a}_i) - \epsilon \sum_{k=1}^{K} \boldsymbol{b}_k log(\boldsymbol{b}_k). \tag{1.13}$$

The last two terms are independent of $\boldsymbol{T}$, thus this KL regularization will change the final minimum but not the argmin compared to the entropy regularization. Note that $\boldsymbol{a}\boldsymbol{b}^\top$ (the

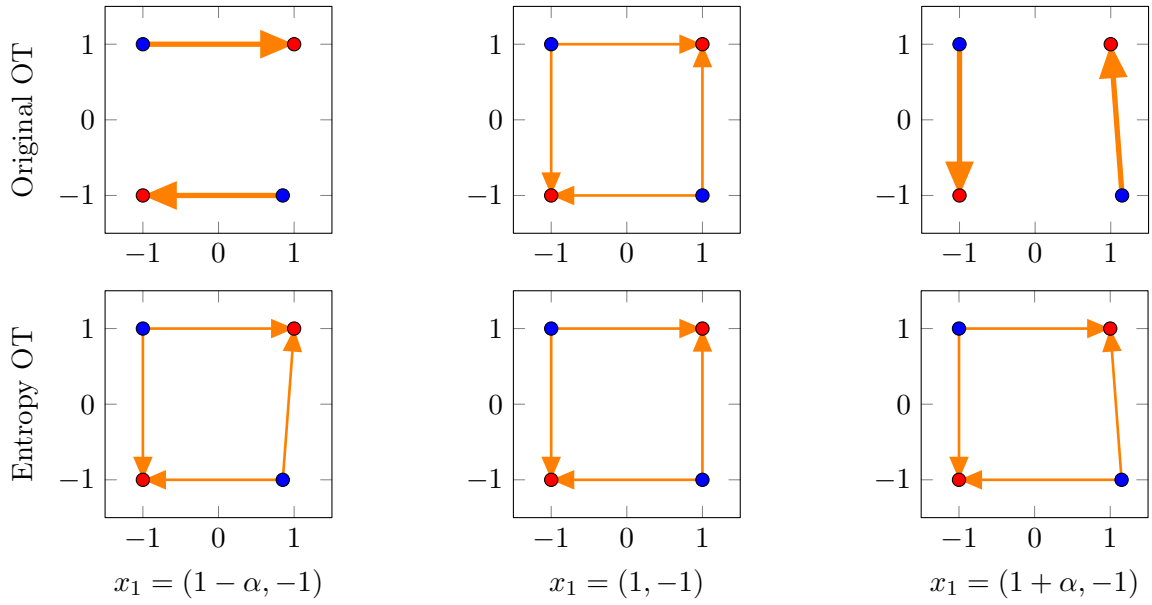


Figure 1.5: *Representation of the discontinuity of the transport plan for a small modification of the points. The top row corresponds to the optimal transport plan for the original OT problem, the bottom row corresponds to the regularized version. From the left to the right, the bottom right blue point is slightly modified,* $(1 \pm \alpha, -1)$ *with a small* α.

Cited on pages [11,16]

matrix associated with $\mu \times \nu$) does not modify the final $\boldsymbol{T}^*$ because it is a rank 1 matrix and thus can be separated. And the only rank 1 matrix which respects the marginal constraint is $\boldsymbol{a}\boldsymbol{b}^\top$. This interpretation is interesting as we will see that sometimes there are better priors than the uniform distribution.

Another advantage of the entropy regularized formulation is that the OT problem becomes continuous, in the sense that a small modification of the distributions will lead to small modifications of the transport plan. This is illustrated in Figure 1.5. By slightly moving the bottom right blue point, the transport plan of the regularized version does not change much. However the transport plan of the original OT problem is completely modified.

The last advantage of such an entropy regularization comes from the simple and fast algorithm that can be used to solve it. Let $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ be the dual variables of Equation (1.8). The Lagrangian $\mathcal{L}a$ is defined as follows:

$$\mathcal{L}a(\boldsymbol{T}, \boldsymbol{\alpha}, \boldsymbol{\beta}) = \sum_{i,k=1}^{I,K} \boldsymbol{T}_{i,k}\boldsymbol{C}_{i,k} + \epsilon \boldsymbol{T}_{i,k} log(\boldsymbol{T}_{i,k}) + \boldsymbol{\alpha}^\top(\boldsymbol{T}\mathbb{1}K - \boldsymbol{a}) + \boldsymbol{\beta}^\top(\boldsymbol{T}^\top\mathbb{1}I - \boldsymbol{b}). \tag{1.14}$$

To find the optimal $\boldsymbol{T}^*$, setting the derivative with respect to $\boldsymbol{T}$ to 0, we get for all $i \in [\![1, I]\!]$ and $k \in [\![1, K]\!]$,

$$0 = \boldsymbol{C}_{ik} + log(\boldsymbol{T}_{ik}) + 1 + \boldsymbol{\alpha}_i + \boldsymbol{\beta}_k \Rightarrow \boldsymbol{T}_{ik} = e^{-\frac{1}{2}-\frac{\boldsymbol{\alpha}_i}{\epsilon}} e^{-\frac{\boldsymbol{C}_{ik}}{\epsilon}} e^{-\frac{1}{2}-\frac{\boldsymbol{\beta}_k}{\epsilon}}. \tag{1.15}$$

The Sinkhorn theorem (Sinkhorn and Knopp, 1967) states that there exists a unique matrix in $\mathcal{U}_{\boldsymbol{ab}}$ of the form $diag(\boldsymbol{u})\boldsymbol{K}diag(\boldsymbol{v})$ if all the elements of $\boldsymbol{u}, \boldsymbol{v}$ and $\boldsymbol{K}$ are strictly positive. More

importantly, this matrix, which corresponds to $\boldsymbol{T}$, can be computed by using the Sinkhorn algorithm (see Algorithm 1) which applies alternatively a KL projection of the matrix $\boldsymbol{K} = e^{-\frac{\boldsymbol{C}}{\epsilon}}$ to μ and ν (lines 4 and 5). The proof of convergence of this alternated projection is studied in (Sinkhorn and Knopp, 1967).

Algorithm 1 Sinkhorn algorithm to solve the regularized Optimal Transport problem between 2 discrete distributions μ and ν. Cited on pages [16,17,30]

Input: ϵ (entropy regularization), $\boldsymbol{a}$ and $\boldsymbol{b}$ (marginals), $\boldsymbol{C}$ (cost matrix)

1: Initialize $\boldsymbol{u}, \boldsymbol{v} = \mathbb{1}_I, \mathbb{1}_K$
2: $\boldsymbol{K} = exp(-\frac{\boldsymbol{C}}{\epsilon})$
3: **for** $p = 1$ **to** P **do**
4: $\quad \forall i \in [\![1, I]\!]\ \boldsymbol{u}_i = \boldsymbol{a}_i / (\boldsymbol{K}\boldsymbol{v})_i$
5: $\quad \forall k \in [\![1, K]\!]\ \boldsymbol{v}_k = \boldsymbol{b}_k / (\boldsymbol{K}^\top \boldsymbol{u})_k$
6: **end for**
7: **return** $diag(\boldsymbol{u})\boldsymbol{K}diag(\boldsymbol{v})$

The Sinkhorn algorithm for OT has received a lot of attention recently and many fast variants have been proposed (Altschuler et al., 2017; Schmitzer, 2019). Moreover, there is a large literature which aims at improving the convergence bounds (Altschuler et al., 2017; Dvurechensky et al., 2018; Lin et al., 2019). As in practice Algorithm 1 has as finite number of iterations, the marginals are not totally respected. In fact, only the marginal associated with ν is satisfied. The difference between the other marginal of $\boldsymbol{T}$ and the real one μ can be used as an early stopping criterion for the algorithm. To simplify the theoretical analysis in this manuscript, the number of iterations P will be supposed to be fixed without any early stopping. Thus we are looking for the worst case in terms of time complexity.

Another interpretation of the Sinkhorn algorithm can be obtained through the lens of the dual formulation of the original OT problem (Peyré et al., 2019, Proposition 2.4) in the discrete case,

$$\max_{\boldsymbol{\alpha} \in \mathbb{R}^I, \boldsymbol{\beta} \in \mathbb{R}^K \mid \boldsymbol{\alpha}_i + \boldsymbol{\beta}_k \leq \boldsymbol{C}_{ik}} \langle \boldsymbol{\alpha}, \boldsymbol{a} \rangle + \langle \boldsymbol{\beta}, \boldsymbol{b} \rangle . \tag{1.16}$$

Both the function to minimize and the constraints are linear in $\boldsymbol{T}$, $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$, thus the value obtained by the primal and the dual are equal. Similarly, the dual of the regularized OT problem (Equation (1.8)) can be written as (Peyré et al., 2019, Proposition 4.4),

$$\max_{\boldsymbol{\alpha} \in \mathbb{R}^I, \boldsymbol{\beta} \in \mathbb{R}^K} \langle \boldsymbol{\alpha}, \boldsymbol{a} \rangle + \langle \boldsymbol{\beta}, \boldsymbol{b} \rangle - \epsilon \left\langle e^{\frac{\boldsymbol{\alpha}}{\epsilon} - \frac{1}{2}}, \boldsymbol{K} e^{\frac{\boldsymbol{\beta}}{\epsilon} - \frac{1}{2}} \right\rangle . \tag{1.17}$$

The key difference is that the dual variables $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ are no longer constrained. This motivates an alternate gradient ascent of Equation (1.17), where each gradient step can be computed in closed form. With a fixed $\boldsymbol{\beta}$, the optimal value of $\boldsymbol{\alpha}$ is

$$\boldsymbol{\alpha}^* = \epsilon log(\boldsymbol{a}) - \epsilon log(\boldsymbol{K} e^{\frac{\boldsymbol{\beta}}{\epsilon} - \frac{1}{2}}) + \epsilon \frac{1}{2}. \tag{1.18}$$

Similarly, for a fixed $\boldsymbol{\alpha}$, the optimal value of $\boldsymbol{\beta}$ is

$$\boldsymbol{\beta}^* = \epsilon log(\boldsymbol{b}) - \epsilon log(\boldsymbol{K}^\top e^{\frac{\boldsymbol{\alpha}}{\epsilon}-\frac{1}{2}}) + \epsilon\frac{1}{2}. \tag{1.19}$$

It is worth noting that this is equivalent to the Sinkhorn iterations with the change of variables $\boldsymbol{u} = e^{\frac{\boldsymbol{\alpha}}{\epsilon}-\frac{1}{2}}$ and $\boldsymbol{v} = e^{\frac{\boldsymbol{\beta}}{\epsilon}-\frac{1}{2}}$.

Lastly, the regularized OT problem can also be seen as a KL projection of the matrix $\boldsymbol{K}$ into the polytope $\Pi_{\mu\nu}$ (Benamou et al., 2015),

$$\min_{\boldsymbol{T}\in\Pi_{\mu\nu}} KL\left(\boldsymbol{T}\middle\|e^{-\frac{\boldsymbol{C}}{\epsilon}}\right) = \min_{\boldsymbol{T}\in\Pi_{\mu\nu}} KL\left(\boldsymbol{T}\|\boldsymbol{K}\right). \tag{1.20}$$

This gives an elegant view of this regularized version.

Note that the Sinkhorn algorithm proposed by (Cuturi, 2013) is used in many Machine Learning applications (Solomon et al., 2015; Frogner et al., 2015; Courty et al., 2017b; Tolstikhin et al., 2017; Genevay et al., 2018; Caron et al., 2020) as a smooth and differential way to align distributions with a relatively fast and simple solver.

1.1.3 Gromov Wasserstein

This section describes the Gromov Wasserstein (GW) problem (Memoli, 2007; Mémoli, 2011) The main motivation is to overcome some limitations of the original OT setting; (i) it is very complex to define a meaningful cost function c when the two distributions are not in the same space; (ii) the original OT formulation cannot handle complex structures of data such as graphs. The idea is to describe the two distributions with two continuous functions $\mathcal{C}^{\mathcal{X}} : (\mathcal{X}, \mathcal{X}) \to \mathbb{R}$ and $\mathcal{C}^{\mathcal{Y}} : (\mathcal{Y}, \mathcal{Y}) \to \mathbb{R}$. Instead of comparing two points as done in the original OT formulation, two distances, $\mathcal{C}^{\mathcal{X}}(\boldsymbol{x}, \boldsymbol{x}')$ and $\mathcal{C}^{\mathcal{Y}}(\boldsymbol{y}, \boldsymbol{y}')$, are now compared with a loss function $\mathcal{L}$. Similarly, instead of matching the point $\boldsymbol{x}$ to $\boldsymbol{y}$, both $\boldsymbol{x}$ and $\boldsymbol{x}'$ are now simultaneously matched to $\boldsymbol{y}$ and $\boldsymbol{y}'$. Thus, with this matching, the pairwise distances $\mathcal{C}^{\mathcal{X}}(\boldsymbol{x}, \boldsymbol{x}')$ and $\mathcal{C}^{\mathcal{Y}}(\boldsymbol{y}, \boldsymbol{y}')$ should be close. This leads to the following quadratic problem,

$$GW\left(\mathcal{C}^{\mathcal{X}}, \mathcal{C}^{\mathcal{Y}}, \mu, \nu\right) = \min_{\pi\in\Pi_{\mu\nu}} \int_{\mathcal{X}\times\mathcal{Y}} \int_{\mathcal{X}\times\mathcal{Y}} \mathcal{L}\left(\mathcal{C}^{\mathcal{X}}(\boldsymbol{x}, \boldsymbol{x}'), \mathcal{C}^{\mathcal{Y}}(\boldsymbol{y}, \boldsymbol{y}')\right) \pi(\boldsymbol{x}, \boldsymbol{y})\pi(\boldsymbol{x}', \boldsymbol{y}'). \tag{1.21}$$

The Gromov Wasserstein function is an isometry between metric measure spaces (Memoli, 2007; Burago et al., 2001) as long as $\mathcal{L}$ is a distance. Basically, as both spaces are included in vector spaces, it is a distance which is invariant to rotation and translation of the distributions: if $GW(\mu, \nu) = 0$ it only means that the two distributions are equal up to a rotation and translation. Interestingly, the GW problem is also a distance between weighted graphs (Chowdhury and Mémoli, 2019), only requiring the pairwise matrices $\boldsymbol{C}^{\mathcal{X}}$ and $\boldsymbol{C}^{\mathcal{Y}}$. This result seems natural as a graph can be represented with an adjacency matrix and thus is already invariant to rotation and translation. However, some work is necessary to transform the graphs into a canonical form to have an actual distance (Chowdhury and Mémoli, 2019). This part is further discussed in Chapter 5 as similar problems arise in our contributions. Similarly to the p-Wasserstein distance, one can define the p-GW distance by taking the p-root of Equation (1.21) with $\mathcal{L}^p$. Similarly to

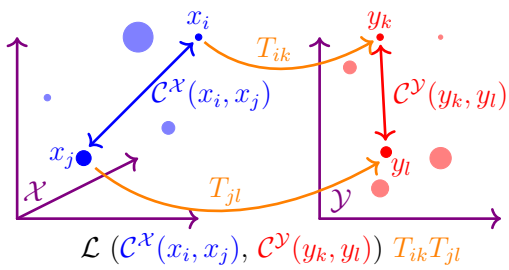


Figure 1.6: *Illustration of GW, with only one term* $\boldsymbol{L}_{ijkl}$ *of the quadruple sum of Eq.* (1.22). *Cited on page [19]*

the OT case, if $\mathcal{L}$ is a distance the GW function is also a distance. By abuse of notation, we will often use throughout this manuscript the term of Gromov Wasserstein *distance* with any $\mathcal{L}$, even if all the properties of an actual metric do not always hold.

Since we will mainly work with discrete distributions (or finite weighted graphs), the discrete formulation of the GW problem reads,

$$GW\left(\mathcal{C}^{\mathcal{X}}, \mathcal{C}^{\mathcal{Y}}, \mu, \nu\right) = \min_{\boldsymbol{T} \in \Pi_{\mu\nu}} \mathcal{E}(\boldsymbol{T}, \boldsymbol{T}), \tag{1.22}$$

with

$$\mathcal{E}(\boldsymbol{T}, \boldsymbol{T}) = \sum_{i,j=1}^{I,I} \sum_{k,l=1}^{K,K} \mathcal{L}(\mathcal{C}^{\mathcal{X}}(\boldsymbol{x}_i, \boldsymbol{x}_j), \mathcal{C}^{\mathcal{Y}}(\boldsymbol{y}_k, \boldsymbol{y}_l))\boldsymbol{T}_{ik}\boldsymbol{T}_{jl} \tag{1.23}$$

$$= \sum_{i,j=1}^{I,I} \sum_{k,l=1}^{K,K} \boldsymbol{L}_{ijkl}\boldsymbol{T}_{ik}\boldsymbol{T}_{jl}. \tag{1.24}$$

The double matching of points ($\boldsymbol{x}_i \to \boldsymbol{y}_k$, $\boldsymbol{x}_j \to \boldsymbol{y}_l$), or equivalently the matching of pairs $((\boldsymbol{x}_i\boldsymbol{x}_k) \to (\boldsymbol{y}_k, \boldsymbol{y}_l))$, is illustrated in Figure 1.6 for fixed indices i, k, j and l.

The most used solver for GW has been proposed by (Peyré et al., 2016) (EGW). It adapts the algorithm of (Rangarajan et al., 1999) to non-uniform marginals. The latter was originally designed for soft-quadratic assignment, which is a particular case of the GW problem. Interestingly, the Sinkhorn algorithm was already used by Rangarajan et al. (1999) at each iteration again with uniform marginals. As the GW problem is non-linear and is neither convex or concave in the general case, the problem is hard to solve. In fact, GW is a generalization of the Quadratic Assignment Problem which is NP-hard (Loiola et al., 2007). Thus, it is worth noting that the proposed methods in this manuscript will only aim to *approximate* the best the GW distance.

The algorithms used to solve the GW problem will be discussed in Section 1.3 and in Chapter 4. We will notably propose two interpretations of EGW, either as a Frank-Wolfe (FW) algorithm (Frank et al., 1956) with an entropy regularized solver for the linear part or as a Mirror Descent algorithm (Beck and Teboulle, 2003).

The GW distance has several applications in machine learning: due to its invariance to rotation and translation, it is a relevant tool for matching and partitioning tasks involving

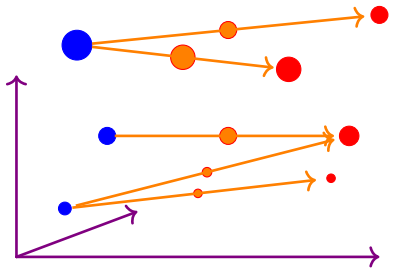


Figure 1.7: *Illustration of a Wasserstein Barycenter (orange) between two discrete distributions.* *Cited on page [20]*

graphs (Xu et al., 2019a,b; Vayer et al., 2019a), by allowing *e.g.* to encode some structure like the shortest path between two vertices. GW has been further used in various other domains, such as Heterogeneous Domain Adaptation (Yan et al., 2018), Shape Matching (Mémoli, 2011; Bronstein et al., 2010; Vayer et al., 2019b), Object Modeling with Deep Learning (Ezuz et al., 2017), Generative Adversarial Networks (Bunne et al., 2019). The Wasserstein distance and the GW distance have also been jointly used in (Vayer et al., 2018) leading to the so-called Fused-Gromov Wasserstein distance that will be introduced in Equation 1.39.

1.1.4 OT barycenter

In this section, we define the notion of barycenter for both the original OT and GW formulations. The barycenter is the extension of the notion of mean. The usual mean $\bar{\boldsymbol{x}} \in \mathcal{X}$ of points $(\boldsymbol{x}_i)_{i\in[\![1,I]\!]} \in \mathcal{X}^I$ with weights $\boldsymbol{a} \in \Delta_I$ can be defined as,

$$\bar{\boldsymbol{x}} = \underset{\boldsymbol{x}\in\mathcal{X}}{\operatorname{argmin}} \sum_i \boldsymbol{a}_i \left\|\boldsymbol{x} - \boldsymbol{x}_i\right\|_2^2 . \tag{1.25}$$

The idea of the barycenter is to replace the usual Euclidean distance by any other distance. As both the OT and GW problems define a distance between probability distributions, one can naturally define the barycenter of such probability distributions. An example of a continuous Wasserstein Barycenter has been already presented in Figure 1.11. The discrete case is illustrated in Figure 1.7. On the latter, the points supporting the Wasserstein Barycenter between the two discrete distributions both weighted by 0.5 are simply located in the middle of each orange arrow. Moreover, the weight associated to each of those points is simply equal to the mass transported by each arrow. The Wasserstein Barycenter generalizes the notion of barycenter between points. Let $(\mu_b)_{b\in[\![1,B]\!]}$ be B discrete distributions in $\mathcal{X}$ with I_B points each. The barycenter of these distributions each associated with weight $(\lambda_b \in \Delta_{I_B})_{b\in[\![1,B]\!]}$ is defined as:

$$WB((\mu_b)_{b\in[\![1,B]\!]}) = \min_{\mu\in\mathcal{P}(\mathcal{X})} \sum_{b=1}^{B} \lambda_b W(\mu, \mu_b). \tag{1.26}$$

A closed-form solution of such a problem is available for Gaussian distributions (Dowson and Landau, 1982), but in general the problem is complex. Under some conditions, the unicity of the solution can be proven (Agueh and Carlier, 2011). Note that many discrete applications

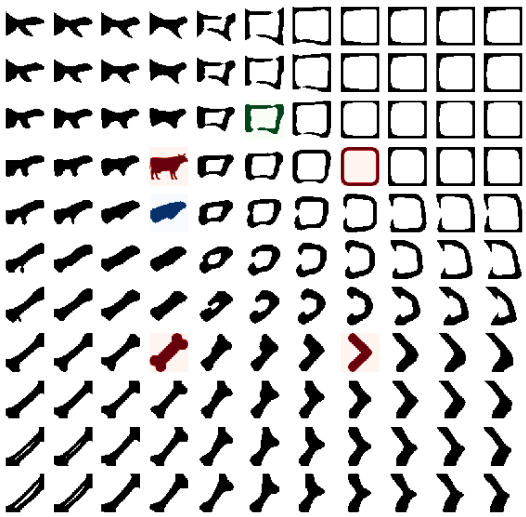


Figure 1.8: *Illustration of a Wasserstein Barycenter (Solomon et al., 2015) with potentially negative weight between four 2 dimensional discrete distributions based on images. The two axis are independent from each other. For instance, the blue image is associated with a weight of* 0.75 *for the cow,* 0.25 *for the bone and 0 for the two other images. The green image is weighted by* 0.625 *for the cow and the square and* -0.125 *for the two others, the cow and square share* 1.25 *of the mass while the bone and the arrow share* -0.25.

Cited on page [20]

use some additional constraints on μ. For example, the support of the Diracs of μ can be fixed (Cuturi and Doucet, 2014), or the number of Diracs limited (Rabin and Peyré, 2011; Cuturi and Doucet, 2014). Chapter 2 will discuss the Wasserstein Barycenter restricted to a smaller subspace. Figure 1.8 gives an example of a Wasserstein Barycenter between four 2 dimensional discrete distributions represented in red (Solomon et al., 2015). This figure extrapolates the barycenter to value of λ_b potentially negative. The Gromov Wasserstein Barycenter (GWB) can be defined in a similar manner, but as the distributions $(\mu_b)_{b\in[\![1,B]\!]}$ are now in potentially different spaces $(\mathcal{X}_b)_{b\in[\![1,B]\!]}$ the space of μ can be chosen freely. Generally, this boils down to choosing the size of the vector space. The GWB problem reads,

$$GWB((\mu_b)_{b\in[\![1,B]\!]}) = \min_{\mu\in P(\mathcal{X})} \sum_{b=1}^{B} \lambda_b GW(\mu, \mu_b). \tag{1.27}$$

Note that as the GW problem is invariant to translation and rotation, there is an infinite number of solutions.

1.2 Machine Learning

In this section, we provide some background knowledge in Machine Learning (ML) and Domain Adaptation. Our contributions to the OT theory will be exploited in the experiments of this thesis to address various ML problems.

1.2.1 Machine Learning setting

Let $\mathcal{X}$ be an input space included in a vector space and $\mathcal{Y}$ the space of labels generally defined as $\{0,1\}^C$ for $C \in \mathbb{N}$ classes. Let us define a distribution $\mu_{\mathcal{Z}} \in \mathcal{P}(\mathcal{Z})$ over the joint space $\mathcal{Z} = \mathcal{X} \times \mathcal{Y}$. The goal of Machine Learning is to find a function $h : \mathcal{X} \to \mathbb{R}^C$, in a set of functions $\mathcal{H}$, which predicts $\boldsymbol{y}$ given $\boldsymbol{x}$ where $(\boldsymbol{x}, \boldsymbol{y}) \sim \mu_{\mathcal{Z}}$. Note that the function h is not restricted to $\mathcal{Y}$ but can have its output in $\mathbb{R}^C$. Even if, for a specific points $\boldsymbol{x}$, the expected class is 1 or equivalently $(0, 1, 0, ..., 0)$, the function h can give a soft prediction such as $(0.1, 0.8, 0.1, 0, ..., 0)$.

To quantify if a function h predicts well the label, a cost function $c : (\mathcal{H}, \mathcal{Z}) \to \mathbb{R}$ is defined so as to measure the disagreement between the prediction and the true label. The goal is then to minimize the so-called true risk (or generalization risk),

$$\min_{h \in \mathcal{H}} \mathcal{R}(h) = \min_{h \in \mathcal{H}} \mathop{\mathbb{E}}_{\boldsymbol{x}, \boldsymbol{y} \sim \mu_{\mathcal{Z}}} \left[c(h, (\boldsymbol{x}, \boldsymbol{y})) \right]. \tag{1.28}$$

The most natural cost function is the so-called $0-1$ loss which basically counts the number of misclassifications made by the function h. Since this function is not convex and not smooth, one usually resorts to surrogates such as the exponential loss, the hinge loss or the sigmoïd loss, to cite a few.

Since $\mu_{\mathcal{Z}}$ is usually unknown, h is learned from a finite training set S of samples supposedly drawn independently from $\mu_{\mathcal{Z}}$. The goal is then to minimize the following empirical risk,

$$\min_{h \in \mathcal{H}} \hat{\mathcal{R}}(h) = \min_{h \in \mathcal{H}} \mathop{\mathbb{E}}_{\boldsymbol{x}, \boldsymbol{y} \sim \hat{\mu}^S_{\mathcal{Z}}} \left[c(h, (\boldsymbol{x}, \boldsymbol{y})) \right]. \tag{1.29}$$

Note that h is often parametrized by some parameters denoted as θ. Problem 1.29, can be rewritten as follows:

$$\min_{\theta} \mathop{\mathbb{E}}_{\boldsymbol{x}, \boldsymbol{y} \sim \hat{\mu}^S_{\mathcal{Z}}} \left[c(h_\theta, (\boldsymbol{x}, \boldsymbol{y})) \right]. \tag{1.30}$$

As the goal is to generalize well on unseen data, some regularizations are often used in addition to the expectation of Equation (1.30) to avoid an over-fitting on the training data.

A very simple non-parametric ML algorithm, that we will used often in this manuscript, is the 1-Nearest Neighbors (Cover and Hart, 1967). Given a point $(\boldsymbol{x}, \boldsymbol{y}) \sim \mu$, the predicted class of $h(\boldsymbol{x})$ is simply the class of the closest point in the training set. This classification algorithm thus relies only on the distance between points in the vector space.

1.2.2 Domain Adaptation

In this section, we present a particular ML task, called Domain Adaptation (DA), which is a sub-field of Transfer Learning (Torrey and Shavlik, 2010). The main idea is to use labeled data of a source domain to improve the performance of a classifier deployed on a related target domain which suffers from a lack of labeled examples. In this manuscript, we address the most complex setting, called unsupervised DA, where there is only unlabeled data available from the target distribution. The difference between the source (μ_s) and the target distributions (μ_t)

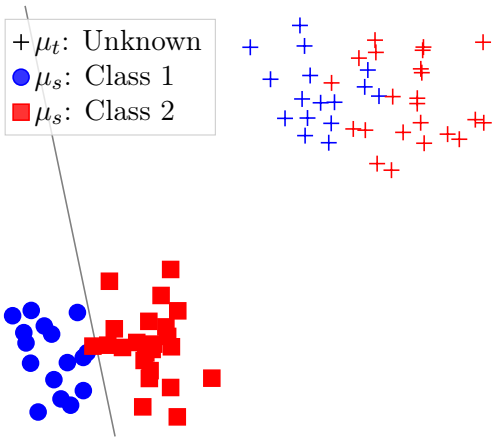


Figure 1.9: *Illustration of a DA task with samples of the source distribution μ_s composed of one Gaussian for each class and samples of the target distributions μ_t linearly shifted from the source distributions. The linear separator is a SVM (Cortes and Vapnik, 1995).*

Cited on page [22]

should be small enough to allow the adaptation. An example of simple shift between the two distributions is illustrated in Figure 1.9.

In this setting, a classifier learned on a source dataset cannot correctly classify the target points. As the final goal is to deploy the model on the target distribution, most of the DA algorithms rely on a two steps procedure (Fernando et al., 2013; Aljundi et al., 2015; Sun et al., 2016a; Pan et al., 2011; Courty et al., 2017b). First, reduce the divergence between the source and target distributions. Then learn a classifier on the source data and predict the class of the target examples. Other methods simultaneously reduce the distance between the distributions and predict the label (Ganin et al., 2016; Sun and Saenko, 2016; Wang et al., 2018; Courty et al., 2017a; Bhushan Damodaran et al., 2018). In this case, a Neural Network (NN) h is often used. The NN is split into 2 parts, h' and h_{pred}. The output space of h' is interpreted as a feature space where the source and target distributions should be close according to a given divergence or distance function. On the other hand, h_{pred} aims at predicting correctly the class of the source labels, preventing the global NN from projecting every point on 0. Indeed, doing so will lead to a divergence of 0 between the two distributions but with a poor classification behavior.

Choosing the hyperparameters of an algorithm is really complex in unsupervised DA as there is no labeled points in the target space. Those hyperparameters can be the architecture of the NN, the number of iterations or the regularization parameters, etc... The main approach to address this task is based on a reverse validation (Zhong et al., 2010). The idea is to predict the label of the target points, called pseudo-labels, then apply again an adaptation from the target to the source using the created pseudo-labels. Using this second adaptation, the label of the source points are predicted and this gives an accuracy score associated with the set of hyperparameters.

In this thesis, we will used the OT theory to reduce the divergence between the two

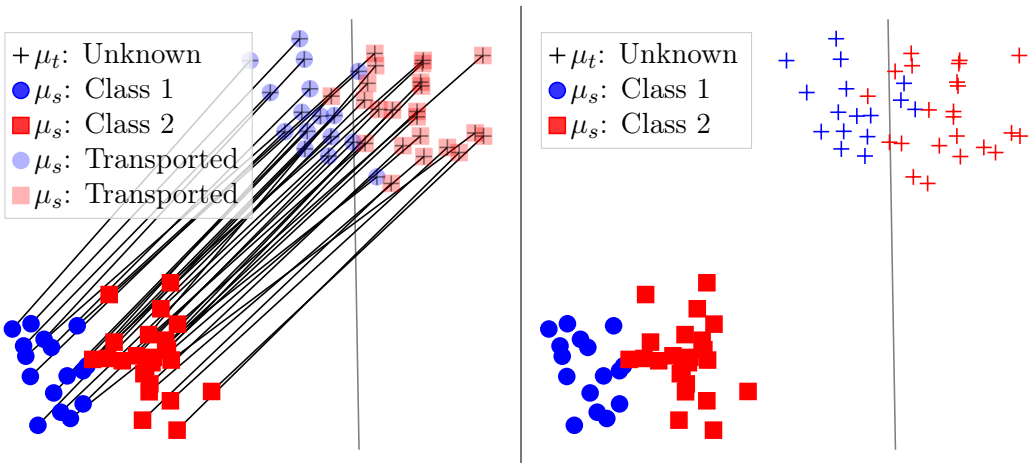


Figure 1.10: *Illustration of a DA task solved with OT. In the left figure, the distribution μ_s is transported to μ_t using the transport plan, then a linear separator is learned. In the right figure, the linear separator has a good prediction on the target points.*

Cited on pages [23,25]

distributions μ_s and μ_t. An illustration of such an adaptation is reported in Figure 1.10. Chapter 2 will be dedicated to address the DA problem by learning, from the dataset, a specific ground cost required for the OT problem.

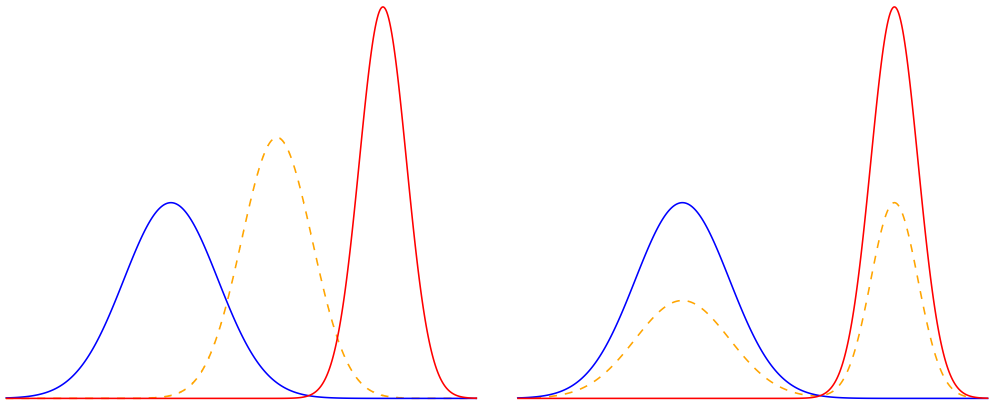
1.2.3 OT-based distances versus Kullback-Leiber divergence in Machine Learning

In this section, we discuss the advantages of the distance induced by solving the OT problem over the Kullback-Leiber divergence (KL) in some Machine Learning applications. A common way to compare probability distributions is to use the KL divergence defined in Equation (1.9) for the particular discrete case. For two continuous probability distributions $\mu \in \mathcal{P}(\mathcal{X})$ and $\nu \in \mathcal{P}(\mathcal{X})$ and their associated probability density functions f_μ and f_ν, where μ is absolutely continuous with respect to ν, the KL divergence reads,

$$KL(\mu||\nu) = \int_{\mathcal{X}} f_\mu(\boldsymbol{x}) log\left(\frac{f_{\mu(\boldsymbol{x})}}{f_{\nu(\boldsymbol{x})}}\right) d\boldsymbol{x}. \tag{1.31}$$

While being commonly used in ML, this divergence has some drawbacks. First, $KL(\mu||\nu)$ is equal to $+\infty$ if the support of ν is not included in the support of μ. This often occurs with discrete distributions. For instance, the KL divergence between two empirical distributions $\hat{\mu}_1$ and $\hat{\mu}_2$ sampled from the same gaussian distribution μ is equal to $+\infty$ almost surely. On the other hand, the Wasserstein distance relies on a ground distance between points, seen as a hyperparameter. When this latter is the Euclidean cost, the Wasserstein distance between two Diracs at position 0 and 1 is equal to 1, while the KL is infinite.

Another difference between the KL divergence and the Wasserstein distance is illustrated in Figure 1.11. The interpolation based on the KL (figure on the right) leads to a distribution with two modes while both μ and ν are unimodal Gaussian distributions. On the other hand,



(a) Wasserstein distance (b) Kullback-Leiber divergence

Figure 1.11: *Interpolation (orange with dash) between two Gaussian's distributions (blue and red) with either the Wasserstein distance or the KL divergence.*

Cited on pages [20,24]

the interpolation based on the Wasserstein distance preservers the geometry of the original distributions by smoothly sliding along the axis when changing the weights of the interpolations.

Because the KL divergence does not handle well the case where the two supports are distinct and because the Wasserstein distance preserves the geometry of the distributions, this latter can advantageously replace the KL distance in some ML applications. The most famous case is the Wasserstein Generative Adversarial Network (WGAN)(Goodfellow et al., 2014; Arjovsky et al., 2017) where the use of the Wasserstein distance to compare the real and the generated distributions allows to have a non vanishing gradient during the gradient descent (more details about the WGAN is available in Appendix E.3). The KL used in the Variational Autoencoder (Kingma and Welling, 2013) to generate the latent space can also be replaced with a Wasserstein distance (Tolstikhin et al., 2017). This latter can also be used as an alternative to the KL divergence as a loss for multi label prediction (Frogner et al., 2015).

Another key advantage of OT compared to the other standard distances between distributions is that the OT plan that allows to align the points can be advantageously exploited. This is the case in DA, as already illustrated in Figure 1.10. Since the Wasserstein distance presents many advantages compared to other distances, a lot of effort has been made during the past few years to overcome the problem related to the computational cost of this distance (Cuturi, 2013; Rabin et al., 2014; Altschuler et al., 2017; Schmitzer, 2019; Peyré et al., 2016).

1.3 First order optimization tools for OT problems

This section briefly presents two methods, the Frank-Wolfe algorithm and the projected Mirror Descent algorithm, that can be used to solve both the Gromov Wasserstein problem and the Optimal Tensor Transport problem (see Chapter 5). For both algorithms, the goal is to minimize

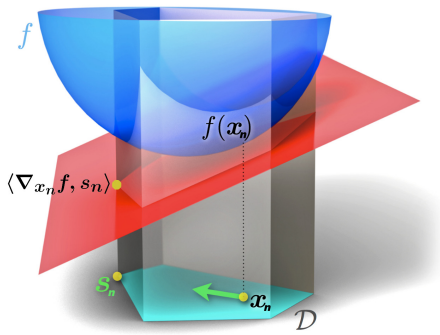


Figure 1.12: *Illustration of an iteration of the FW algorithm in the convex setting. Credit to Stephanie Stutz and Martin Jaggi (Jaggi, 2013).*

Cited on page [26]

a function under constraints.

1.3.1 Frank-Wolfe algorithm

The Frank-Wolfe (FW) algorithm, or Conditional Gradient, has been proposed by Marguerite Frank and Philip Wolfe in 1956 (Frank et al., 1956), see also (Kerdreux et al., 2019) for a recent review. The global goal is to find the minimum of a function f on a convex compact set. When f is a complex function, neither convex or concave, one option is to use a gradient descent scheme, requiring a projection step onto the convex set as the optimization is constrained. Another option is to decompose the problem into several easier subproblems that will allow to get closer to the real solution at each iteration. For instance, the problem can be reduced to a quadratic convex problem that is often easier to solve. The FW algorithm transforms the problem into a linear one by relying on a local linear expansion. Through this section, we will see why this formulation is particularly suitable in the Optimal Transport field.

General settings

Let $\mathcal{D}$ be a convex compact set included in a vector space $\mathcal{X}$. This set $\mathcal{D}$ can be considered as a finite convex polytope in the OT setting as it corresponds to the set of all transport plans. One seeks to minimize a real differentiable function $f : \mathcal{X} \longrightarrow \mathbb{R}$ but only in the set $\mathcal{D}$,

$$\min_{\boldsymbol{x} \in \mathcal{D}} f(\boldsymbol{x}). \tag{1.32}$$

The FW algorithm is given in Algorithm 2: it starts from a point $\boldsymbol{x}_0$ then loops over N iterations. At each iteration n, the gradient of f at the current point $\boldsymbol{x}_n$ is computed. This gradient gives the red hyperplane of Figure 1.12 and its minimization gives the point $\boldsymbol{s}_n$ in $\mathcal{D}$. Then, the current point is partially updated by using a mean between the current point and the minimizer of the hyperplane weighted by α_n. The algorithm can stop after a certain number of

Algorithm 2 Frank-Wolfe algorithm Cited on pages [26,28,28]

Input: f (function), $\mathcal{D}$ (polytope), $(\alpha_n)_{n\in[\![0,N-1]\!]}$ (learning step), $\boldsymbol{x}_0$ (starting point)

1: **for** $n = 0$ **to** $N-1$ **do**
2: $\quad \boldsymbol{s}_n = \underset{\boldsymbol{s}\in\mathcal{D}}{\operatorname{argmin}} \langle \nabla_{\boldsymbol{x}_n} f, \boldsymbol{s}\rangle$
3: $\quad \boldsymbol{x}_{n+1} = (1-\alpha_n)\boldsymbol{x}_n + \alpha_n \boldsymbol{s}_n$
4: **end for**
5: **return** $\boldsymbol{x}_N$

iterations N or a stopping criterion can be used by looking at the FW gap at the iteration n,

$$G(\boldsymbol{x}_n) = \max_{\boldsymbol{s}\in\mathcal{D}} \langle \nabla_{\boldsymbol{x}_n} f, \boldsymbol{x}_n - \boldsymbol{s}\rangle = \langle \nabla_{\boldsymbol{x}_n} f, \boldsymbol{x}_n - \boldsymbol{s}_n\rangle . \tag{1.33}$$

This gap in often called the duality gap as it corresponds to the gap between the primal and dual problem. When f is convex, this duality gap at the iteration n is an upper bound of the distance to the optimal value $f(\boldsymbol{x}^*)$. Indeed, using the convexity of f, for any $\boldsymbol{s}\in\mathcal{D}$,

$$f(\boldsymbol{s}) \geq f(\boldsymbol{x}_n) + \langle \nabla_{\boldsymbol{x}_n} f, \boldsymbol{s} - \boldsymbol{x}_n\rangle \Rightarrow f(\boldsymbol{x}^*) \geq f(\boldsymbol{x}_n) + \min_{\boldsymbol{s}\in\mathcal{D}} \langle \nabla_{\boldsymbol{x}_n} f, \boldsymbol{s} - \boldsymbol{x}_n\rangle \tag{1.34}$$

$$\Rightarrow f(\boldsymbol{x}^*) \geq f(\boldsymbol{x}_n) + \langle \nabla_{\boldsymbol{x}_n} f, \boldsymbol{s}_n - \boldsymbol{x}_n\rangle \tag{1.35}$$

$$\Rightarrow G(\boldsymbol{x}_n) \geq f(\boldsymbol{x}_n) - f(\boldsymbol{x}^*). \tag{1.36}$$

This gives a computable stopping criterion: if at the iteration n, the FW gap is smaller than a threshold, this means that the value $\boldsymbol{x}_n$ is close to the global minimum. This is also a justification for the method, as the FW algorithm minimizes this FW gap at each iteration.

Let us now focus on the general case where f is non necessarily convex. For constrained problems, one cannot rely on the norm of the gradient to find local minima as these latters could be on the edge on the convex set $\mathcal{D}$ without a gradient norm equal to zero. To generalize the notion of local minima to constraint problems, the FW gap can be used instead as a criterion to find local minima (Lacoste-Julien, 2016). A point $\boldsymbol{x}^*$ is a stationary point (either maximum or minimum) of Equation (1.32) if and only if $G(\boldsymbol{x}^*) = 0$ or equivalently if

$$\langle \nabla_{\boldsymbol{x}^*} f, \boldsymbol{x}^*\rangle - \min_{\boldsymbol{s}\in\mathcal{D}} \langle \nabla_{\boldsymbol{x}^*} f, \boldsymbol{s}\rangle = 0. \tag{1.37}$$

Intuitively, if $\boldsymbol{x}_n$, at an iteration n, is a stationary point, the FW algorithm cannot find a better value.

The partial update weighted by $(\alpha_n)_{n\in[\![0,N-1]\!]}$ has two interests for the optimization. First, it allows to converge "slowly" to the desired point without changing completely from one iteration to another. Indeed, the convergence proof, in most cases, requires such a partial update. Second, as the problem solved at each iteration is linear, there is always an optimal solution on the edge on the set $\mathcal{D}$. Thus without a partial update, the algorithm may not find the optimal value if the solution is not on the edge of $\mathcal{D}$. The weights $(\alpha_n)_{n\in[\![0,N-1]\!]}$ associated with the partial updates are hyperparameters that can be chosen at hand, but there are several ways to find

good values. The default choice is to set, for all $n \in [\![0, N-1]\!]$, $\alpha_n = \frac{2}{n+2}$ (Freund and Grigas, 2016). Or, as we will see in Chapter 4, some theoretical bound derivations can provide values for $(\alpha_n)_{n\in[\![0,N-1]\!]}$. However, such values are often too conservative and thus too small in practice, except when the function f yields a concave problem, then the optimal value of α is 1. This is an interesting case as the convergence rate of the Frank Wolfe gap becomes often linear in the number of iterations N instead of depending on the square root of N. The last method optimally finds the best value of α_n at each iteration n by minimizing the following equation,

$$\min_{\alpha\in[0,1]} f\left((1-\alpha)\boldsymbol{x}_n + \alpha \boldsymbol{s}_n\right). \tag{1.38}$$

This line-search of the best value of α can be applied efficiently for some functions f.

To end this section, let us briefly present the stochastic FW algorithm (Lacoste-Julien, 2016), which will be used in Chapter 4. The idea is to approximate the gradient at line 2 in Algorithm 2 with an unbiased estimate. This is possible when the gradient is an expectation of some random variable. The stochastic FW algorithm simply samples one or a few times the random variable instead of computing the expectation entirely. Similarly to stochastic gradient descent, there is two main advantages. First, it reduces the complexity of computing the gradient at each iteration. Second, the stochasticity allows more exploration and might avoid being stuck in a local minimum. We will now discuss why the FW algorithm is interesting in the OT setting.

Relation with Optimal Transport

First of all, the FW algorithm is not useful for the original OT problem as this latter is already linear but it can be used on a barycentric OT problem (Luise et al., 2019), on some minimax OT problem (Paty and Cuturi, 2019a) or for some DA variants (Courty et al., 2017b). It has more applications on the Gromov Wasserstein problem GW is naturally non-linear. The FW algorithm was notably proposed to solve the Fused-Gromov Wasserstein (FGW) problem (Vayer et al., 2018, 2019a). The idea of FGW is to simultaneously minimize a GW term and a classical OT term. It is an interesting tool to compare, align or compute the barycenter of labeled graphs.

With the previous notations and considering $\beta \in [0,1]$, the FGW problem is defined as follows:

$$\min_{\boldsymbol{T}\in\Pi_{\mu\nu}} (1-\beta)\sum_{i,k=1}^{I,K} \mathcal{C}(\boldsymbol{x}_i, \boldsymbol{y}_k)\boldsymbol{T}_{ik} + \beta \sum_{i,j=1}^{I,I}\sum_{k,l=1}^{K,K} \mathcal{L}\left(\mathcal{C}^{\mathcal{X}}(\boldsymbol{x}_i,\boldsymbol{x}_j), \mathcal{C}^{\mathcal{Y}}(\boldsymbol{y}_k,\boldsymbol{x}_l)\right)\boldsymbol{T}_{ik}\boldsymbol{T}_{jl}. \tag{1.39}$$

Vayer et al. (2019a) used a FW algorithm with a line-search to find the best value of α_n at each iteration n. At each iteration of the FW algorithm for FGW, denoting by $\boldsymbol{T}^n$ the last transport plan found, the line 2 of Algorithm 2 is an OT problem,

$$\tilde{\boldsymbol{T}}^{n+1} = \operatorname*{argmin}_{\boldsymbol{T}\in\Pi_{\mu\nu}} \left\langle (1-\beta)\boldsymbol{C} + 2\beta \sum_{i,k=1}^{I,K} \boldsymbol{L}_{i\bullet k\bullet}\boldsymbol{T}^n_{ik}, \boldsymbol{T}\right\rangle. \tag{1.40}$$

This gradient expression is correct only if $\mathcal{C}^{\mathcal{X}}$ and $\mathcal{C}^{\mathcal{Y}}$ are symmetric, the general gradient is given in Equation C.13.

The most used algorithm to solve GW is called Entropy Gromov Wasserstein (EGW) (Peyré et al., 2016). It is also highly related to the FW algorithm. The original interpretation of EGW is given in Section 1.3.2. In EGW, there is no partial update with the step α or equivalently, α is always set to 1. This is not a huge problem in most of the applications as the GW problem is often concave. For instance, if $\mathcal{C}^{\mathcal{X}}$, $\mathcal{C}^{\mathcal{Y}}$ and $\mathcal{L}$ are the squared Euclidean distance functions in their respective spaces, then $\boldsymbol{L}$ yields a concave problem (Redko et al., 2020). Also, to speed up the computation, instead of solving perfectly the OT problem at each iteration, EGW solves the entropy regularized OT,

$$\boldsymbol{T}^{n+1} = \underset{\boldsymbol{T}\in\Pi_{\mu\nu}}{\operatorname{argmin}} \left\langle 2 \sum_{i,k=1}^{I,K} \boldsymbol{L}_{i\bullet k\bullet} \boldsymbol{T}_{ik}^{n}, \boldsymbol{T} \right\rangle - \epsilon \mathcal{H}(\boldsymbol{T}). \tag{1.41}$$

As ϵ is often chosen very small, this additional entropy term will only slightly modify the next transport plan $\boldsymbol{T}^{n+1}$. Thus EGW can be interpreted as a variant of a FW algorithm. The "2" before the double sum can be omitted as it is equivalent to rescaling ϵ. We will see that related convergence bounds to stationary points are also given in (Vayer, 2020), but not for the regularized problem, and more importantly not for the stochastic setting that will be used in Chapter 4. Indeed the stochastic FW algorithm is very interesting for solving the GW problem in comparison to the classical FW algorithm as the gradient can be interpreted as an expectation.

There are other applications of the FW algorithm in OT-related problems, notably OTDA (Courty et al., 2017b) that will be further discussed in Chapter 2. We will now present the projected Mirror Descent algorithm: while the idea is completely different from the FW algorithm, both methods are closely related in the OT context.

1.3.2 Projected Mirror Descent

As explained in Section 1.3.1, one can search for the minimum of the function f by applying a gradient descent scheme. The projected Mirror Descent (MD) algorithm (Nemirovskij and Yudin, 1983; Beck and Teboulle, 2003) generalizes the classical gradient descent method by changing the default Euclidean prior to any Bregman divergence (Bregman, 1967). We first introduce the general Gradient Descent (GD) method then generalize its formulation and explain how it can be applied in the OT setting.

Gradient Descent

One possible interpretation of GD is to suppose that the problem is locally linear and thus approximate the function f by the tangent at a point $\boldsymbol{x}_n$ and search for the next best point $\boldsymbol{x}_{n+1}$. To ensure that the search is done only locally, a squared Euclidean distance regularization is added, weighted by $\alpha > 0$. The GD algorithm starts from a point $\boldsymbol{x}_0$ and then applies the following step for all $n \in \mathbb{N}$,

$$\boldsymbol{x}_{n+1} = \underset{\boldsymbol{x}\in\mathcal{X}}{\operatorname{argmin}} \left\{ f(\boldsymbol{x}_n) + \langle \nabla_{\boldsymbol{x}_n} f, \boldsymbol{x} - \boldsymbol{x}_n \rangle + \frac{1}{2\alpha} \|\boldsymbol{x} - \boldsymbol{x}_n\|_2^2 \right\}. \tag{1.42}$$

By searching where the gradient is equal to 0, the classical GD formula is recovered,

$$0 = \nabla_{\boldsymbol{x}_n} f + \frac{1}{\alpha}(\boldsymbol{x}_{n+1} - \boldsymbol{x}_n) \Longrightarrow \boldsymbol{x}_{n+1} = \boldsymbol{x}_n - \alpha \nabla_{\boldsymbol{x}_n} f. \tag{1.43}$$

The interesting part of this interpretation is that the GD implicitly assumes that the Euclidean distance is used to ensure the locality of the minimum search. If the problem is constrained to $\boldsymbol{x} \in \mathcal{D}$, either the problem 1.42 can be minimized directly with the constraint, or, equivalently, the unconstrained problem can be solved to find $\tilde{\boldsymbol{x}}_{n+1}$ which is then projected to $\mathcal{D}$ according to the Euclidean distance,

$$\boldsymbol{x}_{n+1} = \underset{\boldsymbol{x} \in \mathcal{D}}{\operatorname{argmin}} \left\| \tilde{\boldsymbol{x}}_{n+1} - \boldsymbol{x} \right\|. \tag{1.44}$$

Mirror Descent

The MD method simply replaces the squared Euclidean distance with a Bregman divergence (Bregman, 1967). Let $\phi : \mathbb{X} \to \mathcal{R}$ be a strictly convex differentiable function. The Bregman divergence associated with ϕ is defined as,

$$\mathcal{B}_\phi : (\boldsymbol{x}, \boldsymbol{y}) \mapsto \phi(\boldsymbol{x}) + \phi(\boldsymbol{y}) - \langle \nabla_{\boldsymbol{x}} \phi, \boldsymbol{x} - \boldsymbol{y} \rangle. \tag{1.45}$$

The Bregman divergences are particular cases of divergences, thus contrary to distance it should not necessarily respect the symmetry property nor the triangular inequality. The Euclidean distance, the Mahalanobis distance (Mahalanobis, 1936) or the KL divergence are examples of Bregman divergences.

The Equation (1.42) can be rewritten with the Bregman divergence $\mathcal{B}_\phi$ instead of the Euclidean distance,

$$\boldsymbol{x}_{n+1} = \underset{\boldsymbol{x} \in \mathcal{D}}{\operatorname{argmin}} \left\{ f(\boldsymbol{x}_n) + \langle \nabla_{\boldsymbol{x}_n} f, \boldsymbol{x} - \boldsymbol{x}_n \rangle + \frac{1}{\alpha} \mathcal{B}_\phi(\boldsymbol{x}, \boldsymbol{x}_n) \right\}. \tag{1.46}$$

Removing the terms that are independent of $\boldsymbol{x}$, this problem can be equivalently reformulated as

$$\boldsymbol{x}_{n+1} = \underset{\boldsymbol{x} \in \mathcal{D}}{\operatorname{argmin}} \left\{ \langle \nabla_{\boldsymbol{x}_n} f, \boldsymbol{x} \rangle + \frac{1}{\alpha} \mathcal{B}_\phi(\boldsymbol{x}, \boldsymbol{x}_n) \right\}. \tag{1.47}$$

Relation with the Optimal Transport

As the point $\boldsymbol{x}$ corresponds to the transport plan $\boldsymbol{T}$ in the OT setting, the Bregman divergence should be a divergence between probability distributions. Using the KL as Bregman divergence, Xie et al. (2020) interprets the Sinkhorn algorithm (Algorithm 1) as a one step proximal point method. Xie et al. (2020) proposes to apply several times the Sinkhorn algorithm, using a KL divergence with the previous transport plan $\boldsymbol{T}^n$. Because the original OT problem is linear, one has for all $\boldsymbol{T} \in \Pi_{\mu\nu}$, $\langle \nabla_{\boldsymbol{T}} f, \boldsymbol{C} \rangle = \langle \boldsymbol{T}, \boldsymbol{C} \rangle = f(\boldsymbol{T})$, therefore such a proximal point method is equivalent to the MD algorithm. For two transport plans $(\boldsymbol{T}, \boldsymbol{T}') \in \Pi_{\mu\nu}$, the KL divergence is

defined using the entropy $\mathcal{H}$ for ϕ. The KL divergence is a particular case of Bregman divergence as it can be easily reformulated into,

$$\mathcal{B}_{\mathcal{H}}(\boldsymbol{T}, \boldsymbol{T}') = \sum_{i,k=1}^{I,K} \boldsymbol{T}_{ik} log\left(\frac{\boldsymbol{T}_{ik}}{\boldsymbol{T}'_{ik}}\right) - \sum_{i,k=1}^{I,K} \boldsymbol{T}_{ik} + \sum_{i,k=1}^{I,K} \boldsymbol{T}'_{ik}. \tag{1.48}$$

Because $\boldsymbol{T}$ and $\boldsymbol{T}'$ are probability distributions, the last two terms cancel each other and are thus not useful during the optimization. Starting from the uniform transport plan $\boldsymbol{T}^0 = \boldsymbol{a}\boldsymbol{b}^\top$, the MD algorithm (Xie et al., 2020) with $\alpha = \frac{1}{\epsilon}$ applies the following step for all $n \in [\![0, N]\!]$,

$$\boldsymbol{T}^{n+1} = \underset{\boldsymbol{T} \in \Pi_{\mu\nu}}{\operatorname{argmin}} \left\{ \langle \boldsymbol{T}, \boldsymbol{C} \rangle + \epsilon \sum_{i,k=1}^{I,K} \boldsymbol{T}_{ik} log\left(\frac{\boldsymbol{T}_{ik}}{\boldsymbol{T}^n_{ik}}\right) \right\}. \tag{1.49}$$

The problem can be reformulated similarly to what was done in Section 1.1.2,

$$\boldsymbol{T}^{n+1} = \underset{\boldsymbol{T} \in \Pi_{\mu\nu}}{\operatorname{argmin}} \left\{ \langle \boldsymbol{T}, \boldsymbol{C} - \epsilon log\left(\boldsymbol{T}^n\right) \rangle - \epsilon \mathcal{H}(\boldsymbol{T}) \right\}. \tag{1.50}$$

As a consequence, the transport plan of the previous iteration only modifies the cost matrix $\boldsymbol{C}$, the Sinkhorn algorithm can be applied on this new cost matrix. Intuitively, such iterations can converge to the OT plan $\boldsymbol{T}^*$ as the prior $-\epsilon log\left(\boldsymbol{T}^n\right)$ becomes more and more important compared to the entropy as $\boldsymbol{T}^n$ gets closer to the edge of the polytope. In fact, assuming that the Sinkhorn algorithm outputs the exact argmin at each iteration (it would require an infinite number of Sinkhorn iterations, $P = \infty$), Xie et al. (2020) proves a linear convergence to the Optimal Transport plan $\boldsymbol{T}^*$. For the original OT problem, such a formulation has little practical advantage as it requires to apply several times the Sinkhorn algorithm. However it can be used for the other non-linear OT problems.

A similar approach has been proposed for the Gromov Wasserstein problem (Xu et al., 2019b), the only difference comes from the gradient of $\mathcal{E}$ at the point $\boldsymbol{T}^n$. If $\boldsymbol{L}_{\bullet j \bullet l}$ denotes the $I \times K$ matrix extracted from $\boldsymbol{L} \in \mathcal{T}_{I \times I \times K \times K}$ and assumes the symmetry of the function $\mathcal{C}^{\mathcal{X}}$ and $\mathcal{C}^{\mathcal{Y}}$, $\nabla_{\boldsymbol{T}^n} \mathcal{E} = 2 \sum_{i,k=1}^{I,K} \boldsymbol{L}_{i \bullet k \bullet} \boldsymbol{T}^n_{i,k}$. The algorithm proposed by (Xu et al., 2019b) starts with a transport plan matrix $\boldsymbol{T}^0$ then applies the following step for all $n > 0$,

$$\boldsymbol{T}^{n+1} = \underset{\boldsymbol{T} \in \Pi_{\mu\nu}}{\operatorname{argmin}} \left\{ \left\langle \boldsymbol{T}, 2 \sum_{i,k=1}^{I,K} \boldsymbol{L}_{i \bullet k \bullet} \boldsymbol{T}^n_{i,k} - \epsilon log\left(\boldsymbol{T}^n\right) \right\rangle - \epsilon \mathcal{H}(\boldsymbol{T}) \right\}. \tag{1.51}$$

Similarly to the FW algorithm, this projected MD algorithm is interesting because it reduces a non-linear problem to several linear problems that can be solved efficiently with the Sinkhorn algorithm (Cuturi, 2013). Similarly to the FW case, the "2" factor in front of the double sum can be omitted as it is equivalent to a rescaling of ϵ by a factor 2.

Peyré et al. (2016) uses the same algorithm but aims at solving a regularized GW problem defined as,

$$GW\left(\mathcal{C}^{\mathcal{X}}, \mathcal{C}^{\mathcal{Y}}, \mu, \nu\right) = \min_{\boldsymbol{T} \in \Pi_{\mu\nu}} \mathcal{E}(\boldsymbol{T}, \boldsymbol{T}) - \epsilon \mathcal{H}(\boldsymbol{T}), \tag{1.52}$$

As shown in (Peyré et al., 2016), this leads to the same step without the prior $\epsilon log\,(\boldsymbol{T}^n)$,

$$\boldsymbol{T}^{n+1} = \underset{\boldsymbol{T}\in\Pi_{\mu\nu}}{\operatorname{argmin}} \left\{ \left\langle \boldsymbol{T}, 2\sum_{i,k=1}^{I,K} \boldsymbol{L}_{i\bullet k\bullet} \boldsymbol{T}^n_{i,k} \right\rangle - \epsilon\mathcal{H}(\boldsymbol{T}) \right\}. \tag{1.53}$$

Because the value of ϵ is often very small in order to stay close to the original GW problem, the FW algorithm might be a more natural interpretation for this algorithm.

1.4 Conclusion

Most of the necessary background for the manuscript has been explained in this first chapter. The Optimal Transport theory was presented with the Sinkhorn solver and the 1 dimensional Optimal Transport case. Other extensions were detailed such as the OT Barycenter or the Gromov Wasserstein problem. Then, we explained some Machine Learning tasks that can be tackled using the OT theory, notably the Domain Adaptation task. Lastly, we gave some details on the Frank-Wolfe and Mirror Descent algorithms that will be useful to solve some OT extensions. The next chapter focuses on the Domain Adaptation task using a learned ground distance for the Optimal Transport.

Chapter 2

Metric Learning for Optimal Transport

Abstract

This chapter is based on the paper "Metric Learning for Optimal Transport" published at the IJCAI 2020 conference (Kerdoncuff et al., 2020) and tackles the question of learning the ground metric of the Optimal Transport problem to improve Domain Adaptation tasks. We recall that Domain Adaptation aims at benefiting from a labeled dataset drawn from a *source* distribution to learn an efficient model on examples generated according to a different but related *target* distribution. Creating a domain-invariant representation between the two source and target domains is the most widely used technique. A simple and robust way to perform this task consists in (i) representing the two domains by subspaces described by their respective eigenvectors and (ii) seeking a mapping function which aligns these subspaces. In this chapter, we propose to use Optimal Transport (OT) and its associated Wasserstein distance to perform this alignment. While the idea of using OT in domain adaptation is not new, the original contribution of this chapter is two-fold: (i) we derive a generalization bound on the target error involving several Wasserstein distances. This prompts us to optimize the ground metric of OT to reduce the target risk. (ii) From this theoretical analysis, we design an algorithm (MLOT) which optimizes a Mahalanobis distance leading to a transportation plan that adapts better. Experiments demonstrate the effectiveness of this original approach.

2.1 Introduction

Domain adaptation (DA) has been shown to be very effective in many real world applications, *e.g.*, in computer vision, medical diagnosis, or recommender systems, to cite a few. The main idea is to use labeled data of a source domain to improve the performance of a classifier deployed on a related target domain which suffers from a lack of labeled examples. In this chapter, we address a complex setting, called unsupervised DA, where there is only unlabeled data available from the target distribution.

Different approaches have been proposed to tackle this problem, some of them coming with

theoretical guarantees (see, *e.g.* the survey (Redko et al., 2019c)). One classical way is to learn a common latent space in which the shift between the two distributions is smaller. For instance, the Subspace Alignment algorithm (SA) (Fernando et al., 2013) learns a classifier in a subspace obtained after a linear alignment of the source and target eigenspaces. In a similar manner, the Correlation Alignment (CORAL) (Sun et al., 2016a) uses the covariance of the source and target distributions to reduce the shift, while Transfer Component Analysis (TCA) (Pan et al., 2011) looks for common features between the two domains. Some other works directly learn the target labels but do not gather the two distributions in a common feature space. This is the case of MEDA (Wang et al., 2018) which learns a domain-invariant classifier in Grassman manifold. On the other hand, DA with deep learning has received much attention during the past decade from the computer vision community leading to a substantial amount of research to address visual tasks for which a large amount of training data or a pre-trained model is available (see, *e.g.* the survey (Wang and Deng, 2018)).

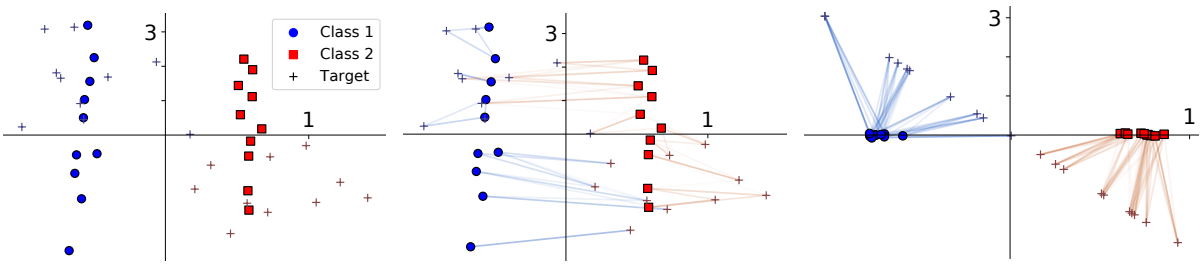


Figure 2.1: *Behavior of MLOT on a toy dataset. On the left, the original source and target examples. In the middle, OTDA fails to transport correctly the blue and red classes. On the right, the proposed MLOT which combines a learned metric and some per-domain dimensionality reduction, leads to a perfect transportation plan. Notice the difference in scale between the two axes.*

Cited on pages [34,42]

More recently, Optimal Transport (OT) has been shown to be a very promising tool to perform DA tasks. We recall that OT consists in mapping two source and target probability measures with a minimal cost of transportation associated to the so-called Wasserstein distance. Beyond its use in deep learning to solve visual DA tasks (see, *e.g.*, (Sun and Saenko, 2016; Bhushan Damodaran et al., 2018)), this idea of reducing the shift by OT has been exploited in a more generic DA setting by the algorithm OTDA (Courty et al., 2017b). OTDA modifies the original Kantorovich optimization problem by resorting to a regularization preventing the transportation plan from moving two source points having different labels onto the same target example. Then, a classifier is learned from the transported labeled source data and deployed over the target distribution. Based on this work, (Courty et al., 2017a) ensures that the final classifier is coherent with the transportation plan.

Inspired from both SA and OTDA, our contribution aims at using OT for domain adaptation by aligning the source and target subspaces. The main conjecture we formulate in this chapter is that the Euclidean distance, usually used, as mentioned in Chapter 1, as the cost matrix in the OT problem, may not be the best metric to perform the adaptation. While learning a better metric (especially a Mahalanobis distance) in OT has been recently studied (Cuturi and Avis, 2014; Genevay et al., 2018; Deshpande et al., 2019; Paty and Cuturi, 2019a), optimizing such a ground metric to address DA tasks has not received attention yet. We fill this gap from both a theoretical and an algorithmic perspective. First, we formally establish a relation between the target error and the magnitude of different Wasserstein distances. This prompts us to see the Wasserstein distance as a parameterized metric that might be minimized, leading to a better transportation plan for DA. We also formally make a link between the Principal Component Analysis (PCA) (Wold et al., 1987) and the minimization of the Wasserstein distance. Based on this theoretical analysis, we propose MLOT, an algorithm which optimizes a Mahalanobis distance that improves the Optimal Transport between the source and target subspaces generated by a PCA. Unlike OTDA which does not change the feature space, MLOT jointly optimizes (i) the dimensionality reduction of the source domain, (ii) the transportation plan between the source and the target and (iii) the underlying metric used in the transportation. The intuition behind MLOT is illustrated in Figure 2.1. The original source and target examples are represented on the left. The second figure shows the limitation of OTDA when the transportation is performed in the original feature space. The figure on the right gives evidence on the advantage of performing a PCA before learning jointly the ground metric and the transportation plan.

The rest of the chapter is organized as follows: Section 2.2 introduces the main principles of OTDA and Metric Learning. Section 2.3 is dedicated to the theoretical contribution of our chapter which leads to the design of our MLOT algorithm in Section 2.4. An extensive experimental study is presented in Section 2.5 before our conclusion in Section 2.6.

2.2 OTDA and Metric Learning

In this section, we briefly present OTDA as introduced in (Courty et al., 2017b), and the Metric Learning theory when the objective is to optimize a Mahalanobis distance.

2.2.1 OTDA

Let $\mathcal{X} \subseteq \mathbb{R}^D$ be a feature space. We consider here a source distribution μ_s and a target distribution μ_t both defined over $\mathcal{X}$ with finite p-order moment with $p \geq 1$. In practice, we deal with the empirical measures μ_s and μ_t supported on I and K examples respectively. We suppose that the two empirical measures are uniformly and independently sampled from sampled from μ_s and μ_t. Considering the Euclidean distance as the cost function, we denote by $\boldsymbol{C}^p$ the $I \times K$ matrix composed of the costs $\boldsymbol{C}^p_{ik} = \left\|\boldsymbol{x}^s_i - \boldsymbol{x}^t_k\right\|_2^p$.

OTDA (Optimal Transport for Domain Adaptation) (Courty et al., 2017b) was the first

attempt to use Optimal Transport for Domain Adaptation. While the original Optimal Transport Problem (1.6) is totally unsupervised, OTDA takes into account the labels of the transported points. Let us suppose that a discrete label $\boldsymbol{Y}_i^s \in \mathcal{Y} = [\![1, c]\!]$ is associated to each source example i, with c the number of classes. OTDA adds a $lr - l2$ group-lasso penalty term (Yuan and Lin, 2006; Blondel et al., 2018), with $r \in [0, 1]$ to prevent two source points of different labels from being sent to the same target location. This takes the form of the following optimization problem:

$$\min_{\boldsymbol{T} \in \Pi(\hat{\mu}_s, \hat{\mu}_t)} \langle \boldsymbol{T}, C^p \rangle - \epsilon \mathcal{H}(\boldsymbol{T}) + \lambda_c \Omega_c(\boldsymbol{T}), \tag{2.1}$$

where $\Omega_c(\boldsymbol{T}) = \sum_{k=1}^{K} \sum_{cl=1}^{c} \|\boldsymbol{T}(\mathcal{I}_{cl}, k)\|_2^r$ and $\boldsymbol{T}(\mathcal{I}_{cl}, k)$ is the column k of matrix $\boldsymbol{T}$ with only the rows corresponding to samples of class cl. Intuitively, for a given target point $\boldsymbol{x}_k^t$, the lr norm between the classes ensures the sparsity between the classes, thus $\boldsymbol{x}_k^t$ should not received mass from two points with different classes. Similarly to the entropy regularization, the $l2$ norm for each class penalizes strong confidence and prompts smooth predictions. See (Blondel et al., 2018) for more details on the group-lasso regularization for Optimal Transport. Interestingly, to solve Equation (2.1), Courty et al. (2017b) proposes to use the generalized conditional gradient algorithm (Bredies et al., 2009) which is a variant of the Frank Wolfe algorithm presented in Section 1.3.1.

2.2.2 Metric Learning

As said before, the cost matrix $\boldsymbol{C}^p$ used in OT is usually set to the Euclidean distance. Though this choice seems natural, it has a direct impact on the quality of the transportation. We suggest here to optimize this cost matrix by learning a metric that allows us to better match in a DA setting the source and target distributions, hopefully in a smaller feature space. Metric Learning (ML), and especially Mahalanobis distance learning, has been widely studied in the literature during the past decade (see, *e.g.*, the surveys (Bellet et al., 2013, 2015)). It typically boils down to optimizing the shape and the orientation of an ellipsoid rather than using the Euclidean ball. More formally, let $\boldsymbol{L} \in \mathbb{R}^{d \times D}$ with $d \in [\![1, D]\!]$ and M be a PSD matrix such as $\boldsymbol{M} = \boldsymbol{L}^\top \boldsymbol{L}$. For all $(i, j) \in [\![1, I]\!]^2$, the squared Mahalanobis distance $D_{\boldsymbol{M}}^2$ parameterized by $\boldsymbol{M}$ is defined as

$$D_{\boldsymbol{M}}^2(\boldsymbol{x}_i^s, \boldsymbol{x}_j^s) = (\boldsymbol{x}_i^s - \boldsymbol{x}_j^s)^\top \boldsymbol{M} (\boldsymbol{x}_i^s - \boldsymbol{x}_j^s) = \|\boldsymbol{L}(\boldsymbol{x}_i^s - \boldsymbol{x}_j^s)\|_2^2. \tag{2.2}$$

Notice that $\boldsymbol{L}$ defines a unique $\boldsymbol{M}$ but there is more than one Cholesky decomposition of $\boldsymbol{M}$. The goal of Metric Learning is to learn either the matrix $\boldsymbol{L}$ or $\boldsymbol{M}$ under semantic constraints, which typically aim to bring examples of the same class closer while pushing away data with different labels (see, *e.g.* LMNN (Weinberger and Saul, 2009) or ITML (Davis et al., 2007)). The problem is often convex in $\boldsymbol{M}$ but the PSD constraint makes the optimization more complicated. The minimization in $\boldsymbol{L}$ is not convex but is simpler and gives good results in practice. In the rest of this chapter, we will denote by $\Omega_l(\boldsymbol{L})$ the underlying objective function of the metric learning problem. Note that learning a Mahalanobis distance for OT has been recently studied (Cuturi and Avis, 2014; Genevay et al., 2018; Deshpande et al., 2019; Paty and Cuturi, 2019b). Our

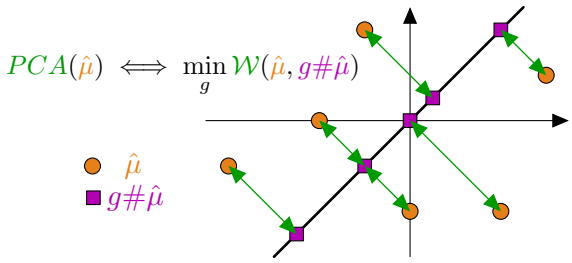


Figure 2.2: *Standard PCA projecting a discrete distribution $\hat{\mu}$ (data in orange) onto a one-dimensional subspace (in purple). This projection is equivalent to finding the optimal mapping function g that minimizes the Wasserstein distance $\mathcal{W}(\hat{\mu}, g\#\hat{\mu})$.*
Cited on page [37]

objective in the rest of this chapter is to show how to optimize such a ground metric when OT is used to address domain adaptation tasks by the alignment of the source and target subspaces.

2.3 Theoretical analysis of Domain Adaptation with OT

In this section, we derive two theoretical results. First, we establish a strong relation between a PCA and the minimization of the Wasserstein distance. Then, we derive a generalization bound on the target error whose terms depend on several Wasserstein distances. By changing the Euclidean ground distance by a Mahalanobis distance, we can see $\mathcal{W}_p^p(\hat{\mu}_s, \hat{\mu}_t)$ as a parameterized distance that might be optimized from training data. This leads to the design of a new Domain Adaptation algorithm, called MLOT (see Section 2.4), which learns the ground metric allowing us to optimize the transportation plan while performing the adaptation.

2.3.1 PCA and Wasserstein Distance

As usually done in OT, let us use the push forward notation. For any measurable function $g : \mathcal{X} \longrightarrow \mathcal{X}$ and distribution μ on $\mathcal{X}$, we define $g\#\mu(B) = \mu(g^{-1}(B))$ for all Borel $B \in \mathcal{B}(\mathcal{X})$. In practice, this means that we draw a point from μ and then apply the transformation g to that point. If g is a linear function, it can be assimilated to its associated matrix G. Let $Dim(Im(g))$ be the dimension of the *affine* subspace formed by the image of g. This notation allows us to define the dimension of a non-centered vector space.

The following theorem aims at showing that PCA is the best way to reduce the dimension of a distribution in the sense of the Wasserstein distance. By dimensionality reduction, we mean that the transformed distribution lies in a subspace (not necessarily centered) of $\mathcal{X}$, but the points are still defined in $\mathcal{X}$ (see Figure 2.2). For the sake of clarity, we consider here the case of the 2-Wasserstein with the Euclidean distance as the underlying ground distance. We also focus on the case of a discrete centered distribution $\hat{\mu}$. The complete derivation for the more general case can be proved in a similar way.

Theorem 1. *Given a set of I examples $\{\boldsymbol{x}_i\}_{i=1}^I$ lying in a D dimensional space and i.i.d. from a distribution μ. Let $\hat{\mu}$ the empirical counterpart of μ defined as $\hat{\mu} = \frac{1}{I}\sum_{i=1}^I \delta_{\boldsymbol{x}_i}$ with $\mathbb{E}(\hat{\mu}) = 0$. Let $d \in [\![1, D]\!]$ and $\boldsymbol{V}$ be the $d \times D$ matrix formed with the first d normalized eigenvectors of the covariance matrix of $\hat{\mu}$. Let $G_d = \{g : \mathbb{R}^D \longrightarrow \mathbb{R}^D | Dim(Im(g)) \leq d\}$. Then, we have*

$$\underset{g \in G_d}{\text{argmin}}\ \mathcal{W}_2^2(\hat{\mu}, g\#\hat{\mu}) = \boldsymbol{V}^\top \boldsymbol{V}. \tag{2.3}$$

Proof. Suppose that the minimizer $g^* \in G_d$ of Problem (2.3) exists, we will show that it is equal to the PCA result. Let $\boldsymbol{T}_{g^*}$ be the optimal coupling matrix associated with g^*. With these notations, we get:

$$\min_{g \in G_d} \mathcal{W}_2^2(\hat{\mu}, g\#\hat{\mu}) = \sum_{i,k=1}^{I,I} \|\boldsymbol{x}_i - g^*(\boldsymbol{x}_k)\|_2^2 (\boldsymbol{T}_{g^*})_{ik}. \tag{2.4}$$

As $\boldsymbol{T}_{g^*}$ is a permutation matrix (divided by I), one can reorder the rows to have the identity matrix (divided by I). We note $\boldsymbol{T}_{\tilde{g}^*}$ such matrix with $\tilde{g}^*$ the associated function. The function $\tilde{g}^*$, is defined as,

$$\forall i \in [\![1, I]\!]\ \tilde{g}^*(\boldsymbol{x}_i) = g^*(\boldsymbol{x}_k) \text{ for } k \text{ such that } (\boldsymbol{T}_{g^*})_{ik} = \frac{1}{I}, \tag{2.5}$$

and is equal to g^* elsewhere. As $\boldsymbol{T}_{g^*}$ is a permutation matrix, such index k always exists. Using this new function $\tilde{g}^*$ and the new associated transport plan $T_{\tilde{g}^*}$, we get:

$$\begin{aligned}
\min_{g \in G_d} \mathcal{W}_2^2(\hat{\mu}, g\#\hat{\mu}) &= \sum_{i,k=1}^{I,I} \|\boldsymbol{x}_i - \tilde{g}^*(\boldsymbol{x}_k)\|_2^2 (\boldsymbol{T}_{\tilde{g}^*})_{ik} \\
&= \frac{1}{I}\sum_{i=1}^{I} \|\boldsymbol{x}_i - \tilde{g}^*(\boldsymbol{x}_i)\|_2^2 \\
&\geq \frac{1}{I}\sum_{i=1}^{I} \|\boldsymbol{x}_i - \boldsymbol{V}^\top \boldsymbol{V} \boldsymbol{x}_i\|_2^2.
\end{aligned}$$

The last line comes from the PCA definition. It shows that $\boldsymbol{V}^\top \boldsymbol{V}$ associated with $\frac{1}{I}\mathbb{I}_I$ is smaller than the optimal solution g^*. But since $\boldsymbol{V}^\top \boldsymbol{V} \in G_d$, it is actually the optimal solution. □

Theorem 1 tells us that PCA is the best way to reduce the dimension in the sense of the Wasserstein distance. The next result provides a generalization bound on the target error where Wasserstein distances are the main terms to minimize.

2.3.2 Upper Bound on the Target Risk

Let $\mathcal{H} = \{h | h : \mathcal{X} \to \mathcal{Y}\}$ be the hypothesis space. We assume the existence of a deterministic ground-truth function $f : \mathcal{X} \to \mathcal{Y}$ which gives the label associated to each point of $\mathcal{X}$. For all $(h, h') \in \mathcal{H}^2$, we extend the notation of the risk $\mathcal{R}$ and define $\mathcal{R}_t(h, h') = P_{x \sim \mu_t}(h(x) \neq h'(x))$ and $\mathcal{R}_s(h, h') = P_{x \sim \mu_s}(h(x) \neq h'(x))$ which represent the disagreement between two classifiers.

The goal of DA is to find the best $h \in \mathcal{H}$ which minimizes the target risk $\mathcal{R}_t(h, f) = P_{x\sim\mu_t}(h(x) \neq f(x))$. In the same way, $\mathcal{R}_s(h, f) = P_{x\sim\mu_s}(h(x) \neq f(x))$ is the source risk. To simplify the notations, let also use $\mathcal{R}_t(h, f) = \mathcal{R}_t(h)$ and $\mathcal{R}_s(h, f) = \mathcal{R}_s(h)$ in which the optimal labeling function f is implicit.

Lemma 1 (Generalization bound (Shen et al., 2017))**.** *Let μ_s, μ_t be two probability measures on $\mathcal{X}$. Assume the hypotheses $h \in \mathcal{H}$ are all κ-Lipschitz continuous for some $\kappa \in \mathbb{R}_+^*$. Then, $\forall (h, h)' \in \mathcal{H}^2$, the following holds:*

$$\mathcal{R}_t(h, h') \leq \mathcal{R}_s(h, h') + 2\kappa \mathcal{W}_2(\mu_s, \mu_t). \tag{2.6}$$

Building on the proof of Redko et al. (2017) based on the seminal works of Ben-David et al. (2007) and Mansour et al. (2009), we derive the following bound.

Theorem 2. *Let $g_s : \mathcal{X} \longrightarrow \mathcal{X}$ and $g_t : \mathcal{X} \longrightarrow \mathcal{X}$. Under the assumption of Lemma 1, $\forall h \in \mathcal{H}$, the following holds:*

$$\begin{aligned}\mathcal{R}_t(h) \leq\ & \mathcal{R}_s(h) + 2\kappa\left[\mathcal{W}_2(g_s\#\hat{\mu}_s, g_t\#\hat{\mu}_t)\right] \\ & + 2\kappa\left[\mathcal{W}_2(\hat{\mu}_s, g_s\#\hat{\mu}_s) + \mathcal{W}_2(g_t\#\hat{\mu}_t, \hat{\mu}_t)\right] \\ & + 2\kappa\left[\mathcal{W}_2(\mu_s, \hat{\mu}_s) + \mathcal{W}_2(\hat{\mu}_t, \mu_t)\right] + \lambda\end{aligned} \tag{2.7}$$

where λ is the combined error of the ideal hypothesis h^* that minimizes the combined error $\mathcal{R}_s(h^*) + \mathcal{R}_t(h^*)$.

Proof. We have :

$$\begin{aligned}\mathcal{R}_t(h) \leq & \mathcal{R}_t(h^*) + \mathcal{R}_t(h^*, h) \\ = & \mathcal{R}_t(h^*) + \mathcal{R}_s(h, h^*) + \mathcal{R}_t(h^*, h) - \mathcal{R}_s(h, h^*) \\ \leq & \mathcal{R}_t(h^*) + \mathcal{R}_s(h, h^*) + 2\kappa\mathcal{W}_2(\mu_s, \mu_t) \\ \leq & \mathcal{R}_t(h^*) + \mathcal{R}_s(h) + \mathcal{R}_s(h^*) \\ & + 2\kappa\left[\mathcal{W}_2(\mu_s, \hat{\mu}_s) + \mathcal{W}_2(\hat{\mu}_s, \mu_t)\right] \\ \leq & \mathcal{R}_s(h) + \lambda \\ & + 2\kappa\left[\mathcal{W}_2(\mu_s, \hat{\mu}_s) + \mathcal{W}_2(\hat{\mu}_s, \hat{\mu}_t) + \mathcal{W}_2(\hat{\mu}_t, \mu_t)\right] \\ \leq & \mathcal{R}_s(h) + 2\kappa\left[\mathcal{W}_2(g_s\#\hat{\mu}_s, g_t\#\hat{\mu}_t)\right] \\ & + 2\kappa\left[\mathcal{W}_2(\hat{\mu}_s, g_s\#\hat{\mu}_s) + \mathcal{W}_2(g_t\#\hat{\mu}_t, \hat{\mu}_t)\right] \\ & + 2\kappa\left[\mathcal{W}_2(\mu_s, \hat{\mu}_s) + \mathcal{W}_2(\hat{\mu}_t, \mu_t)\right] + \lambda.\end{aligned}$$

□

Note that if g_s and g_t are linear, $g_s\#\hat{\mu}_s$ and $g_t\#\hat{\mu}_t$ can be seen as linear projections of the source and target examples taking the form of two matrices that can be learned by any standard metric learning algorithm. It is worth noticing that the previous bound is the first one which jointly relates (i) the target risk in domain adaptation, (ii) the minimization of the Wasserstein

distance and (iii) the metrics that can be learned to get a better transportation plan. We can use Theorem 1 to minimize both $\mathcal{W}_2(\hat{\mu}_s, g_s\#\hat{\mu}_s)$ and $\mathcal{W}_2(g_t\#\hat{\mu}_t, \hat{\mu}_t)$. Note that $\mathcal{W}_2(\mu_s, \hat{\mu}_s)$ and $\mathcal{W}_2(\hat{\mu}_t, \mu_t)$ can be bounded under some assumptions using Theorem 2.1 in (Bolley et al., 2007). Moreover, λ is supposed to be small to allow the adaptation. A theoretical analysis about λ is available in (Redko et al., 2019b).

The last term that has to be minimized in the bound of Theorem 2 is $\mathcal{W}_2(g_s\#\hat{\mu}_s, g_t\#\hat{\mu}_t)$. We address this problem from an algorithmic perspective thanks to our algorithm MLOT presented in the next Section.

Algorithm 3 MLOT Cited on page [40]

Input: η (gradient step) $\boldsymbol{X}^s$ $\boldsymbol{X}^t$ $\boldsymbol{Y}^s$

1: $\boldsymbol{V}_s = PCA(\boldsymbol{X}^s)$, $\boldsymbol{V}_t = PCA(\boldsymbol{X}^t)$
2: $\boldsymbol{L}_s = \boldsymbol{V}_s^\top \boldsymbol{V}_s$, $\boldsymbol{L}_t = \boldsymbol{V}_t^\top \boldsymbol{V}_t$
3: **for** $i = 1$ **to** P **do**
4: $\quad \boldsymbol{T} = \underset{\boldsymbol{T}\in\Pi(\hat{\mu}_s,\hat{\mu}_t)}{\text{argmin}} \langle \boldsymbol{T}, C^2(\boldsymbol{L}_s, \boldsymbol{L}_t)\rangle - \epsilon\mathcal{H}(\boldsymbol{T}) + \lambda_c\Omega_{cl}(\boldsymbol{T})$
5: $\quad \boldsymbol{L}_s = \boldsymbol{L}_s - \eta\nabla_{\boldsymbol{L}_s}(\langle \boldsymbol{T}, C^2(\boldsymbol{L}_s, \boldsymbol{L}_t)\rangle + \lambda_l\Omega_l(\boldsymbol{L}_s))$
6: **end for**
7: $\tilde{\boldsymbol{X}}_s = \boldsymbol{T}\boldsymbol{L}_t\boldsymbol{X}^t$
8: classifier = classifier method($\tilde{\boldsymbol{X}}_s, \boldsymbol{Y}^s$)
9: $\hat{\boldsymbol{Y}}^t$ = classifier($\boldsymbol{X}^t$)
10: **return** $\hat{\boldsymbol{Y}}^t$

2.4 MLOT: Metric Learning in OT for DA

Inspired from OTDA (Courty et al., 2017b) our algorithm MLOT leverages our previous theoretical analysis and resorts to an additional term $\Omega_l(\boldsymbol{L}_s)$ dedicated to optimize a metric allowing us to get a better transportation plan. Let us consider the cost function $C^2(\boldsymbol{L}_s, \boldsymbol{L}_t)_{ij} = \|\boldsymbol{L}_s\boldsymbol{x}_i^s - \boldsymbol{L}_t\boldsymbol{x}_k^t\|_2^2$. MLOT takes the form of the following joint optimization problem:

$$\min_{\boldsymbol{L}_s\in\mathbb{R}^{D\times D}, \boldsymbol{T}\in\Pi(\hat{\mu}_s,\hat{\mu}_t)} \langle \boldsymbol{T}, C^2(\boldsymbol{L}_s, \boldsymbol{L}_t)\rangle - \epsilon\mathcal{H}(\boldsymbol{T}) + \lambda_c\Omega_{cl}(\boldsymbol{T}) + \lambda_l\Omega_l(\boldsymbol{L}_s). \tag{2.8}$$

Note that MLOT only learns the matrix $\boldsymbol{L}_s$ associated to the source data. The reason is twofold. First, it prevents the algorithm from leading to a trivial minimal solution $\mathcal{W}_2(\boldsymbol{L}_s\#\hat{\mu}_s, \boldsymbol{L}_t\#\hat{\mu}_t)$ where both matrices $\boldsymbol{L}_s$ and $\boldsymbol{L}_t$ are null. By this way, MLOT tends to provide two different matrices $\boldsymbol{L}_s$ (which is learned) and $\boldsymbol{L}_t$ (set to $\boldsymbol{V}_t^\top\boldsymbol{V}_t$ according to Theorem 1) which better capture the peculiarities of the two distributions. Second, labels - that are required to learn a metric - are available only in the source domain. To find the solution of Problem (2.8), we minimize the objective function w.r.t. $\boldsymbol{T}$ and $\boldsymbol{L}_s$ alternately. From a practical point of view, we apply one step of gradient descent over $\boldsymbol{L}_s$ and then completely compute the optimal $\boldsymbol{T}$. Since

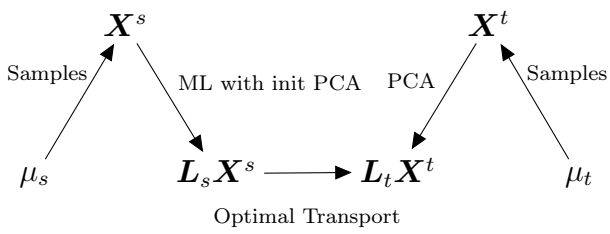


Figure 2.3: *Workflow of MLOT.* *Cited on page [41]*

the problem is not convex, the initialization of $\boldsymbol{L}_s$ is key. According to Theorem 1, we use a PCA to set $\boldsymbol{L}_s = \boldsymbol{V}_s^\top \boldsymbol{V}_s$. At the initialization, $\boldsymbol{L}_s$ is a $D \times D$ matrix of rank d. But note that along the iterations, this rank can increase if it allows a better adaptation. The pseudo-code of MLOT is described in Algorithm 3, where $\boldsymbol{X}^s$ and $\boldsymbol{X}^t$ are the source and target datasets, and $\boldsymbol{Y}^s$ is the set of source labels.

Note that any gradient descent-based metric learning algorithm can be used to learn $\boldsymbol{L}_s$ via the term $\lambda_l \Omega_l(\boldsymbol{L}_s)$. Note also that the computation of the barycenter of the transported points $\tilde{\boldsymbol{X}}_s = \boldsymbol{T}\boldsymbol{L}_t\boldsymbol{X}^t$ makes sense only if the 2-Wasserstein is used. Figure 2.3 summarizes the workflow of MLOT.

2.5 Experiments

In this section, we perform experiments and demonstrate the effectiveness of MLOT compared to OTDA and other baselines on various datasets and types of features.

2.5.1 Datasets

We use the Office-Caltech dataset (Gong et al., 2012) which is a classical benchmark on visual DA. We study the effect of using different features such as SURF features (Bay et al., 2006) and DeCAF Deep Learning features (Donahue et al., 2014). The Office-Caltech dataset is composed of 4 different subsets (Amazon, Caltech, DSRL, Webcam) that are combined in a pairwise manner, to create 12 DA subproblems. The notation $A \to C$ means that Amazon is used as the source and Caltech as the target. There are the same 10 classes in each dataset containing from 157 to 1,123 images.

We also use the Office31 dataset (Saenko et al., 2010) with features extracted from the 7th layer of DeCAF Deep Learning network. The dataset is composed of 4,110 images in 3 subsets (Amazon, DSLR, Webcam) with 31 classes.

2.5.2 Setup and Cross-validation

We compare 10 different methods that are able to handle arbirtary features. We exclude deep learning methods as they require having the original images and fine tuning a network. We compare: **NA**(No Adaptation). The classifier is learned on the source dataset and directly

applied on the target. **LMNN**: Large Margin Nearest Neighbor (Weinberger and Saul, 2009). **SA**: Subspace Alignment (Fernando et al., 2013). **CORAL**: CORrelation ALignment (Sun et al., 2016a). **TCA**: Transfer Component Analysis (Pan et al., 2011). **OT**: Optimal Transport with entropy (Cuturi, 2013). **OTDA**: Optimal Transport with entropy and class regularization (Courty et al., 2017b). $\mathbf{OTDA_p}$: OTDA after a PCA. **JDOT**: Joint Distribution Optimal Transportation (Courty et al., 2017a). **MLOT**: our method. Following (Courty et al., 2017b) the final classification is done with a 1-Nearest Neighbor (1NN), except for JDOT where we use a SVM ($C = 1$).

In unsupervised DA, there is no target label and it is impossible to use the classical cross-validation procedure to choose the best hyper-parameters. To fairly compare methods, we take inspiration from the work of Zhong et al. (2010), introduced in Section 1.2.2, and apply the following strategy for all methods. We first assign pseudo-labels to the target points (using the considered method) and then use these target labels to re-assign labels to the source data, using a basis DA algorithm. Here, we choose SA (Fernando et al., 2013) which has been shown to be one of the most robust DA method. We can then compare the actual source labels with the predicted source ones. We take the set of hyper-parameters that gives the best accuracy over 48 hours, limited to 1000 iterations. This back-and-forth adaptation is done independently for each pair of datasets. MLOT is parameterized by 5 hyper-parameters: the three regularization parameters (ϵ, λ_c, λ_l) which control the trade-off between each term in Eq. (2.8), the number of dimensions kept by the PCA (d) and the number of iterations (P). Note that in these experiments, we used arbitrarily LMNN (Weinberger and Saul, 2009) to learn $\boldsymbol{L}_s$ in the term $\Omega_l(\boldsymbol{L}_s)$. Therefore, an additional parameter has to be tuned corresponding to the margin used in this metric learning algorithm. Note that SA and MLOT resort to a PCA. To speed-up the process, we used a "randomized" -PCA (Halko et al., 2011) and run 10 iterations. This explains why the variance is indicated for these three methods in the reported results in Table 2.1.

The code of the 10 methods is available[1], together with the datasets, the code for the cross-validation that recreates Table 2.1, and the code that produces automatically Figures 2.1 and 2.4.

2.5.3 Analysis of the Results

The results are reported in Table 2.1. For the SURF features, MLOT outperforms, on average, all the other methods. MLOT outperforms OTDA 8 times over the 12 DA subproblems and yields impressive improvements for some cases (*e.g.* $W \to A$ and $C \to D$) and a clear gain on average (1.5 points). The results on Office-Caltech DeCAF6 features show the effectiveness of our method on deep learning features. MLOT outperforms by 1.8 the second best method (here OT). On the Office31 dataset, the best results are obtained by SA. MLOT is still very competitive and outperforms OTDA by 0.9 point on average. The entire table in available in Appendix A.1. In addition, a small experiment about the interest of using jointly a Metric

[1] `https://github.com/Hv0nnus/MLOT`

	Dataset	NA	LMNN	SA	CORAL	TCA	OT	OTDA	$\text{OTDA}_\mathbf{p}$	JDOT	MLOT
SURF	A→C	26.0	40.3	40.2±0.2	25.4	40.0	33.9	40.2	39.4±0.5	39.9	**42.3**±0.6
	A→D	25.5	36.9	39.3±2.5	26.8	31.8	30.6	40.1	39.6±1.1	37.6	**40.8**±0.8
	A→W	29.8	38.0	39.9±1.3	26.8	**41.7**	32.5	37.3	39.8±0.9	38.0	41.3±1.1
	C→A	23.7	46.0	41.3±1.1	23.6	39.8	41.0	**52.7**	48.5±0.7	48.1	51.5±0.8
	C→D	25.5	45.9	45.4±1.2	26.1	44.6	36.9	47.8	51.4±1.4	49.7	**52.2**±1.3
	C→W	25.8	41.7	36.6±1.1	23.7	36.9	28.1	**46.4**	45.8±1.4	43.4	45.9±0.8
	D→A	28.5	31.1	35.4±1.0	28.8	32.9	29.3	32.4	37.8±1.0	32.8	**37.8**±0.7
	D→C	26.3	30.7	32.3±0.6	30.0	31.5	31.7	32.0	33.5±0.7	31.7	**34.4**±0.5
	D→W	63.4	77.3	88.5±1.1	84.4	84.7	**88.8**	**88.8**	87.5±1.2	82.7	87.8±0.7
	W→A	23.0	32.3	32.6±0.5	26.2	29.4	34.1	33.7	37.6±0.6	37.6	**38.0**±0.8
	W→C	19.9	30.4	29.0±0.6	22.6	29.2	30.1	**34.1**	33.3±0.5	33.1	33.2±0.6
	W→D	59.2	86.6	89.5±1.0	84.1	91.7	89.2	**92.4**	91.8±1.2	89.8	90.8±0.8
	AVG	31.4	44.8	45.8±1.0	35.7	44.5	42.2	48.2	48.8±0.9	47.0	**49.7**±0.8
DeCAF6	AVG	71.0	79.4	83.7±0.5	77.2	83.4	83.9	83.2	82.6±0.5	78.2	**84.7**±0.3
Office31	AVG	64.3	64.7	**66.5**±0.2	64.1	64.1	65.3	65.3	65.2±0.1	64.4	66.2±0.1
All datasets	AVG	53.8	62.6	65.1 ± 0.6	58.0	64.0	63.5	65.6	65.6 ± 0.6	63.0	**67.0** ± 0.5

Table 2.1: *Accuracy of all the methods on 3 different types of features. The best method for each dataset is in bold.*

Cited on pages [42,42,42]

Learning and an Optimal Transport algorithm is provided in Appendix A.1.2.

As already mentioned, cross-validating the hyperparameters in unsupervised DA is key and is still an open problem, since we do not have access to labels from the target domain. We performed an experimental comparison to show how the cross-validation method used in this chapter behaves when compared to a scenario where we would select the hyperparameters using the actual labels of the target examples. Table 2.2 reports the gap between the optimal hyperparameters and those obtained by our method inspired from (Zhong et al., 2010). We can see that the ranking of the methods is preserved, even though this experiment shows that there is still room for improving the way we may tune the parameters in unsupervised DA. Note that the other datasets show similar behaviors. To show the specific gain brought by MLOT compared

Cross-validation using...	OT	TCA	LMNN	SA	JDOT	OTDA	OTDA_p	MLOT
target true labels	45.3	45.3	47.9	47.4	48.5	52.8	54	**55.1**
target pseudo-labels	42.2	44.5	44.8	45.8	47.0	48.2	48.8	**49.7**

Table 2.2: *Accuracy comparison on Office-Caltech (SURF features) between a cross-validation method that uses the true target labels (first line) and the cross-validation method used in this chapter that exploits pseudo-labels in the unsupervised DA setting (second line). CORAL and NA are excluded as they do not have hyperparameter.*

Cited on page [43]

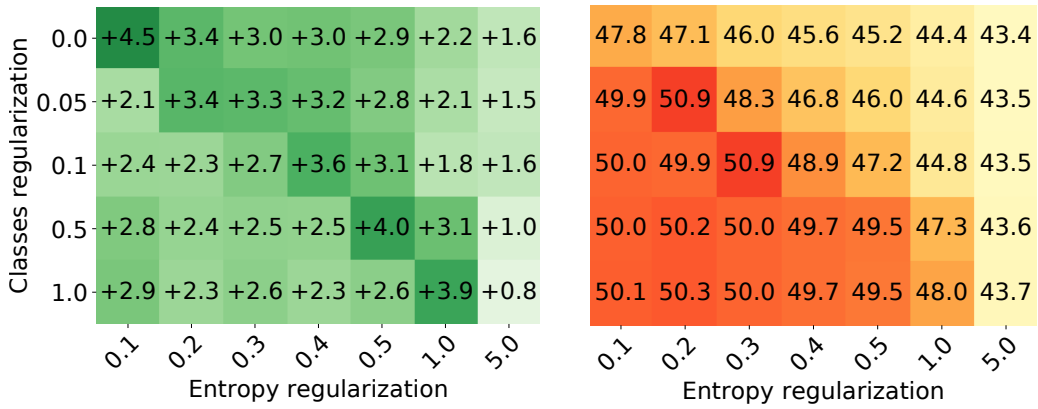


Figure 2.4: *Absolute difference between the mean accuracy of MLOT and OTDA (on the left). The rows (resp. columns) correspond to the results with different values of the regularization parameter λ_c (resp. ϵ) on the entire Office-Caltech dataset with SURF features. On the right, accuracy of MLOT for each pair of parameters.*

Cited on pages [42,43,43]

to OTDA, we performed a last experiment where we set the hyper-parameters of MLOT as follows: $\lambda_l = 1$, $P = 10$, $margin_{LMNN} = 10$, $d = 70$; and we tune ϵ and λ_c. The results are reported in Figure 2.4. It is worth noticing that whatever the set of parameters, MLOT always yields better accuracy, which confirms the interest of learning a metric and using the PCA to initialize $\boldsymbol{L}_s$ and $\boldsymbol{L}_t$. Note that when the entropy term has more importance (last column), the difference between the two methods is smaller because $\boldsymbol{T}$ tends to be uniform. When the class regularization is set to 0 (first row), OTDA becomes similar to OT (Sinkhorn algorithm). This shows the effectiveness of MLOT even without the class regularization. However, the performances drop without this supervised information which tends to show that the metric learned and the class regularization are complementary. Notice that the results of MLOT are quite good for many values of the entropy and class regularizations (Figure 2.4 on the right). The best performance is 50.9 which is better than the result obtained by the cross-validation method. Once again, this is an evidence about the difficulty of tuning parameters in unsupervised DA.

2.6 Conclusion

We proposed in this chapter a new Domain Adaptation (DA) method, called MLOT, benefiting from both Metric Learning (ML) and Optimal Transport (OT). Dedicated to address problems in the complex unsupervised DA setting, MLOT jointly learns a good metric and the optimal transportation plan. A theoretical study has driven the design of MLOT. We derived a bound on the target error which prompts us to learn the ground metric involved in the Wasserstein distance. The experimental study has shown very competitive results and a significant improvement compared to OTDA, the first method coupling OT and DA.

In the next chapter, we will keep analyzing new ways for choosing a good ground metric. It

will not be restricted to DA task as we will explore the interest of robust ground metric in any OT problem by solving a complex min-max problem.

Chapter 3

A Swiss Army Knife for Minimax Optimal Transport

Abstract

In the previous chapter, a new method that jointly learns the ground metric and the transport plan was proposed to solve a Domain Adaptation task. This chapter is based on the following contribution "A Swiss Army Knife for Minimax Optimal Transport" published at ICML 2020 (Dhouib et al., 2020). This work was done in collaboration with Sofiane DHOUIB whose PhD thesis also addressed OT problems. Thus another description of this work is available in his manuscript (Dhouib, 2020). The Optimal transport (OT) problem and its underlying optimization problem is known to have two major restrictions: (i) it strongly depends on the choice of the cost function and (ii) its sample complexity scales exponentially with the dimension. In this chapter, we propose a general formulation of a minimax OT problem that can tackle these limitations by jointly optimizing the cost matrix and the transport plan, allowing us to define a robust distance between distributions. We propose to use a cutting-set method to solve this general problem and show its links and advantages compared to other existing minimax OT approaches. Additionally, we use this method to define a notion of stability allowing us to select a ground metric robust to bounded perturbations. Finally, we provide an experimental study highlighting the efficiency of our approach.

3.1 Introduction

In this chapter, we study a general formulation of the OT problem with a minimax objective function where one seeks an OT plan with respect to (w.r.t.) the worst possible ground metric belonging to an arbitrary and possibly infinite convex set. Such a minimax formulation is of a particular interest as it has been shown previously (i) to reduce the sample complexity and increase the robustness to noise of the original OT problem for high-dimensional data (Paty and Cuturi, 2019a), (ii) to allow to consider submodular cost functions (Alvarez-Melis et al., 2018)

and (iii) to use it as a loss in generative models (Genevay et al., 2018). We advance the study of the minimax OT further by providing the following contributions. First, for an infinite set of cost functions defined by a Mahalanobis distance, we reformulate the minimax OT problem as a minimization of the arbitrary dual norm of the matrix of second-order displacements and show how one can use it to smoothly interpolate between the original OT problem and the minimax formulation of (Paty and Cuturi, 2019a) included as a special case. Second, we provide a generic solver for minimax OT for both regularized and unregularized minimax OT problems and for both finite and infinite families of cost functions contrary to previous work (Paty and Cuturi, 2019a; Alvarez-Melis et al., 2018) that considered the differentiable and strictly convex regularized OT problem only. Finally, we introduce the notion of cost matrix stability and solve its underlying optimization problem. It consists in finding a cost function from a list of possible candidates that leads to a stable transportation cost in its unit ball neighborhood.

The rest of this chapter is organized as follows. In Section 3.2, we provide some preliminary knowledge as well as the notations used throughout the chapter. We then present in Section 3.3 the main contributions of this chapter including a general minimax formulation for the OT problem with an arbitrary convex compact set of cost matrices and discuss an optimization procedure that can be used to solve it as well as its theoretical guarantees. We further proceed by considering the important special cases of the previously introduced problem and showing their relationship to other works on subject. Finally, in Section 3.4, we present an experimental evaluation of our approach for several considered use-cases.

3.2 Preliminary Knowledge

Let μ and ν be two distribution with finite p-order moment on two vector spaces $\mathcal{X}$ and $\mathcal{Y}$. As we will not use any labeled points, we will keep the usual OT notations and use $(\boldsymbol{x}_i)_{i\in[\![1,I]\!]}$ and $(\boldsymbol{y}_k)_{k\in[\![1,K]\!]}$ for the support of the empirical distributions μ and ν. The probability vector associated are $\boldsymbol{a} \in \Delta_I$ and $\boldsymbol{b} \in \Delta_K$ respectively. To simplify the notation of the polytope, we will often use Π instead of $\Pi_{\mu\nu}$ when it is clear from the context.

Minimax OT Two other studies considered the minimax formulation of the OT problem in a setting similar to ours. In the first one, Paty and Cuturi (2019a) showed that one can see the OT problem, with c taken to be the squared Euclidean distance, as a trace minimization problem of the second-order displacement matrix defined for any $\pi \in \Pi_{\mu\nu}$ defined as:

$$\mathcal{W}_2^2(\mu,\nu) = \min_{\pi\in\Pi_{\mu\nu}} \mathrm{Tr}(\mathbf{V}_\pi),$$
$$\mathbf{V}_\pi := \int_{\mathcal{X}\times\mathcal{Y}} (\boldsymbol{x}-\boldsymbol{y})(\boldsymbol{x}-\boldsymbol{y})^\top d\pi(\boldsymbol{x},\boldsymbol{y}).$$

Paty and Cuturi (2019a) further used this expression of the 2-Wasserstein distance to introduce the Subspace Robust Wasserstein (SRW) distance as follows:

$$\mathcal{S}_d^2(\mu,\nu) := \min_{\pi\in\Pi} \mathrm{Tr}^d(\mathbf{V}_\pi) = \min_{\pi\in\Pi} \sum_{r=1}^{d} \lambda_r(\mathbf{V}_\pi), \tag{3.1}$$

where λ_r are the $d \leq D$ largest eigenvalues of $\mathbf{V}_\pi$. Note that considering only the maximization over the $d \leq D$ largest eigenvalues allows to learn a cost matrix of a reduced rank thus tackling the curse of dimensionality issue of calculating the Wasserstein distance for high-dimensional data.

A somehow different way of using the minimax formulation of OT was proposed in (Alvarez-Melis et al., 2018) for c taken to be a submodular function $F : 2^V \to \mathbb{R}$ with V denoting a certain set of available items. In this case, taking the Lovász extension f of F leads to the following optimization problem:

$$\mathrm{StrOT}(\mu,\nu) := \min_{\boldsymbol{T}\in\Pi} \max_{\mathbf{C}\in\mathcal{B}_F} \langle \boldsymbol{T}, \mathbf{C}\rangle ,$$

where $\mathcal{B}_F$ is the base polytope of F defined as $\mathcal{B}_F = \{y \in \mathbb{R}^{|V|} | y(V) = F(V); y(S) \leq F(S), \forall S \subseteq V\}$. A game-theoretic interpretation of this formulation is to consider two players, where Player 1 aims at aligning the two distributions by picking a coupling matrix $\boldsymbol{T}$, while Player 2 resists to it by choosing the cost matrix $\mathbf{C}$ from the set of admissible costs $\mathcal{B}_F$. When F is a modular function, the size of $\mathcal{B}_F$ is 1 thus recovering the original OT problem.

Other related work Three other papers presented an OT-based minimax formulation distantly related to ours. In (Genevay et al., 2018), the authors studied a generative model that uses Sinkhorn divergence as a fitting criterion and proposed to learn a cost function in this framework. Their problem is intrinsically different from ours as we do not consider the density fitting problem where one optimises the parameters of the fitted distribution. On the other hand, in (Li et al., 2019), the authors reduced the regularized OT formulation with relaxed marginal constraints into a minimax problem. Their formulation, however, is also different from ours as it does not seek to learn a cost matrix. Finally, the line of work on the Wasserstein distributionally robust optimization (Kuhn et al., 2019) is also very dissimilar to this chapter as this latter considers finding the best estimator of a density from a Wasserstein ball of a certain radius. We now proceed to the presentation of our contributions.

3.3 Robust Optimal Transport with a convex set of cost Matrices

Below, we formulate the general robust OT problem and highlight its properties in several cases of interest. We further propose and theoretically analyze a general algorithm that can be used to solve it.

3.3.1 Problem formulation

Let $\mathcal{C}$ be an arbitrary set of cost functions defined over $\mathcal{X} \times \mathcal{Y} \to \mathbb{R}$. This set may represent, for instance, a convex combination of cost function candidates provided by several experts, or it can be described by an infinite set of parameters. We impose no particular constraints on the cost functions belonging to $\mathcal{C}$ as long as the corresponding Kantorovich problems admit a solution. We now consider the following minimax problem:

$$\mathrm{RKP}(\Pi, \mathcal{C}) = \min_{\pi \in \Pi} \max_{c \in \mathcal{C}} \mathop{\mathbb{E}}_{\boldsymbol{x},\boldsymbol{y} \sim \pi} \left[c(\boldsymbol{x}, \boldsymbol{y}) \right], \tag{3.2}$$

where we look for a coupling π^* that is robust to the choice of a cost function $c \in \mathcal{C}$, by considering the worst achievable transportation cost. We denote the value at the solution of this problem by $\mathrm{RKP}(\Pi, \mathcal{C})$ where RKP stands for robust Kantorovich problem. We abuse the notation and use $\mathrm{RKP}(\mathcal{P}, \mathcal{C})$ for any set $\mathcal{P} \subset \Pi$ (even non convex) to denote $\mathrm{RKP}(\mathrm{conv}(\mathcal{P}), \mathcal{C})$, i.e., , solving for $\pi \in \mathrm{Conv}\,(\mathcal{P})$. We also extend the Wasserstein notation $\mathcal{W}$ to any set of function $\mathcal{C}$ instead of only the Euclidean distance at the power p, by defining $\mathcal{W}_{\mathcal{C}}(\mu, \nu) := \mathrm{RKP}(\Pi, \mathcal{C})$.

3.3.2 Choice of $\mathcal{C}$

Below, we consider two possible choices for the convex set $\mathcal{C}$. First, we study the infinite family of Mahalanobis distance cost matrices widely used in the metric learning literature (Bellet et al., 2015). Second, we consider a convex hull of a finite family of cost functions as in the example given above.

Infinite family of Mahalanobis distances

For any $\mathbf{u} = (\mathbf{u}_1, ..., \mathbf{u}_D) \in \mathbb{R}^D$ and any $\mathbf{M} \in \mathbb{R}^{D \times D}$, we define their respective p-norm and Schatten p-norm as

$$\|\mathbf{u}\|_p^p = \sum_{1 \leq d \leq D} |\mathbf{u}_d|^p, \ \|\mathbf{M}\|_p^p = \sum_{1 \leq d \leq D} \sigma_d^p(\mathbf{M}),$$

where $p \in [1, +\infty]$ and $(\sigma_d(\mathbf{M}))_{d \in [\![1,D]\!]}$ are $\mathbf{M}$'s singular values. In particular, if $\mathbf{M} \in \mathcal{S}_+^{D \times D}$, where $\mathcal{S}_+^{D \times D}$ denotes the set of symmetric positive semi-definite matrices (PSD), then $\|\mathbf{M}\|_p = \mathrm{Tr}(\mathbf{M}^p)^{\frac{1}{p}}$. We also recall that the dual of a $p-$norm (resp. Schatten $p-$norm) is the $q-$norm (resp. the Schatten $q-$norm) with q equal to $\frac{p}{p-1}$ if $p > 1$, to ∞ if $p = 1$ and to 1 if $p = \infty$.

We now define $\mathcal{C}$ as a family of Mahalanobis cost functions, indexed by bounded matrices $\mathbf{M}$:

$$\mathcal{C} = \{c^{\mathbf{M}} : (\boldsymbol{x}, \boldsymbol{y}) \mapsto (\boldsymbol{x} - \boldsymbol{y})^\top \mathbf{M} (\boldsymbol{x} - \boldsymbol{y}); \|\mathbf{M}\|_p \leq 1\}. \tag{3.3}$$

We can now state the following proposition.[1]

Proposition 1. *Let $\mathcal{C}$ be defined as in (3.3) for $\mathbf{M} \in \mathcal{S}_+^{D \times D}$. Then, $\mathcal{C}$ is a convex compact set of cost functions and for any $p \in [1, +\infty], q = \frac{p}{p-1}$ the following holds:*

[1] All detailed proofs are provided in Appendix B.

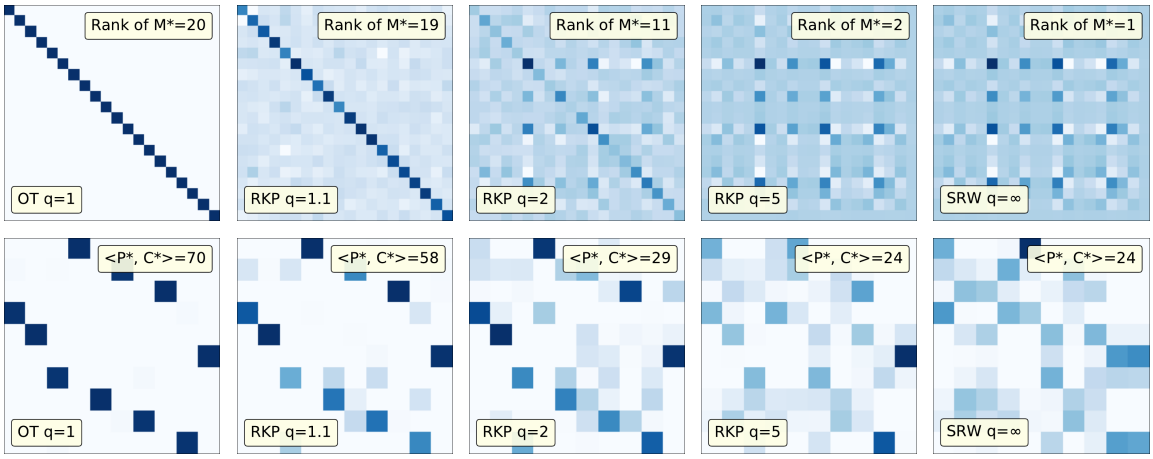


Figure 3.1: *Interpolation between OT and $SRW_{d=1}$ on a binary toy classification problem with each class consisting of 5 points sampled from Gaussians centered on the edge of a 10-dimensional hypercube with $\sigma = 1$ with 10 additional random noise features. The transport is computed between the 2 classes using the setting of Proposition 1 with $q \in \{1, 1.1, 2, 5, \infty\}$.* ***(top row)*** *Mahalanobis matrices $\mathbf{M}^*$ and their rank;* ***(bottom row)*** *Couplings $\boldsymbol{T}^*$ and the associated value of the Wasserstein distance.*

Cited on pages [51,57,102,182,182]

1. $\mathrm{RKP}(\Pi, \mathcal{C}) = \min_{\pi \in \Pi} \|\mathbf{V}_\pi\|_q$. *In particular, we have:*

$$\mathrm{RKP}(\Pi, \mathcal{C}) = \begin{cases} \mathcal{W}_2^2(\mu, \nu), & \text{if } q = 1, \\ \mathcal{S}_1^2(\mu, \nu), & \text{if } q = \infty. \end{cases}$$

2. *For any $\pi \in \Pi$, $\|\mathbf{M}^*\|_p = 1$ and*

$$\mathbf{M}^* = \operatorname*{argmax}_{\mathbf{M} \in \mathcal{S}_+^{D \times D},\ \|\mathbf{M}\|_p \leq 1} \langle \mathbf{V}_\pi, \mathbf{M} \rangle = \left(\frac{\mathbf{V}_\pi}{\|\mathbf{V}_\pi\|_q} \right)^{\frac{q}{p}}.$$

Proof idea. We use the fact that $\mathcal{C}$ is the image of a convex compact set of $\mathbb{R}^{D \times D}$ by a linear mapping to prove its convexity and compactness. Point 1 is a consequence of the equality case of Hölder's inequality, the positive semi-definiteness of matrix $\mathbf{V}_\pi$ and the fact that the Schatten p-norm is the classic p-norm for the vector of a matrix's singular values, which tends to the $\infty-$norm as $q \to \infty$. The second point is a direct consequence of the equality case of Hölder's inequality for Schatten p-norms (Magnus, 1987) using the fact that $\mathbf{V}_\pi$ is PSD. □

This theorem highlights several novel insights. First, it provides a different point of view for a general minimax OT problem with the infinite family of Mahalanobis distances. In particular, it shows that the original OT problem can be seen as a minimax problem when one takes the least restrictive infinity norm for the bound on the matrix parameterizing the Mahalanobis distance, while SRW with $d = 1$ corresponds to the case of the $||\cdot||_1$ norm[2]. This observation

[2]Other values of d for SRW are also covered when using a truncated Schatten p-norm.

is illustrated in Figure 3.1 where we smoothly interpolate between the two boundary cases by solving (3.2) with intermediate values of q. We note that such an interpolation may have interesting implications in practice when one seeks for an explicit control between the original and the minimax OT problems. Second, the optimal expression for $\mathbf{M}^*$ shows that it is proportional to $\mathbf{V}_\pi$ and if this latter captures the displacement in lower dimensions, then $\mathbf{M}^*$ is expected to do so too. This follows from $\mathbf{M}^*$ being a linear combination of $(\boldsymbol{x}_i - \boldsymbol{y}_k)(\boldsymbol{x}_i - \boldsymbol{y}_k)^\top$, where $\{i, k\}$ are indices for which $\pi_{ik} > 0$, making its image included in the span of $\{(\boldsymbol{x}_i - \boldsymbol{y}_k); \pi_{ik} > 0\}$, i.e., the span of displacement directions. This intuition is confirmed in our experiments where we show that even without the rank constraint, solving (3.2) results in a matrix of a reduced rank.

Finally, below we use this result to show that in the case of the Frobenius norm, the PSD property of the learned matrix $\mathbf{M}$ is obtained for free without imposing any additional constraint on the set $\mathcal{C}$.

Corollary 1 (Euclidean norm case)**.** *Let $\mathcal{C}$ be defined with $p = 2$ in* (3.3) *and let* $\mathbf{M}^* = \operatorname{argmax}_{\|\mathbf{M}\|_2 \leq 1} \langle \mathbf{V}_\pi, \mathbf{M} \rangle$. *Then* $\mathbf{M}^* = \dfrac{\mathbf{V}_\pi}{\|\mathbf{V}_\pi\|_2}$, *thus* $\mathbf{M}^*$ *is PSD and* $\|\mathbf{M}^*\|_2 = 1$.

This corollary shows that the case $p = 2$ (Frobenius norm) can be very convenient in practice as PSD constraints increase considerably the computational burden of any optimization problem, yet they are necessary for the obtained cost function to be a true metric.

To conclude the theoretical analysis of the considered case for the minimax problem, we establish a general bound on $\mathrm{RKP}(\Pi, \mathcal{C})$ in terms of the original 2-Wasserstein distance.

Corollary 2. *With the assumptions from Proposition 1, the following inequality holds for any $p \in [1, +\infty]$:*

$$\frac{1}{D^{\frac{1}{p}}} \mathcal{W}_2^2(\mu, \nu) \leq \mathcal{W}_{\mathcal{C}}(\mu, \nu) \leq \mathcal{W}_2^2(\mu, \nu).$$

Note that compared to a similar bound given in (Paty and Cuturi, 2019a, Proposition 2) for the SRW distance, our result does not involve the k term in the left-hand side as we do not impose any explicit constraint on the rank of $\mathbf{M}$.

Finite set of cost functions

Let $\{c_1, ..., c_M\}$ denote a family of candidate cost functions, and let $\mathcal{C} = \mathrm{Conv}\left(\{c_1, ..., c_M\}\right)$ meaning that $\mathcal{C}$ is a convex compact space as it is the convex combination of a finite set. As mentioned in Section 3.2, the optimization of the OT problem with a submodular function F taken as a cost function can be equivalently seen as a minimax OT problem of the following form:

$$\min_{\boldsymbol{T} \in \Pi} \max_{\mathbf{C} \in \mathcal{B}_F} \langle \boldsymbol{T}, \mathbf{C} \rangle,$$

where $\mathcal{B}_F$ is the base polytope of F. We note that the number of vertices of $\mathcal{B}_F$ is finite and thus one can show that the StrOT distance is a particular case of our problem (3.2) when $\mathcal{C}$ is a

finite set of cost functions, i.e.,

$$\mathrm{RKP}(\Pi, \mathrm{Conv}\,(\mathcal{B}_F)) = \mathrm{StrOT}(\mu, \nu).$$

This result establishes the link between our general formulation and that considered in (Alvarez-Melis et al., 2018).

3.3.3 Proposed optimization strategy

We now propose a general solution for optimising (3.2) in the discrete case where $\mathcal{X}$ and $\mathcal{Y}$ are identified respectively with finite sets $(\boldsymbol{x}_i)_{i\in[\![1,I]\!]}$ and $(\boldsymbol{y}_k)_{k\in[\![1,K]\!]}$, while $\mathcal{C}$ is identified with an arbitrary convex set of cost matrices with entries $\mathbf{C}_{ik} = c(\boldsymbol{x}_i, \boldsymbol{y}_k)$. Since $\mathcal{X}$ and $\mathcal{Y}$ are finite, hence bounded, all results from Section 3.3.2 hold in the discrete case.

To proceed, we first note that in our case we cannot apply the optimization techniques used in (Paty and Cuturi, 2019a; Alvarez-Melis et al., 2018) as they both consider the differentiable regularized OT problem in their minimax formulations contrary to our non-differentiable unregularized one. To deal with the latter, we propose to adapt the cutting set method presented in (Mutapcic and Boyd, 2009) for robust optimization to Problem (3.2) that allows us to cover both unregularized and regularized minimax OT problems. In a nutshell, this method consists in alternating between solving a worst-case problem and the corresponding sampled robust minimization problem w.r.t. a set of constraints that grows linearly with iterations and requires for optimized functions to be convex only. In application to Equation (3.2), the high level idea of the proposed algorithm thus would be to solve the maximization problem over $\mathcal{C}$ w.r.t. a small set $\mathcal{P} \subset \Pi$ and add one transportation matrix to $\mathcal{P}$ at each iteration. The implementation of this idea, however, is not straightforward and requires two obstacles to be addressed. First, the original algorithm presented by the authors allows to solve a minimax problem of the form $\min_{\mathbf{C}\in\mathcal{C}} \max_{\boldsymbol{T}\in\Pi} = -\max_{\mathbf{C}\in\mathcal{C}} \min_{\boldsymbol{T}\in\Pi}$ and thus requires from us to prove $\min_{\boldsymbol{T}\in\Pi} \max_{\mathbf{C}\in\mathcal{C}} = \max_{\mathbf{C}\in\mathcal{C}} \min_{\boldsymbol{T}\in\Pi}$ in order to apply it. Second, and similar to the projected supergradient algorithm proposed for SRW, the authors of (Mutapcic and Boyd, 2009) disregard the optimal solution for the variable over which the minimization is performed, i.e., , $\boldsymbol{T}^*$ in our case, and provide a solution for $\mathbf{C}^*$ only. To address these issues, we now present the following result.

Proposition 2. *Let $\mathcal{P}$ be a finite subset of Π. Then, the following holds:*

1. $\mathrm{RKP}(\mathcal{P}, \mathcal{C}) := \mathrm{RKP}(\mathrm{Conv}\,(\mathcal{P}), \mathcal{C})$ *has a saddle point* $(\boldsymbol{T}^*, \mathbf{C}^*)$ *verifying:*

$$\langle \boldsymbol{T}^*, \mathbf{C}^* \rangle_F = \min_{\boldsymbol{T}\in\mathrm{Conv}(\mathcal{P})} \max_{\mathbf{C}\in\mathcal{C}} \langle \boldsymbol{T}, \mathbf{C} \rangle = \max_{\mathbf{C}\in\mathcal{C}} \min_{\boldsymbol{T}\in\mathcal{P}} \langle \boldsymbol{T}, \mathbf{C} \rangle . \tag{3.4}$$

2. $\mathrm{RKP}(\mathcal{P}, \mathcal{C})$ *is equivalent to*

$$\begin{aligned} \mathbf{C}^* &\in \operatorname*{argmax}_{\mathbf{C}\in\mathcal{C}, \omega\geq 0} \ \omega, \\ s.t.\ \langle \boldsymbol{T}, \mathbf{C} \rangle &\geq \ \omega, \ \forall \boldsymbol{T} \in \mathcal{P}. \end{aligned} \tag{3.5}$$

3. $\boldsymbol{T}^* = \sum_{l=1}^{|\mathcal{P}|} \mathbf{q}_l \boldsymbol{T}_l$, *where* $\mathbf{Q} = (\mathbf{q}_l)_{l \in [\![1,|\mathcal{P}|]\!]}, \sum_{l=1}^{|\mathcal{P}|} \mathbf{q}_l = 1$, *are dual variables of Equation* (3.5).

Proof idea. Point 1 is an application of Sion's minimax theorem (Sion, 1958). Point 2 is a reformulation of the right hand side of Equation (3.4). The last point, $\boldsymbol{T}^*$'s expression, is a consequence of the Lagrange duality. □

Several remarks are in order here. First, we note that solving Problem (3.5) directly is intractable in practice for sufficiently large I and K as its number of constraints (size of $\mathcal{P}$) grows extremely fast with the number of points (e.g., equal to $N!$ for $I = K = N$). This motivates the use of the cutting set algorithm that gradually increases the size of the set $\mathcal{P}$ with iterations and allows to solve intermediate problems with a reduced number of constraints efficiently. Second, the theorem is valid for any finite subset $\mathcal{P}$ of Π so that 1) solving $\mathrm{RKP}(\Pi, \mathcal{C})$ can be done by setting $\mathcal{P}$ to the set of vertices of Π and 2) solving the regularized minimax formulation with added convex regularizer on $\boldsymbol{T}$ is covered by considering $\mathrm{RKP}(\tilde{\Pi}, \mathcal{C})$, where $\tilde{\Pi}$ is a convex compact subset of Π (Cuturi, 2013).

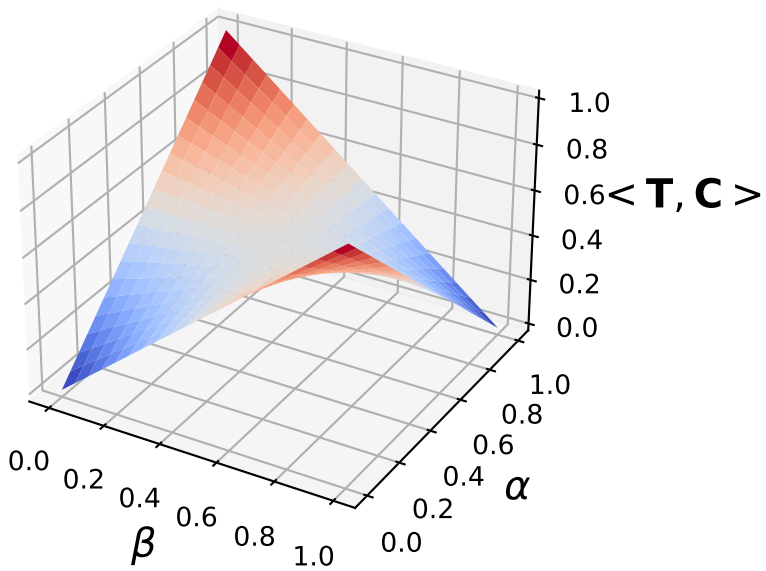


Figure 3.2: *Illustration of a saddle point on a minimax problem. The scalar product between the transport plan and the cost matrix is displayed on the third axis. The α axis allows to move from $\mathbf{C}_1$ to $\mathbf{C}_2$. Similarly the β axis allows to move from $\boldsymbol{T}_1^*$ to $\boldsymbol{T}_2^*$. In both axes, all the possible matrices are represented.*

Cited on page [55]

Third, contrary to the original OT problem, the optimal transport plan $\boldsymbol{T}^*$ is often a convex combination of the edge point of the polytope. We can clearly see such behavior on a very simple example. We consider uniform marginals and a finite set of cost matrices: let $\mathbf{C}_1$ and $\mathbf{C}_2$

be two cost matrices defined as,

$$\mathbf{C}_1 = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}, \mathbf{C}_2 = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}. \tag{3.6}$$

With either $\mathbf{C}_1$ or $\mathbf{C}_2$, we obtain to a Wasserstein distance of 0, with $\boldsymbol{T}_1^*$ and $\boldsymbol{T}_2^*$ defined as,

$$\boldsymbol{T}_1^* = \begin{pmatrix} 0 & 0.5 \\ 0.5 & 0 \end{pmatrix}, \boldsymbol{T}_2^* = \begin{pmatrix} 0.5 & 0 \\ 0 & 0.5 \end{pmatrix}. \tag{3.7}$$

As shown on Figure 3.2, if we note the convex set generated by the two cost matrices $\alpha\mathbf{C}_1 + (1-\alpha)\mathbf{C}_2$, the best value is obtained for $\alpha = 0.5$. Similarly, the best choice for the optimal transport plan is $\boldsymbol{T}^* = \beta\boldsymbol{T}_1^* + (1-\beta)\boldsymbol{T}_2^*$ with $\beta = 0.5$. Thus, the third point in Proposition 2 is key to have access to the exact transport plan $\boldsymbol{T}^*$ as it is (often) not simply on the edge of the polytope.

Algorithm 4 Cutting set method for RKP$(\Pi, \mathcal{C})$ with constraint elimination. Cited on pages [59,59,113]

Input: maxIt, $\mathcal{C}$, $\mathcal{P}_0 \subset \Pi$, thd1, thd2
$t, l \leftarrow 0$
$err, \omega_{-1} \leftarrow \infty$
while $t <$ maxIt **and** $err >$ thd1 **and** $\frac{\omega_{t-1}-\omega_t}{\omega_{t-1}} >$ thd1^2 **do**
    Solve (3.5) to obtain $(\omega_t, \mathbf{C}_t), \mathbf{Q}$
    for l in $\{0, ..., |\mathcal{P}_t| - 1\}$ **do**
        if $\mathbf{q}_l \leq$ thd2 **then**
            $\mathcal{P}_t \leftarrow \mathcal{P}_t \setminus \{\boldsymbol{T}_l\}$
            $\mathbf{Q} \leftarrow \mathbf{Q} \setminus \{\mathbf{q}_l\}$
        end if
    end for
    Find $\boldsymbol{T}_t \in \operatorname{argmin}_{\boldsymbol{T} \in \Pi} \langle \boldsymbol{T}, \mathbf{C}_t \rangle$
    $l \leftarrow \max(l, \langle \boldsymbol{T}_t, \mathbf{C}_t \rangle)$
    $err \leftarrow (\omega_t - l)/l$
    $\mathcal{P}_{t+1} = \mathcal{P}_t \cup \{\boldsymbol{T}_t\}$
    $t \leftarrow t + 1$
end while
return $\sum_{l=0}^{|\mathcal{P}_t|-1} \mathbf{q}_l \boldsymbol{T}_l$, $\mathbf{C}_t$

Our final algorithm, inspired by (Mutapcic and Boyd, 2009, Section 5.1), takes advantage of Proposition 2 and boils down to alternately performing the following two steps for $t \in \{0, \ldots, \text{maxIt}\}$:

Step 1. Find $\mathbf{C}_t$ solving (3.5) over $(\mathcal{P}_t, \mathcal{C})$, where $\mathcal{P}_t$ is a finite subset of Π; let ω_t be the value at the solution.

Step 2. For a fixed matrix $\mathbf{C}_t$ obtained at **Step 1**, find $\boldsymbol{T}_t \in \operatorname{argmin}_{\boldsymbol{T}\in\Pi} \langle \boldsymbol{T}, \mathbf{C}_t \rangle$.

Step 2 of each iteration can make use of any efficient algorithm for solving the classic unregularized optimal transport. Empirically, we observed that even the approximate solutions obtained by solving the entropy regularized formulation of the optimal transport problem ensure the convergence. We further use the constraint dropping strategy (Mutapcic and Boyd, 2009, Sec. 5.3.2) and provide a complete pseudo-code for our algorithm in Algorithm 4, where thd1 and thd2 respectively control the stopping criterion and the constraint elimination. The proposed algorithm is generic and can also be used to solve the problems underlying the SRW and StrOT distances seen previously. Moreover, it acts as a meta-algorithm by implicitly choosing (or learning depending on the construction of the set $\mathcal{C}$) the "right" cost function. This differs from other existing methods on learning the cost matrix in the OT framework (Cuturi and Avis, 2014; Zhao and Zhou, 2018) that usually learn this latter using the *a priori* similarity between the histograms.

Finally, Algorithm 4 is guaranteed to converge in a finite number of iterations with the latter being upper-bounded thanks to the following proposition.

Proposition 3. *Let T be the number of iterations required by Algorithm 4 to reach error $err(T) \leq \text{thd1}$. Then,*

$$T \leq \left(\frac{\operatorname{diam}_\infty(\mathcal{C}) + \operatorname{RKP}(\mathcal{P}_0, \mathcal{C})}{2.\text{thd1}} + 1 \right)^{\dim(\mathcal{C})+1}$$

where $\operatorname{diam}_\infty(\mathcal{C}) := \sup_{\mathbf{C}^1,\mathbf{C}^2\in\mathcal{C},i,j} \left|\mathbf{C}^1_{ik} - \mathbf{C}^2_{ik}\right|$ *and* $\dim(\mathcal{C})$ *is the dimension of the affine hull of* $\mathcal{C}$*. Also,* $\forall t \geq 0$*, we have that* $0 \leq \operatorname{RKP}(\mathcal{P}_t, \mathcal{C}) - \operatorname{RKP}(\Pi, \mathcal{C}) \leq err(t)$*.*

Proof idea. We adapt the proof technique presented in (Mutapcic and Boyd, 2009, Section 5.2) to our case, after re-writing the right hand side of Equation (3.4) as

$$\min_{\mathbf{C}\in\mathcal{C}} \max_{\boldsymbol{T}\in\mathcal{P}_t} \left(-\langle \boldsymbol{T}, \mathbf{C} \rangle\right)$$

to make our problem coincide with the authors' formulation. □

This theorem offers interesting insights regarding the convergence speed of the proposed algorithm. First, it introduces the dependence of the latter on $\operatorname{diam}_\infty(\mathcal{C})$, which can be interpreted as a degree of disagreement between the cost matrices in $\mathcal{C}$ so that one may need more iterations to reach precision *err* when they disagree. Second, the presence of the value of the initial nominal problem $\operatorname{RKP}(\mathcal{P}_0, \mathcal{C})$ reflects the influence of the initialization $\mathcal{P}_0$. Finally, when $\mathcal{C}$ lies in a subspace of a much smaller dimension than $I \times K$ (i.e., in case of the Mahalanobis distance, $\mathcal{C}$ is the image of $D \times D$ matrices by a linear mapping, while for the finite number of matrices, $\dim(\mathcal{C})$ is $\dim(\operatorname{span}(\mathbf{C}_1, ..., \mathbf{C}_M)) - 1)$, the algorithm needs much less iterations as highlighted by the presence of $\dim(\mathcal{C})$ in the exponent.

3.3.4 Variations for different choices of $\mathcal{C}$

Below, we express the maximization problem (3.5) over $\mathcal{P}_t \times \mathcal{C}$ at step $t \geq 0$ of Algorithm 4, for both choices of $\mathcal{C}$ considered in Section 3.3.2, in a more convenient way.

Proposition 4 (Finite set $\mathcal{C}$)**.** *Let $\mathcal{C} = \mathrm{Conv}\left(\{\mathbf{C}_1, ..., \mathbf{C}_M\}\right)$. Then, for $t \geq 0$, solving the problem given in Equation* (3.5) *over $\mathcal{P}_t \times \mathcal{C}$ is equivalent to the following linear program*

$$\begin{aligned} &\min_{\mathbf{p} \in \mathbb{R}_+^M} \mathbb{1}_M^\top \mathbf{p} \\ &s.t. \mathbf{G}\mathbf{p} \geq \mathbb{1}_{|\mathcal{P}_t|}, \end{aligned} \tag{3.8}$$

where $\mathbf{G} \in \mathbb{R}^{|\mathcal{P}_t| \times M}$ with $\mathbf{G}_{lm} = \langle \boldsymbol{T}_l, \mathbf{C}_m \rangle$. Moreover,

$$\mathbf{C}^* = \frac{\sum_{m=1}^M \mathbf{p}_m^* \mathbf{C}_m}{\sum_{m=1}^M \mathbf{p}_m^*}, \qquad \boldsymbol{T}^* = \frac{\sum_l^{|\mathcal{P}_t|} \mathbf{q}_l^* \boldsymbol{T}_l}{\sum_l^{|\mathcal{P}_t|} \mathbf{q}_l^*},$$

where $\mathbf{p}^*$ and $\mathbf{q}^*$ are optimal solutions of (3.8) *and its dual.*

For the case of the infinite family of Mahalanobis distances, we propose a more general result that considers the following set of non-centered Mahalanobis distances:

$$\mathcal{C}_{\mathbf{C}} = \{\mathbf{C} + \mathbf{E}^{\mathbf{M}} \in \mathbb{R}^{I \times K} \mid \mathbf{E}_{ik}^{\mathbf{M}} = (\boldsymbol{x}_i - \boldsymbol{y}_k)^\top \mathbf{M} (\boldsymbol{x}_i - \boldsymbol{y}_k); \mathbf{M} \in \mathcal{S}_+^{D \times D}; \|\mathbf{M}\|_p \leq r\}. \tag{3.9}$$

for an arbitrary radius $r > 0$.

Proposition 5 (Non centered family of Mahalanobis distances)**.** *For a fixed $\mathbf{C}$, let $\mathcal{C}_{\mathbf{C}}$ be defined as in Equation* (3.9)*. Then, for $t \geq 0$, solving Equation* (3.5) *over $\mathcal{P}_t \times \mathcal{C}_{\mathbf{C}}$, is equivalent to solving the following convex program,*

$$\min_{\boldsymbol{T} \in \mathrm{Conv}(\mathcal{P}_t)} r \left\|\mathbf{V}_{\boldsymbol{T}}\right\|_q + \sum_{ik} \boldsymbol{T}_{ik} \mathbf{C}_{ik}. \tag{3.10}$$

Moreover, if $\boldsymbol{T}^*$ is an optimal solution of (3.10)*, then $\mathbf{M}^*$ is as in Equation* (1) *with π replaced by $\boldsymbol{T}^*$.*

In the following, we consider the case of $p = 2$ and in this case, by Corollary 1, $\mathbf{M}^*$ is PSD even without imposing such a constraint. We keep $p = 2$ mainly because it is the most common setting and the efficient library CVXPY (Diamond and Boyd, 2016; Agrawal et al., 2018), used to solve Equation (3.10) at each iteration, covers only the Schatten q-norm for $q = 2$ or $q = 1$. For the other values of q we use a Frank Wolfe algorithm (Frank et al., 1956) to solve Equation (3.10) at each iteration. We can see some results with such a method on Figure 3.1.

3.3.5 Towards the notion of stability of cost matrices

In this section, we define a new notion of OT stability for a cost matrix $\mathbf{C}$ based on a non-centered convex set $\mathcal{C}_{\mathbf{C}}$.

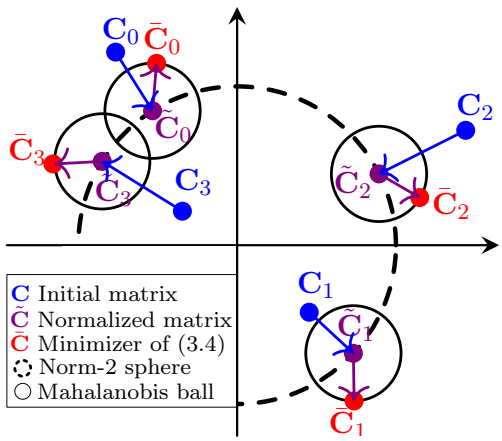


Figure 3.3: *Illustration of the notion of matrix cost stability. Every matrix* $\mathbf{C}_m$ *is normalized so as to get a matrix* $\tilde{\mathbf{C}}_m$ *which lies on the norm-2 sphere.* $\bar{\mathbf{C}}_m$ *is the minimizer of Problem 3.4. The stability (Definition 1) comes from the difference of the cost transports induced by* $\bar{\mathbf{C}}_m$ *and* $\tilde{\mathbf{C}}_m$*.*

Cited on page [58]

Definition 1. *For a cost matrix* $\mathbf{C}$ *and its associated convex set* $\mathcal{C}_{\mathbf{C}}$ *introduced in* (3.9)*, for some* $r > 0$*, we define the stability* $\mathcal{WS}_{\mathbf{C},r}$ *as follows:*

$$\begin{aligned}\mathcal{WS}_{\mathbf{C},r} =& \mathcal{W}_{\mathcal{C}_{\mathbf{C}}}(\mu,\nu) - \mathcal{W}_{\mathbf{C}}(\mu,\nu)\\ =& \min_{\boldsymbol{T}\in\Pi} \max_{\|\mathbf{M}\|\leq r} \left\langle \boldsymbol{T}, \mathbf{C} + \mathbf{E}^{\mathbf{M}} \right\rangle - \min_{\boldsymbol{T}\in\Pi} \left\langle \boldsymbol{T}, \mathbf{C} \right\rangle .\end{aligned}$$

Roughly speaking, Definition 1 tells us that the Wasserstein distance between μ and ν associated with a stable cost matrix $\mathbf{C}$ should not differ much from the Wasserstein distance calculated based on the worst cost matrix in the neighborhood of $\mathbf{C}$. Note that the latter is defined as a Mahalanobis ball allowing us to define the stability of $\mathbf{C}$ w.r.t. the finite sets $\mathcal{X}$ and $\mathcal{Y}$. To be able to compare different stabilities for a family of cost matrices $(\mathbf{C}_m)_{m\in[\![1,M]\!]}$, we normalize each $\mathbf{C}_m$ either by diving its elements by its Frobenius norm or by the associated transport cost $\mathcal{W}_{\mathbf{C}_m}(\mu,\nu)$. Figure 3.3 illustrates the intuition behind the notion of cost matrix stability where the Frobenius norm is used for the normalization.

3.4 Experiments

In this section, we first illustrate our algorithm's speed of convergence and compare it to solving the original LP problem from (3.5). Then, we reproduce a simulated problem from (Paty and Cuturi, 2019a) to assess the algorithm's ability to correctly identify the subspace of a lower dimensionality in which the transformation between the two samples lies. In what follows, we concentrate on comparing our approach with the authors' implementation of SRW while leaving aside the comparison with StrOT for which the implementation is not publicly available. The second part of our experiments is related to the notion of stability defined in Section 3.3.5. We first bring to light a correlation between the stability and the noise resistance of a cost

matrix. Then, we show that selecting the most stable matrix allows to efficiently transport colors between two images in a color transfer task. The code for the different experiments is available on this link[3].

3.4.1 Convergence and execution time

We consider the case where $\mathcal{C}$ is the set of convex combinations of a given number of cost matrices denoted as $|\mathcal{C}|$. The convergence of Algorithm 4 is illustrated in Figure 3.4 (left) by plotting the evolution of the quantity $err(t) := |\omega_t - \langle \boldsymbol{T}_t, \mathbf{C}_t \rangle|$ along the iterations for $|\mathcal{C}| \in \{10, 40, 90\}$. From this plot, we see that the convergence becomes slower as $|\mathcal{C}|$ grows, which is expected because ω_t is the value at the solution of Problem (3.5) over $\mathrm{Conv}\,(\mathcal{P}_t) \times \mathcal{C}$. Second, for $|\mathcal{C}| = 10$, Algorithm 4 already achieves an error $err(t) \leq 10^{-10}$ after $t = 100$ iterations. This confirms that $\mathcal{P}_t$ does not have to grow until it becomes the whole set of vertices of Π, as $|\mathcal{P}_{100}| \leq |\mathcal{P}_0| + 100 \ll N! = 100!$. We also test our algorithm with the entropic regularization of the transport matrix with $\epsilon \in \{1, 0.1, 0.01\}$ as regularization parameter, using Sinkhorn algorithm (Cuturi, 2013) for $|\mathcal{C}| = 40$. For this setting, we initialize it with $\mathcal{P}_0 = \boldsymbol{a}\boldsymbol{b}^\top$ for any $\epsilon > 0$, as this set $\mathcal{P}_0$ is included in the feasible set of entropy-regularized transport (as suggested in the discussion of Proposition 2). Interestingly, we have noticed that the algorithm does not converge if $\mathcal{P}_0$ is a subset of the vertices of transportation polytope Π in the regularized case. The results of this experiment are reported in Figure 3.4 (middle), where we observe the convergence even with the entropy regularization. Additionally, we note that due to the linearity of the mapping $\langle \cdot, \boldsymbol{T} \rangle$ for all $\boldsymbol{T} \in \Pi$, Problem (3.2) can be reformulated as the following LP:

$$\begin{aligned} &\min_{\boldsymbol{T} \in \Pi, \eta \geq 0} \ \eta, \\ \text{s.t.} \ &\langle \boldsymbol{T}, \mathbf{C}_l \rangle \leq \eta \qquad \forall 1 \leq l \leq d. \end{aligned}$$

It turns out that this is nothing more than the dual of Problem (3.5). Under this formulation, solving $\mathrm{RKP}(\Pi, \mathcal{C})$ becomes tractable for $m = n = 100$ and $|\mathcal{C}| \in \{10, 20, ..., 90\}$ and allows us to compare the execution time of solving the LP problem to that of our algorithm in Figure 3.4 (right). As the number of candidate matrices grows, our algorithm becomes much more efficient than solving the full LP problem. This is rather expected since at each iteration, it solves a linear program with much less constraints (the problem is restricted to $\mathrm{Conv}\,(\mathcal{P}_t) \times \mathcal{C}$ instead of $\Pi \times \mathcal{C}$) and it leverages efficient algorithms for solving the OT problem (1.6).

3.4.2 Comparison to SRW

In this series of experiments, we consider the fragmented hypercube dataset studied in (Paty and Cuturi, 2019a) and earlier in (Forrow et al., 2019) and compare RKP to both the SRW and the Wasserstein distances. To proceed, let $(\mathbf{e}_l)_{l \in [\![1,D]\!]}$ be the canonical basis of $\mathbb{R}^D$ and let $(\boldsymbol{x}_i)_{i \in [\![1,I]\!]}$ and $(\boldsymbol{y}_k)_{k \in [\![1,K]\!]}$ be two finite sets drawn i.i.d. from the uniform distribution over

[3] `https://github.com/sofiendhouib/minimax_OT`.

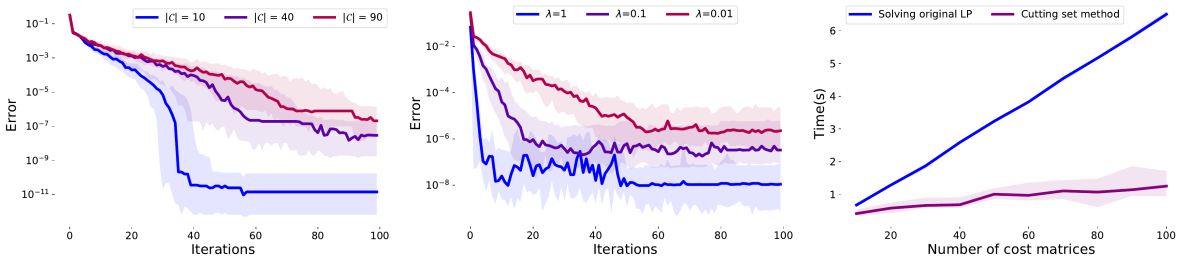


Figure 3.4: ***(left)*** *Evolution of the error along the iterations for* $|\mathcal{C}| \in \{10, 40, 90\}$*;* ***(middle)*** *Evolution of the error with a regularization parameter* $\epsilon \in \{1, 0.1, 0.01\}$*;* ***(right)*** *Execution time of our algorithm vs solving the original LP problem with* $|\mathcal{C}| \in \{10, 20, ..., 90\}$ *and* $n = m = 100$*. The experiments are repeated 30 times. The median and the interval between the first and third quartiles are reported.*

Cited on pages [59,59,59]

the D-dimensional hypercube $\mathcal{U}([-1, 1]^D)$ and its pushforward distribution under the mapping $f : \boldsymbol{x} \mapsto \boldsymbol{x} + 2\,\mathrm{sgn}\,(x) \odot (\sum_{r=1}^{d} \mathbf{e}_r)$, where $\odot$ denotes elementwise multiplication and $d \in [\![1, D]\!]$, respectively. Therefore, by construction, there are d relevant features and $D - d$ features that contain no useful information. Depending on the choice of $\mathcal{C}$, two cases of our algorithm are tested: 1) squared Euclidean distance after projecting on all combinations of two vectors of the canonical basis $\mathcal{C} = \{\mathbf{C}_{s,l} \in \mathbb{R}^{I \times K} | (\mathbf{C}_{s,l})_{ik} = ((\boldsymbol{x}_i - \boldsymbol{y}_k)^\top (\mathbf{e}_s + \mathbf{e}_l))^2; 1 \leq s < l \leq D\}$ and 2) the Mahalanobis ball centered at 0 as defined in Section 3.3.2. Note that in this latter case $\mathrm{rank}(\mathbf{M}^*) = d$.

Figure 3.5 (left) reproduces the experiments of (Paty and Cuturi, 2019a) and shows that the original OT (bottom left) is sensitive to noise, while both SRW and RKP (for the 2 configurations considered) are able to recover the true pushforward transformation. However, while SRW requires a hyperparameter d to constrain the rank of the Mahalanobis matrix, our method is parameter-free since d is found automatically as illustrated in Figure 3.5 (middle). In this figure, we plot the eigenvalues of $\mathbf{M}^*$ for different values of d and observe that the eigengap between the d largest eigenvalues and the $(d + 1)^{th}$ eigenvalue clearly reveals that $\mathrm{rank}(\mathbf{M}^*) = d$.

3.4.3 Stability and noise sensitivity

Below, we illustrate the correlation between the cost matrix stability and the sensitivity of the Wasserstein distance to the presence of noise using both toy and a real-world datasets. The latter one is composed of 100 zeros and 100 ones coming from the MNIST dataset, after reducing its dimensionality to 10 with UMAP (McInnes et al., 2018). The former consists of 100 points drawn from two 10-dimensional Gaussian distributions centered at $\mathbf{0}_{10}$ and $3 \times \mathbf{1}_{10}$ respectively with unit variance. For both datasets, we generate a family of cost matrices $(\mathbf{C}_m)_{m \in [\![1,50]\!]}$ based on random Mahalanobis distances with different norms, normalize them so that their Frobenius norm equals 1 and compute $\mathcal{WS}_{\mathbf{C}_m, r=0.01}$ from Definition 1 for all m. To introduce noise to each $\mathbf{C}_m$, we add a random Mahalanobis cost matrix $\mathbf{E}^{\mathbf{N}}$ with $\|\mathbf{N}\|_2 = r$ to it and compute the

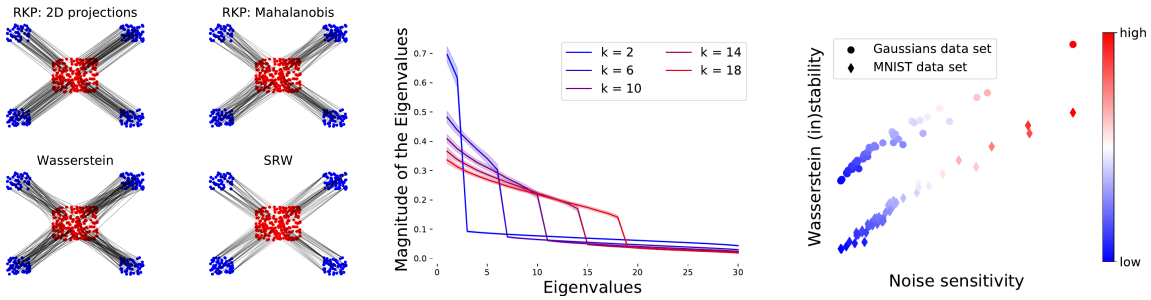


Figure 3.5: ***(left)*** *Results obtained on the fragmented hypercube for $I = K = 250$, $D = 30$ and $d = 2$ with* ***(top row)*** *our approach with 2D projections and Mahalanobis distances;* ***(bottom row)*** *Original OT problem and SRW method of (Paty and Cuturi, 2019a);* ***(middle)*** *Sorted eigenvalues of $\mathbf{M}^*$ obtained using RKP averaged over 100 runs for different values of k reveals a phase transition between d dominant and the $d+1$ eigenvalues;* ***(right)*** *Correlation between the stability and the sensitivity to noise.*

Cited on pages [60,60,61]

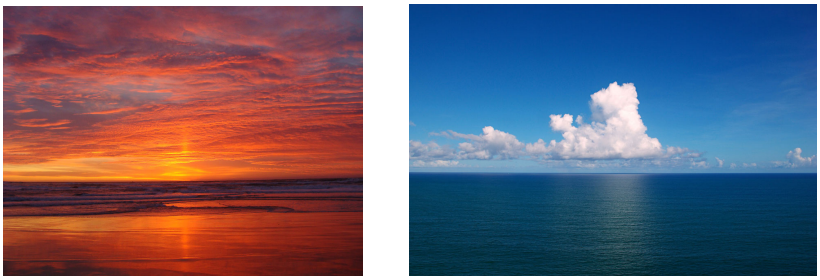


Figure 3.6: *Source (ocean) and target (sky) images considered as probability distributions.*

Cited on page [61]

noise sensitivity defined as:

$$\mathcal{NS}_{\mathbf{C}_m} = \left| \min_{\boldsymbol{T} \in \Pi} \langle \boldsymbol{T}, \mathbf{C}_m \rangle - \min_{\boldsymbol{T} \in \Pi} \left\langle \boldsymbol{T}, \mathbf{C}_m + \mathbf{E}^{\mathbf{N}} \right\rangle \right| .$$

Note that we apply a Mahalanobis noise which has the advantage of taking into account the point distributions and can be applied on any matrix $\mathbf{C}_m$. Figure 3.5 (right) presents the results of this experiment averaged over 200 runs and shows a clear correlation between the stability and noise sensitivity indicating that the most stable matrices are more noise tolerant. Other experiments on the MNIST dataset provided in Appendix B show a similar behavior.

3.4.4 Color transfer

In this last experiment, we show how we can benefit from the notion of stability to address a color transfer task where the goal is to transfer the colors from a blueish sky image to the reddish ocean image shown in Figure 3.6. Here, we use OT between the sets of pixels in the RGB space extracted from both images. For the sake of efficiency, we consider only 200 pixels from each image and generalize the obtained OT mapping to the remaining pixels following

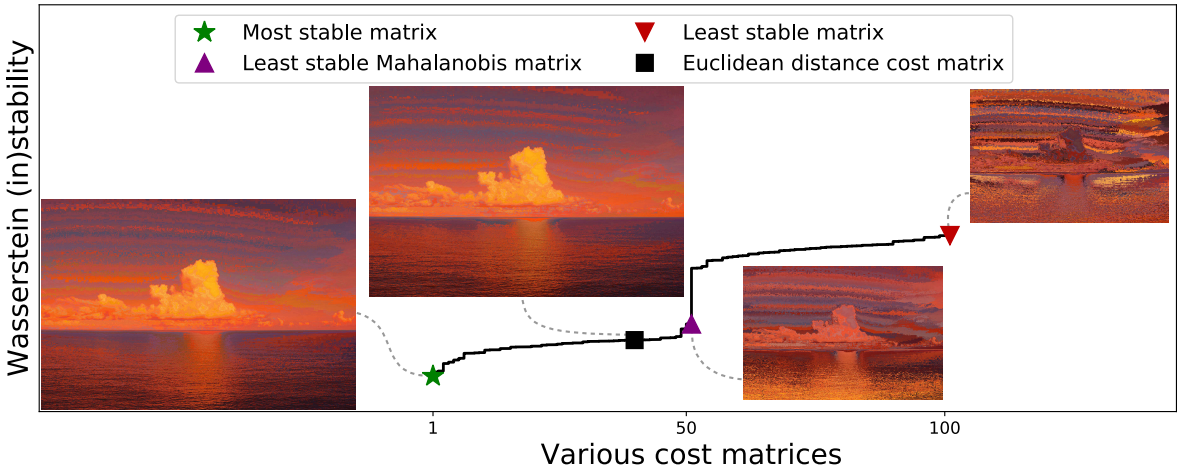


Figure 3.7: *Cost matrices sorted by Wasserstein stability. The first 50 are Mahalanobis cost matrices, while the last 50 are random cost matrices.*

Cited on pages [61,61]

the method detailed in (Ferradans et al., 2014). As before, we use $(\mathbf{C}_m)_{m\in[\![1,50]\!]}$ as meaningful cost matrices and add 50 completely random matrices that are unrelated to the considered task. The results of this experiment given in Figure 3.7 show a significant gap in terms of stability between the Mahalanobis matrices (the first 50 matrices on the x-axis) and the random ones (the last 50). This tends to highlight the fact that the stability can be used as a criterion to select a good cost matrix, and therefore to induce a relevant Mahalanobis distance. This also holds in terms of visual perception as illustrated in Figure 3.7. Even if the most stable matrix is visually very similar to the Euclidean one, a finer evaluation reveals more discontinuities in the center of the picture, on the water.

3.5 Conclusion

In this chapter, we studied a general formulation of the minimax OT problem that consists in optimizing over the coupling matrix w.r.t. the worst cost function from a certain convex set of cost functions. When the latter is given by an infinite family of Mahalanobis distances, we highlight the relation of the considered problem with the existing formulations and characterize the different features of its solutions. We further showed how the underlying optimization problem can be solved in practice using a variation of a cutting set algorithm with theoretical guarantees regarding its convergence speed. Finally, we defined a new notion of stability for cost matrices in OT based on the studied minimax problem and revealed a correlation between this stability and the noise resistance of the matrices. This leads to a criterion that can be used to select a relevant cost function which has been shown to be efficient on both toy and real-world data. A promising line of research might be to find the most stable cost matrix from a continuous set.

Chapter 2 and Chapter 3 both proposed a way to find a good cost matrix for the OT problem. In the next chapter, we focus on the Gromov-Wasserstein (GW) extension. Before learning any loss function for the GW problem, we can notice that the existing solvers are inefficient to approximate the GW distance with complex losses. The next chapter aims to fill this gap.

Chapter 4

Sampled Gromov Wasserstein

Abstract

The two previous chapters focused on providing a good ground metric for the standard Optimal Transport setting. This chapter addresses the Gromov Wasserstein (GW) problem as an extension of the Wasserstein distance to incomparables spaces. It is based on the paper " Sampled Gromov Wasserstein" published in the Machine Learning Journal and presented at the ECML-PKDD 2021 conference (Kerdoncuff et al., 2021). To deal with probability measures lying in different spaces, the Gromov Wasserstein distance, presented in Section 1.1.3, only considers intra-distribution pairwise (dis)similarities. However, for two (discrete) distributions with N points, the state of the art solvers have a $O\left(N^4\right)$ time complexity when using an arbitrary loss function, making most of the real world problems intractable. In this chapter, we introduce a new iterative way to approximate GW, called *Sampled Gromov Wasserstein*, which adapts the stochastic Frank-Wolfe algorithm to the GW case. This simple idea, supported by theoretical convergence guarantees, comes with a $O\left(N^2\right)$ solver. A special case of Sampled Gromov Wasserstein, which can be seen as the natural extension of the well known Sliced Wasserstein to distributions lying in different spaces, reduces even further the complexity to $O\left(N\log(N)\right)$. This chapter ends with experiments on synthetic and real datasets.

4.1 Introduction

Even though the square Euclidean distance is used most of the time to compare points of the distributions in an OT problem, we have seen in the two previous chapters that various other ground metrics can be naturally used or learned to better capture the idiosyncrasies of the application at hand: the Mahalanobis distance (Paty and Cuturi, 2019a), the Earth mover's distance in computer vision tasks, or concave functions in economy such as the square root of the Euclidean distance (Delon et al., 2012), etc. Whatever the cost function, it is worth noting that the OT problem has been originally formulated so as to deal with distributions that are required to lie in the same space. To relax this constraint, we have seen in Chapter 1 a distance between

metric spaces, named Gromov Wasserstein (GW), has been introduced in (Memoli, 2007). It takes the form of the generalization of the well-known Quadratic Assignment problem (Beckman and Koopmans, 1957) with any distribution (Mémoli, 2011) and any loss function (Peyré et al., 2016). The intuition is still to align points between two distributions but the method only relies on pairwise distances, in each space separately. This allows notably to take into account the structure of each distribution while being invariant to rotation and translation.

From an algorithmic perspective, most of the methods used to solve the GW problem resort to the entropic approximation (EGW) of the original GW formulation introduced in (Peyré et al., 2016) and based on a projected Mirror Descent according to the Kullback Leibler divergence. While a naive implementation of the original GW problem leads to a $O\left(N^4\right)$ complexity, Peyré et al. (2016) further show that one can compute GW in $O\left(N^3\right)$ operations for a certain class of losses. Some other attempts have been recently proposed in the literature to speed-up the GW calculation. Sliced Gromov-Wasserstein (SGW) (Vayer et al., 2019b) takes inspiration from the Sliced Wasserstein distance (Rabin and Peyré, 2011) by projecting each distribution in a 1D line and then solving the 1D Gromov-Wasserstein problem efficiently in $O\left(N\log(N)\right)$. The Anchor Energy (AE) distance from Sato et al. (2020) , is also related to the GW distance but simplifies the problem into N^2 linear sub-problems and thus does not approximate the GW distance. The overall time complexity for solving AE is $O\left(N^2\log(N)\right)$. Scalable Gromov-Wasserstein Learning (S-GWL) (Xu et al., 2019a) decomposes recursively the two large probability measures into a set of small pairwise aligned distributions using a common Gromov-Wasserstein barycenter (Peyré et al., 2016). The final transport plan of S-GWL is the aggregation of the result of GW on each small aligned distributions. Instead of using a GW barycenter, (Blumberg et al., 2020) simply uses a classical partition clustering on the points in a vector space. A recent variant of this algorithm, proposed by Chowdhury et al. (2021), uses a 1D Optimal Transport method to approximate the alignment on each small distribution instead of relying on the GW distance. As those three methods are highly related, we will only adapt S-GWL to any loss in the experimental part.

In this chapter, we aim at overcoming the main algorithmic bottleneck of EGW: the multiplication of a 4D tensor with a 2D matrix, which we interpret as an expectation over matrices. We leverage this interpretation, using sampling to approximate the expectation instead of computing it entirely, reducing the complexity to $O\left(N^2\right)$. Unlike SGW and AE which propose simplified distances, we optimize the *original GW distance*. Unlike EGW and S-GWL which have speedups for specific loss functions, we lower the complexity with *any loss function*. We obtain a generic algorithm, called *Sampled Gromov Wasserstein*, supported by theoretical convergence guarantees. We further show that when the number of sampled matrices is 1, the particular 1D case of the OT can be used to compute an update in $O\left(N\log(N)\right)$. This version, called *Pointwise Gromov Wasserstein*, overcomes most of the limitations of SGW (Vayer et al., 2019b) detailed in Section 4.3, while still being very fast. Our contributions are supported by experiments on synthetic and real datasets. Interestingly, those experiments show evidence that

our method outperforms the state of the art when it comes to finding the best compromise between the computation time and the quality of the distance. This behavior takes its origin from (i) the stochastic nature of our method which can reduce the risk to get stuck in local minima and (ii) the fact that the other approaches do not scale well. An experiment on a graph classification task shows that being able to change the loss function without degrading the algorithm complexity is of high interest for finding the one that best fits the problem at hand.

This chapter is organized as follows: Section 4.2 details the notations and the necessary background on GW. Section 4.3 covers the state of the art approaches for solving the underlying problem. Section 4.4 presents our *Sampled Gromov Wasserstein* algorithm, derives convergence guarantees for it, and introduces our very fast specialized variant called *Pointwise Gromov Wasserstein*. Experiments are detailed in Section. 4.5.

4.2 Background on Gromov Wasserstein (GW)

In this chapter, the notations defined in Chapter 1 still hold but we shortly remind the main elements of the GW distance. Let $(\mathcal{X}, \mathcal{C}^{\mathcal{X}})$ be a compact metric space where $\mathcal{X}$ is a set included in a vector space and $\mathcal{C}^{\mathcal{X}}$ its associated metric. Let μ be a distribution with finite p-moment on $(\mathcal{X}, \mathcal{C}^{\mathcal{X}})$. Similarly, $(\mathcal{Y}, \mathcal{C}^{\mathcal{Y}})$ denotes another compact metric space and ν a distribution with finite p-moment on that space. While the OT problem requires the two distributions to lie in the same space, the GW distance allows to compare distributions in different metric spaces. Let $\mathcal{L}$ be a bounded loss function which allows the comparison of two distances. GW (Mémoli, 2009, 2011; Peyré et al., 2016) is defined as follows:

$$GW\left(\mathcal{C}^{\mathcal{X}}, \mathcal{C}^{\mathcal{Y}}, \mu, \nu\right) = \min_{\pi \in \Pi_{\mu\nu}} \int_{(\mathcal{X}\times\mathcal{Y})^2} \mathcal{L}\left(\mathcal{C}^{\mathcal{X}}(x, x'), \mathcal{C}^{\mathcal{Y}}(y, y')\right) d\pi(x, y) d\pi(x', y'). \tag{4.1}$$

The discrete case (see Fig. 4.1) can be formulated as:

$$GW\left(\mathcal{C}^{\mathcal{X}}, \mathcal{C}^{\mathcal{Y}}, a, b\right) = \min_{\boldsymbol{T} \in \Pi_{\mu\nu}} \sum_{i,j=1}^{I,I} \sum_{k,l=1}^{K,K} \boldsymbol{L}_{ijkl} \boldsymbol{T}_{ik} \boldsymbol{T}_{jl}, \tag{4.2}$$

where $\boldsymbol{L}_{ijkl} = \mathcal{L}\left(\mathcal{C}^{\mathcal{X}}(x_i, x_j), \mathcal{C}^{\mathcal{Y}}(y_k, y_l)\right)$. The bounded property of $\mathcal{L}$ is always verified as long as $\mathcal{C}^{\mathcal{X}}$, $\mathcal{C}^{\mathcal{Y}}$ and $\mathcal{L}$ are real value functions and both $\mathcal{X}$ and $\mathcal{Y}$ are bounded. Assuming that the two cost functions $\mathcal{C}^{\mathcal{X}}$ and $\mathcal{C}^{\mathcal{Y}}$ are symmetric, we recall the following notations: $\mathcal{E}(\boldsymbol{A}, \boldsymbol{A}') := \mathcal{E}(\boldsymbol{A}', \boldsymbol{A}) := \sum_{i,j=1}^{I,I} \sum_{k,l=1}^{K,K} \boldsymbol{L}_{ijkl} \boldsymbol{A}_{ik} \boldsymbol{A}'_{jl}$ and $\mathcal{E}(\boldsymbol{A}) := \mathcal{E}(\boldsymbol{A}, \boldsymbol{A})$.

4.3 Approaches to solve GW

We describe here the most used method for solving GW, namely *Entropic Gromov Wasserstein*, as well as two other approaches that aim at lowering the time complexity of the former. As all these methods use an iterative optimization, for the sake of simplicity, we omit in this section the number S of iterations (of the outer loop).

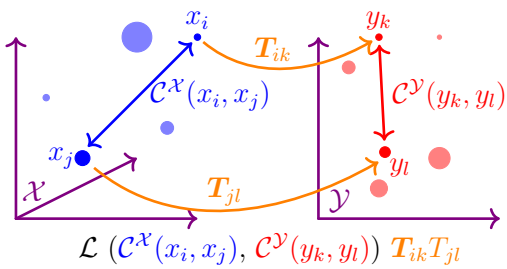


Figure 4.1: *Illustration of GW, with only one term L_{ijkl} of the quadruple sum of Eq. (4.2).*
Cited on page [67]

Entropic Gromov Wasserstein (EGW) As explained in Section 1.3, to solve an approximation of Problem (1.22), the authors of (Peyré et al., 2016) generalize the idea introduced in (Solomon et al., 2016) by using a gradient descent step followed by a projection, both according to the Kullback Leibler (KL) divergence. This Mirror Descent algorithm is applied on the regularization GW problem:

$$GW\left(\mathcal{C}^{\mathcal{X}}, \mathcal{C}^{\mathcal{Y}}, \mu, \nu\right) = \min_{\boldsymbol{T} \in \Pi_{\mu\nu}} \sum_{i,j=1}^{I,I} \sum_{k,l=1}^{K,K} \boldsymbol{L}_{ijkl} \boldsymbol{T}_{ik} \boldsymbol{T}_{jl} - \epsilon \mathcal{H}(\boldsymbol{T}). \tag{4.3}$$

This boils down to a two-step loop. First, from the current estimation of the transport plan $\boldsymbol{T}$, a new matrix defined as $\boldsymbol{\Lambda}_{jl} = \sum_{i,k=1}^{I,K} \boldsymbol{L}_{ijkl} \boldsymbol{T}_{ik}$ is computed, and which can be seen as a cost matrix induced by the current matching $\boldsymbol{T}$. Second, a new estimate of the transport plan is obtained by solving the following entropy regularized OT problem:

$$\min_{\boldsymbol{T} \in \Pi_{\mu\nu}} \langle \boldsymbol{\Lambda}, \boldsymbol{T} \rangle - \epsilon \mathcal{H}(\boldsymbol{T}). \tag{4.4}$$

When the loss $\mathcal{L}(\mathcal{C}^{\mathcal{X}}, \mathcal{C}^{\mathcal{Y}})$ can be decomposed as $f_1(\mathcal{C}^{\mathcal{X}}) + f_2(\mathcal{C}^{\mathcal{Y}}) - h_1(\mathcal{C}^{\mathcal{X}}) h_2(\mathcal{C}^{\mathcal{Y}})$ for some functions (f_1, f_2, h_1, h_2), it is shown that the $\boldsymbol{\Lambda}$ matrix can be computed in $O\left(N^3\right)$. This notably holds for the square loss and the KL divergence. However, in the general case, the complexity is $O\left(N^4\right)$, making this method intractable as N grows, as shown in our experiments.

Sliced Gromov-Wassertein (SGW) In (Rabin and Peyré, 2011), the authors introduce an alternative metric, called Sliced Wasserstein distance, which uses random 1D-projections. The advantage of this method lies in the fact that the 1D OT Problem (1.6) can be simply solved by sorting both empirical distributions (in $O\left(N \log(N)\right)$) and matching the sorted lists (see Section 1.1.2 for further details). In a similar manner, Sliced Gromov-Wasserstein (SGW) (Vayer et al., 2019b) projects each distribution in a common 1D space and solve the Gromov-Wasserstein problem (4.2) in this 1D space efficiently. While being very fast to compute, SGW comes with some limitations: (i) it cannot be used in general on graphs because a feature representation is needed to allow the 1D projection, (ii) it does not output an explicit transport plan which can be a pitfall in some applications like domain adaptation, (iii) it does not approximate the original GW distance and (iv) it is not *naturally* invariant to rotation (although the authors

propose a solution by repeatedly calling SGW). Note that while SGW's theoretical result and the $O(N\log(N))$ time complexity are relying on the square loss, its algorithmic approach can be adapted to handle arbitrary losses. This adaptation results in a $O\left(N^2\right)$ time complexity.

Scalable GW Learning (S-GWL) Scalable Gromov-Wasserstein Learning (Xu et al., 2019a) aims at making GW tractable to large scale graph analysis. It recursively decomposes the two original graphs into a set of smaller sub-graph pairs, using Gromov-Wasserstein barycenters (Peyré et al., 2016). Then, these sub-graphs are matched. The transport plan is updated with a proximal gradient method regularized with a KL divergence. The time complexity is $O\left(N^2\log(N)\right)$ when the cost matrices $\boldsymbol{C}^{\mathcal{X}}$ and $\boldsymbol{C}^{\mathcal{Y}}$ are not sparse and $\mathcal{L}$ is the square loss. However, with an arbitrary $\mathcal{L}$, the gain in complexity does not hold anymore because S-GWL cannot leverage the closed-form solution for the barycenter calculation.

4.4 Scalable GW optimization

We aim to address, in this section, the algorithmic bottleneck of EGW (Peyré et al., 2016) which prevents its use on large scale problems. We propose to compute the GW distance by applying a Frank-Wolfe (FW) algorithm which requires to solve an OT problem from a cost matrix seen as the expectation of a random variable. This allows us to propose a sampling strategy and thus to use a stochastic FW, to drastically reduce the algorithmic complexity of GW. We introduce our algorithm, called Sampled Gromov Wasserstein (**SaGroW**), and then derive its convergence guarantees.

We also present some special cases and a variant of SaGroW: Pointwise Gromov Wasserstein (**PoGroW**) which leverages very efficient 1D OT solvers but does not exhibit the drawbacks of SGW, and $SaGroW^{KL}$ a version using a Kullback-Leibler regularization. We finally show that an appropriate sampling strategy can be also be used to accurately and efficiently approximate the GW distance from a known transport plan.

4.4.1 Sampled Gromov Wasserstein (SaGroW)

It is known that the GW problem as described in Eq. (4.2) is not convex in general and thus difficult to solve. We propose to use a FW algorithm, which has the advantage of reducing the problem to several linear ones. At a given point $\boldsymbol{T}$, to find the direction given by the gradient, we need to find $\boldsymbol{T}'$ such as,

$$\min_{\boldsymbol{T}'\in\Pi_{\mu\nu}} \left\langle \nabla_{\boldsymbol{T}}\mathcal{E}(\boldsymbol{T}), \boldsymbol{T}'\right\rangle = \min_{\boldsymbol{T}'\in\Pi_{\mu\nu}} \left\langle \sum_{j,l=1}^{I,K} \boldsymbol{T}_{jl}\boldsymbol{L}_{\cdot j\,\bullet\, l}, \boldsymbol{T}'\right\rangle \tag{4.5}$$

where $\boldsymbol{L}_{.j.l}$ is an extracted matrix *i.e.*, $(\boldsymbol{L}_{.j.l})_{ik} = \boldsymbol{L}_{ijkl}$.

As the transport plan $\boldsymbol{T}$ sums to 1, we can interpret it as (the parameters of) a categorical distribution on pairs of points (j,l), or equivalently on the associated matrices $\boldsymbol{L}_{.j.l}$. We thus define

a random variable Λ on matrices, defined[1] by the distribution $\mathbb{P}(\Lambda = \boldsymbol{L}_{.j.l}) = \boldsymbol{T}_{jl}\ \forall (j,l) \in [\![1, N]\!]^2$. Leveraging this random variable, the cost matrix $\sum_{j,l} \boldsymbol{T}_{jl}\boldsymbol{L}_{.j.l}$ used in problem (4.5) can be seen as the expectation of Λ. Therefore, the problem can be rewritten as follows:

$$\min_{\boldsymbol{T}' \in \Pi_{\mu\nu}} \left\langle \mathbb{E}\left[\Lambda\right], \boldsymbol{T}' \right\rangle. \tag{4.6}$$

While solving this problem is still in $O\left(N^4\right)$ in general, it presents the advantage of opening the door to a sampling strategy allowing a reduction of the complexity. Indeed, rather than computing the entire expectation $\mathbb{E}\left[\Lambda\right]$, we suggest here to calculate an approximation by sampling M matrices $\{\boldsymbol{C}^m\}_{m=1}^M$. To get a matrix $\boldsymbol{C}^m$ drawn according to the distribution of Λ, it suffices to sample two indices (j_m, l_m) following the weights of the matrix $\boldsymbol{T}$. Consequently, $\boldsymbol{C}^m$ takes the form of the matrix $L_{.j_m.l_m}$. Using these sampled matrices, Problem (4.6) can be approximated as follows:

$$\min_{\boldsymbol{T}' \in \Pi_{\mu\nu}} \left\langle \frac{1}{M} \sum_{m=1}^{M} \boldsymbol{C}^m, \boldsymbol{T}' \right\rangle. \tag{4.7}$$

This approximation comes with two main advantages: (i) it allows a reduction of the computation time of the GW problem and (ii) similarly to a mini batch gradient descent, it might avoid being stuck in local minima and thus might lead to a better transport plan. Even though Problem (4.7) can be solved efficiently with any OT solver, our approach resorts to the Sinkhorn method (Cuturi, 2013) leading to a time complexity of $O\left((M+P)N^2\right)$ due to summing over M matrices and P iterations of the Sinkhorn algorithm. Due to the Sinkhorn method used, our approach is not totally a stochastic FW method. However the provided convergence proof to stationary point will still follow the same lines as the FW proofs (Reddi et al., 2016).

Algorithm 5 gives the pseudo-code of Sampled Gromov Wasserstein (**SaGroW**). In the absence of prior, the transport plan $\boldsymbol{T}_0$ is initialized to the joint distribution $\boldsymbol{a}\boldsymbol{b}^\top$ (line 1). At each iteration, M pairs of indices (j_m, l_m) are sampled from the current transport plan $\boldsymbol{T}_s$ (line 3). Then $\widehat{\boldsymbol{\Lambda}}$, the approximation of the gradient $\mathbb{E}\left[\Lambda\right]$, is computed (line 4) and used in an entropic regularization-based OT problem (4.4) solved using the Sinkhorn algorithm, yielding the plan $\boldsymbol{T}'_s$ (line 5). Similarly to the FW algorithm and to ensure that $\boldsymbol{T}'$ stays close to $\boldsymbol{T}$, line 6 performs a partial update $(1-\alpha)\boldsymbol{T}_s + \alpha\boldsymbol{T}'_s$. This update, inspired by the Frank-Wolfe algorithm, allows us to derive theoretical guarantees (see next section). Notice that Algorithm 5 returns a single transport plan and thus aims at minimizing the original GW problem. In practice, other strategies can be used: as the previous plan $\boldsymbol{T}_s$ and the optimized $\boldsymbol{T}'_s$ can be interpreted as distributions, line 6 can be omitted and replaced by a KL regularization (on line 5) between them, as detailed in Section 4.4.4.

We can notice, that for a new loss $\mathcal{L}$ that would be used, only the simple definition of $\mathcal{L}$ is needed. On the other hand, EGW requires (i) to know if the function $\mathcal{L}$ can be decomposed in a certain way and, if possible, (ii) to implement each of the 4 functions (f_1, f_2, h_1, h_2) separately. The simplicity of SaGroW can be useful for further use in practical applications.

[1] The definition is not rigorous: two matrices $\boldsymbol{L}_{.j.l}$ and $\boldsymbol{L}_{.j'.l'}$ may be equal, and then the probabilities add up.

We end this section by noting that when the expectation is fully computed in SaGroW (*i.e.*, $M = \infty$ and "$M = N^2$" in terms of complexity as sampling becomes useless) and α is set to 1, our method is strictly equivalent to the two steps loop of EGW described in Section 4.3. This connection will be used advantageously in the next section by deriving new convergence guarantees for EGW when the GW problem is concave. In the theoretical results and the experiments we use the sampling with replacement but for values of M close to N^2 the sampling without replacement should be a better choice.

Algorithm 5 SaGroW Cited on pages [70,70,71,72,76,121,122,124]

Require: $\boldsymbol{a}$, $\boldsymbol{b}$ (probability vectors of μ and ν), $\mathcal{C}^{\mathcal{X}}$, $\mathcal{C}^{\mathcal{Y}}$ (cost matrices), $\mathcal{L}$ (loss function), M (number of samples), ϵ (entropy regularization), α (partial update weight)

1: $\boldsymbol{T}_0 = \boldsymbol{a}\boldsymbol{b}^\top$
2: **for** s= 0 **to** S-1 **do**
3: $(j_m, l_m) \sim \text{Sample}(\boldsymbol{T}_s)\ \forall m \in [\![1, M]\!]$
4: $\widehat{\boldsymbol{\Lambda}}_{ik} = \frac{1}{M}\sum_{m=1}^{M} \mathcal{L}(\boldsymbol{C}^{\mathcal{X}}_{i,j_m}, \boldsymbol{C}^{\mathcal{Y}}_{k,l_m})\ \forall i, k \in [\![1, N]\!]$
5: $\boldsymbol{T}'_s =$ solve the regularized OT problem $(\boldsymbol{a}, \boldsymbol{b}, \widehat{\boldsymbol{\Lambda}}, \epsilon)$
6: $\boldsymbol{T}_{s+1} = (1-\alpha)\boldsymbol{T}_s + \alpha \boldsymbol{T}'_s$
7: **end for**
8: **return** $\boldsymbol{T}_{S-1}$

4.4.2 Convergence analysis

In this section, we aim at studying the convergence of Algorithm 5. Note that convergence guarantees have been already derived for EGW in (Peyré et al., 2016). However, based on Rangarajan et al. (1999), this convergence has been proven only when $\boldsymbol{L}$ produces a convex problem or when the entropy regularization term in really high. Unlike Peyré et al. (2016), the guarantees presented in this section have two main advantages: (i) they hold whatever the loss function, (ii) they are tight for small entropy regularization. Note that other results related to the GW problem have been recently derived in the literature. The authors of Xu et al. (2019b) prove the convergence of their proximal point method to a stationary point as long as their regularized GW problem can be solved perfectly at each iteration. Thus, the we don't know how to use such a result as the GW problem can only be approximated in practice. On the other hand, Redko et al. (2020) provides a guarantee on the convergence of Problem (4.2) under the condition that $\boldsymbol{L}$ yields a concave problem.

Our goal is to minimize (4.2), *i.e.*, to minimize $\mathcal{E}(\boldsymbol{T})$ under constraints on the marginals of $\boldsymbol{T}$. Let us now define the FW gap $G(\boldsymbol{T})$ as follows: $G(\boldsymbol{T}) := \mathcal{E}(\boldsymbol{T}, \boldsymbol{T}) - \min_{\boldsymbol{T}' \in \mathcal{U}_{\mu\nu}} \mathcal{E}(\boldsymbol{T}, \boldsymbol{T}')$. In a non convex setting, $\boldsymbol{T}$ is a stationary point of $\mathcal{E}(\boldsymbol{T})$ if and only if $G(\boldsymbol{T}) = 0$ (Reddi et al., 2016). The goal of our Theorem 3 is to provide a guarantee on the convergence of $G(\overline{\boldsymbol{T}})$ with $\overline{\boldsymbol{T}}$ uniformly sampled from $(\boldsymbol{T}_s)_{s \in [\![0, S-1]\!]}$. The convergence is proven on average over these sampling. A practical implementation will naturally take only the last transport plan, $\boldsymbol{T}_{S-1}$, and avoid

unnecessary computations.

Theorem 3. *(Based on Reddi et al. (2016)) For any $\boldsymbol{L}_{ijkl} \in [0, B]$, for any distributions μ and ν with uniform weights $\boldsymbol{a}$ and $\boldsymbol{b}$ respectively, for any optimal solution $\boldsymbol{T}^*$ of Problem (4.2), on average for the transport plan $\overline{\boldsymbol{T}}$ uniformly sampled from $(\boldsymbol{T}_s)_{s\in[\![0,S-1]\!]}$, on average over all the samplings, the following bound holds:*

$$\mathbb{E}\left[G\left(\overline{\boldsymbol{T}}\right)\right] \leq \sqrt{\frac{2B\left(\mathcal{E}(\boldsymbol{T}_0) - \mathcal{E}(\boldsymbol{T}^*)\right)N}{S}} + B\sqrt{\frac{2N}{M}} + \epsilon\log(N).$$

Proof. The complete proof is available in the Appendix C.1.1. It requires a novel lemma that quantifies the difference between the Wasserstein distances obtained with and without the entropic regularization: $0 \leq \langle \boldsymbol{C}, \boldsymbol{T}^\epsilon \rangle - \left\langle \boldsymbol{C}, \boldsymbol{T}^0 \right\rangle \leq \epsilon\log(N)$. We also prove that $\mathcal{E}(\boldsymbol{T})$ is $2N^2$-smooth and we bound the difference between two transport plans: $\|\boldsymbol{T} - \boldsymbol{T}'\|_F \leq \sqrt{\frac{2}{N}}$. Those two results allow us to adapt the proof of Theorem 2 in (Reddi et al., 2016) where our new Lemma is useful as the entropy regularized solvers do not find the exact OT minimum. □

While our bound cannot be explicitly computed as $\boldsymbol{T}^*$ is unknown, it gives meaningful information about Algorithm 5. First of all, it prompts us to initialize $\boldsymbol{T}_0$ so as to get $\mathcal{E}(\boldsymbol{T}_0)$ as close to $\mathcal{E}(\boldsymbol{T}^*)$ as possible. Without any prior information, $\boldsymbol{a}\boldsymbol{b}^\top$ (the uniform plan) appears to be a reasonable choice to avoid degenerated cases. Regarding the regularization parameter, if ϵ is not small enough, the convergence to a stationary point is not guaranteed. On the other hand, we can note that the number of sampled matrices M appears in only one term of the bound. Therefore, the costly complete computation of the expectation ($M = \infty$) would not guarantee the convergence while leading to a $O\left(N^4\right)$ complexity. Thus, our bound prompts us to find a compromise between the number of samples M and the number of iterations S.

As the GW problem has been shown in (Redko et al., 2020) to be often concave, especially with the square loss and the euclidean distance on both spaces, the following Theorem 4 gives a second bound dedicated to address the specific concave case. This result presents the major interest of providing an asymptotic convergence to a stationary point for EGW in this concave case, as the proofs proposed in (Peyré et al., 2016) only cover the convergence of EGW and only for high values of ϵ.

Theorem 4. *With the same notations as in Theorem 3 with the entropy regularization parameter ϵ_s that may now change along the iterations s, when $\boldsymbol{L}$ yields a concave GW problem, the following bound holds:*

$$\mathbb{E}\left[G\left(\overline{\boldsymbol{T}}\right)\right] \leq \frac{\mathcal{E}(\boldsymbol{T}_0) - \mathcal{E}(\boldsymbol{T}^*)}{2S} + B\sqrt{\frac{2N}{M}} + \frac{1}{S}\sum_{s=0}^{S-1}\epsilon_s log(N)$$

We can make the following comments from this bound. First, the convergence is better in the concave case as, unlike in Theorem 3, the first term is now linear in S. Second, as it can be seen in the proof (see Appendix C.1.1), it can be shown that in this concave scenario, the best

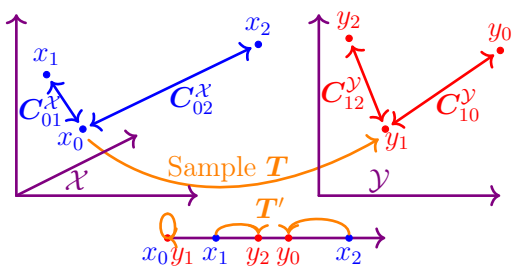


Figure 4.2: *Intuition behind PoGroW when $j, l = 0, 1$ are sampled from $\boldsymbol{T}$: only the distances to x_0 in $\mathcal{X}$ (on the left) and to y_1 in $\mathcal{Y}$ (on the right) characterize a pair, and then $\boldsymbol{T}'$ can be computed in $O(N \log N)$ like in 1D OT.*

Cited on page [73]

value for α is 1. Thus, if we completely compute the matrix $\boldsymbol{\Lambda}$ ($M = \infty$), this bound applies to EGW. For any sequence $(\epsilon_s)_{s\in\mathbb{N}}$ such that $\sum_{s=0}^{S-1} \epsilon_s$ is $o(S)$, the convergence of EGW to a stationary point is guaranteed.

When the regularization parameter $\epsilon = 0$, SaGroW is strictly equivalent to a Stochastic Frank-Wolfe (Reddi et al., 2016). Thus for $\epsilon > 0$, the convergence analysis of this general non-convex setting is very similar, except for the term that depends on ϵ which quantifies the error due to the entropy regularization. Moreover, note that if $\epsilon = 0$, EGW becomes also equivalent to the Frank-Wolfe algorithm (Frank et al., 1956) when its step size α is set to 1. Since the α parameter in our algorithm plays the same role as that of the step size of the Frank-Wolfe algorithm, we might wonder why SaGroW does not compute the optimal value using a line search. To the best of your knowledge, in this general non convex setting, there is no convergence guarantees towards a stationary point for a stochastic Frank-Wolfe algorithm that would make use of the optimal step. Moreover, it is worth noting that this optimal step is expensive ($O\left(N^4\right)$ complexity) to calculate without approximation. Considering an approximation would make the derivation of theoretical guarantees even more challenging.

4.4.3 Particular case: Pointwise GW

We focus in this section on the special case of SaGroW where only one matrix C is sampled (*i.e.*, $M = 1$) at each iteration. This variant, called Pointwise Gromov Wasserstein (**PoGroW**), makes it possible to leverage a dedicated solver to reduce the algorithmic complexity of GW.

When $M = 1$, if we sample a position j, l from $\boldsymbol{T}$, then we seek to minimize the following problem:

$$\min_{\boldsymbol{T}' \in \Pi_{\mu\nu}} \sum_{i,k=1}^{I,K} \mathcal{L}(\mathcal{C}^{\mathcal{X}}(\boldsymbol{x}_i, \boldsymbol{x}_j), \mathcal{C}^{\mathcal{Y}}(\boldsymbol{y}_k, \boldsymbol{y}_l)) \boldsymbol{T}'_{ik}. \tag{4.8}$$

As illustrated in Fig. 4.2, each point in $\mathcal{X}$ (resp. $\mathcal{Y}$) is simply defined by its distance to $\boldsymbol{x}_j$ (resp. $\boldsymbol{y}_l$), as done in papers that define a distribution using a distance to a point (Gelfand et al., 2005; Sato et al., 2020). With a single feature per point, Problem (4.8) can be solved very efficiently

in $O\left(N\log(N)\right)$ like a 1D OT problem: the two lists of distances can be sorted and matched. With non-convex losses, this sorting approach is only an approximation. PoGroW can be seen as a natural GW extension of Sliced Wasserstein where each point is described by its distance to a chosen "anchor" (instead of a position on a line). Recall that the output of Problem (4.8) is a transport plan. And if needed for the application at hand, the GW value can be computed in $O\left(N^2\right)$ (see Section 4.4.5).

In summary, PoGroW has the same low complexity as Sliced Gromov Wasserstein (Vayer et al., 2019b) but it overcomes its main limitations: PoGroW is naturally invariant to rotation; it returns a transport plan; it approximates the actual GW distance; it works with graphs (without having the coordinates for the vertices).

Details of the continuous version of PoGroW

In this section, we explain how to adapt PoGroW to handle continuous distributions. This additional section has not been published in the paper "Sampled Gromov Wasserstein" (Kerdoncuff et al., 2021). The key element of this algorithm is that the continuous one dimensional OT can be solved easily. The continuous GW distance and transport plan have already been studied in (Vayer, 2020; Salmona et al., 2021; Le et al., 2021) in particular cases, but this is the first attempt to approximate the transport plan and the distance for any distribution, cost matrix and convex loss. However, we will see that the resulting GW distance is far from the optimal one.

Let denote f_μ and f_ν the two density functions associated with $\mu \in \mathcal{P}(\mathcal{X})$ ($\mathcal{X} \subset \mathbb{R}^D$) and $\nu \in \mathcal{P}(\mathcal{Y})$ ($\mathcal{Y} \subset \mathbb{R}^{D'}$). We will note F the cumulative density function associated with any density function f. In addition, we note $\mathbb{S}_D^{\boldsymbol{x},r}$ the $D-1$ dimensional sphere centered on $\boldsymbol{x} \in \mathcal{X}$ with radius $r \in \mathbb{R}$.

We start the algorithm with two points $\boldsymbol{x}_0$ and $\boldsymbol{y}_0$, for instance the respective mean of the two distributions. We also set π_0 to the uniform distribution. Similarly to the discrete case, at given points $\boldsymbol{x}_n$ and $\boldsymbol{y}_n$ with $n \in \mathbb{N}$, we create two one dimensional density functions $f_{\boldsymbol{x}_n}$ and $f_{\boldsymbol{y}_n}$ such that:

$$\begin{aligned} f_{\boldsymbol{x}_n} : \quad & \mathbb{R}_+ \to \mathbb{R}_+ \\ & r \mapsto \int_{\mathbb{S}_D^{\boldsymbol{x}_n,r}} f_\mu(\boldsymbol{\theta}) d(\boldsymbol{\theta}) \end{aligned} , \tag{4.9}$$

and similarly for $f_{\boldsymbol{y}_n}$.

Then we have to apply the OT between those two functions, and as we are in a one dimensional case, the solution can be obtained in closed form for two continuous distributions μ and ν and with $\mathcal{L}$ a convex loss function. As explained in Section 1.1.2, the Optimal Transport plan can also be described as a Monge mapping which reads $F_{\boldsymbol{y}_n}^{-1} \circ F_{\boldsymbol{x}_n}$. Both $F_{\boldsymbol{x}_n}$ and $F_{\boldsymbol{y}_n}$ can be evaluated at a specific point easily by integrating $f_{\boldsymbol{x}_n}$ and $f_{\boldsymbol{y}_n}$. Given $\boldsymbol{y}$, the inverse $F_{\boldsymbol{y}_n}^{-1}(\boldsymbol{y})$ can also be evaluated by a dichotomy search. In practice, the evaluation of the three functions $F_{\boldsymbol{y}_n}$, $F_{\boldsymbol{x}_n}$ and $F_{\boldsymbol{y}_n}^{-1}$ are only approximations. However, we can set a very high precision value.

At this point, $F_{\boldsymbol{y}_n}^{-1} \circ F_{\boldsymbol{x}_n}$ is only a $\mathbb{R}_+ \to \mathbb{R}_+$ function, but we can use it to define the transport plan $\pi_{n+1} = (1-\alpha)\pi_n + \alpha\tilde{\pi}_{n+1}$ between the two distributions μ and ν. We define the transport plan $\tilde{\pi}_n$ with its conditional density $f_{\tilde{\pi}_{n+1}^{\boldsymbol{x}}}$ for all fixed $\boldsymbol{x} \in \mathcal{X}$,

$$f_{\tilde{\pi}_{n+1}^{\boldsymbol{x}}}(\boldsymbol{y}) = \begin{cases} 0 \text{ if } \|\boldsymbol{y} - \boldsymbol{y}_n\| \neq F_{\boldsymbol{y}_n}^{-1} \circ F_{\boldsymbol{x}_n}(\|\boldsymbol{x} - \boldsymbol{x}_n\|) \\ \frac{f_\nu(y)}{\int_{\mathbb{S}_{D'}^{\boldsymbol{y}_n, \|\boldsymbol{y}-\boldsymbol{y}_n\|}} f_\nu(\boldsymbol{\theta}) d(\boldsymbol{\theta})} \text{ if not} \end{cases} . \tag{4.10}$$

In other words, the conditional density $f_{\tilde{\pi}_{n+1}^{\boldsymbol{x}}}(\boldsymbol{y})$ is 0 if $r = \|\boldsymbol{x} - \boldsymbol{x}_n\|$ is not matched by the Monge mapping $F_{\boldsymbol{y}_n}^{-1} \circ F_{\boldsymbol{x}_n}$ to $r' = \|\boldsymbol{y} - \boldsymbol{y}_n\|$. If the two radius match, then the mass at this point $\boldsymbol{y}$ is divided by all the masses in the sphere with radius r', to have $f_{\tilde{\pi}_{n+1}^{\boldsymbol{x}}}$ that sum to one.

To find the next $\boldsymbol{x}_{n+1}$ and $\boldsymbol{y}_{n+1}$ points, we sample $\boldsymbol{x}_{n+1} \sim \mu$ and sample $\boldsymbol{y}_{n+1} \sim \pi_{n+1}^{\boldsymbol{x}_{n+1}}$. Note that only one call to each of $F_{\boldsymbol{x}_n}$ and $F_{\boldsymbol{y}_n}^{-1}$ is required at each iteration. Figure 4.3 provides an example of mapping between two continuous distributions.

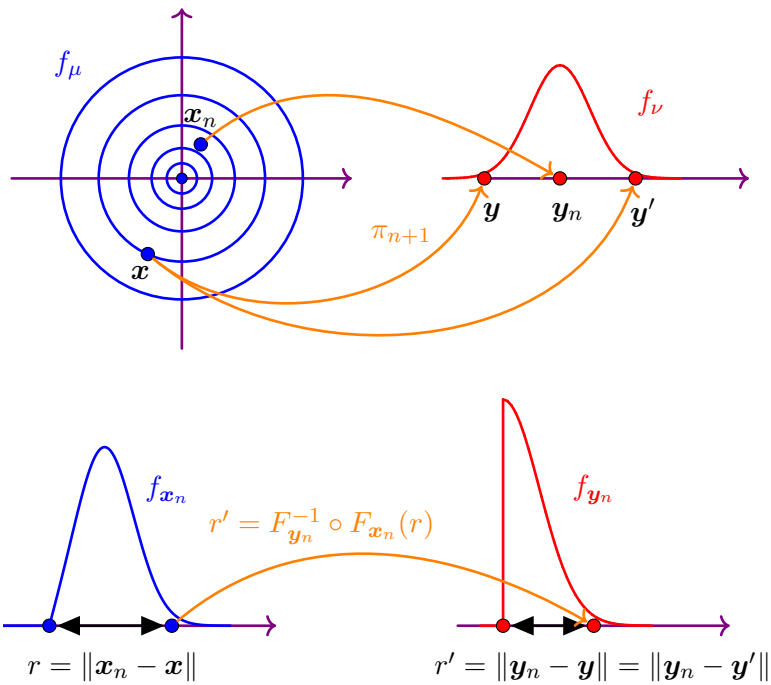


Figure 4.3: *(**Top**) Representation of the two unimodal Gaussian distributions μ and ν in their respective space. $\boldsymbol{x}_n$ and $\boldsymbol{y}_n$ are the two sampled points from the previous transport plan π_n and they induce a new transport plan π_{n+1}. This new transport plan will spread on a sphere the mass of a point $\boldsymbol{x}$, thus in this 1D case half of the mass is sent to $\boldsymbol{y}$ and half to $\boldsymbol{y}'$. (**Bottom**) Representation of the two density functions $f_{\boldsymbol{x}_n}$ and $f_{\boldsymbol{y}_n}$ to explain why π_{n+1} send the mass of $\boldsymbol{x}$ to $\boldsymbol{y}$ and $\boldsymbol{y}'$.*

Cited on page [75]

4.4.4 A KL regularization-based variant

As the transport plan $\boldsymbol{T}$ is a distribution and most GW algorithms progressively update $\boldsymbol{T}$, an interesting idea is to encourage the next plan $\boldsymbol{T}'$ to be close (in terms of KL divergence) to the

current estimate $\boldsymbol{T}$. This idea, already used in (Xu et al., 2019b) based on (Xie et al., 2020), can be applied to our SaGroW algorithm: we name this approach SaGroW^{KL} and describe it below.

In Algorithm 5, we used partial updates to explore the transport plan space while encouraging the new value of $\boldsymbol{T}$ to be close to the preceding one, as reflected in line 6. We suggest here a slight modification, consisting in using a Kullback Leibler (KL) regularization between $\boldsymbol{T}$ and $\boldsymbol{T}'$ in line 5 and removing line 6. We showed in Section 1.3.2 that both (Peyré et al., 2016; Xu et al., 2019b) can be seen as a projected Mirror Descent algorithm. Similarly to (Xu et al., 2019b), SaGroW is equivalent to a stochastic projected Mirror Descent algorithm (Zhou et al., 2017; Zhang and He, 2018; Hanzely and Richtárik, 2021), where the KL divergence is used. The only difference comes from the stochastic approximation of the gradient. This leads to the following sampled optimization problem,

$$\min_{\boldsymbol{T}' \in \Pi_{\mu\nu}} \left\langle \frac{1}{M} \sum_{m=1}^{M} \boldsymbol{C}^m, \boldsymbol{T}' \right\rangle + \epsilon KL(\boldsymbol{T}' || \boldsymbol{T}), \tag{4.11}$$

which can be rearranged into,

$$\min_{\boldsymbol{T}' \in \Pi_{\mu\nu}} \left\langle \frac{1}{M} \sum_{m=1}^{M} \boldsymbol{C}^m - \epsilon log(\boldsymbol{T}), \boldsymbol{T}' \right\rangle - \epsilon \mathcal{H}(\boldsymbol{T}'). \tag{4.12}$$

This regularization allows to take advantage of the Sinkhorn-Knopps solver (Cuturi, 2013) as it is similar to equation (1.8) with a cost function modified to take into account the current prior $\boldsymbol{T}$. Even if ϵ is high, the optimization might lead to a solution close to the edge of the polytope with enough iterations which is not the case with a classical entropy regularization without prior. The time complexity does not increase as it is still $O\left((P+M)N^2\right)$. As this regularization is not specific to our method, we will also use it for EGW during the experiments to allow a fair comparison. On the other hand, note that this regularization cannot be used with PoGroW as it currently does not seem possible to solve 1D entropy-regularized OT in $O\left(Nlog(N)\right)$ (Cuturi et al., 2019). Note also that the convergence Theorem 3 does not hold anymore with this regularization.

4.4.5 Efficient computation the GW distance from a transport plan

This section introduces and evaluates a low-complexity high-accuracy method for the estimation of $\mathcal{E}(\boldsymbol{T})$. Indeed, while SaGroW and PoGroW provide important complexity improvements, one might argue that they only find a good transport plan $\boldsymbol{T}$ and do not provide a value for $\mathcal{E}(\boldsymbol{T})$. An exact computation of $\mathcal{E}(\boldsymbol{T})$ has a $O\left(N^4\right)$ time complexity, and it would dominate the complexity of our algorithms in applications where $\mathcal{E}(\boldsymbol{T})$ is required, for example when GW is used as a dissimilarity measure between graphs. Additionally, having an efficient way of estimating $\mathcal{E}(\boldsymbol{T})$ opens the door to selecting the best transport plan among a set of plans, *e.g.*, obtained by varying the hyper-parameters or the random seed of an algorithm.

We address this issue in this section. Similar to Equation (4.6), we propose to interpret the sums in the definition of $\mathcal{E}(\boldsymbol{T})$ as the expectation of a random variable R (this time real-valued

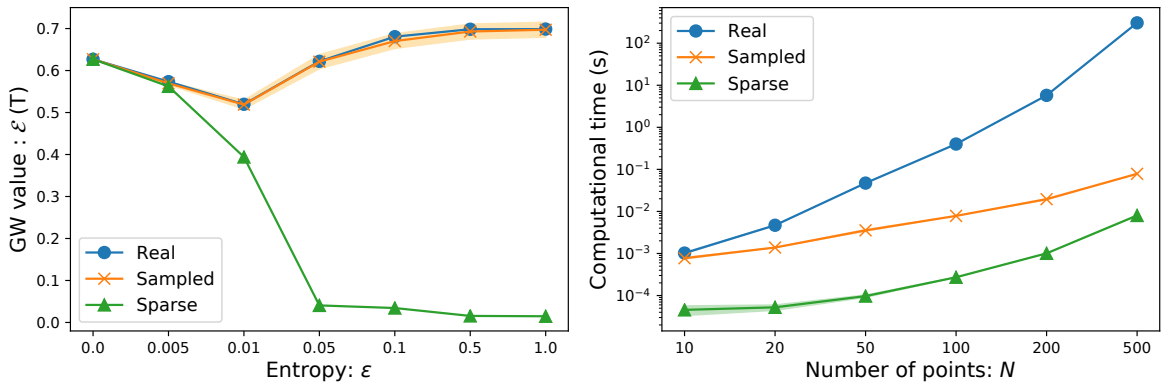


Figure 4.4: *Estimated value of $\mathcal{E}(\boldsymbol{T})$ as sparsity decreases due to an increasing ϵ regularization in EGW (left) and evolution of the time required for its estimation as N grows (right). The absolute loss is used in these experiments and the distributions take the form of two graphs generated using a gaussian random partition graph (Brandes et al., 2003). For a given ϵ and N, the same $\boldsymbol{T}$ (obtained using EGW) is passed to the three considered methods: Real) an exact one which computes completely $\mathcal{E}(\boldsymbol{T})$, Sampled) our sampling method described in Section 4.4.5, and, Sparse) a sparse approximation which keeps only the $2N$ largest values of $\boldsymbol{T}$ and sets the other entries to 0. The mean and 2 standard deviations over 10 runs are displayeded on both figures. When the standard deviation is not visible, it corresponds either to a deterministic method or a value very close to 0.*

Cited on pages [77,77,129]

instead of matrix-valued, so with a quadruple sum), with $\mathbb{P}\left[R = \boldsymbol{L}_{ijkl}\right] = \boldsymbol{T}_{ik}\boldsymbol{T}_{jl}$:

$$\mathbb{E}\left[R\right] = \sum_{i,j,k,l=1}^{I,I,K,K} \boldsymbol{L}_{ijkl}\boldsymbol{T}_{ik}\boldsymbol{T}_{jl}. \tag{4.13}$$

Instead of simply sampling this expectation, we propose to stratify by each index i, j to improve the quality of the estimate. Let U_i be the event "*i is chosen for the first dimension of $\boldsymbol{L}$*" and U'_j be the event "*j is chosen for the second dimension of $\boldsymbol{L}$*". Based on the marginal $\boldsymbol{a}$ and using the law of total expectation, $\mathbb{E}\left[R\right]$ can be rewritten as:

$$\mathbb{E}\left[R\right] = \sum_{i,j=1}^{I,I} \mathbb{P}(U_i \cap U'_j)\mathbb{E}\left[R|U_i \cap U'_j\right] = \sum_{i,j=1}^{I,I} \boldsymbol{a}_i\boldsymbol{a}_j\mathbb{E}\left[R|U_i \cap U'_j\right]. \tag{4.14}$$

For each (i, j), the conditional expectation is approximated using M samples of a random variable X_{ik}, defined by $\mathbb{P}(X_{ij} = \boldsymbol{L}_{ijkl}) = \mathbb{P}\left[R = \boldsymbol{L}_{ijkl}|U_i \cap U'_j\right] = \boldsymbol{T}_{ik}\boldsymbol{T}_{jl}$. Finally, $\hat{R} = \sum_{ij} \boldsymbol{a}_i\boldsymbol{a}_j \frac{1}{M}\sum_{m=1}^{M} X_{ij}^m$ defines an unbiased estimate of the GW distance which can be computed in $O\left(MN^2\right)$ (details about the variance estimate are provided in the Appendix C.1.3).

As shown in Figure 4.4 (left), the prediction is perfect for a sparse transport plan ($\epsilon = 0$), while still being almost perfect and much better than a naive sparse approximation of the OT plan as ϵ increases. Figure 4.4 (right) confirms that this approximation is clearly faster than the exact computation which becomes quickly intractable as N grows.

Table 4.1: *Complexity of each method with an arbitrary loss function, with S iterations, P Sinkhorn iterations, N points in the dataset and M matrix samples. Note that the complexity can be different for specific loss functions.*
Cited on page [77]

Methods	GW approximation	Transport Plan	$\mathcal{E}(\boldsymbol{T})$	Total
EGW	Yes	$S(P+N^2)N^2$	N^4	$S(P+N^2)N^2$
SaGroW	Yes	$S(P+M)N^2$	N^2	$S(P+M)N^2$
S-GWL	Yes	Unknown	SN^2	Unknown
SGW	No	Unavailable	SN^2	SN^2
PoGroW	Yes	$SN\log(N)$	N^2	N^2

Having at our disposal an efficient method for estimating $\mathcal{E}(\boldsymbol{T})$, we can now fully compare, in Table 4.1, the complexity of the state of the art methods with that of SaGroW and PoGroW, for the general case of an arbitrary loss function. From this table, we have evidence that SaGroW leads to a drastic reduction of the algorithmic complexity of EGW. On the other hand, PoGroW fully benefits from the 1D projections, but unlike SGW, it provides a transport plan and does approximate the original GW problem.

4.5 Experiments

In this section[2], we first compare different GW methods on both their *speed* and their *accuracy*. We use here the term *accuracy* to express the capability of the methods to minimize $\mathcal{E}(\boldsymbol{T})$. Indeed, as the exact (optimal) GW distance is unknown for a given dataset (solving this problem is known to be NP-hard), the best method will be the one with the smallest value of $\mathcal{E}(\boldsymbol{T})$. Then, we analyze the impact of the hyperparameters, illustrating that our approach covers a range of very good trade-offs between speed and accuracy. Using a real graph-classification task, we finally illustrate why being able to solve GW for various loss functions is important.

Two efficient implementations of SaGroW and PoGroW are available in the Python for Optimal Transport (POT) library (Flamary et al., 2021).

4.5.1 General setup and methods

We compare SaGroW^{KL} and PoGroW with: (I) EGW (Peyré et al., 2016); (II) EGW^{KL}, a KL regularized version of EGW described in Xu et al. (2019b); (III) EMD-GW, which is similar to EGW$_0$, but uses the OT solver of (Bonneel et al., 2011) as the Sinkhorn algorithm (Cuturi, 2013) cannot handle a null value for ϵ; (IV) S-GWL (Xu et al., 2019a), adapted for arbitrary loss functions using the optimizer of Wright (1996) to update the barycenter; (V) SGW when

[2]The code to reproduce all the experiments, figures and tables is available in the GitHub `https://github.com/Hv0nnus/Sampled-Gromov-Wasserstein`

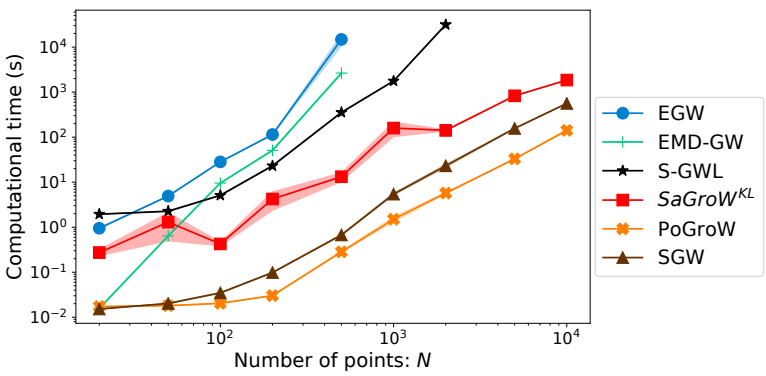


Figure 4.5: *Computational time of various methods to compute the distance between samples from two mixtures of gaussians. The mean and the standard deviation over 10 runs are reported.* *Cited on page [79]*

the points are available, with an adaptation to arbitrary losses; (VI) the uniform transport plan, used as a baseline.

While Section 4.5.3 will detail the impact of the hyperparameters, the next section reports, for each method, the results obtained by the set of parameters with the lowest GW estimation. To take into account the stochasticity of some methods the GW estimation for each hyperparameter set is taken on average over 10 runs. ϵ is chosen among $\{0.001, 0.005, 0.01, 0.005, 0.1\}$ for EGW and EGW^{KL}, and in $\{0.001, 0.01, 0.1, 1, 10, 100\}$ for S-GWL. To have comparable sets of hyperparameters, we fix some of our parameters: in PoGroW, a step of $\alpha = 0.8$, and in SaGroW, the number of samples $M = 10$ and a KL regularization $\epsilon = 1$. Experiments in the Appendices C.2.5 and C.2.6 show that: SaGroW is much less sensitive to ϵ than EGW and $\alpha = 0.8$ is a reasonable choice. The number of iterations S is chosen among $\{10, 100, 500, 1000\}$ to obtain a reasonable accuracy-speed trade-off.

This experiment compares the quality of the transport plan and the computational time of the methods for an increasing number of points N. Each method minimizes Problem (4.2) and returns a transport plan $\boldsymbol{T}$ (besides SGW, see below). In order to assess the quality of this transport plan, $\mathcal{E}(\boldsymbol{T})$ is then computed exactly. Notably, our GW distance approximation (see Section 4.4.5) is *not* used in this first experiment. The mean and standard deviation of $\mathcal{E}(\boldsymbol{T})$ over ten runs are reported.

The loss $\mathcal{L}$ chosen here is the absolute loss in order to show the capacity of our methods to deal with any arbitrary loss function. We remind that EGW, S-GWL and SGW are much faster (with speeds that are comparable to our approach) for some specific losses, such as the squared Euclidean loss (see Appendix C.2.2 and Section 4.5.4).

To include SGW (which needs points to project) in this comparative study, a first dataset uses μ and ν that are composed of N points sampled from two different mixtures of gaussians. Details about the generation of the datasets are available in the Appendix C.2.1.

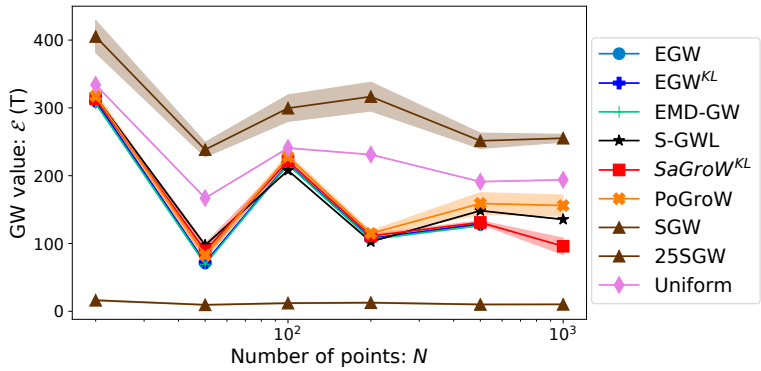


Figure 4.6: *GW distance estimation between samples from two mixtures of gaussians. The mean and standard deviation over 10 runs are reported for the stochastic methods.*

Cited on page [81]

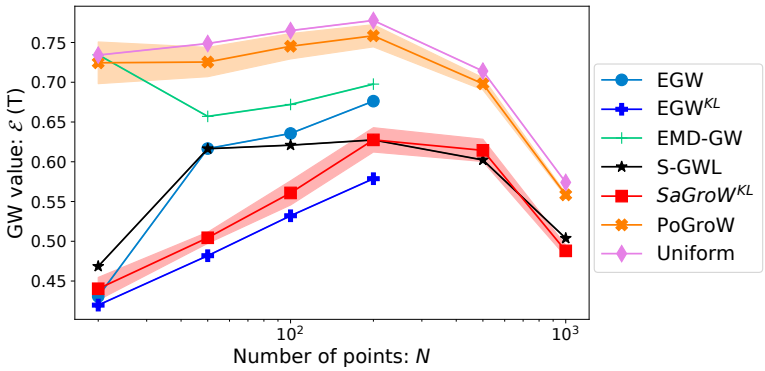


Figure 4.7: *GW distance estimation between synthetic graphs (Brandes et al., 2003). The mean and standard deviation over 10 runs are reported for the stochastic methods.*

Cited on page [81]

4.5.2 Speed and accuracy of the GW estimate

Figure 4.5 shows, in a log-log representation, that EGW and EMD-GW become quickly intractable when the number of points increases and that S-GWL is slightly faster. We exclude EGW^{KL} for the clarity of the figure as it has a computational time similar to EGW. SaGroW, PoGroW and SGW behave better, with a quadratic complexity (linear slope of 2 in log-log) but with different multiplicative factors (offsets in the log-log plot).

Figure 4.6 reports the quality of the obtained GW value. Comparing SGW to other methods is complicated as it does not return a transport plan, nor aims at computing an approximation of the GW distance. We thus report the distance it computes and also the same rescaled by a factor 25. With rescaling, we see that SGW seems to behave more like the uniform transport plan than like the GW methods (which produce better-than-uniform plans). While all other methods predict very similar GW distances, EGW-based methods have often the best accuracy. However, when N reaches 1000 points, we can observe interesting behaviors: EGW is not able to provide any result, PoGroW is the fastest with a lesser accuracy than S-GWL, and SaGroW provides the best value while being much faster than S-GWL.

In a second series of experiments, we make use of graphs that are generated using a gaussian random partition graph (Brandes et al., 2003). On this more difficult dataset, we see in Figure 4.7 that SaGroW is very competitive with the best method EGW^{KL} while being able to scale to more than 200 nodes, which is the limit for all EGW-based methods. With more nodes, SaGroW is as accurate as S-GWL but remains much faster and scalable (computation times are similar to the ones from the first dataset). In this experiment, a key factor of success seems to be the KL regularization, used in EGW^{KL}, S-GWL and SaGroW. This can explain why PoGroW stays close to the uniform baseline.

4.5.3 Hyperparameters analysis

We now focus on the impact of the numbers of iterations S and samples M, showing that these allow our approach to cover a variety of trade-offs between speed and accuracy. More experiments (in the Appendix C.2) consider other parameters such as different loss functions or dataset sizes. We also study in this experiment the impact of the ϵ parameter of other methods.

Figure 4.8 shows that increasing the number of iterations S yields a strong improvement for SaGroW, independently of the number of samples M. Interestingly, the accuracy of SaGroW is similar regardless the value of M. This remark supports the key assumption of this chapter that the entire computation of the expectation is not needed. The standard deviation displayed in Figure 4.8 shows that most runs provide similar GW distances, with enough iterations. However, there is a high variance with less iterations which tends to highlight that the different runs of SaGroW take different paths during the optimization. As shown in Figure 4.9, the speed of EGW and S-GWL does not vary much with ϵ but this parameter needs to be chosen carefully for those methods to reach a good accuracy.

On Figure 4.10 we can see that PoGroW is even faster than SaGroW: it can provide a

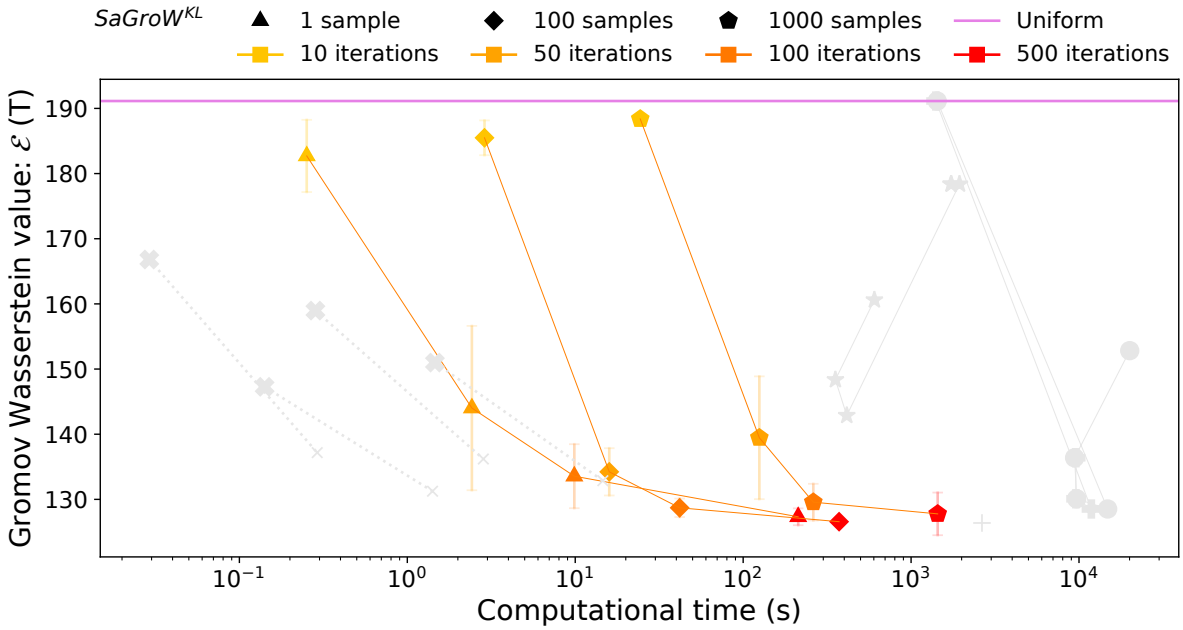


Figure 4.8: *Impact of the number of samples M and the number of iterations S for SaGroW on the GW distance estimation and computational time, for two sets of 500 points sampled from two mixtures of gaussians. The mean and standard deviation over 10 runs are displayed.*

Cited on pages [81,81,131,131,131,133,134,134,190,190,190,190]

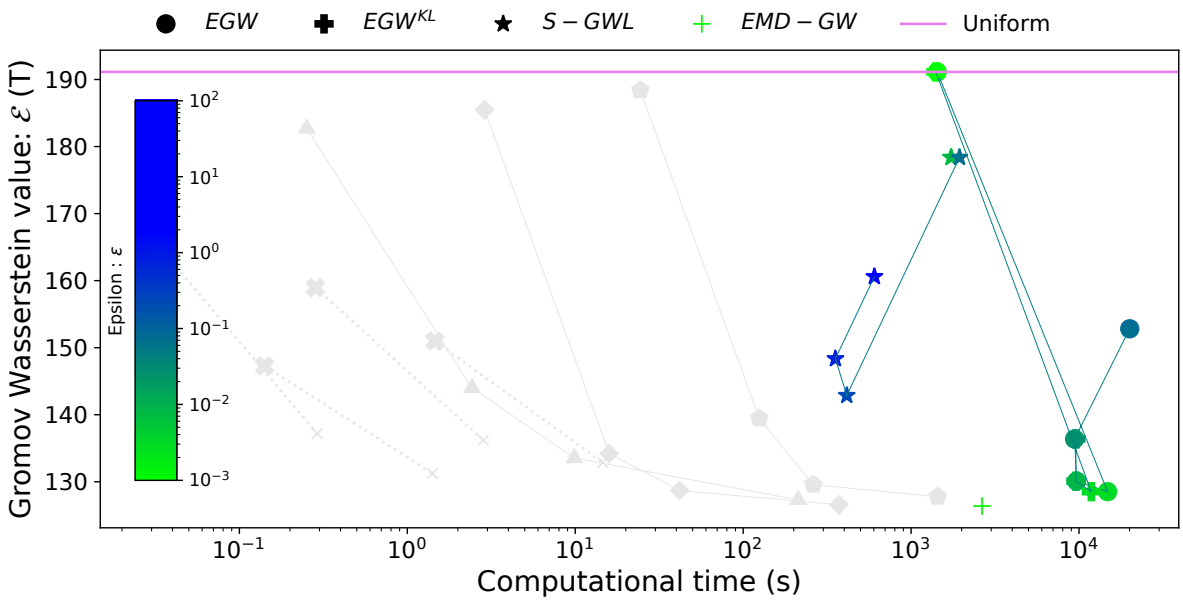


Figure 4.9: *Impact of the Kullback-Leiber regularization ϵ for EGW and S-GWL on the GW distance estimation and computational time, for two sets of 500 points sampled from two mixtures of gaussians.*

Cited on pages [81,131,131,131,133,134,134,190,190,190,190]

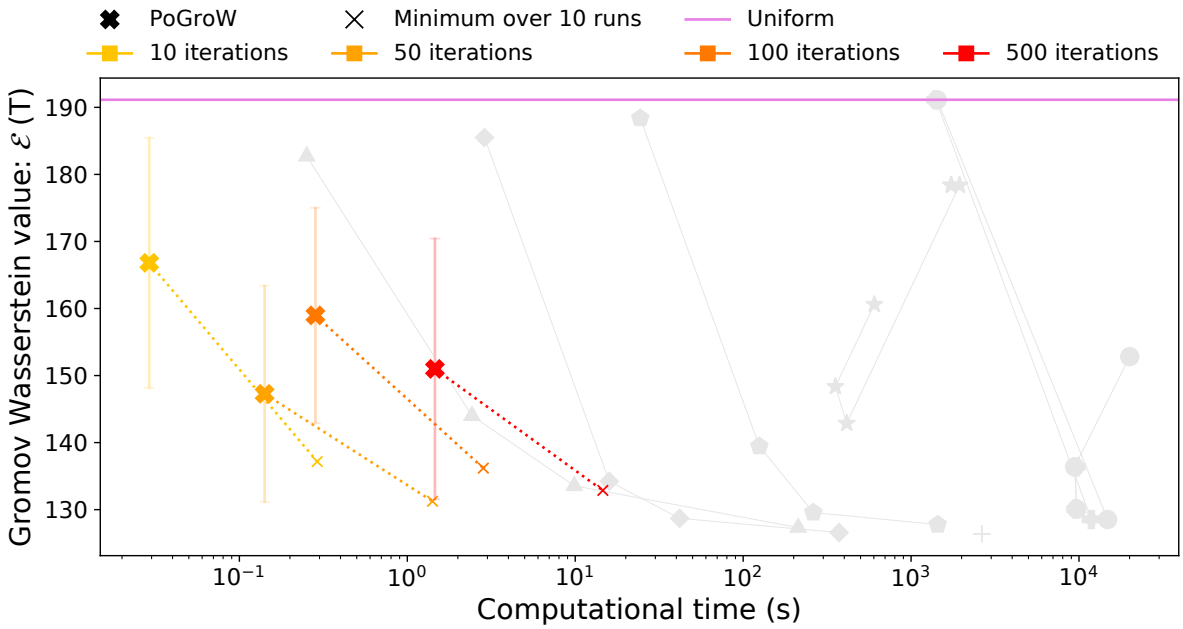


Figure 4.10: *Impact of the number of iterations S for PoGroW on the GW distance estimation and computational time, for two sets of 500 points sampled from two mixtures of gaussians. The mean and standard deviation over 10 runs are displayed. To take advantage of the large stochasticity, the minimum over 10 runs is also display.*

Cited on pages [81,81,131,131,131,133,134,134,190,190,190,190]

reasonable approximation in a second, compared to the three hours required by EGW. Because PoGroW does not resort to a KL regularization, it is more impacted by stochasticity: two runs can yield very different results. This can be used advantageously by keeping the plan that gives the lowest GW estimation among ten runs (crosses on Fig. 4.10). The distance can be approximated using Section 4.4.5 to keep the same time complexity. The combination of SaGroW and PoGroW allows to obtain a good trade-off between speed and accuracy.

Beyond the algorithmic advantages shown above, one last key question remains: is it useful, in an application, to compute the GW distance for other losses than the widely used square loss?

4.5.4 Graph classification

We illustrate here the usefulness of using different loss functions in a context of graph classification. We take the FIRSTMM-DB graph dataset (Neumann et al., 2013) which is the one with the biggest average nodes number (1377) over the database of (Kersting et al., 2016). Each of the 41 graphs of the dataset describes an object from one of the 11 classes (cup, knife, etc.). The distance matrix of each graph $\boldsymbol{C}^{\mathcal{X}}$ and $\boldsymbol{C}^{\mathcal{Y}}$ is computed using the shortest path length, similarly to Mémoli (2011). For each method, we compute the pairwise GW distance matrix. Finally, a 1-Nearest-Neighbor classifier is used to predict the class of each graph (using a leave-one-graph-out scheme).

Table 4.2: *Classification accuracy and computation time of various methods on a 11-class graph classification task. In this summary table, only the hyperparameters yielding the best classification accuracy are reported, for each considered method.*
Cited on page [84]

Methods	Accuracy	Time (s)
S-GWL$_5$	0.44	23.4
EGW$^{KL}_{0.005}$	0.24	41.1
EMD-GW	0.37	16.6
EGW$_{0.001}$	0.22	36.2
Uniform	0.07	0.1
SaGroW$_{p=1}$	**0.49**	11.6
SaGroW$_{p=2}$	0.39	12.7
PoGroW$_{p=2}$	0.39	**0.5**

Section 4.5.2 showed that EGW, EGW^{KL} and S-GWL are very slow with arbitrary loss functions on graphs (with around 1000 nodes). Therefore, we use for them the square loss to allow them to be competitive from a time complexity perspective. We consider ten values for the entropic regularization, $\epsilon \in [10^{-4}, 10^2]$. SGW is excluded as it is unable to handle graphs. For our methods, we set $\epsilon = 0.1$ for SaGroW and $\alpha = 0.8$ for PoGroW and keep $M = 1$, $S = 100$ for both methods. However, ten different loss functions $\mathcal{L}$ are tested, notably $|\boldsymbol{C}^{\mathcal{X}}_{ij} - \boldsymbol{C}^{\mathcal{Y}}_{kl}|^p$ for different values of $p \in [0.5, 3]$.

The results are reported in Table 4.2. Looking at SaGroW, we see that the classical square loss ($p = 2$) is outperformed, *e.g.*, by the absolute loss ($p = 1$) which yields a better classification accuracy. Beyond that, the ability of SaGroW to handle arbitrary losses allows it to get the best overall accuracy, across all the methods. The explanation can be that the $L1$ loss is more robust to outlier nodes, which might be important on this real dataset. Note that while EGW and S-GWL are fast as they are computed with the square loss for $\mathcal{L}$, SaGroW is still slightly faster. PoGroW has a competitive accuracy and even outperforms EGW while being very fast. The complete table with every hyperparameter run is available in the Appendix C.2.7.

While the goal of this experiment is to correctly classify graphs, we can still compare the GW distances obtained from the transport plans returned by all methods. This comparison only makes sense with the same (square) loss for all methods. Averaged over 41^2 distances, SaGroW gets the lowest value of 336, followed by EMD-GW with 341. This highlights the fact that, on a real dataset, the stochasticity used by our method can lead to a better GW distance estimation.

4.5.5 Continuous Gromov Wasserstein

In this section, we propose a last (small) experiment, to illustrate that PoGroW can provide a transport plan for the GW problem between two bounded continuous distributions with known density.

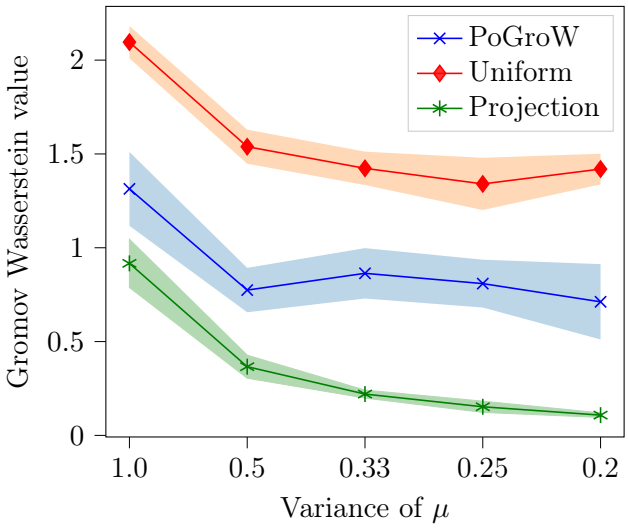


Figure 4.11: *The GW distance is estimated between μ and ν, two Gaussians with $\begin{pmatrix} 1 & 0 \\ 0 & v \end{pmatrix}$ and $\begin{pmatrix} 1 \end{pmatrix}$ variance respectively. Different values of v are tested. Each method outputs a transport plan π and the distance is estimated using a Monte Carlo approximation, the mean and standard deviation are reported.*

Cited on pages [84,85]

Figure 4.11 provides an example of GW distance estimations between μ and ν, two Gaussians with variance equal to $\begin{pmatrix} 1 & 0 \\ 0 & v \end{pmatrix}$ and $\begin{pmatrix} 1 \end{pmatrix}$ respectively. The Euclidean distance is used for $\mathcal{C}^{\mathcal{X}}$ and $\mathcal{C}^{\mathcal{Y}}$, and the squared Euclidean distance is used for $\mathcal{L}$.

The *uniform* baseline uses the uniform distribution $\mu \times \nu$ as transport plan. The *projection* baseline projects the first dimension of the distribution μ onto ν and discards the other axis. This transport plan might not be the best one in general, but it has been proven, with the squared Euclidean distance for $\mathcal{C}^{\mathcal{X}}, \mathcal{C}^{\mathcal{Y}}, \mathcal{L}$, to be at least the best linear Monge mapping (Vayer, 2020, Theorem 4.2.6) or equivalently, the best transport plan restricted to Gaussian distributions (Salmona et al., 2021, Theorem 4.1).

To compute the GW estimation, we sample pairs of points in μ, then apply the transport plan on each point, and compute the loss $\mathcal{L}$ with the initial pair and the transported pair of points. If the transport plan is not a deterministic function, we simply sample again the position in ν according to the distribution given by the transport plan. We can see on Figure 4.11 that Pointwise Gromov Wasserstein is clearly better than the *uniform* baseline, however still far from the *projection* baseline.

4.6 Conclusion

In this chapter, we presented both algorithmic and theoretical contributions to address the still open problem related to the calculation of the Gromov Wasserstein distance. We propose a

method to drastically reduce the time complexity of GW for arbitrary loss functions. To do so, we tackle the bottleneck of the mostly used GW solver, namely EGW, by using a sampling strategy to efficiently approximate the costly sum of N^2 matrices. Our SaGroW algorithm is supported with theoretical convergence guarantees to a stationary point in the general non-convex setting. We also introduce PoGroW, an algorithm which samples only one matrix and allows us to benefit from a very low complexity by using 1D OT. We show that PoGroW overcomes the main issues related to SGW. Experiments on synthetic datasets show that our methods are tractable for a large number of points and offer a good trade-off between speed and accuracy. Finally, a real world experiment on graph classification illustrates the interest of choosing different loss functions. In order to deal with potential outliers, we show that the absolute loss associated with SaGroW gives the highest classification accuracy. This capacity to choose *ad-hoc loss functions* could push the state of the art in various graph applications by unlocking their use with large graphs.

In this chapter, we saw that GW can handle points described with a distance function applied only pairwise. However, other types of point descriptions, for instance based on a "distance" between 3 points, are not handle with the existing OT or GW formulations. We will propose a new OT-based formulation to tackle this problem in the next chapter, allowing us to solve the OT problem on tensors of any order.

Chapter 5

Optimal Tensor Transport

Abstract

This chapter is based on the paper "Optimal Tensor Transport" published at the AAAI 2022 conference (Kerdoncuff et al., 2022), and proposes a general framework which can align tensors of any order. This new formulation encompasses several OT-based problems. We have seen in the previous chapters that OT is a useful tool in machine learning to align finite datasets lying in the same vector space. Based on a similar formulation, Co-Optimal Transport (Co-OT) jointly estimates two distinct transport plans, one for the rows (points) and one for the columns (features), to match two data matrices that use different features. On the other hand, Gromov Wasserstein (GW) looks for a single transport plan from two pairwise distance matrices. Both Co-OT and GW can be seen as specific extensions of OT to more complex data. In this chapter, we propose a new framework, called Optimal Tensor Transport (OTT), which takes the form of a generic formulation that includes OT, GW and Co-OT and can handle tensors of any order by learning possibly multiple transport plans. We derive theoretical results for the resulting new distance and present an efficient way for computing it. We further illustrate the interest of such a formulation in Domain Adaptation and Comparison-based Clustering.

5.1 Introduction

Some extensions of the original OT problem have been proposed in the literature to tackle more complex settings. Chapter 4 highlighted the interest of the Gromov Wasserstein (GW) distance to handle graphs or to compare distributions lying in different spaces. A second variant is Co-Optimal Transport (Co-OT) (Redko et al., 2020) which recently extended OT to datasets lying in different vector spaces, that is with potentially distinct features. The underlying idea is to jointly learn two transport plans. The first one aligns the examples as in standard OT while the second one aligns the most similar features. This has been shown to be of particular interest in heterogeneous DA.

Motivation of this work. While Co-OT and GW already cover a wide range of problems, it is worth noticing that there are still other settings that do not fall into the aforementioned categories (see Figure 5.2). For example, assume that, in each dataset, several distance matrices, or adjacency matrices, or graphs, are available (Figure 5.2). It is then necessary to learn two transport plans as in Co-OT but based on pairwise matrices as in GW. The first transport plan aligns the examples while the second one matches the distance matrices. Similarly, comparison-based learning (Vikram and Dasgupta, 2016; Ukkonen, 2017; Emamjomeh-Zadeh and Kempe, 2018; Ghoshdastidar et al., 2019; Perrot et al., 2020) addresses the problem of learning when neither an explicit representation nor a pairwise distance matrix is available. Instead, only ordinal comparisons, that can be stored in a third order tensor, are available (Figure 5.2). Unfortunately, the existing OT formulations are unable to compute the distance between two datasets represented in this way.

Contributions. In this chapter, we propose *Optimal Tensor Transport* (OTT), a new OT formulation that can handle datasets represented as tensors of any order while potentially learning multiple transport plans. The underlying idea is to jointly match the different dimensions of each tensor with respect to their weights. Depending on the structure of the dataset, different mappings can be forced to be equal. We illustrate the capabilities of our new OTT formulation in Figure 5.1 where we use our approach to align subsets of the MNIST (LeCun et al., 1998) and the USPS datasets (Friedman et al., 2001). Both are represented as 3D-tensors by concatenating the 2D images and OTT is used to match the points (first mapping), pixel rows (second mapping) and pixel columns (third mapping) jointly. One can notice that our approach efficiently matches the digits of the same class (0 and 1) while only using supervision from the MNIST dataset. Furthermore, the pixel-level transport plans are both close to the identity. From a theoretical perspective, we show that OTT subsumes the previous formulations including standard OT, Co-OT, and GW (the top row of Figure 5.2 illustrates how our new OTT setting covers OT, Co-OT and GW). We also show that OTT can be seen as a distance between tensors of any order and thus that it can be used to compute tensor barycenters. From an algorithmic point of view, we rely on the efficient optimization scheme based on sampling as described in Chapter 4, that allows us to drastically reduce the practical complexity of our formulation. Empirically, we demonstrate the interest of OTT in DA as well as in Comparison-based Clustering.

It is worst mentioning that a related problem has been tackled in the literature: the D-regular hypergraphs (Berge, 1984) matching. Such a problem is indeed equivalent to the particular case of OTT where all the transport plans are forced to be the same. But it has either a different formulation or different constraints on the matching. Zass and Shashua (2008) proposes to find a soft matching between D-regular hypergraphs, with uniform inequality constraints, using a Kullback-Leibler objective function. Duchenne et al. (2011) also matches hypergraphs, with a formulation similar to OTT but uses only row constraints for the matching matrix.

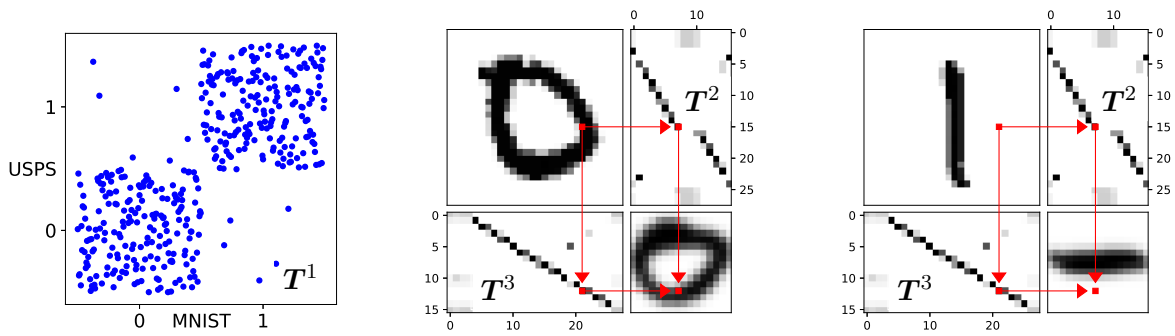


Figure 5.1: ***(Left)*** *Transport plan* $\boldsymbol{T}^1$ *between 400 images (only digits* 0 *and* 1*) of MNIST and USPS datasets;* ***(Middle and Right)*** *(top left) An example from MNIST and (bottom right) an example from USPS with a 90° right rotation; (top right) the OT plan* $\boldsymbol{T}^2$ *between the rows of MNIST and USPS; (bottom left) the OT plan* $\boldsymbol{T}^3$ *between the columns of MNIST and USPS; the arrows explain how to match the pixels between the two datasets using* $\boldsymbol{T}^2$ *and* $\boldsymbol{T}^3$ *obtained with OTT.*

Cited on pages [88,91,139]

5.2 Preliminary knowledge

To facilitate the comprehension of our general framework, we recall the standard Optimal Transport and Gromov Wasserstein formulations, and explain the Co-OT (Redko et al., 2020) problem. For the sake of clarity, we only consider here the discrete case. Nevertheless, all the formulations presented in this section, as well as OTT, can be straightforwardly extended to the continuous case by replacing the sums by integrals over the compared distributions. To adhere to the visual intuition given in Figure 5.2 and prepare for our generalization, we unify the formulations below. In particular, we introduce subscripts and superscripts that are usually not used in the standard formulations.

Optimal Transport (Villani, 2008). Let $\boldsymbol{X}$ and $\boldsymbol{Y}$ be two datasets defined over the same feature space $\mathcal{X}$ (for example $\mathcal{X} = \mathbb{R}^F$), with respectively $I_1 \in \mathbb{N}$ and $K_1 \in \mathbb{N}$ points with weights $\boldsymbol{a}^1 \in \Delta_{I_1}$ and $\boldsymbol{b}^1 \in \Delta_{K_1}$. The optimal transport plan between $\boldsymbol{X}$ and $\boldsymbol{Y}$ can be obtained by solving the following optimization problem:

$$\mathrm{OT} = \min_{\boldsymbol{T}^1 \in \mathcal{U}_{\boldsymbol{a}^1 \boldsymbol{b}^1}} \sum_{i_1=1}^{I_1} \sum_{k_1=1}^{K_1} \mathcal{L}(\boldsymbol{X}_{i_1}, \boldsymbol{Y}_{k_1}) \boldsymbol{T}^1_{i_1 k_1}. \tag{5.1}$$

Here, we replace the usual ground cost c by a loss function $\mathcal{L}$ which measures the cost of aligning two examples $\boldsymbol{X}_i$ and $\boldsymbol{Y}_k$. Note that an extension of OT which is orthogonal to (and could be combined with) everything we cover in this chapter is the **multi-marginal** OT (Carlier, 2003; Moameni, 2014; Pass, 2015; Friedland, 2020) that aligns $R \geq 3$ datasets simultaneously: $\mathcal{L}$ becomes a function of R parameters and $\boldsymbol{T}^1$ an R-order tensor.

Co-Optimal Transport (Redko et al., 2020). Co-Optimal Transport also aims at transporting points from two datasets $\boldsymbol{X}$ and $\boldsymbol{Y}$. However, contrary to standard OT, these

datasets may have different feature spaces $\mathcal{X} \subseteq \mathbb{R}^{I_2}$ and $\mathcal{Y} \subseteq \mathbb{R}^{K_2}$ of respective dimensions I_2 and K_2 and equipped with weights $\boldsymbol{a}^2 \in \Delta_{I_2}$ and $\boldsymbol{b}^2 \in \Delta_{K_2}$. The goal is to jointly match the points with a first transport plan $\boldsymbol{T}^1$ and the features with a second transport plan $\boldsymbol{T}^2$. The Co-OT formulation is as follows:

$$\text{Co-OT} = \min_{\boldsymbol{T}^1 \in \mathcal{U}_{\boldsymbol{a}^1 \boldsymbol{b}^1} \boldsymbol{T}^2 \in \mathcal{U}_{\boldsymbol{a}^2 \boldsymbol{b}^2}} \sum_{i_1, i_2 = 1}^{I_1, I_2} \sum_{k_1, k_2 = 1}^{K_1, K_2} \mathcal{L}(\boldsymbol{X}_{i_1 i_2}, \boldsymbol{Y}_{k_1 k_2}) \boldsymbol{T}^1_{i_1 k_1} \boldsymbol{T}^2_{i_2 k_2}. \tag{5.2}$$

Gromov Wasserstein (Memoli, 2007). Instead of having features describing the examples, let us consider that we only have access to within-dataset pairwise similarities or dissimilarities, that is $\boldsymbol{X}$ and $\boldsymbol{Y}$ are now square matrices of dimensions $I_1 \times I_1$ and $K_1 \times K_1$. It means that the two datasets may have different feature spaces, as in Co-OT, but since these feature spaces are implicit, it is sufficient to learn a single transport plan $\boldsymbol{T}^1$. The Gromov Wasserstein (GW) formulation is defined as follows:

$$\text{GW} = \min_{\boldsymbol{T}^1 \in \mathcal{U}_{\boldsymbol{a}^1 \boldsymbol{b}^1}} \sum_{i_1, i_2 = 1}^{I_1, I_1} \sum_{k_1, k_2 = 1}^{K_1, K_1} \mathcal{L}(\boldsymbol{X}_{i_1 i_2}, \boldsymbol{Y}_{k_1 k_2}) \boldsymbol{T}^1_{i_1 k_1} \boldsymbol{T}^1_{i_2 k_2}. \tag{5.3}$$

It is typical, for both Co-OT and GW, to use a loss function $\mathcal{L}$ (often the squared difference) that operates on two numbers: in Co-OT, $\mathcal{L}$ compares the value of a feature from a point in $\mathcal{X}$ with one feature from a point in $\mathcal{Y}$. In GW, it compares an entry of the pairwise matrix $\boldsymbol{X}$ to one in $\boldsymbol{Y}$. Both these formulations can be extended by allowing $\mathcal{L}$ to compare more complex entries such as F-dimensional vectors in $\mathbb{R}^F$. As illustrated in the top row of Figure 5.2, corresponding to the formulations of Equations (5.1), (5.2), and (5.3), all these approaches solve different problems but still share common principles. Below, we propose a new OT formulation that subsumes all of them.

5.3 Optimal Tensor Transport (OTT)

Given the notational complexity involved in our generic formulation, let us first explain the intuition behind the subscripts associated with OTT as illustrated in Figure 5.2. Both Co-OT and GW work on matrices (that is tensors with $D = 2$ dimensions) and thus will be represented with 2 digits. Since Co-OT uses $E = 2$ different transport plans, $\boldsymbol{T}^1$ for the first dimension, and $\boldsymbol{T}^2$ for second one, computing Co-OT will boil down to solving OTT_{12} as defined below. On the other hand, GW uses a single ($E = 1$) plan $\boldsymbol{T}^1$ for both the first and the second dimensions, thus corresponding to OTT_{11}. Note that two dimensions that share a transport plan must have the same sizes. For instance, GW (OTT_{11}) deals with square matrices.

Starting to generalize, when working with D dimensions, a given OT extension considers $E \leq D$ transport plans and associates a transport plan (index) to each dimension. This is done by specifying an *affectation function* $f : [\![1, D]\!] \to [\![1, E]\!]$ or equivalently, a D-tuple of transport plan indices, that is $f \in [\![1, E]\!]^D$. For instance, Co-OT uses $f = (1, 2)$ which corresponds to the subscript in OTT_{12}.

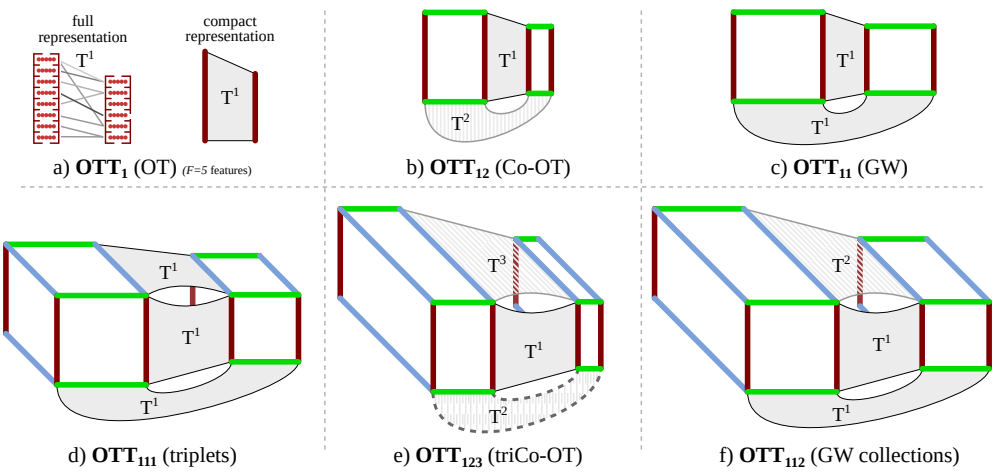


Figure 5.2: *Representation of various formulations of OTT with, each time, the two datasets and the different transport plans (best viewed in color).*

Cited on pages [87,87,87,88,88,90,90,91,91]

For a given $f \in [\![1, E]\!]^D$, we can now detail our OTT_f formulation (denoted OTT when no ambiguity arises) that defines a distance between two datasets $\boldsymbol{X}$ and $\boldsymbol{Y}$, represented as order $D+1$ tensors of respective size $(I_{f(1)}, ..., I_{f(D)}, F)$ and $(K_{f(1)}, ..., K_{f(D)}, F)$. The first D dimensions will be matched between the two datasets using the transport plans, while the last dimension (F) is the feature dimension used to compare 2 points with the loss $\mathcal{L}$. To simplify the rest of the chapter, we will suppose that $F = 1$, as done in Co-OT and GW above. The OTT distance between $\boldsymbol{X}$ and $\boldsymbol{Y}$ relies on finding a list of optimal transport plans $(\boldsymbol{T}^e)_{e\in[\![1,E]\!]}$ under some constraints (on the marginal) defined respectively by the weight vectors $(\boldsymbol{a}^e)_{e\in[\![1,E]\!]}$ and $(\boldsymbol{b}^e)_{e\in[\![1,E]\!]}$. OTT is defined as follows:

$$\text{OTT}_f(\boldsymbol{X}, \boldsymbol{Y}, (\boldsymbol{a}^e)_{e\in[\![1,E]\!]}, (\boldsymbol{b}^e)_{e\in[\![1,E]\!]}) = \min_{\forall e\, \boldsymbol{T}^e \in \mathcal{U}_{\boldsymbol{a}^e \boldsymbol{b}^e}} \mathcal{E}_f\left(\boldsymbol{X}, \boldsymbol{Y}, (\boldsymbol{T}^e)_{e\in[\![1,E]\!]}\right) \tag{5.4}$$

$$\text{where } \mathcal{E}_f\left(\boldsymbol{X}, \boldsymbol{Y}, (T^e)_{e\in[\![1,E]\!]}\right) = \sum_{i_1,...,i_D=1}^{I_{f(1)},...,I_{f(D)}} \sum_{k_1,...,k_D=1}^{K_{f(1)},...,K_{f(D)}} \mathcal{L}(\boldsymbol{X}_{i_1...i_D}, \boldsymbol{Y}_{k_1...k_D}) \prod_{d=1}^{D} \boldsymbol{T}^{f(d)}_{i_d k_d}.$$

From this general formulation and looking at Equations (5.1), (5.2) and (5.3) with the support of Figure 5.2, one can check that OT corresponds to OTT_1 (with F possibly > 1), Co-OT is equivalent to OTT_{12} and GW corresponds to OTT_{11}. Our OTT formulation makes it possible to handle new forms of datasets as illustrated in the second row of Figure 5.2. In the experiments (see Section 5.6), we will specifically consider two versions of OTT, each with order 3 tensors: (i) OTT_{111} corresponds to datasets of triplets (like GW but with triplets instead of pairs); (ii) OTT_{112} works with datasets that are collections of adjacency matrices. Figure 5.1 gives an illustration of a third kind of datasets, where OTT_{123} has been applied on collections of images, like Co-OT but with three dimensions. Even though we will focus on 3D-tensors in our experiments, it is worth noticing that the theoretical results derived in Section 5.5 and the proposed algorithm presented in the next section hold for any tensor order and might be used

with higher tensor dimension ($D = 4$) in comparison based learning tasks (Ghoshdastidar et al., 2019) or to match hypergraphs (Berge, 1984).

5.4 Algorithm to solve OTT

In this section, we detail how to efficiently solve the main optimization problem behind Equation 5.4. As described in the previous chapter and in Section 1.3.2, the most used method for solving GW, EGW (Peyré et al., 2016), can be seen as a Mirror Descent scheme (Beck and Teboulle, 2003) with the Kullback-Leibler divergence on a regularized version of GW: $\min_{\boldsymbol{T}\in\mathcal{U}_{\boldsymbol{a}^1\boldsymbol{b}^1}} \mathcal{E}(\boldsymbol{T}) + KL(\boldsymbol{T}, \boldsymbol{a}^1\boldsymbol{b}^{1\top})$. Thus, at a point $\boldsymbol{T}^1$, Peyré et al. (2016) shows that the Mirror Descent step is equivalent to the entropy regularization OT problem (Cuturi, 2013):

$$\min_{\boldsymbol{T}\in\mathcal{U}_{\boldsymbol{a}^1\boldsymbol{b}^1}} \langle \nabla_{\boldsymbol{T}^1}\mathcal{E}, \boldsymbol{T}\rangle + \epsilon KL(\boldsymbol{T}, \boldsymbol{a}^1\boldsymbol{b}^{1\top}). \tag{5.5}$$

Xu et al. (2019b) based on (Xie et al., 2020) changes the uniform distribution $\boldsymbol{a}^1\boldsymbol{b}^{1\top}$ in Equation (5.5) to the previous transport plan $\boldsymbol{T}^1$. In fact, this is equivalent to applying a Mirror Descent algorithm on the original GW problem (Equation (5.3)) instead of the regularized one. Based on this analysis, we will also use a Mirror Descent algorithm to solve the OTT problem.

When the goal is to find multiple transport plans, we propose to use an alternating approach, similar to Co-OT, where each transport plan is optimized in turn while the others remain fixed. In summary, we combine the idea of the existing solver of Co-OT and GW and apply an alternate Mirror Descent algorithm with the KL divergence for OTT, with the main bottleneck being the computation of the gradient of $\mathcal{E}$. The pseudo-code of our approach is presented in Algorithm 6. The main steps are the following:

Algorithm 6 OTT Cited on pages [92,94,96]

Require: datasets $\boldsymbol{X}, \boldsymbol{Y}$, weights $(\boldsymbol{a}^e)_{e\in[\![1,E]\!]}, (\boldsymbol{b}^e)_{e\in[\![1,E]\!]}$, loss function $\mathcal{L}$, nb. of samples M, regularization ϵ

1: $\forall e \in [\![1, E]\!],\ \boldsymbol{T}^e = \boldsymbol{a}^e\boldsymbol{b}^{e\top}$
2: **for** s= 0 **to** S-1 **do**
3: **for** $e = 1$ **to** E **do**
4: $\widehat{\nabla_{\boldsymbol{T}^e}\mathcal{E}} = M$ samples of the gradient using Equation (5.7).
5: $\boldsymbol{T}^e = \min_{\boldsymbol{T}\in\mathcal{U}_{\boldsymbol{a}^e\boldsymbol{b}^e}} \left\langle \widehat{\nabla_{\boldsymbol{T}^e}\mathcal{E}}, \boldsymbol{T}\right\rangle + \epsilon KL(\boldsymbol{T}, \boldsymbol{T}^e)$
6: **end for**
7: **end for**

Step 1: We initialize the transport plans (line 1) with the marginal product.

Step 2: We compute the gradient of $\mathcal{E}$. For the sake of clarity, we assume that the aligned tensors are "cubic", that is all their dimensions are of the same size N. In this case, the overall

gradient with respect to $\boldsymbol{T}^e$ is a N^2 matrix:

$$\nabla_{\boldsymbol{T}^e}\mathcal{E} = \sum_{\{d'|f(d')=e\}} \sum_{\substack{i_1,\dots,i_{d'-1}=1 \\ i_{d'+1},\dots,i_D=1}}^{\substack{I_{f(1)},\dots,I_{f(d'-1)} \\ I_{f(d'+1)},\dots,I_{f(D)}}} \sum_{\substack{k_1,\dots,k_{d'-1}=1 \\ k_{d'+1},\dots,k_D=1}}^{\substack{K_{f(1)},\dots,K_{f(d'-1)} \\ K_{f(d'+1)},\dots,K_{f(D)}}} \mathcal{L}\begin{pmatrix} \boldsymbol{X}_{i_1\dots i_{d'-1},\bullet,i_{d'+1}\dots i_D} \\ \boldsymbol{Y}_{k_1\dots k_{d'-1},\bullet,k_{d'+1}\dots k_D} \end{pmatrix} \prod_{\substack{d=1 \\ d\neq d'}}^{D} \boldsymbol{T}^{f(d)}_{i_d k_d}. \quad (5.6)$$

Unfortunately, computing the overall gradient exactly would be too expensive. Indeed, a naive approach leads to a complexity of $O\left(N^{2D}\right)$ operations which is prohibitively high. To simplify the computation, a first idea would be to generalize the approach used for GW by Peyré et al. (2016) to our problem. This would reduce the complexity to $O\left(N^{D+1}\right)$ for a particular class of functions $\mathcal{L}$, notably the square loss. We provide a proof of this approach in the supplementary material. Nevertheless, the latter remains too expensive as soon as $D = 3$. Thus, instead, we will rely on the idea explore in the previous chapter, and uses a stochastic Mirror Descent. Similarly to the GW case, we can notice that the gradient of $\mathcal{E}$ with respect to $\boldsymbol{T}^e$ can be seen as a sum of expectations over matrices of size N^2, $(\Lambda^{d'})_{\{d'|f(d')=e\}}$ such that:

$$\mathbb{P}\left(\Lambda^{d'} = \mathcal{L}\begin{pmatrix} \boldsymbol{X}_{i_1\dots i_{d'-1},\bullet,i_{d'+1}\dots i_D}, \\ \boldsymbol{Y}_{k_1\dots k_{d'-1},\bullet,k_{d'+1}\dots k_D} \end{pmatrix}\right) = \prod_{d=1|d\neq d'}^{D} \boldsymbol{T}^{f(d)}_{i_d k_d}$$

$$\text{with} \quad \sum_{\substack{i_1,\dots,i_{d'-1}=1 \\ i_{d'+1},\dots,i_D=1}}^{\substack{I_{f(1)},\dots,I_{f(d'-1)} \\ I_{f(d'+1)},\dots,I_{f(D)}}} \sum_{\substack{k_1,\dots,k_{d'-1}=1 \\ k_{d'+1},\dots,k_D=1}}^{\substack{K_{f(1)},\dots,K_{f(d'-1)} \\ K_{f(d'+1)},\dots,K_{f(D)}}} \prod_{\substack{d=1 \\ d\neq d'}}^{D} \boldsymbol{T}^{f(d)}_{i_d k_d} = 1$$

since $\forall e \in [\![1, E]\!]$, $\sum_{i,k=1}^{I_e,K_e} \boldsymbol{T}^e_{ik} = 1$. The gradient can then be reformulated as:

$$\nabla_{\boldsymbol{T}^e}\mathcal{E} = \sum_{\{d'|f(d')=e\}} \mathbb{E}\left(\Lambda^{d'}\right). \quad (5.7)$$

It means that it is possible to obtain an unbiased estimate of the gradient in $O\left(MN^2\right)$ operations where M is the number of elements sampled to estimate the expectation in Equation (5.7). Note that we assume that D is negligibly small compared to N.

Step 3: The last step (line 5) requires to solve a regularized OT problem, that can be efficiently solved using the Sinkhorn solver (Xu et al., 2019b; Cuturi, 2013).

5.5 Theoretical results

In this section, we derive two main theoretical results. We first show through Theorem 5 that as long as the cost function is a proper distance, then OTT is a distance between D-order tensors. As a consequence, we can naturally define the OTT barycenters between tensors. Then, Theorem 6 states that the optimal barycenter can be found in closed form for particular loss functions.

Theorem 5. *OTT is a distance between weighted tensors $(\boldsymbol{X}, (\boldsymbol{a}^e)_{e\in[\![1,E]\!]})$ and $(\boldsymbol{Y}, (\boldsymbol{b}^e)_{e\in[\![1,E]\!]})$, represented in canonical form, for any affectation function f, as long as $\mathcal{L}$ is a proper distance.*

The proof is provided in the supplementary material. This result notably extends the distance proof of Co-OT (Redko et al., 2020) to matrices of different sizes and to non-uniform weights. Even though their comparison with OTT is out of the scope of this chapter, notice that other distances exist between higher-order tensors (De Lathauwer et al., 2000; Lai et al., 2013; Liu et al., 2010).

We can then define the OTT barycenter of several tensors with any affectation function f.

Definition 2. *(OTT barycenter) Given $B \in \mathbb{N}$ weighted tensors of sizes $((K_e^b)_{e\in[\![1,E]\!]})_{b\in[\![1,B]\!]}$, $\left(\boldsymbol{X}^b \in \mathbb{R}^{K^b_{f(1)}\dots K^b_{f(D)}}, (\boldsymbol{b}^{e,b} \in \Delta_{K_e^b})_{e\in[\![1,E]\!]}\right)_{b\in[\![1,B]\!]}$. Let $(\lambda_1, ..., \lambda_B) \in \Delta_B$ be the weights quantifying the importance of each tensor. For fixed size $(I_e)_{e\in[\![1,E]\!]}$ and fixed marginals $(\boldsymbol{a}^e \in \Delta_{I_e})_{e\in[\![1,E]\!]}$, the OTT barycenter is defined as follows:*

$$\min_{\boldsymbol{X}\in\mathbb{R}^{I_{f(1)}\dots I_{f(D)}}} \sum_{b=1}^{B} \lambda_b OTT(\boldsymbol{X}, \boldsymbol{X}^b, (\boldsymbol{a}^e)_{e\in[\![1,E]\!]}, (\boldsymbol{b}^{e,b})_{e\in[\![1,E]\!]}). \tag{5.8}$$

Note that the barycenter could also be defined in a similar manner with the marginals $(\boldsymbol{a}^e)_{e\in[\![1,E]\!]}$ not fixed.

To solve Problem (5.8), we propose to minimize alternatively the objective function w.r.t. $\boldsymbol{X}$ and $(\boldsymbol{T}^{e,b})_{e\in[\![1,E]\!]}$, the transport plans between $\boldsymbol{X}$ and $\boldsymbol{X}^b$. The latters can be found independently for each $b \in [\![1, B]\!]$ using Algorithm 6. Interestingly, $\boldsymbol{X}$ can be found in closed form for particular loss functions, which generalises, notably to Co-OT, a known result for OT and GW (Peyré et al., 2016). This is summarized in the next theorem.

Theorem 6. *Assume that the loss $\mathcal{L}$ is continuous and can be written as $\mathcal{L}(x, y) = f_1(x) + f_2(y) - h_1(x)h_2(y)$ with four functions (f_1, f_2, h_1, h_2) such that $\frac{f_1'}{h_1'}$ is invertible. Further assume that $\mathcal{L}(x, y) \underset{x\to\pm\infty}{\longrightarrow} +\infty$. For fixed $((\boldsymbol{T}^{e,b})_{e\in[\![1,E]\!]})_{b\in[\![1,B]\!]}$, the optimal solution $\boldsymbol{X}^*_{i_1,\dots,i_D}$ of Problem (5.8) reads,*

$$\boldsymbol{X}^*_{i_1,\dots,i_D} = \left(\frac{\nabla f_1}{\nabla h_1}\right)^{-1} \left(\sum_{b=1}^{B} \lambda_b \sum_{k_1,\dots,k_D=1}^{K_1^b,\dots,K_D^b} h_2(\boldsymbol{X}^b_{k_1,\dots,k_D}) \prod_{d=1}^{D} \frac{T^{f(d),b}_{i_d,k_d}}{\boldsymbol{a}^{f(d)}_{i_d}}\right), \tag{5.9}$$

for all $(i_d \in [\![1, I_{f(d)}]\!])_{d\in[\![1,D]\!]}$. In particular, when $\mathcal{L}$ is the squared euclidean distance,

$$\boldsymbol{X}^*_{i_1,\dots,i_D} = \sum_{b=1}^{B} \lambda_b \sum_{k_1,\dots,k_D=1}^{K_1^b,\dots,K_D^b} \boldsymbol{X}^b_{k_1\dots k_D} \prod_{d=1}^{D} \frac{\boldsymbol{T}^{f(d),b}_{i_d k_d}}{\boldsymbol{a}^{f(d)}_{i_d}}.$$

Note that to obtain a barycenter using loss functions that are not covered by Theorem 6, for example the absolute loss, one can resort to a gradient based optimization scheme.

5.6 Experiments

In this section, we illustrate the interest of the OTT formulation on two different tasks[1]. First, following the success of OT in Domain Adaptation (Courty et al., 2017b), we propose to predict

[1]The code to reproduce all the experiments will be available on the github:`https://github.com/Hv0nnus`

the genres of recent movies based on labeled older movies by relying only on users preferences. We advantageously use a 3D-tensor formulation to take into account the particularity of each user. In a second experiment, we use the OTT barycenter formulation in a Comparison-Based Clustering task.

Even if the computation of the exact value of the distance is never necessary in the experiments, the sampling scheme used to approximate the gradient can be used to approximate this distance, as done and explained in Chapter 4 for GW.

Because the interest of the stochastic Mirror Descent algorithm was already widely explored in the previous chapter, we focus on providing interesting applications to our OTT formulation. Nevertheless, an experiment specific to our new formulation is available in the Appendix D.5.1, which shows that the gradient is well approximated with a few samples even when $D > 2$.

5.6.1 Domain Adaptation (DA)

We address here a DA task on the Movielens dataset (Harper and Konstan, 2015). The goal is to adapt a model learned on old movies (source) to predict the genres of new movies (target).

Datasets. We build a 3D-tensor $\boldsymbol{X}^1$ based on the ratings of the users. The entry (i, j, k) of $\boldsymbol{X}^1$ is 1 if the user i preferred the movie j over the movie k, -1 if the movie k is preferred over the movie j and 0 if the user i cannot choose. As the users of the database did not rate every movie, we use the 0.33 percentile of their personal rates as a default rating. For both the new and old movies, we identify 4 different groups of movies: *Thriller/Crime/Drama* (T), *Fantasy/Sci-Fi* (F), *War/Western* (W), and *Children's/Animation* (C). We then create 6 pairwise binary classification datasets of 200 movies each by selecting 2 classes among the four aforementioned ones. We assume that we have access to all the labels for the old movies (source) but only to a single label per class, randomly selected, for the new movies (target). The goal is to learn a model that is as accurate as possible on the target. Since many movies have a small number of ratings and many users only rated a few movies, we focus on the 100 users with the highest number of ratings and the 200 most rated films for those users.

Baselines. Even though OTT, to the best of our knowledge, is the first algorithm that allows a direct DA on such tensor-based datasets, we still propose various baselines by reducing the 3D-tensors into matrices by averaging along one dimension. **Rdm** is a first naive baseline that simply outputs random labels. **SVM** applies a SVM (Cortes and Vapnik, 1995) classifier only on the target domain, by using the average pairwise movies matrix as features. **S-GWL** (Xu et al., 2019a) interprets the average pairwise movies matrix as an adjacency matrix of a graph and matches the nodes of the two graphs. **GW** (Peyré et al., 2016) solves the GW problem on the average pairwise movies matrix. **Co-OT** takes the average over one movie dimension, contrary to the three baselines above, which leads to a 2D-tensor (movies, users). The two axes are then mapped jointly between the new and old movies. For every method that provides a transport plan $\boldsymbol{T}$ between the movies, the class of a target movie y^t_j is predicted via label propagation (Redko et al., 2019a) of the source label y^s: $y^t_j = \boldsymbol{T}^2_{j\bullet} y^s$. The stochastic methods

Table 5.1: *Accuracy on 6 DA tasks with the hyperparameters found using the unsupervised proposed method. To evaluate the best possible performance reachable by each method, AVG^{best} displays the accuracy with the best hyperparameters using the ground truth of the target domain.* *Cited on pages [96,96,96]*

Datasets	Rdm	SVM	S-GWL	GW	Co-OT	OTT
T,F	50.4±3.5	62.5	63.0	62.0	72.0	**80.8**±0.7
T,C	50.6±3.0	69.0	77.0	78.0	83.0	**97.0**±0.2
T,W	51.2±4.0	32.5	61.0	63.0	65.5	**71.3**±5.0
F,C	49.4±2.3	**74.5**	72.0	74.0	74.0	70.2±4.0
F,W	49.6±3.4	53.0	53.0	60.5	47.0	**67.9**±2.3
C,W	49.2±3.3	60.0	57.0	52.0	67.5	**76.8**±6.4
AVG	50.0±3.3	58.6	63.8	64.9	68.2	**77.3**±3.1
AVG (best)	50.0±3.3	58.6	66.3	71.0	70.7	**78.9**±2.9

are run 10 times and the mean and standard deviation are reported.

Experimental setup and hyperparameter tuning. As the initialization is key to avoid local minima, we take advantage of both the labels and our stochastic algorithm by sampling only the labelled points in the source and target for the first gradient estimation. The squared euclidean loss is used for $\mathcal{L}$ and we estimate the gradient of OTT using $M = 1000$ samples. P is set to 1000 iterations in Algorithm 6. For each method that uses the OT Sinkhorn solver, notably OTT, we replace it with the semi-supervised algorithm OTDA proposed by Courty et al. (2017b), which adds a $l_p - l_1$ regularization to benefit from the available source labels. In DA, tuning the hyperparameters is often key as there is not enough target labeled movies. As the goal of DA is to reduce the divergence between the two datasets (Ben-David et al., 2007; Redko et al., 2019c), we can use the distance between the source and the target as a criterion to choose the hyperparameters for each method. The Kullback-Leibler regularization parameter ϵ of the Sinkhorn method (Cuturi, 2013) is selected in the range $[10^{-5}, 10^{2}]$ and the class regularization η of OTDA (Courty et al., 2017b) in $[10^{-4}, 10^{1}]$. The hyperparameters selection is limited to 24 hours for each method and dataset.

Results. The accuracy of each method is reported in Table 5.1. OTT achieves better performances than the other baselines on 5 out of 6 datasets. This result was expected as OTT is the only method which takes full advantage of the 3D structure of the data. Interestingly, even by using the ground truth over the target domain to tune the hyperparameters of the baselines (that would be cheating), the line AVG (best) of Table 5.1 shows that OTT still behaves better.

We now analyze the impact of the different hyperparameters on the accuracy. We report the results on each dataset in the supplementary material and only consider the average, representative, trend in Figure 5.3. The leftmost plot displays the accuracy for increasing values of the KL regularization parameter ϵ. The black markers correspond to the lowest achieved distance for each method. It is worth noting that this usually corresponds to a reasonable

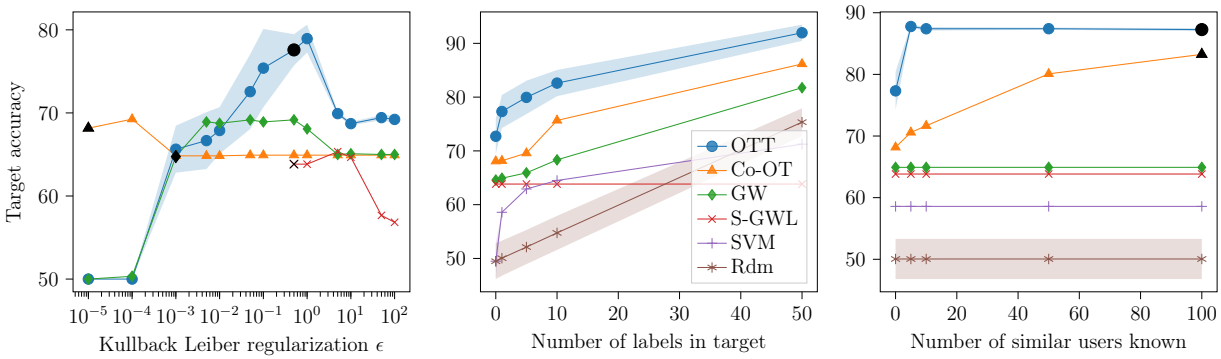


Figure 5.3: *Target accuracy averaged over all the datasets. The shadow area represents the standard deviation for the stochastic methods. (**Left**) Target accuracy for various values of ϵ. The black symbols correspond to the value of ϵ associated with the lowest distance on average of each method. (**Middle**) Target accuracy for an increasing target supervision. (**Right**) Target accuracy for an increasing number of similar known users who rated both old and new movies. Again, the black symbols correspond to the lowest distances.*

Cited on pages [96,96,96]

accuracy, which supports our hyperparameter tuning procedure. We notice a similar behaviour for the η parameter of OTDA as reported in the supplementary material. In Figure 5.3 (middle), we report the target accuracy with respect to the number of target labels available. We can notice that OTT is always better, even in the completely unsupervised scenario. Lastly, in the experiments reported in Table 5.1, we never use the fact that the users comparing the movies are the same for both old and new movies. Here, we study the impact of making this information available. To this end, we fix the transport plan for an increasing number of users. Figure 5.3 (right) shows that such an additional information can greatly improve the target accuracy of the methods that can handle it, especially OTT. Interestingly, as indicated with the black marker, the smallest distance is achieved with the highest number of similar users, which corresponds to the highest number of constraints on the users transport plan. This supports the key assumption of this experiment: a good matching between users leads to a better matching of similar movies. This also highlights the limit of the proposed solver as it may struggle to find the global minimum.

5.6.2 Comparison based clustering using OTT barycenters

In this second series of experiments, we show that OTT is competitive for addressing an unbalanced comparison-based clustering task. Comparison-based learning deals with the problem of learning from examples when neither an explicit representation nor a pairwise distance matrix is available (Vikram and Dasgupta, 2016; Ukkonen, 2017; Emamjomeh-Zadeh and Kempe, 2018; Ghoshdastidar et al., 2019; Perrot et al., 2020). Instead, it is assumed that only triplet comparisons of the form "is x_i closer to x_j than to x_k?" are available. This field stems from the fact that relative judgments are usually easier than absolute ones for human observers (Shepard,

Table 5.2: *ARI for unbalanced comparison-based clustering tasks on MNIST dataset. Each line corresponds to the average over 10 different combinations of classes, each run 10 times.*
Cited on page [99]

nb. examples per class	AddS3	AddS3$_s$	t-STE	t-STE$_s$	OTT
200,20,20	0.43±0.12	0.8±0.17	0.56±0.07	**0.91**±0.04	**0.91**±0.04
30,3,1	0.28±0.09	0.82±0.17	0.49±0.14	**0.91**±0.14	0.89±0.14
30,3,3	0.37±0.13	0.78±0.23	0.52±0.09	**0.93**±0.19	0.87±0.19
300,30,10	0.28±0.04	0.83±0.07	0.48±0.1	**0.89**±0.04	**0.89**±0.04
AVG	0.34±0.09	0.81±0.16	0.51±0.1	**0.91**±0.08	0.89±0.1

1962; Young, 1987; Stewart et al., 2005). For example, triplet-based queries are easier to answer than exact distance estimations. Given a set of examples and a given number of triplet comparisons, a dataset can be represented as a third order tensor where the entry (i, j, k) contains 1 if example x_i is closer to x_j than to x_k and -1 otherwise. In comparison based clustering, the goal is to identify relevant groups in the examples, using only the information contained in the aforementioned tensor. As the three dimensions of the cubic tensor correspond to the same points we will use OTT_{111}

Setting. To show the interest of our method for clustering unbalanced triplet datasets, we take inspiration from the experimental setup of Perrot et al. (2020). For a given dataset, we find the OTT$_{111}$ barycenter ($b = 1$) of size (I_1, I_1, I_1) where I_1 is the number of clusters that we are looking for. The intuition is that similar examples should be sent by the transport plan to the same point in the barycenter since the latter summarizes the initial points.

In this experiment, we consider some 3-class unbalanced subsamples of the MNIST dataset (LeCun et al., 1998). For a given number of examples per class (for example, 200,20,20), we consider 10 random draws for the 3 classes and for each of these, further consider 10 random draws for the actual images. Given N points in each unbalanced dataset, we randomly select $N \log(N)^3$ triplets of the form $d(x_i, x_j) > d(x_i, x_k)$ as suggested by Perrot et al. (2020). The distance between two digits is the euclidean distance after an UMAP projection in 2 dimensions. To simulate a real dataset, some noise is artificially added by randomly flipping $d(x_i, x_j) > d(x_i, x_k)$ to $d(x_i, x_j) < d(x_i, x_k)$ with probability 0.1 for each triplet selected.

Baselines. We use two main triplet clustering baselines: (i) **t-STE** which projects the triplets into a vector space followed by k-means (Lloyd, 1982), and (ii) **AddS3** (Perrot et al., 2020) which estimates a pairwise similarity matrix also followed by k-means (Lloyd, 1982).

As the OT formulation requires the marginal as prior, we assume that the proportions of the different clusters are known. To stay fair in the comparison, we propose two variants (AddS3$_s$, t-STE$_s$) of the baselines where we replace the k-means step by an OT barycenter step which takes the marginal information into account.

We use the squared euclidean loss for OTT to take advantage of the closed form derived in

Theorem 6. We use default hyperparameters, reported in the supplementary material, for t-STE, AddS3, and OTT with the KL regularization parameter set to $\epsilon = 0.1$. To ensure convergence, we also set the number of samples $M = 100$ and the number of iteration $S = 500$ between each of the 20 barycenter updates.

The Adjusted Rand Index (ARI) (Hubert and Arabie, 1985) between the prediction and the ground truth is displayed in Table 5.2. The choice of the unbalanced setting is motivated by the fact that the two other baselines do not take into account this unbalancedness information during their first step, while OTT directly uses the entire information as it has only one step.

Results. Overall, OTT has better performances than AddS3$_s$ on average on every dataset while being slightly worse than t-STE$_s$. Furthermore, for both AddS3 and t-STE, using the unbalancedness information improves the performances. The closeness between our approach and t-STE$_s$ is further investigated in the supplementary material, where we show a theoretical connection between t-STE and the OTT barycenter when the cross entropy is used as a loss and the transport plan is fixed.

5.7 Conclusion

In this chapter, we presented Optimal Tensor Transport (OTT), a new OT formulation. It can be used to align high dimensional tensors using one or several transport plans. This formulation generalizes various existing OT problems, such as GW and Co-OT. Furthermore, we proved that OTT is a proper distance which may be of great interest for tensor comparison. We also proposed an efficient algorithm to solve the underlying problem and demonstrated the competitiveness of OTT in Domain Adaptation and Comparison-based clustering.

While our new approach unlocks new applications, this comes with a cost. First, despite having access to a solver that drastically reduces the computational complexity of the formulation, it still does not scale so well empirically and may not be practical for larger datasets. Furthermore, it requires access to weights for each of the tensor dimensions, which might not always be available in practice. Finally, we leave for future work a natural extension, Fused-OTT, inspired by Vayer et al. (2018), that would combine several OTT problems together. This approach could allow us to align datasets that are independently represented by multiple tensors of potentially different orders.

Conclusion

This thesis explores the interest of using the Optimal Transport theory in Machine Learning. More specifically, it focuses on the ground metric of the Optimal Transport problem. The choice of this ground metric is a key element in any OT-based application and our claims is that the (squared) Euclidean distance is too often selected as a default choice. The first part of this thesis tackles this problem by learning a Mahalanobis distance in the context of a Domain Adaptation task. In this particular setting, we show that OT can use advantageously the available labels of the source domain to learn a suitable distance. This joint learning process between the transport plan and the metric leads to an algorithm called *Metric Learning for Optimal Transport* (MLOT). We perform experiments in Domain Adaptation that show the interest of optimizing such a ground metric. While MLOT has been designed in the context of a DA setting, it is worth noting that in any Machine Learning application, choosing the right cost function is tricky. To address this task, we propose a general way to handle several cost functions, or cost matrices, by looking at the worst case scenario. In this setting, the transport plan found does not rely upon only one cost matrix and is more robust to noise. Based on this formulation, a notion of stability for cost matrices is proposed in order to be able to choose the most stable cost matrix. It corresponds to is a matrix with an associated Wasserstein distance that does not change much when looking at the worst case in a Mahalanobis ball around it. Experiments show a correlation between this notion of stability and the noise sensitivity of the considered matrices.

In the second part of the thesis, we focus on extensions of the OT formulation, notably the Gromov Wasserstein distance able to compare distributions that do not necessarily lie in the same space. To approximate efficiently the (hard) Gromov Wasserstein problem, the existing solvers rely on particular loss functions. To address this limitation, we propose a fast stochastic algorithm, able to handle arbitrary losses. This method comes with a convergence bound to a stationary point which covers the existing EGW method in the concave case. When the number of samples is reduced to one, a very fast variant can be used, relying on the efficient 1D OT solver. We highlight the interest of using various losses in a graph classification experiments. Based on this scalable algorithm, we introduce a generalization scheme encompassing both the Wasserstein distance and the Gromov Wasserstein distance as well as the Co-OT distance. The resulting formulation is called *Optimal Tensor Transport*. This new distance can handle tensors of arbitrary size instead of using only positions in a vector space or pairwise distances. We perform experiments including comparison-based learning tasks in a triplet-based setting where

the points are defined only by theirs relations with two other points.

As often in research, this thesis might open more questions than it answers. Some possible promising research directions are given below, for each of the different chapters of the manuscript.

Metric Learning for Optimal Transport A natural extension of our method would be a "Deep" version of MLOT to allow its use directly on the original images in computer vision tasks. Instead of minimizing the objective function with respect to the Mahalanobis metric, we could instead minimize the entire Deep Neural Network (NN) that generates the feature space. Thus, the metric learned would be the entire NN instead of only a linear function. Generally, an existing NN, already trained on a huge dataset such as Image Net (Deng et al., 2009), is used and then the weights of this NN are only fine-tuned. With this setting, the choice of the hyperparameters are more complex. First, there might be more hyperparameters due to the NN. But more importantly, the computational time would increase a lot and the reverse validation used in this chapter might take too long to reasonably cover the large hyperparameter grid. Even if several hyperparameters have to be set by default, such an approach would probably lead to better results for the classification of images.

The second possible extension is to use an automatic differentiation software for the OTDA algorithm, at each iteration of MLOT, to keep track of the impact of $\boldsymbol{L}$ on the transport plan $\boldsymbol{T}$. In this case, instead of simply alternating the optimization between the transport plan $\boldsymbol{T}$ and $\boldsymbol{L}$, $\boldsymbol{T}$ would be a non-fixed variable depending on $\boldsymbol{L}$. Such an approach might lead to a better minimum than a naive alternate optimization as the problem is non convex. The main problem is the time and the memory required, as OTDA uses several calls of the Sinkhorn algorithm.

A last possible extension of MLOT would be to handle the semi-supervised setting where some labels are available in the target domain. In this case, the same ML algorithm could be used in the target domain.

A Swiss Army Knife for Minimax Optimal Transport While the cutting-set method proposed is very general, a "small" minimax problem has still to be solved at each iteration. In practice, we use only specific sets of cost matrices (Mahalanobis-based set or finite number of matrices) and for other type of sets, the algorithm has to be modified. For instance, one might resort to a Frank-Wolfe algorithm as used to generate Figure 3.1, while we used only linear solver for the experiments. Thus, a general method to solve such a part of the algorithm would be interesting to explore.

The proposed notion of Wasserstein stability might be more interesting if the search for a stable cost matrix was not reduced to a finite number of cost matrices. Indeed, in practice, we only select a certain number of cost matrices and test the stability for each of them. We could imagine to search for the most stable cost matrix in an infinite set, for instance in the polytope generated by some matrices.

Sampled Gromov Wasserstein One of the main issues of the convergence proof to a stationary point provided in this chapter is the assumption that the Sinkhorn algorithm outputs the exact $\boldsymbol{T}^*$ of the entropy regularized OT Problem 1.8. In practice, as the number of iterations is limited, the result is only an approximation which does not fully respect the marginal constraints. Thus, the bound is not totally correct and might be improved in a future work. Moreover, as the speed of convergence heavily depends on ϵ, we might replace the Sinkhorn maximum number of iterations P by a tighter value depending on ϵ. To finish, the bound related to the sampling part can be greatly improved, instead of relying on the worst case scenario, we could use some tight concentration inequalities.

In a very short period of time, during the publications of the paper associated with this chapter, two other works (Sato et al., 2020; Chowdhury et al., 2021) used the 1D OT solver by defining each point by their distance to another point. Thus, this general idea might be a promising future line of research in the Gromov Wasserstein field.

Optimal Tensor Transport An interesting line of work would be to prove the convergence of the proposed algorithm to stationary points. One of the convergence proof for the Mirror Descent algorithm, in the wide literature, might cover the particular case of the proposed algorithm. Such a proof would also cover the *Sampled Gromov Wasserstein* algorithm as a particular case.

A straightforward extension of the OTT formulation is the "Fused"-OTT, similarly to the Fused-Gromov Wasserstein (Vayer et al., 2018). It would allow to align datasets that are represented by several tensors with different orders instead of only one. The algorithm to solve such a problem would not change much, as the gradient could still be interpreted as a sum of expectations.

We hope that this thesis was interesting and more importantly will allow the emergence of new ideas and create new links between the different fields of statistics, optimization, Machine Learning and Optimal Transport.

Appendix A

Appendix of Metric Learning for Optimal Transport

A.1 Domain Adaptation experiments

This section is dedicated to give further details on the Domain Adaptation experiment of Chapter 2.

A.1.1 Complete table of the main DA experiment

Table A.1 shows the entire result of the main DA experiments of Chapter 2.

A.1.2 Metric Learning and OTDA separately

To show the advantage of jointly optimizing the metric and the transport, we compare MLOT against a Metric Learning algorithm (LMNN) followed by some Optimal Transport method (OTDA). Those experiments have been run with the default hyperparameters and are not cross-validated. Table A.2 shows the corresponding results with and without randomized-PCA. In both cases, the alternate optimization is better than a simple combination of Metric Learning and Optimal Transport.

Non-linear version of MLOT

In MLOT, the transformations learned are linear, $\boldsymbol{L}_s$ and $\boldsymbol{L}_t$ are matrices. In this paragraph, we shortly explore the use of non-linear transformations instead of $\boldsymbol{L}_s$ and $\boldsymbol{L}_t$. To do so, a new implementation using the automatic differentiation of PyTorch (Paszke et al., 2017) framework has been designed. To allow a fair comparison, the two versions, $\text{MLOT}^{\text{linear}}$ and $\text{MLOT}^{\text{non-linear}}$ as been run with Pytorch. The optimizer used is Adam (Kingma and Ba, 2014) with default momentum, the learning rate is cross-validated. The two matrices $\boldsymbol{L}_s$ and $\boldsymbol{L}_t$ are replaced by a 4

	Dataset	NA	LMNN	SA	CORAL	TCA	OT	OTDA	OTDA$_\mathbf{p}$	JDOT	MLOT
SURF	A→C	26.0	40.3	40.2±0.2	25.4	40.0	33.9	40.2	39.4±0.5	39.9	**42.3**±0.6
	A→D	25.5	36.9	39.3±2.5	26.8	31.8	30.6	40.1	39.6±1.1	37.6	**40.8**±0.8
	A→W	29.8	38.0	39.9±1.3	26.8	**41.7**	32.5	37.3	39.8±0.9	38.0	41.3±1.1
	C→A	23.7	46.0	41.3±1.1	23.6	39.8	41.0	**52.7**	48.5±0.7	48.1	51.5±0.8
	C→D	25.5	45.9	45.4±1.2	26.1	44.6	36.9	47.8	51.4±1.4	49.7	**52.2**±1.3
	C→W	25.8	41.7	36.6±1.1	23.7	36.9	28.1	**46.4**	45.8±1.4	43.4	45.9±0.8
	D→A	28.5	31.1	35.4±1.0	28.8	32.9	29.3	32.4	37.8±1.0	32.8	**37.8**±0.7
	D→C	26.3	30.7	32.3±0.6	30.0	31.5	31.7	32.0	33.5±0.7	31.7	**34.4**±0.5
	D→W	63.4	77.3	88.5±1.1	84.4	84.7	**88.8**	**88.8**	87.5±1.2	82.7	87.8±0.7
	W→A	23.0	32.3	32.6±0.5	26.2	29.4	34.1	33.7	37.6±0.6	37.6	**38.0**±0.8
	W→C	19.9	30.4	29.0±0.6	22.6	29.2	30.1	**34.1**	33.3±0.5	33.1	33.2±0.6
	W→D	59.2	86.6	89.5±1.0	84.1	91.7	89.2	**92.4**	91.8±1.2	89.8	90.8±0.8
	AVG	31.4	44.8	45.8±1.0	35.7	44.5	42.2	48.2	48.8±0.9	47.0	**49.7**±0.8
DeCAF6	A→C	71.7	80.6	81.1±0.3	72.2	77.7	82.2	80.5	79.7±0.3	**83.1**	81.6±0.4
	A→D	73.9	81.5	83.4±0.7	72.0	**84.1**	77.7	76.4	73.9±1.3	70.7	75.3±0.8
	A→W	68.1	72.5	74.7±1.0	64.1	71.9	71.2	71.9	75.0±0.6	**76.9**	72.1±0.3
	C→A	87.3	**91.4**	87.9±0.4	81.3	89.1	90.4	86.8	87.6±0.4	90.5	90.8±0.1
	C→D	79.6	79.0	**87.1**±0.7	80.3	82.8	82.8	80.9	80.1±0.5	85.4	83.3±0.7
	C→W	72.5	78.0	**85.9**±0.7	68.8	79.7	78.0	80.3	78.5±0.4	76.9	79.7±0.3
	D→A	49.9	70.1	84.3±0.5	78.0	**86.0**	84.4	84.2	82.6±0.4	64.9	85.0±0.3
	D→C	42.0	63.7	72.2±0.4	71.8	75.2	81.2	80.9	81.0±0.3	64.8	**83.2**±0.3
	D→W	91.5	95.6	99.2±0.2	**99.3**	99.0	99.3	99.3	98.9±0.3	98.6	98.6±0.1
	W→A	62.5	73.4	76.1±0.6	69.7	81.8	81.5	81.3	77.4±0.3	64.4	**85.8**±0.4
	W→C	55.3	67.0	72.7±0.4	69.1	73.6	78.6	75.7	78.0±0.4	62.0	**81.1**±0.2
	W→D	98.1	**100.**	100.±0.0	100.	100.	100.	100.	99.0±0.3	100.	99.4±0.0
	AVG	71.0	79.4	83.7±0.5	77.2	83.4	83.9	83.2	82.6±0.5	78.2	**84.7**±0.3
Office31	A→D	59.6	50.6	**60.6**±0.5	55.6	58.0	56.8	55.4	53.6±0.0	58.6	54.7±0.1
	A→W	54.0	51.8	**56.3**±0.2	56.1	52.7	49.2	50.8	53.3±0.1	51.9	54.1±0.2
	D→A	42.4	48.0	45.0±0.1	42.2	44.0	48.8	48.1	48.3±0.1	45.7	**50.4**±0.0
	D→W	90.9	94.7	93.2±0.1	91.7	91.7	92.1	**95.0**	94.0±0.1	90.9	92.6±0.1
	W→A	40.8	43.8	45.0±0.1	41.0	42.5	**48.3**	46.9	46.1±0.1	44.4	47.6±0.1
	W→D	97.8	**99.0**	98.6±0.0	98.0	95.6	96.8	95.4	96.0±0.0	94.6	97.7±0.1
	AVG	64.3	64.7	**66.5**±0.2	64.1	64.1	65.3	65.3	65.2±0.1	64.4	66.2±0.1
All datasets	AVG	53.8	62.6	65.1 ± 0.6	58.0	64.0	63.5	65.6	65.6 ± 0.6	63.0	**67.0** ± 0.5

Table A.1: *Accuracy of all the methods on 3 different types of features. The best method for each dataset is in bold.*

Cited on pages [42,105]

layers neural network with sigmoid as activation function for MLOT$^{\text{non-linear}}$. Compared to the setting of MLOT, the number of iterations is no longer cross-validated but two regularization terms are added which compute the squared Frobenius norm between the initial space after the PCA and the space transform by the two networks. This regularization ensures to stay close to the initialization. Similarly to the initial MLOT setup, both networks are learned between each

Table A.2: *Comparison between MLOT and ML + OTDA$_p$ on Office-Caltech dataset with SURF features with default hyperparameters.*

Dataset	ML + OTDA$_\mathrm{p}$	MLOT	ML + OTDA$_\mathrm{p}$ (rdm)	MLOT (rdm)
AVG	48.9	**50.2**	49.2 ± 0.9	**50.6** ± 0.9

Table A.3: *Linear and non-linear version of MLOT with Pytorch on SURF features.*

Cross-validation using	target pseudo-labels		target true labels	
Dataset	MLOT^{linear}	MLOT$^{\text{non-linear}}$	MLOT^{linear}	MLOT$^{\text{non-linear}}$
A→C	40.5	38.9	43.1	42.9
A→D	39.5	40.1	45.9	49
A→W	42.7	40	52.9	52.5
C→A	53.1	52.9	53.4	55.7
C→D	51.6	49	51.6	55.4
C→W	43.1	43.1	56.6	56.9
D→A	36.6	37.3	43.2	40.9
D→C	34.3	34.4	36.7	37.1
D→W	89.2	88.1	90.8	90.8
W→A	34.8	36.1	44.5	43.8
W→C	31.8	33.6	36.6	37.2
W→D	89.8	91.1	92.4	92.4
AVG	48.9	48.7	54.0	54.6

Optimal Transport computation. Table A.3 reports the result for the Office-Caltech dataset on SURF features. This PyTorch version has worst performances than the initial setup of MLOT which could be due to the optimizer or the two regularization terms or the optimization of $\boldsymbol{L}_t$ that was initially fixed. However, this method still gives competitive result. More importantly, Table A.3 shows that a non-linear network does not give better result on average. An advantage of those deep formulations is the ability to fine tune the Neural Network that extract the features. But the comparison with the other classical methods would be complex as they would not rely on the same feature space.

Appendix B

Appendix of Swiss-Army Knife for Optimal Transport

This appendix contains the proofs for the different theoretical results of the chapter, as well as more details on the experimental part provided for the sake of reproducibility.

B.1 Proofs from Section 3.3.2

Claim (footnote 1 in Chapter 3) $\mathcal{C}$ defined in (3.3) is a convex compact set.

Proof. Denoting $\mathcal{F}(\mathcal{X} \times \mathcal{Y}, \mathbb{R})$ the set of real valued functions on $\mathcal{X} \times \mathcal{Y}$, let:

$$\begin{aligned} \Phi : \mathbb{R}^{D\times D} &\mapsto \mathcal{F}(\mathcal{X} \times \mathcal{Y}, \mathbb{R}) \\ \mathbf{M} &\mapsto c^{\mathbf{M}} : (\boldsymbol{x}, \boldsymbol{y}) \mapsto (\boldsymbol{x} - \boldsymbol{y})^\top \mathbf{M}(\boldsymbol{x} - \boldsymbol{y}) \end{aligned}$$

Notice that Φ is linear and its domain is a finite dimensional vector space, hence Φ is continuous. Moreover, we have $\mathcal{C} = \Phi\left(\mathcal{B}_p^D\right)$ where

$$\mathcal{B}_p^D := \{\mathbf{M} \in \mathbb{R}^{D\times D}; \|\mathbf{M}\|_p \leq 1\}$$

is the unit ball of norm $\|.\|_p$, which is compact and convex. As a result, $\mathcal{C}$ is a convex compact set. □

Proposition 6. *Let $\mathcal{C}$ be defined as in* (3.3) *for $\mathbf{M} \in \mathcal{S}_+^{D\times D}$. Then, $\mathcal{C}$ is a convex compact set of cost functions and the following holds:*

$$\max_{c\in\mathcal{C}} \mathop{\mathbb{E}}_{\boldsymbol{x},\boldsymbol{y}\sim\pi}[c(\boldsymbol{x}, \boldsymbol{y})] = \max_{\mathbf{M}\in\mathcal{S}_+^{D\times D}\ \|\mathbf{M}\|_p\leq 1} \langle \mathbf{V}_\pi, \mathbf{M}\rangle = \|\mathbf{V}_\pi\|_q$$

implying $\mathrm{RKP}(\Pi, \mathcal{C}) = \min_{\pi\in\Pi} \|\mathbf{V}_\gamma\|_q$. *Furthermore, for any $\pi \in \Pi$,*

$$\mathbf{M}^* = \operatorname*{argmax}_{\mathbf{M}\in\mathcal{S}_+^{D\times D}\ \|\mathbf{M}\|_p\leq 1} \langle \mathbf{V}_\pi, \mathbf{M}\rangle = \left(\frac{\mathbf{V}_\gamma}{\|\mathbf{V}_\gamma\|_q}\right)^{\frac{q}{p}} \tag{B.1}$$

verifies $\|\mathbf{M}^*\|_p = 1$. *In particular, for* $p = \infty$, $\min_{\pi\in\Pi} \|\mathbf{V}_\pi\|_1 = \mathcal{W}_2^2(\mu,\nu)$, *i.e., , we recover the classic 2-Wasserstein distance.*

Proof. With the notations used to prove that $\mathcal{C}$ is a convex compact, adding the PSD constraint on $\mathbf{M}$ can be done by considering the image of $\mathcal{B}_{p+}^D := \mathcal{B}_p^D \cap \mathcal{S}_+^{D\times D}$ by mapping Φ. $\mathcal{B}_{p+}^D$ is a convex compact set as it is the intersection of a convex compact set and a convex cone (the PSD cone). For fixed $\pi \in \Pi$, we compute the maximum of $\mathop{\mathbb{E}}_{\boldsymbol{x},\boldsymbol{y}\sim\pi}\left[c^{\mathbf{M}}(\boldsymbol{x},\boldsymbol{y})\right]$ over $\mathbf{M} \in \mathcal{B}_{p+}^D$.

$$\begin{aligned}
\max_{\mathbf{M}\in\mathcal{B}_{p+}^D} \mathop{\mathbb{E}}_{\boldsymbol{x},\boldsymbol{y}\sim\pi}[c(\boldsymbol{x},\boldsymbol{y})] &= \max_{\mathbf{M}\in\mathcal{B}_{p+}^D} \mathop{\mathbb{E}}_{\boldsymbol{x},\boldsymbol{y}\sim\pi}\left[(\boldsymbol{x}-\boldsymbol{y})^\top \mathbf{M}(\boldsymbol{x}-\boldsymbol{y})\right] \\
&= \max_{\mathbf{M}\in\mathcal{B}_{p+}^D} \mathop{\mathbb{E}}_{\boldsymbol{x},\boldsymbol{y}\sim\pi}\left[\mathrm{Tr}\Big((\boldsymbol{x}-\boldsymbol{y})(\boldsymbol{x}-\boldsymbol{y})^\top \mathbf{M}\Big)\right] \\
&= \max_{\mathbf{M}\in\mathcal{B}_{p+}^D} \mathop{\mathbb{E}}_{\boldsymbol{x},\boldsymbol{y}\sim\pi}\left[\Big\langle (\boldsymbol{x}-\boldsymbol{y})(\boldsymbol{x}-\boldsymbol{y})^\top, \mathbf{M}\Big\rangle\right] \\
&= \max_{\mathbf{M}\in\mathcal{B}_{p+}^D} \langle \mathbf{V}_\pi, \mathbf{M}\rangle .
\end{aligned}$$

where we used properties of the trace operator, the linearity of the expectation and the definition of $\mathbf{V}_\pi$.

This maximum is achieved for $\mathbf{M}^*$ verifying $\|\mathbf{M}^*\|_p = \|\mathbf{M}^*\|_p^p = \mathrm{Tr}\{(\mathbf{M}^*)^p\} = 1$. In fact, supposing this is not the case, i.e., $\|\mathbf{M}^*\|_p < 1$, then $\mathbf{M}^{**} = \frac{\mathbf{M}^*}{\|\mathbf{M}^*\|}$ verifies $\langle \mathbf{V}_\pi, \mathbf{M}^{**}\rangle > \langle \mathbf{V}_\pi, \mathbf{M}^*\rangle$, which contradicts $\mathbf{M}^*$'s optimality.
Using the equality case of the Hölder inequality for Schatten p-norms (Magnus, 1987, Theorem 5), the only PSD matrix achieving this maximum is:

$$\mathbf{M}^* = \left(\frac{\mathbf{V}_\pi^q}{\mathrm{Tr}\{\mathbf{V}_\pi^q\}}\right)^{\frac{1}{p}} = \left(\frac{\mathbf{V}_\pi}{\|\mathbf{V}_\pi\|_q}\right)^{\frac{q}{p}}$$

and the value of the maximum is $\|\mathbf{V}_\pi\|_q$. Taking the minimum over $\pi \in \Pi$, we obtain:

$$\min_{\pi\in\Pi} \max_{\mathbf{M}\in\mathcal{B}_{p+}^D} \mathop{\mathbb{E}}_{\boldsymbol{x},\boldsymbol{y}\sim\pi}[c(\boldsymbol{x},\boldsymbol{y})] = \min_{\pi\in\Pi} \|\mathbf{V}_\pi\|_q .$$

In particular, for $p = \infty$, the corresponding dual norm is $\|\cdot\|_1$, and we have:

$$\begin{aligned}
\min_{\pi\in\Pi} \|\mathbf{V}_\pi\|_1 &= \min_{\pi\in\Pi} \mathrm{Tr}\{\mathbf{V}_\pi\} \\
&= \min_{\pi\in\Pi} \mathop{\mathbb{E}}_{\boldsymbol{x},\boldsymbol{y}\sim\pi}\left[\|\boldsymbol{x}-\boldsymbol{y}\|^2\right] \\
&= \mathcal{W}_2^2(\mu,\nu).
\end{aligned}$$

This concludes the proof. □

Corollary 3 (Euclidean norm case)**.** *Let* $\mathcal{C}$ *be defined with* $p = 2$ *in* (3.3) *and let* $\mathbf{M}^* = \mathrm{argmax}_{\|\mathbf{M}\|_2\le 1} \langle \mathbf{V}_\pi, \mathbf{M}\rangle$. *Then* $\mathbf{M}^* = \dfrac{\mathbf{V}_\pi}{\|\mathbf{V}_\pi\|_2}$, *thus* $\mathbf{M}^*$ *is PSD and* $\|\mathbf{M}^*\|_2 = 1$.

Proof. $\sup_{\|\mathbf{M}\|_2\le 1} \langle \mathbf{V}_\pi, \mathbf{M}\rangle$ is achieved, without imposing that $\mathbf{M}$ is PSD, for $\mathbf{M} = \frac{\mathbf{V}_\pi}{\|\mathbf{V}_\pi\|}$ (by the equality case of the Cauchy-Schwartz inequality). This matrix is PSD as $\mathbf{V}_\pi$ is PSD, and has unit norm. □

Corollary 4. *With the assumptions from Proposition 1, the following inequality holds for any* $p \in [1, +\infty]$*:*

$$\frac{1}{D^{\frac{1}{p}}} \mathcal{W}_2^2(\mu, \nu) \leq \mathcal{W}_{\mathcal{C}}(\mu, \nu) \leq \mathcal{W}_2^2(\mu, \nu). \tag{B.2}$$

Proof. Let $\pi \in \Pi$. We have for any $p \geq 1$, $\|\mathbf{V}_\pi\|_p \leq \|\mathbf{V}_\pi\|_1$ by the monotonicity of Schatten $p-$norm, and we have $\min_{\pi \in \Pi} \|\mathbf{V}_\pi\|_1 = \mathcal{W}_2^2(\mu, \nu)$. Taking the infimum over $\pi \in \Pi$ yields the right hand side inequality in (2). To obtain the left hand side, notice that $\mathbf{A} := D^{-\frac{1}{p}} \mathbb{I}_D$ verifies $\|\mathbf{A}\|_p \leq 1$, and that $\mathbf{A}$ is PSD, so that $c^{\mathbf{A}} \in \mathcal{C}$. Thus,

$$\min_{\pi \in \Pi} \langle \mathbf{V}_\pi, \mathbf{A} \rangle \leq \mathcal{W}_{\mathcal{C}}(\mu, \nu).$$

Finally, notice that the left hand side in the previous inequality equals $\frac{1}{D^{\frac{1}{p}}} \mathcal{W}_2^2(\mu, \nu)$, thus concluding the proof. □

B.2 Proofs from Section 3.3.3

Let $\mathcal{X}$ and $\mathcal{Y}$ are identified respectively with finite sets $\{\boldsymbol{x}_i\}_{i=1}^I$ and $\{\boldsymbol{y}_k\}_{k=1}^K$, hence $\mathcal{C}$ is identified with a convex compact set of cost matrices with entries $(\mathbf{C})_{ik} = c(\boldsymbol{x}_i, \boldsymbol{y}_k)$.

Proposition 7. *Let* $\mathcal{P}$ *be a finite subset of* Π*. The problem* $\mathrm{RKP}(\mathcal{P}, \mathcal{C}) := \mathrm{RKP}(\mathrm{Conv}\,(\mathcal{P}), \mathcal{C})$ *has a saddle point* $(\boldsymbol{T}^*, \mathbf{C}^*)$ *verifying*

$$\langle \boldsymbol{T}^*, \mathbf{C}^* \rangle = \min_{\boldsymbol{T} \in \mathrm{Conv}(\mathcal{P})} \max_{\mathbf{C} \in \mathcal{C}} \langle \boldsymbol{T}, \mathbf{C} \rangle \tag{B.3}$$

$$= \max_{\mathbf{C} \in \mathcal{C}} \min_{\boldsymbol{T} \in \mathcal{P}} \langle \boldsymbol{T}, \mathbf{C} \rangle . \tag{B.4}$$

Moreover, solving $\mathrm{RKP}(\mathcal{P}, \mathcal{C})$ *is equivalent to solving*

$$\begin{aligned} \mathbf{C}^* \in & \operatorname*{argmax}_{\mathbf{C} \in \mathcal{C}, \omega \geq 0} \ \omega \\ s.t. & \langle \boldsymbol{T}, \mathbf{C} \rangle \geq \ \omega, \ \forall \boldsymbol{T} \in \mathcal{P}. \end{aligned} \tag{B.5}$$

Also, $\boldsymbol{T}^* = \sum_{l=1}^{|\mathcal{P}|} \boldsymbol{q}_l \boldsymbol{T}_l$*, where* $\{\boldsymbol{q}_l\}_{l=1}^{|\mathcal{P}|}, \sum_l \boldsymbol{q}_l = 1$*, are dual variables of* (3.5)*. In particular, solving* $\mathrm{RKP}(\Pi, \mathcal{C})$ *can be done by setting* $\mathcal{P}$ *as the set of vertices of* Π*.*

Proof. Since the set $\mathcal{P}$ is finite, $\mathrm{Conv}\,(\mathcal{P})$ is a convex compact set. Also, by definition, $\mathcal{C}$ is a convex compact set. Moreover, we note that for any $(\boldsymbol{T}, \mathbf{C}) \in \Pi \times \mathcal{C}$, the functions $\langle \boldsymbol{T}, \cdot \rangle$ and $\langle \cdot, \mathbf{C} \rangle$ are linear. By applying Sion's min-max theorem (Sion, 1958), problem $\mathrm{RKP}(\mathcal{P}, \mathcal{C}) := \mathrm{RKP}(\mathrm{Conv}\,(\mathcal{P}), \mathcal{C})$ has at least a saddle point, and any saddle point $(\boldsymbol{T}^s, \mathbf{C}^s)$ verifies:

$$\begin{aligned} \langle \boldsymbol{T}^s, \mathbf{C}^s \rangle &= \min_{\boldsymbol{T} \in \mathrm{Conv}(\mathcal{P})} \max_{\mathbf{C} \in \mathcal{C}} \langle \boldsymbol{T}, \mathbf{C} \rangle \\ &= \max_{\mathbf{C} \in \mathcal{C}} \min_{\boldsymbol{T} \in \mathrm{Conv}(\mathcal{P})} \langle \boldsymbol{T}, \mathbf{C} \rangle . \end{aligned} \tag{B.6}$$

However, for any fixed $\mathbf{C} \in \mathcal{C}$, the linearity of $\langle \cdot, \mathbf{C} \rangle$ implies that its minimum on $\mathrm{Conv}(\mathcal{P})$ is achieved on one of its vertices, i.e., :

$$\forall \mathbf{C} \in \mathcal{C} \min_{\boldsymbol{T} \in \mathrm{Conv}(\mathcal{P})} \langle \boldsymbol{T}, \mathbf{C} \rangle = \min_{\boldsymbol{T} \in \mathcal{P}} \langle \boldsymbol{T}, \mathbf{C} \rangle \Rightarrow \max_{\mathbf{C} \in \mathcal{C}} \min_{\boldsymbol{T} \in \mathrm{Conv}(\mathcal{P})} \langle \boldsymbol{T}, \mathbf{C} \rangle \quad = \max_{\mathbf{C} \in \mathcal{C}} \min_{\boldsymbol{T} \in \mathcal{P}} \langle \boldsymbol{T}, \mathbf{C} \rangle . \tag{B.7}$$

Combining (B.6) and (B.7) yields Equation (B.4).
Moreover, by the saddle point's definition, we have: $\mathbf{C}^* \in \mathrm{argmax}_{\mathbf{C} \in \mathcal{C}} \min_{\boldsymbol{T} \in \mathcal{P}} \langle \boldsymbol{T}, \mathbf{C} \rangle$. Using the fact that $\mathcal{P}$ is finite, we obtain the equivalent Problem (B.5). What is left is computing $\boldsymbol{T}^*$'s value. To this end, let us introduce $\mathcal{I}_{\mathcal{C}}$, the convex indicator function of set $\mathcal{C}$, defined by:

$$\begin{aligned} \mathcal{I}_{\mathcal{C}} : \mathbf{C} \mapsto 0 \quad & \text{if} \quad \mathbf{C} \in \mathcal{C} \\ + \infty \quad & \text{otherwise} \end{aligned}$$

Also, notice that ω is nonnegative even without imposing this condition. In fact, assuming that the cost matrices in $\mathcal{C}$ have positive values, we have $\min_{\boldsymbol{T} \in \mathcal{P}} \langle \boldsymbol{T}, \mathbf{C} \rangle \geq 0$, for all $\mathbf{C} \in \mathcal{C}$. If ω^*, the value of ω at the solution was negative, its maximality contradicts the condition $\min_{\boldsymbol{T} \in \mathcal{P}} \langle \boldsymbol{T}, \mathbf{C} \rangle \geq 0$. Hence, Problem (B.5) is equivalent to the following:

$$\begin{aligned} \max_{\mathbf{C} \in \mathbb{R}^{I \times K}, \omega \in \mathbb{R}} & \omega - \mathcal{I}_{\mathcal{C}}(\mathbf{C}), \\ & \text{s.t} \, \langle \boldsymbol{T}, \mathbf{C} \rangle \geq \omega \quad \forall \boldsymbol{T} \in \mathcal{P}. \end{aligned}$$

The Lagrangian of the previous problem is:

$$\mathcal{L}a(\mathbf{q}, \mathbf{C}, \omega) = \omega - \mathcal{I}_{\mathcal{C}}(\mathbf{C}) + \sum_{l=1}^{|\mathcal{P}|} q_l (\langle \boldsymbol{T}_l, \mathbf{C} \rangle - \omega), \tag{B.8}$$

where l indexes the finite set of matrices $\mathcal{P}$, $q_l \geq 0$ for all $l \in \{1, ..., |\mathcal{P}|\}$ denote the dual variables of the constraints, and $\mathbf{q} = (q_1, ..., q_{|\mathcal{P}|})$. A known optimization result (Boyd and Vandenberghe, 2004, Section 5.4.2) implies that that the solution to the primal, $(\mathbf{C}^*, \omega^*)$ and the solution to the dual, $\mathbf{q}^* = (q_1^*, ..., q_l^*)$ form a saddle point of the Lagrangian, which implies:

$$\mathcal{L}a(\mathbf{q}^*, \mathbf{C}^*, \omega^*) = \max_{\mathbf{C}, \omega} \mathcal{L}a(\mathbf{q}^*, \mathbf{C}, \omega) \tag{B.9}$$

Deriving the Lagrangian with respect to ω yields:

$$\sum_l \mathbf{q}_l^* = 1. \tag{B.10}$$

In addition to this condition, knowing that the value of the Lagrangian is finite at the solution, we have $\mathcal{I}_{\mathcal{C}}(\mathbf{C}^*) = 0$. Substituting the last two conditions in Equation (B.9) yields:

$$\begin{aligned} \mathcal{L}a(\mathbf{q}^*, \mathbf{C}^*, \omega^*) &= \langle \boldsymbol{T}^*, \mathbf{C}^* \rangle \\ &= \max_{\mathbf{C} \in \mathbb{R}^{I \times K}} \langle \boldsymbol{T}^*, \mathbf{C} \rangle - \mathcal{I}_{\mathcal{C}}(\mathbf{C}) \\ &= \max_{\mathbf{C} \in \mathcal{C}} \langle \boldsymbol{T}^*, \mathbf{C} \rangle \end{aligned} \tag{B.11}$$

where $\boldsymbol{T}^*$ is defined as in the proposition. Also, Equation (B.10) implies that there is at least one $l' \in \{1, ..., |\mathcal{P}|\}$ verifying $\mathbf{q}_{l'} > 0$, and hence

$$\omega^* = \langle \boldsymbol{T}_{l'}, \mathbf{C}^* \rangle = \min_{\boldsymbol{T} \in \mathcal{P}} \langle \boldsymbol{T}, \mathbf{C}^* \rangle = \min_{\boldsymbol{T} \in \mathrm{Conv}(\mathcal{P})} \langle \boldsymbol{T}, \mathbf{C}^* \rangle \tag{B.12}$$

Moreover, by the Lagrangian's definition, we have $\omega^* = \mathcal{L}a(\mathbf{q}^*, \mathbf{C}^*, \omega^*)$. This latter equation combined with (B.12) and (B.11) yields:

$$\langle \boldsymbol{T}^*, \mathbf{C}^* \rangle = \max_{\mathbf{C} \in \mathcal{C}} \langle \boldsymbol{T}^*, \mathbf{C} \rangle = \min_{\boldsymbol{T} \in \mathrm{Conv}(\mathcal{P})} \langle \boldsymbol{T}, \mathbf{C}^* \rangle \tag{B.13}$$

i.e., $(\boldsymbol{T}^*, \mathbf{C}^*)$ is a saddle point of $\mathrm{RKP}(\mathcal{P}, \mathcal{C})$. □

Proposition 8. *Let T be the number of iterations required by 4 to reach error $err(T) \leq \mathrm{thd1}$. Then,*

$$T \leq \left(\frac{\mathrm{diam}_\infty(\mathcal{C}) + \mathrm{RKP}(\mathcal{P}_0, \mathcal{C})}{2.\mathrm{thd1}} + 1 \right)^{\dim(\mathcal{C})+1}$$

where $\mathrm{diam}_\infty(\mathcal{C}) := \sup_{\mathbf{C}^1, \mathbf{C}^2 \in \mathcal{C}, i,j} \left| \mathbf{C}^1_{ik} - \mathbf{C}^2_{ik} \right|$ *and* $\dim(\mathcal{C})$ *is the dimension of the affine hull of* $\mathcal{C}$*. Also,* $\forall t \geq 0$*, we have that* $0 \leq \mathrm{RKP}(\mathcal{P}_t, \mathcal{C}) - \mathrm{RKP}(\Pi, \mathcal{C}) \leq err(t)$*.*

Proof. In this proof, we use the notation $\|\mathbf{A}\|_1 = \sum_{ik} |\mathbf{A}_{ik}|$ and $\|\mathbf{A}\|_\infty = \sup_{ik} |\mathbf{A}|_{ik}$. We note that these notations are only used in this proof and do not apply to the rest of the appendix, as they do not correspond to the Schatten-1 and ∞ norms.

We apply the result given in (Mutapcic and Boyd, 2009, Section 5.2) to our case. To this end, since our nominal problem corresponds to $\mathcal{P}_0$, we define its feasible set $\mathcal{F}_0$ as:

$$\mathcal{F}_0 = \{(\omega, \mathbf{C}) \in \mathbb{R}_+ \times \mathcal{C} | \omega \leq \min_{\boldsymbol{T} \in \mathcal{P}_0} \langle \boldsymbol{T}, \mathbf{C} \rangle\}.$$

Also, we define

$$\|(\omega, \mathbf{C})\|_\infty := |\omega| + \|\mathbf{C}\|_\infty . \tag{B.14}$$

For every $(\omega, \mathbf{C}^1), (\gamma, \mathbf{C}^2) \in \mathcal{F}_0$ and for every constraint, i.e., , for every $\boldsymbol{T} \in \mathcal{P}_0$, we have:

$$\begin{aligned}
|(\langle \boldsymbol{T}, \mathbf{C}^1 \rangle - \omega) - (\langle \boldsymbol{T}, \mathbf{C}^2 \rangle - \gamma)| &\leq \left| \langle \boldsymbol{T}, \mathbf{C}^1 \rangle - \langle \boldsymbol{T}, \mathbf{C}^2 \rangle \right| + |\omega - \gamma| && \text{(B.15)} \\
&\leq \|\boldsymbol{T}\|_1 \left\| \mathbf{C}^1 - \mathbf{C}^2 \right\|_\infty + |\omega - \gamma| && \text{(B.16)} \\
&\leq \left\| \mathbf{C}^1 - \mathbf{C}^2 \right\|_\infty + |\omega - \gamma| && \text{(B.17)} \\
&= \left\| (\omega, \mathbf{C}^1) - (\gamma, \mathbf{C}^2) \right\|_\infty && \text{(B.18)}
\end{aligned}$$

where $(\omega, \mathbf{C}^1) - (\gamma, \mathbf{C}^2) := (\omega - \gamma, \mathbf{C}^1 - \mathbf{C}^2)$. (B.15) is due to the triangle inequality, followed by the Hölder inequality to obtain (B.16). Then, since $\mathcal{P}_0 \subset \Pi$ and any matrix in Π has all of its entries bounded by 1, we obtain (B.17). Lastly, we used definition (B.14) to obtain (B.18).

To establish the bound as done in (Mutapcic and Boyd, 2009), we also need to find the radius R of a ball that contains the feasible set $\mathcal{F}_0$, and we consider the affine hull of $\mathcal{C}$ instead of $\mathbb{R}^{I \times K}$ as the space containing $\mathcal{C}$. It is then sufficient to bound the diameter of $\mathcal{F}_0$, denoted $\mathrm{diam}_\infty(\mathcal{F}_0)$ and to take half of the bound for R. To this end, for any $(\omega, \mathbf{C}^1), (\gamma, \mathbf{C}^2) \in \mathcal{F}_0$,

$$\left\| (\omega, \mathbf{C}^1) - (\gamma, \mathbf{C}^2) \right\|_\infty = \left\| (\omega - \gamma, \mathbf{C}^1 - \mathbf{C}^2) \right\|_\infty$$

$$= \left\|\mathbf{C}^1 - \mathbf{C}^2\right\|_\infty + |\omega - \gamma|$$
$$\leq \operatorname{diam}_\infty(\mathcal{C}) + |\omega - \gamma|.$$

We have:

$$\omega - \gamma \leq \omega \leq \min_{\boldsymbol{T} \in \mathcal{P}_0} \langle \boldsymbol{T}, \mathbf{C} \rangle \leq \mathrm{RKP}(\mathcal{P}_0, \mathcal{C}).$$

We can obtain this bound also for $\gamma - \omega$, hence for $|\gamma - \omega|$. Taking the supremum over $(\omega, \mathbf{C}^1), (\gamma, \mathbf{C}^2) \in \mathcal{F}_0$ (the definition of a diameter), we obtain:

$$\operatorname{diam}_\infty(\mathcal{F}_0) \leq \operatorname{diam}_\infty(\mathcal{C}) + \mathrm{RKP}(\mathcal{P}_0, \mathcal{C}).$$

We can then set radius R as half of the previous upper bound, leading to the bound on the number of iterations T. For the second result of the proposition, for any $t \geq 0$, we have:

$$\min_{\boldsymbol{T} \in \Pi} \langle \boldsymbol{T}, \mathbf{C} \rangle \leq \mathrm{RKP}(\Pi, \mathcal{C}) \leq \mathrm{RKP}(\mathcal{P}_t, \mathcal{C}),$$

where the left inequality is due to taking the maximum over $\mathcal{C}$, while the right one is due to the set inclusion $\mathcal{P}_t \subset \Pi$. Thus,

$$0 \leq \mathrm{RKP}(\mathcal{P}_t, \mathcal{C}) - \mathrm{RKP}(\Pi, \mathcal{C}) \leq \mathrm{RKP}(\mathcal{P}_t, \mathcal{C}) - \min_{\boldsymbol{T} \in \Pi} \langle \boldsymbol{T}, \mathbf{C} \rangle .$$

In Algorithm 4, the right hand side is equal to $err(t)$, which yields the result. □

B.3 Proofs from Section 3.3.4

We first prove the following lemma that will be helpful in the following proofs.

Lemma 2. *Let c and d be two positive integers. The dual of the linear program*

$$\max_{\mathbf{p} \in \Delta_d, \omega \geq 0} \omega$$
$$s.t.\ \mathbf{G}\mathbf{p} \geq \omega \mathbb{1}_c, \tag{B.19}$$

is the linear program

$$\min_{\mathbf{q} \in \Delta_c, \gamma \geq 0} \gamma$$
$$s.t. \mathbf{G}^\top \mathbf{q} \leq \gamma \mathbb{1}_d,$$

Proof. We will transform (B.19) to a standard LP formulation. To this end, let $\mathbf{v} = (p_1, ..., p_d, \omega)$, i.e the concatenation of $\mathbf{p}$ and ω. Also, we transform the equality condition $\mathbb{1}_d^\top \mathbf{p} = 1$ into the two inequalities $\mathbb{1}_d^\top \mathbf{p} \leq 1$ and $-\mathbb{1}_d^\top \mathbf{p} \leq -1$. We construct the following matrix:

$$\mathbf{F} = \begin{bmatrix} -\mathbf{G} & \mathbb{1}_c \\ \mathbb{1}_d^\top & 0 \\ -\mathbb{1}_d^\top & 0 \end{bmatrix}$$

Having $c+2$ rows and $d+1$ columns. Then, (B.19) can be re-written under the standard form:

$$\begin{aligned} \max \quad & \mathbf{e}_{d+1}^\top \mathbf{v} \\ \text{s.t} \quad & \mathbf{F}\mathbf{v} \leq \mathbf{e}_{c+1} - \mathbf{e}_{c+2} \\ & \mathbf{v} \geq 0 \end{aligned}$$

where $\mathbf{e}_i$ denotes the vectors of $\mathbb{R}^{d+1}$'s canonical basis. This latter problem has the following dual:

$$\begin{aligned} \min \quad & (\mathbf{e}_{c+1} - \mathbf{e}_{c+2})^\top \mathbf{w} \\ \text{s.t} \quad & \mathbf{F}^\top \mathbf{w} \geq \mathbf{e}_{d+1} \\ & \mathbf{w} \geq 0 \end{aligned}$$

Using the fact that

$$\mathbf{F}^\top = \begin{bmatrix} -\mathbf{G}^\top & \mathbb{1}_d & -\mathbb{1}_d \\ \mathbb{1}_c^\top & 0 & 0 \end{bmatrix}$$

and denoting $\mathbf{w} = (q_1, ..., q_c, \gamma_1, \gamma_2)$, and $\mathbf{q} = (q_1, ..., q_c)$, the dual is written:

$$\min \quad \gamma_1 - \gamma_2 \tag{B.20}$$

$$\text{s.t} \quad \mathbf{G}^\top \mathbf{q} \leq (\gamma_1 - \gamma_2)\mathbb{1}_d \tag{B.21}$$

$$\mathbb{1}_c^\top \mathbf{q} \geq 1 \tag{B.22}$$

$$\mathbf{q} \geq 0 \tag{B.23}$$

$$\gamma_1, \gamma_2 \geq 0 \tag{B.24}$$

Setting $\gamma = \gamma_1 - \gamma_2$, from (B.21) and the fact that $\mathbf{G}$ has positive elements (Frobenius products between cost matrices and transport matrices), we have $\gamma \geq 0$. Also, for γ^*, $\mathbf{q}^*$ the solution of the dual, we necessarily have $\mathbb{1}_c^\top \mathbf{q}^* = 1$. In fact, assuming that $\mathbb{1}_c^\top \mathbf{q}^* > 1$ and dividing (B.21) by $\mathbb{1}_c^\top \mathbf{q}^*$, we see that $\gamma^{**}, \mathbf{q}^{**}$ defined by $\mathbf{q}^{**} = \frac{\mathbf{q}^*}{\mathbb{1}_c^\top \mathbf{q}^*}$ and $\gamma^{**} = \frac{\gamma^*}{\mathbb{1}_c^\top \mathbf{q}^*}$ verify all the constraints, whereas $\gamma^{**} < \gamma^*$. This latter inequality contradicts the minimality of γ. Hence, the dual formulation is proven. □

Proposition 9 (Finite set $\mathcal{C}$)**.** *Let $\mathcal{C} = \mathrm{Conv}(\{\mathbf{C}_1, ..., \mathbf{C}_M\})$. Then, for $t \geq 0$, solving the problem given in* (3.5) *over $\mathcal{P}_t \times \mathcal{C}$ is equivalent to the following linear program*

$$\begin{aligned} & \min_{\mathbf{p} \in \mathbb{R}_+^M} \mathbb{1}_M^\top \mathbf{p} \\ & \text{s.t. } \mathbf{G}\mathbf{p} \geq \mathbb{1}_{|\mathcal{P}_t|}, \end{aligned} \tag{B.25}$$

where $\mathbf{G} \in \mathbb{R}^{|\mathcal{P}_t| \times M}$ is defined by $\mathbf{G}_{lm} = \langle \boldsymbol{T}_l, \mathbf{C}_m \rangle$.
Moreover,

$$\mathbf{C}^* = \frac{\sum_{m=1}^M p_m^* \mathbf{C}_m}{\sum_{m=1}^M p_m^*}, \qquad \boldsymbol{T}^* = \frac{\sum_l^{|\mathcal{P}_t|} \mathbf{q}_l^* \boldsymbol{T}_l}{\sum_l^{|\mathcal{P}_t|} \mathbf{q}_l^*},$$

where $\mathbf{p}^*$ and $\mathbf{q}^*$ are optimal solutions of (3.8) *and its dual.*

Proof. Since $\mathcal{C}$ is the convex hull of matrices $\{\mathbf{C}_1, ..., \mathbf{C}_M\}$, i.e the set of their convex combinations, problem (3.5) can be formulated as follows:

$$\max_{\mathbf{p}\in\Delta_M,\omega\geq 0} \omega$$
$$\text{s.t. } \omega \leq \sum_m p_m \langle \boldsymbol{T}_l, \mathbf{C}_m \rangle \qquad \forall 1 \leq l \leq |\mathcal{P}_t|$$

Let $\mathbf{G}_{lm}$ be the matrix whose elements are: $\mathbf{G}_{lm} = \langle \boldsymbol{T}_l, \mathbf{C}_m \rangle$. The previous problem can be re-written:

$$\max_{\mathbf{p}\in\Delta_d,\omega\geq 0} \omega$$
$$\text{s.t. } \mathbf{G}\mathbf{p} \geq \omega \mathbb{1}_{|\mathcal{P}_t|}, \tag{B.26}$$

Since the probability simplex Δ_M can be expressed as:

$$\Delta_M = \left\{ \frac{\mathbf{p}}{\mathbb{1}_M^\top \mathbf{p}}; \mathbf{p} \in \mathbb{R}_+^M \setminus \{0\} \right\}$$

the previous problem is equivalent to

$$\max_{\mathbf{p}\in\mathbb{R}_+^M,\omega\geq 0} \omega$$
$$\text{s.t. } \mathbf{G}\mathbf{p} \geq \omega \mathbb{1}_M^\top \mathbf{p} \mathbb{1}_{|\mathcal{P}_t|}.$$

By setting $\omega \mathbb{1}_M^\top \mathbf{p} = 1$ (same technique used to derive primal SVM optimization problem as a constrained norm minimization problem), which proves formulation (3.8). Also, from the change of variables that we made on $\mathbf{p}$, we obtain

$$\mathbf{C}^* = \frac{\sum_{m=1}^M \mathbf{p}_m^* \mathbf{C}_m}{\sum_{m=1}^M \mathbf{p}_m^*},$$

where $\mathbf{p}^*$ is the solution of Problem (3.8).

Now we focus on the second part of the proof, to obtain the expression of $\boldsymbol{T}^*$, the other component of the saddle point. By the result in Lemma 2, denoting $\tilde{\mathbf{q}}^*$ the dual variables of Problem (B.26), $\tilde{\mathbf{q}}^*$ is a solution to the following dual problem:

$$\min_{\mathbf{q}\in\Delta_{|\mathcal{P}_t|},\gamma\geq 0} \gamma$$
$$\text{s.t. } \mathbf{G}^\top \mathbf{q} \leq \gamma \mathbb{1}_M, \tag{B.27}$$

By the same argument used to obtain the equivalent formulation (B.25), Problem (B.27) is equivalent to:

$$\max_{\mathbf{q}\in\mathbb{R}_+^D} \mathbb{1}_{|\mathcal{P}_t|}^\top \mathbf{q}$$
$$\text{s.t. } \mathbf{G}^\top \mathbf{q} \leq \mathbb{1}_M, \tag{B.28}$$

where the components of the solution $\tilde{\mathbf{q}}^*$ by normalizing solution $\mathbf{q}^*$ of the previous problem, which yields the expression of $\boldsymbol{T}^*$. Finally, it is sufficient to notice that Problems (B.28) and (B.25) are each the dual of the other, to conclude the proof. □

To proceed for the next proposition, we recall the definition of the set

$$\mathcal{C}_{\mathbf{C}} = \{\mathbf{C} + \mathbf{E}^{\mathbf{M}} \in \mathbb{R}^{I\times K} \,|\, \mathbf{E}^{\mathbf{M}}_{ik} = (\boldsymbol{x}_i - \boldsymbol{y}_k)^\top \mathbf{M}(\boldsymbol{x}_i - \boldsymbol{y}_k); \mathbf{M} \in \mathcal{S}_+^{D\times D}; \|\mathbf{M}\|_p \leq r\}$$

for given cost matrix $\mathbf{C}$ and radius $r > 0$.

Proposition 10 (Non centered family of Mahalanobis distances)**.** *For a fixed* $\mathbf{C}$*, let* $\mathcal{C}_{\mathbf{C}}$ *be defined as in* (3.9)*. Then, for* $t \geq 0$*, solving* (3.5) *over* $\mathcal{P}_t \times \mathcal{C}_{\mathbf{C}}$*, is equivalent to solving the following convex program*

$$\min_{\boldsymbol{T}\in \mathrm{Conv}(\mathcal{P}_t)} r \|\mathbf{V}_{\boldsymbol{T}}\|_q + \sum_{ik} \boldsymbol{T}_{ik}\mathbf{C}_{ik}. \tag{B.29}$$

Moreover, if $\boldsymbol{T}^*$ *is an optimal solution of Equation* (B.29)*, then* $\mathbf{M}^*$ *is given by Equation* (B.1) *with* γ *replaced by* $\boldsymbol{T}^*$*.*

Proof. $\mathcal{C}$ in this case is convex compact, as it is the same as $\mathcal{C}$ presented in Proposition 6, up to a translation by a matrix $\mathbf{C}$.

By Proposition 7, solving Problem (3.5) is equivalent to solving

$$\min_{\boldsymbol{T}\in \mathrm{Conv}(\mathcal{P}_t)} \max_{\mathbf{D}\in\mathcal{C}_{\mathbf{C}}} \langle \boldsymbol{T}, \mathbf{D}\rangle$$

However for any matrix $\boldsymbol{T} \in \mathrm{Conv}\,(\mathcal{P}_t)$, and for $r\mathcal{B}^D_{p+} = \{r\mathbf{M}, \mathbf{M} \in \mathcal{B}^D_{p+}\}$ ($\mathcal{B}^D_{p+}$ is defined as in the proof of 6), we have:

$$\begin{aligned}
\max_{\mathbf{D}\in\mathcal{C}_{\mathbf{C}}} \langle \boldsymbol{T}, \mathbf{D}\rangle &= \max_{\mathbf{M}\in r\mathcal{B}^D_{p+}} \sum_{ik} \boldsymbol{T}_{ik}((\boldsymbol{x}_i - \boldsymbol{y}_k)\mathbf{M}(\boldsymbol{x}_i - \boldsymbol{y}_k) + (\mathbf{C})_{ik}) \\
&= \max_{\mathbf{M}\in \mathcal{B}^D_{p+}} r\sum_{ik} \boldsymbol{T}_{ik}((\boldsymbol{x}_i - \boldsymbol{y}_k)\mathbf{M}(\boldsymbol{x}_i - \boldsymbol{y}_k)) + \sum_{ik} \boldsymbol{T}_{ik}(\mathbf{C})_{ik} \\
&= r\|\mathbf{V}_{\boldsymbol{T}}\|_q + \sum_{ik} \boldsymbol{T}_{ik}\mathbf{C}_{ik}
\end{aligned}$$

where in the last line, we used the developments done in Proposition 6, from which we also get the expression of $\mathbf{M}^*$. For the case $p = 2$, we use the result of Corrolary 3, where the PSD constraint is not needed. □

B.4 Experimental evaluations

In this section, we add the details needed to reproduce the experiments from the chapter using the code provided[1]. We also provide more experimental results for the considered evaluation scenarios and full-size figures presented in a reduced size in the main appendix. For all of the experiments, threshold thd2 used for constraint elimination is set to 10^{-12}.

Section 3.4.1: Convergence and execution time Convergence curves for first two plots are obtained for threshold value thd1= 0. The value for the first threshold is to let the algorithm

[1] `https://github.com/sofiendhouib/minimax_OT`.

perform all of the iterations, set to 100. As for the right figure, we set the maximum number of iterations to 1000 and thd1 to 10^{-8}, and we use MOSEK solver to solve the LP formulation, for which we set all tolerance values to 10^{-8}.

Section 3.4.2 Hypercube We set maxIter to 10, $\mathcal{P}_0$ is set to the uniform distribution and thd1$= 10^{-8}$. The experiment is reproduced 100 times.

Section 3.4.3: Stability and noise sensitivity The parameters used in this experiment for all additiional data sets are the same as for the MNIST 0-to-1 dataset and the two Gaussians. The maximum number of iterations of the cutting set method, maxIter is set to 10. $\mathcal{P}_0$ is set to the uniform distribution, thd1$= 10^{-20}$. The Mahalanobis ball has the radius $r = 0.01$. The 50 cost matrices are created with random Mahalanobis projections and different norms taking values in $(2, 3, 4, 5, 10)$. We also add the cost matrix associated with the squared Euclidean distance. Each cost matrix is divided by its Frobenius norm. The noise sensitivity is computed over 200 runs. In all examples of Figure B.1, the sensitivity to noise is correlated to the stability

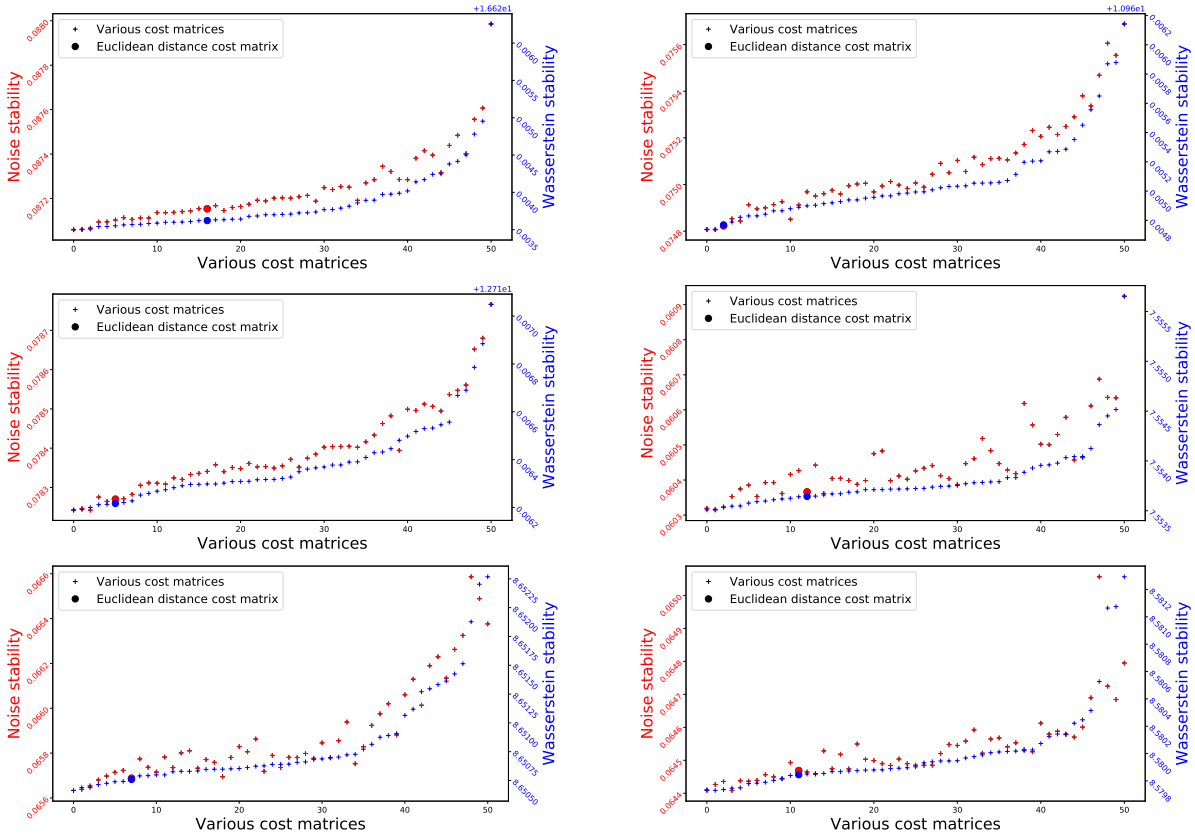


Figure B.1: ***Left to right, top to bottom****: Gaussians, MNIST 0-to-1, MNIST 3-to-0, MNIST 6-to-5, MNIST 7-to-1, MNIST 7-to-4 data sets. Y-axis (left) is the difference between the OT cost with* $\mathbf{C}_i$ *and* $\mathbf{C}_i + \mathbf{E}^{\mathbf{M}}$*. Y-axis (right), the Wasserstein stability defined in Section 3.5. Each column is a different cost matrix, the matrices are ordered by the Wasserstein stability.*

Cited on page [118]

of the cost matrix. The cost matrix associated with the squared Euclidean distance is often stable and robust to noise which is predictable as it is the most used distance in OT. However,

it is never the best cost matrix in terms of our notion of stability.

Section 3.4.4: Color transfer We use the same setting as above with the following parameters: maxIter is set to 200, thd1= 10^{-8}, $r = 0.001$ and we divide each cost matrix element-wise by its corresponding transport cost. On Figure B.2, we first provide images from Chapter 3 in a bigger size in order to see more fine-grained details.

Figure B.2: ***Top row****: Original images of ocean sunset and ocean sky.* ***Middle row****: (****left****) most stable cost matrix, (****right****) squared Euclidean based cost matrix. Bottom row: (****left****) least stable Mahalanobis cost matrix, (****right****) least stable cost matrix. Notice the quality difference between the most stable matrix and the squared Euclidean based one in the area just under the cloud.*

Cited on page [119]

Figure B.3 presents additional visualizations for a new pair of images. The obtained results are in line with experiments shown in the main appendix and exhibit similar behaviour.

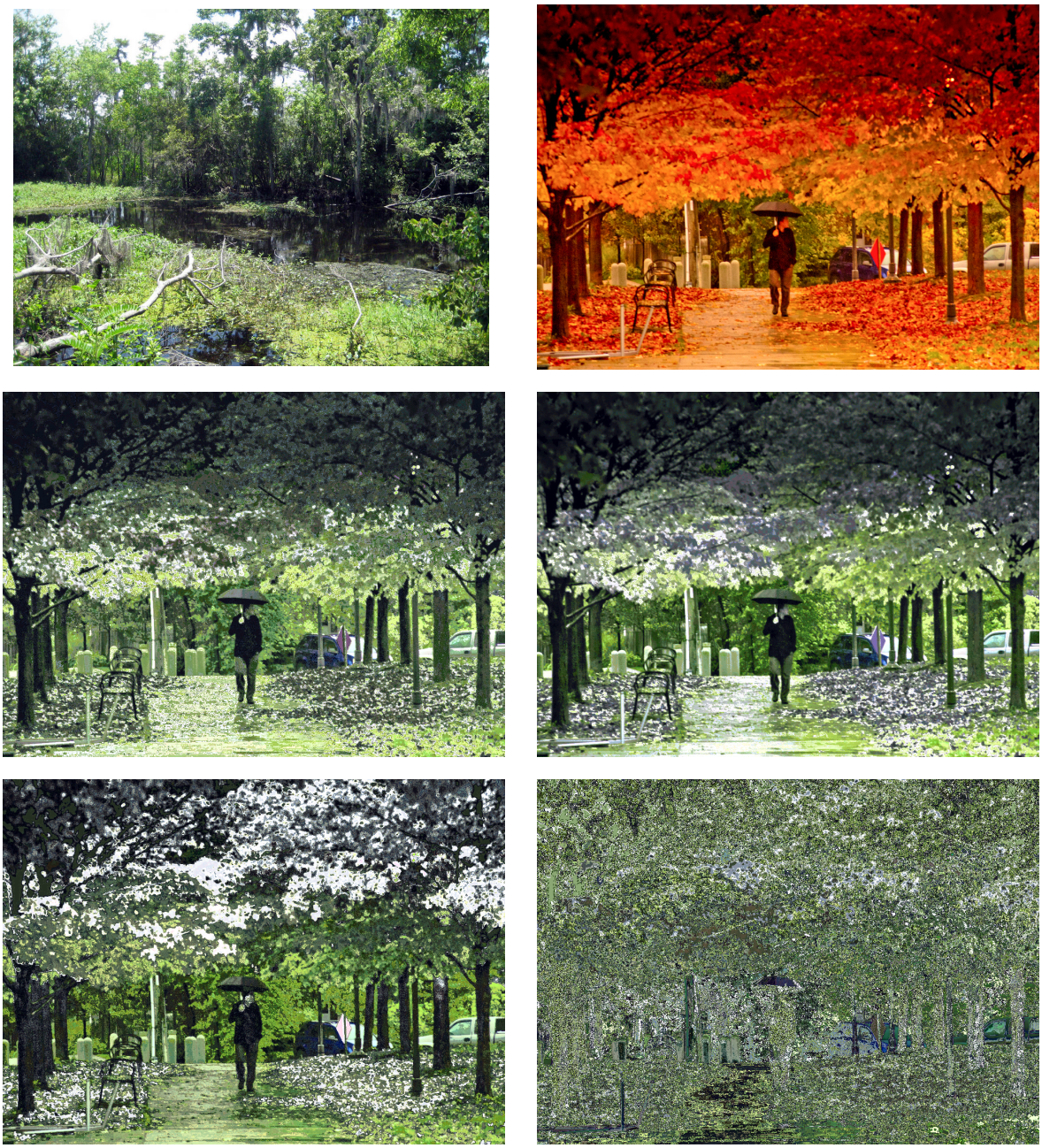


Figure B.3: ***Top row****: Original images of woods and autumn.* ***Middle row****: (****left****) most stable cost matrix, (****right****) Euclidean based cost matrix. Bottom row: (****left****) least stable Mahalanobis cost matrix, (****right****) least stable cost matrix.*

Cited on page [120]

Appendix C

Appendix of Sampled Gromov Wasserstein

C.1 Scalable GW optimization

C.1.1 Detailed derivations for the convergence

This section gives the (very) detailed derivations used to obtain the convergence properties of Section 4.4.2 of Algorithm 5 (from the Chapter 4).

Goal and context

First, let's give a few reminders of the context and the final result. The proposed algorithm runs for S iterations, and averages M sampled cost matrices (obtained by sampling pairs of indices), at each iteration. We provide here a proof of convergence to a stationary point for any arbitrary loss $\mathcal{L}$. Previous algorithms relied on having some particular loss $\mathcal{L}$ to be efficient. When $M = \infty$ and $\alpha = 1$, the proposed algorithm is equivalent to EGW.
We are interested in $G(\boldsymbol{T}) \stackrel{\text{def}}{=} \mathcal{E}(\boldsymbol{T}, \boldsymbol{T}) - \min_{\boldsymbol{T}'} \mathcal{E}(\boldsymbol{T}, \boldsymbol{T}')$. In a non-convex setting, $\boldsymbol{T}$ is a stationary point of $\mathcal{E}(\boldsymbol{T})$ if and only if $G(\boldsymbol{T}) = 0$ (Reddi et al., 2016). We recall the assumptions and notations:

- We suppose $\boldsymbol{C}^{\mathcal{X}}$ and $\boldsymbol{C}^{\mathcal{Y}}$ symmetric. This assumption is notably satisfied if $\boldsymbol{C}^{\mathcal{X}}$ and $\boldsymbol{C}^{\mathcal{Y}}$ are metrics.
- We define $\mathcal{E}(\boldsymbol{A}, \boldsymbol{A}') \stackrel{\text{def}}{=} \mathcal{E}(\boldsymbol{A}', \boldsymbol{A}) = \sum_{i,j=1}^{I,I} \sum_{k,l=1}^{K,K} \boldsymbol{L}_{ijkl} \boldsymbol{A}_{ik} \boldsymbol{A}'_{jl}$
- We overload the notation if the two parameters are the same: $\mathcal{E}(\boldsymbol{A}) \stackrel{\text{def}}{=} \mathcal{E}(\boldsymbol{A}, \boldsymbol{A})$
- We assume that $0 \leq \boldsymbol{L}_{ijkl} \leq B$. This value B can be found in $O\left(N^2\right)$ with any losses $\mathcal{L}$ that increase when $|\boldsymbol{C}^{\mathcal{X}}_{ij}$ - $\boldsymbol{C}^{\mathcal{Y}}_{kl}|$ increases, by looking at the extreme values of the two matrix $\boldsymbol{C}^{\mathcal{Y}}$ and $\boldsymbol{C}^{\mathcal{X}}$.

More precisely, the bound that we will prove here is the following (Theorem 3 in Chapter 4):

$$\mathbb{E}\left(G\left(\overline{\boldsymbol{T}}\right)\right) \leq \sqrt{\frac{2B\left(\mathcal{E}(\boldsymbol{T}_0) - \mathcal{E}(\boldsymbol{T}^*)\right)N}{S}} + B\sqrt{\frac{2N}{M}} + \epsilon\log(N).$$

Where $\boldsymbol{T}^*$ is the optimal (unknown) solution of GW, i.e., $\boldsymbol{T}^* = \underset{\boldsymbol{T}\in\Pi_{\mu\nu}}{\operatorname{argmin}}\,\mathcal{E}(\boldsymbol{T})$ and the expectation is taken on all the sampling done during the algorithm and on $\overline{\boldsymbol{T}}$.

The notation of the Algorithm 5 are slightly different in this appendix, as we make the distinction between $\boldsymbol{T}_s'^{\epsilon}$ and $\boldsymbol{T}_s'$. $\boldsymbol{T}_s'^{\epsilon}$ is the transport plan given by the OT-Sinkhorn solver, while $\boldsymbol{T}_s'$ is the exact minimum transport plan.

Our proof is inspired by the Theorem 2 from (Reddi et al., 2016) but we additionally consider the entropy regularization with notably the lemma 3 which is specific to the OT problem. To give all details while trying to improve readability, we first prove some intermediate results.

Necessary intermediate results

We first prove the following new lemma which quantifies the difference between the Wasserstein distance with and without the entropy regularization, for a generic OT problem with a cost matrix $\boldsymbol{C}$. Note that a related bound was proposed by Genevay et al. (2019) or Blondel et al. (2018) but include the entropy regularization while, here, we are only concerned about the difference between the scalar product.

Lemma 3. *Let $\boldsymbol{T}^{\epsilon}$ (resp. $\boldsymbol{T}^0$) be the optimal solution of a discrete OT problem with (resp. without) entropy regularization. We suppose the simplified case with N points in each empirical distribution and with uniform marginal distributions. We will note $\boldsymbol{C}$ the $N \times N$ cost matrix of this problem.*

$$0 \;\leq\; \langle \boldsymbol{C}, \boldsymbol{T}^{\epsilon}\rangle - \left\langle \boldsymbol{C}, \boldsymbol{T}^0\right\rangle \;\leq\; \epsilon\log(N) \tag{C.1}$$

Proof. The positivity is obtained by definition of $\boldsymbol{T}^0$ (it minimizes $\langle \boldsymbol{C}, \boldsymbol{T}\rangle$). The right-hand side inequality can be derived as follows (where $\mathcal{H}(\boldsymbol{T})$ denotes the entropy of $\boldsymbol{T}$):

$$\begin{aligned}
\langle \boldsymbol{C}, \boldsymbol{T}^{\epsilon}\rangle - \left\langle \boldsymbol{C}, \boldsymbol{T}^0\right\rangle &= \langle \boldsymbol{C}, \boldsymbol{T}^{\epsilon}\rangle - \left\langle \boldsymbol{C}, \boldsymbol{T}^0\right\rangle - \epsilon\mathcal{H}(\boldsymbol{T}^{\epsilon}) + \epsilon\mathcal{H}(\boldsymbol{T}^{\epsilon}) && \text{(C.2)}\\
&\leq \left\langle \boldsymbol{C}, \boldsymbol{T}^0\right\rangle - \left\langle \boldsymbol{C}, \boldsymbol{T}^0\right\rangle - \epsilon\mathcal{H}(\boldsymbol{T}^0) + \epsilon\mathcal{H}(\boldsymbol{T}^{\epsilon}) && \text{(C.3)}\\
&\leq \epsilon\mathcal{H}(\boldsymbol{T}^{\epsilon}) - \epsilon\mathcal{H}(\boldsymbol{T}^0) && \text{(C.4)}\\
&\leq -\,\epsilon\log\left(N^{-2}\right) + \epsilon\log\left(N^{-1}\right) && \text{(C.5)}\\
&= \epsilon\log(N) && \text{(C.6)}
\end{aligned}$$

Line C.3 : by definition, $\boldsymbol{T}^{\epsilon}$ minimizes $\langle \boldsymbol{C}, \boldsymbol{T}\rangle - \epsilon\mathcal{H}(\boldsymbol{T})$. Line C.5 $\boldsymbol{T}^0$ is a permutation and $\boldsymbol{T}^{\epsilon}$ is at worse (in terms of $\mathcal{H}()$) uniform.

□

Interestingly, this bound does not depend directly depend on $\boldsymbol{C}$ (still, $\boldsymbol{C}$ impacts the value of $\boldsymbol{T}^0$, $\boldsymbol{T}^{\epsilon}$). A scale increase of $\boldsymbol{C}$ will virtually reduce ϵ in comparison, thus $\boldsymbol{T}^{\epsilon}$ will be closer to

$\boldsymbol{T}^0$. Note that the bound can be adapted to the general case (arbitrary distributions), then the bound is $\epsilon(\mathcal{H}(\mu) + \mathcal{H}(\nu))$ as we bound $\mathcal{H}(\boldsymbol{T}^0)$ by 0 and $\mathcal{H}(\boldsymbol{T}^\epsilon)$ by $\mathcal{H}(\mu \times \nu)$.

Let $(\boldsymbol{T}, \boldsymbol{T}') \in \Pi^2_{\mu\nu}$. We now derive several intermediate results with these arbitrary transport plans $\boldsymbol{T}$ and $\boldsymbol{T}'$, in a simplified case when $I = K = N$ (same number of points in each empirical distribution).

We start with a bound on the maximal distance between these transport plans (in terms of Frobenius norm):

$$\begin{aligned}
\left\|\boldsymbol{T} - \boldsymbol{T}'\right\|_F &= \sqrt{\|\boldsymbol{T} - \boldsymbol{T}'\|_F^2} && \text{(C.7)}\\
&\leq \sqrt{\|\boldsymbol{T}\|_F^2 + \|\boldsymbol{T}'\|_F^2} && \text{(C.8)}\\
&\leq \sqrt{\sum_{i,k=1}^{I,K} \boldsymbol{T}_{ik}^2 + \sum_{i,k=1}^{I,K} \boldsymbol{T}_{ik}'^2} && \text{(C.9)}\\
&\leq \sqrt{N\left(\frac{1}{N}\right)^2 + N\left(\frac{1}{N}\right)^2} && \text{(C.10)}\\
&= \sqrt{\frac{2}{N}}. && \text{(C.11)}
\end{aligned}$$

Line C.8 : the triangular inequality is used. Line C.10 : for doubly stochastic matrices, the highest Frobenius norm is obtained with a permutation (fewer and thus bigger values give a bigger norm), the permutation has N non-zero values equal to $\frac{1}{N}$.

For completeness, we prove that the gradient of $\mathcal{E}(\boldsymbol{T})$ is expressed in terms of $\boldsymbol{T}$. We prove it with $\boldsymbol{L}$ symmetric, in the sense that $\boldsymbol{L}_{ijkl} = \boldsymbol{L}_{jilk}$, which is implied if the cost matrices are symmetric. For all indices $(a, b) \in (\llbracket 1, I \rrbracket, \llbracket 1, K \rrbracket)$, we have:

$$\begin{aligned}
\frac{\partial \mathcal{E}}{\partial \boldsymbol{T}_{ab}}(\boldsymbol{T}) &= \frac{\partial}{\partial \boldsymbol{T}_{ab}} \sum_{i,j=1}^{I,I} \sum_{k,l=1}^{K,K} \boldsymbol{L}_{ijkl} \boldsymbol{T}_{ik}\, \boldsymbol{T}_{jl} && \text{(C.12)}\\
&= \frac{\partial}{\partial \boldsymbol{T}_{ab}} \left(\boldsymbol{L}_{aabb}\, \boldsymbol{T}_{ab}^2 + \left(2 \sum_{cd \neq ab} \boldsymbol{L}_{acbd} \boldsymbol{T}_{cd} \right) \boldsymbol{T}_{ab} \right) && \text{(C.13)}\\
&= 2\, \boldsymbol{L}_{aabb} \boldsymbol{T}_{ab} + 2 \sum_{cd \neq ab} \boldsymbol{L}_{acbd} \boldsymbol{T}_{cd} && \text{(C.14)}\\
&= 2 \sum_{cd} \boldsymbol{L}_{acbd} \boldsymbol{T}_{cd} && \text{(C.15)}\\
\nabla \mathcal{E}(\boldsymbol{T}) &= 2 \sum_{jl} \boldsymbol{L}_{.j.l} \boldsymbol{T}_{jl} && \text{(C.16)}\\
&= \sum_{jl} (\boldsymbol{L}_{.j.l} + \boldsymbol{L}_{j.l.}) \boldsymbol{T}_{jl} \text{ in the case where } \boldsymbol{L} \text{ is not symmetric.} && \text{(C.17)}
\end{aligned}$$

We can also prove that $\mathcal{E}$ is $2BN^2$-smooth, as follows:

$$\begin{aligned}
\left\|\nabla\mathcal{E}(\boldsymbol{T}) - \nabla\mathcal{E}(\boldsymbol{T}')\right\|_F &= \left\|2\sum_{jl}\boldsymbol{L}_{.j.l}\boldsymbol{T}_{jl} - 2\sum_{jl}\boldsymbol{L}_{.j.l}\boldsymbol{T}'_{jl}\right\|_F && \text{(C.18)}\\
&= \left\|2\sum_{jl}\boldsymbol{L}_{.j.l}\left(\boldsymbol{T}_{jl} - \boldsymbol{T}'_{jl}\right)\right\|_F && \text{(C.19)}\\
&= \sqrt{\sum_{ik}\left(2\sum_{jl}\boldsymbol{L}_{ijkl}\left(\boldsymbol{T}_{jl} - \boldsymbol{T}'_{jl}\right)\right)^2} && \text{(C.20)}\\
&= \sqrt{\sum_{ik}\left(2\left\langle\boldsymbol{L}_{i.k.}, \boldsymbol{T} - \boldsymbol{T}'\right\rangle\right)^2} && \text{(C.21)}\\
&\leq \sqrt{\sum_{ik}\left(2\left\|\boldsymbol{L}_{i.k.}\right\|_F\left\|\boldsymbol{T} - \boldsymbol{T}'\right\|_F\right)^2} && \text{(C.22)}\\
&\leq \sqrt{4\sum_{jl}B^2N^2\left\|\boldsymbol{T} - \boldsymbol{T}'\right\|_F^2} && \text{(C.23)}\\
&\leq 2B\sqrt{N^4\left\|\boldsymbol{T} - \boldsymbol{T}'\right\|_F^2} && \text{(C.24)}\\
&\leq 2BN^2\left\|\boldsymbol{T} - \boldsymbol{T}'\right\|_F. && \text{(C.25)}
\end{aligned}$$

Line C.22 uses the Cauchy–Schwarz inequality. Line C.23 uses $0 \leq \boldsymbol{L}_{ijkl} \leq B$.

The following Lemma 4 is the same as the one provided in Reddi et al. (2016) and will allow to start the proof.

Lemma 4. *If $f : \mathbb{R}^D \longrightarrow \mathbb{R}$ is L-smooth, then for all $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^D$.*

$$f(\boldsymbol{x}) \leq f(\boldsymbol{y}) + \langle\nabla f(\boldsymbol{y}), \boldsymbol{x} - \boldsymbol{y}\rangle + \frac{L}{2}\|\boldsymbol{x} - \boldsymbol{y}\|^2$$

Proof of the theorem

Theorem 7 (Based on (Reddi et al., 2016))**.** *For any $\boldsymbol{L}_{ijkl} \in [0, 1]$, for any distributions μ and ν with uniform weights $\boldsymbol{a}$ and $\boldsymbol{b}$ respectively, for any optimal solution $\boldsymbol{T}^*$ of Problem (4.2), on average for the transport plan $\overline{\boldsymbol{T}}$ uniformly sampled from $(\boldsymbol{T}_s)_{s\in[\![0,S-1]\!]}$, on average over all the samplings, the following bound holds:*

$$\mathbb{E}\left(G\left(\overline{\boldsymbol{T}}\right)\right) \leq \sqrt{\frac{2B\left(\mathcal{E}(\boldsymbol{T}_0) - \mathcal{E}(\boldsymbol{T}^*)\right)N}{S}} + B\sqrt{\frac{2N}{M}} + \epsilon\log(N).$$

Proof. $\boldsymbol{T}_s$ and $\boldsymbol{T}'^{\epsilon}_s$ are the transport plan obtain in the Algorithm 5. $\boldsymbol{T}'_s = \boldsymbol{T}'^{0}_s$ is the solution without entropy regularization.

Let $\hat{\boldsymbol{T}}'_s = \underset{\boldsymbol{T}'_s\in\mathcal{U}_{\mu\nu}}{\operatorname{argmin}}\langle\boldsymbol{T}'_s, \nabla\mathcal{E}(\boldsymbol{T}_s)\rangle = \underset{\boldsymbol{T}'_s\in\mathcal{U}_{\mu\nu}}{\operatorname{argmax}}\langle\boldsymbol{T}'_s, -\nabla\mathcal{E}(\boldsymbol{T}_s)\rangle$ and $\widehat{\boldsymbol{\Lambda}}_s$ the sum of matrices sampled M times at iteration s.

$$\mathcal{E}(\boldsymbol{T}_{s+1}) \leq \mathcal{E}(\boldsymbol{T}_s) + \langle \nabla \mathcal{E}(\boldsymbol{T}_s), \boldsymbol{T}_{s+1} - \boldsymbol{T}_s \rangle + \frac{2BN^2}{2} \|\boldsymbol{T}_{s+1} - \boldsymbol{T}_s\|^2 \tag{C.26}$$

$$\leq \mathcal{E}(\boldsymbol{T}_s) + \left\langle \nabla \mathcal{E}(\boldsymbol{T}_s), \alpha(\boldsymbol{T}_s'^{\epsilon} - \boldsymbol{T}_s) \right\rangle + BN^2 \|\alpha(\boldsymbol{T}_s'^{\epsilon} - \boldsymbol{T}_s)\|^2 \tag{C.27}$$

$$\leq \mathcal{E}(\boldsymbol{T}_s) + \left\langle \nabla \mathcal{E}(\boldsymbol{T}_s), \alpha(\boldsymbol{T}_s'^{\epsilon} - \boldsymbol{T}_s) \right\rangle + BN^2 \alpha^2 \sqrt{\frac{2}{N}}^2 \tag{C.28}$$

$$= \mathcal{E}(\boldsymbol{T}_s) + \alpha \left\langle 2\widehat{\boldsymbol{\Lambda}}_s, \boldsymbol{T}_s'^{\epsilon} - \boldsymbol{T}_s \right\rangle + \alpha \left\langle \nabla \mathcal{E}(\boldsymbol{T}_s) - 2\widehat{\boldsymbol{\Lambda}}_s, \boldsymbol{T}_s'^{\epsilon} - \boldsymbol{T}_s \right\rangle + BN^2 \alpha^2 \sqrt{\frac{2}{N}}^2 \tag{C.29}$$

$$= \mathcal{E}(\boldsymbol{T}_s) + \alpha \left\langle 2\widehat{\boldsymbol{\Lambda}}_s, \boldsymbol{T}_s'^{\epsilon} - \boldsymbol{T}_s + \boldsymbol{T}_s' - \boldsymbol{T}_s' \right\rangle + \alpha \left\langle \nabla \mathcal{E}(\boldsymbol{T}_s) - 2\widehat{\boldsymbol{\Lambda}}_s, \boldsymbol{T}_s'^{\epsilon} - \boldsymbol{T}_s \right\rangle$$
$$+ BN^2 \alpha^2 \sqrt{\frac{2}{N}}^2 \tag{C.30}$$

$$= \mathcal{E}(\boldsymbol{T}_s) + \alpha \left\langle 2\widehat{\boldsymbol{\Lambda}}_s, \boldsymbol{T}_s' - \boldsymbol{T}_s \right\rangle + \alpha 2 \left\langle \widehat{\boldsymbol{\Lambda}}_s, \boldsymbol{T}_s'^{\epsilon} - \boldsymbol{T}_s' \right\rangle + \alpha \left\langle \nabla \mathcal{E}(\boldsymbol{T}_s) - 2\widehat{\boldsymbol{\Lambda}}_s, \boldsymbol{T}_s'^{\epsilon} - \boldsymbol{T}_s \right\rangle$$
$$+ BN^2 \alpha^2 \sqrt{\frac{2}{N}}^2 \tag{C.31}$$

$$\leq \mathcal{E}(\boldsymbol{T}_s) + \alpha \left\langle 2\widehat{\boldsymbol{\Lambda}}_s, \hat{\boldsymbol{T}}_s' - \boldsymbol{T}_s \right\rangle + \alpha 2\epsilon \log(N) + \alpha \left\langle \nabla \mathcal{E}(\boldsymbol{T}_s) - 2\widehat{\boldsymbol{\Lambda}}_s, \boldsymbol{T}_s'^{\epsilon} - \boldsymbol{T}_s \right\rangle$$
$$+ BN^2 \alpha^2 \sqrt{\frac{2}{N}}^2 \tag{C.32}$$

$$= \mathcal{E}(\boldsymbol{T}_s) + \alpha \left\langle \nabla \mathcal{E}(\boldsymbol{T}_s), \hat{\boldsymbol{T}}_s' - \boldsymbol{T}_s \right\rangle + \alpha \left\langle \nabla \mathcal{E}(\boldsymbol{T}_s) - 2\widehat{\boldsymbol{\Lambda}}_s, \boldsymbol{T}_s'^{\epsilon} - \hat{\boldsymbol{T}}_s' \right\rangle + \alpha 2\epsilon \log(N)$$
$$+ BN^2 \alpha^2 \sqrt{\frac{2}{N}}^2 \tag{C.33}$$

$$= \mathcal{E}(\boldsymbol{T}_s) - 2\alpha G(\boldsymbol{T}_s) + \alpha \left\langle \nabla \mathcal{E}(\boldsymbol{T}_s) - 2\widehat{\boldsymbol{\Lambda}}_s, \boldsymbol{T}_s'^{\epsilon} - \hat{\boldsymbol{T}}_s' \right\rangle + \alpha 2\epsilon \log(N) + BN^2 \alpha^2 \sqrt{\frac{2}{N}}^2 \tag{C.34}$$

$$\leq \mathcal{E}(\boldsymbol{T}_s) - 2\alpha G(\boldsymbol{T}_s) + \sqrt{\frac{2}{N}} \alpha \|\nabla \mathcal{E}(\boldsymbol{T}_s) - 2\widehat{\boldsymbol{\Lambda}}_s\| + \alpha 2\epsilon \log(N) + BN^2 \alpha^2 \sqrt{\frac{2}{N}}^2 \tag{C.35}$$

The line C.26 uses the smoothness of $\mathcal{E}$. The line C.27 uses the definition of the update. The line C.28 uses the bound between transports plans. The line C.29 adds artificially the $2\widehat{\boldsymbol{\Lambda}}_s$ term. The line C.30 adds artificially the $\boldsymbol{T}_s'$ term. The line C.31 separate two terms. The line C.32 uses the Lemma 3 with $\widehat{\boldsymbol{\Lambda}}_s$ as cost matrix and use the definition of $\boldsymbol{T}_s'$. The line C.33 uses the following equalities,

$$\left\langle 2\widehat{\boldsymbol{\Lambda}}_s, \hat{\boldsymbol{T}}_s' - \boldsymbol{T}_s \right\rangle + \left\langle \nabla \mathcal{E}(\boldsymbol{T}_s) - 2\widehat{\boldsymbol{\Lambda}}_s, \boldsymbol{T}_s'^{\epsilon} - \boldsymbol{T}_s \right\rangle \tag{C.36}$$

$$= \left\langle 2\widehat{\boldsymbol{\Lambda}}_s, \hat{\boldsymbol{T}}_s' - \boldsymbol{T}_s - \boldsymbol{T}_s'^{\epsilon} + \boldsymbol{T}_s \right\rangle + \left\langle \nabla \mathcal{E}(\boldsymbol{T}_s), \boldsymbol{T}_s'^{\epsilon} - \boldsymbol{T}_s + \hat{\boldsymbol{T}}_s' - \hat{\boldsymbol{T}}_s' \right\rangle \tag{C.37}$$

$$= \left\langle 2\widehat{\boldsymbol{\Lambda}}_s, \hat{\boldsymbol{T}}_s' - \boldsymbol{T}_s'^{\epsilon} \right\rangle + \left\langle \nabla \mathcal{E}(\boldsymbol{T}_s), \boldsymbol{T}_s'^{\epsilon} - \hat{\boldsymbol{T}}_s' \right\rangle + \left\langle \nabla \mathcal{E}(\boldsymbol{T}_s), \hat{\boldsymbol{T}}_s' - \boldsymbol{T}_s \right\rangle \tag{C.38}$$

$$= \left\langle \nabla \mathcal{E}(\boldsymbol{T}_s) - 2\widehat{\boldsymbol{\Lambda}}_s, \boldsymbol{T}_s'^{\epsilon} - \hat{\boldsymbol{T}}_s' \right\rangle + \left\langle \nabla \mathcal{E}(\boldsymbol{T}_s), \hat{\boldsymbol{T}}_s' - \boldsymbol{T}_s \right\rangle. \tag{C.39}$$

The line C.34 uses the definition of $G(\boldsymbol{T}_s)$. The line C.35 applies Cauchy Schwartz inequality and bound the difference between OT plan.

To bound the difference between the real expectation $\nabla \mathcal{E}(\boldsymbol{T}_s)$ and the sampling $2\widehat{\boldsymbol{\Lambda}}_s$, the following result is needed. Let define M random variable, $\boldsymbol{z}_m = \boldsymbol{L}_{.j_m.l_m} - \sum_{jl} \boldsymbol{L}_{.j.l} \boldsymbol{T}_{jl}$. They have

0 mean and each $\boldsymbol{z}_m$ are independent from each other. Moreover, $\|\boldsymbol{z}_m\|=\|\boldsymbol{L}_{.j_m.l_m}-\sum_{jl}\boldsymbol{L}_{.j.l}\boldsymbol{T}_{jl}\|\leq\sqrt{\sum_{ik}B^2}=BN$.

$$\mathbb{E}(\|\nabla\mathcal{E}(\boldsymbol{T}_s)-2\widehat{\boldsymbol{\Lambda}}_s\|)=\mathbb{E}(\|2\sum_{jl}\boldsymbol{L}_{.j.l}\boldsymbol{T}_{jl}-\frac{2}{M}\sum_{m=1}^{M}\boldsymbol{L}_{.j_m.l_m}\|) \tag{C.40}$$

$$=\frac{1}{M}2\mathbb{E}(\|\sum_{m=1}^{M}\boldsymbol{z}_m\|) \tag{C.41}$$

$$=\frac{2}{M}\sqrt{(\mathbb{E}(\|\sum_{m=1}^{M}\boldsymbol{z}_m\|))^2} \tag{C.42}$$

$$\leq\frac{2}{M}\sqrt{\mathbb{E}(\|\sum_{m=1}^{M}\boldsymbol{z}_m\|^2)}\qquad\text{Jensen Inequality} \tag{C.43}$$

$$=\frac{2}{M}\sqrt{\sum_{m=1}^{M}\sum_{r=1}^{M}\mathbb{E}(\langle\boldsymbol{z}_m,z_r\rangle)} \tag{C.44}$$

$$=\frac{2}{M}\sqrt{\sum_{m=1}^{M}\mathbb{E}(\|\boldsymbol{z}_m\|^2)} \tag{C.45}$$

$$\leq\frac{2}{M}\sqrt{\sum_{m=1}^{M}\mathbb{E}(B^2N^2)} \tag{C.46}$$

$$=\frac{2}{M}\sqrt{\sum_{m=1}^{M}B^2N^2} \tag{C.47}$$

$$=\frac{2}{M}\sqrt{MB^2N^2} \tag{C.48}$$

$$=\frac{2BN}{\sqrt{M}} \tag{C.49}$$

This result can be used directly on the bound, after averaging over all the sampling.

$$\mathbb{E}(\mathcal{E}(\boldsymbol{T}_{s+1}))\leq\mathbb{E}(\mathcal{E}(\boldsymbol{T}_s))-2\alpha\mathbb{E}(G(\boldsymbol{T}_s))+\sqrt{\frac{2}{N}}\alpha\mathbb{E}(\|\nabla\mathcal{E}(\boldsymbol{T}_s)-2\widehat{\boldsymbol{\Lambda}}_s\|)+\alpha2\epsilon\log(N)$$
$$+BN^2\alpha^2\sqrt{\frac{2}{N}}^2 \tag{C.50}$$

$$\leq\mathbb{E}(\mathcal{E}(\boldsymbol{T}_s))-2\alpha\mathbb{E}(G(\boldsymbol{T}_s))+\sqrt{\frac{2}{N}}\alpha\frac{2BN}{\sqrt{M}}+\alpha2\epsilon\log(N)+BN^2\alpha^2\sqrt{\frac{2}{N}}^2 \tag{C.51}$$

Thus,

$$2\alpha\mathbb{E}(G(\boldsymbol{T}_s))\leq\mathbb{E}(\mathcal{E}(\boldsymbol{T}_s))-\mathbb{E}(\mathcal{E}(\boldsymbol{T}_{s+1}))+2\sqrt{\frac{2}{N}}\alpha\frac{BN}{\sqrt{M}}+\alpha2\epsilon\log(N)+BN^2\alpha^2\sqrt{\frac{2}{N}}^2. \tag{C.52}$$

We set sum over all s on both side.

$$2\alpha\sum_{s=0}^{S-1}\mathbb{E}(G(\boldsymbol{T}_s))\leq\mathcal{E}(\boldsymbol{T}_0)-\mathbb{E}(\mathcal{E}(\boldsymbol{T}_{S-1}))+S2\sqrt{\frac{2}{N}}\alpha\frac{BN}{\sqrt{M}}+S\alpha2\epsilon\log(N)+SBN^2\alpha^2\sqrt{\frac{2}{N}}^2 \tag{C.53}$$

$$\leq \mathcal{E}(\boldsymbol{T}_0) - \mathbb{E}(\mathcal{E}(\boldsymbol{T}_{S-1})) + S2\sqrt{\frac{2}{N}}\alpha\frac{BN}{\sqrt{M}} + S\alpha 2\epsilon\log(N) + SBN^2\alpha^2\sqrt{\frac{2}{N}}^2 \tag{C.54}$$

We use the definition of $\overline{\boldsymbol{T}}$ for $G(\overline{\boldsymbol{T}})$. Notice that the following line is correct only on average for the random variable $\overline{\boldsymbol{T}}$. This part is not clearly specified in the original proof of Reddi et al. (2016). We use also the definition of $\boldsymbol{T}^*$ for the second inequality.

$$\mathbb{E}(G(\overline{\boldsymbol{T}})) \leq \frac{\mathcal{E}(\boldsymbol{T}_0) - \mathbb{E}(\mathcal{E}(\boldsymbol{T}_{S-1}))}{2S\alpha} + \sqrt{\frac{2}{N}}\frac{BN}{\sqrt{M}} + \epsilon\log(N) + BN^2\alpha\sqrt{\frac{1}{N}}^2 \tag{C.55}$$

$$\leq \frac{\mathcal{E}(\boldsymbol{T}_0) - \mathcal{E}(\boldsymbol{T}^*)}{2S\alpha} + B\sqrt{\frac{2N}{M}} + \epsilon\log(N) + BN\alpha \tag{C.56}$$

We derive the function $f(\alpha) = \frac{\mathcal{E}(\boldsymbol{T}_0)-\mathcal{E}(\boldsymbol{T}^*)}{2S\alpha} + B\sqrt{\frac{2N}{M}} + \epsilon\log(N) + BN\alpha$ with respect to α.

$$\frac{df}{d\alpha}(\alpha) = 0 \iff -\frac{\mathcal{E}(\boldsymbol{T}_0) - \mathcal{E}(\boldsymbol{T}^*)}{2S\alpha^2} + BN = 0 \tag{C.57}$$

$$\iff \frac{1}{\alpha^2} = \frac{2S}{\mathcal{E}(\boldsymbol{T}_0) - \mathcal{E}(\boldsymbol{T}^*)}BN \tag{C.58}$$

$$\iff \alpha = \sqrt{\frac{\mathcal{E}(\boldsymbol{T}_0) - \mathcal{E}(\boldsymbol{T}^*)}{2SBN}} \tag{C.59}$$

As $\mathcal{E}(\boldsymbol{T}_0) - \mathcal{E}(\boldsymbol{T}^*) \geq 0$, the second derivative is positive, thus f is convex, therefore we have the minimum. We can replace α and find the final bound,

$$\mathbb{E}(G(\overline{\boldsymbol{T}})) \leq \frac{\mathcal{E}(\boldsymbol{T}_0) - \mathcal{E}(\boldsymbol{T}^*)}{2S\sqrt{\frac{\mathcal{E}(\boldsymbol{T}_0)-\mathcal{E}(\boldsymbol{T}^*)}{2BSN}}} + B\sqrt{\frac{2N}{M}} + \epsilon\log(N) + BN\sqrt{\frac{\mathcal{E}(\boldsymbol{T}_0) - \mathcal{E}(\boldsymbol{T}^*)}{2BSN}} \tag{C.60}$$

$$\leq \frac{\mathcal{E}(\boldsymbol{T}_0) - \mathcal{E}(\boldsymbol{T}^*)}{2S}\sqrt{\frac{2BSN}{\mathcal{E}(\boldsymbol{T}_0) - \mathcal{E}(\boldsymbol{T}^*)}} + B\sqrt{\frac{2N}{M}} + \epsilon\log(N) + \sqrt{\frac{(\mathcal{E}(\boldsymbol{T}_0) - \mathcal{E}(\boldsymbol{T}^*))BN}{2S}} \tag{C.61}$$

$$\leq \sqrt{\frac{(\mathcal{E}(\boldsymbol{T}_0) - \mathcal{E}(\boldsymbol{T}^*))BN}{2S}} + B\sqrt{\frac{2N}{M}} + \epsilon\log(N) + \sqrt{\frac{(\mathcal{E}(\boldsymbol{T}_0) - \mathcal{E}(\boldsymbol{T}^*))BN}{2S}} \tag{C.62}$$

$$\leq \sqrt{\frac{2(\mathcal{E}(\boldsymbol{T}_0) - \mathcal{E}(\boldsymbol{T}^*))BN}{S}} + B\sqrt{\frac{2N}{M}} + \epsilon\log(N). \tag{C.63}$$

□

We will now prove the second theorem using the same proof.

Theorem 8. *With the same notations as in Theorem 7 with the entropy ϵ_s that may now change along the iterations, when L yields a concave GW problem the following bound holds:*

$$\mathbb{E}\left(G\left(\overline{\boldsymbol{T}}\right)\right) \leq \frac{\mathcal{E}(\boldsymbol{T}_0) - \mathcal{E}(\boldsymbol{T}^*)}{2S} + B\sqrt{\frac{2N}{M}} + \frac{1}{S}\sum_{s=0}^{S-1}\epsilon_s\log(N)$$

Proof. The first difference is line C.53, the sum from 0 to $S-1$ cannot be changed to S as ϵ_s may change along the iterations and is now a sum over $\sum_{s=0}^{S-1}$. The second difference is in Lemma 4, were the last term disappear as GW is concave. Thus, in line C.56, as the last term is not present the optimal value of α is 1, which gives the proposed bound. □

C.1.2 A KL regularization-based variant

In this section, we will discuss the convergence of the KL variant. A related convergence proof is proposed in (Xu et al., 2019b) were the authors aim at solving GW using a proximal point method,

$$\min_{\boldsymbol{T}\in\Pi_{\mu\nu}} \mathcal{E}(\boldsymbol{T},\boldsymbol{T}) + \epsilon KL(\boldsymbol{T}||\boldsymbol{T}^n).$$

However, it does not cover our case where $\min_{\boldsymbol{T}\in\Pi_{\mu\nu}} \mathcal{E}(\boldsymbol{T},\boldsymbol{T}^n) + \epsilon KL(\boldsymbol{T}||\boldsymbol{T}^n)$ is minimized at each iteration with the expectation approximated by a sampling. Without sampling, this optimization can be seen as a Majorization-Minization method (Sun et al., 2016b),

$$\begin{aligned}
\mathcal{E}(\boldsymbol{T}) &\leq \mathcal{E}(\boldsymbol{T}^n) + \langle \nabla\mathcal{E}(\boldsymbol{T}^n), \boldsymbol{T}-\boldsymbol{T}^n\rangle + \frac{2BN^2}{2}\|\boldsymbol{T}-\boldsymbol{T}^n\|^2 && \text{(C.64)}\\
&\leq \mathcal{E}(\boldsymbol{T}^n) + \langle \nabla\mathcal{E}(\boldsymbol{T}^n), \boldsymbol{T}-\boldsymbol{T}^n\rangle + BN^2|\boldsymbol{T}-\boldsymbol{T}^n|_1^2 && \text{(C.65)}\\
&\leq \mathcal{E}(\boldsymbol{T}^n) + \langle \nabla\mathcal{E}(\boldsymbol{T}^n), \boldsymbol{T}-\boldsymbol{T}^n\rangle + BN^2 KL(\boldsymbol{T}||\boldsymbol{T}^n). && \text{(C.66)}
\end{aligned}$$

Were the first line is the line C.26 in the proof of Theorem 7. The second line use the fact that the L2 norm is bigger than the L1 norm. The last line uses the Pinsker's inequality.
While the last inequality seems to be a good starting point, we could not directly derive (or find in the literature) a bound that applies with sampling and the KL term (that makes the use of Sinkhorn-Knopps possible). Thus, while this interpretation seems interesting, the question of the convergence is left open and would need to be studied in a future work.

C.1.3 Approximating the Gromov Wasserstein distance

This section gives mathematical details for the estimation of the Gromov Wasserstein distance from a given transport plan. Our approach to compute the GW distance will take inspiration from the idea of sampling $\boldsymbol{T} \in \Pi_{\mu\nu}$ (i.e., with marginals $\boldsymbol{a}$ and $\boldsymbol{b}$).

Let define a new random variable $P(R = \boldsymbol{L}_{ijkl}) = \boldsymbol{T}_{ik}\boldsymbol{T}_{jl}$. This definition is not totally rigorous: two values $\boldsymbol{L}_{ijkl}$ and $\boldsymbol{L}_{i'j'k'l'}$ may be equal, the actual probability is then the sum of the probabilities. The GW distance can now be seen as an expectation,

$$\sum_{ijkl} \boldsymbol{L}_{ijkl}\boldsymbol{T}_{ik}\boldsymbol{T}_{jl} = \mathbb{E}(R). \tag{C.67}$$

Instead of simply sampling this expectation, we propose to stratify by each index i, j to improve the quality of the estimator. Let U_i be the event "*i is chosen for the first dimension of* $\boldsymbol{L}$" and U'_j be the event "*j is chosen for the second dimension of* $\boldsymbol{L}$". Using the rule of total expectation,

the expectation can be transformed to,

$$\mathbb{E}(R) = \sum_{ij} \mathbb{P}(U_i \cap U'_j)\mathbb{E}(R|U_i \cap U'_j) \tag{C.68}$$

$$= \sum_{ij} \boldsymbol{a}_i \boldsymbol{a}_j \mathbb{E}(R|U_i \cap U'_j). \tag{C.69}$$

For any $(i,j) \in [\![1,N]\!]^2$, we denote as X_{ij} the random variable defined by: $\mathbb{P}(X_{ij} = \boldsymbol{L}_{ijkl}) = \mathbb{P}(R = \boldsymbol{L}_{ijkl}|U_i \cap U'_j)$. Thus, we use $\hat{R} = \sum_{ij} \boldsymbol{a}_i \boldsymbol{a}_j \frac{1}{M} \sum_{m=1}^{M} X_{ij}^m$ to estimate the Gromov Wasserstein distance. This estimator is unbiased and comes with a tight estimator of the standard deviation as shown on the Figure 4.4 of Chapter 4,

$$\hat{\sigma}_{\tilde{R}} = \sqrt{\sum_{ij} \boldsymbol{a}_i^2 \boldsymbol{a}_j^2 \frac{1}{M-1} \sum_{m=1}^{M} \left(X_{ij}^m - \left(\frac{1}{M} \sum_{m'=1}^{M} X_{ij}^{m'} \right) \right)^2}. \tag{C.70}$$

We recommend taking at least $M = 2$, to have access to the standard deviation. Note that we can look only at a sub-sample of the indices i, k ($\sqrt{N \log(N)}$ instead of all the N points), to have an approximation of the distance in $N \log(N)$. This is useful when coupled with Pointwise Gromov Wasserstein, however, the predicted distance might be far from the real one without any standard deviation to quantify the error.

C.2 Experiments

C.2.1 General setup and methods

We remind that the code to reproduce all the experiments, figures and tables is available in the GitHub repository: `https://github.com/Hv0nnus/Sampled-Gromov-Wasserstein`.

Gaussians mixtures

This section explains how the Gaussians mixtures are created with a Gaussian Random Partition Graph (Brandes et al., 2003) based on Stochastic Block Model (Holland et al., 1983). The Algorithm 7 describe how to sample N points. This algorithm will create some Gaussians separated from each other and some values will be sample from those Gaussians.

For the experiment, the dimension space D is set to 10 and 20 for the distributions μ and ν. The Euclidean distance is used on both spaces to compute $\boldsymbol{C}^{\mathcal{X}}$ and $\boldsymbol{C}^{\mathcal{Y}}$.

Gaussian Random Partition Graph

For the second experiment, we generate graphs using a Gaussian Random Partition Graph (Brandes et al., 2003) with intra-cluster probability of 0.5, extra-cluster probability of 0.1, the number of nodes in each cluster is sampled from a Gaussian with mean $min(\frac{N}{2}, 200)$ and a variance of 5. The adjacency matrix of each graph is used for $\boldsymbol{C}^{\mathcal{X}}$ and $\boldsymbol{C}^{\mathcal{Y}}$. We set $\boldsymbol{a}$ and $\boldsymbol{b}$ to the uniform distribution.

Algorithm 7 Gaussians mixtures dataset. Cited on page [129]

Require: N, D, $v_{in} = 2$, $v_{out} = 10$, $v_{points} = 5$, $c = \min(200, \frac{N}{2})$

$n = 0$
$Positions = [\,]$
while $n < N$ **do**
  $mean = \mathcal{N}(\mathbf{0}, v_{out} \times \mathbb{I}_D)$
  $n' = \mathcal{N}(c, v_{points})$
  Add n' sample of $\mathcal{N}(mean, v_{in} \times \mathbb{I}_D)$ to $Positions$
  $n = n + n'$
end while
return N first points of $Positions$

C.2.2 Speed and accuracy of the GW estimate

We reproduce the figures available in Chapter 4 in Figure C.1 and Figure C.2. The left part of Figure C.2 is omitted from the paper: it shows similar time complexity compared to the left part of Figure C.1.

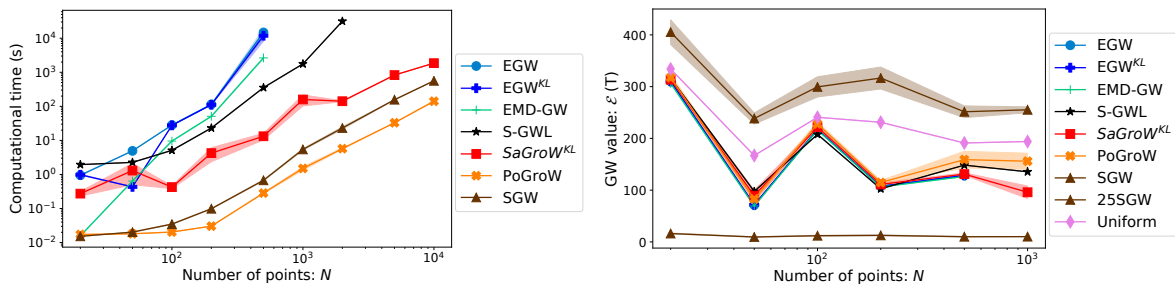


Figure C.1: *Computational time (left) and GW distance estimation (right) between points sampled from mixtures of Gaussians.*

Cited on pages [129,129]

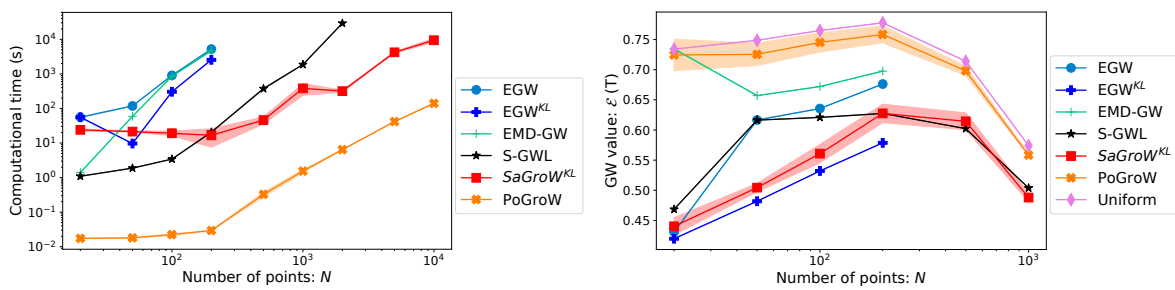


Figure C.2: *Computational time (left) and GW distance estimation (right) on synthetic graphs (Brandes et al., 2003).*

Cited on pages [129,129]

Figure C.3 shows that the computation time is clearly different with the square loss. To

facilitate the comparison, we keep, for every method, the same hyperparameter used for the absolute loss. Those parameters may not be optimal, especially S-GWL which seems to perform poorly. On the left Figure C.3, SGW is faster that PoGroW because the distance can be computed in $O(N\log(N))$, and the entire algorithm is efficiently parallelized. Because of its $O(N^2)$ complexity, SaGroW is still faster than S-GWL and EGW for a high number of points.

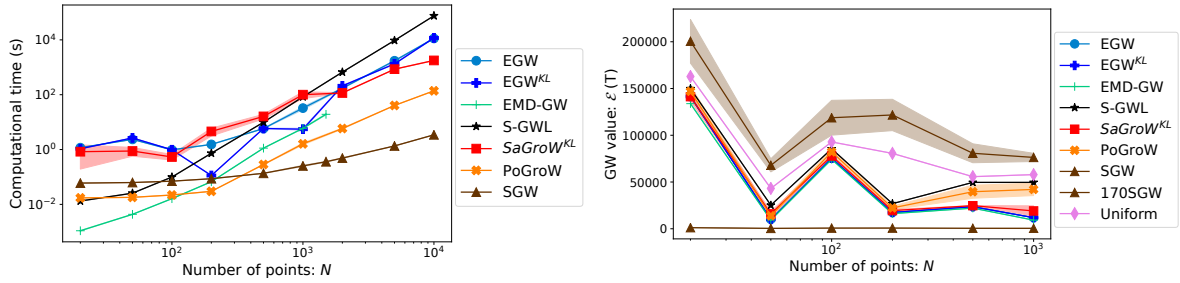


Figure C.3: *Computational time (left) and GW distance estimation (right) between sampled points from mixtures of Gaussians with the square loss.*

Cited on pages [130,130]

C.2.3 Hyperparameter analysis

In this section, we plot figures similar to Figures 4.8, 4.9 and 4.10 from Chapter 4.

Figure C.4 shows the difference between the square loss and the absolute one, to compare computational times. While our method remains the same, the other method improve their computational time.

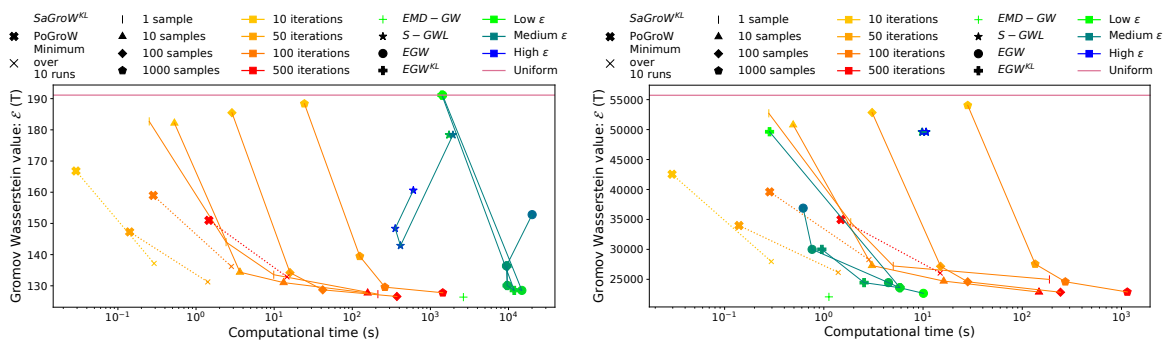


Figure C.4: *Similar to the Figures 4.8, 4.9 and 4.10 in Chapter 4.* ***(Left)*** *Absolute loss.* ***(Right)*** *Square loss.*

Cited on page [131]

Figure C.5 shows similar expected behavior for a graphs dataset.

Figure C.6 shows a very easy situations, where every method probably finds the right GW distance. In this case, PoGroW is very competitive even for the square loss.

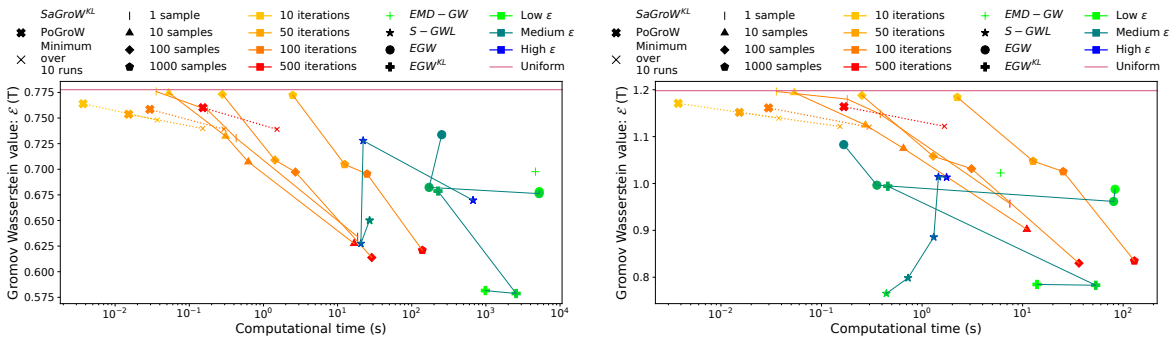


Figure C.5: *Hyperparameters analysis on a Stochastic Block Model dataset with 200 nodes for each graphs. (**Left**) Absolute loss. (**Right**) Square loss.*

Cited on page [131]

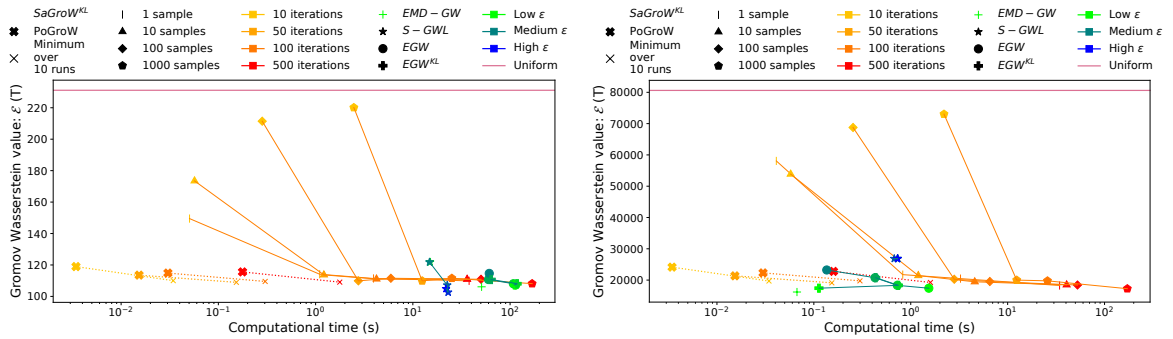


Figure C.6: *Hyperparameters analysis on a mixture of Gaussians with 200 points sampled for each distributions. (**Left**) Absolute loss. (**Right**) Square loss.*

Cited on page [131]

Figure C.7 highlights the interest of our method even for a very small N (20 nodes in each graph). In this case, SaGroW obtains the best transport plan for the square loss.

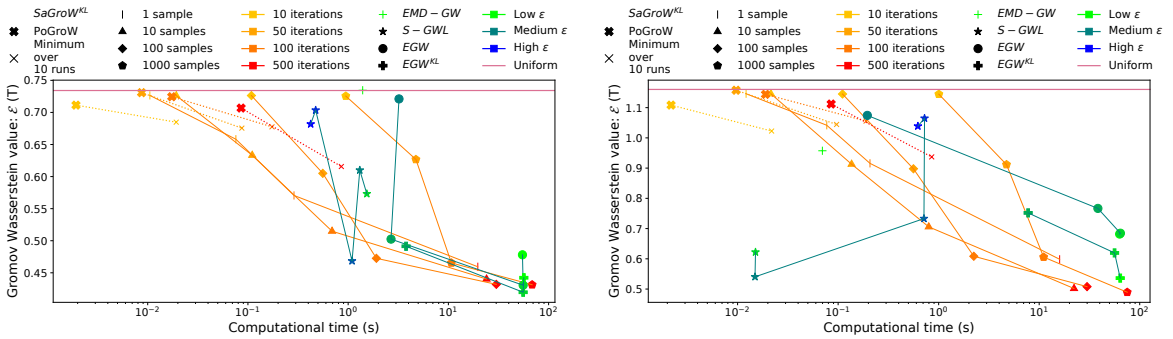


Figure C.7: *Hyperparameters analysis on a Stochastic Block Model dataset with only 20 nodes for each graphs. (**Left**) Absolute loss. (**Right**) Square loss.*

Cited on page [132]

Figure C.8 (left) shows an interesting example when every method seems stuck in the same

local minimum and S-GWL finds a better transport plan which is probably the global minimum.

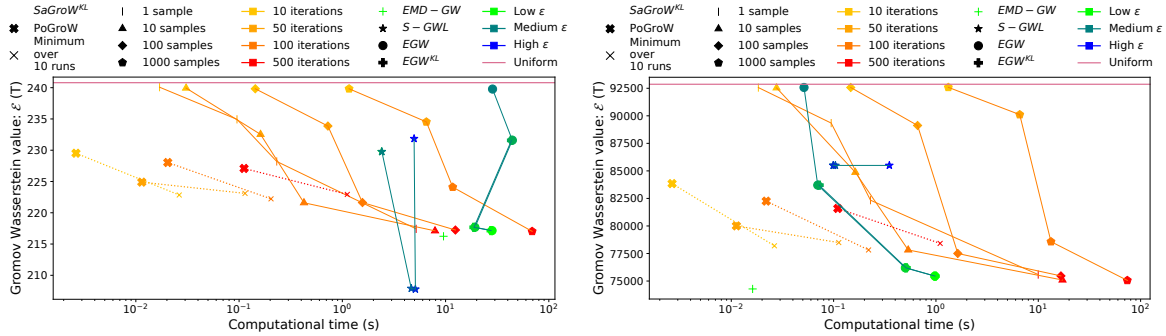


Figure C.8: *Hyperparameters analysis on a mixture of Gaussians with 100 points sampled for each distributions. (**Left**) Absolute loss. (**Right**) Square loss.*

Cited on page [132]

Lastly, Figure C.9 shows that even with 1000 iterations, SaGroW doesn't seem to converge. The value of ϵ is too high in this case and needed to be lowered to avoid too many iterations. However, SaGroW still obtains a better plan than S-GWL for the absolute loss.

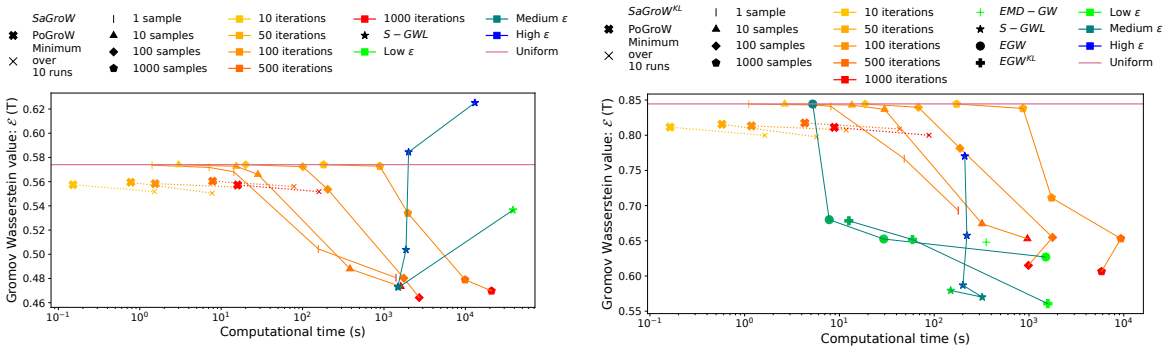


Figure C.9: *Hyperparameters analysis on a Stochastic Block Model dataset with 1000 nodes for each graphs. (**Left**) Absolute loss. (**Right**) Square loss.*

Cited on page [133]

C.2.4 Small experiment on SaGroW without the KL regularization

In this experiment, we replace SaGroW^{KL} by SaGroW and reproduce Figures 4.8, 4.9 and 4.10 in Chapter 4. We use $\epsilon = 0.1$ for this experiment and $\alpha = 0.8$. The Figure C.10 shows that the value of $\epsilon = 0.1$ is too high on this dataset. Section C.2.5 highlights the difficulty to choose a good value of entropy regularization while the KL regularization is much more robust to this choice.

As the number of sample increases, the performance of SaGroW tend to EGW$_{0.1}$, which is the left-most point. This behaviour is expected as SaGroW become similar to EGW when the expectation is completely computed. However, the performance improves slowly with the number of iterations. This might be due to the lack of memory from one iteration to the other,

as the transport plan $\boldsymbol{T}_s$ may vary a lot between two iterations. This illustrates the advantage of the KL regularization which completely take into account the previous transport plan.

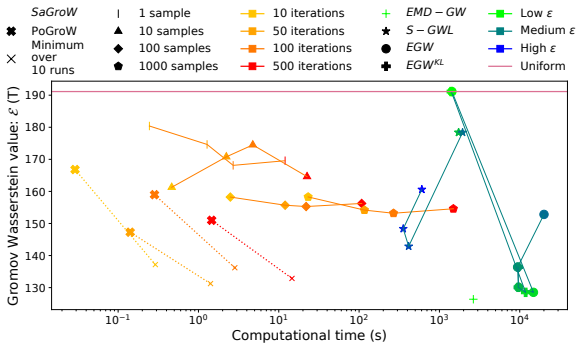


Figure C.10: *Similar to the Figures 4.8, 4.9 and 4.10 in Chapter 4, with SaGroW^{KL} replaced by SaGroW.*

Cited on page [133]

C.2.5 Small experiment on the entropy parameter

Table C.1 shows that EGW is really sensitive to the entropy regularization, with only 4 values of ϵ that give a Gromov Wasserstein distance different from 0.75. This value of 0.75 corresponds to the uniform matrix. In contrast, due to the KL regularization instead of the classical entropy, SaGroW^{KL} never returns the uniform matrix. Moreover, SaGroW^{KL} gives a reasonable value for a wide range of parameters (from 0.05 to 1).

The entropy regularization ensure to stay close to the uniform. Thus, for high value of ϵ it will always stay close to the uniform. The KL regularization ensure that the next value will be close to the previous one. In such a case, with enough iterations, one can still converge to a local minimum. This is intuition was given in (Xu et al., 2019b) based on (Xie et al., 2020).

C.2.6 Small experiment on the α parameter

Tables C.2 and C.3 analyze the impact of α and the number of iterations. The most important information is that a high value of α seems a good choice. A high value of α ensure to always be close to the edge of the polytope, were the optimal value is assumed to be. For a concave problem (Table C.3), the best value to choose is 1. In Table C.2 the best value around 0.75 - 0.9. Thus, it might not be very interesting to cross validate this parameter, and a value around 0.8 seems a reasonable choice.

On average, it is better to apply many iterations. This is especially true for small value of α were the GW distance changes very slowly. We see on this experiment, the limit of the convergence proof. In practice, we will never use a small value of α, even if the convergence is ensured.

Table C.1: *Gromov Wasserstein estimation for different values of ϵ. The dataset is composed of 2 graphs created with a Gaussian Random Partition Graph (Brandes et al., 2003) with 50 points each. The mean cluster size is set to* 25 *and the variance to* 5*. The probability of intra-cluster connection is* 0.5 *while the inter-cluster is set to* 0.1*. The GW distances reported are averaged over 10 iterations. The absolute distance is used for $\mathcal{L}$. The number of iteration of SaGroW^{KL} is* 1000 *with one sample per iteration.*

Cited on page [134]

ϵ	SaGroW^{KL}	EGW
0.001	0.73	0.75
0.005	0.59	0.63
0.01	0.55	0.62
0.05	0.51	0.67
0.1	0.51	0.71
0.5	0.52	0.75
1	0.52	0.75
5	0.62	0.75
10	0.68	0.75

Table C.2: *Gromov Wasserstein distance for PoGroW with different values of α and different number of iterations. The dataset is composed of 2 graphs created with a Gaussian Random Partition Graph (Brandes et al., 2003) with* 50 *points. The absolute distance is used for $\mathcal{L}$.*

Cited on pages [134,134]

S\α	0.001	0.01	0.1	0.25	0.5	0.75	0.9	0.99	0.999	1
1	74.87	74.87	74.82	74.63	74.03	73.07	72.33	71.82	71.77	71.76
10	74.87	74.82	74.73	74.52	74.88	74.69	72.28	71.84	71.80	71.79
100	74.82	74.35	73.36	73.94	74.67	68.34	68.76	71.91	71.93	71.94
1000	74.37	73.76	72.90	73.32	73.09	72.12	70.62	72.97	73.01	70.69

C.2.7 Graph classification

Tables C.4 and C.5 gives the complete table of the graphs' classification experiment. The best parameter taken for each of the method is not of the edge on the parameter range. Thus, a good parameter is found for each method. Notice that the performance of PoGroW are very similar for different value of power p. This can be explained by the fact that the transport plan found does not depend on the loss used. The 1D optimal transport plan is the same for all convex loss functions. For the case $p = 0.5$, PoGroW does not find the perfect transport plan at each iteration as we suppose the loss convex. For $p = 1$ the problem might be degenerated, many

Table C.3: *Gromov Wasserstein distance for PoGroW with different values of α and different number of iterations. The dataset is composed of 2 samples of 50 points of mixtures of Gaussians. The absolute distance is used for $\mathcal{L}$.*

Cited on pages [134,134]

S\α	0.001	0.01	0.1	0.25	0.5	0.75	0.9	0.99	0.999	1
1	166.88	166.87	166.28	163.53	154.03	138.40	126.08	117.62	116.73	116.63
10	166.83	166.37	163.81	160.98	117.23	85.41	85.27	85.24	79.84	79.84
100	166.48	163.31	105.36	90.42	137.56	76.93	79.02	73.80	73.64	73.62
1000	163.38	134.57	86.59	80.47	79.43	78.28	77.45	77.14	77.11	77.11

different transport plans can be optimal. We can suppose p slightly higher than one to avoid the problem. Caracciolo et al. (2020) proposes a bound for the 1D OT concave case.

Other losses than the absolute loss at power p have been tested. Only the exponential square $(1 - e^{-(\boldsymbol{C}^{\mathcal{X}} - \boldsymbol{C}^{\mathcal{Y}})^2})$ has a reasonable accuracy.

Table C.4: *Complete table of the classification experiment (1/2).*

Cited on page [134]

Dataset	Accuracy	GW Distance	Time (s)
S-GWL$_{0.005}$	0.1	400	14.7
S-GWL$_{0.01}$	0.1	400	13.9
S-GWL$_{0.05}$	0.1	400	13.9
S-GWL$_{0.1}$	0.1	400	12.6
S-GWL$_{0.5}$	0.17	390	12.0
S-GWL$_{1}$	0.29	374	11.0
S-GWL$_{5}$	0.44	362	23.4
S-GWL$_{10}$	0.41	377	27.8
S-GWL$_{50}$	0.41	374	34.1
S-GWL$_{100}$	0.39	372	33.2
EGW$_{0.0001}$	0.07	430	0.1
EGW$_{0.0005}$	0.07	429	4.5
EGW$_{0.001}$	0.22	412	36.2
EGW$_{0.005}$	0.22	375	42.6
EGW$_{0.01}$	0.12	383	25.1
EGW$_{0.05}$	0.15	408	6.2
EGW$_{0.1}$	0.12	420	2.0
EGW$_{0.5}$	0.07	429	0.3
EGW$_{1}$	0.07	429	0.3
EGW$^{KL}_{0.0001}$	0.07	430	0.1
EGW$^{KL}_{0.0005}$	0.07	429	0.1
EGW$^{KL}_{0.001}$	0.15	419	0.2
EGW$^{KL}_{0.005}$	0.24	375	41.1
EGW$^{KL}_{0.01}$	0.12	383	25.8
EGW$^{KL}_{0.05}$	0.15	408	7.1
EGW$^{KL}_{0.1}$	0.12	420	2.4
EGW$^{KL}_{0.5}$	0.07	429	0.4
EGW$^{KL}_{1}$	0.07	429	0.4
EMD-GW	0.37	341	16.6

Table C.5: *Complete table of the classification experiment (2/2.*

Cited on page [134]

Dataset	Accuracy	GW Distance	Time (s)
$\text{SaGroW}_{p=0.5}$	0.41		12.3
$\text{SaGroW}_{p=1}$	**0.49**		11.6
$\text{SaGroW}_{p=1.5}$	**0.49**		13.6
$\text{SaGroW}_{p=2}$	0.39	**336**	12.7
$\text{SaGroW}_{p=2.5}$	0.37		12.3
$\text{SaGroW}_{p=3}$	0.27		11.8
SaGroW $(1 - e^{-\lvert \boldsymbol{C}^{\mathcal{X}} - \boldsymbol{C}^{\mathcal{Y}} \rvert})$	0.05		0.7
SaGroW $(1 - e^{-\frac{\lvert \boldsymbol{C}^{\mathcal{X}} - \boldsymbol{C}^{\mathcal{Y}} \rvert}{10}})$	0.05		0.8
SaGroW $(1 - e^{-(\boldsymbol{C}^{\mathcal{X}} - \boldsymbol{C}^{\mathcal{Y}})^2})$	0.27		13.9
SaGroW $(1 - e^{\frac{(\boldsymbol{C}^{\mathcal{X}} - \boldsymbol{C}^{\mathcal{Y}})^2}{100}})$	0.05		0.8
$\text{PoGroW}_{p=0.5}$	0.37		0.5
$\text{PoGroW}_{p=1}$	0.37		0.5
$\text{PoGroW}_{p=1.5}$	0.37		0.5
$\text{PoGroW}_{p=2}$	0.39	373	0.5
$\text{PoGroW}_{p=2.5}$	0.32		0.6
$\text{PoGroW}_{p=3}$	0.27		0.5
PoGroW $(1 - e^{-(\boldsymbol{C}^{\mathcal{X}} - \boldsymbol{C}^{\mathcal{Y}})^2})$	0.2		0.5
PoGroW $(1 - e^{\lvert \boldsymbol{C}^{\mathcal{X}} - \boldsymbol{C}^{\mathcal{Y}} \rvert})$	0.05		0.5
PoGroW $(1 - e^{\frac{\lvert \boldsymbol{C}^{\mathcal{X}} - \boldsymbol{C}^{\mathcal{Y}} \rvert}{10}})$	0.05		0.5
PoGroW $(\frac{1}{e^{-(\boldsymbol{C}^{\mathcal{X}} - \boldsymbol{C}^{\mathcal{Y}})^2}} - 1)$	0.1		0.2
Uniform	0.07	430	0.1

Appendix D

Appendix of Optimal Tensor Transport

Notations

To simplify the notations, we will sometime use the following shortcuts:

$$\sum_{\forall d\ i_d} = \sum_{i_1,\dots,i_D=1}^{I_{f(1)},\dots,I_{f(D)}} \tag{D.1}$$

$$\sum_{\forall d\neq d'\ i_d} = \sum_{\substack{i_1,\dots,i_{d'-1}=1\\ i_{d'+1},\dots,i_D=1}}^{\substack{I_{f(1)},\dots,I_{f(d'-1)}\\ I_{f(d'+1)},\dots,I_{f(D)}}} \tag{D.2}$$

$$\prod_{d\neq d'} = \prod_{\substack{d=1\\ d\neq d'}}^{D} \tag{D.3}$$

D.1 Illustration of the difference between OTT and Co-OT

In this section, we provide the same pixel transportation image provided in (Redko et al., 2020) for both Co-OT and OTT_{123}. This pixel transportation is illustrated Figure D.1, where the transport plans used for OTT are the ones displayed Figure 5.1.

For Co-OT, many pixels have no information as we use only 0 and 1 labels, thus the colored USPS image has some coherence only in the middle of the image. On the other hand, with OTT_{123} we force columns to be mapped on (whole) columns and rows to be mapped on (whole) rows. By doing so, OTT can extrapolate using the row/column structure inherent to images and the color visualization is much smoother than for Co-OT.

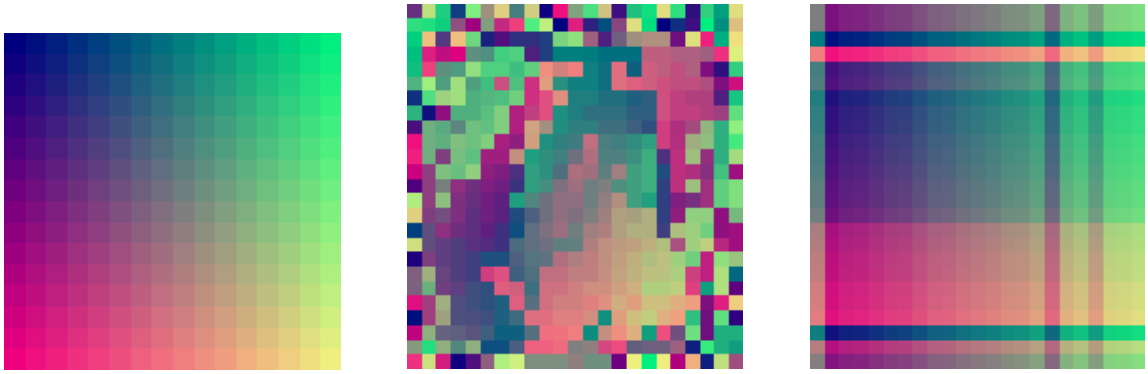


Figure D.1: *(**Left**) MNIST image with a color associated with each pixel. (**Middle** and **Right**) USPS image colored by the transportation of the left image by the transport plan found with Co-OT and OTT_{123} respectively. Few rows and columns are deleted in this representation as they have no mass.*

Cited on page [139]

D.2 Mirror Descent

In this section, we prove that the method proposed by Xu et al. (2019b) is a Mirror Descent algorithm on the original GW problem. More specifically, we prove that Equation (7) in their Section 3.1 is the same step as the step used in a Mirror Descent algorithm with Kullback-Leibler divergence. The Mirror Descent method, at a given point $\boldsymbol{T}_s$ at the iteration s, searches for the next minimum $\boldsymbol{T}_{s+1}$ with,

$$\boldsymbol{T}_{s+1} = \underset{\boldsymbol{T} \in \Pi_{\mu\nu}}{\operatorname{argmin}} \langle \nabla_{\boldsymbol{T}_s} \mathcal{E}, \boldsymbol{T} \rangle + \epsilon KL(\boldsymbol{T}, \boldsymbol{T}_s). \tag{D.4}$$

Which is equivalent to,

$$\boldsymbol{T}_{s+1} = \underset{\boldsymbol{T} \in \Pi_{\mu\nu}}{\operatorname{argmin}} \langle \nabla_{\boldsymbol{T}_s} \mathcal{E} - \epsilon log(\boldsymbol{T}_s), \boldsymbol{T} \rangle + \epsilon \langle \boldsymbol{T}, log(\boldsymbol{T}) \rangle . \tag{D.5}$$

The only difference is the missing factor 2 in (Peyré et al., 2016) that should be here due to the derivative, but both problem are equivalent with a re-scaling of ϵ by a factor 2.

D.3 Complexity of the gradient computation of OTT

We prove in this section that the gradient of OTT, which is necessary for the Mirror Descent algorithm (Beck and Teboulle, 2003), can be computed in a time complexity of $O\left(N^{D+1}\right)$ for particular loss functions. This generalizes a known result for GW (Peyré et al., 2016). Note that in the associated Chapter, we propose a more efficient approach based on sampling. We only mention this result for the sake of comparison with state-of-the-art approaches.

Here, we assume that the feature dimension F is small and that every other dimension of the tensor is N to simplify the notations. We also assume that the order of the tensor D is fixed

and small. We analyze the time complexity only with respect to N. First, we recall the gradient of $\mathcal{E}$ using the notations introduced at the start of this supplementary material:

$$\nabla_{\boldsymbol{T}^e}\mathcal{E} = \sum_{\{d'|f(d')=e\}} \sum_{\forall d\neq d'\ i_d} \sum_{\forall d\neq d'\ k_d} \mathcal{L}(\boldsymbol{X}_{i_1\dots i_{d'-1},\bullet,i_{d'+1}\dots i_D}, \boldsymbol{Y}_{k_1\dots k_{d'-1},\bullet,k_{d'+1}\dots k_D}) \prod_{d\neq d'} T^{f(d)}_{i_d k_d}. \tag{D.6}$$

We suppose that the loss $\mathcal{L}$ is continuous and can be written as $\mathcal{L}(x,y) = f_1(x)+f_2(y)-h_1(x)h_2(y)$ with four functions $(f_1, f_2 : \mathbb{R}^F \to \mathbb{R})$ and $(h_1, h_2 : \mathbb{R}^F \to \mathbb{R}^F)$. This is notably the case for the square euclidean distance or the Kullback-Leibler divergence. As each element of the first sum of Equation (D.6) will be computed independently, we can fix d'. We thus have to compute the following term,

$$\begin{aligned}
&\sum_{\forall d\neq d'\ i_d} \sum_{\forall d\neq d'\ k_d} f_1(\boldsymbol{X}_{i_1\dots i_{d'-1},\bullet,i_{d'+1}\dots i_D}) \prod_{d\neq d'} T^{f(d)}_{i_d k_d} \\
+ &\sum_{\forall d\neq d'\ i_d} \sum_{\forall d\neq d'\ k_d} f_2(\boldsymbol{Y}_{k_1\dots k_{d'-1},\bullet,k_{d'+1}\dots k_D}) \prod_{d\neq d'} T^{f(d)}_{i_d k_d} \\
+ &\sum_{\forall d\neq d'\ i_d} \sum_{\forall d\neq d'\ k_d} h_1(\boldsymbol{X}_{i_1\dots i_{d'-1},\bullet,i_{d'+1}\dots i_D}) h_2(\boldsymbol{Y}_{k_1\dots k_{d'-1},\bullet,k_{d'+1}\dots k_D}) \prod_{d\neq d'} T^{f(d)}_{i_d k_d}.
\end{aligned} \tag{D.7}$$

Note that $\boldsymbol{X}_{i_1\dots i_{d'-1},\bullet,i_{d'+1}\dots i_D}$ is a vector of size $(N,1,F)$ and as f_1 is applied only on the features dimension, $f_1(\boldsymbol{X}_{i_1\dots i_{d'-1},\bullet,i_{d'+1}\dots i_D})$ is a vector of size $(N,1)$. Similarly, $f_2(\boldsymbol{Y}_{k_1\dots k_{d'-1},\bullet,k_{d'+1}\dots k_D})$ is a vector of size $(1,N)$. As the gradient is a $(N\times N)$ matrix, the sum between the first two terms should be understood as a broadcasting sum. The same holds for $h_1(\boldsymbol{X}_{i_1\dots i_{d'-1},\bullet,i_{d'+1}\dots i_D})$ and $h_2(\boldsymbol{Y}_{k_1\dots k_{d'-1},\bullet,k_{d'+1}\dots k_D})$ of size $(N\times F)$ and $(F\times N)$, the product is a matrix of size $(N\times N)$ The first two double sums can be computed as,

$$\sum_{\forall d\neq d'\ i_d} f_1(\boldsymbol{X}_{i_1\dots i_{d'-1},\bullet,i_{d'+1}\dots i_D}) \prod_{d\neq d'} \boldsymbol{a}^{f(d)}_{i_d} + \sum_{\forall d\neq d'\ k_d} f_2(\boldsymbol{Y}_{k_1\dots k_{d'-1},\bullet,k_{d'+1}\dots k_D}) \prod_{d\neq d'} \boldsymbol{b}^{f(d)}_{k_d}. \tag{D.8}$$

There is $D-1$ sums, each of them operating over N indices. Hence, the complexity of each sum is $O\left(N^{D-1}N\right)$. As a consequence, the overall complexity of Equation (D.8) is $O\left(N^D\right)$.

The last term requires several tensor/matrix multiplications as it can be reformulated as,

$$\sum_{\forall d\neq d'\ i_d} h_1(\boldsymbol{X}_{i_1\dots i_{d'-1},\bullet,i_{d'+1}\dots i_D}) \left(\sum_{\forall d\neq d'\ k_d} h_2(\boldsymbol{Y}_{k_1\dots k_{d'-1},\bullet,k_{d'+1}\dots k_D}) \prod_{d\neq d'} T^{f(d)}_{i_d k_d} \right). \tag{D.9}$$

The $D-1$ internal sums can be seen as $D-1$ tensor/matrices multiplications, each between the index d of the tensor and the second index of the matrix $T^{f(d)}$. Each of these multiplications have a time complexity of $O\left(N^{D+1}\right)$. We provide an example, after the proof, for $D=3$. If we call this new tensor H, we can reformulate the problem as,

$$\sum_{\forall d\neq d'\ i_d} h_1(\boldsymbol{X}_{i_1\dots i_{d'-1},\bullet,i_{d'+1}\dots i_D}) H_{i_1\dots i_{d'-1},\bullet,i_{d'+1}\dots i_D}. \tag{D.10}$$

This can be seen as a standard multiplication of two tensors on every dimension except on the dimension d' for both tensor. This is equivalent to the multiplication of two matrices of size (N, N^{D-1}) and (N^{D-1}, N). The time complexity is again $O\left(N\times N^{D-1}\times N\right) = O\left(N^{D+1}\right)$.

Finally, the entire time complexity is $O\left(N^{D+1}\right)$, which is better than the naive $O\left(N^{2D}\right)$.

Example for $D = 3$. Let us explain how to compute Equation (D.9) with $D = 3$ in a $O\left(N^{3+1}\right)$ time complexity. We will suppose, without loss of generality, that $d' = 3$ in this example. We are interested in the tensor H of Equation (D.10) or equivalently of the inner sum of Equation (D.9) for all $(i_1, i_2) \in [\![1, N]\!]^2$,

$$\sum_{k_1=1}^{N} \sum_{k_2=1}^{N} h_2(\boldsymbol{Y}_{k_1,k_2,\bullet}) \boldsymbol{T}^{f(1)}_{i_1,k_1} \boldsymbol{T}^{f(2)}_{i_2,k_2} = H_{i_1,i_2,\bullet}. \tag{D.11}$$

We rearrange the terms to make the multiplication between a 3-order tensor and a transport plan appear,

$$\forall (i_1, i_2) \in [\![1, N]\!]^2 \ \sum_{k_1=1}^{N} \left(\sum_{k_2=1}^{N} h_2(\boldsymbol{Y}_{k_1,k_2,\bullet}) \boldsymbol{T}^{f(2)}_{i_2,k_2} \right) T^{f(1)}_{i_1,k_1}. \tag{D.12}$$

For all $(k_1, i_2) \in [\![1, N]\!]^2$ we note $H'_{k_1,i_2,\bullet} = \sum_{k_2=1}^{N} h_2(\boldsymbol{Y}_{k_1,k_2,\bullet}) \boldsymbol{T}^{f(2)}_{i_2,k_2}$. H' can be computed with $N^2 \times N \times N$ operations as a multiplication between a tensor (N, N, N, F) and a matrix (N, N) along the second dimension on both sides. We have now a similar formulation as in Equation (D.12) but with only one transport plan left,

$$\forall (i_1, i_2) \in [\![1, N]\!]^2 \ \sum_{k_1=1}^{N} H'_{k_1,i_2,\bullet} \boldsymbol{T}^{f(1)}_{i_1,k_1}. \tag{D.13}$$

We apply the same process, and define for all $(i_1, i_2) \in [\![1, N]\!]^2$, $H_{i_1,i_2,\bullet} = \sum_{k_1=1}^{N} H'_{k_1,i_2,\bullet} \boldsymbol{T}^{f(1)}_{i_1,k_1}$. This new tensor H can be computed with $N^2 \times N \times N$ operations. The difference is that the dimension of the sum is the first one for the tensor H'.

We finally have to compute,

$$\sum_{i_1=1}^{N} \sum_{i_2=1}^{N} h_1(\boldsymbol{X}_{i_1,i_2,\bullet}) H_{i_1,i_2,\bullet}, \tag{D.14}$$

which is equivalent to the Equation (D.9). We can see it as a multiplication between two tensors of size (N, N, N, F) along the first two dimensions as well as the last dimension. Thus, the time complexity is $O\left(N \times N^2 \times N\right)$, that is $O\left(N^4\right)$.

D.4 Theoretical results

We prove that OTT is a distance for weighted tensors using (Villani, 2008; Chowdhury and Mémoli, 2019; Redko et al., 2020).

Theorem 9. *OTT is a distance between weighted tensors represented in canonical form ($\boldsymbol{X}$, $(\boldsymbol{a}^e)_{e \in [\![1,E]\!]}$) and ($\boldsymbol{Y}$, $(\boldsymbol{b}^e)_{e \in [\![1,E]\!]}$) for any affectation function f, as long as $\mathcal{L}$ is a proper distance.*

Proof. We fix a tensor order D, an affectation function $f : [\![1, D]\!] \to [\![1, E]\!]$ and a loss $\mathcal{L}$.

We start the proof by explaining the natural canonical form of a dataset. Let $(\boldsymbol{X}, (\boldsymbol{a}^e)_{e \in [\![1,E]\!]})$ be a weighted tensor.

- For all $e \in [\![1, E]\!]$ and for all $i \in [\![1, I^e]\!]$ such that $\boldsymbol{a}_i^e = 0$ we delete this weighted and all the corresponding value in the tensor $\boldsymbol{X}$. This is a natural reduction as if the weight is 0 this is equivalent to no point.

- Duplicated points should be merged together. Two points that would be transported by the same transport plan $\boldsymbol{T}^e$ for a fixed $e \in [\![1, E]\!]$, describe by $(i_e, i'_e) \in [\![1, I_e]\!]^2$ are equal if,

$$\forall d' \in [\![1, D]\!] | f(d') = e \,:\, \boldsymbol{X}_{\bullet_1, \ldots, \bullet_{d'-1}, i_e, \bullet_{d'+1}, \ldots, \bullet_D} = \boldsymbol{X}_{\bullet_1, \ldots, \bullet_{d'-1}, i'_e, \bullet_{d'+1}, \ldots, \bullet_D}. \tag{D.15}$$

  Those extracted $(D-1)$-order tensors define entirely each of those points. If two points are equal, then we delete one of them and add the two probability $\boldsymbol{a}_{i_e}^e$ and $\boldsymbol{a}_{i'_e}^e$. Notice that we look at every dimension of the tensor associated with the transport plan $\boldsymbol{T}^e$ and delete simultaneously the points in every dimension. This result is quite logical in a vector space, if two points are in the same location, they should be merged.

- The weighted dataset is invariant to any permutation of its points. To create a canonical form, we can reorder it using any rules. We propose a "minimum" rule that will put the smallest point first. We only have to define an inequality, that respect at least the converse and the transitivity proprieties, to compare two points $(i_e, i'_e) \in [\![1, I_e]\!]^2$ for a fixed $e \in [\![1, E]\!]$. Using the description of a point above, we simply compare the real value one by one. First, we take the smallest dimension, $d' = min(\{d' \in [\![1, D]\!] | f(d') = e\})$, and compare the first value,

$$\boldsymbol{X}_{1, \ldots, 1, i_e, 1, \ldots, 1} \text{ and } \boldsymbol{X}_{1, \ldots, 1, i'_e, 1, \ldots, 1}, \tag{D.16}$$

  then, if they are equal, we move to the next and compare,

$$\boldsymbol{X}_{2, \ldots, 1, i_e, 1, \ldots, 1} \text{ and } \boldsymbol{X}_{2, \ldots, 1, i'_e, 1, \ldots, 1}), \tag{D.17}$$

  and so on. If the two entire extracted tensors are equal, we do it again with the second smallest dimension d''. If all the extracted tensors are equal, the two points should be merged according to the canonical form describe above. Thus, we can reorder the points, notice that the same permutation should be made also for the weights $\boldsymbol{a}^e$.

None of those three modifications of the tensor $\boldsymbol{X}$ will change the transport plan nor the OTT distance that depend on $\boldsymbol{X}$. All those reductions to canonical form could be avoided if we suppose that the tensor $\boldsymbol{X}$ is generated from distributions in vector space. But in this chapter, we have focused on the practical part and suppose only the existence of the tensor $\boldsymbol{X}$. We will now prove that OTT is a distance.

Symmetry As $\mathcal{L}$ is symmetric, OTT is also symmetric.

Positiveness As $\mathcal{L}$ and T are always positive, OTT is always positive.

Identity of indiscernibles To prove that $OTT(\boldsymbol{X}, \boldsymbol{X}, (\boldsymbol{a}^e)_{e\in[\![1,E]\!]}, (\boldsymbol{a}^e)_{e\in[\![1,E]\!]}) = 0$, we set every transport plan T^e to the identity matrix. As $\forall x\ \mathcal{L}(x,x) = 0$, the entire sum is 0. But because the minimum is smaller than this particular case and also always positive, we have the desired result $OTT(\boldsymbol{X}, \boldsymbol{X}, (\boldsymbol{a}^e)_{e\in[\![1,E]\!]}, (\boldsymbol{a}^e)_{e\in[\![1,E]\!]}) = 0$.

We will now prove the opposite, let $(\boldsymbol{X}, \boldsymbol{Y})$ be D-order tensors datasets of sizes $(I_{f(1)} \times ... \times I_{f(D)} \times F, K_{f(1)} \times ... \times K_{f(D)} \times F)$ with weights $(\boldsymbol{a}^e \in \Delta_{I_e}, \boldsymbol{b}^e \in \Delta_{K_e})_{e\in[\![1,E]\!]}$ in a canonical form. The canonical form is inspired from the proof that GW is a distance for graphs (Chowdhury and Mémoli, 2019). Let us suppose that $OTT(\boldsymbol{X}, \boldsymbol{Y}) = 0$, we will show that $\boldsymbol{X} = \boldsymbol{Y}$. Furthermore, we will note $(T^e)_{e\in[\![1,E]\!]}$ the optimal transport plans.

We will proceed by contradiction and suppose that there exist two strictly positive values in the same row of a transport plan, more precisely we suppose that there exist $e \in [\![1, E]\!]$, $i \in [\![1, I_e]\!]$, $(k, k') \in [\![1, K_e]\!]^2$ such that $\boldsymbol{T}^e_{i,k} > 0$ and $\boldsymbol{T}^e_{i,k'} > 0$. Let us fix $d' \in \{d' \in [\![1, D]\!] | f(d') = e\}$ and all the indices

$$(k_1...k_{d'-1}, k_{d'+1}...k_D) \in ([\![1, K_{f(1)}]\!]...[\![1, K_{f(d'-1)}]\!], [\![1, K_{f(d'+1)}]\!]...[\![1, K_{f(D)}]\!]). \tag{D.18}$$

As all the marginal are strictly positive, there exist

$$(i_1...i_{d'-1}, i_{d'+1}...i_D) \in ([\![1, I_{f(1)}]\!]...[\![1, I_{f(d'-1)}]\!], [\![1, I_{f(d'+1)}]\!]...[\![1, I_{f(D)}]\!]) \tag{D.19}$$

such that $T^{f(d)}_{i_d,k_d} > 0$ for all $d \in [\![1, D]\!]$ with $d \neq d'$. As the transport plans are strictly positive on those indices and $\boldsymbol{T}^e_{i,k} > 0$ and $\boldsymbol{T}^e_{i,k'} > 0$, the transport plans' product is strictly positive, thus the loss should be equal to 0,

$$\mathcal{L}\left(\boldsymbol{X}_{i_1...i_{d'-1},i,i_{d'+1}...i_D}, \boldsymbol{Y}_{k_1...k_{d'-1},k,k_{d'+1}...k_D}\right) = 0, \tag{D.20}$$

$$\mathcal{L}\left(\boldsymbol{X}_{i_1...i_{d'-1},i,i_{d'+1}...i_D}, \boldsymbol{Y}_{k_1...k_{d'-1},k',k_{d'+1}...k_D}\right) = 0. \tag{D.21}$$

As the loss is a distance, we have $\boldsymbol{Y}_{k_1...k_{d'-1},k,k_{d'+1}...k_D} = \boldsymbol{Y}_{k_1...k_{d'-1},k',k_{d'+1}...k_D}$. This is true for all $d' \in \{d' \in [\![1, D]\!] | f(d') = e\}$ and all the indices

$$(k_1...k_{d'-1}, k_{d'+1}...k_D) \in ([\![1, K_{f(1)}]\!]...[\![1, K_{f(d'-1)}]\!], [\![1, K_{f(d'+1)}]\!]...[\![1, K_{f(D)}]\!]), \tag{D.22}$$

thus, those two points should have been merged, this is in contradiction with the canonical form supposition. We can do the same for the columns of the transport plans instead of the rows.

We know that the transport plans have only one element in each of its rows and columns, thus they are necessary squared matrices as all the marginal are strictly positive. Let α be the smallest strictly positive values of the transport plans $(\boldsymbol{T}^e)_{e\in[\![1,E]\!]}$. We also define the permutation matrices $\boldsymbol{P}^e$ associated to each $\boldsymbol{T}^e$ by replacing each strictly positive value in $\boldsymbol{T}^e$ to 1 in $\boldsymbol{P}^e$. We have $\alpha^{I_e}\boldsymbol{P}^e \leq \boldsymbol{T}^e$ elements-wise, thus

$$0 \leq \sum_{\forall d\ i_d} \sum_{\forall d\ k_d} \mathcal{L}(\boldsymbol{X}_{i_1,...,i_D}, \boldsymbol{Y}_{k_1,...,k_D}) \prod_{d=1}^{D} \alpha^{I_{f(d)}} \boldsymbol{P}^{f(d)}_{i_d,k_d} \tag{D.23}$$

$$\leq \sum_{\forall d\ i_d} \sum_{\forall d\ k_d} \mathcal{L}(\boldsymbol{X}_{i_1,...,i_D}, \boldsymbol{Y}_{k_1,...,k_D}) \prod_{d=1}^{D} \boldsymbol{T}^{f(d)}_{i_d,k_d} \tag{D.24}$$

$$=0. \tag{D.25}$$

If we note σ^e the permutation function associated with $\boldsymbol{P}^e$, we have,

$$0 = \sum_{\forall d\ i_d} \mathcal{L}(\boldsymbol{X}_{i_1,\dots,i_D}, \boldsymbol{Y}_{\sigma^{f(1)}(i_1),\dots,\sigma^{f(D)}(i_D)}). \tag{D.26}$$

As we have decided that the datasets are invariant to any points permutations, if both $\boldsymbol{X}$ and $\boldsymbol{Y}$ are in canonical form, they are equal. Since those permutations correspond to the transport plans $(\boldsymbol{T}^e_{e\in[\![1,E]\!]}$, we have $\boldsymbol{a}^e_i = \boldsymbol{b}^e_{\sigma^e(i)}$ for all $e \in [\![1, E]\!]$ and $i \in [\![1, I_e]\!]$. Thus, we have also $\boldsymbol{a}^e = \boldsymbol{b}^e$.

We prove that:

$$OTT(\boldsymbol{X}, \boldsymbol{Y}, (\boldsymbol{a}^e)_{e\in[\![1,E]\!]}, (\boldsymbol{b}^e)_{e\in[\![1,E]\!]}) = 0 \iff (\boldsymbol{X}, (\boldsymbol{a}^e)_{e\in[\![1,E]\!]}) = (\boldsymbol{Y}, (\boldsymbol{b}^e)_{e\in[\![1,E]\!]}). \tag{D.27}$$

Triangle inequality We now prove the triangle inequality. Let $(\boldsymbol{X}, \boldsymbol{Y}, \boldsymbol{Z})$ be D-order tensors datasets of sizes $(I_{f(1)} \times ... \times I_{f(D)} \times F, K_{f(1)} \times ... \times K_{f(D)} \times F, J_{f(1)} \times ... \times J_{f(D)} \times F)$ with weights $(\boldsymbol{a}^e \in \Delta_{I_e}, \boldsymbol{b}^e \in \Delta_{K_e}, \boldsymbol{r}^e \in \Delta_{J_e})_{e\in[\![1,E]\!]}$. We suppose $\forall d \in [\![1, D]\!]\ \boldsymbol{r}^{f(d)} > 0$ as we can delete the corresponding $(D-1)$-order tensor in the tensor Z if one term is 0 and it would not alter the transport plan nor the distance. We note $(\boldsymbol{T}^e_{xy})_{e\in[\![1,E]\!]}$ the optimal transport plans between $\boldsymbol{X}$ and $\boldsymbol{Y}$ and $(\boldsymbol{T}^e_{yz})_{e\in[\![1,E]\!]}$ the optimal transport plans between $\boldsymbol{Y}$ and $\boldsymbol{Z}$. Similarly to Redko et al. (2020) we will use the gluing lemma (Villani, 2008) to construct a coupling between $\boldsymbol{X}$ and $\boldsymbol{Z}$,

$$\forall e \in [\![1, E]\!]\ (\boldsymbol{T}^e_{xz}) = (\boldsymbol{T}^e_{xy})\,\mathrm{diag}\left(\frac{1}{\boldsymbol{b}^e}\right)(\boldsymbol{T}^e_{yz}). \tag{D.28}$$

Let $e \in [\![1, E]\!]$, we show that $(\boldsymbol{T}^e_{xz}) \in \mathcal{U}_{\boldsymbol{a}^e\boldsymbol{r}^e}$:

$$\forall j \in [\![1, J_a]\!]\ \sum_{i=1}^{I_e}(\boldsymbol{T}^e_{xz})_{ij} = \sum_{i=1}^{I_e}\sum_{k=1}^{K_e}(\boldsymbol{T}^e_{xy})_{ik}\frac{1}{\boldsymbol{b}^e_k}(\boldsymbol{T}^e_{yz})_{kj} \tag{D.29}$$

$$= \sum_{k=1}^{K_e}(\boldsymbol{T}^e_{yz})_{kj} \tag{D.30}$$

$$= \boldsymbol{r}^a_j, \tag{D.31}$$

$$\forall i \in [\![1, I_a]\!]\ \sum_{j=1}^{J_e}(\boldsymbol{T}^e_{xz})_{ij} = \sum_{j=1}^{J_e}\sum_{k=1}^{K_e}(\boldsymbol{T}^e_{xy})_{ik}\frac{1}{\boldsymbol{b}^e_k}(\boldsymbol{T}^e_{yz})_{kj} \tag{D.32}$$

$$= \sum_{k=1}^{K_e}(\boldsymbol{T}^e_{xy})_{ik} \tag{D.33}$$

$$= \boldsymbol{a}^a_i. \tag{D.34}$$

We now prove the triangle inequality:

$$OTT(\boldsymbol{X}, \boldsymbol{Z}, (\boldsymbol{a}^e)_{e\in[\![1,E]\!]}, (\boldsymbol{r}^e)_{e\in[\![1,E]\!]}) \tag{D.35}$$

$$\leq \sum_{i_1,\dots,i_D=1}^{I_{f(1)},\dots,I_{f(D)}} \sum_{j_1,\dots,j_D=1}^{J_{f(1)},\dots,J_{f(D)}} \mathcal{L}(\boldsymbol{X}_{i_1,\dots,i_D}, \boldsymbol{Z}_{j_1\dots,j_D}) \prod_{d=1}^{D} (\boldsymbol{T}_{xz}^{f(d)})_{i_d,j_d} \tag{D.36}$$

$$= \sum_{i_1,\dots,i_D=1}^{I_{f(1)},\dots,I_{f(D)}} \sum_{j_1,\dots,j_D=1}^{J_{f(1)},\dots,J_{f(D)}} \sum_{k_1,\dots,k_D=1}^{K_{f(1)},\dots,K_{f(D)}} \mathcal{L}\left(\boldsymbol{X}_{i_1,\dots,i_D}, \boldsymbol{Z}_{j_1,\dots,j_D}\right) \prod_{d=1}^{D} \frac{(\boldsymbol{T}_{xy}^{f(d)})_{i_d k_d} (\boldsymbol{T}_{yz}^{f(d)})_{k_d j_d}}{\boldsymbol{b}_{k_d}^{f(d)}} \tag{D.37}$$

$$\begin{aligned}
&\leq \sum_{\forall d\ i_d} \sum_{\forall d\ j_d} \sum_{\forall d\ k_d} \mathcal{L}\left(\boldsymbol{X}_{i_1,\dots,i_D}, \boldsymbol{Y}_{k_1,\dots,k_D}\right) \prod_{d=1}^{D} \frac{(\boldsymbol{T}_{xy}^{f(d)})_{i_d k_d} (\boldsymbol{T}_{yz}^{f(d)})_{k_d j_d}}{\boldsymbol{b}_{k_d}^{f(d)}} \\
&\quad + \sum_{\forall d\ i_d} \sum_{\forall d\ j_d} \sum_{\forall d\ k_d} \mathcal{L}\left(\boldsymbol{Y}_{k_1,\dots,k_D}, \boldsymbol{Z}_{j_1,\dots,j_D}\right) \prod_{d=1}^{D} \frac{(\boldsymbol{T}_{xy}^{f(d)})_{i_d k_d} (\boldsymbol{T}_{yz}^{f(d)})_{k_d j_d}}{\boldsymbol{b}_{k_d}^{f(d)}}
\end{aligned} \tag{D.38}$$

$$\begin{aligned}
&= \sum_{\forall d\ i_d} \sum_{\forall d\ k_d} \mathcal{L}\left(\boldsymbol{X}_{i_1,\dots,i_D}, \boldsymbol{Y}_{k_1,\dots,k_D}\right) \prod_{d=1}^{D} (\boldsymbol{T}_{xy}^{f(d)})_{i_d k_d} \sum_{\forall d\ j_d} \prod_{d=1}^{D} \frac{(\boldsymbol{T}_{yz}^{f(d)})_{k_d j_d}}{\boldsymbol{b}_{k_d}^{f(d)}} \\
&\quad + \sum_{\forall d\ j_d} \sum_{\forall d\ k_d} \mathcal{L}\left(\boldsymbol{Y}_{k_1,\dots,k_D}, \boldsymbol{Z}_{j_1,\dots,j_D}\right) \prod_{d=1}^{D} (\boldsymbol{T}_{yz}^{f(d)})_{k_d j_d} \sum_{\forall d\ i_d} \prod_{d=1}^{D} \frac{(\boldsymbol{T}_{xy}^{f(d)})_{i_d k_d}}{\boldsymbol{b}_{k_d}^{f(d)}}
\end{aligned} \tag{D.39}$$

$$\begin{aligned}
&= \sum_{\forall d\ i_d} \sum_{\forall d\ k_d} \mathcal{L}\left(\boldsymbol{X}_{i_1,\dots,i_D}, \boldsymbol{Y}_{k_1,\dots,k_D}\right) \prod_{d=1}^{D} (\boldsymbol{T}_{xy}^{f(d)})_{i_d k_d} \prod_{d=1}^{D} \sum_{j_d=1}^{J_{f(d)}} \frac{(\boldsymbol{T}_{yz}^{f(d)})_{k_d j_d}}{\boldsymbol{b}_{k_d}^{f(d)}} \\
&\quad + \sum_{\forall d\ j_d} \sum_{\forall d\ k_d} \mathcal{L}\left(\boldsymbol{Y}_{k_1,\dots,k_D}, \boldsymbol{Z}_{j_1,\dots,j_D}\right) \prod_{d=1}^{D} (\boldsymbol{T}_{yz}^{f(d)})_{k_d j_d} \prod_{d=1}^{D} \sum_{i_d=1}^{I_{f(d)}} \frac{(\boldsymbol{T}_{xy}^{f(d)})_{i_d k_d}}{\boldsymbol{b}_{k_d}^{f(d)}}
\end{aligned} \tag{D.40}$$

$$\begin{aligned}
&= \sum_{\forall d\ i_d} \sum_{\forall d\ k_d} \mathcal{L}\left(\boldsymbol{X}_{i_1,\dots,i_D}, \boldsymbol{Y}_{k_1,\dots,k_D}\right) \prod_{d=1}^{D} (\boldsymbol{T}_{xy}^{f(d)})_{i_d k_d} \\
&\quad + \sum_{\forall d\ j_d} \sum_{\forall d\ k_d} \mathcal{L}\left(\boldsymbol{Y}_{k_1,\dots,k_D}, \boldsymbol{Z}_{j_1,\dots,j_D}\right) \prod_{d=1}^{D} (\boldsymbol{T}_{yz}^{f(d)})_{k_d j_d}
\end{aligned} \tag{D.41}$$

$$= OTT(\boldsymbol{X}, \boldsymbol{Y}, (\boldsymbol{a}^e)_{e\in[\![1,E]\!]}, (\boldsymbol{b}^e)_{e\in[\![1,E]\!]}) + OTT(\boldsymbol{Y}, \boldsymbol{Z}, (\boldsymbol{b}^e)_{e\in[\![1,E]\!]}, (\boldsymbol{r}^e)_{e\in[\![1,E]\!]}). \tag{D.42}$$

In Equation (D.39), the product/sum inversion is possible as no element in the product depend on $(j_d)_{d\in[\![1,D]\!]}$. Similarly, in Equation (D.40), the sum product inversion is allowed as only one term in the product depends on the sum. In addition, each of those sums are equal to 1 by definition of $(\boldsymbol{T}_{xy}^e)_{e\in[\![1,E]\!]}$, this leads to Equation (D.41).

OTT respects the triangle inequality. □

We now propose to prove Theorem 10 with a more general formulation than Theorem 6 proposed in Chapter 5. We extend to the case where the loss in a function of $\mathbb{R}^F \to \mathbb{R}$ for F, not necessarily restricted to 1 as supposed in Chapter 5 to simplify the notations. First, we recall the definition of OTT barycenter.

Definition 3. *(OTT barycenter) Given $B \in \mathbb{N}$ weighted tensors of sizes $((K_e^b)_{e\in[\![1,E]\!]})_{b\in[\![1,B]\!]}$, $\left(\boldsymbol{X}^b \in \mathbb{R}^{K_{f(1)}^b \dots K_{f(D)}^b \times F}, (\boldsymbol{b}^{e,b} \in \Delta_{K_e^b})_{e\in[\![1,E]\!]}\right)_{b\in[\![1,B]\!]}$. Let $(\lambda_1, \dots, \lambda_B) \in \Delta_B$ be the weights quantifying the importance of each tensor.*

For fixed size $(I_e)_{e\in[\![1,E]\!]}$ and weights $(\boldsymbol{a}^e \in \Delta_{I_e})_{e\in[\![1,E]\!]}$, the OTT barycenter is defined as follows:

$$\min_{\boldsymbol{X}\in\mathbb{R}^{I_{f(1)}\cdots I_{f(D)}\times F}} \sum_{b=1}^{B} \lambda_b OTT(\boldsymbol{X}, \boldsymbol{X}^b, (\boldsymbol{a}^e)_{e\in[\![1,E]\!]}, (\boldsymbol{b}^{e,b})_{e\in[\![1,E]\!]}). \tag{D.43}$$

Theorem 10. *Assume that the loss $\mathcal{L}$ is continuous and can be written as $\mathcal{L}(\boldsymbol{x},\boldsymbol{y}) = f_1(\boldsymbol{x}) + f_2(\boldsymbol{y}) - h_1(\boldsymbol{x})h_2(\boldsymbol{y})$ with four functions $(f_1, f_2 : \mathbb{R}^F \to \mathbb{R})$ and $(h_1, h_2 : \mathbb{R}^F \to \mathbb{R}^F)$ such that the function $(\nabla h_1)^{-1l}\nabla f_1 : \mathbb{R}^F \longrightarrow \mathbb{R}^F$ is invertible, where the $F\times F$ matrix $(\nabla h_1)^{-1l}(x)$ is the left inverse of the $F\times F$ matrix $\nabla h_1(\boldsymbol{x})$, $\forall \boldsymbol{x}\in\mathbb{R}^F$. We also suppose that, $\forall r\in[\![1,F]\!]$ $\mathcal{L}(\boldsymbol{x},\boldsymbol{y}) \underset{\boldsymbol{x}_r\to\pm\infty}{\longrightarrow} +\infty$. For fixed $(\boldsymbol{T}^{e,b})_{e\in[\![1,E]\!]}$, the optimal solution $\boldsymbol{X}^*$ of Problem (D.43) reads,*

$$\boldsymbol{X}^*_{i_1,\dots,i_D} = \left((\nabla h_1)^{-1l}\nabla f_1\right)^{-1} \left(\sum_{b=1}^{B}\lambda_b \sum_{k_1,\dots,k_D=1}^{K_1^b,\dots,K_D^b} h_2(\boldsymbol{X}^b_{k_1,\dots,k_D}) \prod_{d=1}^{D} \frac{\boldsymbol{T}^{f(d),b}_{i_d k_d}}{\boldsymbol{a}^{f(d)}_{i_d}}\right) \tag{D.44}$$

for all $(i_d \in [\![1, I_{f(d)}]\!])_{d\in[\![1,D]\!]}$.

In particular, when $\mathcal{L}$ is the squared euclidean distance,

$$\boldsymbol{X}^*_{i_1,\dots,i_D} = \sum_{b=1}^{B}\lambda_b \sum_{k_1,\dots,k_D=1}^{K_1^b,\dots,K_D^b} \boldsymbol{X}^b_{k_1\dots k_D} \prod_{d=1}^{D} \frac{\boldsymbol{T}^{f(d),b}_{i_d k_d}}{\boldsymbol{a}^{f(d)}_{i_d}}. \tag{D.45}$$

Proof. For fixed $((\boldsymbol{T}^{e,b})_{e\in[\![1,E]\!]})_{b\in[\![1,B]\!]}$, we are looking for the derivative of Equation (5.8) with respect to $\boldsymbol{X}_{i_1,\dots,i_D}$ and we equate it to 0 to find the optimal solution. $\boldsymbol{X}_{i_1,\dots,i_D}$ might be a vector if the features dimension is not reduced to 1. Each gradient in the following equation is a vector of size F.

$$0 = \sum_{b=1}^{B}\lambda_b \sum_{k_1,\dots,k_D=1}^{K_1^b,\dots,K_D^b} \nabla_{\boldsymbol{X}_{i_1,\dots,i_D}} \mathcal{L}(\boldsymbol{X}_{i_1,\dots,i_D}, \boldsymbol{X}^b_{k_1,\dots,k_D}) \prod_{d=1}^{D} \boldsymbol{T}^{f(d),b}_{i_d k_d} \tag{D.46}$$

$$\iff 0 = \sum_{b=1}^{B}\lambda_b \sum_{\forall d\ k_d} \left(\nabla f_1\left(\boldsymbol{X}_{i_1,\dots,i_D}\right) - \nabla h_1(\boldsymbol{X}_{i_1,\dots,i_D}) h_2(\boldsymbol{X}^b_{k_1,\dots,k_D})\right) \prod_{d=1}^{D} \boldsymbol{T}^{f(d),b}_{i_d k_d} \tag{D.47}$$

$$\iff 0 = \sum_{b=1}^{B}\lambda_b \sum_{\forall d\ k_d} \left(\nabla f_1\left(\boldsymbol{X}_{i_1,\dots,i_D}\right) - \nabla h_1(\boldsymbol{X}_{i_1,\dots,i_D}) h_2(\boldsymbol{X}^b_{k_1,\dots,k_D})\right) \prod_{d=1}^{D} \boldsymbol{T}^{f(d),b}_{i_d k_d} \tag{D.48}$$

$$\begin{aligned}\iff 0 = {} & \nabla f_1\left(\boldsymbol{X}_{i_1,\dots,i_D}\right) \sum_{b=1}^{B}\lambda_b \sum_{\forall d\ k_d} \prod_{d=1}^{D} \boldsymbol{T}^{f(d),b}_{i_d k_d} \\ & - \nabla h_1(\boldsymbol{X}_{i_1,\dots,i_D}) \sum_{b=1}^{B}\lambda_b \sum_{\forall d\ k_d} h_2(\boldsymbol{X}^b_{k_1,\dots,k_D}) \prod_{d=1}^{D} \boldsymbol{T}^{f(d),b}_{i_d k_d}\end{aligned} \tag{D.49}$$

$$\iff \nabla f_1\left(\boldsymbol{X}_{i_1,\dots,i_D}\right) \sum_{b=1}^{B}\lambda_b \prod_{d=1}^{D} \boldsymbol{a}^{f(d)}_{i_d} = \nabla h_1(\boldsymbol{X}_{i_1,\dots,i_D}) \sum_{b=1}^{B}\lambda_b \sum_{\forall d\ k_d} h_2(\boldsymbol{X}^b_{k_1,\dots,k_D}) \prod_{d=1}^{D} \boldsymbol{T}^{f(d),b}_{i_d k_d} \tag{D.50}$$

$$\iff (\nabla h_1)^{-1l}(\boldsymbol{X}_{i_1,\dots,i_D})\nabla f_1(\boldsymbol{X}_{i_1,\dots,i_D}) \prod_{d=1}^{D} \boldsymbol{a}^{f(d)}_{i_d} = \sum_{b=1}^{B}\lambda_b \sum_{\forall d\ k_d} h_2(\boldsymbol{X}^b_{k_1,\dots,k_D}) \prod_{d=1}^{D} \boldsymbol{T}^{f(d),b}_{i_d k_d} \tag{D.51}$$

$$\iff \boldsymbol{X}_{i_1,\dots,i_D} = \left((\nabla h_1)^{-1l}\nabla f_1\right)^{-1} \left(\sum_{b=1}^{B}\lambda_b \sum_{\forall d\ k_d} h_2(\boldsymbol{X}^b_{k_1,\dots,k_D}) \prod_{d=1}^{D} \frac{\boldsymbol{T}^{f(d),b}_{i_d k_d}}{\boldsymbol{a}^{f(d)}_{i_d}}\right) \tag{D.52}$$

Table D.1: *Error for the gradient approximation for an increasing number of samples.*
Cited on page [148]

M	$\|\|\hat{\nabla}\mathcal{E} - \nabla\mathcal{E}\|\|_F / \|\|\nabla\mathcal{E}\|\|_F$
1	1.036 ± 0.1068
5	0.472 ± 0.0247
10	0.353 ± 0.0298
50	0.157 ± 0.0184
100	0.112 ± 0.0108
500	0.050 ± 0.0048
1000	0.034 ± 0.0023
5000	0.017 ± 0.0027
10000	0.011 ± 0.0008
50000	0.005 ± 0.0004

There is only one vector $\boldsymbol{X}_{i_1,\ldots,i_D}$ found as $(\nabla h_1)^{-1l}\nabla f_1$ is invertible. Thus, if we suppose that $\boldsymbol{X}_{i_1,\ldots,i_D}$ is a maxima or an inflection point, then there is no minimum in $\mathbb{R}^{I_{f(1)}\times\ldots\times I_{f(D)}\times F}$ as OTT is continuous. This is in contradiction with the hypothesis that the loss tends to $+\infty$ when $x_r \longrightarrow +\infty$ for any $r \in [\![1, F]\!]$. Thus, the value obtained is a minima.

For the squared euclidean distance, for any $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^F$, $\nabla h_1(\boldsymbol{x})$ is the identity matrix, $h_2(\boldsymbol{y}) = 2\boldsymbol{y}$ and $\nabla f_1(\boldsymbol{x}) = 2\boldsymbol{x}$. □

D.5 Experiments

D.5.1 The number of samples M

In this section, we will discuss how we chose M in practice for all the experiments. The main idea is to chose M to obtain the best trade-off between the computational efficiency of each iteration and the precision of the gradient approximation. Hence, from a time complexity point of view, assuming that the tensors are cubic (with each dimension of size N), the main bottleneck in terms of efficiency (excluding the gradient approximation) would be the Sinkhorn algorithm with a complexity $O(PN^2)$, independent of D, where P is the number of Sinkhorn iterations which is usually of the order of 100 or 1000. Thus, since the gradient approximation step has a complexity of $O(MN^2)$, M should be of the same order as P to avoid a needless increase in terms of time complexity. From a precision of the gradient approximation point of view, choosing M as large as possible is always beneficial. This is illustrated in the following experiment where we apply OTT_{122} between two datasets of the DA experiments, "Thriller" and "Fantasy" (without any label information). We evaluate the precision of the gradient approximation during the 50th iteration with increasing values of M in Table D.1 The results correspond to the means and standard deviations obtained over 10 runs. In this experiment, $D = 2$, $I_1 = 100$, and $I_2 = 200$,

thus the number of possible samples is $100^2 \times 200^2 = 4 \times 10^8$. Here, we notice that choosing M close to S, that is 1000, already leads to good approximations. Choosing larger values would only lead to marginal improvements. Figure D.16 also shows that the target accuracy in a Domain Adaptation task increase with a higher number of samples M.

D.5.2 Domain Adaptation (DA)

Hyperparameters

This section will describe the key hyperparameters used by all the methods for reproductibility purpose. Except for the Kullback-Leibler regularization ϵ and class regularization η, we keep the default parameters provided with the code of each algorithm.

S-GWL

- Loss: squared euclidean distance
- Outer iteration: 4000
- Max iteration for the barycenter: 4
- We use the automatic update of the marginal $\boldsymbol{a}$ proposed by S-GWL. Similar results were obtained without any automatic update of the marginal.

OTT

- Loss: squared euclidean distance
- Number of samples M: 1000
- Number of iterations S: 1000
- Number of outer iterations of OTDA (Courty et al., 2017b): 10
- Number of inner iterations of OTDA (Courty et al., 2017b): 200

Co-OT

- Loss: squared euclidean distance
- Number of iterations S: 10
- Number of outer iterations of OTDA (Courty et al., 2017b): 10
- Number of inner iterations of OTDA (Courty et al., 2017b): 200

GW

- Loss: squared euclidean distance
- Number of iterations S: 1000
- Number of outer iterations of OTDA (Courty et al., 2017b): 10
- Number of inner iterations of OTDA (Courty et al., 2017b): 200

SVM

- Square L2 penality C: 1.0 (Any value will give the same result has there is only 1 point per class)

Figures for each parameter and datasets

In this section we present the Figures evoked in Section 5.6.1 for each dataset instead of the average. We also display in Figures D.2 to D.8 the values of the various computed distances rescaled between 0 and 1 for every transport method to more clearly demonstrate the correlation between the distance and the target accuracy. This support the choice of using the distance to select the hyperparameters on this Domain Adaptation task. In addition, we also plot similar figures related to the class regularization η and quickly analyze the performance for an increasing number of samples for the gradient estimation.

Kullback-Leibler and classes regularization We show on Figures D.2 to D.8 the impact of the Kullback-Leibler and classes regularization.

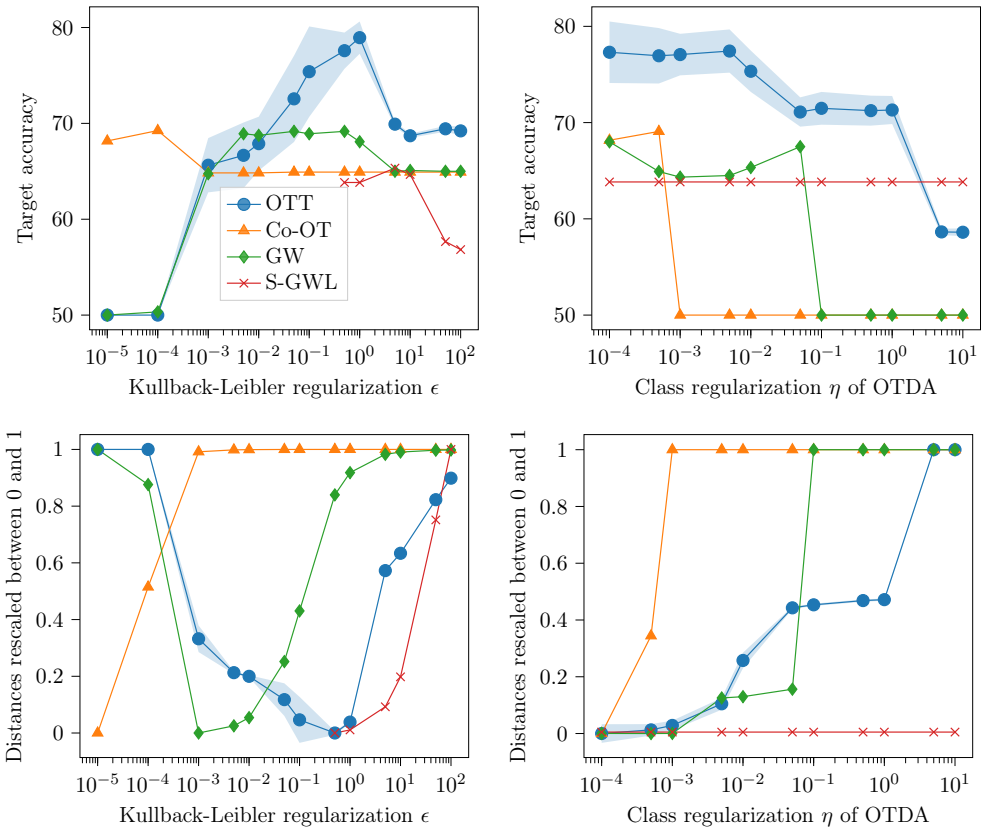


Figure D.2: *(**Top row**) Average target accuracy for an increasing Kullback-Leibler and classes regularization values. (**Bottom row**) Distances of the different methods for an increasing Kullback-Leibler and classes regularization values. The distances have been re-scaled between 0 and 1.*

Cited on pages [150,150]

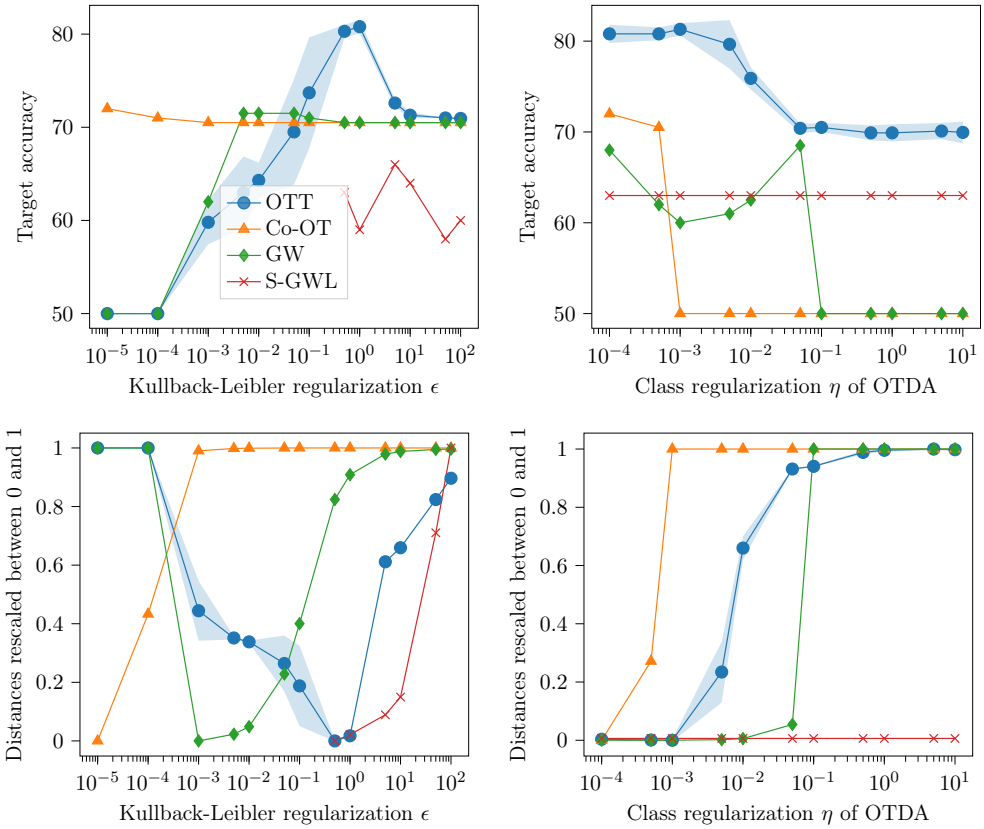


Figure D.3: *(**Top row**) Target accuracy for an increasing Kullback-Leibler and classes regularization values with the dataset composed of the two classes Thriller/Crime/Drama and Fantasy/Sci-Fi. (**Bottom row**) Distances of the different methods for an increasing Kullback-Leibler and classes regularization values with the dataset composed of the two classes Thriller/Crime/Drama and Fantasy/Sci-Fi. The distances have been re-scaled between 0 and 1.*

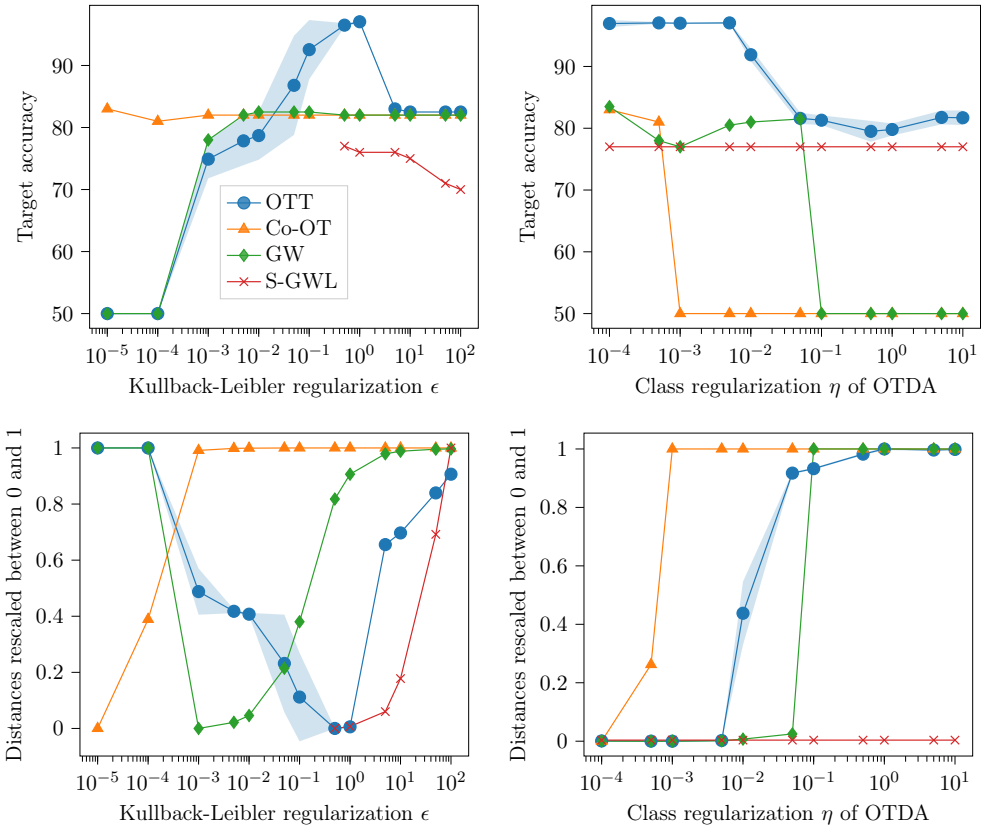


Figure D.4: *(**Top row**) Target accuracy for an increasing Kullback-Leibler and classes regularization values with the dataset composed of the two classes Children's/Animation and Fantasy/Sci-Fi. (**Bottom row**) Distances of the different methods for an increasing Kullback-Leibler and classes regularization values with the dataset composed of the two classes Children's/Animation and Fantasy/Sci-Fi. The distances have been re-scaled between 0 and 1.*

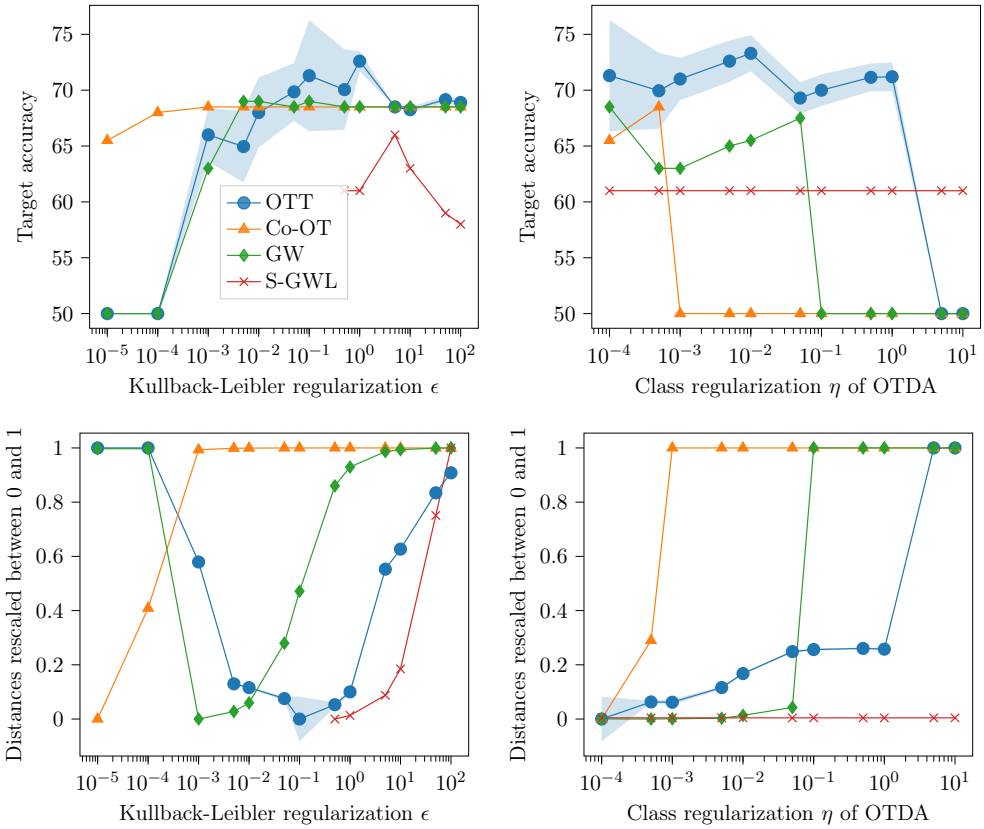


Figure D.5: *(**Top row**) Target accuracy for an increasing Kullback-Leibler and classes regularization values with the dataset composed of the two classes Thriller/Crime/Drama and War/Western. (**Bottom row**) Distances of the different methods for an increasing Kullback-Leibler and classes regularization values with the dataset composed of the two classes Thriller/Crime/Drama and War/Western. The distances have been re-scaled between 0 and 1.*

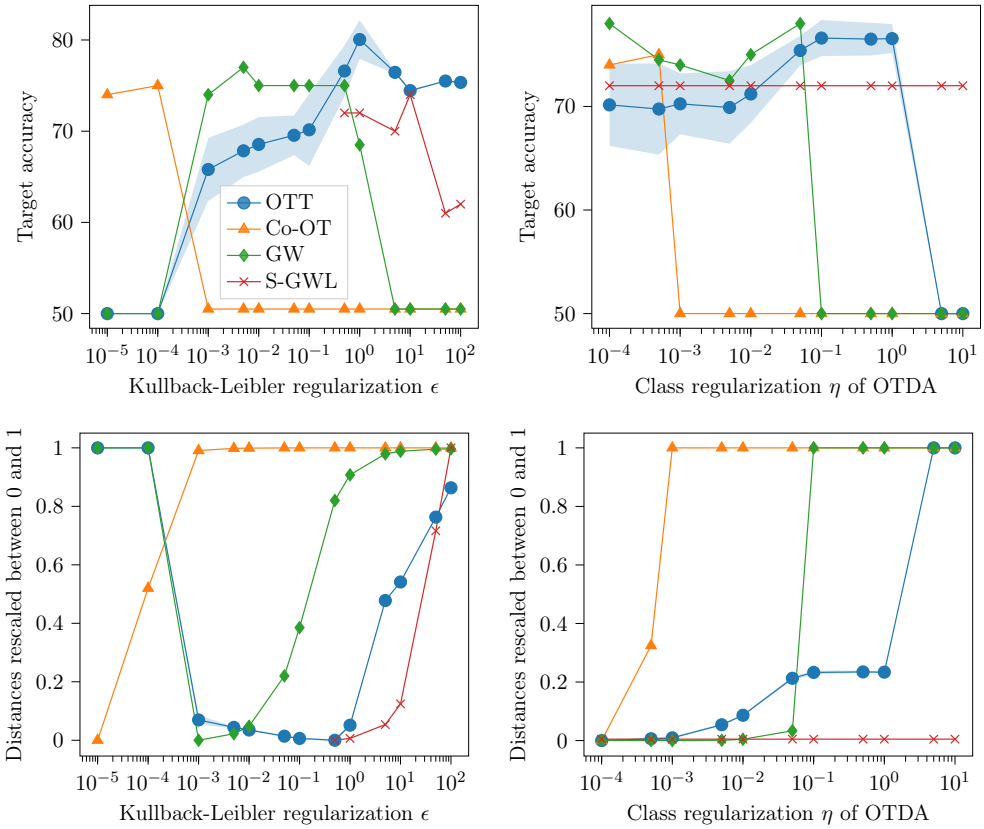


Figure D.6: *(**Top row**) Target accuracy for an increasing Kullback-Leibler and classes regularization values with the dataset composed of the two classes Fantasy/Sci-Fi and Children's/Animation. (**Bottom row**) Distances of the different methods for an increasing Kullback-Leibler and classes regularization values with the dataset composed of the two classes Fantasy/Sci-Fi and Children's/Animation. The distances have been re-scaled between 0 and 1.*

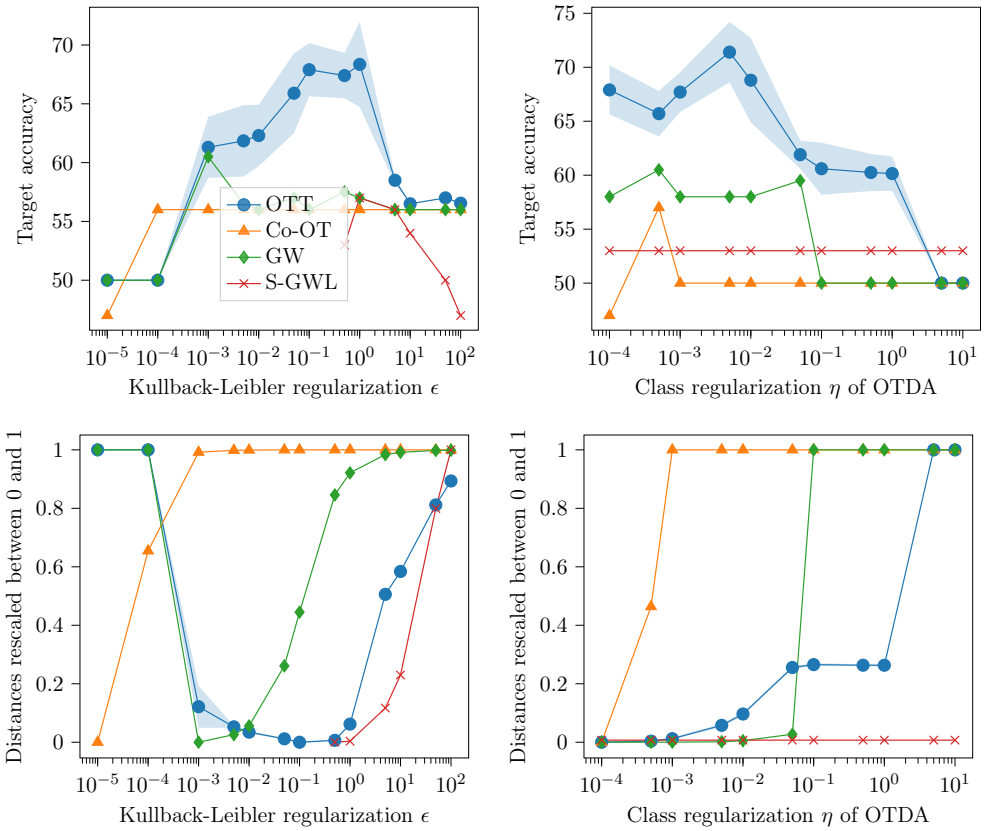


Figure D.7: ***(Top row)*** *Target accuracy for an increasing Kullback-Leibler and classes regularization values with the dataset composed of the two classes Fantasy/Sci-Fi and War/Western.* ***(Bottom row)*** *Distances of the different methods for an increasing Kullback-Leibler and classes regularization values with the dataset composed of the two classes Fantasy/Sci-Fi and War/Western. The distances have been re-scaled between 0 and 1.*

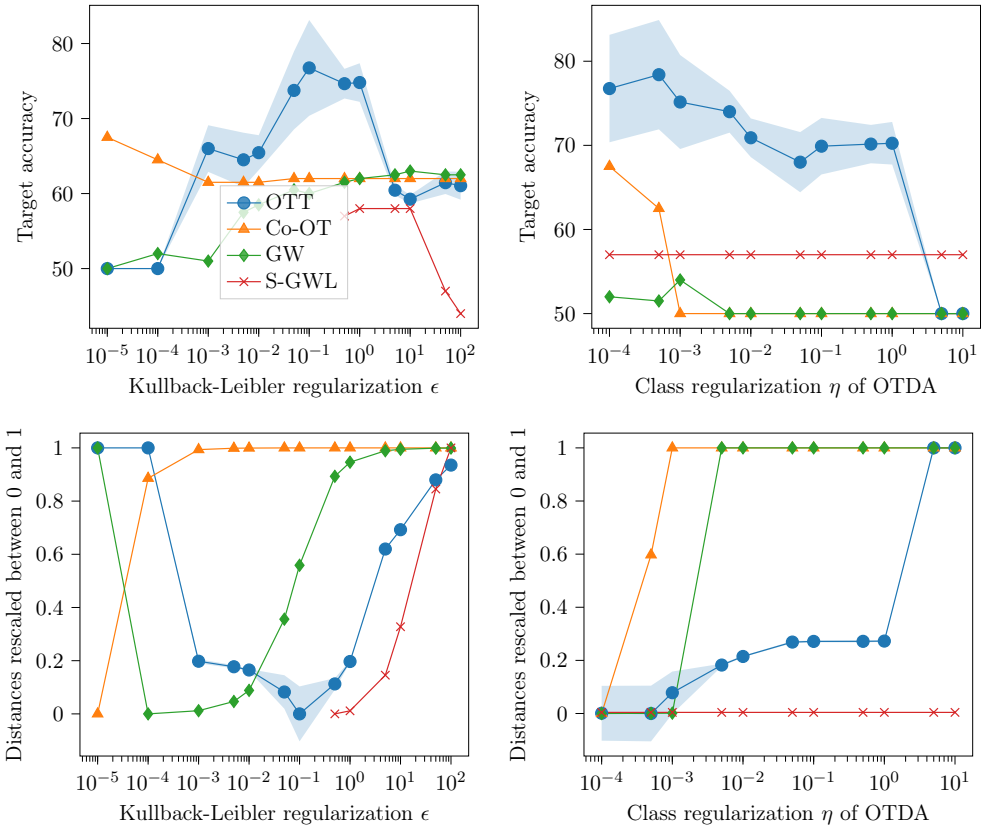


Figure D.8: *(**Top row**) Target accuracy for an increasing Kullback-Leibler and classes regularization values with the dataset composed of the two classes Children's/Animation and War/Western. (**Bottom row**) Distances of the different methods for an increasing Kullback-Leibler and classes regularization values with the dataset composed of the two classes Children's/Animation and War/Western. The distances have been re-scaled between 0 and 1.*

Cited on pages [150,150]

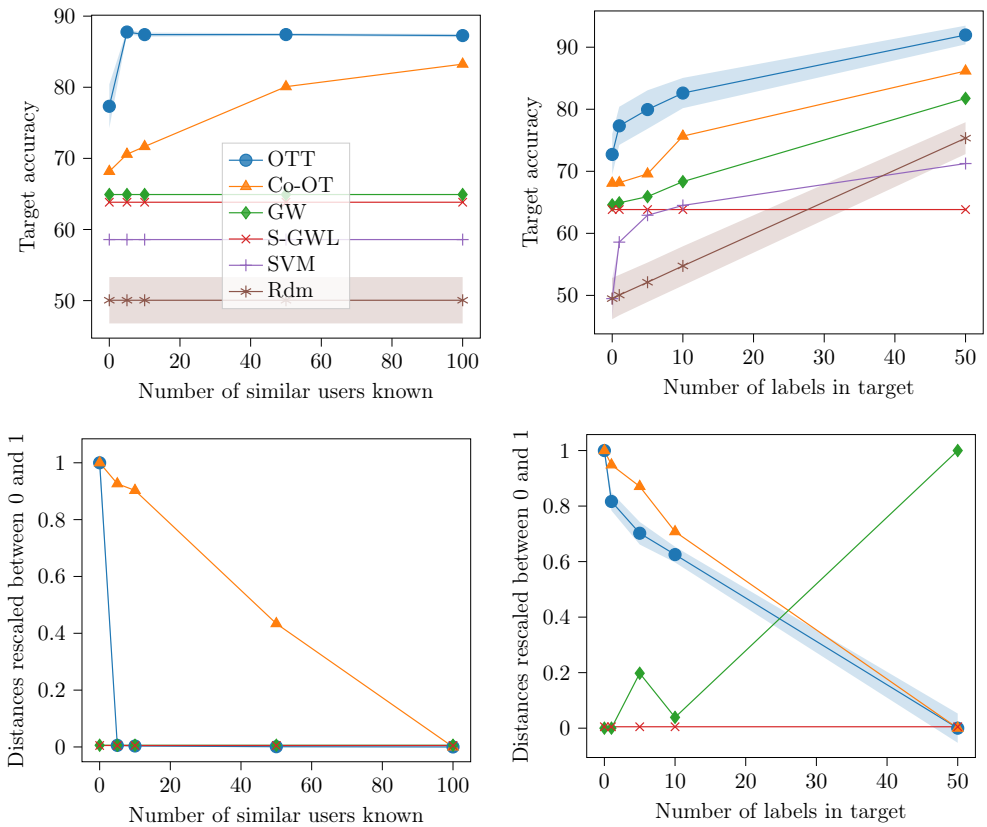


Figure D.9: *(**Top row**) Average target accuracy for an increasing number of users known and labels available in the target domain. (**Bottom row**) Distances of the different methods for an increasing number of users known and labels available in the target domain. The distances have been re-scaled between 0 and 1.*

Cited on page [158]

Increasing the supervision We show in Figures D.9 to D.15 the impact of an increase in terms of supervision with respect to the pairwise information that the same users rated old and new movies and the number of labels available for the target.

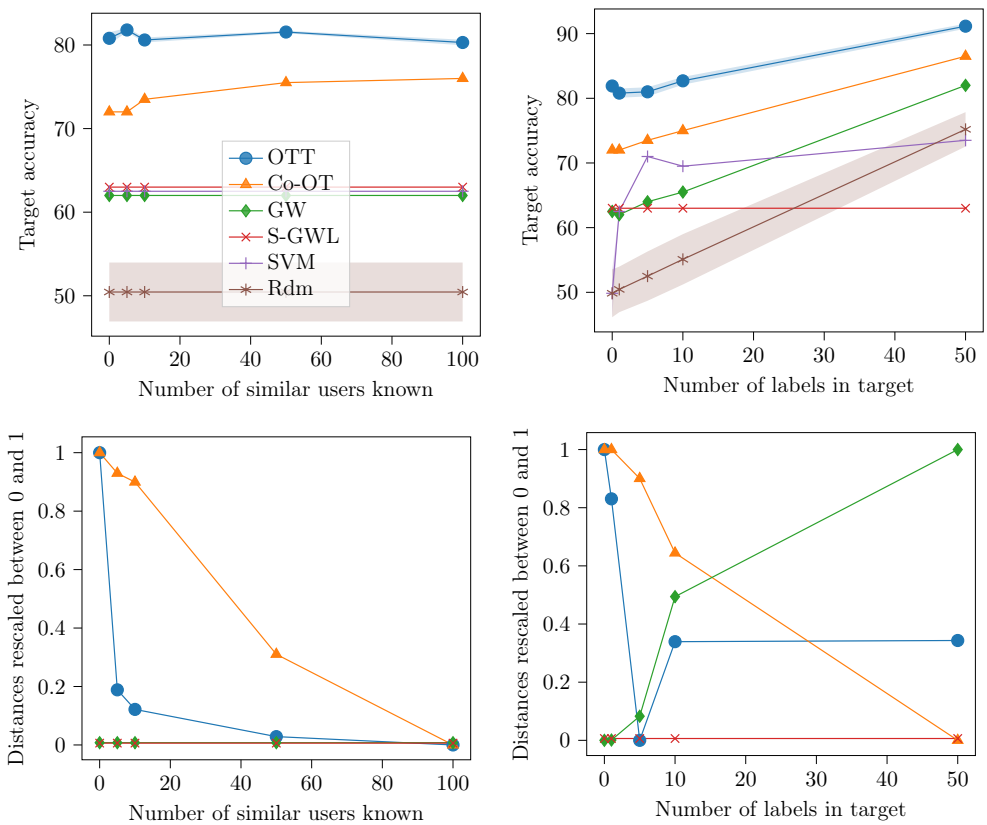


Figure D.10: *(**Top row**) Target accuracy for an increasing increasing number of users known and label available in the target domain with the dataset composed of the two classes Thriller/Crime/Drama and Fantasy/Sci-Fi. (**Bottom row**) Distances of the different methods for an increasing increasing number of users known and label available in the target domain with the dataset composed of the two classes Thriller/Crime/Drama and Fantasy/Sci-Fi. The distances have been re-scaled between 0 and 1.*

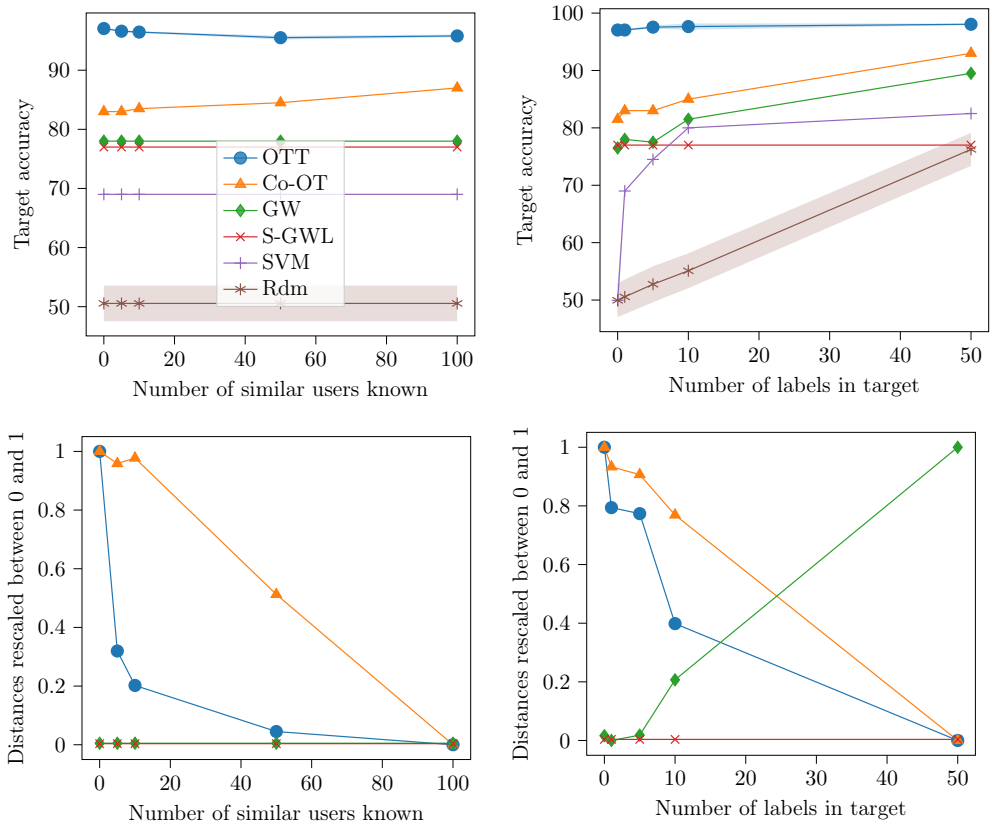


Figure D.11: *(**Top row**) Target accuracy for an increasing increasing number of users known and label available in the target domain with the dataset composed of the two classes Children's/Animation and Fantasy/Sci-Fi. (**Bottom row**) Distances of the different methods for an increasing increasing number of users known and label available in the target domain with the dataset composed of the two classes Children's/Animation and Fantasy/Sci-Fi. The distances have been re-scaled between 0 and 1.*

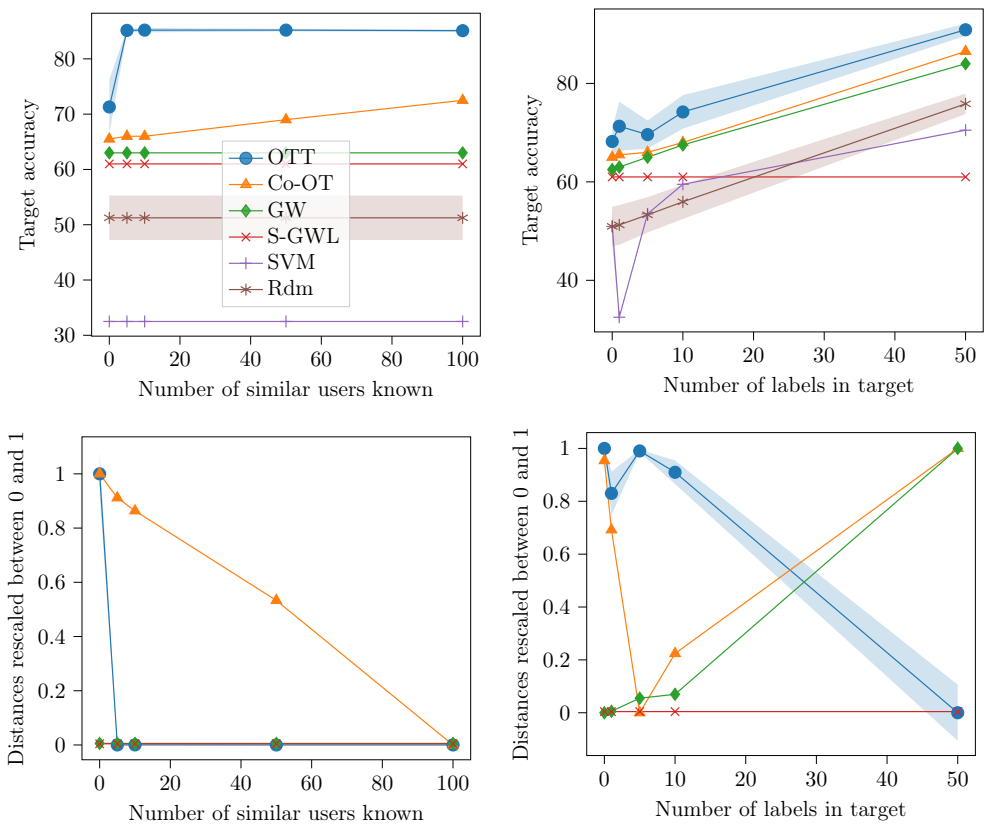


Figure D.12: *(**Top row**) Target accuracy for an increasing increasing number of users known and label available in the target domain with the dataset composed of the two classes Thriller/Crime/Drama and War/Western. (**Bottom row**) Distances of the different methods for an increasing increasing number of users known and label available in the target domain with the dataset composed of the two classes Thriller/Crime/Drama and War/Western. The distances have been re-scaled between 0 and 1.*

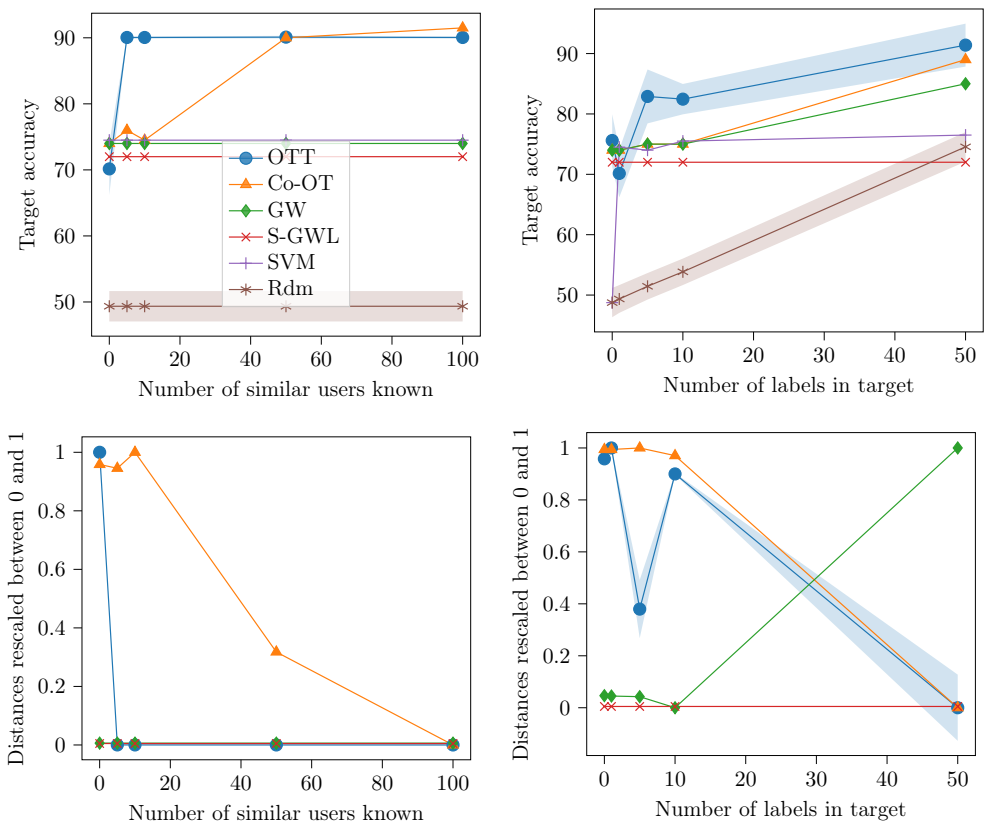


Figure D.13: *(**Top row**) Target accuracy for an increasing increasing number of users known and label available in the target domain with the dataset composed of the two classes Fantasy/Sci-Fi and Children's/Animation. (**Bottom row**) Distances of the different methods for an increasing increasing number of users known and label available in the target domain with the dataset composed of the two classes Fantasy/Sci-Fi and Children's/Animation. The distances have been re-scaled between 0 and 1.*

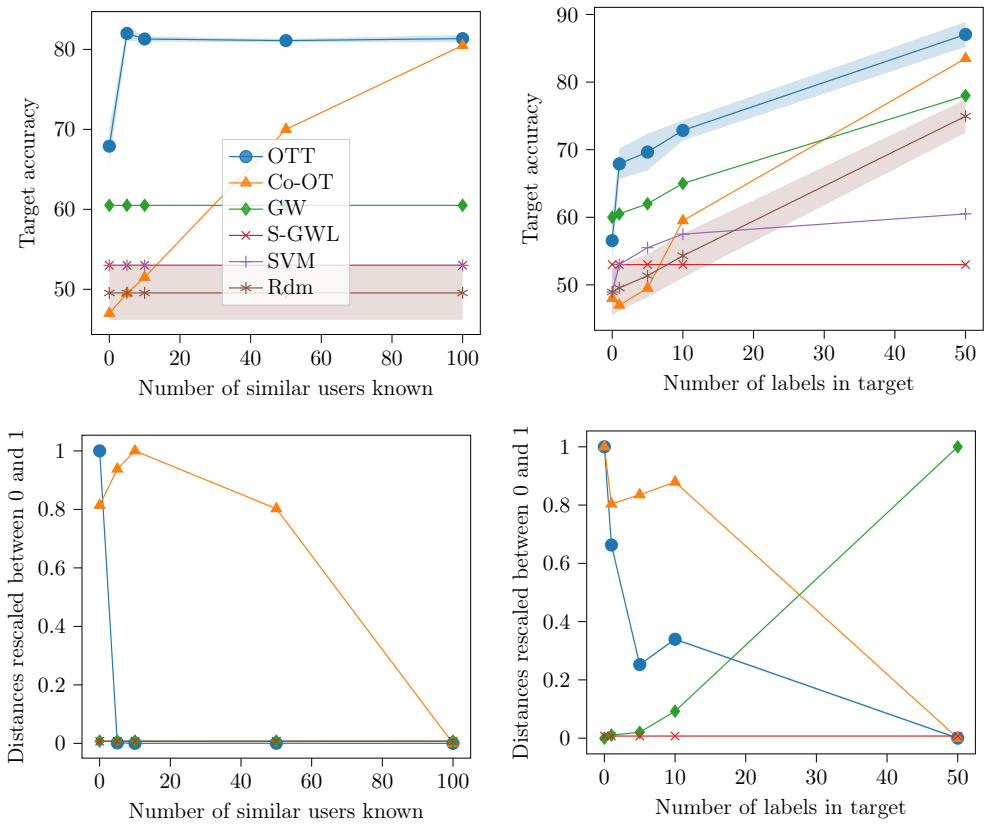


Figure D.14: *(**Top row**) Target accuracy for an increasing increasing number of users known and label available in the target domain with the dataset composed of the two classes Fantasy/Sci-Fi and War/Western. (**Bottom row**) Distances of the different methods for an increasing increasing number of users known and label available in the target domain with the dataset composed of the two classes Fantasy/Sci-Fi and War/Western. The distances have been re-scaled between 0 and 1.*

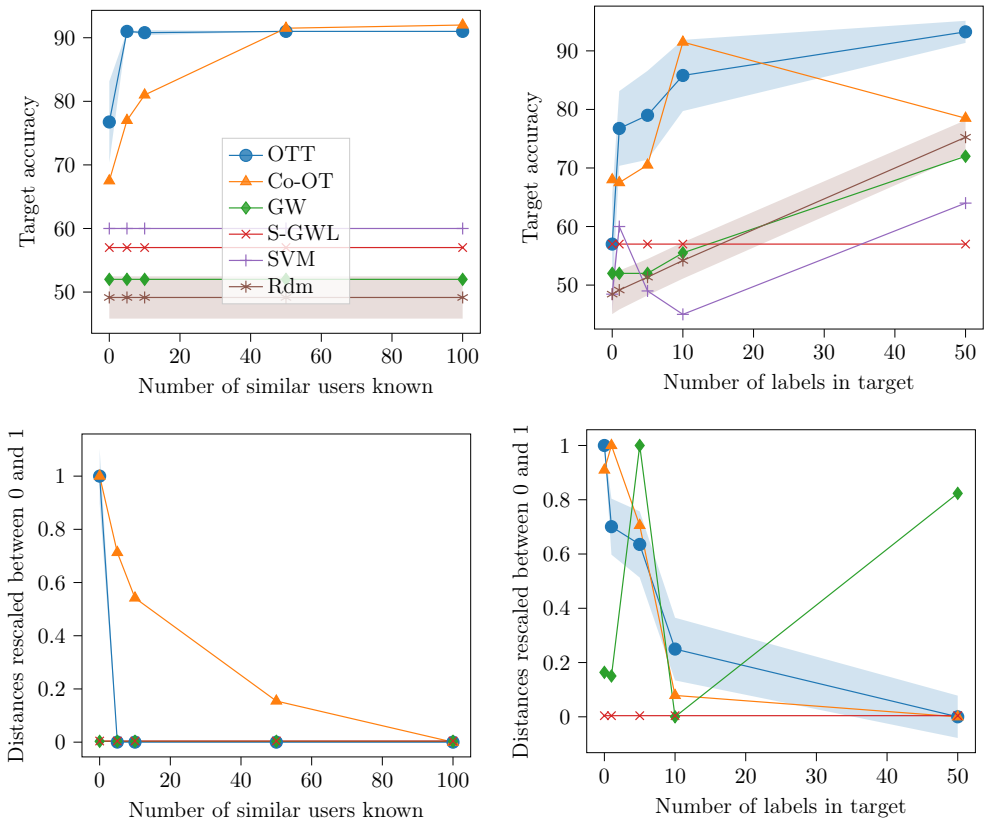


Figure D.15: *(**Top row**) Target accuracy for an increasing increasing number of users known and label available in the target domain with the dataset composed of the two classes C2hildren's/Animation and War/Western. (**Bottom row**) Distances of the different methods for an increasing increasing number of users known and label available in the target domain with the dataset composed of the two classes C2hildren's/Animation and War/Western. The distances have been re-scaled between 0 and 1.*

Cited on page [158]

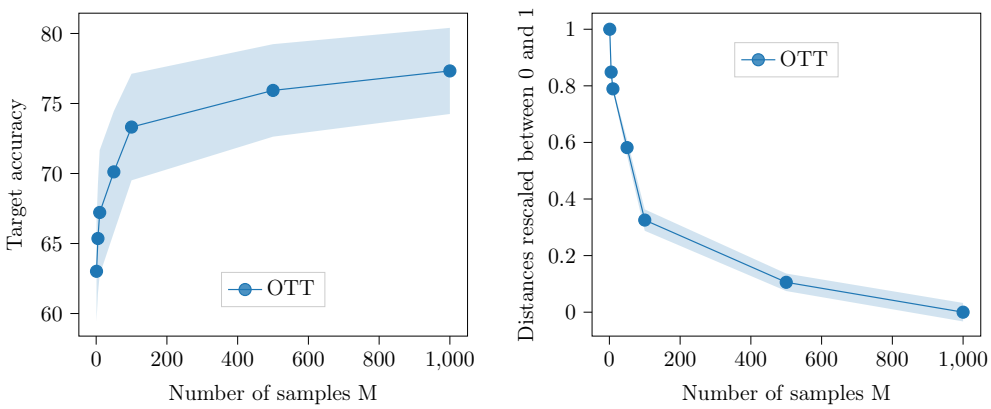


Figure D.16: *Average target accuracy and distance over all the dataset for an increasing number of samples for the estimation of the gradient.*

Cited on pages [148,165]

Increasing number of samples Figure D.16 shows that having a higher number of samples M for the estimation of the gradient improves the performances of OTT.

D.5.3 Comparison based clustering using OTT barycenter

Theoretical link between t-STE and OTT

In this section, we show that t-STE (Van Der Maaten and Weinberger, 2012) is just a OTT_{111} barycenter with fixed transport plan. While in the closed form presented in Theorem 6 we minimized directly the value of the tensor $\boldsymbol{X}$, we could also minimize another related variable, such as points in a vector space x which generate $\boldsymbol{X}(x)$. This is the principle of many triplet embedding methods, which are looking for points x in a vector space that respect as closely as possible the triplets provided in $\boldsymbol{X}^1$. Theorem 11 shows that a widely used method for triplet embedding, t-STE (Van Der Maaten and Weinberger, 2012), is a particular case of OTT barycenter.

Theorem 11. *We suppose that $\mathcal{T}$ is a list of triplets which can also be represented with a cubic 3-order tensor $\boldsymbol{X}^1$ of size (I_1, I_1, I_1) with, at the position i_1, i_2, i_3 the number of occurrences of the triplet (i_1, i_2, i_3) in $\mathcal{T}$. Let $x = (x_i)_{i\in[\![1,I_1]\!]}$ be I_1 points in a vector space $\mathbb{R}^q$. We can then set the tensor $\boldsymbol{X}$ to the t-STE or the STE formula as given in (Van Der Maaten and Weinberger, 2012), for STE: $\boldsymbol{X}_{i_1,i_2,i_3} = \frac{exp(-\|x_{i_1}-x_{i_2}\|^2)}{exp(-\|x_{i_1}-x_{i_2}\|^2)+exp(-\|x_{i_1}-x_{i_3}\|^2)}$. If $\mathcal{L}$ is the cross-entropy and f is the constant function (all the 3 transport plans are similar), then STE is a particular case of OTT with the identity matrix $\mathbf{Id}$ (divided by I_1) of size I_1 as the transport plan,*

$$\max_{x\in\mathbb{R}^{I_1\times q}} \sum_{(i_1,i_2,i_3)\in\mathcal{T}} \log(\boldsymbol{X}_{i_1i_2i_3}(x)) = I_1^3 \min_{x\in\mathbb{R}^{I_1\times q}} \sum_{b=1}^{1} \mathcal{E}\left(\boldsymbol{X}(x), \boldsymbol{X}^1, \frac{\mathbf{Id}}{I_1}\right). \tag{D.53}$$

Proof. We start from the STE formulation and reformulate the problem,

$$\max_{x\in\mathbb{R}^{I_1\times q}} \sum_{(i_1,i_2,i_3)\in\mathcal{T}} \log(\boldsymbol{X}_{i_1,i_2,i_3}(x)) \tag{D.54}$$

$$= \max_{x\in\mathbb{R}^{I_1\times q}} \sum_{i_1,i_2,i_3=1}^{I_1,I_1,I_1} \boldsymbol{X}^1_{i_1,i_2,i_3} \log(\boldsymbol{X}_{i_1,i_2,i_3}(x)) \tag{D.55}$$

$$= \min_{x\in\mathbb{R}^{I_1\times q}} \sum_{i_1,i_2,i_3=1}^{I_1,I_1,I_1} -\boldsymbol{X}^1_{i_1,i_2,i_3} \log(\boldsymbol{X}_{i_1,i_2,i_3}(x)) \tag{D.56}$$

$$= \min_{x\in\mathbb{R}^{I_1\times q}} \sum_{i_1,i_2,i_3=1}^{I_1,I_1,I_1} \sum_{k_1,k_2,k_3=1}^{I_1,I_1,I_1} -\boldsymbol{X}^1_{i_1,i_2,i_3} \log(\boldsymbol{X}_{k_1,k_2,k_3}(x))\mathbf{Id}_{i_1,k_1}\mathbf{Id}_{i_2,k_2}\mathbf{Id}_{i_3,k_3} \tag{D.57}$$

$$= \min_{x\in\mathbb{R}^{I_1\times q}} \sum_{i_1,i_2,i_3=1}^{I_1,I_1,I_1} \sum_{k_1,k_2,k_3=1}^{I_1,I_1,I_1} \mathcal{L}\left(\boldsymbol{X}^1_{i_1,i_2,i_3}, \boldsymbol{X}_{k_1,k_2,k_3}(x)\right)\mathbf{Id}_{i_1,k_1}\mathbf{Id}_{i_2,k_2}\mathbf{Id}_{i_3,k_3} \tag{D.58}$$

$$= I_1^3 \min_{x\in\mathbb{R}^{I_1\times q}} \sum_{i_1,i_2,i_3=1}^{I_1,I_1,I_1} \sum_{k_1,k_2,k_3=1}^{I_1,I_1,I_1} \mathcal{L}\left(\boldsymbol{X}^1_{i_1,i_2,i_3}, \boldsymbol{X}_{k_1,k_2,k_3}(x)\right)\frac{\mathbf{Id}_{i_1,k_1}}{I_1}\frac{\mathbf{Id}_{i_2,k_2}}{I_1}\frac{\mathbf{Id}_{i_3,k_3}}{I_1}. \tag{D.59}$$

Note that, all the permutations are equivalent to the identity as x_i and x_j can be exchanged. In addition, the identity matrix is not necessarily the optimal value, thus the OTT barycenter might lead to a better optimal, but loses the pairwise connection that can be useful for interpretation. □

Interestingly, this new interpretation of t-STE in the light of Theorem 11 gives a good theoretical justification of the choice of the log function in t-STE which is nothing more than a cross entropy between 3D-tensor. One specificity of OTT barycenter, compared to t-STE, is that the size of the barycenter $\boldsymbol{X}$ is not necessarily the size of $\boldsymbol{X}^1$. Thus, it is notably possible to use a small size for $\boldsymbol{X}$ which will aggregate similar points from $\boldsymbol{X}^1$. This idea has been used advantageously in the experiment to apply a direct clustering of a triplet dataset.

Hyperparameters used

In this section, we detail the hyperparameters used in the experiment.

OTT

- Loss type: squared euclidean distance
- Number of samples M: 100
- Number of iterations S: 500
- Kullback-Leibler regularization: 0.1

AddS3

- Number of iterations of the k-means: 300

t-STE

- Degrees of freedom in student T kernel: 0
- Number of iterations of k-means: 300
- Maximum number of iterations: 1000
- L2 regularization constant: 0

Detailed tables of the experiment

We display the ARI for comparison based clustering in Table D.2 which is similar to the one provided in Chapter 5, without averaging the different classes. This table shows similar behaviour, OTT is very often better than AddS3$_s$ while being comparable to t-STE$_s$ on most datasets.

Additionally, we present several experiments in the balanced case in Table D.3 where the proportion of classes are similar. In this case, OTT is slightly worse than the two other baselines

Table D.2: *ARI for unbalanced comparison-based clustering tasks on MNIST dataset. Each line corresponds to the average over 10 runs.*

Cited on page [166]

nb. examples per class|classes	AddS3	AddS3$_s$	t-STE	t-STE$_s$	OTT
200,20,20|0,1,2	0.35±0.02	0.89±0.19	0.36±0.02	**0.97**±0.02	0.96±0.02
200,20,20|1,3,4	0.35±0.02	0.92±0.03	0.34±0.01	0.94±0.03	**0.94**±0.03
200,20,20|1,9,4	0.35±0.02	0.92±0.03	0.34±0.01	0.94±0.03	**0.94**±0.03
200,20,20|2,9,8	0.32±0.03	0.8±0.05	0.72±0.21	**0.82**±0.06	0.8±0.06
200,20,20|3,4,0	0.72±0.32	0.52±0.25	**0.94**±0.02	0.88±0.18	0.87±0.06
200,20,20|3,4,9	0.38±0.06	0.83±0.18	0.39±0.16	**0.9**±0.05	0.9±0.04
200,20,20|5,7,0	0.57±0.34	0.76±0.35	0.93±0.04	**0.94**±0.04	0.93±0.05
200,20,20|6,4,8	0.35±0.03	0.89±0.2	0.33±0.01	0.96±0.02	**0.97**±0.02
200,20,20|6,8,5	0.36±0.03	0.81±0.24	0.4±0.18	0.93±0.04	**0.94**±0.03
200,20,20|7,1,9	0.53±0.35	0.69±0.22	**0.86**±0.05	0.8±0.06	0.8±0.05
30,3,1|0,1,2	0.32±0.08	0.8±0.33	0.35±0.17	0.93±0.12	**0.93**±0.12
30,3,1|1,3,4	0.28±0.08	0.85±0.33	0.37±0.21	**1.0**±0.01	0.96±0.09
30,3,1|1,9,4	0.27±0.1	0.96±0.09	0.3±0.03	**0.99**±0.01	0.95±0.09
30,3,1|2,9,8	0.29±0.08	0.87±0.15	0.35±0.17	**0.9**±0.14	0.84±0.15
30,3,1|3,4,0	0.2±0.1	0.61±0.24	**0.9**±0.12	0.85±0.15	0.84±0.21
30,3,1|3,4,9	0.29±0.06	0.84±0.21	0.33±0.17	**0.87**±0.15	0.84±0.21
30,3,1|5,7,0	0.25±0.06	0.7±0.01	0.85±0.22	0.88±0.15	**0.9**±0.15
30,3,1|6,4,8	0.31±0.03	0.94±0.12	0.29±0.03	**0.96**±0.09	0.9±0.13
30,3,1|6,8,5	0.31±0.05	0.9±0.14	0.29±0.03	**0.97**±0.1	0.9±0.14
30,3,1|7,1,9	0.27±0.23	0.76±0.12	0.84±0.23	0.81±0.15	**0.84**±0.16
30,3,3|0,1,2	0.4±0.11	0.85±0.19	0.38±0.11	**0.89**±0.12	0.83±0.19
30,3,3|1,3,4	0.37±0.05	0.85±0.24	0.35±0.03	**0.99**±0.01	0.93±0.18
30,3,3|1,9,4	0.37±0.05	0.85±0.25	0.36±0.03	**0.99**±0.01	0.92±0.18
30,3,3|2,9,8	0.34±0.06	0.8±0.24	0.38±0.17	**0.87**±0.12	0.81±0.24
30,3,3|3,4,0	0.47±0.38	0.5±0.25	0.94±0.07	**0.95**±0.09	0.8±0.17
30,3,3|3,4,9	0.38±0.1	0.83±0.21	0.37±0.17	**0.89**±0.12	0.85±0.19
30,3,3|5,7,0	0.27±0.21	0.73±0.29	0.91±0.09	**0.95**±0.09	0.89±0.19
30,3,3|6,4,8	0.36±0.04	0.89±0.19	0.36±0.03	**0.96**±0.07	0.92±0.18
30,3,3|6,8,5	0.36±0.04	0.89±0.19	0.36±0.03	**0.98**±0.07	0.93±0.18
30,3,3|7,1,9	0.38±0.27	0.59±0.23	**0.81**±0.19	0.77±0.17	0.8±0.2
300,30,10|0,1,2	0.3±0.02	0.96±0.02	0.3±0.01	0.96±0.03	**0.96**±0.02
300,30,10|1,3,4	0.29±0.01	0.93±0.02	0.28±0.0	0.89±0.1	**0.93**±0.03
300,30,10|1,9,4	0.29±0.01	**0.94**±0.03	0.28±0.01	0.93±0.03	0.93±0.04
300,30,10|2,9,8	0.28±0.03	**0.8**±0.05	0.42±0.26	0.77±0.04	0.73±0.06
300,30,10|3,4,0	0.25±0.04	0.59±0.23	**0.92**±0.03	0.91±0.06	0.85±0.06
300,30,10|3,4,9	0.3±0.04	0.85±0.1	0.4±0.22	0.9±0.04	**0.91**±0.03
300,30,10|5,7,0	0.3±0.19	0.68±0.02	0.9±0.04	**0.91**±0.04	0.88±0.06
300,30,10|6,4,8	0.3±0.02	0.93±0.09	0.28±0.01	0.92±0.09	**0.96**±0.03
300,30,10|6,8,5	0.3±0.02	0.9±0.1	0.34±0.19	**0.94**±0.03	0.93±0.03
300,30,10|7,1,9	0.25±0.03	0.69±0.04	0.65±0.27	0.76±0.05	**0.78**±0.05
AVG	0.34±0.09	0.81±0.16	0.51±0.1	**0.91**±0.08	0.89±0.1

Table D.3: *ARI for balanced comparison-based clustering tasks on MNIST dataset. Each line corresponds to the average over 10 different combinations of classes, each run 10 times.*
Cited on page [166]

nb. examples per class	AddS3	AddS3$_s$	t-STE	t-STE$_s$	OTT
10,10,10	0.9±0.08	0.9±0.1	0.88±0.13	**0.93**±0.11	0.9±0.11
20,20,20	0.88±0.06	0.87±0.07	0.86±0.11	**0.91**±0.08	0.86±0.08
30,30,30	0.91±0.05	0.9±0.05	0.87±0.1	**0.92**±0.07	0.91±0.07
40,40,40	**0.92**±0.06	0.91±0.05	0.86±0.1	**0.92**±0.06	0.9±0.06
50,50,50	**0.92**±0.06	**0.92**±0.06	0.85±0.11	**0.92**±0.06	0.9±0.06
60,60,60	**0.92**±0.05	0.91±0.04	0.86±0.09	**0.92**±0.05	0.9±0.05
70,70,70	**0.92**±0.05	**0.92**±0.05	0.85±0.07	**0.92**±0.05	0.88±0.05
80,80,80	**0.92**±0.06	0.91±0.03	0.85±0.1	**0.92**±0.06	0.86±0.06
90,90,90	**0.92**±0.11	**0.92**±0.11	0.87±0.09	**0.92**±0.11	0.8±0.11
100,100,100	**0.92**±0.13	**0.92**±0.13	0.85±0.09	**0.92**±0.13	0.73±0.13
AVG	0.91±0.04	0.91±0.05	0.86±0.1	**0.92**±0.05	0.86±0.08

while still being competitive. This is not surprising since OTT, contrary to AddS3 and t-STE, was not specifically designed to handle triplet comparisons. Instead, it is a general purpose Optimal Transport formulation between tensors of potentially high order that can be used to solve multiple kind of tasks.

Appendix E

Unfinished research related to the contributions of this thesis

This last appendix covers two interesting subjects related to the thesis but which were not completely explored. In addition, it present the software "Optimal Slide Transition" that can be used for a presentation.

E.1 Pointwise Wasserstein and Sliced Wasserstein

While Pointwise Gromov Wasserstein was originally designed for GW, we can still adapt it for the original OT problem. Contrary to the GW problem, and similarly to Sliced Wasserstein, Pointwise Wasserstein defines a new problem.

First, let us explain the Sliced Wasserstein (SW) distance (Pitié et al., 2007; Rabin and Peyré, 2011). This is a distance that takes the average over all the Wasserstein distances computed after projection on 1D lines. In theory, all the lines that cross 0 should be used to have the complete distance. In practice, only some of them are sampled. With the usual notations and Ω being the $(D-1)$-dimensional unit sphere, the Sliced Wasserstein distance is defined as,

$$\mathcal{SW}(\mu,\nu) = \int_{\Omega} \mathcal{W}(\langle \bullet, \boldsymbol{\theta} \rangle \#\mu, \langle \bullet, \boldsymbol{\theta} \rangle \#\nu) d\boldsymbol{\theta}. \tag{E.1}$$

Here $\langle \bullet, \boldsymbol{\theta} \rangle \#\mu$ means that the distribution μ is pushed forward by the function $\boldsymbol{x} \to \langle \boldsymbol{x}, \boldsymbol{\theta} \rangle$. The SW distance has some applications in GAN (Nadjahi et al., 2019; Nguyen et al., 2020; Deshpande et al., 2018). Some extensions have been proposed in the literature, e.g., Generalized Sliced Wasserstein (Kolouri et al., 2019) considers the scalar product as a particular case of function used in a Generalized Radeon transform (Kuchment, 2006).

Instead of comparing the points by their distance to a line, we propose to compare them by their distance to a point. With now Ω any subset of $\mathbb{R}^D$, we can define the Pointwise Wasserstein

as,

$$\mathcal{PW}(\mu,\nu) = \int_{\Omega} \mathcal{W}(\|\bullet - \boldsymbol{\theta}\| \#\mu, \|\bullet - \boldsymbol{\theta}\| \#\nu) d\boldsymbol{\theta}. \tag{E.2}$$

The most natural choice for Ω will be the support of both μ and ν. In this case, with two discrete distributions, the points are defined only by their respective distances, which can be useful if the points do not lie in a known vector space. If we set Ω to the $(D-1)$-dimensional sphere of radius r, as r tends to ∞: $\frac{\mathcal{PW}}{r} \to \mathcal{SW}$. Indeed, for a given $x \in \mathbb{R}^D$, the other directions, orthogonal to the direction given by $\boldsymbol{\theta}$, become negligible: $\frac{\|\boldsymbol{x}-\boldsymbol{\theta}\|}{r} \to \langle \boldsymbol{x}, \frac{\boldsymbol{\theta}}{r} \rangle$. In other words, the Pointwise Wasserstein distance generalizes the Sliced Wasserstein distance.

Note that the Pointwise Wasserstein formulation is not covered by the Generalized Sliced Wasserstein (Kolouri et al., 2019) as $\|\bullet - \boldsymbol{\theta}\|$ is not homogeneous of degree one in $\boldsymbol{\theta}$.

E.2 Optimal Slide Transition

This section presents Optimal Slide Transition (OST), which is a software that allows to have transitions between your PDF pages. A specific type of transitions is used: it applies the optimal transport mapping between the two consecutive pages[1]. This work was done in collaboration with Rémi EMONET who implemented the *html* code to launch the presentation. Note that this code has been tested on a few computers but might not work correctly on some others as it uses some bash commands.

E.2.1 Algorithm details

First of all, OST transforms each PDF into a PNG image. Thus, each slide can be represented as a distribution with Dirac at each non-white pixel. All the Dirac have uniform weights. Then OST computes each transport plan between slides separately. To apply the OT between two images, we need to apply a hierarchical Optimal Transport (Mérigot, 2011) as there is too many Diracs in each image (approximately 1 million per image).

The idea is to apply a K-means clustering (with the hyperparameter K chosen by the user) in the two images separately. Then we apply the OT between the centroid of each cluster with a mass proportional to the number of points. For each cluster matched together, we don't know the detailed transport plan between the two clusters. So we simply apply again a hierarchical OT between all the points of the two clusters. If the number of Diracs in the distributions are smaller than K, then a simple OT algorithm is applied. We can easily see on an example that the transport is not truly optimal, the Voronoi cells generated by the K-means clustering being matched in blocks. Every time the OT algorithm is used, the Euclidean distance between the points are used plus (with a small weight) the euclidean distance in RGB color.

In addition to the hierarchical OT methods, an adapted version of Sliced Wasserstein (see Section E.1 for details about Sliced Wasserstein) method was tested but did not provide a good-looking transport.

[1] You can have a look at `https://hv0nnus.github.io/TransitionPDF/pres.html` for an example.

E.2.2 Practical details on how to use OST

You have to clone the github repository: `https://github.com/Hv0nnus/TransitionPDF`. Then you have to replace the PDF file *Presentation_OT.pdf* with your own presentation and run the following command: python main.py - -qualityx 25 - -qualityy 25 - -qualityx_pres 50 - -qualityy_pres 50 - -K 100. Wait the end, then launch *pres.html* and use the arrow of the keyboard for your presentation. The hyperparameters can be tuned as follows:

- Use a higher value for *qualityx/y* (250) to increase the quality of transition between slides. This values will increase the time required to compute the transport plan and the necessary memory to store the transport plan.
- Use a higher value for *qualityx_pres* and *qualityy_pres* (500) to increase the quality of the slide itself (it will be a PNG). Both of those values are more or less the number of pixels in the x and y axes.
- K is a hyperparameter of the method used for OT on such big images. The bigger it is, the closer it gets to the real transport.

E.3 Gromov Wasserstein and Wasserstein GAN

This section explains why we can see the Gromov Wasserstein problem as a linear Wasserstein Generative Adversarial Network (GAN).

Let first present the Wasserstein GAN (WGAN) (Arjovsky et al., 2017). Given a theoretical distribution $\mu \in \mathcal{P}(\mathcal{X})$, one would like to be able to sample new elements from μ. The idea is to learn a generator function $g : \mathcal{Y} \to \mathcal{X}$, that will try to simulate μ with point sampled from a simple distribution (often Gaussian) $\nu \in \mathcal{P}(\mathcal{Y})$. To compare $g\#\nu$ and μ, one can either use a KL divergence (Goodfellow et al., 2014), or a Wasserstein distance (Arjovsky et al., 2017). Instead of using the classical Wasserstein distance, the dual formulation of the OT is used. The general Kantorovich dual (Villani, 2008) of the classical OT problem reads,

$$\min_{|f(\boldsymbol{x})-h(\boldsymbol{z})|\leq c(\boldsymbol{x},\boldsymbol{z})} \mathop{\mathbb{E}}_{\boldsymbol{x}\sim\mu}[f(\boldsymbol{x})] - \mathop{\mathbb{E}}_{\boldsymbol{z}\sim\nu'}[h(\boldsymbol{z})]\,, \tag{E.3}$$

for two μ and ν'-integrable functions f and h. As a distance is used for the cost function: $f = h$ (Villani, 2008, Particular Case 5.4). Thus, only one 1-Lipschitz function f, called discriminator, should be computed. Denoting $Lip_c = \{f|\frac{|f(\boldsymbol{x})-f(\boldsymbol{z})|}{c(\boldsymbol{x},\boldsymbol{z})} \leq 1\}$ the set of 1-Lipschitz function, the Kantorovich dual formulation reads,

$$\min_{f\in Lip_c} \mathop{\mathbb{E}}_{\boldsymbol{x}\sim\mu}[f(\boldsymbol{x})] - \mathop{\mathbb{E}}_{\boldsymbol{z}\sim\nu'}[f(\boldsymbol{z})]\,. \tag{E.4}$$

As in practice it is complex to select all the existing functions, both the generator g and the discriminator f are parametrized by Neural Networks. To introduce the GAN formulation,

the distribution ν' should be seen as the distribution $g\#\nu$, and equivalently "$\boldsymbol{z} = g(\boldsymbol{y})$". The objective function of a Wasserstein GAN reads,

$$\min_{g} \max_{f \in Lip_c} \mathop{\mathbb{E}}_{\boldsymbol{x}\sim\mu} [f(\boldsymbol{x})] - \mathop{\mathbb{E}}_{\boldsymbol{y}\sim\nu} [f(g(\boldsymbol{y}))] . \tag{E.5}$$

In practice, the entire distribution μ is not available but only the associated empirical distribution $\hat{\mu}$. To solve this problem by alternate gradient descent, ν is also sampled a few times at each iteration.

The link between GW and WGAN is essentially based on the Lemma 4.1 and Lemma 4.3 in (Alvarez-Melis et al., 2019), which show the relation between Gromov Wasserstein and their invariant OT formulation. A related Theorem, 4.2.1 available in (Vayer, 2020, Theorem 4.2.1) proposes a similar theorem. We provide in Theorem 12 the link between the GW and GAN in the discrete case, which is useful in practice. A similar theorem can be obtained in the continuous case.

Theorem 12. *We first define $\mathcal{X} \subseteq \mathbb{R}^D$ and $\mathcal{Y} \subseteq \mathbb{R}^{D'}$. Let $\mu = \sum_{i=1}^{I} \boldsymbol{a}_i \boldsymbol{x}_i$ be a discrete distribution with $(\boldsymbol{x}_i)_{i\in[\![1,I]\!]} \in \mathcal{X}^I$ and Let $\nu = \sum_{k=1}^{K} \boldsymbol{b}_k \boldsymbol{y}_k$ be a discrete distribution with $(\boldsymbol{y}_i)_{k\in[\![1,K]\!]} \in \mathcal{Y}^K$. We additionally suppose that ν is whitened, $\int \boldsymbol{y}^\top \boldsymbol{y} d\nu(\boldsymbol{y}) = \mathbb{I}_{D'}$. We denote $\boldsymbol{X} \in \mathbb{R}^{D\times I}$ and $\boldsymbol{Y} \in \mathbb{R}^{D'\times K}$ the matrices defined as $\forall i \in [\![1, I]\!]$ $\boldsymbol{X}_{\bullet i} = \boldsymbol{x}_i$ and $\forall k \in [\![1, K]\!]$ $\boldsymbol{X}_{\bullet k} = \boldsymbol{x}_k$. Let define the cost functions and the loss as, $\mathcal{C}^{\mathcal{X}}(\boldsymbol{x}_i, \boldsymbol{x}_j) = \boldsymbol{x}_i^\top \boldsymbol{x}_j$, $\mathcal{C}^{\mathcal{Y}}(\boldsymbol{y}_k, \boldsymbol{y}_l) = \boldsymbol{y}_k^\top \boldsymbol{y}_l$ and $\mathcal{L}(\boldsymbol{x}, \boldsymbol{y}) = |\boldsymbol{x} - \boldsymbol{y}|^2$. We set f and h the two dual functions associated with the OT problem defined by μ, ν and the cost matrix $\boldsymbol{C}_{ik} = \|\boldsymbol{x}_i - g(\boldsymbol{y}_k)\|_2^2$. If we confuse a linear function g and its associated matrix $\boldsymbol{G}$ and denote $\mathcal{F} = \{\boldsymbol{G} \in \mathbb{R}^{D\times D'} | \, \|\boldsymbol{G}\|_F = 1\}$ the set of linear functions, we have:*

$$\min_{\boldsymbol{T}\in\Pi_{\mu\nu}} \sum_{i,j,k,l=1}^{I,I,K,K} \mathcal{L}(\mathcal{C}^{\mathcal{X}}(\boldsymbol{x}_i, \boldsymbol{x}_j) - \mathcal{C}^{\mathcal{Y}}(\boldsymbol{y}_k, \boldsymbol{y}_l))\boldsymbol{T}_{ik}\boldsymbol{T}_{jl} \tag{E.6}$$

$$= \min_{g\in\mathcal{F}} \max_{f(\boldsymbol{x}_i)-h(g(\boldsymbol{y}_j))\leq\|\boldsymbol{x}_i-g(\boldsymbol{y}_j)\|_2^2} C_1 \left[\mathop{\mathbb{E}}_{\boldsymbol{x}\sim\mu} [f(\boldsymbol{x})] - \mathop{\mathbb{E}}_{\boldsymbol{y}\sim\nu} [h(g(\boldsymbol{y}))] \right] + C_2, \tag{E.7}$$

with C_1 and C_2 both independent of g and $\boldsymbol{T}$.

As the two constants C_1 and C_2 are independent of g and $\boldsymbol{T}$, they will not change the optimization and the final generator g. We recover the classical WGAN formulation with some constraints on ν and g. The whitening property of ν is not a huge problem for GAN as in practice, ν can be chosen arbitrarily. Additionally, there are two different discriminator functions f and h, instead of one (Villani, 2008, Particular Case 5.4), as the squared euclidean distance does not respect the triangle inequality. To handle WGAN with the squared Euclidean distance or on even more complex cost function one can rely on existing methods (Liu et al., 2019; Laschos et al., 2019). Let us now prove the theorem:

Proof. We start from the GW formulation,

$$\min_{\boldsymbol{T}\in\Pi_{\mu\nu}} \sum_{i,j,k,l=1}^{I,I,K,K} |\mathcal{C}^{\mathcal{X}}(\boldsymbol{x}_i, \boldsymbol{x}_j) - \mathcal{C}^{\mathcal{Y}}(\boldsymbol{y}_k, \boldsymbol{y}_l)|^2 \boldsymbol{T}_{ik}\boldsymbol{T}_{jl} \tag{E.8}$$

$$= \min_{\boldsymbol{T}\in\Pi_{\mu\nu}} \sum_{i,j,k,l=1}^{I,I,K,K} -2\mathcal{C}^{\mathcal{X}}(\boldsymbol{x}_i,\boldsymbol{x}_j)\mathcal{C}^{\mathcal{Y}}(\boldsymbol{y}_k,\boldsymbol{y}_l)\boldsymbol{T}_{ik}\boldsymbol{T}_{jl} + C_2'' \tag{E.9}$$

$$= \min_{\boldsymbol{T}\in\Pi_{\mu\nu}} -2\sum_{i,l=1}^{I,K}\left(\sum_{j=1}^{I}\mathcal{C}^{\mathcal{X}}(\boldsymbol{x}_i,\boldsymbol{x}_j)\boldsymbol{T}_{jl}\right)\left(\sum_{k=1}^{K}\boldsymbol{T}_{ik}\mathcal{C}^{\mathcal{Y}}(\boldsymbol{y}_k,\boldsymbol{y}_l)\right) + C_2'' \tag{E.10}$$

$$= \min_{\boldsymbol{T}\in\Pi_{\mu\nu}} -2\sum_{i,l=1}^{I,K}\left(\boldsymbol{X}^\top\boldsymbol{X}\boldsymbol{T}\right)_{il}\left(\boldsymbol{T}\boldsymbol{Y}^\top\boldsymbol{Y}\right)_{il} + C_2'' \tag{E.11}$$

$$= \min_{\boldsymbol{T}\in\Pi_{\mu\nu}} -2\left\langle \boldsymbol{X}^\top\boldsymbol{X}\boldsymbol{T}, \boldsymbol{T}\boldsymbol{Y}^\top\boldsymbol{Y}\right\rangle + C_2'' \tag{E.12}$$

$$= \min_{\boldsymbol{T}\in\Pi_{\mu\nu}} -2\left\|\boldsymbol{X}\boldsymbol{T}\boldsymbol{Y}^\top\right\|_F^2 + C_2'' \tag{E.13}$$

$$= \max_{\boldsymbol{T}\in\Pi_{\mu\nu}} 2\left\|\boldsymbol{X}\boldsymbol{T}\boldsymbol{Y}^\top\right\|_F^2 + C_2'' \tag{E.14}$$

$$= \max_{\boldsymbol{G}\in\mathcal{F}}\max_{\boldsymbol{T}\in\Pi_{\mu\nu}} 2C_1\left\langle \boldsymbol{X}\boldsymbol{T}\boldsymbol{Y}^\top, \boldsymbol{G}\right\rangle + C_2'' \tag{E.15}$$

$$= \max_{\boldsymbol{G}\in\mathcal{F}}\max_{\boldsymbol{T}\in\Pi_{\mu\nu}} 2C_1\left\langle \boldsymbol{T}, \boldsymbol{X}^\top\boldsymbol{G}\boldsymbol{Y}\right\rangle + C_2'' \tag{E.16}$$

$$= \max_{\boldsymbol{G}\in\mathcal{F}}\max_{\boldsymbol{T}\in\Pi_{\mu\nu}} C_1\left[-\sum_{il=1}^{I,K}\boldsymbol{T}_{il}\left\|\boldsymbol{x}_i-\boldsymbol{G}\boldsymbol{y}_l\right\|_2^2 + \sum_{i=1}^{I}\boldsymbol{a}_i\left\|\boldsymbol{x}_i\right\|_2^2 + \sum_{k=1}^{K}\boldsymbol{b}_k\left\|\boldsymbol{G}\boldsymbol{y}_k\right\|_2^2\right] + C_2'' \tag{E.17}$$

$$= \max_{\boldsymbol{G}\in\mathcal{F}}\max_{\boldsymbol{T}\in\Pi_{\mu\nu}} -C_1\sum_{i,k=1}^{I,K}\left\|\boldsymbol{x}_i-\boldsymbol{G}\boldsymbol{y}_k\right\|_2^2\boldsymbol{T}_{ik} + C_2' + \sum_{k=1}^{K}\boldsymbol{b}_k\operatorname{tr}\left(\boldsymbol{y}_k^\top\boldsymbol{G}^\top\boldsymbol{G}\boldsymbol{y}_k\right) \tag{E.18}$$

$$= \max_{\boldsymbol{G}\in\mathcal{F}}\max_{\boldsymbol{T}\in\Pi_{\mu\nu}} -C_1\sum_{i,k=1}^{I,K}\left\|\boldsymbol{x}_i-\boldsymbol{G}\boldsymbol{y}_k\right\|_2^2\boldsymbol{T}_{ik} + C_2' + \operatorname{tr}\left(\boldsymbol{G}^\top\boldsymbol{G}\sum_{k=1}^{K}\boldsymbol{b}_k\boldsymbol{y}_k^\top\boldsymbol{y}_k\right) \tag{E.19}$$

$$= \max_{\boldsymbol{G}\in\mathcal{F}}\max_{\boldsymbol{T}\in\Pi_{\mu\nu}} -C_1\sum_{i,k=1}^{I,K}\left\|\boldsymbol{x}_i-\boldsymbol{G}\boldsymbol{y}_k\right\|_2^2\boldsymbol{T}_{ik} + C_2' + \operatorname{tr}\left(\boldsymbol{G}^\top\boldsymbol{G}\right) \tag{E.20}$$

$$= \min_{\boldsymbol{G}\in\mathcal{F}}\min_{\boldsymbol{T}\in\Pi_{\mu\nu}} C_1\sum_{i,k=1}^{I,K}\left\|\boldsymbol{x}_i-\boldsymbol{G}\boldsymbol{y}_k\right\|_2^2\boldsymbol{T}_{ik} - C_2' - \left\|\boldsymbol{G}\right\|^2 \tag{E.21}$$

$$= \min_{g\in\mathcal{F}}\min_{\boldsymbol{T}\in\Pi_{\mu\nu}} C_1\sum_{i,k=1}^{I,K}\left\|\boldsymbol{x}_i-g(\boldsymbol{y}_k)\right\|_2^2\boldsymbol{T}_{ik} + C_2 \tag{E.22}$$

$$= \min_{g\in\mathcal{F}}\max_{f(\boldsymbol{x}_i)-h(g(\boldsymbol{y}_j))\leq\|\boldsymbol{x}_i-g(\boldsymbol{y}_j)\|_2^2} C_1\left[\mathop{\mathbb{E}}_{\boldsymbol{x}\sim\mu}\left[f(\boldsymbol{x})\right] - \mathop{\mathbb{E}}_{\boldsymbol{y}\sim\nu}\left[h(g(\boldsymbol{y}))\right]\right] + C_2. \tag{E.23}$$

Line (E.9) transforms the square absolute loss into the negative scalar product with two additional terms that do depend on the marginal $\boldsymbol{a}$ and $\boldsymbol{b}$ but not anymore on $\boldsymbol{T}$. Line (E.16), which is quite artificial, uses the maximum closed form of the matrix $\boldsymbol{G}$ under the squared Frobenius norm constraint. Line (E.23) uses the dual formulation of the OT, as the equality is true for all $g \in \mathcal{F}$, this is also true for the minimum. □

This type of linear GAN has probably not a huge interest in practice, but this strong connection might be used to propose new WGAN or GW variants. While this has not been

explored in this thesis, we propose the Fused-WGAN which can be proved similarly to Theorem 12.

The Fused Gromov Wasserstein formulation (Vayer et al., 2018, 2019a) uses both the original OT problem and the GW problem, as shown on Equation (1.39) in Section 1.3.1. In addition to the notations of Theorem 12, c is any real value function $\mathcal{X} \times \mathcal{Y} \to \mathbb{R}$. Fused-WGAN is defined as,

$$\min_{\boldsymbol{T} \in \Pi_{\mu\nu}} \sum_{i,j,k,l=1}^{I,I,K,K} \mathcal{L}(\mathcal{C}^{\mathcal{X}}(\boldsymbol{x}_i, \boldsymbol{x}_j) - \mathcal{C}^{\mathcal{Y}}(\boldsymbol{y}_k, \boldsymbol{y}_l))\boldsymbol{T}_{ik}\boldsymbol{T}_{jl} + \beta \sum_{i,k}^{I,K} c(\boldsymbol{x}_i, \boldsymbol{y}_k)\boldsymbol{T}_{ik} \tag{E.24}$$

$$= \min_{g \in \mathcal{F}} \max_{f(\boldsymbol{x}_i) + h(g(\boldsymbol{y}_j)) \leq \|\boldsymbol{x}_i - g(\boldsymbol{y}_j)\|_2^2 + \beta c(\boldsymbol{x}_i, \boldsymbol{y}_k)} C_1 \left[\mathop{\mathbb{E}}_{\boldsymbol{x} \sim \mu} [f(\boldsymbol{x})] - \mathop{\mathbb{E}}_{\boldsymbol{y} \sim \nu} [h(g(\boldsymbol{y}))] \right] + C_2, \tag{E.25}$$

with β a regularization parameter.

The idea behind this formulation is to decide in which parts of the space $\mathcal{Y}$ will generate specific parts of the distribution μ. Similarly to Fused Gromov-Wasserstein, the function c will use additional labels of μ. For instance, we can use the first axis of $\mathcal{Y}$ to separate the generation of men and women people,

$$c(\boldsymbol{x}, \boldsymbol{y}) = \begin{cases} \boldsymbol{y}, \text{if men}(\boldsymbol{x}) \\ -\boldsymbol{y}, \text{otherwise} \end{cases} \tag{E.26}$$

This will ensure that if $\boldsymbol{y}$ is positive, it will be matched to women examples, thus generating women examples. This notion is highly related to the conditional GAN (Mirza and Osindero, 2014).

Appendix F

Résumé en Français

L'apprentissage automatique est un sous-domaine de l'intelligence artificielle, il vise à créer des algorithmes qui simulent l'intelligence humaine pour résoudre des tâches complexes. La particularité d'un processus d'apprentissage automatique est qu'il ne nécessite pas d'être programmé explicitement et s'appuie plutôt sur des exemples d'entraînement. Ces derniers sont décrits soit directement avec des caractéristiques dans un espace vectoriel (e.g., poids, âge, prix...) soit avec une représentation structurée (e.g., image, texte, son...). Dans certaines applications, les échantillons ne sont représentés que relativement les uns aux autres, par exemple dans des graphes constitués de connexions entre noeuds, comme dans les réseaux sociaux ou les molécules. L'idée derrière l'apprentissage automatique est de concevoir un algorithme qui résout une tâche souhaitée en apprenant une fonction à partir de l'ensemble des données disponibles et qui généralisera bien sur des exemples non vus. L'une des tâches les plus courantes est la classification binaire supervisée, où l'objectif est de distinguer deux classes. Par exemple, étant donné un ensemble d'apprentissage de photos étiquetées de chats et de chiens, un algorithme de classification apprend à faire la distinction entre les deux catégories. Il sera bon s'il prédit bien la classe des images non vues. Cette capacité de généralisation est généralement estimée à partir d'un ensemble dit de test. Au cours de la dernière décennie, l'apprentissage automatique a fait l'objet d'une grande attention avec diverses applications cibles, dans la vision par ordinateur (segmentation sémantique, suivi d'objets), les arts (génération d'images ou de musique), la détection des fraudes, pour n'en citer que quelques-unes.

Dans de nombreux scénarios d'apprentissage automatique, la comparaison de deux ensemble de points ou plus généralement de mesures de probabilité présente un grand intérêt. Cela peut se produire lorsqu'on évalue la similarité entre deux images représentées par des nuages de points dans l'espace RVB, entre deux textes codés par des mots incorporés ou entre deux ensembles de cellules dont le génome/transcriptome est décrit numériquement dans un espace de caractéristiques.

Une direction possible pour aborder cette tâche est d'utiliser la théorie du Transport Optimal

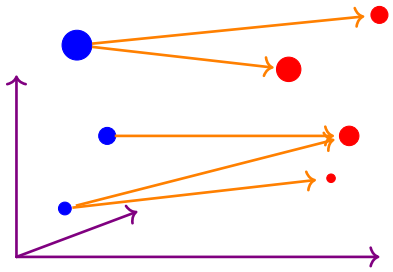


Figure F.1: *Illustration du problème du Transport Optimal entre deux distributions discrètes.* *Cited on page [175]*

(OT), introduite à l'origine par Gaspard Monge (Monge, 1781), visant à répondre à la question suivante : comment déplacer des ressources à partir de certains endroits pour satisfaire des exigences à d'autres endroits avec le moins d'effort possible, en ce sens que le coût global du déplacement des ressources est minimisé. Par exemple, les ressources peuvent être des soldats venant de différentes bases militaires qui doivent être envoyés à différents endroits de la ligne de front, et le commandant militaire veut minimiser le coût moyen de déplacement des soldats pour éviter les efforts inutiles. Tel qu'il a été introduit à l'origine, le problème de Monge empêchait la possibilité de répartir les soldats provenant de la même base militaire, ce qui conduisait à un problème mal posé sans garantie d'unicité et d'existence de la solution. Plus de 200 ans plus tard, Leonid Kantorovitch (Kantorovich, 1942) a proposé une formulation mathématique relaxée du problème de transport qui définit notamment une distance réelle entre des distributions de probabilité : la distance de Wasserstein. Cette dernière a fait l'objet d'une grande attention au cours des dernières années par la communauté de l'apprentissage automatique en tant qu'outil puissant pour comparer des mesures de probabilité. Un exemple de la solution de Transport Optimal est visible sur la Figure F.1, où la masse entière de la distribution bleue est envoyée sur la distribution rouge par les flèches orange. La distance de Wasserstein entre les deux distributions est la somme de toutes les distances orange pondérées par la masse correspondante transportée sur chaque flèche. Il s'avère qu'un élément crucial du Transport Optimal est la fonction utilisée comme distance entre les échantillons des deux nuages de points. Dans un tel cadre discret, le problème du Transport Optimal est entièrement défini par une matrice de coûts, qui contient toutes les distances par paire entre les points des deux mesures de probabilité. Dans le domaine du Transport Optimal, cette matrice de coûts est généralement appelée *métrique de terrain*, *fonction de terrain* ou *coût de terrain*. La distance la plus courante et la plus naturelle est la distance euclidienne, augmentée d'une certaine puissance p, ce qui conduit à la p-Wasserstein distance.

La métrique de terrain est au cœur de plusieurs contributions de ce manuscrit. En particulier, nous exploitons le cadre d'apprentissage métrique pour apprendre une métrique du terrain sous la forme d'une distance de Mahalanobis utilisée pour traiter les tâches d'adaptation au domaine, c'est-à-dire lorsque nous observons un changement de distribution entre les données d'apprentissage et de test. Nous étudions également, dans la première partie de cette thèse,

d'un point de vue théorique comment apprendre une métrique de terrain stable.

Une deuxième série de contributions repose sur l'évolutivité du Transport Optimal. En effet, certaines variantes visent à comparer des mesures de probabilité qui ne se trouvent pas nécessairement dans le même espace de caractéristiques. C'est le cas de la distance de Gromov Wasserstein dont la complexité algorithmique empêche son utilisation sur de grands jeux de données. Nous proposons dans cette thèse un algorithme rapide pour ce problème basé sur des stratégies d'échantillonnage. A partir de cet algorithme, nous introduisons une extension généralisant les problèmes de Wasserstein et de Gromov Wasserstein, ce qui nous permet d'appliquer le Transport Optimal sur des tenseurs de haute dimension.

Financement et contexte de cette thèse. Cette thèse s'inscrit dans le cadre du projet TADALoT[1], financé par la région Auvergne-Rhône-Alpes (France) avec le Pack Ambition Recherche (2017, 17 011047 01). Cette thèse a été réalisée au sein de l'équipe Data Intelligence du laboratoire Hubert Curien. Ce laboratoire est une unité de recherche (UMR 5516) entre l'Université Jean Monnet de Saint-Étienne, le CNRS et l'Institut d'Optique Graduate School.

Organisation Le manuscrit est composé de 5 chapitres.

- Le premier chapitre fournit le contexte nécessaire pour le reste du document. Il introduit en détail la théorie du Transport Optimal, certains solveurs et certaines extensions, notamment le problème de Gromov-Wasserstein. Ensuite, quelques connaissances de base sont présentées sur le cadre standard de l'apprentissage automatique et le cadre de l'adaptation au domaine. Ce chapitre présente également les algorithmes Frank-Wolfe et Mirror Descent en tant que méthodes d'optimisation clés pour les formulations du Transport Optimal étendues.

- Le deuxième chapitre explore la capacité d'apprentissage de la métrique de base du problème de Transport Optimal pour une tâche d'adaptation de domaine non supervisée. Cette dernière vise à déployer sur une distribution cible un modèle appris à partir de données sources étiquetées. Le choix de la distance euclidienne pour comparer des points peut ne pas être le meilleur choix compte tenu de la tâche à accomplir. Nous proposons plutôt d'apprendre une distance de Mahalanobis en utilisant une approche d'apprentissage de métrique. L'avantage de la distance de Mahalanobis vient de sa capacité à réduire la dimensionnalité de l'espace des caractéristiques en apprenant une matrice de faible rang. Une limite de généralisation de l'erreur cible est dérivée pour guider la conception de notre algorithme, appelé Apprentissage de Métrique pour le Transport Optimal (MLOT).

- Le troisième chapitre se concentre d'un point de vue théorique sur la stabilité du Transport Optimal dans le pire scénario consistant à sélectionner **la pire matrice de coûts possible**. Ceci conduit à un problème min-max visant à réduire la distance tout en recherchant

[1] https://twitwi.github.io/tadalot/

la matrice de coûts, dans un ensemble éventuellement infini, qui augmente le plus cette distance. Le solveur proposé, basé sur une méthode élégante de "Cut in Plane", trouve les points-selle de ce problème min-max. Quelques expériences soulignent l'intérêt de l'algorithme par rapport à l'état de l'art en termes de complexité temporelle et de réduction du bruit. En nous appuyant sur ce cadre théorique, nous définissons une notion de *stabilité locale* pour les matrices de coûts afin de sélectionner la plus stable. Nous montrons que cette stabilité est fortement corrélée avec une notion de sensibilité au bruit. Nous présentons quelques expériences pour sélectionner une matrice de coût stable dans une tâche de transfert de couleur.

- Le quatrième chapitre est consacré à une extension de la théorie du Transport Optimal aux espaces incomparables, connue sous le nom de problème de Gromov-Wasserstein. Les solveurs existants reposent sur des fonctions de perte spécifiques pour calculer efficacement le gradient et ne sont pas adaptés à un cadre général. Nous proposons une méthode rapide pour résoudre le problème de Gromov-Wasserstein pour n'importe quelle fonction de perte, qui repose sur une approximation stochastique du gradient en recourant à un schéma d'échantillonnage. Nous analysons également la convergence vers un point stationnaire de l'algorithme proposé qui inclut notamment une preuve de convergence pour les solveurs de Gromov-Wasserstein existants dans le cas concave. Enfin, nous montrons qu'une amélioration encore plus rapide peut être obtenue lorsque l'approximation stochastique est basée sur un seul exemple, ce qui nous permet d'utiliser le solveur rapide unidimensionnel du Transport Optimal.

- Le cinquième chapitre est consacré à la définition d'une **nouvelle distance entre tenseurs d'ordres arbitraires**. Dans le cadre classique du Transport Optimal, la fonction de coût compare des vecteurs qui proviennent de points dans un espace vectoriel ; dans le problème de Gromov-Wasserstein, on compare des vecteurs (souvent de taille 1) qui proviennent de matrices. Nous proposons le problème du Transport Optimal entre Tenseurs (OTT), où la perte peut comparer des vecteurs provenant de tenseurs d'ordres quelconques. Pour ce faire, nous proposons un cadre qui permet d'utiliser différents plans de transport le long des différentes dimensions au lieu de s'appuyer sur un unique plan de transport. Avec l'introduction d'une nouvelle notion de barycentre pour OTT, nous montrons des résultats compétitifs dans une tâche de clustering basée sur la comparaison. De plus, quelques expériences d'adaptation de domaine sont menées sur des jeux de données représentés sous forme de tenseurs, ce qui souligne l'intérêt de cette nouvelle formulation par rapport à l'état de l'art.

Pour des raisons de clarté du manuscrit, certaines figures, expériences ainsi que de longues preuves sont reléguées dans les annexes. Deux travaux intéressants, liés à cette thèse, mais qui n'ont pas été explorés à leur plein potentiel sont disponibles en annexe E.1 et E.3. De plus, comme contribution technique, l'appendice E.2 présente un logiciel développé pendant cette

thèse qui crée des transitions entre les diapositives d'une présentation. Ces transitions sont optimales, dans le sens où elles minimisent le coût du passage d'une diapositive à une autre.

Table des Matières

Remerciement **iii**

Introduction **1**

Liste des Publications **5**

Liste des Notations **7**

1 Contexte **9**

1.1 Transport Optimal 9

1.2 Apprentissage Machine 21

1.3 Méthodes d'optimisation au premier ordre pour le Transport Optimal 25

1.4 Conclusion 32

2 Apprentissage de Métrique pour le Transport Optimal **33**

2.1 Introduction 33

2.2 OTDA et apprentissage de métrique 35

2.3 Analyse théorique de l'Adaptation de Domaine avec le Transport Optimal . . . 37

2.4 MLOT: Apprentissage de Métrique en Transport Optimal pour l'Adaptation de Domaine 40

2.5 Experiments 41

2.6 Conclusion 44

3 Un Couteau Suisse pour le Problème Minimax du Transport Optimal **47**

3.1 Introduction 47

3.2 Connaissances préalable 48

3.3 Transport Optimal Robuste avec un ensemble convexe de matrices de coût . . . 49

3.4 Experiments 58

3.5 Conclusion 62

4 Sampled Gromov Wasserstein **65**

4.1 Introduction 65

4.2 Contexte sur Gromov Wasserstein (GW) 67
4.3 Approches pour résoudre GW 67
4.4 Optimisation de GW 69
4.5 Expériences 78
4.6 Conclusion 85

5 Transport Optimal entre Tenseurs 87
5.1 Introduction 87
5.2 Connaissances préalable 89
5.3 Transport Optimal entre Tenseurs (OTT) 90
5.4 Algorithme pour résoudre OTT 92
5.5 Résultats théoriques 93
5.6 Expériences 94
5.7 Conclusion 99

Conclusion 101

A Appendice de l'Apprentissage de Métrique pour le Transport Optimal 105
A.1 Expériences en Adaptation de Domaine 105

B Appendice du Couteau Suisse pour le Problème Minimax du Transport Optimal 109
B.1 Preuves de la Section 3.3.2 109
B.2 Preuves de la Section 3.3.3 111
B.3 Preuves de la Section 3.3.4 114
B.4 Évaluations expérimental 117

C Appendice de Sampled Gromov Wasserstein 121
C.1 GW optimisation 121
C.2 Expériences 129

D Appendice du Transport Optimal entre tenseurs 139
D.1 Illustration de la différence entre OTT et Co-OT 139
D.2 Mirror Descent 140
D.3 Complexité du calcul du gradient de OTT 140
D.4 Résultats théoriques 142

E Recherches inachevées liées aux contributions de cette thèse 169
E.1 Pointwise Wasserstein et Sliced Wasserstein 169
E.2 Transition de diapositive optimale 170
E.3 Gromov Wasserstein et Wasserstein GAN 171

F Résumé en Français **175**
F.0 Listes des Figures 194
F.0 Listes des Tables 196
F.0 Listes des Algorithmes 197
F.0 Bibliographie 199

Conclusion

Cette thèse explore l'intérêt d'utiliser la théorie du Transport Optimal en Apprentissage Automatique. Plus précisément, elle se concentre sur la métrique de terrain du Transport Optimal. Le choix de cette métrique de terrain est un élément clé dans toute application basée sur la théorie du Transport Optimal et nous pensons que la distance euclidienne (au carré) est trop souvent choisie par défaut. La première partie de cette thèse aborde ce problème en apprenant une distance de Mahalanobis dans le contexte d'une tâche d'adaptation au domaine. Dans ce contexte particulier, nous montrons que le Transport Optimal peut utiliser avantageusement les étiquettes disponibles du domaine source pour apprendre une distance appropriée. Ce processus d'apprentissage conjoint entre le plan de transport et la métrique conduit à un algorithme appelé *Metric Learning for Optimal Transport* (MLOT). Nous réalisons des expériences d'adaptation de domaine qui montrent l'intérêt d'optimiser une telle métrique de terrain. Bien que MLOT ait été conçu dans le contexte de l'adaptation de domaine, il est intéressant de noter que dans toute application de Machine Learning, le choix de la bonne fonction de coût est délicat. Pour résoudre cette tâche, nous proposons une manière générale de traiter plusieurs fonctions de coût, ou matrices de coût, en examinant le scénario le plus défavorable. Dans ce contexte, le plan de transport trouvé ne dépend pas d'une seule matrice de coûts et est plus robuste au bruit. Sur la base de cette formulation, une notion de stabilité pour les matrices de coûts est proposée afin de pouvoir choisir la matrice de coûts la plus stable. Elle correspond à une matrice à laquelle est associée une distance de Wasserstein qui ne change pas beaucoup lorsqu'on regarde le pire cas dans une boule de Mahalanobis centrée autour de cette matrice. Les expériences montrent une corrélation entre cette notion de stabilité et la sensibilité au bruit des matrices considérées.

Dans la deuxième partie de la thèse, nous nous concentrons sur les extensions de la formulation du Transport Optimal, notamment la distance de Gromov Wasserstein capable de comparer des distributions qui ne se trouvent pas nécessairement dans le même espace. Pour approximer efficacement le problème (difficile) de Gromov Wasserstein, les solveurs existants s'appuient sur des fonctions de perte particulières. Pour remédier à cette limitation, nous proposons un algorithme stochastique rapide, capable de gérer des fonctions de pertes arbitraires. Cette méthode est accompagnée d'une borne de convergence vers un point stationnaire qui couvre notamment une méthode existante (EGW) dans le cas concave. Lorsque le nombre d'échantillons est réduit à un, une variante très rapide peut être utilisée, en s'appuyant sur le solveur unidimensionnel du Transport Optimal. Nous soulignons l'intérêt d'utiliser différentes

pertes dans des expériences de classification de graphes. Sur la base de cet algorithme évolutif, nous introduisons un schéma de généralisation englobant à la fois la distance de Wasserstein et la distance de Gromov Wasserstein ainsi que la distance Co-OT. La formulation qui en résulte est appelée *Transport Optimal entre tenseurs*. Cette nouvelle distance permet de traiter des tenseurs de taille arbitraire au lieu d'utiliser uniquement les positions dans un espace vectoriel ou les distances paire à paire. Nous réalisons des expériences incluant des tâches d'apprentissage basées sur la comparaison dans un cadre basé sur les triplets où les points sont définis uniquement par leurs relations avec deux autres points.

Comme souvent dans la recherche, cette thèse pourrait ouvrir plus de questions qu'elle n'offre de réponse. Quelques directions de recherche prometteuses possibles sont données ci-dessous, pour chacun des différents chapitres du manuscrit.

Apprentissage métrique pour le Transport Optimal Comme ce chapitre se concentre principalement sur une application pratique, les trois extensions proposées sont des algorithmes.

Une extension naturelle de notre méthode serait une version "Deep" de MLOT pour permettre son utilisation directement sur les images originales dans les tâches de vision par ordinateur. La méthode MLOT peut être utilisée pour des tâches de vision par ordinateur, du moins pour tout ensemble de données basées sur des images. Au lieu de minimiser la fonction objectif par rapport à la métrique de Mahalanobis, nous pourrions plutôt minimiser l'ensemble du réseau neuronal profond (NN) qui génère l'espace des caractéristiques. Ainsi, la métrique apprise serait l'ensemble du réseau neuronal au lieu d'une simple fonction linéaire. Généralement, un réseau neuronal existant, déjà formé sur un énorme ensemble de données tel que Image Net (Deng et al., 2009), est utilisé et les poids de ce réseau sont seulement ajustés. Dans ce cas, le choix des hyperparamètres est plus complexe. Tout d'abord, il pourrait y avoir plus d'hyperparamètres en raison du Réseau de Neurone utilisé. Mais surtout, le temps de calcul augmente considérablement et la validation inverse, utilisée dans ce chapitre, peut prendre trop de temps pour couvrir raisonnablement la grande grille d'hyperparamètres. Même si plusieurs hyperparamètres doivent être définis par défaut, une telle approche conduirait probablement à de meilleurs résultats pour la classification d'images.

La deuxième extension possible consiste à utiliser un programme de différenciation automatique de l'algorithme OTDA, à chaque itération de MLOT, pour suivre l'impact de $\boldsymbol{L}$ sur le plan de transport $\boldsymbol{T}$. Dans ce cas, au lieu de simplement alterner l'optimisation entre le plan de transport $\boldsymbol{T}$ et $\boldsymbol{L}$, $\boldsymbol{T}$ serait une variable non fixe dépendant de $\boldsymbol{L}$. Une telle approche pourrait conduire à un meilleur minimum qu'une optimisation naïve alternée car le problème n'est pas convexe. Le principal problème est le temps et la mémoire nécessaires, car OTDA utilise plusieurs appels de l'algorithme de Sinkhorn.

Une dernière extension possible de MLOT serait de traiter le cadre semi-supervisé où certains labels sont disponibles dans le domaine cible. Dans ce cas, le même algorithme d'apprentissage de métrique pourrait être utilisé dans le domaine cible.

Un couteau suisse pour le Transport Optimal Minimax Bien que la méthode de " Cut in Plane" proposée soit très générale, un problème de minimax "petit" doit toujours être résolu à chaque itération. En pratique, nous n'utilisons que des ensembles spécifiques de matrices de coûts (ensemble basé sur Mahalanobis ou nombre fini de matrices) et pour d'autres types d'ensembles, l'algorithme doit être modifié. Par exemple, on pourrait avoir recours à un algorithme de Frank-Wolfe comme celui utilisé pour générer la Figure 3.1, alors que nous n'avons utilisé qu'un solveur linéaire pour les expériences. Ainsi, une méthode générale pour résoudre une telle partie de l'algorithme serait intéressante à explorer. L'algorithme de Frank-Wolfe qui a été utilisé pour générer la Figure 3.1 pourrait être une bonne direction pour fournir une approximation générale de minimax pour tout ensemble.

La notion proposée de stabilité de Wasserstein pourrait être plus intéressante si la recherche d'une matrice de coût stable n'était pas réduite à un nombre fini de matrices de coût. En effet, en pratique, on ne sélectionne qu'un certain nombre de matrices de coûts et on teste la stabilité pour chacune d'entre elles. On pourrait imaginer de chercher la matrice de coût la plus stable dans un ensemble infini, par exemple dans le polytope généré par certaines matrices. Cela conduira à un min maxmin très complexe.

Sampled Gromov Wasserstein L'un des principaux problèmes de la preuve de convergence vers un point stationnaire fournie dans ce chapitre est l'hypothèse selon laquelle l'algorithme de Sinkhorn produit le $\boldsymbol{T}^*$ exact du problème OT régularisé par l'entropie 1.8. En pratique, comme le nombre d'itérations est limité, le résultat n'est qu'une approximation qui ne respecte pas totalement les contraintes marginales. Ainsi, la borne de convergence n'est pas totalement correcte et pourrait être améliorée dans un travail futur. De plus, comme la vitesse de convergence dépend fortement de ϵ, nous pourrions remplacer le nombre maximal d'itérations de Sinkhorn P par une valeur plus petite dépendant de ϵ. Pour finir, la borne liée à la partie échantillonnage peut être grandement améliorée, au lieu de se baser sur le pire scénario, nous pourrions utiliser certaines inégalités de concentration. Plus généralement, la limite proposée peut être considérablement réduite.

Dans un laps de temps très court, pendant les publications de l'article associé à ce chapitre, deux autres travaux (Sato et al., 2020; Chowdhury et al., 2021) ont utilisé le solveur unidimensionnel du Transport Optimal en définissant chaque point par sa distance à un autre point. Ainsi, cette idée générale pourrait être une ligne de recherche future prometteuse dans autour du problème de Gromov Wasserstein.

Du point de vue de l'application, tout problème qui repose sur GW pourrait utiliser SaGroW pour permettre plus de choix pour la fonction de perte ou PoGroW pour avoir une approximation très rapide.

Transport Optimal entre Tenseurs Une piste de travail intéressante serait de prouver la convergence de l'algorithme proposé vers des points stationnaires. Comme il existe une importante littérature sur la convergence des algorithmes de Mirror Descent, l'une des preuves

de convergence pourrait être utilisée ou adaptée au cas particulier des OTT. L'une des preuves de convergence pour l'algorithme de Mirror Descent, dans la vaste littérature, pourrait couvrir le cas particulier de l'algorithme proposé. Une telle preuve couvrirait également l'algorithme *Sampled Gromov Wasserstein* comme cas particulier.

Une extension directe de la formulation OTT est le "Fused"-OTT, de manière similaire au Fused-Gromov Wasserstein (Vayer et al., 2018). Il permettrait d'aligner des ensembles de données qui sont représentés par plusieurs tenseurs d'ordres différents au lieu d'un seul. L'algorithme pour résoudre un tel problème ne changerait pas beaucoup, car le gradient pourrait toujours être interprété comme une somme d'espérance.

Nous espérons que cette thèse a été intéressante et surtout qu'elle permettra l'émergence de nouvelles idées et créera de nouveaux liens entre les différents domaines de la statistique, de l'optimisation, du Machine Learning et du Transport Optimal.

List of Figures

1 Illustration of the Optimal Transport problem between two discrete distributions.
Cited on page [1] 2

1.1 Illustration of the Optimal Transport barycenter (dashed orange line) between two continuous distributions (blue and red).
Cited on page [9] 10

1.2 Representation in two dimensions of the Optimal Transport problem. The $\boldsymbol{C}$ vector represents the position of the cost matrix in the space. The best transport plan $\boldsymbol{T}^*$ is the one which is the "most" perpendicular to the vector $\boldsymbol{C}$ because the scalar product has to be minimized.
Cited on pages [12,12] 12

1.3 Representation in three dimensions of the Optimal Transport problem. The polytope of the possible transport plans is now represented in a subspace of the matrix space.
Cited on pages [12,12,12] 13

1.4 1D Optimal Transport (**Left**) Optimal transport plan with any strictly convex cost function. (**Right**) Any transport plan is optimal with the absolute cost, which is less intuitive.
Cited on pages [13,13] 14

1.5 Representation of the discontinuity of the transport plan for a small modification of the points. The top row corresponds to the optimal transport plan for the original OT problem, the bottom row corresponds to the regularized version. From the left to the right, the bottom right blue point is slightly modified, $(1 \pm \alpha, -1)$ with a small α.
Cited on pages [11,16] 16

1.6 Illustration of GW, with only one term $\boldsymbol{L}_{ijkl}$ of the quadruple sum of Eq. (1.22).
Cited on page [19] 19

1.7 Illustration of a Wasserstein Barycenter (orange) between two discrete distributions.
Cited on page [20] 20

1.8 What is behind the cow? 21

1.9 Illustration of a DA task with samples of the source distribution μ_s composed of one Gaussian for each class and samples of the target distributions μ_t linearly shifted from the source distributions. The linear separator is a SVM (Cortes and Vapnik, 1995).
Cited on page [22] 23

1.10 Illustration of a DA task solved with OT. In the left figure, the distribution μ_s is transported to μ_t using the transport plan, then a linear separator is learned. In the right figure, the linear separator has a good prediction on the target points.
Cited on pages [23,25] 24

1.11 Interpolation (orange with dash) between two Gaussian's distributions (blue and red) with either the Wasserstein distance or the KL divergence.
Cited on pages [20,24] 25

1.12 Illustration of an iteration of the FW algorithm in the convex setting. Credit to Stephanie Stutz and Martin Jaggi (Jaggi, 2013).
Cited on page [26] 26

2.1 Behavior of MLOT on a toy dataset. On the left, the original source and target examples. In the middle, OTDA fails to transport correctly the blue and red classes. On the right, the proposed MLOT which combines a learned metric and some per-domain dimensionality reduction, leads to a perfect transportation plan. Notice the difference in scale between the two axes.
Cited on pages [34,42] 34

2.2 Standard PCA projecting a discrete distribution $\hat{\mu}$ (data in orange) onto a one-dimensional subspace (in purple). This projection is equivalent to finding the optimal mapping function g that minimizes the Wasserstein distance $\mathcal{W}(\hat{\mu}, g\#\hat{\mu})$.
Cited on page [37] 37

2.3 Workflow of MLOT. Cited on page [41] 41

2.4 Absolute difference between the mean accuracy of MLOT and OTDA (on the left). The rows (resp. columns) correspond to the results with different values of the regularization parameter λ_c (resp. ϵ) on the entire Office-Caltech dataset with SURF features. On the right, accuracy of MLOT for each pair of parameters.
Cited on pages [42,43,43] 44

3.1 Interpolation between OT and $\text{SRW}_{d=1}$ on a binary toy classification problem with each class consisting of 5 points sampled from Gaussians centered on the edge of a 10-dimensional hypercube with $\sigma = 1$ with 10 additional random noise features. The transport is computed between the 2 classes using the setting of Proposition 1 with $q \in \{1, 1.1, 2, 5, \infty\}$. **(top row)** Mahalanobis matrices $\mathbf{M}^*$ and their rank; **(bottom row)** Couplings $\boldsymbol{T}^*$ and the associated value of the Wasserstein distance.
Cited on pages [51,57,102,182,182] 51

3.2 Illustration of a saddle point on a minimax problem. The scalar product between the transport plan and the cost matrix is displayed on the third axis. The α axis allows to move from $\mathbf{C}_1$ to $\mathbf{C}_2$. Similarly the β axis allows to move from $\boldsymbol{T}_1^*$ to $\boldsymbol{T}_2^*$. In both axes, all the possible matrices are represented.
Cited on page [55] 54

3.3 Illustration of the notion of matrix cost stability. Every matrix $\mathbf{C}_m$ is normalized so as to get a matrix $\tilde{\mathbf{C}}_m$ which lies on the norm-2 sphere. $\bar{\mathbf{C}}_m$ is the minimizer of Problem 3.4. The stability (Definition 1) comes from the difference of the cost transports induced by $\bar{\mathbf{C}}_m$ and $\tilde{\mathbf{C}}_m$.
Cited on page [58] 58

3.4 **(left)** Evolution of the error along the iterations for $|\mathcal{C}| \in \{10, 40, 90\}$; **(middle)** Evolution of the error with a regularization parameter $\epsilon \in \{1, 0.1, 0.01\}$; **(right)** Execution time of our algorithm vs solving the original LP problem with $|\mathcal{C}| \in \{10, 20, ..., 90\}$ and $n = m = 100$. The experiments are repeated 30 times. The median and the interval between the first and third quartiles are reported.
Cited on pages [59,59,59] 60

3.5 **(left)** Results obtained on the fragmented hypercube for $I = K = 250$, $D = 30$ and $d = 2$ with **(top row)** our approach with 2D projections and Mahalanobis distances; **(bottom row)** Original OT problem and SRW method of (Paty and Cuturi, 2019a); **(middle)** Sorted eigenvalues of $\mathbf{M}^*$ obtained using RKP averaged over 100 runs for different values of k reveals a phase transition between d dominant and the $d + 1$ eigenvalues; **(right)** Correlation between the stability and the sensitivity to noise.
Cited on pages [60,60,61] 61

3.6 Source (ocean) and target (sky) images considered as probability distributions.
Cited on page [61] 61

3.7 Cost matrices sorted by Wasserstein stability. The first 50 are Mahalanobis cost matrices, while the last 50 are random cost matrices.
Cited on pages [61,61] 62

4.1 Illustration of GW, with only one term L_{ijkl} of the quadruple sum of Eq. (4.2).
Cited on page [67] 68

4.2 Intuition behind PoGroW when $j, l = 0, 1$ are sampled from $\boldsymbol{T}$: only the distances to x_0 in $\mathcal{X}$ (on the left) and to y_1 in $\mathcal{Y}$ (on the right) characterize a pair, and then $\boldsymbol{T}'$ can be computed in $O(N \log N)$ like in 1D OT.
Cited on page [73] 73

4.3 (**Top**) Representation of the two unimodal Gaussian distributions μ and ν in their respective space. $\boldsymbol{x}_n$ and $\boldsymbol{y}_n$ are the two sampled points from the previous transport plan π_n and they induce a new transport plan π_{n+1}. This new transport plan will spread on a sphere the mass of a point $\boldsymbol{x}$, thus in this 1D case half of the mass is sent to $\boldsymbol{y}$ and half to $\boldsymbol{y}'$. (**Bottom**) Representation of the two density functions $f_{\boldsymbol{x}_n}$ and $f_{\boldsymbol{y}_n}$ to explain why π_{n+1} send the mass of $\boldsymbol{x}$ to $\boldsymbol{y}$ and $\boldsymbol{y}'$.
Cited on page [75] 75

4.4 Estimated value of $\mathcal{E}(\boldsymbol{T})$ as sparsity decreases due to an increasing ϵ regularization in EGW (left) and evolution of the time required for its estimation as N grows (right). The absolute loss is used in these experiments and the distributions take the form of two graphs generated using a gaussian random partition graph (Brandes et al., 2003). For a given ϵ and N, the same $\boldsymbol{T}$ (obtained using EGW) is passed to the three considered methods: *Real*) an exact one which computes completely $\mathcal{E}(\boldsymbol{T})$, *Sampled*) our sampling method described in Section 4.4.5, and, *Sparse*) a sparse approximation which keeps only the $2N$ largest values of $\boldsymbol{T}$ and sets the other entries to 0. The mean and 2 standard deviations over 10 runs are displayeded on both figures. When the standard deviation is not visible, it corresponds either to a deterministic method or a value very close to 0.
Cited on pages [77,77,129] 77

4.5 Computational time of various methods to compute the distance between samples from two mixtures of gaussians. The mean and the standard deviation over 10 runs are reported.
Cited on page [79] 79

4.6 GW distance estimation between samples from two mixtures of gaussians. The mean and standard deviation over 10 runs are reported for the stochastic methods.
Cited on page [81] 80

4.7 GW distance estimation between synthetic graphs (Brandes et al., 2003). The mean and standard deviation over 10 runs are reported for the stochastic methods.
Cited on page [81] 80

4.8 Impact of the number of samples M and the number of iterations S for SaGroW on the GW distance estimation and computational time, for two sets of 500 points sampled from two mixtures of gaussians. The mean and standard deviation over 10 runs are displayed.
Cited on pages [81,81,131,131,131,133,134,134,190,190,190,190] 82

4.9 Impact of the Kullback-Leiber regularization ϵ for EGW and S-GWL on the GW distance estimation and computational time, for two sets of 500 points sampled from two mixtures of gaussians.
Cited on pages [81,131,131,131,133,134,134,190,190,190,190] 82

4.10 Impact of the number of iterations S for PoGroW on the GW distance estimation and computational time, for two sets of 500 points sampled from two mixtures of gaussians. The mean and standard deviation over 10 runs are displayed. To take advantage of the large stochasticity, the minimum over 10 runs is also display.
Cited on pages [81,81,131,131,131,133,134,134,190,190,190,190] 83

4.11 The GW distance is estimated between μ and ν, two Gaussians with $\begin{pmatrix} 1 & 00 & v \end{pmatrix}$ and $\begin{pmatrix} 1 \end{pmatrix}$ variance respectively. Different values of v are tested. Each method outputs a transport plan π and the distance is estimated using a Monte Carlo approximation, the mean and standard deviation are reported.
Cited on pages [84,85] 85

5.1 **(Left)** Transport plan $\boldsymbol{T}^1$ between 400 images (only digits 0 and 1) of MNIST and USPS datasets; **(Middle and Right)** (top left) An example from MNIST and (bottom right) an example from USPS with a 90° right rotation; (top right) the OT plan $\boldsymbol{T}^2$ between the rows of MNIST and USPS; (bottom left) the OT plan $\boldsymbol{T}^3$ between the columns of MNIST and USPS; the arrows explain how to match the pixels between the two datasets using $\boldsymbol{T}^2$ and $\boldsymbol{T}^3$ obtained with OTT.
Cited on pages [88,91,139] 89

5.2 Representation of various formulations of OTT with, each time, the two datasets and the different transport plans (best viewed in color).
Cited on pages [87,87,87,88,88,90,90,91,91] 91

5.3 Target accuracy averaged over all the datasets. The shadow area represents the standard deviation for the stochastic methods. (**Left**) Target accuracy for various values of ϵ. The black symbols correspond to the value of ϵ associated with the lowest distance on average of each method. (**Middle**) Target accuracy for an increasing target supervision. (**Right**) Target accuracy for an increasing number of similar known users who rated both old and new movies. Again, the black symbols correspond to the lowest distances.
Cited on pages [96,96,96] 97

B.1 **Left to right, top to bottom**: Gaussians, MNIST 0-to-1, MNIST 3-to-0, MNIST 6-to-5, MNIST 7-to-1, MNIST 7-to-4 data sets. Y-axis (left) is the difference between the OT cost with $\mathbf{C}_i$ and $\mathbf{C}_i + \mathbf{E}^{\mathbf{M}}$. Y-axis (right), the Wasserstein stability defined in Section 3.5. Each column is a different cost matrix, the matrices are ordered by the Wasserstein stability.
Cited on page [118] 118

B.2 **Top row**: Original images of ocean sunset and ocean sky. **Middle row**: (**left**) most stable cost matrix, (**right**) squared Euclidean based cost matrix. Bottom row: (**left**) least stable Mahalanobis cost matrix, (**right**) least stable cost matrix. Notice the quality difference between the most stable matrix and the squared Euclidean based one in the area just under the cloud.
Cited on page [119] 119

B.3 **Top row**: Original images of woods and autumn. **Middle row**: (**left**) most stable cost matrix, (**right**) Euclidean based cost matrix. Bottom row: (**left**) least stable Mahalanobis cost matrix, (**right**) least stable cost matrix.
Cited on page [120] 120

C.1 Computational time (left) and GW distance estimation (right) between points sampled from mixtures of Gaussians.
Cited on pages [129,129] 130

C.2 Computational time (left) and GW distance estimation (right) on synthetic graphs (Brandes et al., 2003).
Cited on pages [129,129] 130

C.3 Computational time (left) and GW distance estimation (right) between sampled points from mixtures of Gaussians with the square loss.
Cited on pages [130,130] 131

C.4 Similar to the Figures 4.8, 4.9 and 4.10 in Chapter 4. (**Left**) Absolute loss. (**Right**) Square loss.
Cited on page [131] 131

C.5 Hyperparameters analysis on a Stochastic Block Model dataset with 200 nodes for each graphs. (**Left**) Absolute loss. (**Right**) Square loss.
Cited on page [131] 132

C.6 Hyperparameters analysis on a mixture of Gaussians with 200 points sampled for each distributions. (**Left**) Absolute loss. (**Right**) Square loss.
Cited on page [131] 132

C.7 Hyperparameters analysis on a Stochastic Block Model dataset with only 20 nodes for each graphs. (**Left**) Absolute loss. (**Right**) Square loss.
Cited on page [132] 132

C.8 Hyperparameters analysis on a mixture of Gaussians with 100 points sampled for each distributions. (**Left**) Absolute loss. (**Right**) Square loss.
Cited on page [132] 133

C.9 Hyperparameters analysis on a Stochastic Block Model dataset with 1000 nodes for each graphs. (**Left**) Absolute loss. (**Right**) Square loss.
Cited on page [133] 133

C.10 Similar to the Figures 4.8, 4.9 and 4.10 in Chapter 4, with SaGroW^{KL} replaced by SaGroW.
Cited on page [133] 134

D.1 (**Left**) MNIST image with a color associated with each pixel. (**Middle** and **Right**) USPS image colored by the transportation of the left image by the transport plan found with Co-OT and OTT$_{123}$ respectively. Few rows and columns are deleted in this representation as they have no mass.
Cited on page [139] 140

D.2 (**Top row**) Average target accuracy for an increasing Kullback-Leibler and classes regularization values. (**Bottom row**) Distances of the different methods for an increasing Kullback-Leibler and classes regularization values. The distances have been re-scaled between 0 and 1.
Cited on pages [150,150] 151

D.3 (**Top row**) Target accuracy for an increasing Kullback-Leibler and classes regularization values with the dataset composed of the two classes *Thriller/Crime/Drama* and *Fantasy/Sci-Fi*. (**Bottom row**) Distances of the different methods for an increasing Kullback-Leibler and classes regularization values with the dataset composed of the two classes *Thriller/Crime/Drama* and *Fantasy/Sci-Fi*. The distances have been re-scaled between 0 and 1.
. 152

D.4 (**Top row**) Target accuracy for an increasing Kullback-Leibler and classes regularization values with the dataset composed of the two classes *Children's/Animation* and *Fantasy/Sci-Fi*. (**Bottom row**) Distances of the different methods for an increasing Kullback-Leibler and classes regularization values with the dataset composed of the two classes *Children's/Animation* and *Fantasy/Sci-Fi*. The distances have been re-scaled between 0 and 1.
. 153

D.5 (**Top row**) Target accuracy for an increasing Kullback-Leibler and classes regularization values with the dataset composed of the two classes *Thriller/Crime/Drama* and *War/Western*. (**Bottom row**) Distances of the different methods for an increasing Kullback-Leibler and classes regularization values with the dataset composed of the two classes *Thriller/Crime/Drama* and *War/Western*. The distances have been re-scaled between 0 and 1.
. 154

D.6 (**Top row**) Target accuracy for an increasing Kullback-Leibler and classes regularization values with the dataset composed of the two classes *Fantasy/Sci-Fi* and *Children's/Animation*. (**Bottom row**) Distances of the different methods for an increasing Kullback-Leibler and classes regularization values with the dataset composed of the two classes *Fantasy/Sci-Fi* and *Children's/Animation*. The distances have been re-scaled between 0 and 1. 155

D.7 (**Top row**) Target accuracy for an increasing Kullback-Leibler and classes regularization values with the dataset composed of the two classes *Fantasy/Sci-Fi* and *War/Western*. (**Bottom row**) Distances of the different methods for an increasing Kullback-Leibler and classes regularization values with the dataset composed of the two classes *Fantasy/Sci-Fi* and *War/Western*. The distances have been re-scaled between 0 and 1. 156

D.8 (**Top row**) Target accuracy for an increasing Kullback-Leibler and classes regularization values with the dataset composed of the two classes *Children's/Animation* and *War/Western*. (**Bottom row**) Distances of the different methods for an increasing Kullback-Leibler and classes regularization values with the dataset composed of the two classes *Children's/Animation* and *War/Western*. The distances have been re-scaled between 0 and 1.
Cited on pages [150,150] 157

D.9 (**Top row**) Average target accuracy for an increasing number of users known and labels available in the target domain. (**Bottom row**) Distances of the different methods for an increasing number of users known and labels available in the target domain. The distances have been re-scaled between 0 and 1.
Cited on page [158] 158

D.10 (**Top row**) Target accuracy for an increasing increasing number of users known and label available in the target domain with the dataset composed of the two classes *Thriller/Crime/Drama* and *Fantasy/Sci-Fi*. (**Bottom row**) Distances of the different methods for an increasing increasing number of users known and label available in the target domain with the dataset composed of the two classes *Thriller/Crime/Drama* and *Fantasy/Sci-Fi*. The distances have been re-scaled between 0 and 1. 159

D.11 (**Top row**) Target accuracy for an increasing increasing number of users known and label available in the target domain with the dataset composed of the two classes *Children's/Animation* and *Fantasy/Sci-Fi*. (**Bottom row**) Distances of the different methods for an increasing increasing number of users known and label available in the target domain with the dataset composed of the two classes *Children's/Animation* and *Fantasy/Sci-Fi*. The distances have been re-scaled between 0 and 1. 160

D.12 (**Top row**) Target accuracy for an increasing increasing number of users known and label available in the target domain with the dataset composed of the two classes *Thriller/Crime/Drama* and *War/Western*. (**Bottom row**) Distances of the different methods for an increasing increasing number of users known and label available in the target domain with the dataset composed of the two classes *Thriller/Crime/Drama* and *War/Western*. The distances have been re-scaled between 0 and 1. 161

D.13 (**Top row**) Target accuracy for an increasing increasing number of users known and label available in the target domain with the dataset composed of the two classes *Fantasy/Sci-Fi* and *Children's/Animation*. (**Bottom row**) Distances of the different methods for an increasing increasing number of users known and label available in the target domain with the dataset composed of the two classes *Fantasy/Sci-Fi* and *Children's/Animation*. The distances have been re-scaled between 0 and 1. 162

D.14 (**Top row**) Target accuracy for an increasing increasing number of users known and label available in the target domain with the dataset composed of the two classes *Fantasy/Sci-Fi* and *War/Western*. (**Bottom row**) Distances of the different methods for an increasing increasing number of users known and label available in the target domain with the dataset composed of the two classes *Fantasy/Sci-Fi* and *War/Western*. The distances have been re-scaled between 0 and 1. 163

D.15 (**Top row**) Target accuracy for an increasing increasing number of users known and label available in the target domain with the dataset composed of the two classes *C2hildren's/Animation* and *War/Western*. (**Bottom row**) Distances of the different methods for an increasing increasing number of users known and label available in the target domain with the dataset composed of the two classes *C2hildren's/Animation* and *War/Western*. The distances have been re-scaled between 0 and 1.
Cited on page [158] 164

D.16 Average target accuracy and distance over all the dataset for an increasing number of samples for the estimation of the gradient.
Cited on pages [148,165] 164

F.1 Illustration du problème du Transport Optimal entre deux distributions discrètes.
Cited on page [175] 176

List of Tables

2.1 Accuracy of all the methods on 3 different types of features. The best method for each dataset is in bold.
Cited on pages [42,42,42] 43

2.2 Accuracy comparison on Office-Caltech (SURF features) between a cross-validation method that uses the true target labels (first line) and the cross-validation method used in this chapter that exploits pseudo-labels in the unsupervised DA setting (second line). CORAL and NA are excluded as they do not have hyperparameter.
Cited on page [43] 43

4.1 Complexity of each method with an arbitrary loss function, with S iterations, P Sinkhorn iterations, N points in the dataset and M matrix samples. Note that the complexity can be different for specific loss functions.
Cited on page [77] 78

4.2 Classification accuracy and computation time of various methods on a 11-class graph classification task. In this summary table, only the hyperparameters yielding the best classification accuracy are reported, for each considered method.
Cited on page [84] 84

5.1 Accuracy on 6 DA tasks with the hyperparameters found using the unsupervised proposed method. To evaluate the best possible performance reachable by each method, AVG^{best} displays the accuracy with the best hyperparameters using the ground truth of the target domain.
Cited on pages [96,96,96] 96

5.2 ARI for unbalanced comparison-based clustering tasks on MNIST dataset. Each line corresponds to the average over 10 different combinations of classes, each run 10 times.
Cited on page [99] 98

A.1 Accuracy of all the methods on 3 different types of features. The best method for each dataset is in bold.
Cited on pages [42,105] 106

A.2 Comparison between MLOT and ML + OTDA_p on Office-Caltech dataset with SURF features with default hyperparameters. 107
A.3 Linear and non-linear version of MLOT with Pytorch on SURF features. 107

C.1 Gromov Wasserstein estimation for different values of ϵ. The dataset is composed of 2 graphs created with a Gaussian Random Partition Graph (Brandes et al., 2003) with 50 points each. The mean cluster size is set to 25 and the variance to 5. The probability of intra-cluster connection is 0.5 while the inter-cluster is set to 0.1. The GW distances reported are averaged over 10 iterations. The absolute distance is used for $\mathcal{L}$. The number of iteration of SaGroW^{KL} is 1000 with one sample per iteration.
Cited on page [134] 135
C.2 Gromov Wasserstein distance for PoGroW with different values of α and different number of iterations. The dataset is composed of 2 graphs created with a Gaussian Random Partition Graph (Brandes et al., 2003) with 50 points. The absolute distance is used for $\mathcal{L}$.
Cited on pages [134,134] 135
C.3 Gromov Wasserstein distance for PoGroW with different values of α and different number of iterations. The dataset is composed of 2 samples of 50 points of mixtures of Gaussians. The absolute distance is used for $\mathcal{L}$.
Cited on pages [134,134] 136
C.4 Complete table of the classification experiment (1/2).
Cited on page [134] 137
C.5 Complete table of the classification experiment (2/2.
Cited on page [134] 138

D.1 Error for the gradient approximation for an increasing number of samples.
Cited on page [148] 148
D.2 ARI for unbalanced comparison-based clustering tasks on MNIST dataset. Each line corresponds to the average over 10 runs.
Cited on page [166] 167
D.3 ARI for balanced comparison-based clustering tasks on MNIST dataset. Each line corresponds to the average over 10 different combinations of classes, each run 10 times.
Cited on page [166] 168

List of Algorithms

1 Sinkhorn algorithm to solve the regularized Optimal Transport problem between 2 discrete distributions μ and ν. Cited on pages [16,17,30] 17

2 Frank-Wolfe algorithm Cited on pages [26,28,28] 27

3 MLOT Cited on page [40] 40

4 Cutting set method for RKP$(\Pi, \mathcal{C})$ with constraint elimination. Cited on pages [59,59,113] 55

5 SaGroW Cited on pages [70,70,71,72,76,121,122,124] 71

6 OTT Cited on pages [92,94,96] 92

7 Gaussians mixtures dataset. Cited on page [129] 130

Bibliography

Agrawal, A., Verschueren, R., Diamond, S., and Boyd, S. (2018). A rewriting system for convex optimization problems. *Journal of Control and Decision*, 5(1):42–60. (Cited on page 57.)

Agueh, M. and Carlier, G. (2011). Barycenters in the wasserstein space. *SIAM Journal on Mathematical Analysis*. (Cited on page 20.)

Ahuja, R. K., Magnanti, T. L., and Orlin, J. B. (1988). Network flows. (Cited on page 13.)

Aljundi, R., Emonet, R., Muselet, D., and Sebban, M. (2015). Landmarks-based kernelized subspace alignment for unsupervised domain adaptation. In *CVPR*. (Cited on page 23.)

Altschuler, J., Weed, J., and Rigollet, P. (2017). Near-linear time approximation algorithms for optimal transport via sinkhorn iteration. *arXiv preprint arXiv:1705.09634*. (Cited on pages 17 and 25.)

Alvarez-Melis, D., Jaakkola, T., and Jegelka, S. (2018). Structured optimal transport. In *AISTATS*, pages 1771–1780. (Cited on pages 47, 48, 49, and 53.)

Alvarez-Melis, D., Jegelka, S., and Jaakkola, T. S. (2019). Towards optimal transport with global invariances. In *The 22nd International Conference on Artificial Intelligence and Statistics*. (Cited on page 172.)

Arjovsky, M., Chintala, S., and Bottou, L. (2017). Wasserstein generative adversarial networks. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*. (Cited on pages 25 and 171.)

Bay, H., Tuytelaars, T., and Van Gool, L. (2006). Surf: Speeded up robust features. In *ECCV*. (Cited on page 41.)

Beck, A. and Teboulle, M. (2003). Mirror descent and nonlinear projected subgradient methods for convex optimization. *Operations Research Letters*. (Cited on pages 19, 29, 92, and 140.)

Beckman, M. and Koopmans, T. (1957). Assignment problems and the location of economic activities. *Econometrica*, 25:53–76. (Cited on page 66.)

Bellet, A., Habrard, A., and Sebban, M. (2013). A survey on metric learning for feature vectors and structured data. *arXiv preprint arXiv:1306.6709*. (Cited on page 36.)

Bellet, A., Habrard, A., and Sebban, M. (2015). *Metric Learning.* Synthesis Lectures on Artificial Intelligence and Machine Learning. Morgan & Claypool Publishers. (Cited on pages 36 and 50.)

Ben-David, S., Blitzer, J., Crammer, K., and Pereira, F. (2007). Analysis of representations for domain adaptation. In *NIPS.* (Cited on pages 39 and 96.)

Benamou, J.-D., Carlier, G., Cuturi, M., Nenna, L., and Peyré, G. (2015). Iterative bregman projections for regularized transportation problems. *SIAM Journal on Scientific Computing.* (Cited on page 18.)

Berge, C. (1984). *Hypergraphs: combinatorics of finite sets.* (Cited on pages 88 and 92.)

Bertsimas, D. and Tsitsiklis, J. N. (1997). *Introduction to linear optimization.* (Cited on page 12.)

Bhushan Damodaran, B., Kellenberger, B., Flamary, R., Tuia, D., and Courty, N. (2018). Deepjdot: Deep joint distribution optimal transport for unsupervised domain adaptation. In *ECCV.* (Cited on pages 23 and 34.)

Birkhoff, G. (1946). Three observations on linear algebra. *Univ. Nac. Tacuman, Rev. Ser. A.* (Cited on page 12.)

Blondel, M., Seguy, V., and Rolet, A. (2018). Smooth and sparse optimal transport. In *International Conference on Artificial Intelligence and Statistics.* (Cited on pages 36 and 122.)

Blumberg, A. J., Carriere, M., Mandell, M. A., Rabadan, R., and Villar, S. (2020). Mrec: a fast and versatile framework for aligning and matching point clouds with applications to single cell molecular data. *arXiv preprint arXiv:2001.01666.* (Cited on page 66.)

Bolley, F., Guillin, A., and Villani, C. (2007). Quantitative concentration inequalities for empirical measures on non-compact spaces. *PTRF.* (Cited on page 40.)

Bonneel, N., Van De Panne, M., Paris, S., and Heidrich, W. (2011). Displacement interpolation using lagrangian mass transport. In *Proceedings of the 2011 SIGGRAPH Asia Conference.* (Cited on pages 13 and 78.)

Boyd, S. P. and Vandenberghe, L. (2004). *Convex optimization.* Cambridge University Press. (Cited on page 112.)

Brandes, U., Gaertler, M., and Wagner, D. (2003). Experiments on graph clustering algorithms. In *European Symposium on Algorithms.* (Cited on pages 77, 80, 81, 129, 130, 135, 188, 190, and 196.)

Bredies, K., Lorenz, D. A., and Maass, P. (2009). A generalized conditional gradient method and its connection to an iterative shrinkage method. *Computational Optimization and Applications.* (Cited on page 36.)

Bregman, L. M. (1967). The relaxation method of finding the common point of convex sets and its application to the solution of problems in convex programming. *USSR computational mathematics and mathematical physics*. (Cited on pages 29 and 30.)

Brenier, Y. (1991). Polar factorization and monotone rearrangement of vector-valued functions. *Communications on pure and applied mathematics*. (Cited on page 11.)

Bronstein, A. M., Bronstein, M. M., Kimmel, R., Mahmoudi, M., and Sapiro, G. (2010). A gromov-hausdorff framework with diffusion geometry for topologically-robust non-rigid shape matching. *International Journal of Computer Vision*. (Cited on page 20.)

Brualdi, R. A. (2006). *Combinatorial matrix classes*. (Cited on page 11.)

Bunne, C., Alvarez-Melis, D., Krause, A., and Jegelka, S. (2019). Learning generative models across incomparable spaces. In *International Conference on Machine Learning*. (Cited on page 20.)

Burago, D., Burago, I. D., Burago, Y., Ivanov, S., Ivanov, S. V., and Ivanov, S. A. (2001). *A course in metric geometry*. (Cited on page 18.)

Caracciolo, S., D'Achille, M. P., Erba, V., and Sportiello, A. (2020). The dyck bound in the concave 1-dimensional random assignment model. *Journal of Physics A: Mathematical and Theoretical*. (Cited on pages 14 and 136.)

Carlier, G. (2003). On a class of multidimensional optimal transportation problems. *Journal of convex analysis*. (Cited on page 89.)

Caron, M., Misra, I., Mairal, J., Goyal, P., Bojanowski, P., and Joulin, A. (2020). Unsupervised learning of visual features by contrasting cluster assignments. In *Thirty-fourth Conference on Neural Information Processing Systems (NeurIPS)*. (Cited on page 18.)

Chowdhury, S. and Mémoli, F. (2019). The gromov–wasserstein distance between networks and stable network invariants. *Information and Inference: A Journal of the IMA*. (Cited on pages 18, 142, and 144.)

Chowdhury, S., Miller, D., and Needham, T. (2021). Quantized gromov-wasserstein. *arXiv preprint arXiv:2104.02013*. (Cited on pages 66, 103, and 183.)

Cortes, C. and Vapnik, V. (1995). Support-vector networks. *Machine learning*. (Cited on pages 23, 95, and 186.)

Courty, N., Flamary, R., Habrard, A., and Rakotomamonjy, A. (2017a). Joint distribution optimal transportation for domain adaptation. In *NIPS*. (Cited on pages 23, 34, and 42.)

Courty, N., Flamary, R., Tuia, D., and Rakotomamonjy, A. (2017b). Optimal transport for domain adaptation. *PAMI*. (Cited on pages 18, 23, 28, 29, 34, 35, 36, 40, 42, 94, 96, 149, and 150.)

Cover, T. and Hart, P. (1967). Nearest neighbor pattern classification. *IEEE Transactions Information Theory*, 13(1):21–27. (Cited on page 22.)

Cuturi, M. (2013). Sinkhorn distances: Lightspeed computation of optimal transport. In *Advances in neural information processing systems*. (Cited on pages 14, 15, 18, 25, 31, 42, 54, 59, 70, 76, 78, 92, 93, and 96.)

Cuturi, M. and Avis, D. (2014). Ground metric learning. *The Journal of Machine Learning Research*. (Cited on pages 35, 36, and 56.)

Cuturi, M. and Doucet, A. (2014). Fast computation of wasserstein barycenters. In *International conference on machine learning*. (Cited on page 21.)

Cuturi, M., Teboul, O., and Vert, J.-P. (2019). Differentiable ranking and sorting using optimal transport. In *Advances in Neural Information Processing Systems*. (Cited on page 76.)

Davis, J. V., Kulis, B., Jain, P., Sra, S., and Dhillon, I. S. (2007). Information-theoretic metric learning. In *ICML*. (Cited on page 36.)

De Lathauwer, L., De Moor, B., and Vandewalle, J. (2000). A multilinear singular value decomposition. *SIAM journal on Matrix Analysis and Applications*. (Cited on page 94.)

Delon, J., Salomon, J., and Sobolevski, A. (2012). Local matching indicators for transport problems with concave costs. *SIAM Journal on Discrete Mathematics*. (Cited on page 65.)

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., and Fei-Fei, L. (2009). Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*. (Cited on pages 102 and 182.)

Deshpande, I., Hu, Y., Sun, R., Pyrros, A., Siddiqui, N., Koyejo, S., Zhao, Z., Forsyth, D. A., and Schwing, A. G. (2019). Max-sliced wasserstein distance and its use for gans. *CoRR*. (Cited on pages 35 and 36.)

Deshpande, I., Zhang, Z., and Schwing, A. G. (2018). Generative modeling using the sliced wasserstein distance. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. (Cited on page 169.)

Dhouib, S. (2020). *Contributions to unsupervised domain adaptation: Similarity functions, optimal transport and theoretical guarantees*. PhD thesis, Université de Lyon. (Cited on page 47.)

Dhouib, S., Redko, I., Kerdoncuff, T., Emonet, R., and Sebban, M. (2020). A swiss army knife for minimax optimal transport. In *International Conference on Machine Learning*. (Cited on pages 5 and 47.)

Diamond, S. and Boyd, S. (2016). CVXPY: A Python-embedded modeling language for convex optimization. *Journal of Machine Learning Research*, 17(83):1–5. (Cited on page 57.)

Donahue, J., Jia, Y., Vinyals, O., Hoffman, J., Zhang, N., Tzeng, E., and Darrell, T. (2014). Decaf: A deep convolutional activation feature for generic visual recognition. In *ICML*. (Cited on page 41.)

Dowson, D. and Landau, B. (1982). The fréchet distance between multivariate normal distributions. *Journal of multivariate analysis*. (Cited on page 20.)

Duchenne, O., Bach, F., Kweon, I.-S., and Ponce, J. (2011). A tensor-based algorithm for high-order graph matching. *IEEE transactions on pattern analysis and machine intelligence*. (Cited on page 88.)

Dvurechensky, P., Gasnikov, A., and Kroshnin, A. (2018). Computational optimal transport: Complexity by accelerated gradient descent is better than by sinkhorn's algorithm. In *International conference on machine learning*. (Cited on page 17.)

Emamjomeh-Zadeh, E. and Kempe, D. (2018). Adaptive hierarchical clustering using ordinal queries. In *Symposium on Discrete Algorithms*. (Cited on pages 88 and 97.)

Ezuz, D., Solomon, J., Kim, V. G., and Ben-Chen, M. (2017). Gwcnn: A metric alignment layer for deep shape analysis. In *Computer Graphics Forum*. (Cited on page 20.)

Fernando, B., Habrard, A., Sebban, M., and Tuytelaars, T. (2013). Unsupervised visual domain adaptation using subspace alignment. In *ICCV*. (Cited on pages 23, 34, and 42.)

Ferradans, S., Papadakis, N., Peyré, G., and Aujol, J.-F. (2014). Regularized Discrete Optimal Transport. *SIAM Journal on Imaging Sciences*, 7(3):1853–1882. (Cited on page 62.)

Flamary, R., Courty, N., Gramfort, A., Alaya, M. Z., Boisbunon, A., Chambon, S., Chapel, L., Corenflos, A., Fatras, K., Fournier, N., Gautheron, L., Gayraud, N. T., Janati, H., Rakotomamonjy, A., Redko, I., Rolet, A., Schutz, A., Seguy, V., Sutherland, D. J., Tavenard, R., Tong, A., and Vayer, T. (2021). Pot: Python optimal transport. *Journal of Machine Learning Research*, 22(78):1–8. (Cited on pages 13 and 78.)

Forrow, A., Hütter, J.-C., Nitzan, M., Rigollet, P., Schiebinger, G., and Weed, J. (2019). Statistical optimal transport via factored couplings. In *AISTATS*, pages 2454–2465. (Cited on page 59.)

Frank, M., Wolfe, P., et al. (1956). An algorithm for quadratic programming. *Naval research logistics quarterly*. (Cited on pages 19, 26, 57, and 73.)

Freund, R. M. and Grigas, P. (2016). New analysis and results for the frank–wolfe method. *Mathematical Programming*. (Cited on page 28.)

Friedland, S. (2020). Tensor optimal transport, distance between sets of measures and tensor scaling. *arXiv preprint arXiv:2005.00945*. (Cited on page 89.)

Friedman, J., Hastie, T., Tibshirani, R., et al. (2001). *The elements of statistical learning.* (Cited on page 88.)

Frogner, C., Zhang, C., Mobahi, H., Araya-Polo, M., and Poggio, T. (2015). Learning with a wasserstein loss. *arXiv preprint arXiv:1506.05439.* (Cited on pages 18 and 25.)

Ganin, Y., Ustinova, E., Ajakan, H., Germain, P., Larochelle, H., Laviolette, F., Marchand, M., and Lempitsky, V. (2016). Domain-adversarial training of neural networks. *The journal of machine learning research.* (Cited on page 23.)

Gelfand, N., Mitra, N. J., Guibas, L. J., and Pottmann, H. (2005). Robust global registration. In *Symposium on geometry processing.* (Cited on page 73.)

Genevay, A., Chizat, L., Bach, F., Cuturi, M., and Peyré, G. (2019). Sample complexity of sinkhorn divergences. In *The 22nd International Conference on Artificial Intelligence and Statistics.* (Cited on page 122.)

Genevay, A., Peyre, G., and Cuturi, M. (2018). Learning generative models with sinkhorn divergences. In *AISTATS*, pages 1608–1617. (Cited on pages 18, 35, 36, 48, and 49.)

Ghoshdastidar, D., Perrot, M., and von Luxburg, U. (2019). Foundations of comparison-based hierarchical clustering. In *Advances in Neural Information Processing Systems.* (Cited on pages 88, 92, and 97.)

Gong, B., Shi, Y., Sha, F., and Grauman, K. (2012). Geodesic flow kernel for unsupervised domain adaptation. In *CVPR.* (Cited on page 41.)

Goodfellow, I. J., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. (2014). Generative adversarial nets. In *NIPS*, pages 2672–2680. (Cited on pages 25 and 171.)

Halko, N., Martinsson, P.-G., and Tropp, J. A. (2011). Finding structure with randomness: Probabilistic algorithms for constructing approximate matrix decompositions. *SIAM review.* (Cited on page 42.)

Hanzely, F. and Richtárik, P. (2021). Fastest rates for stochastic mirror descent methods. *Computational Optimization and Applications.* (Cited on page 76.)

Harper, F. M. and Konstan, J. A. (2015). The movielens datasets: History and context. *Acm transactions on interactive intelligent systems (tiis).* (Cited on page 95.)

Holland, P. W., Laskey, K. B., and Leinhardt, S. (1983). Stochastic blockmodels: First steps. *Social networks.* (Cited on page 129.)

Hubert, L. and Arabie, P. (1985). Comparing partitions. *Journal of classification.* (Cited on page 99.)

Jaggi, M. (2013). Revisiting frank-wolfe: Projection-free sparse convex optimization. In *International Conference on Machine Learning*. (Cited on pages 26 and 186.)

Jaynes, E. T. (1957). Information theory and statistical mechanics. *Physical review*. (Cited on page 14.)

Kantorovich, L. (1942). On the translocation of masses. *Doklady of the Academy of Sciences of the USSR*, 37:199–201. (Cited on pages 1, 10, and 176.)

Kerdoncuff, Tanguy Michaëel, P., Emonet, R., and Sebban, M. (2022). Optimal tensor transport. In *AAAI*. (Cited on pages 5 and 87.)

Kerdoncuff, T., Emonet, R., and Sebban, M. (2020). Metric learning in optimal transport for domain adaptation. In Bessiere, C., editor, *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI-20*, pages 2162–2168. International Joint Conferences on Artificial Intelligence Organization. Main track. (Cited on pages 5 and 33.)

Kerdoncuff, T., Emonet, R., and Sebban, M. (2021). Sampled gromov wasserstein. *Machine Learning*. (Cited on pages 5, 65, and 74.)

Kerdreux, T., d'Aspremont, A., and Pokutta, S. (2019). Restarting frank-wolfe. In *The 22nd International Conference on Artificial Intelligence and Statistics*. (Cited on page 26.)

Kersting, K., Kriege, N. M., Morris, C., Mutzel, P., and Neumann, M. (2016). Benchmark data sets for graph kernels. (Cited on page 83.)

Kingma, D. P. and Ba, J. (2014). Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*. (Cited on page 105.)

Kingma, D. P. and Welling, M. (2013). Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*. (Cited on page 25.)

Kolouri, S., Nadjahi, K., Simsekli, U., Badeau, R., and Rohde, G. K. (2019). Generalized sliced wasserstein distances. *arXiv preprint arXiv:1902.00434*. (Cited on pages 169 and 170.)

Kuchment, P. (2006). Generalized transforms of radon type and their applications. In *Proceedings of Symposia in Applied Mathematics*. (Cited on page 169.)

Kuhn, D., Esfahani, P. M., Nguyen, V. A., and Shafieezadeh-Abadeh, S. (2019). Wasserstein distributionally robust optimization: Theory and applications in machine learning. *CoRR*, abs/1908.08729. (Cited on page 49.)

Kullback, S. and Leibler, R. A. (1951). On information and sufficiency. *The annals of mathematical statistics*. (Cited on page 15.)

Lacoste-Julien, S. (2016). Convergence rate of frank-wolfe for non-convex objectives. *arXiv preprint arXiv:1607.00345*. (Cited on pages 27 and 28.)

Lai, Z., Xu, Y., Yang, J., Tang, J., and Zhang, D. (2013). Sparse tensor discriminant analysis. *IEEE transactions on Image processing.* (Cited on page 94.)

Laschos, V., Tinapp, J., and Obermayer, K. (2019). Training generative networks with general optimal transport distances. *arXiv preprint arXiv:1910.00535.* (Cited on page 172.)

Le, K., Le, D., Nguyen, H., Do, D., Pham, T., and Ho, N. (2021). Entropic gromov-wasserstein between gaussian distributions. *arXiv preprint arXiv:2108.10961.* (Cited on page 74.)

LeCun, Y., Bottou, L., Bengio, Y., and Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE.* (Cited on pages 88 and 98.)

Li, R., Ye, X., Zhou, H., and Zha, H. (2019). Learning to match via inverse optimal transport. *J. Mach. Learn. Res.*, 20:80:1–80:37. (Cited on page 49.)

Lin, T., Ho, N., and Jordan, M. (2019). On efficient optimal transport: An analysis of greedy and accelerated mirror descent algorithms. In *International Conference on Machine Learning.* (Cited on page 17.)

Liu, H., Gu, X., and Samaras, D. (2019). Wasserstein gan with quadratic transport cost. In *Proceedings of the IEEE/CVF International Conference on Computer Vision.* (Cited on page 172.)

Liu, Y., Liu, Y., and Chan, K. C. (2010). Tensor distance based multilinear locality-preserved maximum information embedding. *IEEE Transactions on neural networks.* (Cited on page 94.)

Lloyd, S. (1982). Least squares quantization in pcm. *IEEE transactions on information theory.* (Cited on page 98.)

Loiola, E. M., de Abreu, N. M. M., Boaventura-Netto, P. O., Hahn, P., and Querido, T. (2007). A survey for the quadratic assignment problem. *European journal of operational research.* (Cited on page 19.)

Luise, G., Salzo, S., Pontil, M., and Ciliberto, C. (2019). Sinkhorn barycenters with free support via frank-wolfe algorithm. *Advances in Neural Information Processing Systems.* (Cited on page 28.)

Magnus, J. R. (1987). A representation theorem for (trAp)1/p. Other publications TiSEM, Tilburg University, School of Economics and Management. (Cited on pages 51 and 110.)

Mahalanobis, P. C. (1936). On the generalized distance in statistics. (Cited on page 30.)

Mansour, Y., Mohri, M., and Rostamizadeh, A. (2009). Domain adaptation: Learning bounds and algorithms. *arXiv:0902.3430.* (Cited on page 39.)

McInnes, L., Healy, J., and Melville, J. (2018). Umap: Uniform manifold approximation and projection for dimension reduction. *arXiv preprint arXiv:1802.03426.* (Cited on page 60.)

Memoli, F. (2007). On the use of Gromov-Hausdorff Distances for Shape Comparison. In Botsch, M., Pajarola, R., Chen, B., and Zwicker, M., editors, *Eurographics Symposium on Point-Based Graphics*. The Eurographics Association. (Cited on pages 18, 66, and 90.)

Mémoli, F. (2009). Spectral gromov-wasserstein distances for shape matching. In *2009 IEEE 12th International Conference on Computer Vision Workshops, ICCV Workshops*. (Cited on page 67.)

Mémoli, F. (2011). Gromov–wasserstein distances and the metric approach to object matching. *Foundations of computational mathematics*. (Cited on pages 18, 20, 66, 67, and 83.)

Mérigot, Q. (2011). A multiscale approach to optimal transport. In *Computer Graphics Forum*. (Cited on page 170.)

Mirza, M. and Osindero, S. (2014). Conditional generative adversarial nets. *arXiv preprint arXiv:1411.1784*. (Cited on page 174.)

Moameni, A. (2014). Multi-marginal monge–kantorovich transport problems: A characterization of solutions. *Comptes Rendus Mathematique*. (Cited on page 89.)

Monge, G. (1781). Mémoire sur la théorie des déblais et des remblais. *Histoire de l'Académie royale des sciences de Paris*. (Cited on pages 1, 11, and 176.)

Mutapcic, A. and Boyd, S. P. (2009). Cutting-set methods for robust convex optimization with pessimizing oracles. *Optimization Methods and Software*. (Cited on pages 53, 55, 56, and 113.)

Nadjahi, K., Durmus, A., Simsekli, U., and Badeau, R. (2019). Asymptotic guarantees for learning generative models with the sliced-wasserstein distance. *Advances in Neural Information Processing Systems*. (Cited on page 169.)

Nemirovskij, A. S. and Yudin, D. B. (1983). Problem complexity and method efficiency in optimization. (Cited on page 29.)

Neumann, M., Moreno, P., Antanas, L., Garnett, R., and Kersting, K. (2013). Graph kernels for object category prediction in task-dependent robot grasping. In *Online Proceedings of the Eleventh Workshop on Mining and Learning with Graphs*. (Cited on page 83.)

Nguyen, K., Ho, N., Pham, T., and Bui, H. (2020). Distributional sliced-wasserstein and applications to generative modeling. In *International Conference on Learning Representations*. (Cited on page 169.)

Pan, S. J., Tsang, I. W., Kwok, J. T., and Yang, Q. (2011). Domain adaptation via transfer component analysis. *IEEE Transactions on Neural Networks*. (Cited on pages 23, 34, and 42.)

Pass, B. (2015). Multi-marginal optimal transport: theory and applications. *ESAIM: Mathematical Modelling and Numerical Analysis*. (Cited on page 89.)

Paszke, A., Gross, S., Chintala, S., Chanan, G., Yang, E., DeVito, Z., Lin, Z., Desmaison, A., Antiga, L., and Lerer, A. (2017). Automatic differentiation in pytorch. (Cited on page 105.)

Paty, F. and Cuturi, M. (2019a). Subspace robust wasserstein distances. In *ICML*, pages 5072–5081. (Cited on pages 28, 35, 47, 48, 49, 52, 53, 58, 59, 60, 61, 65, and 187.)

Paty, F.-P. and Cuturi, M. (2019b). Subspace robust wasserstein distances. In *International Conference on Machine Learning*. (Cited on page 36.)

Perrot, M., Esser, P. M., and Ghoshdastidar, D. (2020). Near-optimal comparison based clustering. *arXiv preprint arXiv:2010.03918*. (Cited on pages 88, 97, and 98.)

Peyré, G., Cuturi, M., et al. (2019). Computational optimal transport. *Foundations and Trends® in Machine Learning*. (Cited on pages 12, 13, and 17.)

Peyré, G., Cuturi, M., and Solomon, J. (2016). Gromov-wasserstein averaging of kernel and distance matrices. In *International Conference on Machine Learning*. (Cited on pages 19, 25, 29, 31, 32, 66, 67, 68, 69, 71, 72, 76, 78, 92, 93, 94, 95, and 140.)

Pitié, F., Kokaram, A. C., and Dahyot, R. (2007). Automated colour grading using colour distribution transfer. *Computer Vision and Image Understanding*. (Cited on page 169.)

Rabin, J., Ferradans, S., and Papadakis, N. (2014). Adaptive color transfer with relaxed optimal transport. In *2014 IEEE International Conference on Image Processing (ICIP)*. (Cited on page 25.)

Rabin, J. and Peyré, G. (2011). Wasserstein regularization of imaging problem. In *2011 18th IEEE International Conference on Image Processing*. (Cited on pages 21, 66, 68, and 169.)

Rangarajan, A., Yuille, A., and Mjolsness, E. (1999). Convergence properties of the softassign quadratic assignment algorithm. *Neural Computation*. (Cited on pages 19 and 71.)

Reddi, S. J., Sra, S., Póczos, B., and Smola, A. (2016). Stochastic frank-wolfe methods for nonconvex optimization. In *2016 54th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*. (Cited on pages 70, 71, 72, 73, 121, 122, 124, and 127.)

Redko, I., Courty, N., Flamary, R., and Tuia, D. (2019a). Optimal transport for multi-source domain adaptation under target shift. In *The 22nd International Conference on Artificial Intelligence and Statistics*. (Cited on page 95.)

Redko, I., Habrard, A., and Sebban, M. (2017). Theoretical analysis of domain adaptation with optimal transport. In *ECML PKDD*. (Cited on page 39.)

Redko, I., Habrard, A., and Sebban, M. (2019b). On the analysis of adaptability in multi-source domain adaptation. *Machine Learning*. (Cited on page 40.)

Redko, I., Morvant, E., Habrard, A., Sebban, M., and Bennani, Y. (2019c). *Advances in Domain Adaptation Theory*. Elsevier. (Cited on pages 34 and 96.)

Redko, I., Vayer, T., Flamary, R., and Courty, N. (2020). Co-optimal transport. In *NeurIPS 2020-Thirty-four Conference on Neural Information Processing Systems*. (Cited on pages 29, 71, 72, 87, 89, 94, 139, 142, and 145.)

Rubner, Y., Guibas, L. J., and Tomasi, C. (1997). The earth mover's distance, multi-dimensional scaling, and color-based image retrieval. In *Proceedings of the ARPA image understanding workshop*. (Cited on page 13.)

Saenko, K., Kulis, B., Fritz, M., and Darrell, T. (2010). Adapting visual category models to new domains. In *ECCV*. (Cited on page 41.)

Salmona, A., Delon, J., and Desolneux, A. (2021). Gromov-wasserstein distances between gaussian distributions. *arXiv preprint arXiv:2104.07970*. (Cited on pages 74 and 85.)

Sato, R., Cuturi, M., Yamada, M., and Kashima, H. (2020). Fast and robust comparison of probability measures in heterogeneous spaces. *arXiv preprint arXiv:2002.01615*. (Cited on pages 66, 73, 103, and 183.)

Schmitzer, B. (2019). Stabilized sparse scaling algorithms for entropy regularized transport problems. *SIAM Journal on Scientific Computing*. (Cited on pages 17 and 25.)

Shen, J., Qu, Y., Zhang, W., and Yu, Y. (2017). Wasserstein distance guided representation learning for domain adaptation. *arXiv preprint arXiv:1707.01217*. (Cited on page 39.)

Shepard, R. N. (1962). The analysis of proximities: Multidimensional scaling with an unknown distance function. i. *Psychometrika*. (Cited on page 97.)

Shirdhonkar, S. and Jacobs, D. W. (2008). Approximate earth mover's distance in linear time. In *2008 IEEE Conference on Computer Vision and Pattern Recognition*. (Cited on page 13.)

Sinkhorn, R. and Knopp, P. (1967). Concerning nonnegative matrices and doubly stochastic matrices. *Pacific Journal of Mathematics*. (Cited on pages 16 and 17.)

Sion, M. (1958). On general minimax theorems. *Pacific J. Math.*, 8(1):171–176. (Cited on pages 54 and 111.)

Solomon, J., De Goes, F., Peyré, G., Cuturi, M., Butscher, A., Nguyen, A., Du, T., and Guibas, L. (2015). Convolutional wasserstein distances: Efficient optimal transportation on geometric domains. *ACM Transactions on Graphics (TOG)*. (Cited on pages 18 and 21.)

Solomon, J., Peyré, G., Kim, V. G., and Sra, S. (2016). Entropic metric alignment for correspondence problems. *ACM Transactions on Graphics (TOG)*. (Cited on page 68.)

Stewart, N., Brown, G. D. A., and Chater, N. (2005). Absolute identification by relative judgment. *Psychological review*. (Cited on page 98.)

Sun, B., Feng, J., and Saenko, K. (2016a). Return of frustratingly easy domain adaptation. In *AAAI*. (Cited on pages 23, 34, and 42.)

Sun, B. and Saenko, K. (2016). Deep coral: Correlation alignment for deep domain adaptation. In *ECCV*. (Cited on pages 23 and 34.)

Sun, Y., Babu, P., and Palomar, D. P. (2016b). Majorization-minimization algorithms in signal processing, communications, and machine learning. *IEEE Transactions on Signal Processing*. (Cited on page 128.)

Tarjan, R. E. (1997). Dynamic trees as search trees via euler tours, applied to the network simplex algorithm. *Mathematical Programming*. (Cited on page 13.)

Tolstikhin, I., Bousquet, O., Gelly, S., and Schoelkopf, B. (2017). Wasserstein auto-encoders. *arXiv preprint arXiv:1711.01558*. (Cited on pages 18 and 25.)

Torrey, L. and Shavlik, J. (2010). Transfer learning. In *Handbook of research on machine learning applications and trends: algorithms, methods, and techniques*. (Cited on page 22.)

Ukkonen, A. (2017). Crowdsourced correlation clustering with relative distance comparisons. *arXiv preprint arXiv:1709.08459*. (Cited on pages 88 and 97.)

Van Der Maaten, L. and Weinberger, K. (2012). Stochastic triplet embedding. In *2012 IEEE International Workshop on Machine Learning for Signal Processing*. (Cited on page 165.)

Vayer, T. (2020). *A contribution to Optimal Transport on incomparable spaces*. PhD thesis, Lorient. (Cited on pages 29, 74, 85, and 172.)

Vayer, T., Chapel, L., Flamary, R., Tavenard, R., and Courty, N. (2018). Fused gromov-wasserstein distance for structured objects: theoretical foundations and mathematical properties. *arXiv preprint arXiv:1811.02834*. (Cited on pages 20, 28, 99, 103, 174, and 184.)

Vayer, T., Chapel, L., Flamary, R., Tavenard, R., and Courty, N. (2019a). Optimal transport for structured data with application on graphs. In *ICML 2019-36th International Conference on Machine Learning*. (Cited on pages 20, 28, and 174.)

Vayer, T., Flamary, R., Tavenard, R., Chapel, L., and Courty, N. (2019b). Sliced gromov-wasserstein. In *NeurIPS 2019-Thirty-third Conference on Neural Information Processing Systems*. (Cited on pages 20, 66, 68, and 74.)

Vikram, S. and Dasgupta, S. (2016). Interactive bayesian hierarchical clustering. In *International Conference on Machine Learning*. (Cited on pages 88 and 97.)

Villani, C. (2008). *Optimal transport: old and new.* (Cited on pages 10, 89, 142, 145, 171, and 172.)

Wang, J., Feng, W., Chen, Y., Yu, H., Huang, M., and Yu, P. S. (2018). Visual domain adaptation with manifold embedded distribution alignment. In *ACM-MM.* (Cited on pages 23 and 34.)

Wang, M. and Deng, W. (2018). Deep visual domain adaptation: A survey. *Neurocomputing.* (Cited on page 34.)

Weinberger, K. Q. and Saul, L. K. (2009). Distance metric learning for large margin nearest neighbor classification. *JMLR.* (Cited on pages 36 and 42.)

Wold, S., Esbensen, K., and Geladi, P. (1987). Principal component analysis. *Chemometrics and intelligent laboratory systems.* (Cited on page 35.)

Wright, M. H. (1996). Direct search methods: Once scorned, now respectable. *Pitman Research Notes in Mathematics Series.* (Cited on page 78.)

Xie, Y., Wang, X., Wang, R., and Zha, H. (2020). A fast proximal point method for computing exact wasserstein distance. In *Uncertainty in Artificial Intelligence.* (Cited on pages 30, 31, 76, 92, and 134.)

Xu, H., Luo, D., and Carin, L. (2019a). Scalable gromov-wasserstein learning for graph partitioning and matching. In *Advances in neural information processing systems.* (Cited on pages 20, 66, 69, 78, and 95.)

Xu, H., Luo, D., Zha, H., and Duke, L. C. (2019b). Gromov-wasserstein learning for graph matching and node embedding. In *International Conference on Machine Learning.* (Cited on pages 20, 31, 71, 76, 78, 92, 93, 128, 134, and 140.)

Yan, Y., Li, W., Wu, H., Min, H., Tan, M., and Wu, Q. (2018). Semi-supervised optimal transport for heterogeneous domain adaptation. In *IJCAI.* (Cited on page 20.)

Young, F. W. (1987). *Multidimensional scaling: History, theory, and applications.* (Cited on page 98.)

Yuan, M. and Lin, Y. (2006). Model selection and estimation in regression with grouped variables. *Journal of the Royal Statistical Society: Series B (Statistical Methodology).* (Cited on page 36.)

Zass, R. and Shashua, A. (2008). Probabilistic graph and hypergraph matching. In *2008 IEEE Conference on Computer Vision and Pattern Recognition.* (Cited on page 88.)

Zhang, S. and He, N. (2018). On the convergence rate of stochastic mirror descent for nonsmooth nonconvex optimization. *arXiv preprint arXiv:1806.04781.* (Cited on page 76.)

Zhao, P. and Zhou, Z. (2018). Label distribution learning by optimal transport. In *AAAI*, pages 4506–4513. (Cited on page 56.)

Zhong, E., Fan, W., Yang, Q., Verscheure, O., and Ren, J. (2010). Cross validation framework to choose amongst models and datasets for transfer learning. In *ECML PKDD*. (Cited on pages 23, 42, and 43.)

Zhou, Z., Mertikopoulos, P., Bambos, N., Boyd, S., and Glynn, P. W. (2017). Stochastic mirror descent in variationally coherent optimization problems. *Advances in Neural Information Processing Systems*. (Cited on page 76.)

Abstract The Optimal Transport theory not only defines a notion of distance between probability measures, but can also align two distributions. During the past few years, Optimal Transport found many applications in Machine Learning, such as the approximation of a distribution in GANs for the generation of new points or the adaptation of labeled source data to unlabeled target examples to solve transfer learning tasks. Given a ground metric that allows to compare two points of a vector space, the *transport* between distributions is said to be *optimal* when it minimizes the global cost for moving one distribution to another. Although often difficult to compute, the corresponding Wasserstein distance intuitively generalizes the usual metrics between points on a vector space to the space of probability measures. In Machine Learning, the Euclidean distance is the most used ground metric despite the wide variety of possible candidates. In this thesis, we study the interest of learning more complex metrics to solve Machine Learning problems with two distinct ideas. The first one uses additional label information to improve the transportation for a classification task, while the second one proposes to choose the most stable metric in a set of functions to perform the Optimal Transport between two distributions. Based on a sampling strategy, we also propose an efficient algorithm for the Gromov Wasserstein problem, an extension of Optimal Transport which handles similarity matrices computed in incomparable vector spaces. Finally, based on this latter, we present a generalization of the Optimal Transport problem that defines a distance between tensors of arbitrary dimension.

Résumé La théorie du Transport Optimal permet non seulement de définir une notion de distance entre distributions de probabilité, mais propose aussi une correspondance entre celles-ci sous la forme d'un plan de transport. Cette théorie a été à la base de nombreux récents travaux en Apprentissage Machine, notamment dans les GANs pour l'approximation de distributions à des fins de génération de nouveaux exemples ou en adaptation de domaine. Étant donnée une métrique permettant de comparer deux points d'un espace vectoriel, le *transport* entre distributions est dit *optimal* si il minimise le coût global pour déplacer une distribution vers une autre. Bien que souvent difficile à calculer, la distance obtenue de Wasserstein généralise de manière intuitive les métriques usuelles entre points d'un espace vectoriel. En Apprentissage Machine, la distance Euclidienne est souvent exploitée par défaut comme métrique de base malgré la grande variété d'autres fonctions candidates possibles. Dans cette thèse, nous abordons tout d'abord l'intérêt d'apprendre des métriques plus complexes pour résoudre des problèmes d'Apprentissage Machine. La première contribution utilise des informations additionnelles d'étiquettes pour optimiser une métrique de Mahalanobis, tandis que la seconde propose de choisir la métrique la plus stable dans un ensemble de fonction candidates pour effectuer le Transport Optimal entre deux distributions. Nous proposons également un algorithme efficace pour résoudre le problème difficile de Gromov Wasserstein, une extension du Transport Optimal permettant de comparer des matrices de similarité venant d'espaces vectoriels différents. Enfin, en s'appuyant sur ce nouvel algorithme, nous présentons une nouvelle extension du problème du Transport Optimal qui définit une distance entre tenseurs de dimension quelconque.