Eulerian methods for inverse problems using optimal transport

Matthieu Heitz

N° d'ordre NNT : 2020LYSE1054

THÈSE DE DOCTORAT DE L'UNIVERSITÉ DE LYON
opérée au sein de
l'Université Claude Bernard Lyon 1

École Doctorale 512
InfoMaths

Spécialité de doctorat : Informatique

Soutenue publiquement le 18 mai 2020, par :
Matthieu Heitz

Eulerian methods for inverse problems using optimal transport

---

Devant le jury composé de :

M. Courty Nicolas	Pr. des Universités	Université Bretagne-Sud	Rapporteur
M. Mérigot Quentin	Professeur	Université Paris-Sud	Rapporteur
Mme. Chaine Raphaëlle	Pr. des Universités	Université Lyon 1	Examinatrice
Mme. Delon Julie	Pr. des Universités	Université Paris Descartes	Examinatrice
M. Bonneel Nicolas	Chargé de Recherche CNRS	Université Lyon 1	Co-directeur de thèse
M. Coeurjolly David	Directeur de Recherche CNRS	Université Lyon 1	Directeur de thèse
M. Cuturi Marco	Professeur	ENSAE / Google Brain Paris	Invité
M. Peyré Gabriel	Directeur de Recherche CNRS	Ecole Normale Supérieure	Invité

UNIVERSITÉ CLAUDE BERNARD - LYON 1

Président de l'Université	M. Frédéric FLEURY
Président du Conseil Académique	M. Hamda BEN HADID
Vice-président du Conseil d'Administration	M. Didier REVEL
Vice-président du Conseil des Etudes et de la Vie Universitaire	M. Philippe CHEVALIER
Vice-président de la Commission de Recherche	
Directrice Générale des Services	M. Damien VERHAEGHE

COMPOSANTES SANTÉ

Faculté de Médecine Lyon-Est - Claude Bernard	Doyen : M. Gilles RODE
Faculté de Médecine et Maïeutique Lyon Sud Charles. Mérieux	Doyenne : Mme Carole BURILLON
UFR d'Odontologie	Doyenne : Mme Dominique SEUX
Institut des Sciences Pharmaceutiques et Biologiques	Directrice : Mme Christine VINCIGUERRA
Institut des Sciences et Techniques de la Réadaptation	Directeur : M. Xavier PERROT
Département de Formation et Centre de Recherche en Biologie Humaine	Directrice : Mme Anne-Marie SCHOTT

COMPOSANTES ET DEPARTEMENTS DE SCIENCES ET TECHNOLOGIE

UFR Biosciences	Directrice : Mme Kathrin GIESELER
Département Génie Electrique et des Procédés (GEP)	Directrice : Mme Rosaria FERRIGNO
Département Informatique	Directeur : M. Behzad SHARIAT
Département Mécanique	Directeur : M. Marc BUFFAT
UFR - Faculté des Sciences	Administrateur provisoire : M. Bruno ANDRIOLETTI
UFR (STAPS)	Directeur : M. Yannick VANPOULLE
Observatoire de Lyon	Directeur : Mme Isabelle DANIEL
Ecole Polytechnique Universitaire Lyon 1	Directeur : M. Emmanuel PERRIN
Ecole Supérieure de Chimie, Physique, Electronique (CPE Lyon)	Directeur : M. Gérard PIGNAULT
Institut Universitaire de Technologie de Lyon 1	Directeur : M. Christophe VITON
Institut de Science Financière et d'Assurances	Directeur : M. Nicolas LEBOISNE
ESPE	Administrateur provisoire : M. Pierre CHAREYRON

Titre. Méthodes eulériennes pour les problèmes inverses en transport optimal

Résumé. Cette thèse a pour but de développer de nouvelles méthodes numériques pour résoudre des problèmes inverses en transport optimal. On trouve les problèmes inverses dans diverses disciplines telles que l'astronomie, la géophysique, ou l'imagerie médicale, mais aussi dans des domaines plus proches du sujet de cette thèse, à savoir la vision par ordinateur, l'informatique graphique, et l'apprentissage automatique. Les problèmes inverses sont en général difficiles à résoudre car ils sont souvent mal posés (nombre infini de solutions, instabilités), et les modèles non-linéaires du transport optimal apportent des défis supplémentaires. Cependant, ces problèmes sont importants à résoudre car ils nous permettent d'obtenir des résultats sur des quantités qui ne sont pas directement observables, ce qui peut apporter de précieuses informations dans de nombreux cas. Les techniques existantes pour résoudre les problèmes inverses en traitement d'image et du signal et en apprentissage automatique considèrent souvent les histogrammes comme des vecteurs euclidiens. Elles ne parviennent donc pas à saisir et traiter correctement les relations sous-jacentes entre les *bins* des histogrammes, définies par la géométrie du domaine. Le transport optimal résout ce problème en définissant une distance entre histogrammes (et plus généralement entre distributions de probabilité) basée sur les distances entre les bins. Dans cette thèse, nous adaptons deux tâches classiques de l'apprentissage automatique à la géométrie du transport optimal : l'apprentissage de dictionnaire et l'apprentissage de métrique. Nos méthodes résolvent ces tâches en tant que problèmes d'optimisation et sont fondées sur la régularisation entropique du transport optimal, et la différentiation automatique. La régularisation fournit des approximations rapides, robustes et régulières (lisses) du transport, ce qui est essentiel pour obtenir des algorithmes d'optimisation efficaces. La différentiation automatique apporte une alternative rapide et fiable à la dérivation analytique manuelle, ce qui conduit à des méthodes flexibles. Nous illustrons nos deux algorithmes sur des applications en traitement d'image et en traitement du langage naturel.

Keywords. Transport optimal, Problèmes inverses, Optimisation, Informatique graphique, Vision par ordinateur, Machine learning

Laboratoire. Laboratoire d'InfoRmatique en Image et Systèmes d'information (LIRIS)
25 avenue Pierre de Coubertin 69100 Villeurbanne

Title. Eulerian methods for inverse problems using optimal transport

Abstract. The goal of this thesis is to develop new numerical methods to address inverse problems using optimal transport. Inverse problems appear in many disciplines such as astronomy, geophysics or medical imaging, but also in fields closer to the focus of this thesis, namely computer vision, computer graphics, and machine learning. They are difficult problems by nature as they are often not well posed: they may have an infinite number of solutions and/or instabilities. Furthermore, inverse problems that involve non-linear models such as optimal transport yield additional challenges. However, they are important problems to solve since they give access to quantities that are not directly observable, which provides major insight in many cases. Existing techniques for inverse problems in signal/image processing and machine learning often treat histograms as Euclidean data, thus failing to grasp the underlying relationships between the bins, defined by the geometry of the domain. Optimal transport addresses this issue by building on the distance between bins to produce a distance between histograms (and more generally probability distributions). In this thesis, we adapt two well-known machine learning tasks to the optimal transport framework: dictionary learning and metric learning. Our methods address these tasks as optimization problems and rely on the entropic regularization of optimal transport, and automatic differentiation. The regularization provides fast, robust and smooth approximations of the transport, which are essential features for efficient optimization schemes. Automatic differentiation provides a fast and reliable alternative to manual analytical derivation, resulting in flexible frameworks. We illustrate our algorithms with applications in image processing and natural language processing.

Keywords. Optimal transport, Inverse problems, Optimization, Computer graphics, Computer vision, Machine learning

Laboratory. Laboratoire d'InfoRmatique en Image et Systèmes d'information (LIRIS)
25 avenue Pierre de Coubertin 69100 Villeurbanne

Remerciements

J'aimerais tout d'abord remercier mes rapporteurs, Nicolas Courty et Quentin Mérigot, d'avoir accepté ce rôle. Merci pour votre relecture attentive, et vos retours constructifs. Merci à Raphaëlle Chaine et Julie Delon d'avoir accepté le rôle d'examinatrice de ma thèse. Merci pour vos questions pertinentes durant la soutenance.

Je remercie mes encadrants de thèse de Paris, Gabriel Peyré et Marco Cuturi,. Je suis honoré d'avoir pu collaborer avec vous, et j'aurais souhaité que ce soit plus fréquent, mais le fait de ne pas être dans la même ville n'a pas aidé. J'aimerais en particulier vous remercier pour votre livre *Computational Optimal Transport* [PC18], duquel j'ai énormément appris, et qui m'a permis de dissiper le brouillard dans lequel j'étais, au milieu des publications de transport optimal. Gabriel, tu m'as été d'une aide précieuse à de nombreuses occasions, tu as toujours été disponible malgré ton emploi du temps, et j'en suis très reconnaissant. Je ne comprends toujours pas comment tu arrives à tout faire, je te soupconne sérieusement de posséder un Retourneur de Temps®.

Je remercie bien sûr mes encadrants de thèse de Lyon, Nicolas Bonneel et David Coeurjolly. Nicolas, merci de m'avoir donné l'opportunité de faire cette thèse, j'ai appris beaucoup de choses, sur des sujets très variés, et j'en suis très heureux. David, j'ai beaucoup apprécié ton encadrement, et en particulier ta pédagogie. Tu as compris mes difficultés liées à mon perfectionnisme, et tu as su me rappeler, toujours avec bienveillance, ce qui est important et ce qui ne l'est pas. Merci donc pour ces 4 années de collaboration.

J'aimerais remercier l'équipe administrative du LIRIS: Brigitte, Isabelle, Catherine et Sylvie, qui font et ont fait un travail incomparable, et qui ont rendu toutes les tâches administratives aussi simples qu'agréables. Merci pour tous ces bons moments partagés. Merci aussi à Saïd pour son soutien et pour nos discussions.

Merci à tous les responsables d'UE, qui m'ont permis d'enseigner durant cette thèse: Nicolas Pronost, Vincent Nivoliers, Hamid Ladjal, Florence Zara, Raphaëlle Chaine, et Nicolas Louvet.

Je remercie également Bruno Lévy, Frédéric Lagoutière, et Hugo Lavenant pour leur aide durant ma thèse, merci pour nos discussions rares, mais très précieuses. Merci Hugo pour ton aide avec le postdoc à Vancouver.

J'aimerais ensuite remercier tous mes collègues, qui m'ont été d'un grand soutien durant cette thèse, et qui ont rendu ce travail beaucoup moins solitaire:

- Agathe, merci pour ton excentricité rieuse, et pour ta passion pour des trucs que personne ne connait (sauf Pink Floyd et les chiens). Merci d'avoir partagé la douleur d'un gradient qui bugge, je me suis senti beaucoup moins seul.
- Alexis, mon co-admirateur de François Perusse, merci pour ta joie de vivre, et pour ta rhétorique.
- Alice, merci pour ta bonne humeur communicative, ta belle énergie, nos chansons, nos discussions, les vidéos Sympa, le week-end au chalet, Fab Caro, lus purtus du rugulude et plus encore.
- Antoine D., merci pour ta dialectique infatigable, ton humour redoutable, et les jeux de société.
- Antoine W., merci pour les calembours, les discussions philosophiques plus ou moins sérieuses, et les discussions scientifiques en toute circonstance. Merci à toi et à Vesna pour les bons repas partagées et cette super rando, on aurait du en faire plus. Merci de t'être occupé si soigneusement de nos plantes et de notre chère et tendre Ortie, que je remercie également au passage.
- Arthur, merci pour ta bienveillance sympathique et ton dynamisme. Merci pour les conseils de grimpe, System, la fête de la musique, les concerts, les rêves de Parkour, le kouign-amann, le chouchen et tout le reste.
- Béa, merci pour nos discussions en hongrois, malgré mon piètre niveau, et les soirées multiculti.
- Basile, merci pour ton ardeur de programmation, et pour ce magnifique transport d'images voronoïées.
- Charles, merci pour ta sérénité charismatique, pour notre course à la soutenance, et pour ta résilience de publication. Merci pour tous ces bons souvenirs de caynoning, de canoë, de fêtes des lumières, et j'en passe.
- Gabrielle, merci pour ton amour partagé des chats, et pour les mémorables battles d'aventures félines avec Béarzi.
- Jocelyn B., merci pour ton oenophilie communicative et ta cuisine délicieuse.
- Jocelyn M., merci pour ta quiétude habituelle et pour ton aide en transport optimal. Merci d'être toujours prêt à galéjer, mais pas trop quand même.

- Julia S., merci pour les fous rires journaliers, pour le co-soutien durant ces longues années de perdition, pour tes facéties, ton humanité et ta persévérance. Merci d'avoir été ma plus fidèle co-bureau, et d'y avoir partagé détresses et enthousiasmes.
- Loïs, merci pour ta jovialité paisible et pour nos échanges sur l'acheminement idéal.
- Maxime, merci pour ta nonchalance adorable, ton âme d'artiste, notre musique, le ski, les randonnées, les jeux de mongolfière, et tellement plus. Même si tu es parti vers d'autres contrées, tu ne te feras pas oublier si facilement.
- Rémi, merci pour ton pragmatisme cynique. Merci de toujours apporter une bonne dose de scepticisme à nos débats, et un bon Saint Agaûne à nos soirées.
- Rémy D., merci pour ta culture infinie, et de toujours nous avoir tenu au courant de ce qu'il se passait sur Twitter #fracturenumérique.
- Rémy C., merci pour ta curiosité universelle, et ta passion des choses bien faites.
- Samuel, être surnaturel, se nourissant exclusivement de poivre. Ton intensité, et ton optimalité m'impressioneront toujours, merci.
- Simon, merci pour tes affabulations et ton humour pince-sans-rire.
- Thibaut, c'est l'histoire d'un mec qui rentre dans un café,... et plouf. C'est parce qu'en fait comme il *rentre* dans le café, ça fait plouf en fait. Comme si tu tombais dans un liquide tu vois ? Merci pour ta pêche légendaire, et tes blagues désopilantes.
- Thomas, décidément, entre la prépa CPE, le LIRIS, le bureau, Vancouver, et Kitware, on a un destin commun BB: "La vie, c'est comme une boîte de chocolat, on sait toujours qu'on va retomber sur Thomas Caissard". Merci pour toutes ces années depuis la prépa, passées à raconter n'importe quoi et se fendre la poire. Merci de ne jamais rien prendre au sérieux, et de m'avoir convaincu de l'infériorité d'Archinulix.
- Valentin, merci d'être prêt à résoudre n'importe quelle problème mathématique (discret) à toute heure et à pousser des hypothèses farfelues jusqu'au bout. Merci pour ton éloquence, pour nos rigolades, et pour toutes ces choses qui ont fait de toi un super co-bureau.

- Vincent, merci pour ta zénitude mystérieuse, et d'avoir été un précurseur de ma mise au vert.
- Yana, merci pour ton calme enjoué, et pour les gâteaux. Un jour ou l'autre, il faudra prendre les escaliers, tmtc.
- Yohan, merci pour ta ténacité politique et ton adulation du C++. Merci de m'avoir transmis la passion de la mayo du Domus.

Ensuite, j'aimerais remercier mes amis de CPE, avec qui j'aurais du aller boire des coups plus souvent. Mika, merci pour ta bienveillance et ton aide à de nombreuses reprises pour que je comprenne "le web". Tristan, merci pour ton soutien éternel, et pour les week-ends avec les cigales. Chazz, ça y est, j'ai enfin fini mes calculs. Ça fait 42, comme tu pouvais t'en douter. Merci à tous les autres, Pinzutu et non Pinzutu.

Merci à mes parents, qui mine de rien, m'ont permis d'arriver jusque là. Ce n'est pas rien, et j'en suis très reconnaissant. Merci pour votre soutien. Merci à mes grand-parents d'avoir assisté à ma soutenance en visio, aussi ésotérique soit-elle. Merci à mes frères d'être de si cools frangins, et merci à ma marraine d'être la gentillesse et la générosité incarnées.

Enfin, je remercie Julia L., qui m'a soutenu du début jusqu'à la fin, et que je pourrai à présent soutenir en retour. Merci de m'avoir épaulé, et de m'avoir poussé à sortir me balader, voyager, faire du ski, des randonnées, etc. J'ai été très heureux de partager tous ces moments avec toi, et j'en espère beaucoup d'autres.

Contents

Résumé en français

Cette thèse a pour but de développer de nouvelles méthodes numériques pour résoudre des problèmes inverses en transport optimal. Les problèmes inverses sont une large catégorie de problèmes où l'on cherche les facteurs qui expliquent l'observation d'un phénomène donné. Par exemple, en astronomie, l'image d'une étoile obtenue avec un télescope est légèrement déformée et floutée par les lentilles, par rapport à l'image réelle. Le problème inverse dans ce cas consiste à retrouver l'image réelle de l'étoile, à partir de l'image obtenue par le télescope, et d'un modèle de la manière dont le télescope déforme la réalité.

Mathématiquement, appelons f la fonction (ou le modèle) qui transforme les facteurs (ou paramètres) x en observations $y = f(x)$. Le problème *direct* consiste à trouver y à partir de x et est généralement résoluble car il suffit s'appliquer la fonction f à x. Dans l'exemple précédent, il s'agit simplement de capturer une image de l'étoile avec le télescope. Le problème *inverse* est l'inverse du problème direct, c'est-à-dire qu'il faut retrouver les paramètres x qui amènent à l'observation d'un y donné : trouver la vraie image de l'étoile à partir de l'image capturée. En général, on considère les problèmes inverses pour lesquels f est une fonction complexe et non inversible. Ils sont souvent difficiles à résoudre car ils peuvent être mal posés (nombre infini de solutions, instabilités), et les modèles non linéaires (tels que ceux du transport optimal) apportent des défis supplémentaires. Cependant, ces problèmes sont importants à résoudre car ils nous permettent d'obtenir des résultats sur des quantités qui ne sont pas directement observables, ce qui peut apporter des informations précieuses dans de nombreux cas. On trouve également les problèmes inverses dans des disciplines plus proches du sujet de cette thèse, telles que la vision par ordinateur ("shape from shading", séparation de signaux, super résolution), l'informatique graphique (problème de rendu inverse, "inpainting") et l'apprentissage automatique (estimation de paramètres, inférence statistique, apprentissage supervisé).

En tant que sciences des données, ces disciplines utilisent fréquemment des outils statistiques tels que des distributions de probabilité, par exemple sous la forme d'histogrammes. Cependant, les méthodes existantes résolvant des problèmes inverses considèrent souvent les histogrammes comme des vecteurs euclidiens de $\mathbb{R}^N$. Elles ne parviennent donc pas à saisir et traiter correctement les relations sous-jacentes entre les *bins* des histogrammes, définies par la géométrie du domaine.

Le transport optimal est une théorie mathématique qui résout ces problèmes en utilisant la distance entre les bins pour définir une distance entre histogrammes

(et plus généralement entre distributions de probabilité) qui est géométriquement plus significative que la distance euclidienne. Le transport optimal a été introduit par le mathématicien Gaspard Monge dans son “Mémoire sur la théorie des déblais et des remblais” en 1781 [Mon81]. Il pose le problème comme suit : étant donné un tas de terre d'un volume donné et un trou du même volume négatif, quelle est la manière optimale de transporter toute la terre du tas vers le trou ? Transporter une portion de terre a un coût qui dépend de la distance parcourue, et le coût total de transport est obtenu en additionnant le coût de transport de chaque portion de terre multipliée par sa masse. Un transport est dit *optimal* quand il minimise ce coût total, et c'est ce coût minimal qui définit la distance entre des distributions de probabilités (vues comme des tas de terre), appelée distance de Wasserstein. La théorie du transport optimal ne définit pas seulement une distance, elle fournit aussi de puissants outils non linéaires pour modéliser des problèmes faisant appel à des notions de moyenne, de chemin ou de coût. En particulier, la moyenne pondérée entre des distributions, au sens de la distance de Wasserstein est appelée barycentre de Wasserstein, et la trouver est un problème inverse en soi.

Les problèmes inverses auxquelles nous nous sommes intéressés dans cette thèse sont ceux qui utilisent ces barycentres de Wasserstein dans le modèle, c'est-à-dire dans le problème *direct*. Nous cherchons donc à inverser un modèle qui contient lui-même des résolutions de problèmes inverses. Ce genre de problème a encore été peu étudié : la majorité des problèmes inverses impliquant le transport optimal n'utilisent que des distances de Wasserstein et pas de barycentres. Nos algorithmes font appel à la régularisation entropique du transport optimal, qui fournit des approximations rapides, robustes et régulières (lisses) du transport, ce qui est essentiel pour obtenir des algorithmes d'optimisation efficaces. Ils utilisent également les techniques de différenciation automatique, qui apportent une alternative rapide et fiable à la dérivation analytique manuelle, et qui conduit à des méthodes flexibles. Nous étendons deux problèmes classiques de l'apprentissage automatique à la géométrie du transport optimal : l'apprentissage de dictionnaire et l'apprentissage de métrique.

L'apprentissage de dictionnaire consiste à trouver, à partir de plusieurs éléments en entrée (images, histogrammes, etc.), un dictionnaire restreint d'éléments de référence (appelés *atomes*) qui en soient les plus représentatifs. Pour s'assurer que les atomes soient représentatifs, il faut qu'ils puissent ensemble reconstruire chaque élément d'entrée le plus fidèlement possible (selon un certain critère). Dans le cas classique, cette reconstruction s'opère avec des combinaisons linéaires entre les atomes, pondérées par des poids. Nous étendons le cas classique aux histogrammes, en remplaçant les combinaisons linéaires par des barycentres de

Wasserstein, également pondérés par des poids. Nous posons le problème comme la minimisation de l'erreur de reconstruction des histogrammes d'entrée, et optimisons conjointement les atomes et les poids. Nous illustrons notre algorithme au travers d'applications en traitement d'image et traitement du langage naturel, et présentons également quelques extensions de cet algorithme.

L'apprentissage de métrique consiste classiquement à apprendre un ensemble de distances entre des éléments, afin qu'elles correspondent aux informations prescrites de similarité et dissimilarité entre ces éléments. En transport optimal, la distance de Wasserstein est directement dépendante de la distance entre les points du domaine, souvent appelée la *métrique*. Quand cette métrique est le carré de la distance euclidienne, tous les éléments de matières (portions de terre) se déplacent en ligne droite lors du transport. Dans le cas où on observerait un transport de matière dont les trajectoires ne sont pas rectilignes, on peut tenter de résoudre le problème inverse suivant : existe-t-il une métrique pour laquelle ce déplacement est optimal, et si oui, laquelle ? Nous posons le problème d'apprendre, à partir d'une séquence de déplacement de matière sur un graphe, la métrique selon laquelle ce transport serait optimal. La séquence de déplacements est un ensemble de distributions de probabilité sur les noeuds d'un graphe à différents instants, et la métrique est une métrique de graphe, c'est-à-dire un poids sur chaque arête traduisant la facilité de déplacement de la matière à travers cette arête. Nous calculons les distances géodésiques sur le graphe à l'aide de l'équation de diffusion, ce qui permet un algorithme d'optimisation efficace. Nous validons notre approche sur des données synthétiques, puis sur l'apprentissage de variations de couleurs dans des séquences d'images.

1 Introduction

Inverse problems are a broad category of problems in which we aim to find the factors that explain the observation of a phenomenon. A classical inverse problem in signal processing is the source separation, also known as the "cocktail party problem": if you record the sound of people chatting at a cocktail party, you get a superposition of different voices and chatter. The problem is then to recover the factors that explain the recorded sound, which in this case consists in separating individual voices in the crowd. Another typical inverse problem is tomography, an imaging technique in which we reconstruct an image of a physical body's interior, from sectional measurements taken from outside. Tomography is used for example in radiology to reconstruct the volume of a body part from absorption or emission measurements along multiple directions (CT-Scan, MRI, etc.), but also in geophysics to retrieve physical properties (density, conductivity, etc.) of the Earth's interior from acoustic waves observed at its surface. These examples demonstrate the relevance of inverse problems in various fields, and the importance of studying them.

Mathematically, let f be the function that transforms the factors, or parameters x into the observations $y = f(x)$. The *direct* or *forward problem* is to find y given x, and is generally solvable assuming we are able to apply f. The *inverse problem* is the "inverse" of the *direct* problem, that is, to find the parameters x that explain the observation of y. In general, we consider inverse problems where f is a complex non-invertible function. Such problems are important and theoretically interesting because they inform us on quantities that are not directly observable, which provides major insight in many disciplines. Inverse problems where f is a linear function have been well studied. When they are discrete (or discretized), they result in linear systems of equations, and a large panel of tools exists to solve these systems. Non-linear inverse problems, however, are inherently more challenging as there is no "one-fits-all" approach to solve them. One usually tackles them using optimization methods, as we detail in the next section. They are still actively studied and are the problems of interest in this dissertation.

Inverse problems are frequent in data-oriented disciplines such as image/signal processing and machine learning, which are important bricks of our fields of interest, that is, computer graphics and computer vision. These two fields can actually be seen

as inverse to each other: one goes from a structured scene to rendered images while the other infers structure from real images. Inverse problems in signal processing include deconvolution, inpainting, source separation, super resolution or shape from shading. In machine learning, tasks such as model fitting, statistical inference, or supervised learning can be considered inverse problems.

As data sciences, signal processing and machine learning commonly use statistical distributions, such as histograms, to model, represent or summarize data. Some examples are color palettes or reflectance functions in computer graphics, SIFT features, histograms of oriented gradients, or Hough transforms in computer vision, and bag-of-words, histogram features or distance histograms in machine learning. In these fields, existing techniques for solving inverse problems involving histograms often treat them as Euclidean data i.e. (feature) vectors in $\mathbb{R}^N$. Therefore, they fail to grasp the underlying relationships between the bins, defined by the geometry of the "ground" domain. For instance, when comparing two color histograms, two neighboring bins represent similar colors, however the Euclidean distance only compares histograms bin-wise and therefore does not account for that similarity.

Optimal transport (OT) is a mathematical theory that addresses this issue by building on the *ground distance* between bins to establish a distance between histograms (and more generally probability distributions) that is geometrically more meaningful than the Euclidean distance. OT was introduced by the French mathematician Gaspard Monge in "Mémoire sur la théorie des déblais et des remblais" in 1781 [Mon81]. Informally, he posed the problem as follows: Given a pile of earth of a certain volume and a pit of that same negative volume, what is the optimal way to transport earth from the pile to the pit? An illustration is provided in Figure 1.4. Transporting earth has a cost, which depends on the distance traveled by each particle of earth. The total cost of transport is obtained by summing the transport costs of each particle. A transport is *optimal* when it minimizes this total cost. This minimum cost defines a distance between probability distributions (piles of earth). Not restricted to distances, optimal transport also provides powerful non-linear modeling tools for problems dealing with averages, paths or costs. We are thus interested in solving inverse problems for which the forward problem involves OT and is therefore non-linear.

One of the main challenges in this thesis is posed by non-linear inverse problems: they are often not well posed (see [Kir11]), which makes them difficult to solve. For example, different values for the parameters x may lead to the same observations y. Moreover, solving OT problems is costly and addressing OT inverse problems is even more so, as it requires repeated calls to OT solvers during the optimization process.

Therefore, another challenge is to develop scalable and robust algorithms to mitigate these issues and solve inverse problems of large size in reasonable time.

1.1 Context of the thesis

In this section, we introduce inverse problems in further detail, then present some basic notions of probability theory and finally present a brief historical review on optimal transport.

1.1.1 Inverse problems

Following the notations of the introduction, f is a function, and x and y are elements of the domain and codomain of f respectively, such that $y = f(x)$. We often designate f as a *model* depending on parameters x, producing a prediction $f(x)$ to model the observations y. See Figure 1.1 for an illustration.

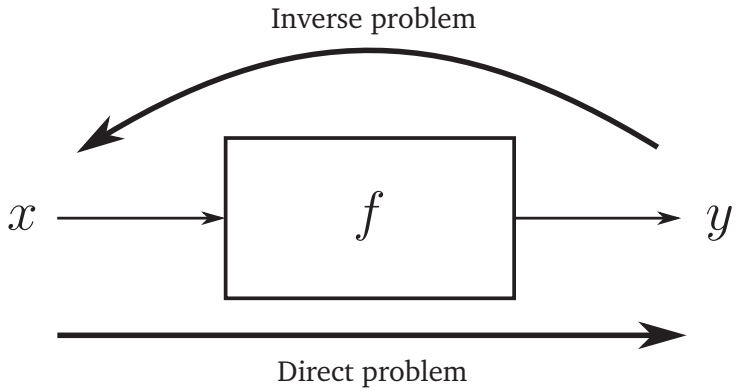


Figure 1.1: Illustration of the direct and inverse problems

We will not recall the theory of linear inverse problems, which are solved as linear systems of equations, but refer the reader to textbooks on linear algebra. As we just mentioned, interesting non-linear inverse problems are generally not well posed (what is called ill-posed [Kir11]), which means that they might have either zero or more than one solutions, or that they may have stability issues : the parameters x do not vary continuously with changes in the observations y. When inverse problems have no solution (i.e. they are over-constrained), one either relaxes some constraints of the problem, or finds the best approximate solution, which is the one that minimizes a given similarity criterion between y and $f(x)$. When inverse

problems have multiple solutions (i.e. they are under-constrained), one often resorts to the regularization of the problem which adds prior assumptions about the solution, such as sparsity, smoothness, etc. This restriction of the problem reduces the space of solutions and averts overfitting.

Inverse problems are typically written as optimization problems, as we attempt to minimize the difference between the prediction of the model and the observations. Formally, they can be written as

$$\min_x \ \mathcal{E}(x) \overset{\text{def.}}{=} \mathcal{L}(f(x), y)\,, \tag{1.1}$$

where $\mathcal{L}$ is a similarity criterion, also called *loss function*, and $\mathcal{E}$ is called the *energy* or *objective function*. When there are multiple observations and the model aims to reconstruct each of them as faithfully as possible, the formulation simply extends to

$$\min_x \ \mathcal{E}(x) \overset{\text{def.}}{=} \sum_i \mathcal{L}(f(x, i), y_i). \tag{1.2}$$

Solving an inverse problem is then achieved with an optimization method chosen according to the problem at hand, and generally requires the computation of the gradient of the energy $\mathcal{E}$ with respect to the parameters x.

1.1.2 Notions of probability

In this dissertation, the mathematical objects of interest are *probability distributions*, also called *probability measures*. A *probability distribution* is a real-valued function that assigns to sets of events, the probability of their realization. Identical to random variables, probability distributions can be *discrete* or *continuous*, depending on whether they measure a countable or uncountable set of possible outcomes Ω. For example, the roll of a die leads to a discrete set of possible outcomes $\{1, 2, 3, 4, 5, 6\}$, whereas the random variable describing the angle at which a weather vane points is measured on the continuous set $[0, 2\pi[$.

We define the probability simplex as the set of vectors whose components are positive and sum up to 1:

$$\Sigma_N \overset{\text{def.}}{=} \left\{ \boldsymbol{a} \in \mathbb{R}_+^N | \sum_{i=1}^N \boldsymbol{a}_i = 1 \right\}. \tag{1.3}$$

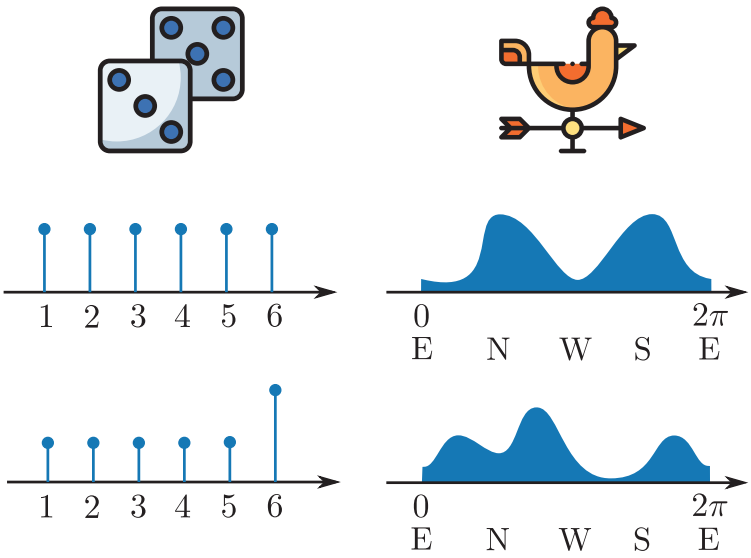


Figure 1.2: Left column: 1-D *discrete* distribution of a fair die (top) and of a trick die (bottom) Right column: 1-D *continuous* distribution of wind directions in Lyon (top) and Marseille (bottom).

We define a discrete distribution (or discrete measure) α as a sum of weighted Dirac distributions supported at locations $x_1, \dots, x_M$:

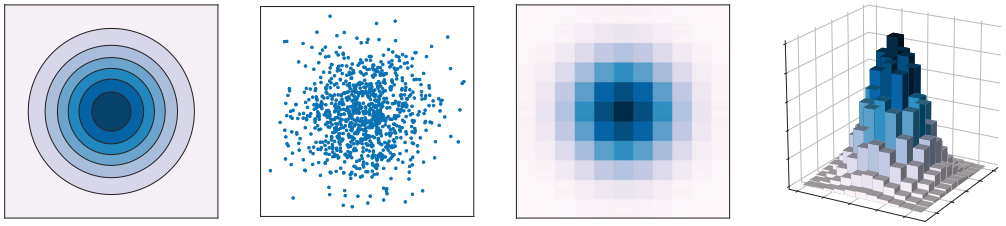
$$\alpha = \sum_{i=1}^{M} \boldsymbol{a}_i \delta_{x_i}, \tag{1.4}$$

with the weight vector $\boldsymbol{a} = (\boldsymbol{a}_i)_i$ belonging to the probability simplex Σ_M.

When a probability distribution is continuous, it is generally associated with a *probability density function*: the probability of a (continuous) subset of Ω is then obtained by integrating the density function over that subset. Illustrations of 1-dimensional discrete and continuous distributions are shown in Figure 1.2.

Dealing with continuous distributions on a computer requires a proper discretization. Two common ways to do so are the *Lagrangian* and *Eulerian* approaches, borrowed from the field of fluid mechanics. When we consider a fluid motion, the Lagrangian approach is to follow individual particles and study their properties such as their position, velocity, etc. On the other hand, the Eulerian approach is to look at fixed points of the domain and study quantities such as the fluid density or velocity at each of those points. See Figure 1.3 for examples of such discretizations on a 2-D distribution. In summary:

- A *Lagrangian* discretization of a continuous distribution is a discrete distribution with point locations sampled from the continuous one, and fixed weights (generally uniform).
- A *Eulerian* discretization of a continuous distribution is a discrete distribution with fixed point locations (usually) on a grid, and weights matching the density function of the continuous distribution integrated over each cell defined by the grid.



(a) (b) (c) (d)

Figure 1.3: (a) 2-D continuous distribution. (b) Lagrangian discrete distribution, which is a realization of the first one (samples are drawn from it). (c) Eulerian discrete distribution of the first distribution, also called a *histogram*. (d) 3-D representation of the histogram, with height proportional to the density of the continuous distribution.

We call a "histogram" a Eulerian discretization of a continuous distribution. Histograms can be defined solely by their weight vector (and the extent of their domain) since the point locations are implicitly defined by the grid. In this dissertation, we deal with histograms that are defined on square d-dimensional domains, such that the number of bins $N = n^{\mathrm{d}}$, with n the number of grid samples per axis. We generally refer to the domain dimension d as the histogram's dimension, even though that term can sometimes be interpreted as its total size N in the literature. We will interchangeably use the terms "distribution", "measure" or "density" for general distributions (either discrete or continuous). Probability is often referred to as "mass", by analogy with the law of mass conservation in physics, since the total probability over a domain Ω is always 1.

1.1.3 Optimal transport

We now present OT more formally, starting with the Monge problem, introduced in 1781 [Mon81]. We consider two probability distributions μ and ν supported on respective measure spaces X and Y. The Monge problem is to find a transport map T that assigns to every $x \in X$ a single $y = T(x) \in Y$, such that all mass of μ is

transported to ν, and the total transport cost is minimized (see Figure 1.4). The Monge problem is therefore written as

$$\min_{T \text{ s.t. } T_{\#}\mu=\nu} \int_X c(x, T(x)) d\mu(x), \tag{1.5}$$

with $c : X \times Y \to \mathbb{R}$ the cost function, often taken as a power p of the Euclidean distance. Essentially, we sum for each point x in X the amount of mass $d\mu(x)$ present at that point multiplied by the cost $c(x, y)$ of transporting it to its destination $y = T(x)$, and seek to minimize that sum. The rigorous definition of the push-forward operator $T_{\#}$ is beyond the scope of this thesis, but it is essentially the "twin" of the map T that acts on probability distributions instead of points. Therefore, $T_{\#}\mu = \nu$ ensures that all mass of μ is transported to ν, i.e. no mass is lost.

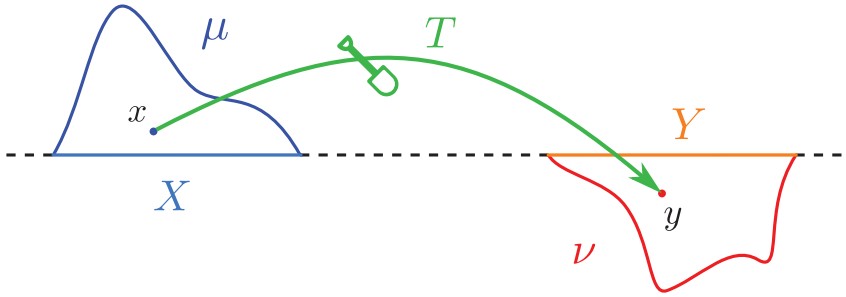


Figure 1.4: Illustration of optimal transport between two distributions μ and ν supported on 1-D spaces X and Y. The map T transports each point x of X to a point $y = T(x)$ in Y.

When dealing with discrete distributions, the Monge formulation can model an optimal assignment problem. Indeed, when both distributions have the same number of Dirac masses and uniform weights, the Monge problem searches for a one-to-one matching (a bijection) between points, that minimizes the total cost.

In 1942, the Russian mathematician Leonid Kantorovich recast this problem [Kan42] and applied it to the optimal allocation of resources, for which he won a Nobel prize in economics. The Kantorovich formulation is a relaxation of the Monge formulation (1.5). Instead of modeling the point assignments with a map T, it uses a joint probability distribution (or coupling) π over the product space $X \times Y$, with $\pi(x, y)$ giving how much mass is transported from x to y. This allows a separation of the mass present at x to different destinations y, which was not possible in the Monge formulation since T is a map (to each x is assigned a single $y = T(x)$). Therefore, the Kantorovich problem is to find

$$\mathscr{L}(\mu, \nu) \overset{\text{def.}}{=} \min_{\pi \in \mathcal{U}(\mu,\nu)} \int_{X \times Y} c(x, y) d\pi(x, y) \tag{1.6}$$

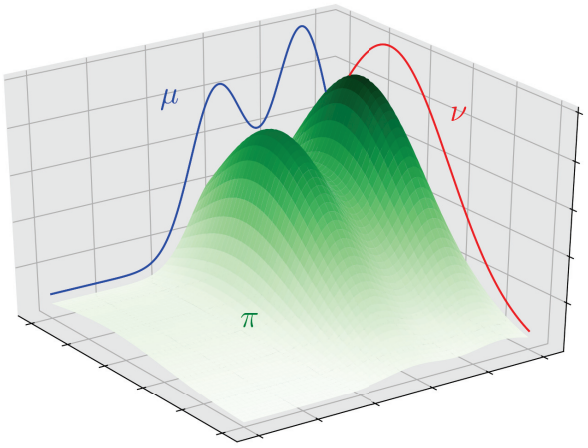


Figure 1.5: Illustration of a coupling π with its two marginals μ and ν. Integrating π along one axis gives μ, and integrating it along the other axis gives ν. This coupling is not optimal, it is simply the product $\pi(x, y) = \mu(x)\nu(y)$.

with $\mathcal{U}(\mu, \nu)$ the set of couplings that have μ and ν as marginals, i.e. that verifies

$$\int_Y d\pi(x, y) = d\mu(x) \quad \text{and} \quad \int_X d\pi(x, y) d\nu(y) \tag{1.7}$$

An illustration of a coupling and its marginals is shown in Figure 1.5.

In practice, we generally use the Kantorovich formulation to solve OT problems because it is less restrictive and easy to discretize, as described in the following chapter (2.1.1).

One of the main advantages of OT is that it defines a distance between probability distributions. If the cost is taken as a power of a distance d on the domain $X = Y$: $c(x, y) = d(x, y)^p$, then $\mathscr{L}(\mu, \nu)^{1/p}$ defines a distance between probability measures, called the p-Wasserstein distance. This distance is often more geometrically significant than other distribution distances, because it takes into account the underlying geometry of the domain, which is encoded in the distance function d. It is worthy to note that the p-Wasserstein distance is defined for *arbitrary* distributions, meaning that they can be discrete or continuous (see Figure 1.3). This versatility enables the use of OT in various fields of application.

The p-Wasserstein distance is often referred to as the *earth mover's distance* (EMD) when $p = 1$ and as the *quadratic Wasserstein distance* when $p = 2$. In this thesis, we focus on the latter, which we simply call *Wasserstein distance* for simplicity. The space of probability measures endowed with the Wasserstein distance is called the Wasserstein space. The distance function d on the domain is frequently referred to

as the *ground distance*, following Rubner et al. [Rub+00], and the domain as the *ground space* by extension.

The next surge of interest for OT appeared in the mathematics community in the nineties, with the landmark paper of Brenier [Bre91]. Further research unveiled the intermediate role of optimal transport between fluid mechanics, probability theory, geometry, optimization and partial differential equations. In particular, some of these advances were rewarded with Fields medals, for C. Villani in 2010 and A. Figalli in 2018. As many applied mathematicians and statisticians worked on developing the optimal transport theory, efficient methods to solve OT problems *computationally* appeared in the 2010's, which led to an increasing use of the tools provided by OT, in data science, computer graphics and computer vision. Applications of OT in these fields are detailed in the next chapter (2.1.6). We refer the reader to the books by Villani for a *theoretical* review of OT [Vil03; Vil09], and to those by Peyré and Cuturi [PC18], and Santambrogio [San15] for a more *computational* review.

To summarize, optimal transport is an attractive tool from both the theoretical and applied perspectives, and is getting increasingly popular in various communities for its versatility.

We are therefore interested in tackling inverse problems in computer graphics, computer vision and machine learning, using optimal transport. Such problems are important as they could improve existing methods dealing with probability distributions, since OT handles them in a geometrically meaningful way. One specificity of the inverse problems we address is that, contrary to most learning problems using OT, they do not employ the Wasserstein distance simply as a loss function. They involve quantities that are themselves built upon the Wasserstein distance, such as Wasserstein barycenters (see 2.1.4). OT is therefore used *in* the model we aim to invert, rather than solely at its output for comparing reconstructions.

In this manuscript, we solve two inverse problems that are machine learning problems, namely dictionary learning and metric learning, in the context of optimal transport. In brief, dictionary learning consists in finding a restricted basis of elements which can best represent a (large) dataset, and metric learning is the search of a set of distances between elements in a space, so that they match prescribed similarity information. Both of our algorithms are based on the entropic regularization of OT, which smooths computed quantities (distances, barycenters). This introduces bias in the results, but allows for fast computations and differentiability.

1.2 Structure of the manuscript

In chapter 2, we introduce the concepts of OT in greater detail, as well as the background on dictionary learning and metric learning. In chapter 3, we develop a non-linear unsupervised dictionary learning algorithm in the Wasserstein space. In chapter 4, we introduce a new framework to learn the ground metric of optimal transport, where it is restricted to be a geodesic distance on a graph. Finally, we conclude this thesis in chapter 5, and discuss possible future work.

1.3 Notations

- $\boldsymbol{u}$: vectors are in lowercase bold letters
- $\mathbf{A}$: matrices are in uppercase bold letters
- $\mathbf{A}^T$: transpose of matrix $\mathbf{A}$
- $\mathbb{1}_N$: vector of ones of size N
- $\mathbf{Id}_N, \mathbf{Id}$: identity matrix of size $N \times N$ or of implicit size
- $\mathrm{diag}(\boldsymbol{u})$: matrix with diagonal $\boldsymbol{u}$ and zero otherwise
- $\langle \boldsymbol{u}, \boldsymbol{v} \rangle \overset{\text{def.}}{=} \boldsymbol{u}^T \boldsymbol{v} = \sum_i \boldsymbol{u}_i \boldsymbol{v}_i$: inner product between vectors
- $\langle \mathbf{A}, \mathbf{B} \rangle \overset{\text{def.}}{=} tr(\mathbf{A}^T \mathbf{B}) = \sum_{ij} \mathbf{A}_{ij} \mathbf{B}_{ij}$: Frobenius inner product between matrices
- $\odot$: Hadamard (entry-wise) product between vectors or matrices
- $\log(\cdot), \exp(\cdot)$: entry-wise logarithm and exponential of vectors and matrices
- $[\boldsymbol{a}_i]_j$: j-th coordinate of the i-th vector of a family $\mathbf{A} = [\boldsymbol{a}_1, \ldots, \boldsymbol{a}_m]$.

2 Preliminaries

In this chapter, we first provide the background on *computational* optimal transport, and in particular on its entropic regularization. We then proceed to a state of the art on applications of OT in machine learning and computer graphics, as well as on the two concepts our contributions rely on, that is, dictionary learning and metric learning.

2.1 Optimal transport

In the introduction, we have presented optimal transport in the more theoretical continuous setting. Since this thesis focuses on the computation of optimal transport, we present the context in the discrete setting, and consider distributions that are histograms. We define the set of admissible couplings between two histograms $\boldsymbol{a}$ and $\boldsymbol{b}$ as the set of matrices having $\boldsymbol{a}$ and $\boldsymbol{b}$ as marginal distributions (discrete equivalent of (1.7)):

$$\Pi(\boldsymbol{a},\boldsymbol{b}) \overset{\text{def.}}{=} \left\{ \mathbf{P} \in \mathbb{R}_+^{N\times N} \;\middle|\; \mathbf{P}\mathbb{1}_N = \boldsymbol{a} \;\text{ and }\; \mathbf{P}^T\mathbb{1}_N = \boldsymbol{b} \right\}. \tag{2.1}$$

An admissible coupling matrix $\mathbf{P} \in \Pi(\boldsymbol{a},\boldsymbol{b})$ is also called a transport plan, because $\mathbf{P}_{i,j}$ can be interpreted as the amount of mass to be transported from bin i of $\boldsymbol{a}$ to bin j of $\boldsymbol{b}$.

2.1.1 The Kantorovich problem

Optimal transport aims to find the transport plan $\mathbf{P}$ that minimizes a total cost, which is the mass transported multiplied by its cost of transportation. This is called the Kantorovich problem and it is written

$$\mathrm{W}_{\mathbf{C}}(\boldsymbol{a},\boldsymbol{b}) \overset{\text{def.}}{=} \min_{\mathbf{P}\in\Pi(\boldsymbol{a},\boldsymbol{b})} \langle \mathbf{C}, \mathbf{P} \rangle \tag{2.2}$$

The matrix $\mathbf{C} \in \mathbb{R}_+^{N\times N}$ defines the cost $\mathbf{C}_{i,j}$ of transporting one unit of mass from bin i to j. In the Eulerian setting (our case of study), this cost matrix is fixed, as the

point locations are the same for all histograms defined on the grid. In the Lagrangian setting however this cost matrix is computed for each pair of distribution. This nuance is important when differentiating $\mathrm{W}_{\mathbf{C}}(\boldsymbol{a}, \boldsymbol{b})$ with respect to its inputs.

As mentioned in 1.1.3, if the cost matrix derives from a distance d on the domain, i.e. $\mathbf{C}_{i,j} = \mathbf{D}_{i,j}^p = d(x_i, x_j)^p$, then $\mathrm{W}_{\mathbf{C}}(\boldsymbol{a}, \boldsymbol{b})^{1/p}$ defines a distance between histograms, called the p-Wasserstein distance.

The transportation problem (2.2) is a linear program (LP), and was in fact one of the choice applications during the developments of linear programming algorithms, pioneered by, among others, Tolstoı [Tol30], Hitchcock [Hit41] and Kantorovich himself [Kan42]. We refer to [PC18, §3.1] for more details on the Kantorovich linear program, and to [Bon+11] for a comparison of simplex algorithms to solve it.

2.1.2 Entropic regularization

Solving optimal transport has been a challenge until the last decade because of the computational burden it involves. The best linear solver to date for OT is the network simplex [KK12] with a worst-case time complexity of $O(N^3)$, and $O(N^2)$ in some configurations [Bon+11]. Moreover, p-Wasserstein distances lack smoothness and are not very robust. Indeed, a slight change in the input measures can perturb the transport plan to a significant extent.

For this reason, cheaper approximations and regularizations of the original problem (2.2) have been proposed in order to address those issues. For example, the sliced approximation computes OT distances between d-dimensional histograms, from OT distances between their 1D projections, which are computable in closed-form [Rab+11; Bon+15]. Some methods seek embeddings of the Wasserstein space to reduce the complexity of computing distances, using the Wavelet transform [SJ08], or auto-encoders [Cou+17]. Better theoretical bounds on additive approximations of the transport have recently been proposed, based on specialized linear programs [Qua18].

Another line of research relies on regularizations of the problem. Among them, the entropy regularization [Cut13] adds a trade-off between a minimum transport cost and a maximum Shannon entropy of the transport plan. This leads to a simple and computationally efficient algorithm, which we detail later on. The entropic regularization has had a large impact in the fields of machine learning and computer graphics (see section 2.1.6).

Other regularizations of the transport map have been proposed. Some of them aim to reduce artifacts in color transfer, either as a post-processing step via non-local filtering [Rab+10], or directly integrated in the formulation using the L^p norm of the transport map's gradient [Fer+13]. Others, like the entropic regularization, are designed to reduce the complexity of the problem. An example is the quadratic regularization [Blo+18; ES18] which, unlike the entropic one, leads to sparse approximations of the transport plan. Finally, unified frameworks to handle any strongly convex regularizers have also been introduced [Dvu+18; Des+18b].

In this thesis we have worked mainly with the entropic regularization, because it produces smooth, and thus differentiable quantities which is very convenient for solving inverse problems. Moreover, the scalability it provides compared to simplex algorithms is crucial for machine learning and computer graphics applications involving large size histograms. The entropy-regularized problem is written as

$$\mathrm{W}_{\mathbf{C}}^{\varepsilon}(\boldsymbol{a},\boldsymbol{b}) \overset{\text{def.}}{=} \min_{\mathbf{P}\in\Pi(\boldsymbol{a},\boldsymbol{b})} \langle \mathbf{C},\mathbf{P}\rangle - \varepsilon \mathrm{H}(\mathbf{P}), \tag{2.3}$$

where $\mathrm{H}(\mathbf{P}) \overset{\text{def.}}{=} -\sum_{i,j} \mathbf{P}_{i,j}\left(\log(\mathbf{P}_{i,j})-1\right)$ is the discrete entropy.

This regularization introduces a smoothness in the transport plan by allowing its entropy to increase. The obtained quantity is a smooth approximation of the exact Wasserstein distance, which is controlled by ε. One can see the addition of entropy as having a *probabilistic* assignment, meaning that we tolerate that points are not matched one to one, but are rather distributed among the neighboring target points. Moreover, it transforms the problem into one that is solvable through an inexpensive iterative scaling method, as we are about to see.

Equation (2.3) is a constrained optimization problem, and in order to solve it as an unconstrained one, we add Lagrange multipliers for the constraints [Uza14]. Introducing the vectors $\boldsymbol{f}$ and $\boldsymbol{g}$ for the equality constraints in $\Pi(\boldsymbol{a},\boldsymbol{b})$, the Lagrangian of (2.3) is

$$\mathrm{L}(\mathbf{P},\boldsymbol{f},\boldsymbol{g}) \overset{\text{def.}}{=} \langle \mathbf{C},\mathbf{P}\rangle - \varepsilon \mathrm{H}(\mathbf{P}) - \langle \boldsymbol{f}, \mathbf{P}\mathbb{1}_N - \boldsymbol{a}\rangle - \left\langle \boldsymbol{g}, \mathbf{P}^T\mathbb{1}_N - \boldsymbol{b}\right\rangle. \tag{2.4}$$

Let us now write the first order optimality condition of this functional, for each entry $\mathbf{P}_{i,j}$:

$$\begin{aligned}
&\frac{\partial \mathrm{L}(\mathbf{P}, \boldsymbol{f}, \boldsymbol{g})}{\partial \mathbf{P}_{i,j}} = 0 \\
\Rightarrow\ & \mathbf{C}_{i,j} - \varepsilon \log(\mathbf{P}_{i,j}) - \boldsymbol{f}_i - \boldsymbol{g}_j = 0 \\
\Rightarrow\ & \varepsilon \log(\mathbf{P}_{i,j}) = -\mathbf{C}_{i,j} + \boldsymbol{f}_i + \boldsymbol{g}_j \\
\Rightarrow\ & \mathbf{P}_{i,j} = \exp(\boldsymbol{f}_i/\varepsilon) \exp(-\mathbf{C}_{i,j}/\varepsilon) \exp(\boldsymbol{g}_j/\varepsilon). \qquad (2.5)
\end{aligned}$$

As a result, the optimal transport plan can be written as

$$\mathbf{P} = \mathrm{diag}(\boldsymbol{u}) \mathbf{K} \, \mathrm{diag}(\boldsymbol{v}), \qquad (2.6)$$

with the kernel $\mathbf{K}$, and the so-called scaling vectors $\boldsymbol{u}$ and $\boldsymbol{v}$ defined as:

$$\mathbf{K} \overset{\text{def.}}{=} \exp(-\mathbf{C}/\varepsilon) \qquad (2.7)$$

$$\boldsymbol{u} \overset{\text{def.}}{=} \exp(\boldsymbol{f}/\varepsilon) \qquad (2.8)$$

$$\boldsymbol{v} \overset{\text{def.}}{=} \exp(\boldsymbol{g}/\varepsilon). \qquad (2.9)$$

The vectors $\boldsymbol{f}$ and $\boldsymbol{g}$ are also called the *dual potentials*, because they appear in the dual problem associated with (2.3), see [PC18, §4.4].

The kernel $\mathbf{K}$ is sometimes referred to as the Gibbs kernel, in analogy with the Gibbs measure, which is also known in physics as the Boltzmann distribution. This distribution is used to describe the probability that a system is in a particular state (or configuration), depending on that state's energy and the temperature. Following that analogy, the entropic parameter ε is sometimes called the "temperature" [Dup+16; Fey+19], in particular in the seminal paper that first proposed the Sinkhorn algorithm to solve an assignment problem [KY94].

The Sinkhorn algorithm

Sinkhorn and Knopp [SK67] proved that a fixed-point iteration can be used to obtain such a decomposition. It consists in alternatively scaling the rows and columns of the kernel $\mathbf{K}$ so that it satisfies the marginal constraints of $\Pi(\boldsymbol{a}, \boldsymbol{b})$: $\mathbf{P}\mathbb{1} = \boldsymbol{a}$ and $\mathbf{P}^T\mathbb{1} = \boldsymbol{b}$. This is equivalent to performing Bregman projections [Bre67] for the

Kullback-Leibler divergence, as described later in this section. Denoting $\mathbf{P}^{(0)} = \mathbf{K}$, this iterative procedure can be written as follows:

$$\mathbf{P}^{(2l+1)} = \operatorname{diag}\left(\frac{\boldsymbol{a}}{\mathbf{P}^{(2l)}\mathbb{1}}\right)\mathbf{P}^{(2l)} \tag{2.10}$$

$$\mathbf{P}^{(2l+2)} = \mathbf{P}^{(2l+1)}\operatorname{diag}\left(\frac{\boldsymbol{b}}{{\mathbf{P}^{(2l+1)}}^T\mathbb{1}}\right). \tag{2.11}$$

However, considering the decomposition (2.6), we show hereafter that these iterates can be simplified to handle only the scaling vectors $\boldsymbol{u}$ and $\boldsymbol{v}$.

Let us denote $\mathbf{P}^{(2l)} = \operatorname{diag}(\boldsymbol{u}^{(l)})\mathbf{K}\operatorname{diag}(\boldsymbol{v}^{(l)})$ and $\mathbf{P}^{(2l+1)} = \operatorname{diag}(\boldsymbol{u}^{(l+1)})\mathbf{K}\operatorname{diag}(\boldsymbol{v}^{(l)})$, and recall that $\boldsymbol{a} \odot \boldsymbol{b} \overset{\text{def.}}{=} \operatorname{diag}(\boldsymbol{a})\boldsymbol{b}$ is the component-wise multiplication of vectors. Then, we can simply the following terms:

$$\begin{aligned}\mathbf{P}^{(2l)}\mathbb{1} &= \operatorname{diag}(\boldsymbol{u}^{(l)})\mathbf{K}\operatorname{diag}(\boldsymbol{v}^{(l)})\mathbb{1}\\ &= \operatorname{diag}(\boldsymbol{u}^{(l)})\mathbf{K}\boldsymbol{v}^{(l)}\\ &= \boldsymbol{u}^{(l)} \odot \mathbf{K}\boldsymbol{v}^{(l)}\end{aligned} \tag{2.12}$$

$$\begin{aligned}{\mathbf{P}^{(2l+1)}}^T\mathbb{1} &= \left(\operatorname{diag}(\boldsymbol{u}^{(l+1)})\mathbf{K}\operatorname{diag}(\boldsymbol{v}^{(l)})\right)^T\mathbb{1}\\ &= \operatorname{diag}(\boldsymbol{v}^{(l)})^T\mathbf{K}^T\operatorname{diag}(\boldsymbol{u}^{(l+1)})^T\mathbb{1}\\ &= \boldsymbol{v}^{(l)} \odot \mathbf{K}^T\boldsymbol{u}^{(l+1)}.\end{aligned} \tag{2.13}$$

Equation (2.10) thus becomes

$$\begin{aligned}\operatorname{diag}(\boldsymbol{u}^{(l+1)})\mathbf{K}\operatorname{diag}(\boldsymbol{v}^{(l)}) &= \operatorname{diag}\left(\frac{\boldsymbol{a}}{\boldsymbol{u}^{(l)} \odot \mathbf{K}\boldsymbol{v}^{(l)}}\right)\operatorname{diag}(\boldsymbol{u}^{(l)})\mathbf{K}\operatorname{diag}(\boldsymbol{v}^{(l)})\\ \Rightarrow \operatorname{diag}(\boldsymbol{u}^{(l+1)}) &= \operatorname{diag}\left(\frac{\boldsymbol{a}}{\boldsymbol{u}^{(l)} \odot \mathbf{K}\boldsymbol{v}^{(l)}}\right)\operatorname{diag}(\boldsymbol{u}^{(l)})\\ \Rightarrow \boldsymbol{u}^{(l+1)} &= \frac{\boldsymbol{a}}{\mathbf{K}\boldsymbol{v}^{(l)}}.\end{aligned}$$

By a similar calculation for (2.11), we obtain the following scheme, which is known by many names in the literature (see [PC18, Remark 4.4] for a historical review), among which, the *Sinkhorn algorithm*:

$$\begin{aligned}\boldsymbol{u}^{(l+1)} &= \frac{\boldsymbol{a}}{\mathbf{K}\boldsymbol{v}^{(l)}}\\ \boldsymbol{v}^{(l+1)} &= \frac{\boldsymbol{b}}{\mathbf{K}^T\boldsymbol{u}^{(l+1)}},\end{aligned} \tag{2.14}$$

with $\boldsymbol{v}^{(0)} = \mathbb{1}_N$. The corresponding algorithm is detailed in Algorithm 1.

This simplification is a major advantage of this method, as it does not require the computation or storage of the full coupling matrix $\mathbf{P}$. Moreover, it only performs matrix-vector products and element-wise multiplications and divisions, which is highly parallelizable, as shown at the end of this section.

The regularized Wasserstein distance $\mathrm{W}_{\mathbf{C}}^{\varepsilon}(\boldsymbol{a}, \boldsymbol{b})$ can also be computed without constructing the entire coupling matrix, relying only on the scaling vectors and the input histograms. From (2.5), we have that $\log \mathbf{P} = \log \boldsymbol{u}\mathbb{1}^T + \log \mathbf{K} + \mathbb{1} \log \boldsymbol{u}^T$. Then, (2.3) becomes

$$\begin{aligned}
\langle \mathbf{C}, \mathbf{P} \rangle - \varepsilon \mathrm{H}(\mathbf{P}) &= \langle -\varepsilon \log \mathbf{K}, \mathbf{P} \rangle + \varepsilon \langle \mathbf{P}, \log \mathbf{P} \rangle \\
&= \varepsilon \langle \mathbf{P}, \log \mathbf{P} - \log \mathbf{K} \rangle \\
&= \varepsilon \left\langle \mathbf{P}, \log \boldsymbol{u}\mathbb{1}^T + \mathbb{1} \log \boldsymbol{v}^T \right\rangle \\
&= \varepsilon \left\langle \mathbf{P}, \log \boldsymbol{u}\mathbb{1}^T \right\rangle + \left\langle \mathbf{P}, \mathbb{1} \log \boldsymbol{v}^T \right\rangle \\
&= \varepsilon \langle \mathbf{P}\mathbb{1}, \log \boldsymbol{u} \rangle + \left\langle \mathbb{1}^T \mathbf{P}, \log \boldsymbol{v}^T \right\rangle \\
&= \varepsilon \langle \mathbf{P}\mathbb{1}, \log \boldsymbol{u} \rangle + \left\langle \mathbf{P}^T \mathbb{1}, \log \boldsymbol{v} \right\rangle .
\end{aligned}$$

At convergence, the marginal constraints are satisfied, therefore the regularized Wasserstein distance, also called the Sinkhorn distance, can be approximated through L iterations:

$$\mathrm{W}_{\mathbf{C}}^{\varepsilon\,(L)}(\boldsymbol{a}, \boldsymbol{b}) = \varepsilon \left(\left\langle \boldsymbol{a}, \log(\boldsymbol{u}^{(L)}) \right\rangle + \left\langle \boldsymbol{b}, \log(\boldsymbol{v}^{(L)}) \right\rangle \right). \tag{2.15}$$

Algorithm 1 `Sinkhorn`: Computation of the entropy-regularized Wassersein distance $\mathrm{W}_{\mathbf{C}}^{\varepsilon}(\boldsymbol{a}, \boldsymbol{b})$

Input: Histograms $\boldsymbol{a}, \boldsymbol{b} \in \Sigma_N$, kernel $\mathbf{K} = \exp(-\mathbf{C}/\varepsilon) \in \mathbb{R}^{N \times N}$, number of iterations L
$\boldsymbol{v}^{(0)} = \mathbf{1}_N$
for $l = 1 \dots L$ **do**
  $\boldsymbol{u}^{(l)} = \boldsymbol{a} / \mathbf{K}\boldsymbol{v}^{(l-1)}$
  $\boldsymbol{v}^{(l)} = \boldsymbol{b} / \mathbf{K}^T \boldsymbol{u}^{(l)}$
end for
Output: $\mathrm{W}_{\mathbf{C}}^{\varepsilon\,(L)}(\boldsymbol{a}, \boldsymbol{b}) = \varepsilon \left(\left\langle \boldsymbol{a}, \log(\boldsymbol{u}^{(L)}) \right\rangle + \left\langle \boldsymbol{b}, \log(\boldsymbol{v}^{(L)}) \right\rangle \right)$

Influence of the regularization parameter

In Figure 2.1, we show how the regularization parameter ε controls the entropy (smoothness) of the transport plan. The left-most image is obtained by setting ε high enough, such that $\mathbf{P}$ reaches maximum entropy: $h(\mathbf{P}) = h(\boldsymbol{a}) + h(\boldsymbol{b})$. It is then equivalent to the contingency table of two independent variables distributed according to $\boldsymbol{a}$ and $\boldsymbol{b}$: $\mathbf{P} = \boldsymbol{a}\boldsymbol{b}^T$. The following images show how the entropy (diffuseness) of the transport plan decreases as ε tends to 0. The right-most image is the true transport plan $\mathbf{P}^*$ obtained by solving the LP (2.2) with a network simplex

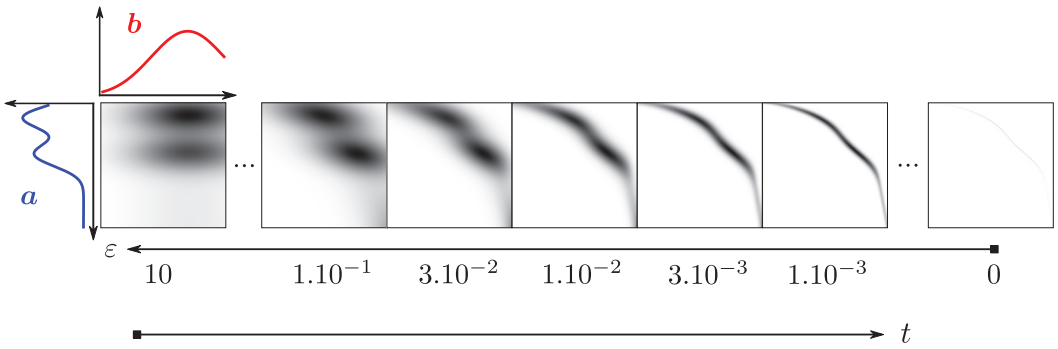


Figure 2.1: Effect of entropic regularization on the transport plan, depending on ε. Large values of ε result in a transport plan equivalent to the outer-product $\boldsymbol{a}\boldsymbol{b}^T$ (far-left). As ε decreases, the approximation gets closer and closer to the exact transport plan (far-right), but it requires more and more iterations (thus more time) since convergence decreases with ε.

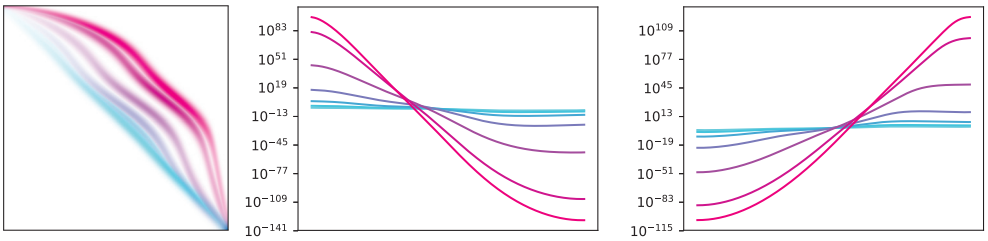
algorithm [Bon+11], which corresponds to $\varepsilon = 0$ in (2.3), and the state of minimal entropy.

It is important to note that as ε decreases, the Sinkhorn algorithm is slower to converge. The convergence of the Sinkhorn algorithm can be estimated by looking at the marginal constraint violation, which we measure as a relative error:

$$\epsilon^{(l)} = \frac{\left|\left|\mathbf{P}^{(l)}\mathbb{1} - \boldsymbol{a}\right|\right|_2}{||\boldsymbol{a}||_2} = \frac{\left|\left|\boldsymbol{u}^{(l)}\mathbf{K}\boldsymbol{v}^{(l)} - \boldsymbol{a}\right|\right|_2}{||\boldsymbol{a}||_2}. \tag{2.16}$$

Empirically, we observe that $\epsilon^{(L)} = O(\varepsilon L)$, which means that when decreasing ε, the number of iterations L necessary to reach the same level of convergence grows in $O(\varepsilon^{-1})$. The convergence of the algorithm is therefore independent of the histogram size N, as demonstrated in [Cut13, §5]. In this empirical study, Cuturi [Cut13] also argues that in practice, it is sufficient to set a fixed number L of iterations, rather than choosing a stopping criterion that depends on a convergence measure such as $\epsilon^{(l)}$ or the relative reduction of the energy.

A practical issue with the Sinkhorn algorithm is that numerical instability increases as ε decreases. Figure 2.2 illustrates the evolution of the transport plan and the scaling vectors $\boldsymbol{u}$ and $\boldsymbol{v}$ during the Sinkhorn algorithm. We see that for $\varepsilon = 1.10^{-3}$, after 5000 iterations the range of values in the scalings is $[10^{-141}, 10^{109}]$. In the example of Figure 2.1, numerical limits (i.e. division by zero which leads to NaN) are reached in the scaling vectors, for $\varepsilon = 3.10^{-4}$ and $L = 500$. This issue can however be addressed by conducting the Sinkhorn iterations in the log domain, as mentioned by Benamou et al. [Ben+15] and implemented in Schmitzer [Sch16].



(a) Transport plan **(b)** Scaling $\boldsymbol{u}$ **(c)** Scaling $\boldsymbol{v}$

Figure 2.2: Evolution of the transport plan and scaling vectors $\boldsymbol{u}$ and $\boldsymbol{v}$ during Sinkhorn iterations (from blue to red), for logarithmically spaced iterations: $\{4, 17, 71, 292, 1209, 5000\}$, and $\varepsilon = 1.10^{-3}$.

To conclude, tuning the parameter ε is a trade-off between the coarseness of the approximation, the diffusiveness of the optimal transport plan, and the computational effort required to solve the problem.

Links to the KL divergence

The Kullback-Leibler divergence, also known as *relative entropy* or *information gain*, between coupling matrices is defined as

$$\mathrm{KL}(\mathbf{P}|\mathbf{K}) \overset{\text{def.}}{=} \sum_{i,j} \mathbf{P}_{i,j} \left(\log\left(\frac{\mathbf{P}_{i,j}}{\mathbf{K}_{i,j}} \right) - 1 \right). \tag{2.17}$$

Problem (2.3) can be rewritten as a KL projection of the kernel $\mathbf{K}$ on the constraint polytope $\Pi(\boldsymbol{a}, \boldsymbol{b})$:

$$\begin{aligned}
\mathrm{W}_{\mathbf{C}}^{\varepsilon}(\boldsymbol{a}, \boldsymbol{b}) &= \min_{\mathbf{P} \in \Pi(\boldsymbol{a}, \boldsymbol{b})} \langle \mathbf{P}, \mathbf{C} \rangle - \varepsilon \mathrm{H}(\mathbf{P}) \\
&= \min_{\mathbf{P} \in \Pi(\boldsymbol{a}, \boldsymbol{b})} \sum_{i,j} \mathbf{P}_{i,j} \mathbf{C}_{i,j} + \varepsilon \left(\mathbf{P}_{i,j} \log(\mathbf{P}_{i,j}) - \mathbf{P}_{i,j} \right) \\
&= \min_{\mathbf{P} \in \Pi(\boldsymbol{a}, \boldsymbol{b})} \sum_{i,j} \mathbf{P}_{i,j} (-\varepsilon \log(\mathbf{K}_{i,j})) + \varepsilon \left(\mathbf{P}_{i,j} \log(\mathbf{P}_{i,j}) - \mathbf{P}_{i,j} \right) \\
&= \min_{\mathbf{P} \in \Pi(\boldsymbol{a}, \boldsymbol{b})} \varepsilon \sum_{i,j} \mathbf{P}_{i,j} \log(\frac{\mathbf{P}_{i,j}}{\mathbf{K}_{i,j}}) - \mathbf{P}_{i,j} \\
&= \min_{\mathbf{P} \in \Pi(\boldsymbol{a}, \boldsymbol{b})} \varepsilon \, \mathrm{KL}(\mathbf{P}|\mathbf{K}).
\end{aligned} \tag{2.18}$$

The optimal coupling $\mathbf{P}_{\varepsilon}^{*}$ is therefore the one that minimizes the information gained from $\mathbf{K}$ to $\mathbf{P}$, while still lying in the convex polytope $\Pi(\boldsymbol{a}, \boldsymbol{b})$.

The admissibility set $\Pi(\boldsymbol{a}, \boldsymbol{b})$ can be written as the intersection of two convex sets:

$$\mathcal{C}_1 \overset{\text{def.}}{=} \left\{ \mathbf{P} \in \mathbb{R}_+^{N \times N} \;\middle|\; \mathbf{P}\mathbb{1}_N = \boldsymbol{a} \right\} \text{ and } \mathcal{C}_2 \overset{\text{def.}}{=} \left\{ \mathbf{P} \in \mathbb{R}_+^{N \times N} \;\middle|\; \mathbf{P}^T\mathbb{1}_N = \boldsymbol{b} \right\}. \quad (2.19)$$

As mentioned in [Ben+15], starting from $\mathbf{P}^{(0)} = \mathbf{K}$, iteratively performing Bregman projections (for the KL divergence) of $\mathbf{P}$ on the two convex sets $\mathcal{C}_1$ and $\mathcal{C}_2$ converges to the optimal solution of (2.18). This is in essence what the Sinkhorn algorithm (2.14) does.

Parallelization

The Sinkhorn algorithm can easily be generalized to compute multiple distances at once. Indeed, considering there are R distances to compute, one only needs to stack the scaling vectors into $N \times R$ matrices $\mathbf{U}$ and $\mathbf{V}$, stack the marginals into matrices $\mathbf{A}$ and $\mathbf{B}$, and obtain the parallel Sinkhorn algorithm:

$$\begin{aligned} \mathbf{U}^{(l)} &= \frac{\mathbf{A}}{\mathbf{K}\mathbf{V}^{(l-1)}} \\ \mathbf{V}^{(l)} &= \frac{\mathbf{B}}{\mathbf{K}^T\mathbf{U}^{(l)}}, \end{aligned} \quad (2.20)$$

with $\mathbf{U}^{(0)} = \mathbb{1}_{N \times R}$. All operations are highly parallelizable, which makes the algorithm very efficient when implemented on a multicore CPU or on a GPU.

Kernel applications and separability

As noted in [Sol+15], when histograms lie on a Euclidean grid and the transport cost is the squared Euclidean distance, a special structure arises in the kernel $\mathbf{K}$ which allows for faster applications of it. Since $\mathbf{K} = \exp(-\mathbf{D}^2/\varepsilon)$, the application to a vector $\mathbf{K}\boldsymbol{b}$ can be replaced by a convolution with a Gaussian kernel of standard deviation $\sigma = \sqrt{\varepsilon}$. This avoids computing and storing the full kernel $\mathbf{K}$ of size N^2, which would be prohibitive in memory for large-scale applications.

Moreover, the squared Euclidean distance is separable, meaning that it can be separated along each axis:

$$c(\boldsymbol{x}, \boldsymbol{y}) = ||\boldsymbol{x} - \boldsymbol{y}||_2^2 = \sum_{i=1}^{\mathrm{d}} (\boldsymbol{x}_i - \boldsymbol{y}_i)^2 = \sum_{i=1}^{\mathrm{d}} c_i(\boldsymbol{x}_i, \boldsymbol{y}_i). \quad (2.21)$$

The separability of the cost into a sum makes the kernel separable into a product of 1-dimensional kernels. We can then simply apply each of those d kernels sequentially. This reduces computation time when histograms have dimension $\mathrm{d} > 1$.

We illustrate this process on a 2-dimensional grid of size $n \times n$. We use different notations than the vector and matrix ones used so far, for the sake of generality. Applying a kernel of the form $K = \exp\left(-\frac{c}{\varepsilon}\right)$ to a 2-D function b is performed as such:

$$r_2(i,j) \overset{\text{def.}}{=} \sum_{k=1}^{n}\sum_{l=1}^{n} \exp\left(-\frac{c((i,j),(k,l))}{\varepsilon}\right) b(k,l).$$

Assuming a separable cost such that $c((i,j),(k,l)) \overset{\text{def.}}{=} c_y(i,k) + c_x(j,l)$, this operation amounts to performing two 1-dimensional kernel applications:

$$\begin{aligned} r_1(k,j) &= \sum_{l=1}^{n} \exp\left(-\frac{c_x(j,l)}{\varepsilon}\right) b(k,l) \\ r_2(i,j) &= \sum_{k=1}^{n} \exp\left(-\frac{c_y(i,k)}{\varepsilon}\right) r_1(k,j). \end{aligned}$$

For a d-dimensional histogram of $N = n^{\mathrm{d}}$ bins, applying a separable kernel amounts to performing a sequence of d steps, where each step computes n operations per point. This results in a time complexity in $O(n^{\mathrm{d}+1}) = O(N^{\frac{\mathrm{d}+1}{\mathrm{d}}})$ instead of $O(N^2)$.

Extensions

Recently, Altschuler et al. [Alt+18] introduced a method to accelerate the Sinkhorn algorithm. Simultaneously, there have been considerable efforts to study the convergence and approximation properties of the Sinkhorn algorithm [Alt+17; CK18] and its variances [Dvu+18].

2.1.3 Displacement interpolation

Given two histograms $\boldsymbol{a}$ and $\boldsymbol{b}$, $W_{\mathbf{C}}(\boldsymbol{a},\boldsymbol{b})$ not only represents the total cost of transporting one to the other, but also a distance, which is the length of the shortest path (a geodesic) between them in the Wasserstein space.

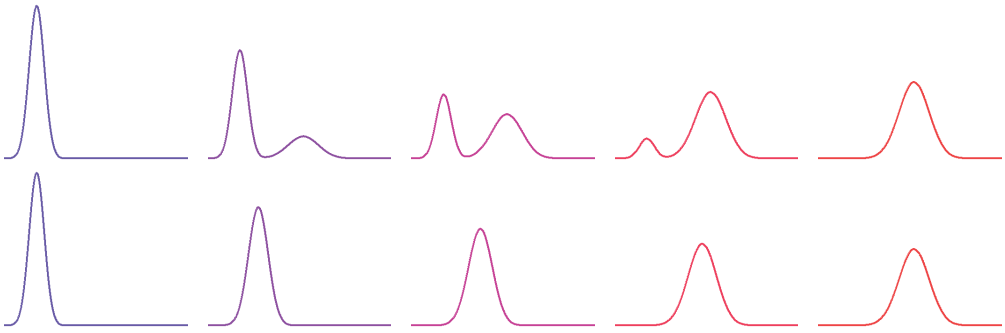


Figure 2.3: Comparison of Euclidean (top row) and displacement interpolation (bottom row) for 1-dimensional histograms

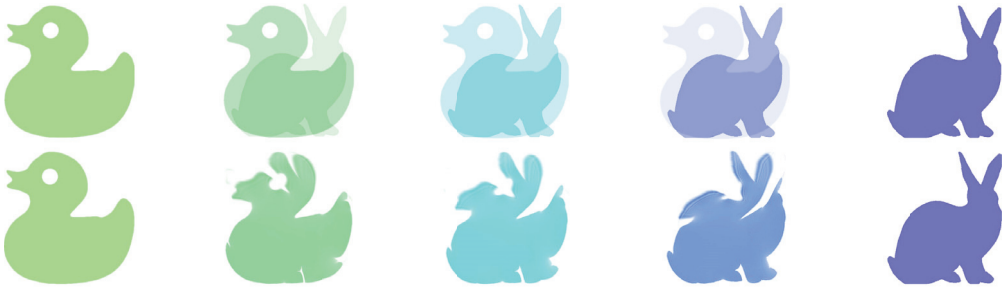


Figure 2.4: Comparison of Euclidean (top row) and displacement interpolation (bottom row) for 2-dimensional histograms (images)

A *displacement interpolation* [McC97] is a way of walking along the geodesic from $\boldsymbol{a}$ to $\boldsymbol{b}$. It is defined for $t \in [0,1]$ as

$$\boldsymbol{\gamma}_{\mathbf{C}}(\boldsymbol{a},\boldsymbol{b},t) \overset{\text{def.}}{=} \min_{\boldsymbol{r}\in\Sigma_N} (1-t)W_{\mathbf{C}}(\boldsymbol{a},\boldsymbol{r}) + tW_{\mathbf{C}}(\boldsymbol{r},\boldsymbol{b}). \tag{2.22}$$

Contrary to its Euclidean counterpart, such an interpolation allows *lateral* displacements of mass on the domain, as seen in Figure 2.3 and Figure 2.4. This ability to take into account the geometry of the domain is one of the reasons OT has been adopted in many scientific communities to handle probability distributions.

In practice, we can approximate displacement interpolations using regularized Wasserstein barycenters (see next section). We denote such approximations by $\boldsymbol{\gamma}^{\varepsilon}_{\mathbf{C}}$.

2.1.4 Wasserstein barycenters

The Fréchet mean is a generalization of the weighted Euclidean average to metric spaces, that is, the point that minimizes the weighted sum of squared distances to each point of a set. The Wasserstein barycenter, introduced by Agueh and Carlier

[AC11], follows that definition and is a weighted average of histograms in the Wasserstein space. It is also the generalization of the displacement interpolation to more than 2 histograms. Let $A = \boldsymbol{a}_1, \ldots, \boldsymbol{a}_S \in \Sigma_N$ be a family of histograms, and $\boldsymbol{\lambda} \in \Sigma_S$ be weights associated to each of them, the Wasserstein barycenter is defined as

$$\boldsymbol{p}_{\mathbf{C}}\left(A, \boldsymbol{\lambda}\right) \overset{\text{def.}}{=} \underset{\boldsymbol{b} \in \Sigma_N}{\operatorname{argmin}} \sum_{s=1}^{S} \lambda_s W_{\mathbf{C}}(\boldsymbol{a}_s, \boldsymbol{b}). \tag{2.23}$$

This definition assumes that all histograms lie on the same grid. Otherwise, separate cost matrices $\mathbf{C}_s$ must be used for each distance, and care must be taken to optimize the positions of the barycenter's masses [CD14; Ben+15; Ye+17; Cla+18]. When all weights are equal ($\boldsymbol{\lambda}_s = 1/S$), we call the barycenter *centroid*.

Cuturi and Doucet [CD14] introduced the regularized counterpart of the Wasserstein barycenter, which is the building block of the learning algorithms we developed. The entropy-regularized Wasserstein barycenter of the family A, with weights $\boldsymbol{\lambda}$ is defined as

$$\boldsymbol{p}^{\varepsilon}_{\mathbf{C}}\left(A, \boldsymbol{\lambda}\right) \overset{\text{def.}}{=} \underset{\boldsymbol{b} \in \Sigma_N}{\operatorname{argmin}} \sum_{s=1}^{S} \lambda_s W^{\varepsilon}_{\mathbf{C}}(\boldsymbol{a}_s, \boldsymbol{b}). \tag{2.24}$$

When the metric $\mathbf{C}$ is Euclidean, we simply denote the regularized barycenter as $\boldsymbol{p}_{\varepsilon}\left(A, \boldsymbol{\lambda}\right)$.

This new formulation yields a smooth and convex functional, which can be solved with a generalization of the Sinkhorn algorithm [Ben+15]: let $\boldsymbol{v}_s^{(0)} = \mathbb{1}_N, \forall s \in [1, \ldots, S]$,

$$\begin{aligned}
\boldsymbol{u}_s^{(l)} &= \frac{\boldsymbol{a}_s}{\mathbf{K}\boldsymbol{v}_s^{(l-1)}}, \quad \forall s \in \{1, \ldots, S\} \\
\boldsymbol{b}^{(l)} &= \prod_{s=1}^{S} (\mathbf{K}^T \boldsymbol{u}_s^{(l)})^{\lambda_s} \\
\boldsymbol{v}_s^{(l)} &= \frac{\boldsymbol{b}^{(l)}}{\mathbf{K}^T \boldsymbol{u}_s^{(l)}}, \quad \forall s \in \{1, \ldots, S\}.
\end{aligned} \tag{2.25}$$

We denote $\boldsymbol{p}_{\varepsilon}^{(L)} = \boldsymbol{b}^{(L)}$ the approximate regularized barycenter after L iterations, as can be done for regularized Wasserstein distances (see 2.1.2). These iterations implicitly optimize S coupling matrices $\mathbf{P}_s = \operatorname{diag}(\boldsymbol{u}_s)\mathbf{K}\operatorname{diag}(\boldsymbol{v}_s)$, each describing the transport between histogram $\boldsymbol{a}_s$ and the barycenter $\boldsymbol{b}$. The corresponding algorithm is detailed in Algorithm 2. We show examples of barycenters of different

shapes in Figure 2.5, for the 2-D and 3-D setting. We also demonstrate the smoothing effect of the entropic regularization on the barycenters in Figure 2.6.

Algorithm 2 `SinkhornBarycenter`: Computation of the entropy-regularized Wasserstein barycenter $\boldsymbol{p}_\varepsilon(A, \boldsymbol{\lambda})$

Input: Histograms $A = \boldsymbol{a}_1, \dots, \boldsymbol{a}_S \in \Sigma_N$, weights $\boldsymbol{\lambda} \in \Sigma_S$, kernel $\mathbf{K} = \exp(-\mathbf{C}/\varepsilon) \in \mathbb{R}^{N \times N}$, number of iterations L
$\boldsymbol{v}^{(0)} = \mathbf{1}_N$
for $l = 1 \dots L$ **do**
  $\forall s, \boldsymbol{q}_s^{(l)} = \mathbf{K}^T \left(\frac{\boldsymbol{a}_s}{\mathbf{K}\boldsymbol{v}_s^{(l-1)}} \right)$
  $\boldsymbol{b}^{(l)} = \prod_s (\boldsymbol{q}_s^{(l)})^{\lambda_s}$
  $\forall s, \boldsymbol{v}_s^{(l)} = \frac{\boldsymbol{b}^{(l)}}{\boldsymbol{q}_s^{(l)}}$
end for
Output: Barycenter $\boldsymbol{p}_\varepsilon^{(L)} = \boldsymbol{b}^{(L)}$

There are many alternatives to compute Wasserstein barycenters. Some of them use descent algorithms on the dual problem, either through non-smooth optimization [Car+15], or by smoothing the problem [CP15; CP18] with entropic regularization. Others use semi-discrete formulations to compute barycenters of continuous distributions i.e. for which samples can be drawn [Sta+17; Cla+18], with applications in large-scale Bayesian inference, super-sampling and blue-noise approximations. Finally, recent methods have been developed to compute sharper barycenters than what the Sinkhorn barycenter can do, using proximal point methods [Xie+18], or a new formulation of the gradient of the sharp Sinkhorn loss [Lui+18]. However, both methods require the full computation of the transport plan or the kernel, which is prohibitive in large-scale applications.

2.1.5 Other numerical methods for optimal transport

Solving OT problems numerically is not restricted to the linear program and the entropic regularization. There exist multiple formulations of optimal transport, leading to a multitude of methods, specific to parameters such as the dimension of the data, the nature of the distributions, the types of problems, or the cost function. In this section, we briefly present the main formulations of optimal transport.

As mentioned in 1.1.3, when dealing with two discrete distributions with the same number of Dirac masses and uniform weights, optimal transport boils down to an assignment problem, which can be solved with combinatorial algorithms, such as the Hungarian algorithm [Kuh55], or the auction algorithm [Ber81]. Other schemes

(a) Barycenters of 2-dimensional shapes **(b)** Barycenters of 3-dimensional shapes

Figure 2.5: Examples of Wasserstein barycenters of 2 and 3-dimensional shapes. Extracted from Peyré and Cuturi [PC18]. The shapes displayed here are thresholded densities.

Figure 2.6: Influence of the entropic regularization for barycenters in dimension 2. Barycentric interpolation between 2 distributions (on each row), for different values of ε, top to bottom $3.10^{-4}, 1.10^{-3}, 3.10^{-3}, 1.10^{-2}$, and $L = 200$ iterations.

have been developed to accelerate the computation for particular settings or cost functions [LO07; Del+10].

The semi-discrete setting involves the computation of optimal transport between a continuous measure and a discrete one. It is closely related to the geometrical concept of Laguerre diagrams [Aur+98], which are weighted versions of Voronoi diagrams. The semi-discrete problem can be solved with convex optimization in 2-D [Mér11] and 3-D [Lév15]. Contrary to entropic regularization which finds an approximation of the transport, these algorithms solve the exact problem, which can be necessary for some applied inverse problems, such as caustic design [Mey+18]. However, existing algorithms only solve the problem for standard cost functions, such as the squared Euclidean distance, but not for arbitrary ones. Applications include shape interpolation [Lév15], fluid simulation [GM18], or early-universe reconstruction [Bre+03].

The dynamic formulation of optimal transport, initiated by Benamou and Brenier [BB00], searches for the optimal transport as a geodesic in the Wasserstein space. It can be transformed into a fluid mechanics formulation, where we seek a velocity field transporting one measure to the other, and all intermediate densities. The conservation of mass between the two input measures is enforced by restricting the velocity field to be incompressible. Such a scheme is solved through the discretization on a grid, and first-order convex non-smooth optimization schemes, based on the proximal operator. The addition of the time dimension in the equation makes the problem computationally involved. More details can be found in the review article on dynamical OT problems by Papadakis et al. [Pap+14].

The sliced paradigm computes Wasserstein distances between high-dimensional densities, by computing the 1-D transport between their projections on random 1-dimensional lines (slices). For 1-dimensional densities, the Wasserstein distance is computable in closed-form through their inverse cumulative density functions. The numerical algorithm consists in approximating the integral over all possible directions of the 1-D transport via Monte-Carlo integration. This formulation is a new metric between measures based on OT, rather than an approximation of the true Wasserstein distance: the optimality of the transport between slices does not guarantee the optimality of the transport in the higher-dimensional space. However, the two distances behave similarly in several cases, which makes sliced OT a competitive alternative. Sliced OT has been used mainly in computer graphics [Pit+05; Rab+11; Bon+15; BC19] and machine learning [Kol+16; Car+17; Des+18a].

Another set of approaches to solve optimal transport numerically is based on stochastic optimization [Gen+16; Arj+17; Bou+17; Gen+17]. It is used for large-scale OT

problems, i.e. when dealing with large amounts of high-dimensional data. These approaches have been especially useful for training generative adversarial networks (see also 2.1.6), and rely on reformulations of the transport problem, cast as expectation maximization problems. However, these methods are designed to compare high-dimensional continuous distributions, which means they are discretized with a Lagrangian approach, and we can only access them by sampling from them.

We refer to the books of Peyré and Cuturi [PC18] and Santambrogio [San15] for extensive surveys on computational optimal transport.

2.1.6 Applications of optimal transport in computer graphics and machine learning

Optimal transport appeared in machine learning and computer graphics through computer vision. Rubner et al. [Rub+00] used the Wasserstein distance as a dissimilarity measure between distributions of texture descriptors and color, to perform an image retrieval task via clustering. At the time, the best solver available for OT was the transportation simplex, whose complexity limited applications to small histograms (less than 300 samples). Fortunately, the last 10 years have seen many new numerical methods to solve OT problems, which has led to fruitful investigations in the aforementioned research areas.

Computer graphics The displacement interpolation is a popular tool for its ability to warp, deform, and transport distributions in a meaningful way, depending on the chosen cost function. Such interpolations have been used for shapes [Sol+15; Lév15], BRDFs [Bon+11], and distributions on discrete surfaces [Lav+18]. A recurrent application of OT is color transfer, with the seminal work of Pitié and Kokaram [PK07], because the transport plan precisely defines a mapping between histogram bins. It has then been extended to reduce artifacts [Rab+10], or improve robustness to mass variation [Fer+13]. Because color spaces are 2 or 3-dimensional and require fine discretizations to avoid quantization errors, early works relied on approximations such as 1D projections [Rab+10; Bon+15], or worked on coarse discretizations [Fer+13]. The advent of entropic regularization led to consider large-scale histograms in 2 and 3 dimensions [Sol+15; Bon+16]. Color transfer has served as a demonstration application for new numerical frameworks to solve OT problems [Blo+18; KT17; BC19], in particular barycenters [Pap+11; Fer+13; Bon+15; Bon+16], because they allow for color normalization (or *midway histogram equalization* [Del04]) between multiple images, and for transferring colors of a

family of images to a single target. Optimal transport has been used in medical imaging, to define a distance between brain MRIs [GM17], to average brain MRIs [Bon+16], for image registration and warping [Hak+04], and for surface and fiber registration [Fey+17]. Other applications include texture synthesis and mixing [Rab+11; Bon+15; Tar+16], gradient flows [Pey15; CP15; Ben+16; Sch16], image editing [Per+16], image segmentation [RP15; SS13], image denoising [Lel+14], caustic design [Mey+18] and fluid simulation [GM18]. A sample of visuals from different applications of OT to computer graphics is shown in Figure 2.7.

Machine learning Distributions are ubiquitous in machine learning, because they provide insight into data, holding more information than aggregate quantities such as the mean, or the standard deviation. Because optimal transport compares distributions in a way that is sensitive to the underlying geometry, it has been adopted as a loss function to replace distances and divergences that cannot handle shifted patterns, such as L^2 or KL. Moreover, the entropic regularization [Cut13] has helped the development of learning algorithms involving OT, since it converts the original Wasserstein distance into a fast, smooth, differentiable, and more robust loss. The Wasserstein loss is used for example in representation learning, via matrix factorization [SL11; Zen+14], dictionary learning [Rol+16], discriminant analysis [Fla+18], or auto-encoders [Cou+17].

The Wasserstein distance has also been used for generative models, where it replaces information-based divergences (KL, Jensen-Shannon) classically used to measure the goodness of fit between a parametric distribution and the empirical distribution of the observations. The idea of using the Wasserstein distance for minimum distance estimation and density fitting dates back to the theoretical work of [Bas+06]. Recent examples include Boltzmann machines [Mon+16], statistical inference [Ber+17] and generative adversarial networks [Arj+17]. Various generative models have also been used to motivate the use of different formulations and approximations of the Wasserstein distance, improving tractability [Gen+17], convergence [Seg+18], stability [Des+18a] or accuracy [Xie+18].

OT is also used for domain adaptation, where the quantity of interest is the transport plan. Indeed, OT can define a mapping between the source and target data distribution, in order to assign labels in the target dataset from those in the source one [Cou+14; Per+16; Cou+16].

A common representation of text in machine learning and natural language processing (NLP) is the *bag-of-words* model, which considers texts or documents as multisets (unordered, allowed repetitions) of words. A histogram of words can

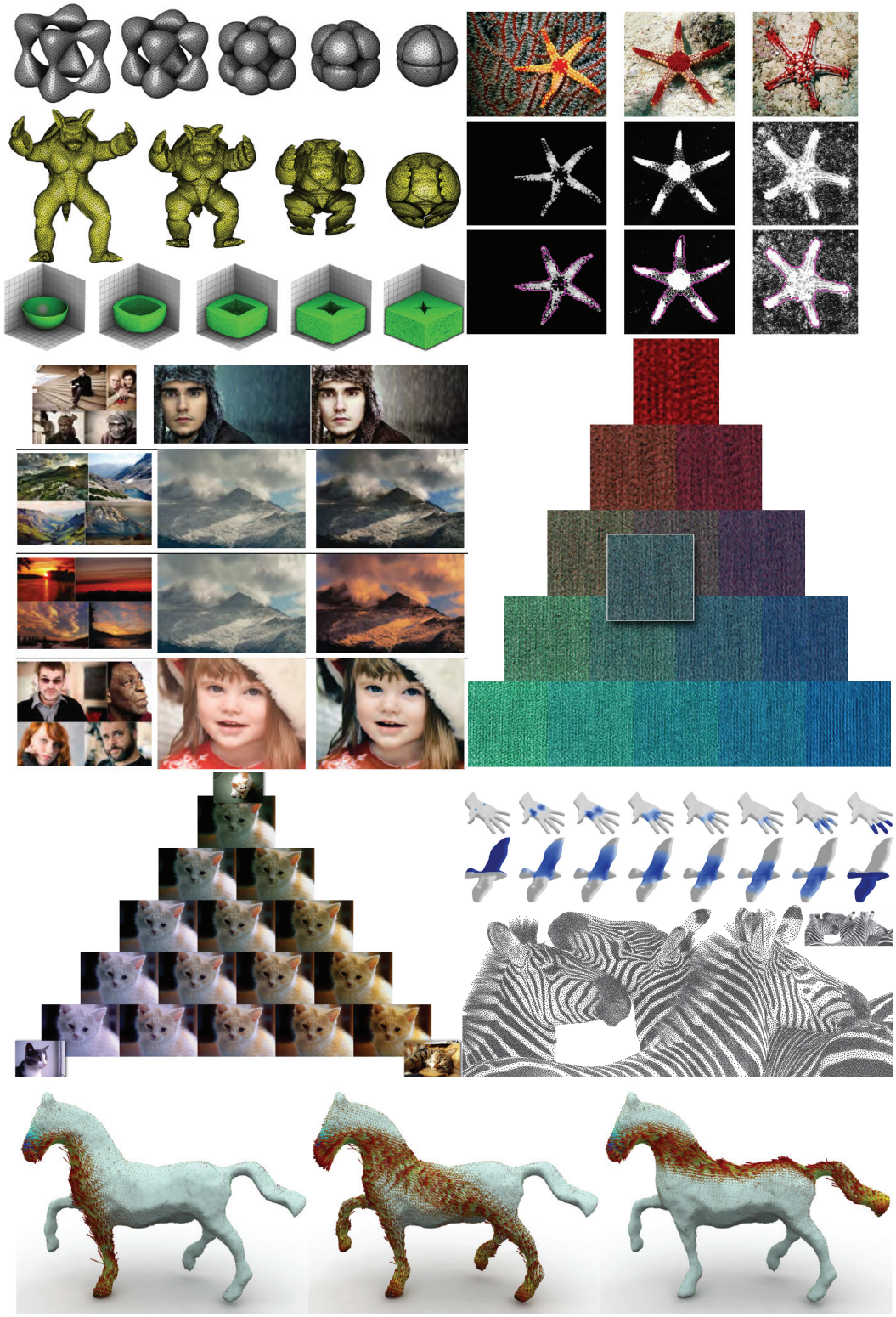


Figure 2.7: A sample of applications of optimal transport to computer graphics, extracted from different publications. Top to bottom, left to right : shape interpolation [LS17], image segmentation [SS13], color transfer [Bon+16], texture synthesis and color transfer [Bon+15], interpolation on discrete surfaces [Lav+18], blue-noise approximation [de +12], and vector field interpolation [Sol+14].

then be assembled, giving the frequency of occurence of each word. Consequently, the histogram forms a sort of signature, or *feature* of its document, and it can be used for machine learning tasks, such as classification or clustering. The introduction of word embeddings in Euclidean space, based on contextual words [Mik+13; Pen+14] allowed the definition of a ground metric in the word space that is close to a semantic distance. These embeddings have been largely used as a ground metric for OT to compare distributions of words. Examples include comparing image tags [Fro+15], defining a distance (the *Word Mover's Distance*) between documents for k-NN-based classification [Kus+15; Hua+16], or finding *topics* through barycenters of documents [Rol+16]. Recent methods jointly learn the topics and the word embeddings [Xu+18a], or use a hierarchical model of documents, topics and words known as Latent Dirichlet Allocation (LDA) [Ble03], to handle large-scale datasets [Yur+19]. In LDA, documents are modeled as distributions over topics and topics as distributions over words. This model involves a sparse Dirichlet distribution prior on topics and documents, which translates the idea that documents cover a restricted number of topics, and topics comprise a restricted number of words.

In the next sections, we will further discuss the two machine learning tasks of interest in this dissertation, that is, dictionary learning and metric learning.

2.1.7 Inverse problems in optimal transport

Even though the frameworks that use OT for learning tasks can be seen as inverse problems, the OT loss has also been used for more "classical" inverse problems when the data are probability distributions. For instance, Indyk and Price [IP11] perform sparse recovery and compressed sensing under the EMD loss, and Burger et al. [Bur+12] regularize a density estimation problem using the Wasserstein metric as a data fidelity term. The OT loss has also proven useful in tomography as it accounts for misalignments and shifted patterns in the data. Karlsson and Ringh [KR17] introduce a general scheme to regularize inverse problems with the OT loss, based on a Sinkhorn-like proximal operator, and apply it to computerized tomography (CT scans). Adler et al. [Adl+17] also work on reconstructing CT scans using a Wasserstein loss, but they propose to learn a reconstruction operator with neural networks, based on a primal-dual scheme. Finally, Abraham et al. [Abr+17] attempt to use multi-marginal OT in order to reconstruct CT scans from only a few views. Other examples of tomography using the OT loss have been proposed in geophysics for full waveform inversion, using different formulations of the transport, namely the Kantorovich-Rubinstein norm [Mét+16], or the Monge-Ampère equation [Eng+16].

Most of these methods only use OT as the loss function to compare the output of the model with the ground truth. In this thesis, we are more interested in *nested* inverse problems, i.e. ones that optimize quantities that are already built upon the Wasserstein distance, such as Wasserstein barycenters. For instance, Bonneel et al. [Bon+16] search for histogram projections on a histogram simplex, through the regression of the Wasserstein barycentric coordinates. The framework we present in chapter 3 is in line with this method, since we additionally regress the vertices of the histogram simplex to obtain a non-linear (Wasserstein) dictionary learning method (WDL) [Sch+18]. Simou and Frossard [SF18] adapt WDL for signals supported on graphs, and apply it on small synthetic graphs. Dognin et al. [Dog+19] use Wasserstein barycenters for consensus computations in model ensembling methods, with applications in multilabel learning and classification. Xu et al. [Xu+18a] perform the joint learning of topics and word embeddings in a Wasserstein fashion, leading to an approach that combines dictionary (topic) learning with Wasserstein barycenters, and ground metric (word embedding) learning. Our second contribution which is detailed in chapter 4 also fits that category, since we optimize the ground metric with a functional that involves a displacement interpolation computed through Wasserstein barycenters. Finally, let us note the work of Yurochkin et al. [Yur+19], which involves two nested OT problems for document classification, based on the hierarchical LDA model mentioned earlier. However, the authors do not optimize the two problems jointly; they instead resort to a precomputation of the inner one.

2.2 Dictionary learning

In this section, we introduce the preliminaries and the context of our first contribution which is the Wasserstein dictionary learning algorithm presented in chapter 3. We first introduce the concept of dictionary learning (DL), and recall the previous work in its linear and non-linear form. We then detail DL methods that use optimal transport.

2.2.1 Linear dictionary learning

Dictionary learning belongs to the more general concept of representation learning (or feature learning), which encompasses all methods that search for a new representation of some raw data (i.e. new coordinates in a new basis or space), in order to improve a task such as classification, or clustering.

The linear version of dictionary learning seeks a matrix decomposition of the form

$$\mathbf{X} \approx \mathbf{D}\mathbf{\Lambda} \tag{2.26}$$

where:

- the input data $\mathbf{X} \in \mathbb{R}^{N \times M}$ contains M data points of dimension N
- the dictionary $\mathbf{D} \in \mathbb{R}^{N \times S}$ contains S atoms of dimension N
- the codes $\mathbf{\Lambda} \in \mathbb{R}^{S \times M}$ are M weight vectors of dimension S.

Therefore, DL aims to find a basis of elements that represents the input data, and weights that specify how to reconstruct each data point from that new basis.

Dictionary learning methods differ by the constraints they impose on the dictionary and the codes, and by the completeness of the dictionary. Methods that seek a complete or under-complete dictionary ($S \leq N$) are useful for dimensionality reduction. One of them is the well-known principal component analysis (PCA) [Pea01]. It can be seen as a matrix decomposition which constrains atoms to be orthogonal, and that maximizes the variance explained by each atom. Non-negative matrix factorization (NMF) [LS99] enforces all values in the dictionary and the codes to be positive, which leads to a parts-based decomposition. Independent component analysis (ICA) [HO00] seeks a dictionary of atoms that are statistically independent, which is often sought for blind source separation [ZP01].

Other methods consider that there are more independent causes that could explain a given data point, than its number of dimensions, therefore they learn an over-complete dictionary ($S > N$). Sparse coding [OF97; Lee+07] is one of these methods and adds the constraint that data points should be represented by very few atoms, which means that codes must be sparse. It contrasts with sparse approximation techniques, where signals are sparsely decomposed over a *predefined* dictionary, such as the Fourier or wavelet basis. Sparse PCA [D'a+05] on the other hand imposes sparsity constraints on the dictionary, leading to atoms that do not necessarily explain all the variance, or are not orthogonal, contrary to PCA. Nevertheless, this sort of decomposition is desirable when each dimension of the input data can be interpreted separately, and therefore atoms can be assigned to them more easily. K-SVD [Aha+06; Rub+08] is a generalization of the k-means algorithm, where K atoms are learned, and the sparsity constraint on the weights is less strict.

Dictionary learning methods have been successfully applied to many scientific domains, and have been extended many times. For example, Smaragdis and Raj [SR06] propose PLCA, which extends NMF to handle probability distributions, by

adding the constraints that each atom and each code must belong to the probability simplex (i.e. weights sum to one), and uses the KL divergence as a reconstruction loss. Mairal et al. [Mai+09] introduce a supervised dictionary learning method, where they learn a dictionary that is shared between classes and adapted to the classification task, and an indicator function for class association. Mairal et al. [Mai+10] adapt some of these classical DL methods for large-scale datasets, by proposing online learning versions based on stochastic optimization. Thanou et al. [Tha+14] develop a sparse coding algorithms for graph signals, effectively taking into account the irregular geometry of the domain. At the intersection of dictionary learning and metric learning, Wang et al. [Wan+12] propose a multi-metric learning, where a dictionary of base metrics is learned and each training instance has a metric that is a linear combination of the base metrics (see 2.3). For more details on sparse coding and dictionary learning, many surveys have been published for different methods or applications, and are well reported in [Xu+17].

2.2.2 Non-linear dictionary learning

Dictionary learning can be made non-linear by changing the linear combination to reconstruct the data points, by a non-linear function:

$$\mathbf{X} \approx f(\mathbf{D}, \mathbf{\Lambda}) \tag{2.27}$$

One of the most common methods to introduce non-linearity is to use the kernel method, or the "kernel trick". It consists in using a non-linear kernel function that sends data points to a higher-dimensional space in which a linear classifier could potentially separate classes. The role of the kernel is therefore to linearize the non-linearity that is inherent to the input data. Kernel methods have been successfully applied to dictionary learning methods such as PCA [Sch+97], K-SVD [Ngu+12], NMF [Lee+09] or supervised dictionary learning [Gan+13]. Manifold learning consists in accounting for the intrinsic geometric properties of data points that lie on a manifold when performing clustering or dimensionality reduction on them. Dictionary learning and sparse coding have been generalized to data on Riemannian manifolds [Xie+13], Grassmanian manifolds [Har+11] or the space of positive symmetric definite (PSD) matrices [Jay+13]. Another dimensionality reduction technique that has been extended to manifolds is PCA, with the principal geodesic analysis (PGA) [Fle+04]. It is a tangential approximation that consists in projecting data points on the tangent space at the Fréchet mean using the logarithmic map, and then computing a Euclidean PCA in that space.

2.2.3 OT-flavored dictionary learning

The PGA approach has been well-studied for probability distributions in the Wasserstein space. In this setting, the principal components (which are vectors for Euclidean PCA), are replaced by geodesics, and can be represented as optimal maps from the Fréchet mean (barycenter) to a particular distribution. Wang et al. [Wan+13] first introduce a method to compute PCA for images cast as histograms, which does not rely on the PGA approach, but on an isometric linear embedding. Bigot et al. [Big+17] implement the PGA method for probability distributions. Their approach is limited to 1D distributions, because the logarithmic map has a closed-form in that case. Boissard et al. [Boi+15] parametrize the optimal maps (principal geodesics) by linear combinations of a family of maps. Seguy and Cuturi [SC15] use Sinkhorn distances, and a relaxed version of geodesics based on vector fields on the distribution's domain. Cazelles et al. [Caz+17] focuses on learning exact geodesic PCA (directly on the manifold, as opposed to the tangential approximation of PGA) for 1D distributions, and propose two efficient algorithms based on the proximal Forward-Backward scheme.

OT has also been used in dimensionality reduction based on matrix decomposition. Sandler and Lindenbaum [SL11] introduce the EMD-NMF, where they learn an NMF for which the comparison of reconstructions with input data (the fitting loss) is done with the earth mover's distance. They present applications in texture classification and face recognition. Ricci et al. [Ric+13] follow the same path, except that they additionally learn the ground metric of the OT loss before hand from the input data, and apply it for automatically extracting complex behaviors from crowded scenes. The work of Zen et al. [Zen+14] is very close to that of Ricci et al., the difference being that they *simultaneously* learn the ground metric and the matrix decomposition, using alternated optimization. Rolet et al. [Rol+16] improve the EMD-NMF by replacing the EMD with the Sinkhorn distance, which leads to faster and more robust computations. Flamary et al. [Fla+16] follow the PLCA approach [SR06] described earlier and replace the KL divergence by the Wasserstein distance, to evaluate reconstructions. They develop their framework for audio processing, and learn atoms that are representative of note spectra to perform music transcription.

Every method we mentionned in this paragraph is non-linear, because of the presence of OT as a loss, but they all rely on linear combinations between atoms and codes. The WDL method we introduce in chapter 3 differs from them because it replaces linear combinations by (non-linear) Wasserstein barycenters. As previously mentionned, Simou and Frossard [SF18] adapted WDL to graph signals, and Xu et al.

[Xu+18a] extended our work for NLP, by jointly learning topics (with Wasserstein barycenters) and word embeddings (ground distance).

2.3 Metric learning

In this section, we bring the context for our second contribution, which is the OT metric learning algorithm, presented in chapter 4. We first introduce the basic concepts of metric learning, then study the previous work on learning the ground metric of optimal transport.

2.3.1 Classical metric learning

In machine learning, metric learning is the task of inferring a metric on a domain using side information, such as examplar points that should be close or far away from each other. The assumption behind such methods is that metrics should be chosen within parameterized families and tailored for a task and data at hand, rather than selected among a few handpicked candidates. Metric learning algorithms are supervised, often learning from similarity and dissimilarity constraints between pairs of samples (x_i should be close to x_j), or triplets (x_i is closer to x_j than to x_k). Metric learning has applications in different tasks, such as classification, image retrieval, or clustering. For instance, for classification purposes, the learned metric brings closer samples of the same class and drives away samples of different classes [Xin+03].

Metric learning methods are either linear or non-linear, depending on the formulation of the metric with respect to its inputs. We will briefly recall various metric learning approaches, but refer the reader to recent surveys [Bel+15; Kul13] for further reading.

A widely-used linear metric function is the squared distance derived from the scalar product:

$$\mathrm{d}^2_{\mathbf{M}}(\boldsymbol{x}_1, \boldsymbol{x}_2) = (\boldsymbol{x}_1 - \boldsymbol{x}_2)^T \mathbf{M} (\boldsymbol{x}_1 - \boldsymbol{x}_2), \tag{2.28}$$

where $\boldsymbol{x}_1, \boldsymbol{x}_2 \in \mathbb{R}^{\mathrm{d}}$ and $\mathbf{M} \in \mathbb{R}^{\mathrm{d} \times \mathrm{d}}$. For d to be a distance in the mathematical sense, $\mathbf{M}$ must be a positive semi-definite (p.s.d) matrix. In this case $\mathbf{M}$ can be decomposed as $\mathbf{M} = \mathbf{L}^T \mathbf{L}$, and (2.28) becomes

$$\mathrm{d}^2_{\mathbf{M}}(\boldsymbol{x}_1, \boldsymbol{x}_2) = ||\mathbf{L}x_1 - \mathbf{L}x_2||_2^2 . \tag{2.29}$$

Thus, searching for such a metric is equivalent to searching for a global linear transformation of the data. This formulation is very close to the Mahalanobis distance, where the matrix $\mathbf{M}$ is chosen as the inverse covariance matrix of the data. For that reason, metrics of the form (2.28) are often referred to as a "Mahalanobis distances" in the metric learning literature. They are employed for example in the popular Large Margin Nearest Neighbors algorithm (LMNN) [Wei+06] along with a k-NN approach, to improve classification. Other linear methods [Che+09; PB16] choose not to satisfy all distance axioms (unlike the Mahalanobis distance) for more flexibility and because they are not essential for the distance to agree with human perception of similarities [Bel+15]. Non-linear methods include the prior embedding of the data (kernel trick) before performing a linear method [TL07; Wan+11], or other non-linear metric functions [Cho+05; Ked+12]. Facing problems where the data samples are histograms, researchers have developed metric learning methods based on distances that are better suited for histograms such as χ^2 [Ked+12; Yan+15] or the Wasserstein distance, which we describe in more detail.

2.3.2 Ground metric learning

The Wasserstein distance is employed in machine learning to compare histograms while accounting for the underlying geometry of their domain. This geometry is defined by the ground metric, which is usually carefully selected depending on the problem at hand, so that the Wasserstein distance is meaningful. It is often taken as the Euclidean or squared Euclidean distance. Therefore, being able to circumvent the selection of the ground metric by learning it is attractive from an applied perspective. Moreover, in computer graphics and machine learning, the ground space is often low-dimensional, e.g. age for population pyramids, $\mathbb{R}^2$ for bivariate distributions, or RGB space for color histograms. Therefore, when dealing with such distributions, instead of using a metric learning method in the high-dimensional distribution space (as classically done in methods described in the previous paragraph), we can perform it at lower cost in the low-dimensional ground space and use optimal transport to leverage the learned ground metric into a new metric in the distribution space.

In this thesis, "ground metric learning" (GML) refers to the general problem of learning the ground metric of optimal transport, rather than to the particular work of Cuturi and Avis [CA14], who introduced that terminology. Their method consists in learning a ground cost that is a true metric (definite, symmetric and satisfying triangle inequalities) using supervised information from a set of histograms. This method requires projecting matrices onto the cone of metric matrices, which is

known to require a cubic effort in the size of these matrices [Bri+08]. GML algorithms generally differ in the way they parameterize the metric, the input data they use to learn it, or their domain of application. Wang and Guibas [WG12] follow the approach of Cuturi and Avis [CA14] but drop the requirement that the learned cost must be a metric, and add support for triplet constraints. Zen et al. [Zen+14] use the approach of Cuturi and Avis [CA14] to enhance previous results on Non-negative Matrix Decomposition with a Wasserstein loss (EMD-NMF) [SL11], by alternatively learning the matrix decomposition and the ground metric. Dupuy et al. [Dup+16] learn a similarity matrix from the observation of a fixed transport plan, and use it to propose factors explaining weddings across groups in populations. Both use the entropic regularization of Wasserstein distances.

More recently, Xu et al. [Xu+18b] combined several previous ideas to create a new metric learning algorithm. It is a regularized Wasserstein distance-flavored LMNN scheme, with a Mahalanobis distance as ground metric, but with multiple local metrics [WS08] and a global one. Flamary et al. [Fla+18] adapt the classical Linear Discriminant Analysis [Fis36] to the Wasserstein space, i.e. they learn a linear embedding of the data in a lower-dimensional space, in which they optimize regularized Wasserstein distances to bring closer point sets of the same class, and push farther point sets of different classes. Su and Wu [SW19] learn a ground metric that is a Mahalanobis distance between vectors, in order to define a meta-distance between sequences of vectors, by alternatively updating the ground metric, and the sequence alignments.

Another line of research on GML is once again from the NLP community. Huang et al. [Hua+16] improve document classification with the *Word Mover's Distance* (introduced by Kusner et al. [Kus+15]), by learning a linear transformation of the *word2vec* word embedding [Mik+13] as the ground metric, and a weight vector which reflects the importance of words for distinguishing classes. As mentionned previously, Xu et al. [Xu+18a] perform the joint learning of topics and word embeddings in a Wasserstein fashion, leading to an approach that combines dictionary (topic) learning through Wasserstein barycenters, with ground metric (word embedding) learning. It is similar to the work of Zen et al. [Zen+14], in the sense that they both achieve an OT-driven dimensionality reduction by updating the ground metric in an alternated way. However, Zen et al. perform linear combinations and use the Wasserstein loss to compare reconstructions with training data, while Xu et al. use the non-linear combinations presented in WDL (chapter 3) with an L^2 loss. Finally, Xu et al. [Xu+19] perform graph matching through the joint learning of the transport between graphs and the node embeddings. They use the Gromov-Wasserstein

distance which allows to compare distributions on different spaces when we have access to intra-space distances, but not to inter-space ones.

Similarly to the above works, the framework we propose in chapter 4 is a GML method, but it differs in the formulation of the ground metric. We search for metrics that are geodesic distances on graphs, via a diffusion equation.

Our method also differs in the data we learn from since the observations that are fed to our algorithm are snapshots of a density evolution, and not pair or triplet constraints. We use displacement interpolations to reconstruct that movement, thus our objective function contains multiple inverse problems involving OT distances, as noted in section 2.1.7. This contrasts with simpler formulations where the objective function or the constraints directly depend on OT distances [CA14; WG12; Hua+16; Xu+18b]. Furthermore, the goal in these previous works is to perform supervised classification, a goal we do not seek directly in our approach. Nevertheless, our learning algorithm is supervised, since we provide the exact timestamps of each sample in the sequence.

The difference between our method and that of Dupuy et al. [Dup+16] lies in the available observations: they learn from a fixed matching (transport plan) whereas we learn from a sequence of mass displacements from which we infer both the metric and an optimal transport plan. Our method thus does not require identifying information on traveling masses. The fact that this identification is not required ranks among the most important and beneficial contributions of the OT geometry to data sciences, notably biology [Sch+19]. Finally, our method differs in the formulation of the ground metric. We use a non-linear approach based on the diffusion equation, whereas Cuturi and Avis [CA14] and Zen et al. [Zen+14] learn a full symmetrical matrix that can be constrained to satisfy triangle inequalities. Dupuy et al. [Dup+16] use a bilinear form parameterized by an affinity matrix, and [WS08; Hua+16; Xu+18b] employ Mahalanobis distances.

Our approach to metric learning corresponds to setting up an optimization problem over the space of geodesic distances on graphs, which is closely related to the continuous problem of optimizing Riemannian metrics. Optimizing metrics from functionals involving geodesic distances has been considered in [Ben+10]. This has recently been improved in [MD17] using automatic differentiation to compute the gradient of the functional involved, which is also the approach we take in our work. This type of metric optimization problem has also been studied within the OT framework (see for instance [But+04]), but these works are only concerned with convex problems (typically maximization of geodesic or OT distances), while our metric learning problem is highly non-convex.

2.4 Conclusion

Entropy regularization unveiled fast algorithms to approximate optimal transport distances, which unlocked their use in various fields that usually deal with large-scale data, such as computer graphics. The smoothness of this approximation also made the regularized Wasserstein distance popular in machine learning, as a differentiable loss function between densities that accounts for the geometry of the underlying space. Yet, even with this regularization, inverse problems dealing with quantities built upon Wasserstein distances (such as barycenters) are difficult to solve, because they are non-linear, in general non-convex, and computationally expensive.

In this chapter, we presented the context of this thesis. We introduced the computational optimal transport literature with a focus on entropy-regularized problems, and their applications to machine learning and computer graphics. We then introduced the context and related works of the two projects presented in the next chapters: dictionary learning with OT, and metric learning with OT.

3 Dictionary Learning

In this chapter we present our first contribution, which is the development of a dictionary learning algorithm based on the geometry of optimal transport. As described in 2.2, dictionary learning is a type of representation learning method as it consists in finding a new space for data points, in which a particular task (classification, compression, etc.) can be achieved more easily. Dictionary learning is an inverse problem in the sense that it aims to find the parameters of a model (the atoms and weights) that best reconstruct a given set of data points. The model is non-linear since the reconstructions are carried out with Wasserstein barycenters.

This project is a joint work with Morgan Schmitz, a PhD student from CEA[1] at Paris-Saclay. The results of this collaboration have been published in the SIAM Journal on Imaging Sciences [Sch+18], and partially in the proceedings of the SPIE conference "Wavelets and Sparsity XVII" [Sch+17]. We also released the source code [2]. For an introduction to classical dictionary learning, we refer the reader to 2.2.

3.1 Overview

The framework we proposed is a non-linear unsupervised dictionary learning method in the Wasserstein space. Given a set of discrete measures as input, our goal is to find a small number of atoms that are representative of them. In other words, we want to reconstruct each input as faithfully as possible (according to some loss function), by combining atoms together. The non-linearity of our method stems from the fact that we combine atoms using Wasserstein barycenters (2.1.4), instead of linear combinations as usually done in linear dictionary learning (2.2.1). This particularity allows our algorithm to account for the geometry of the domain, for example, by detecting similarities in shifted patterns, as seen in Figure 3.1. In order to learn the dictionary and the weights (codes) to recover each input, we minimize a non-convex functional with a quasi-Newton solver, and compute gradients with respect to the dictionary and weights using algorithmic (or automatic) differentiation [GW08].

[1] Comissariat à l'énergie atomique et aux énergies alternatives
[2] https://github.com/matthieuheitz/WassersteinDictionaryLearning

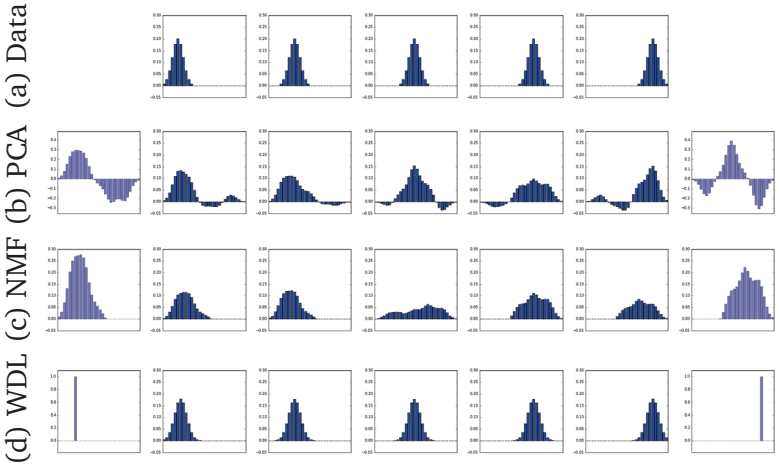


Figure 3.1: This figure illustrates the motivation for a Wasserstein dictionary learning method. Given 5 data points (first row), we find 2 atoms (next rows, first and last column) with different dictionary learning methods: PCA, NMF and our method, WDL. WDL is the only method that yields atoms that are probability distributions, and recovers the data points accurately. PCA fails to do so as it allows for negative values, and NMF, which is based on a positivity constraint, fails to approximate data points as it does not allow for lateral movements on the domain. The introduction of an entropic regularization is responsible for the blurring effect in the reconstructions, however the learned atoms become sharper than data points to counteract that effect (see the Diracs in the last row).

We developed several extensions for our framework, such as a stabilization of the Sinkhorn algorithm in log domain to obtain sharper results, a separable convolution kernel in log domain, two schemes to accelerate Sinkhorn iterations (the warm start strategy and the heavy ball method), and a generalization to the unbalanced setting.

We applied our method to the modeling of point spread functions (PSF) under different wavelengths, the detection of key moments in a sequence of cardiac images, and the clustering of facial expressions (Wasserstein faces) as well as of literary works.

Since this project is a shared work, I will only present in detail the parts of it that I was involved in. Morgan Schmitz worked on the calculation of the gradients, the development of the application for PSFs, the heavy ball acceleration scheme, and the extension to the unbalanced setting. For my part, I worked on the development of the cardiac imaging, the Wasserstein faces and literary work applications, on the stabilization of the algorithm in log-domain, and the warm-start acceleration strategy. My work focused on applications to large-scale histograms and the development of efficient and scalable algorithms for them.

3.2 Method

Let $\mathbf{X} \in \mathbb{R}^{N \times M}$ be the input data, that is, M histograms $\boldsymbol{x}_i \in \Sigma_N$. Let us denote S the number of atoms desired. In this chapter, we will only consider under-complete dictionaries ($S < N$), as the applications we develop are oriented towards dimensionality reduction rather than sparse approximations (see 2.2.1). Yet, our framework is general and would be suited to seek over-complete dictionaries, provided that a sparsity constraint is added for the weights.

We propose learning a dictionary $\mathbf{D} \overset{\text{def.}}{=} (\boldsymbol{d}_1, \dots, \boldsymbol{d}_S) \in (\Sigma_N)^S$ containing S atoms that are histograms, and a set of barycentric weights $\boldsymbol{\Lambda} \overset{\text{def.}}{=} (\boldsymbol{\lambda}_1, \dots, \boldsymbol{\lambda}_M) \in (\Sigma_S)^M$ such that each barycenter $\boldsymbol{p}(\mathbf{D}, \boldsymbol{\lambda}_i)$ approximates input $\boldsymbol{x}_i$ as faithfully as possible, according to a loss function $\mathcal{L}$ (see Table 3.1). That is, the functional to minimize is

$$\min_{\mathbf{D}, \boldsymbol{\Lambda}} \mathcal{E}(\mathbf{D}, \boldsymbol{\Lambda}) \overset{\text{def.}}{=} \sum_{i=1}^{M} \mathcal{L}\left(\boldsymbol{p}_\varepsilon(\mathbf{D}, \boldsymbol{\lambda}_i), \boldsymbol{x}_i\right). \tag{3.1}$$

As mentioned in the previous chapter (2.1.4), we use entropy-regularized Wasserstein barycenters because they are smooth and convex quantities. Throughout this chapter, we use the squared Euclidean distance for the ground metric $\mathbf{C}$, which is why we omit it in the barycenter notation (see 2.1.4).

Differentiating this energy requires the computation of the Jacobians of the barycenter with respect to the dictionary $\mathbf{D}$ and the weights $\boldsymbol{\Lambda}$. As mentionned in 2.1.7, our method follows the work of Bonneel et al. [Bon+16], which performs the regression of Wasserstein barycentric coordinates, using regularized barycenters. They showed that computing the Jacobian of the Wasserstein barycenter with respect to the weights requires solving a linear system of size N^2, which is intractable when dealing with large-scale histograms and also suffers from stability issues. Instead, they minimize a functional that is based on the approximate regularized barycenter $\boldsymbol{p}_\varepsilon^{(L)}$ computed after a fixed number L of iterations (see 2.1.4), and calculate an algorithmic differentiation of the Jacobian. In other words, they compute the exact gradient of the barycenter's approximation rather than an approximation of the exact barycenter's gradient. We directly follow that approach, therefore the objective function we minimize is

$$\min_{\mathbf{D}, \boldsymbol{\Lambda}} \mathcal{E}_L(\mathbf{D}, \boldsymbol{\Lambda}) \overset{\text{def.}}{=} \sum_{i=1}^{M} \mathcal{L}\left(\boldsymbol{p}_\varepsilon^{(L)}(\mathbf{D}, \boldsymbol{\lambda}_i), \boldsymbol{x}_i\right). \tag{3.2}$$

Name	$\mathcal{L}(\boldsymbol{a}, \boldsymbol{b})$	$\nabla\mathcal{L}$
Total Variation $\mathcal{L}_1$	$\|\boldsymbol{a}-\boldsymbol{b}\|_1$	$\operatorname{sign}(\boldsymbol{a}-\boldsymbol{b})$
Quadratic $\mathcal{L}_2$	$\|\boldsymbol{a}-\boldsymbol{b}\|_2^2$	$2(\boldsymbol{a}-\boldsymbol{b})$
Kullback-Leibler $\mathcal{L}_{KL}$	$\mathrm{KL}(\boldsymbol{a}\|\boldsymbol{b})$	$\log(\boldsymbol{a}/\boldsymbol{b})-1$
Wasserstein $\mathcal{L}_W$	$W_{\mathbf{C}}^{\varepsilon\,(L)}(\boldsymbol{a}, \boldsymbol{b})$	$\varepsilon \log(\boldsymbol{u}^{(L)})$

Table 3.1: Examples of loss functions and their gradient with respect to $\boldsymbol{a}$. For the Wasserstein case, the loss is computed iteratively with the Sinkhorn algorithm (1), and $\boldsymbol{u}^{(L)}$ is the first scaling vector, obtained after L iterations. Computing the gradient of the Wasserstein loss in this manner has been done in [CD14; Bon+16], and relies on the envelope theorem [MS02]. The number of iterations L for this loss must be set to a large enough value that ensures sufficient convergence of the Sinkhorn algorithm. Some authors [Gen+17] rather advise to pursue automatic differentiation through the loss.

This problem is in general not convex, neither jointly in $\mathbf{D}$ and $\mathbf{\Lambda}$, nor independently in either. Consequently, we seek a local minimum by computing the gradients and following a descent optimization scheme. The problem in this form does not display the additional constraints on the atoms and the weights, that are to lie in their respective probability simplices Σ_N, Σ_S. We enforce these conditions through a change of variable in order to carry out unconstrained optimization.

Because $\mathcal{E}_L$ is a sum over all input datapoints, and for the sake of simplicity, we focus on the derivation of the energy for a single datapoint $x \in \Sigma_N$, which reduces the codes matrix $\mathbf{\Lambda}$ to a single weight vector $\boldsymbol{\lambda}_i \in \Sigma_S$. Differentiating (3.2) gives

$$\begin{aligned} \nabla_{\mathbf{D}} \mathcal{E}_L(\mathbf{D}, \mathbf{\Lambda}) &= \left[\partial_{\mathbf{D}} \boldsymbol{p}_\varepsilon^{(L)}(\mathbf{D}, \boldsymbol{\lambda}_i)\right]^T \nabla\mathcal{L}(\boldsymbol{p}_\varepsilon^{(L)}(\mathbf{D}, \boldsymbol{\lambda}_i), x) \\ \nabla_{\boldsymbol{\lambda}_i} \mathcal{E}_L(\mathbf{D}, \mathbf{\Lambda}) &= \left[\partial_{\boldsymbol{\lambda}_i} \boldsymbol{p}_\varepsilon^{(L)}(\mathbf{D}, \boldsymbol{\lambda}_i)\right]^T \nabla\mathcal{L}(\boldsymbol{p}_\varepsilon^{(L)}(\mathbf{D}, \boldsymbol{\lambda}_i), x). \end{aligned} \tag{3.3}$$

The gradient of the loss function $\nabla\mathcal{L}$ is generally a closed-form formula that is simple to compute. See Table 3.1 for examples of the loss functions we used in our applications, and their gradient. The other terms are the Jacobians of the approximate Wasserstein barycenter with respect to the dictionary and the weights, which we compute through algorithmic differentiation, as performed by [Bon+16]. The algorithmic differentiation for these Jacobians consists in the recursive backward differentiation of the Sinkhorn barycenter's iterations (2.25). This can be calculated manually, or by employing an automatic differentiation software (such as Theano [Tea+16] or PyTorch [Pas+17]).

We present in Algorithm 3 the "manual" calculations of both Jacobians in (3.3) which are for a single reconstruction and its gradient, as mentioned earlier. The proofs for these formulas can be found in the publication and its supplemental material

[Sch+18]. The "forward" loop is simply the Sinkhorn barycenter algorithm that reconstructs an input $\boldsymbol{x}$ from the dictionary. Evaluating the gradients is then achieved through two "backward" loops (one for the dictionary, one for the weights), which require some intermediate results from the forward loop. These two intermediate variables $\boldsymbol{y}_s^{(l)}$ and $\boldsymbol{q}_s^{(l)}$ are of size SNL and need to be fully stored between the loops. Note that both backward loops have descending iterators, which stems from the fact that in backward differentiation, the gradient is computed from the last variable to the first.

Identically to the Sinkhorn and Sinkhorn barycenter algorithms, these schemes only rely on element-wise operations and on the application of the kernel matrix $\mathbf{K}$ and its transpose. The ground metric $\mathbf{C}$ being Euclidean, and the domain being a square grid, the multiplication by $\mathbf{K}$ (which is intractable for large N) can be performed with a separable Gaussian convolution, as in [Sol+15]. For this reason, multiplications with $\mathbf{K}$ are denoted as a convolution operator K in Algorithm 3. Applying the convolution kernel being the bottleneck operation, we can see that both backward loops have the same complexity as the forward loop, since they both require two kernel applications.

For our applications (see 3.4), we chose to enforce the positivity and unit mass constraints on $\boldsymbol{d}_s$ and $\boldsymbol{\lambda}_i$ through the following change of variable:

$$\forall s, \boldsymbol{d}_s \overset{\text{def.}}{=} F_N(\boldsymbol{\alpha}_s) \overset{\text{def.}}{=} \frac{\exp(\boldsymbol{\alpha}_s)}{\sum_{j=1}^{N} \exp([\boldsymbol{\alpha}_s]_j)} \qquad \boldsymbol{\lambda}_i \overset{\text{def.}}{=} F_S(\boldsymbol{\beta}_i) \overset{\text{def.}}{=} \frac{\exp(\boldsymbol{\beta}_i)}{\sum_{j=1}^{S} \exp([\boldsymbol{\beta}_i]_j)}. \tag{3.4}$$

Therefore, we optimize the new variables $\mathbf{A} \overset{\text{def.}}{=} (\boldsymbol{\alpha}_s)_{s=1}^S$ and $\boldsymbol{\beta}_i$ in place of $\mathbf{D}$ and $\boldsymbol{\lambda}_i$. The energy to minimize is then

$$\mathcal{G}_L(\mathbf{A}, \boldsymbol{\beta}_i) \overset{\text{def.}}{=} \mathcal{E}_L(F(\mathbf{A}), F_S(\boldsymbol{\beta}_i)), \tag{3.5}$$

where $F(\mathbf{A}) \overset{\text{def.}}{=} (F_N(\boldsymbol{\alpha}_1), \dots, F_N(\boldsymbol{\alpha}_S)) = \mathbf{D}$.

Differentiating (3.5) yields

$$\begin{aligned} \nabla_{\mathbf{A}} \mathcal{G}_L(\mathbf{A}, \boldsymbol{\beta}_i) &= [\partial F(\mathbf{A})]^T \nabla_{\mathbf{D}} \mathcal{E}_L\left(F(\mathbf{A}), F_S(\boldsymbol{\beta}_i)\right) = [\partial F(\mathbf{A})]^T \nabla_{\mathbf{D}} \mathcal{E}_L\left(\mathbf{D}, \boldsymbol{\Lambda}\right) \\ \nabla_{\boldsymbol{\beta}_i} \mathcal{G}_L(\mathbf{A}, \boldsymbol{\beta}_i) &= [\partial F_S(\boldsymbol{\beta}_i)]^T \nabla_{\boldsymbol{\lambda}_i} \mathcal{E}_L\left(F(\mathbf{A}), F_S(\boldsymbol{\beta}_i)\right) = [\partial F_S(\boldsymbol{\beta}_i)]^T \nabla_{\boldsymbol{\lambda}_i} \mathcal{E}_L\left(\mathbf{D}, \boldsymbol{\Lambda}\right), \end{aligned} \tag{3.6}$$

where $[\partial F_P(\boldsymbol{u})]^T = \partial F_P(\boldsymbol{u}) = \left(\mathbf{Id}_P - F_P(\boldsymbol{u}) \mathbb{1}_P^T\right) \operatorname{diag}(F_P(\boldsymbol{u}))$, P being the size of the vector $\boldsymbol{u}$ in question.

Algorithm 3 `SinkhornGrads`: Computation of gradients w.r.t the dictionary and the barycentric weights, for a single input histogram x_i.

Inputs: Atoms $\boldsymbol{d}_1, \dots, \boldsymbol{d}_S \in \Sigma_N$, histogram $\boldsymbol{x}_i \in \Sigma_N$, weight vector $\boldsymbol{\lambda}_i \in \Sigma_S$

$\forall s, \boldsymbol{v}_s^{(0)} = \mathbf{1}_N$

for $l = 1 \dots L$ **do** ▷ Forward loop - Sinkhorn barycenter

$\forall s, \boldsymbol{y}_s^{(l-1)} = K \boldsymbol{v}_s^{(l-1)}$

$\forall s, \boldsymbol{q}_s^{(l)} = K^T \left(\frac{\boldsymbol{d}_s}{\boldsymbol{y}_s^{(l-1)}} \right)$

$\boldsymbol{p}^{(l)} = \prod_s \left(\boldsymbol{q}_s^{(l)} \right)^{[\boldsymbol{\lambda}_i]_s}$

$\forall s, \boldsymbol{v}_s^{(l)} = \frac{\boldsymbol{p}^{(l)}}{\boldsymbol{q}_s^{(l)}}$

end for

$\boldsymbol{w} = \mathbf{0}_S$

$\forall s, \boldsymbol{r}_s = \mathbf{0}_N$

$\boldsymbol{g} = \nabla \mathcal{L}(\boldsymbol{p}^{(L)}, \boldsymbol{x}_i) \odot \boldsymbol{p}^{(L)}$

for $l = L \dots 1$ **do** ▷ Backward loop - weights

$\forall s, w_s = w_s + < \log \boldsymbol{q}_s^{(l)}, \boldsymbol{g} >$

$\forall s, \boldsymbol{r}_s = -K^T \left(K \left(\frac{[\boldsymbol{\lambda}_i]_s \boldsymbol{g} - \boldsymbol{r}_s}{\boldsymbol{q}_s^{(l)}} \right) \odot \frac{\boldsymbol{d}_s}{(\boldsymbol{y}_s^{(l-1)})^2} \right) \odot \boldsymbol{v}_s^{(l-1)}$

$\boldsymbol{g} = \sum_s \boldsymbol{r}_s$

end for

$\forall s, \boldsymbol{y}_s = \mathbf{0}_N$

$\forall s, \boldsymbol{z}_s = \mathbf{0}_N$

$\boldsymbol{n} = \nabla \mathcal{L}(\boldsymbol{p}^{(L)}, \boldsymbol{x}_i)$

for $l = L \dots 1$ **do** ▷ Backward loop - dictionary

$\forall s, \boldsymbol{c}_s = K((\left[\boldsymbol{\lambda}_i\right]_s \boldsymbol{n} - \boldsymbol{z}_s) \odot \boldsymbol{v}_s^{(l)})$

$\forall s, \boldsymbol{y}_s = \boldsymbol{y}_s + \frac{\boldsymbol{c}_s}{\boldsymbol{y}_s^{(l-1)}}$

$\forall s, \boldsymbol{z}_s = -\frac{\mathbb{1}_N}{\boldsymbol{q}_s^{(l-1)}} \odot K^T \frac{\boldsymbol{d}_s \odot \boldsymbol{c}_s}{(\boldsymbol{y}_s^{(l-1)})^2}$

$\boldsymbol{n} = \sum_s \boldsymbol{z}_s$

end for

Outputs: $\boldsymbol{p}_\varepsilon^{(L)}(\mathbf{D}, \boldsymbol{\lambda}_i) \overset{\text{def.}}{=} \boldsymbol{p}^{(L)}, \nabla_{\mathbf{D}} \mathcal{E}^{(L)} \overset{\text{def.}}{=} [\boldsymbol{y}_1, \dots, \boldsymbol{y}_S], \nabla_{\boldsymbol{\lambda}_i} \mathcal{E}^{(L)} \overset{\text{def.}}{=} \boldsymbol{w}$

Keep in mind that we differentiate the energy (3.2) for a single datapoint, and that in order to differentiate it for M of them, one only needs to sum the energy over the list of M weights vectors $\mathbf{B} \overset{\text{def.}}{=} (\beta_i)_{i=1}^M$:

$$\mathcal{G}_L(\mathbf{A}, \mathbf{B}) \overset{\text{def.}}{=} \sum_i \mathcal{G}_L(\mathbf{A}, \beta_i). \tag{3.7}$$

The change of variable introduced in (3.4) allows us to use an unconstrained optimization scheme, which makes our framework general enough to use any suitable optimizer. For our applications, we chose the quasi-Newton solver L-BFGS in order to find a local minimum of the non-convex energy function (3.5). An overall algorithm describing the integration of our functional with that solver is given in the publication [Sch+18, Algorithm 2].

Instead of conducting an alternated optimization scheme like most dictionary learning methods, we *simultaneously* optimize the transformed dictionary and weights $\mathbf{A}$ and $\mathbf{B}$ by concatenating them into a single vector. That vector of size $S \times N + M \times S$ is then given to the solver as the variable to optimize. This construction is advantageous in our setting, because the intermediate quantities that are computed during the forward pass (loop) are necessary to compute both gradients. As a result, we evaluate the objective function only once in order to update both variables. However, this structure means that the dictionary and the weights are updated with the same step size, which often leads to asymmetries in convergence, e.g. the algorithm will stop in a local minimum with respect to the dictionary, but not to the weights. To alleviate this issue, we introduce a hyperparameter ζ, which scales the gradient with respect to the weights, and which should be adjusted to make sure that the dictionary and the weights are being optimized to a comparable extent.

3.3 Extensions

3.3.1 Log-domain stabilization

For some applications (e.g. compression), one may want to recover original data points from their representation very precisely. In our framework, reconstructing the input data is impaired by the entropic regularization, since it introduces a blur in the transport plan and the barycenters. This blur is controlled by the parameter ε: lower values mean lower regularization, which yields sharper barycenters. However, this parameter cannot be chosen too small because of numerical limitations. Some entries in the kernel $\mathbf{K} = \exp(-\mathbf{C}/\varepsilon)$ might reach representation limits, and as

seen in Figure 2.2, entries in the scaling vectors diverge as Sinkhorn iterations accumulate (which leads to numerical errors). Therefore, the Sinkhorn algorithm (1) and its barycenter version (2) cannot reconstruct input histograms to an arbitrary precision. In order to avoid this issue, we follow the stabilization scheme introduced by [Chi+16; Sch16] which conducts the Sinkhorn iterations in the log domain. The iterative scheme then relies on the *dual potentials* (or *dual scalings*) $\boldsymbol{f}$ and $\boldsymbol{g}$ (see (2.8) and (2.9)), defined as

$$\boldsymbol{f}^{(l)} \stackrel{\text{def.}}{=} \varepsilon \log\left(\boldsymbol{u}^{(l)}\right) \qquad \boldsymbol{g}^{(l)} \stackrel{\text{def.}}{=} \varepsilon \log\left(\boldsymbol{v}^{(l)}\right). \tag{3.8}$$

Since we are more interested in computing OT barycenters rather than distances, we present the log-domain scheme for the Sinkhorn *barycenter* iterations, requiring one pair of scalings for each input histogram. Taking the log of the scaling updates in (2.25), one has

$$\begin{aligned} \boldsymbol{f}_s^{(l)} &= \varepsilon\left[\log\left(\boldsymbol{a}_s\right) - \log\left(\mathbf{K}\exp\left(\boldsymbol{g}_s^{(l-1)}/\varepsilon\right)\right)\right] \\ \boldsymbol{g}_s^{(l)} &= \varepsilon\left[\log\left(\boldsymbol{b}^{(l)}\right) - \log\left(\mathbf{K}^T\exp\left(\boldsymbol{f}_s^{(l)}/\varepsilon\right)\right)\right]. \end{aligned} \tag{3.9}$$

In order to only deal with stable quantities, we need to compute the terms $\log\left(\mathbf{K}\exp\left(\boldsymbol{g}_s^{(l-1)}/\varepsilon\right)\right)$ and $\log\left(\mathbf{K}\exp\left(\boldsymbol{f}_s^{(l)}/\varepsilon\right)\right)$ without explicitly taking the exponential of $\boldsymbol{g}_s^{(l-1)}/\varepsilon$ or $\boldsymbol{f}_s^{(l)}/\varepsilon$ because it would cause overflows. We detail hereafter two different ways to do so.

Stabilized kernel The first way is to introduce a stabilized kernel $\tilde{\mathbf{K}}(\boldsymbol{f},\boldsymbol{g})$ defined as:

$$\tilde{\mathbf{K}}(\boldsymbol{f},\boldsymbol{g}) = \exp\left(\frac{-\mathbf{C}+\boldsymbol{f}\mathbb{1}^T+\mathbb{1}\boldsymbol{g}^T}{\varepsilon}\right). \tag{3.10}$$

Indeed, the quantity $\mathbf{C}_{i,j}-\boldsymbol{f}_i-\boldsymbol{g}_j$ stays bounded, so the extreme values of $\boldsymbol{f}$ and $\boldsymbol{g}$ cancel each other out. Note that this stabilized kernel is equal to the transport plan $\mathbf{P}$ (see (2.5)), which means that $\tilde{\mathbf{K}}(\boldsymbol{f}^{(l)},\boldsymbol{g}^{(l)})$ is the approximation of the transport plan after l iterations $\mathbf{P}^{(l)}$.

We obtain the following scheme and refer the reader to the publication [Sch+18, §4.1.1] for details:

$$\begin{aligned} \boldsymbol{f}_s^{(l)} &= \varepsilon\left[\log\left(\boldsymbol{a}_s\right) - \log\left(\tilde{\mathbf{K}}(\boldsymbol{f}_s^{(l-1)},\boldsymbol{g}_s^{(l-1)})\mathbb{1}\right)\right] + \boldsymbol{f}_s^{(l-1)} \\ \boldsymbol{g}_s^{(l)} &= \varepsilon\left[\log\left(\boldsymbol{b}^{(l)}\right) - \log\left(\tilde{\mathbf{K}}(\boldsymbol{f}_s^{(l)},\boldsymbol{g}_s^{(l-1)})^T\mathbb{1}\right)\right] + \boldsymbol{g}_s^{(l-1)}. \end{aligned} \tag{3.11}$$

However, such a scheme requires the computation of two stabilized kernels per Sinkhorn iteration, which is very costly since they are of size N^2, and not separable because of the added terms in the exponential. Moreover, each of those kernels would need to be stored or recomputed for the backward pass. Thus, this scheme is not appropriate for our large scale setting.

Another variant of this scheme, called *absorption iterations* [Chi+16; Sch16] is a midway computation where the scaling updates are carried out in the usual domain, but the extreme values are absorbed in the stabilized kernel (3.10) from time to time. This reduces the number of kernels to compute and store, however it does not alleviate the quadratic size of the kernels, which is in general the memory and computational bottleneck.

Separable log kernel The second way to compute the terms mentioned above is more compatible with our large-scale setting. It consists in applying the kernel $\mathbf{K}$ through separable convolutions, as presented in 2.1.2, but in the log domain with a stabilization technique. In the log domain, an operator called the *log-sum-exp* appears when applying the kernel:

$$[\log(\mathbf{K}\exp(\boldsymbol{g}/\varepsilon))]_i = \log\left(\sum_j \exp\left(\frac{-\mathbf{C}_{i,j}+\boldsymbol{g}_j}{\varepsilon}\right)\right) \stackrel{\text{def.}}{=} [K_{LSE}(\boldsymbol{g})]_i \tag{3.12}$$

$$[\log(\mathbf{K}^T\exp(\boldsymbol{f}/\varepsilon))]_i = \log\left(\sum_j \exp\left(\frac{-\mathbf{C}^T_{i,j}+\boldsymbol{f}_j}{\varepsilon}\right)\right) \stackrel{\text{def.}}{=} [K^T_{LSE}(\boldsymbol{f})]_i\,. \tag{3.13}$$

This operator acts as a soft maximum, and its computation can be stabilized by shifting the input values by their maximum and adding it after the operator, to avoid the exponentiation of large values:

$$\log\left(\sum_j \exp\left(\boldsymbol{a}_j\right)\right) = \log\left(\sum_j \exp\left(\boldsymbol{a}_j - \max_j \boldsymbol{a}_j\right)\right) + \max_j \boldsymbol{a}_j.\,. \tag{3.14}$$

In addition to a better time complexity, the separable convolution yields a better space complexity because it avoids computing any matrix of size N^2, which is crucial in our setting. An example of the process in 2 dimensions is detailed in Algorithm 4.

Algorithm 4 `LogSepKernel` K_{LSE}: Application of a 2-D separable kernel in log-domain

Input: Separable cost function $C = C_x + C_y$, 2-D function in log-domain $g \in \mathbb{R}^{n \times n}$
$\forall k, j,\ x_l(k,j) = -\frac{C_x(j,l)}{\varepsilon} + g(k,l)$
$\forall k, j,\ r_1(k,j) = \log\left(\sum_l^n \exp(x_l - \max_l x_l)\right) + \max_l x_l$
$\forall i, j,\ y_k(i,j) = -\frac{C_y(i,k)}{\varepsilon} + r_1(k,j)$
$\forall i, j,\ r_2(i,j) = \log\left(\sum_k^n \exp(y_k - \max_k y_k)\right) + \max_k y_k$
Output: Convoluted function in log-domain $K_{LSE}(g) = r_2$

The forward iterations then become

$$\begin{aligned} \boldsymbol{f}_s^{(l)} &= \varepsilon \left[\log\left(\boldsymbol{a}_s\right) - K_{LSE}(\boldsymbol{g}_s^{(l-1)})\right] \\ \boldsymbol{g}_s^{(l)} &= \varepsilon \left[\log\left(\boldsymbol{b}^{(l)}\right) - K_{LSE}^T(\boldsymbol{f}_s^{(l)})\right]. \end{aligned} \tag{3.15}$$

For the backward loops, intermediate values can be negative and real-valued logarithms are not suited. While complex valued logarithms solve this problem, they come at a prohibitive computational cost. Instead, we store the sign of the input values and compute logarithms of absolute values. When exponentiating, the stored sign is used to recover the correct values.

To sum up, this scheme allows to perform efficient optimization algorithms based on Wasserstein barycenters, which can deal with low levels of regularization. The price to pay is an overhead cost due to supplementary logarithms and exponentials to compute. It is important to note however, that for very small levels of regularization, discretization artifacts will appear in the barycenters, as observed by Schmitzer [Sch16].

3.3.2 Warm start

The warm start strategy, often used in optimization problems, consists in using the solution of a previous optimization problem that is close to the current one, as initialization point in order to speed up the convergence. Our method relies on performing an iterative optimization process which, at each iteration, calls upon *another* iterative scheme, namely the forward Sinkhorn loop that computes the barycenters. We will denote the optimizer iterations as "steps" to differentiate them from Sinkhorn iterations. As described in 2.1.4, the Sinkhorn algorithm is usually initialized with constant scaling vectors. However, in our case, since each step performs a new Sinkhorn loop, the last scaling vectors of the previous step can be used to initialize the scaling vectors of the current one, thus "warm-starting"

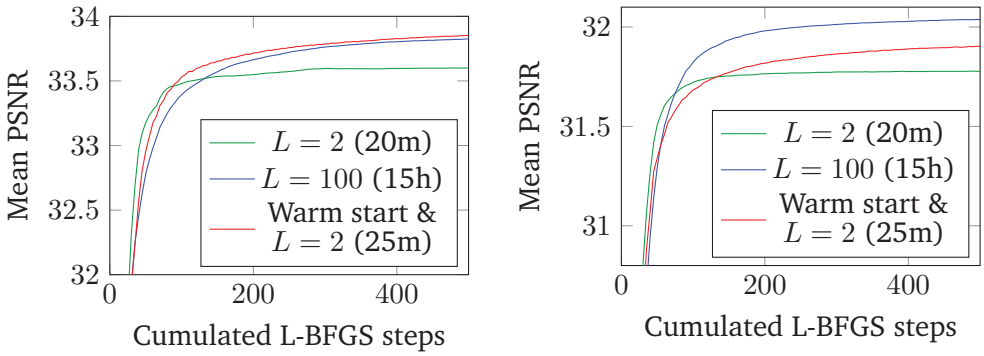
the barycenter computation. This technique therefore accumulates the Sinkhorn iterations as we accumulate optimization steps. This has several consequences: a gain in precision and time, a potential increase in the instability of the scaling vectors, and the energy we minimize changing over time.

First, the last scaling vectors of the previous step are closer to those of the current one than a vector of constant value. Therefore, the Sinkhorn algorithm converges more rapidly, and the final barycenters computed at each step gain accuracy compared to the classical version of the algorithm.

Second, as mentioned in 3.3.1, the scaling vectors may become unstable when computing a large number of Sinkhorn iterations. When using a warm start strategy, iterations accumulate, which may consequently degrade the stability of the scaling vectors. For example, using 20 Sinkhorn iterations running through 50 steps, a warm start would lead to barycenters computed with scaling vectors comparable to those obtained after 1000 Sinkhorn iterations. When instabilities become an issue, we couple the warm start approach with our log-domain stabilization. The reduced speed of log-domain computations is largely compensated by the fact that our warm start allows the computation of less Sinkhorn iterations for an equivalent or better result.

Third, when differentiating (3.2), we consider the initial, warm-started (as opposed to initializing $\boldsymbol{v}_s^{(0)}$ to $\mathbb{1}_N$) values given to the scaling vectors to be constant and independent of weights and atoms. This amounts to considering a different energy to minimize at each optimization step. While it is inconsequential for first order optimizations such as gradient descent, the quasi-Newton solver we use (L-BFGS) relies on previous iterations to estimate a Hessian, and is thus sensitive to such perturbations. Primarily, it can cause L-BFGS to exit prematurely since the computed gradients do not correspond to the energy that is minimized. To mitigate this issue, we simply restart L-BFGS every fixed number of steps, with the dictionary, the weights and the last scaling vectors of the previous run.

We demonstrate the benefits of the warm start in Figure 3.2. We plot the evolution of the mean PSNR (Peak Signal-to-Noise Ratio) of the reconstructions throughout the L-BFGS steps for different settings, for the two datasets used in 3.4.2. For these examples, we used the KL loss (since it gave the best reconstructions overall), we did not have to use the log-domain stabilization, and we restarted L-BFGS every 10 steps. At an equal number of Sinkhorn iterations L, enabling the warm start always yields better reconstructions after a certain number of steps. It comes at a small overhead cost in time (around 25%), because L-BFGS line search routine requires more evaluations at start. For the example in Figure 3.2a, the computation times



(a) MUG dataset: woman

(b) MUG dataset: man

Figure 3.2: Evolution of the mean PSNR of the reconstructions against cumulated L-BFGS steps, for 3 configurations, on two cases of the MUG datasets used in the Wasserstein faces application (see 3.4.2). The KL loss was used for this experiment. We see that with the same number of Sinkhorn iterations ($L = 2$), enabling warm start yields better reconstructions than disabling it, in roughly the same time (red and green curves).

are 20 minutes for $L = 2$, 25 minutes for warm restart and $L = 2$, and 15 hours for $L = 100$. In this particular case, enabling the warm start with 2 Sinkhorn iterations yields even better results than having 100 Sinkhorn iterations without warm start, and with a 36 gain factor in time. For the second dataset (Figure 3.2b), enabling the warm start does not yield as good results as when running 100 Sinkhorn iterations. However, it would require considerably more than 2 Sinkhorn iterations, hence a lot more time, to achieve the same result without it. The computation times in all three cases are similar to the previous example.

3.3.3 Sinkhorn heavy ball

An acceleration method often used for first order optimization methods, and sometimes referred to as the *heavy ball* method, consists in adding a momentum term to the parameter update:

$$\theta^{(l+1)} = \theta^{(l)} - \alpha \nabla f(\theta^{(l)}) + \tau \left(\theta^{(l)} - \theta^{(l-1)}\right), \tag{3.16}$$

with θ the optimized variable, f the energy function, α the step size, and τ the momentum coefficient. It dates back to [Pol64] and is closely related to Nesterov's Accelerelated Gradient [Nes83]. Peyré et al. [Pey+16] introduced a similar relaxation scheme in the Sinkhorn updates (for the particular case of tensor-valued OT), in order to make the fixed-point scheme contractant. We apply this acceleration

technique to our scalar-valued barycenter setting, which results in a supplementary averaging step in the iterations:

$$\begin{aligned}
\tilde{\boldsymbol{u}}_s^{(l)} &= \frac{\boldsymbol{a}_s}{\mathbf{K}\boldsymbol{v}_s^{(l-1)}} \\
\boldsymbol{u}_s^{(l)} &= (\boldsymbol{u}_s^{(l-1)})^{\tau} (\tilde{\boldsymbol{u}}_s^{(l)})^{1-\tau} \\
\tilde{\boldsymbol{v}}_s^{(l)} &= \frac{\boldsymbol{b}^{(l)}}{\mathbf{K}^T \boldsymbol{u}_s^{(l)}} \\
\boldsymbol{v}_s^{(l)} &= (\boldsymbol{v}_s^{(l-1)})^{\tau} (\tilde{\boldsymbol{v}}_s^{(l)})^{1-\tau}.
\end{aligned} \tag{3.17}$$

This formulation comes from the addition of the momentum term $\tau \left(\boldsymbol{f}_s^{(l-1)} - \tilde{\boldsymbol{f}}_s^{(l)} \right)$ in the log-domain, which explains the multiplicative updates in the normal domain. Note that this scheme is similar but different from the *heavy ball* method, which would yield the following momentum term: $\tau \left(\boldsymbol{f}_s^{(l-1)} - \boldsymbol{f}_s^{(l-2)} \right)$. While setting $\tau = 0$ gives the classical Sinkhorn barycenter iterations (2.25), choosing a negative value for τ can accelerate the convergence significantly. The modification of this scheme implies that the gradients with respect to the dictionary and the weights should be recalculated, or one can also resort to an automatic differentiation library. The corresponding algorithm can be found in the publication [Sch+18, Algorithm 5].

3.3.4 Unbalanced

Introduced by Benamou [Ben03], the idea of unbalanced optimal transport is to solve OT between distributions of unequal masses, by allowing the creation and destruction of mass. This implies the relaxation of mass conservation constraints on the transport plan, i.e. it is not necessary that the marginals of the transport plan reconstruct the input distributions exactly. In our publication [Sch+18, §4.4, §5.6], we detail the modified forward scheme for the unbalanced setting as proposed by Chizat et al. [Chi+16] and an example application to motivate its use. It is based on the replacement of equality constraints between the coupling marginals and input distributions with a minimization of the KL divergence between them. Regarding the differentiation of such a scheme, we employed an automatic differentiation library as for the *heavy ball* extension.

3.4 Experiments

In this section, we present the different applications we developed for our framework. In the publication [Sch+18, §5.1], we compared our method to the Wasserstein principal geodesics [SC15], since a displacement interpolation between the 2 atoms of a dictionary learned on a set of measures could potentially match the first principal geodesic. We found that the two methods can produce very similar results, but they do not correspond in all cases, for various reasons such as the non-convexity of our functional or the different approximations employed in each.

An application that was primarily developed by our colleagues at CEA is the one for point spread functions (PSF) [Sch+18, §5.2] and we will not discuss it in detail. In summary, we managed to precisely model the variations of the Euclid telescope's PSF under different wavelengths of incoming light. We learned a dictionary of 2 atoms, which corresponded to the PSFs for the two extreme wavelengths of the considered spectrum, allowing to interpolate between them for any wavelength. This showed that OT interpolations model PSFs variations across wavelengths well.

For the first two experiments, we chose to cast digital images as probability distributions through normalization, i.e. divide each pixel value by the sum of all pixel values, in order to get a total sum (density) of 1. Normalizing images so that they sum up to 1 can have unintended effects, in particular when the densities are very different across images. For example, if an image contains an object A, and the next image contains A and another object B, then object A will not have the same mass in both images when they are normalized. This will lead to potentially undesirable mass transfers from B to A, to account for that mass imbalance. Using the unbalanced extension described above would help mitigate this problem. However, in the datasets we use, images have relatively small differences in total density (max 1% in the cardiac sequence, and max 5% in the Wasserstein faces), therefore the potential artifacts of normalization are negligible.

As mentioned in 2.2, dictionary learning is one way of conducting representation learning. We explain hereafter an example of the dimensionality reduction capabilities of our method. By assigning each atom a point in $\mathbb{R}^{\mathrm{d}}$, one can also assign to each input frame a point in that space, that is the Euclidean barycenter of the atom's points weighted by the coefficients learned by our algorithm. To ensure that each position in this simplex is defined by a unique set of coordinates, one needs to embed S atoms in $\mathbb{R}^{S-1}$, because one of the weights can be deduced from the others as they sum to 1. This *barycentric space* is a new representation of the input data

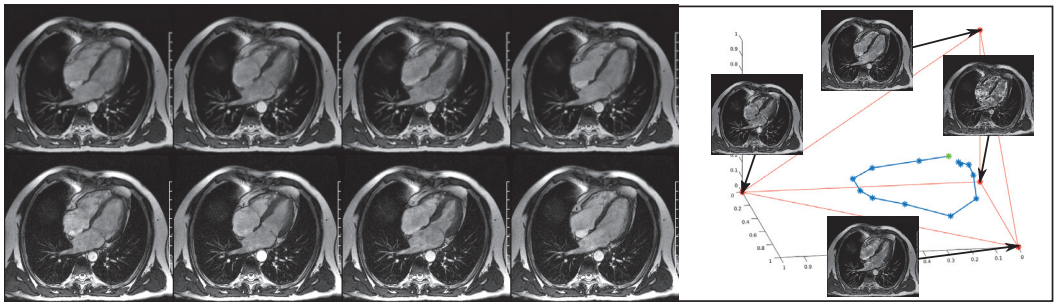


Figure 3.3: Left: Comparison between 4 frames (out of 13) of the measures (lower row) and the same reconstructed frames (upper row). Right: Plot of the reconstructed frames (blue points) by their barycentric coordinates in the 4-atoms basis, with each atom (red points) at the vertices of the tetrahedron. The green point is the first frame.

based on the Wasserstein distance, in which one can perform different tasks such as clustering or classification.

3.4.1 Cardiac sequences

An image sequence of a beating heart is *a priori* cyclic, and contains a significant amount of redundant information. Being able to learn an efficient representation of the sequence is interesting for compression purposes and anomaly detection. If we search for key moments in the sequence, we might be able to reconstruct it entirely by interpolating between those moments.

We tested our dictionary learning algorithm on a reconstructed MRI sequence of one complete heartbeat cycle. We chose to learn $S = 4$ atoms from a sequence of $M = 13$ frames of size $N = 272 \times 240$, we chose a regularization $\varepsilon = 2$, and a scaling factor between weights and atoms of $\zeta = N/(100 * M)$. We initialized the weights randomly, and atoms with constant values. We used a quadratic loss because it provided the best results in terms of reconstruction and representation. We found that $L = 25$ iterations for the Sinkhorn algorithm is a good trade-off between computation time and precision.

In Figure 3.3, we display four initial frames out of 13, and their respective reconstructions. We also plot the four learned atoms as the vertices of a tetrahedron, which is a way to visualize the barycentric space mentioned earlier. We observe that our algorithm identifies key moments of the sequence, between which frames can be interpolated with optimal transport. The identified atoms do not correspond to input frames because we do not impose sparsity constraints on the weights (which would bring barycentric points closer to the edges of the simplex), but also because

of the bias introduced by the entropic regularization. Since all frames can be recomputed from their barycentric coordinates, the learned representation is indeed a compressed version of the sequence.

Another use case would be anomaly detection: if we learn from a sequence of multiple cycles of a beating heart, looking at the barycentric path would reveal if there are significant swerves in the cyclic trajectory, which would indicate an irregularity. Finally, the learned representation can also be used for super-resolution, as we can linearly interpolate between the barycentric coordinates of the input frames with arbitrary time resolution.

3.4.2 Wasserstein faces

It has been shown that images of faces, when properly aligned, span a low-dimensional space that can be obtained via PCA. These principal components, called EigenFaces, are widely used for face recognition [TP91]. We show that, with the right setting, our dictionary learning algorithm can produce atoms that can be interpreted more easily than their linear counterparts, and can be used to edit a human face's appearance.

We illustrate this application on the MUG facial expression dataset [Aif+10]. From the raw images of the MUG database, we isolated faces and converted the images to grayscale. The resulting images are in Figure 3.4(a). We can optionally invert the colors and apply a power factor γ similar to a gamma-correction. We used a total of $M = 20$ images ($N = 224 \times 224$) of a single person performing 5 facial expressions, and learned dictionaries of $S = 5$ atoms using PCA, NMF, a K-SVD implementation [Rub+08] and our proposed method. For the latter, we set the number of Sinkhorn iterations to $L = 100$ and the maximum number of L-BFGS steps to 450, which ensure acceptable levels of convergence for this application. The weights were randomly initialized and the atoms were initialized as constant.

We performed a cross validation using two datasets, four loss functions, four values for γ $(1, 2.2, 3, 5)$, 3 values of $\zeta(25, 50, 100)$, and colors either inverted or not. We found that none of the γ values we tested gave significantly better results, in terms of reconstruction error. However, $\zeta = 100$ yielded the best reconstructions overall, and interestingly, inverting colors improved the result for our method in most cases. We can conclude that when dealing with faces, it is better to transport the thin and dark zones (eyebrows, mouth, creases) than the large and bright ones (cheeks, forehead, chin).

As illustrated by Figure 3.4, our method reaches similarly successful reconstructions given the low number of atoms, with a slightly higher mean PSNR of 33.8 compared to PSNRs of 33.6, 33.5 and 33.6 for PCA, NMF and K-SVD respectively. An example with another dataset in shown in Figure 3.5.

We show in Figure 3.6 and Figure 3.7 the atoms obtained when using different loss functions, for the two same datasets. This shows how sensible the learned atoms are to the chosen fitting loss, which highlights the necessity for its careful selection if atoms' interpretability is important for the application at hand.

Lastly, we showcase an appealing feature of our method: the atoms that it computes allow for facial editing. We demonstrate this application in Figure 3.8: starting from the isobarycenter of the atoms, by interpolating weights towards a particular atom, we add some of the corresponding emotion to the face.

3.4.3 Literature Learning

We use our algorithm to represent literary work. To this end, we use a bag-of-words representation [SM86], where each book is represented by a histogram of its words. In this particular application, the cost matrix $\mathbf{C}$ (distance between each word) is computed exhaustively and stored. We use a semantic distance between words. These distances were computed from the Euclidian embedding provided by the GloVe database (Global Vectors for Word Representation) [Pen+14]. Our learning algorithm is unsupervised and evaluates similarity between books based on their word histogram, which is representative of their lexical field. Consequently we expect the algorithm to sort books by either author, writing style, or genre.

To demonstrate our algorithm's performance, we created a database of 20 books by 5 different authors. In order to keep the problem size reasonable we only considered words that are between 7 and 8 letters long. In our case, it is better to deal with long words, because they have a higher chance of holding discriminative information than shorter ones.

The results can be seen in Figure 3.9. Our algorithm is able to group the novels by author, recognizing the proximity of lexical fields across the different books. The atom 0 seems to be representing Charlotte Brontë's style, the atoms 1 and 4 Mark Twain's, the atom 2 Arthur Conan Doyle's, and the atom 3 Jane Austen's. Charles Dickens books appear to share an extended amount of vocabulary with the other authors without it differing enough to be represented by its own atom like others are.

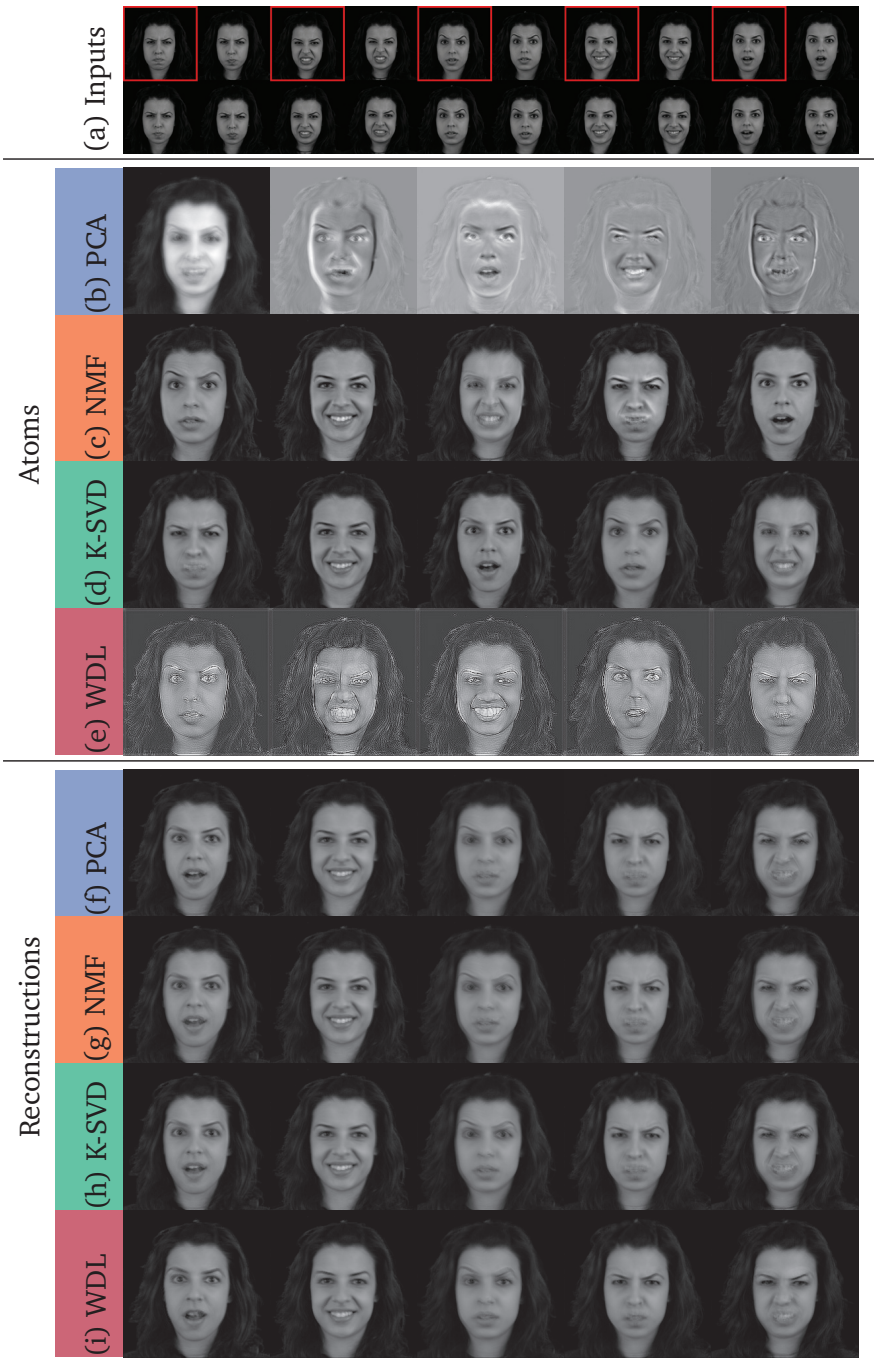


Figure 3.4: We compare our method with Eigenfaces [TP91], Non-negative Matrix Factorization (NMF) and K-SVD [Rub+08] as a tool to represent faces on a low dimensional space. Given a dataset of 20 images of the same person from the MUG dataset [Aif+10] performing 5 facial expressions 4 times (row (a) illustrates each expression), we project the dataset on the first 5 EigenFaces (row (b)). The reconstructed faces corresponding to the highlighted input images are shown in row (f). Row (c) and (d) respectively show atoms obtained using NMF and K-SVD, and row (g) and (h) their respective reconstructions. Using our method with a KL loss, we obtain 5 atoms shown in row (e) that produce the reconstructions in row (i).

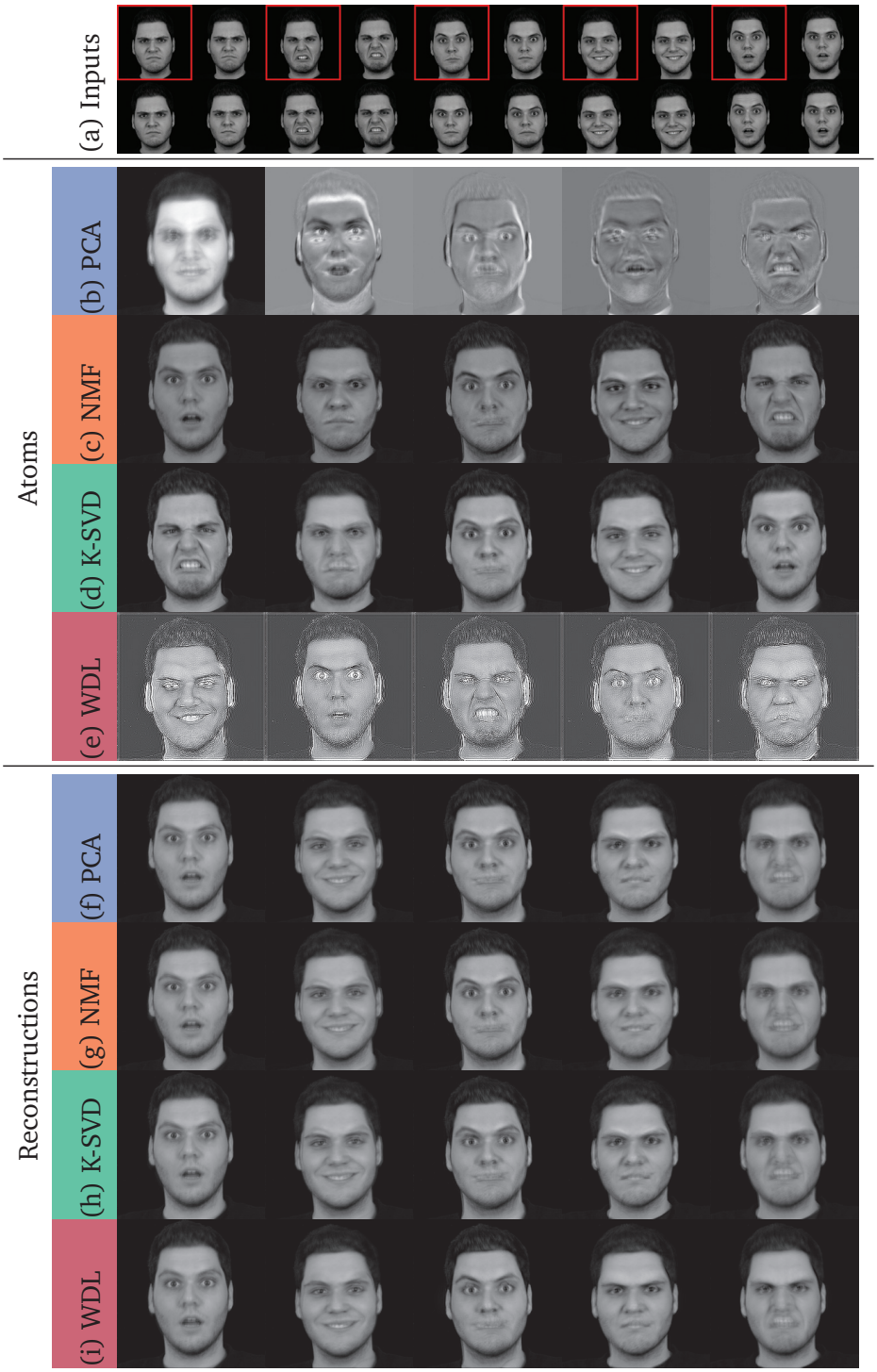


Figure 3.5: Similarly to Figure 3.4, we compare our method (using a KL loss) to the Eigenfaces approach, NMF and K-SVD as a tool to represent faces on a low dimensional space.

Figure 3.6: We compare the atoms (columns 1 to 5) obtained using different loss functions, ordered by fidelity of the reconstructions to the input measures (using the mean PSNR), from best to worst: the Kullback-Leibler divergence (a) $\overline{PSNR} = 32.03$, the quadratic loss (b) $\overline{PSNR} = 31.93$, the total variation loss (c) $\overline{PSNR} = 31.41$ and the Wasserstein loss (d) $\overline{PSNR} = 30.33$. In the last column, we show the reconstruction of the same input image for each loss. We notice that from (a) to (d), the atoms' visual appearance seems to increase even though the reconstruction quality decreases.

Figure 3.7: Similarly to Figure 3.6, we compare the atoms obtained using different loss functions, ranking them by mean PSNR: (a) $\overline{PSNR} = 33.81$, (b) $\overline{PSNR} = 33.72$, (c) $\overline{PSNR} = 32.95$ and (d) $\overline{PSNR} = 32.34$

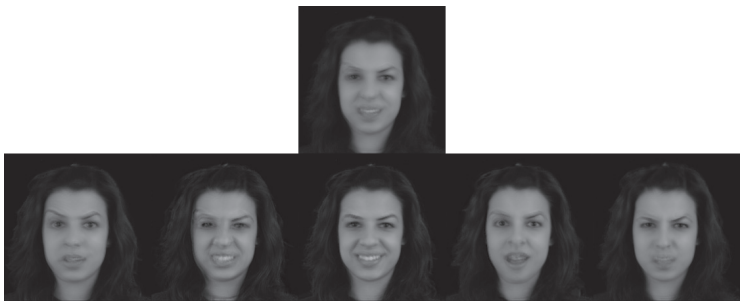


Figure 3.8: Face editing : Using the atoms shown in row (a) of Figure 3.6, we interpolate between the atoms isobarycenter (top image) and each one of the atoms (giving it a relative contribution of 30%). This allows to emphasize each emotion (bottom images) when starting from a neutral face.

This representation could be used together with the Wasserstein barycentric coordinates method [Bon+16] for clustering applications. By projecting a new book (its histogram of words) written by one of these authors onto the simplex composed of the learned atoms, one would obtain coordinates that could describe by which author the book is most likely written.

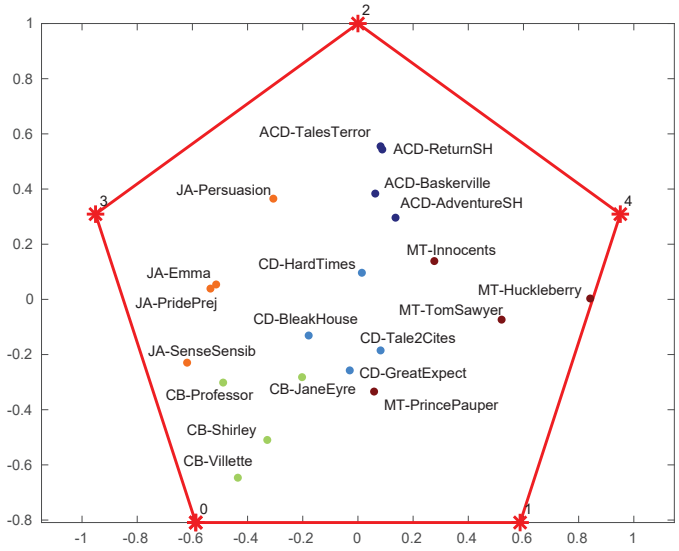


Figure 3.9: Using our algorithm, we look at word histograms of novels, and learn 5 atoms in a sample of 20 books by 5 authors. Each book is plotted according to its barycentric coordinates with regard to the learned atoms as explained in 3.4.1.

3.5 Conclusion

The choice of the number of atoms S to learn is not straightforward, and has to be tailored to each application. As with K-SVD, it is a tradeoff between the time necessary to compute the atoms, and the required accuracy of reconstruction.

We introduced a non-linear dictionary learning approach that uses the optimal transport geometry by fitting data with Wasserstein barycenters of the learned atoms. We presented an algorithm to compute this representation based on the entropic regularization of optimal transport distances, as well as several variants and extensions of our method. We finally illustrated the versatility of our framework on various applications.

Future work. In order to decrease the computational time of the Sinkhorn algorithm, various extensions have been proposed after the publication of our method, such as low-rank approximations of the kernel [Alt+18], a multi-scale approach [GM17] and an over-relaxation of the iterates [Thi+17] that is similar to the one presented in 3.3.3. Implementing these extensions with an automatic differentiation library would avoid calculating the gradients of the new functionals manually.

Moreover, a more robust alternative to our log-domain stabilization scheme for computing sharp barycenters have been proposed by Xie et al. [Xie+18]. It is based on inexact proximal point iterations that converge to the exact optimal solution. It is however less scalable than our approach since it requires multiple computations of the full kernel and transport plan. This method could be used in our framework, but would only be appropriate for small-scale applications requiring high precision, such as the one on point spread functions.

4 Metric Learning

As discussed in the previous chapters, optimal transport (OT) is a powerful tool to compare probability measures on geometric domains (such as Euclidean spaces, surfaces or graphs). The interesting insight provided by OT lies in its ability to leverage prior knowledge on how "close" and related the points of the domain are, or, more generally, how close two observations are. This prior knowledge is usually encoded as a "ground metric" [Rub+00], which defines the cost of moving mass between points.

Because the Wasserstein distance is geodesic, OT can also be used to compute *displacement* interpolations between probability measures, as seen in 2.1.3. When two discrete probability distributions are supported on a Euclidean space, and the ground metric is itself the Euclidean distance (the most widely used setting in applications), theory tells us that the displacement interpolation between these two measures only involves particles moving along straight lines, from a point in the starting measure to another in the end measure. Imagine that, on the contrary, we observe a time series of measures in which mass displacements do not seem to match that assumption. We then cast the following inverse problem: under which ground metric could this observed mass displacement be considered optimal? The goal of our approach here is precisely to answer that question. We give an illustrative example in Figure 4.1, where we show that we learn a ground metric that deforms the space such that the sequence of mass displacements that is observed is close to a Wasserstein geodesic with that ground metric.

First of all, it is possible that the mass displacement does not follow a transport that is optimal for any kind of ground metric, for example when the start and end points are the same (cyclic movements). In our study, we rule out these cases and only consider instances where it is reasonable to assume that there exists such a metric, i.e. under-constrained problems.

The main choice in our approach relies on looking at diffusion-based geodesic distances [YC16] as the space of candidate ground metrics. We then minimize the reconstruction error between measures that are observed at intermediary time stamps and measures interpolated with that ground metric. The problem we tackle is challenging in terms of time and memory complexity, due to repeated calls to

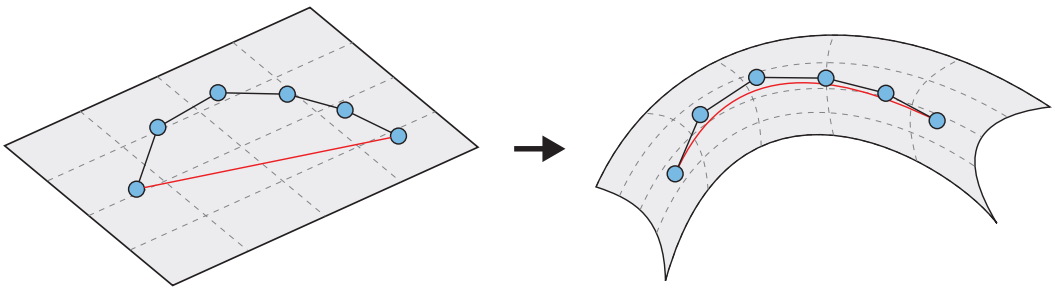


Figure 4.1: Left: before metric learning, the sequence of observed measures (blue points) lies in the Wasserstein space of probability distributions with a Euclidean ground metric. The observed sequence does not match the Wasserstein geodesic (red line) between the first and last element. Right: after modifying the ground metric iteratively, the Wasserstein space is now deformed in such a way that the geodesic between the first and last element in this new geodesic space (red curve) is as close as possible to the sequence.

solve Wasserstein barycenter problems with a non-Euclidean metric, during the non-linear optimization process. We address these issues using a sparse resolution of a diffusion equation, yielding a tractable algorithm. The optimization is performed using a quasi-Newton solver and automatic differentiation to compute the gradients of the functional, through a direct differentiation of Sinkhorn iterations [Bon+16; Gen+17]. However, such approaches can suffer from prohibitive memory footprint, which we can avoid by providing closed-form gradient formulas for the diffusion process. We validate the proposed algorithm on two-dimensional synthetic datasets, and on the learning of color variations in image sequences.

4.1 Problem statement

In this chapter, we consider discrete probability measures on graphs, i.e. sums of weighted Dirac distributions supported on a graph's vertices. Similarly to histograms defined on fixed grids, such measures are defined on a fixed *graph*, thus they can be represented solely by their weight vector, which belongs to the probability simplex Σ_N. We refer to both these discrete measures and their weight vector simply as "measures" for brevity.

We parameterize the metric by a positive weight $w_{i,j}$ associated to the edge connecting vertices i and j. This should be understood as being inversely proportional to the length of the edge, and conveys how easily mass can travel through it. Additionally, we set $w_{i,j} = 0$ when vertices i and j are not connected.

We aim to carry out metric learning using OT where the ground cost is the square of the geodesic distance associated to the weighted graph. Instead of optimizing a full adjacency matrix $\mathbf{W} = (w_{i,j})_{i,j}$ which has many zero entries that we do not wish to optimize, we define the vector $\boldsymbol{w}$ as the concatenation of all non-zero metric parameters $w_{i,j} > 0$, that is, for which vertices i and j are connected. This imposes a fixed connectivity on the graph, and we denote by $\mathcal{N}(i)$ the set of neighboring vertices of vertex i.

In order to define an efficient and regular functional, we consider Varadhan's formula [Var67] as in Solomon et al. [Sol+15], which relates the solution of the heat equation to geodesic distances. The transport cost is then defined as the log of the heat kernel, and this kernel is itself approximated using S sub-steps of an implicit Euler scheme. Our ground metric formulation is thus

$$\mathbf{C}_{\boldsymbol{w}} \overset{\text{def.}}{=} -\varepsilon \log\left((\mathbf{Id} - \frac{\varepsilon}{4S}\mathbf{L}_{\boldsymbol{w}})^{-S} \right), \tag{4.1}$$

where $\mathbf{L}_{\boldsymbol{w}}$ is a Laplacian operator parameterized by the graph weights $\boldsymbol{w}$. In practice, we never explicitly compute $\mathbf{C}_{\boldsymbol{w}}$, but directly use the diffusion kernel

$$\mathbf{K} \overset{\text{def.}}{=} (\mathbf{Id} - \frac{\varepsilon}{4S}\mathbf{L}_{\boldsymbol{w}})^{-S} = \exp\left(-\mathbf{C}_{\boldsymbol{w}}/\varepsilon\right). \tag{4.2}$$

Details concerning these computations can be found in section 4.2.2.

We now apply our metric formulation (4.1) to the case where the input data is a dynamic evolution of density, which we model as a displacement interpolation. Let $(\boldsymbol{h}_{t_i})_{i=1}^{P} \in \Sigma_N$ be observations at P consecutive time steps t_i of a movement of mass. We aim to retrieve the metric weights $\boldsymbol{w}$ for which an OT displacement interpolation approximates best this mass evolution. This corresponds to an OT regression scheme parameterized by the metric, and leads to the following optimization problem:

$$\min_{\boldsymbol{w}} \sum_{i=1}^{P} \mathcal{L}\left(\boldsymbol{\gamma}^{\varepsilon}_{\mathbf{C}_{\boldsymbol{w}}}(\boldsymbol{h}_{t_1}, \boldsymbol{h}_{t_P}, t_i), \boldsymbol{h}_{t_i}\right) + f(\boldsymbol{w}), \tag{4.3}$$

where $\boldsymbol{\gamma}^{\varepsilon}_{\mathbf{C}_{\boldsymbol{w}}}(\boldsymbol{h}_{t_1}, \boldsymbol{h}_{t_P}, t_i)$ is the measure interpolated at time t_i between $\boldsymbol{h}_{t_1}$ and $\boldsymbol{h}_{t_P}$ (as defined in (2.22)), $\mathcal{L}$ is a loss function between measures, and $f(\boldsymbol{w})$ is a regularization term detailed in 4.2.3.

In the numerical examples, we consider 2-D and 3-D datasets discretized on uniform square grids, so that the graph is simply the graph of 4 or 6 nearest neighbors on this grid.

4.2 Method

In this section, we detail the different components of the proposed algorithm. The objective function (4.3) is non-convex, and we minimize it with an L-BFGS quasi-Newton algorithm, to compute a local minimum of the energy. The L-BFGS algorithm requires the evaluation of the energy function, as well as its gradient with respect to the inputs. In this case, evaluating the energy function (4.3) requires reconstructing the sequence of input measures using a displacement interpolation (through Wasserstein barycenters) between the first and last measure, and assessing the quality of the reconstructions. The gradient is calculated through automatic differentiation, which provides high flexibility when adjusting the framework.

4.2.1 Alternative strategies to displacement interpolation

As mentioned in 2.1.3, we compute displacement interpolations using regularized Wasserstein barycenters (see Algorithm 2), for which the main computational burden is to apply the kernel matrix $\mathbf{K}$ on $\mathbb{R}^N$ vectors. We briefly describe an attempt to compute displacement interpolations with a cheaper solution.

Solomon et al. [Sol+15] proposed the *Wasserstein propagation* algorithm, which generalizes the Wasserstein barycenter algorithm, and allows to compute displacement interpolations with around half the number of kernel calls. Unfortunately, our experiments show that this algorithm accumulates the blur introduced by the regularization, across the interpolation: as we get towards the middle of the interpolation, measures get more diffuse at each step, with the middle interpolation being the most affected. This method is only applicable if the diffusiveness of the interpolations does not impede the effectiveness of the application, or if one can achieve a very low blur at the end points resulting in a reasonable blur in the middle. In our case, a varying blur in the interpolations prevents a good reconstruction of the inputs, and we cannot achieve a very small blur since small values of ε yield a poor approximation of the heat kernel (see Figure 4.9). Consequently, this algorithm cannot be used in our setting. A comparison of a displacement interpolation obtained with the Sinkhorn barycenter algorithm and the Wasserstein propagation is presented in Figure 4.2.

Figure 4.2: Comparison of a displacement interpolation between two densities ($\varepsilon = 1.10^{-3}$), with the Sinkhorn barycenter algorithm (top row), and the Wasserstein propagation algorithm (bottom row). We observe an accumulation of the blurring effect for the Wasserstein propagation algorithm.

4.2.2 Computing geodesic distances

When the domain is a grid and the metric is Euclidean, applying the kernel boils down to a separable convolution with a Gaussian kernel, as discussed in 3.3.1. However, for an arbitrary metric as in this case, computing the kernel $\mathbf{K}$ requires all-pairs geodesic distances on the domain, and applying it during the Sinkhorn iterations requires $O(N^2)$ operations per iteration (matrix-vector multiplication). Having discretized the metric on the graph's edges, computing the all-pairs squared geodesic distances matrix $\mathbf{C}_w$ can be achieved using either a graph approach, or a diffusion-based approach.

Graph approach

A classical method to compute shortest paths on a graph is Dijkstra's algorithm. Computing the geodesic distance between one point and all the others is achieved in $O(N \log N)$ operations so all-pairs geodesic distances are obtained in $O(N^2 \log N)$. However, Dijkstra's geodesic distances are non-smooth with respect to the graph weights. Moreover, since efficient implementations of Dijkstra's algorithm rely on non differentiable operations (insertion and removal on priority queues), one would need a differentiable implementation of it. Such an implementation would be too costly and simpler methods exist.

Another method is the Floyd-Warshall algorithm (Algorithm 5) which directly computes all-pairs geodesic distances in $O(N^3)$ operations. The computed distances are still non-smooth, but the individual operations of the algorithm (additions and minimums) *can* be differentiated algorithmically.

However, one remaining problem with these approaches is that they require the full calculation and storage of the geodesic distance matrix (which is of size N^2), in order

Algorithm 5 `Floyd-Warshall`: Computation of all-pairs geodesic distances in a graph

Inputs: $\mathbf{A} \in \mathbb{R}^{n \times n}$: weighted adjacency matrix of the graph, with $0 < \mathbf{A}_{i,j} < \infty$ the length of the edge between vertices i and j, and $\mathbf{A}_{i,j} = \infty$ if i and j are not connected.

for $i = 1 \dots n$ **do**
  for $j = 1 \dots n$ **do**
    for $k = 1 \dots n$ **do**
      $\mathbf{A}_{i,j} = \min(\mathbf{A}_{i,j}, \mathbf{A}_{i,k} + \mathbf{A}_{k,j})$
    end for
  end for
end for

Outputs: $\mathbf{A}$ containing all-pairs geodesic distances

to build the kernel $\mathbf{K}$ for the Sinkhorn algorithm. This quickly becomes prohibitive in time and memory as the number of points grows ($\approx$ 12GB for measures with $N = 200^2$ points, and $\approx$ 30GB for measures with $N = 40^3$ points).

Diffusion-based approach

In sharp contrast, this section details our approach which leverages Varadhan's formula [Var67] to have faster evaluations of the kernel, and smooths the dependency between the geodesic distance kernel and the metric, which makes it differentiable.

We approximate the kernel $\mathbf{K}$ by the heat kernel associated to geodesic distances on the domain, as done by Solomon et al. [Sol+15] for computing OT on discrete surfaces. Indeed, it has been shown by Varadhan [Var67] that the solution to the heat equation (with variable coefficients) can be used to obtain geodesic distances on a Riemannian manifold. Let us consider the heat equation with a Dirac as initial heat distribution:

$$\begin{cases} \frac{\partial h(x,y,t)}{\partial t} = \Delta_g h(x,y,t) \\ h(x,y,0^+) = \delta_y(x) \end{cases} \tag{4.4}$$

where Δ_g is the Laplace operator parametrized by the Riemannian metric tensor g, and d_g is the geodesic distance on the manifold. Varadhan's second formula is

$$\lim_{t \to 0} [-4t \log(h(x,y,t))] = d_g^2(x,y). \tag{4.5}$$

This means that $h(x,y,t) \overset{t\to 0}{\sim} \exp(-d_g^2(x,y)/4t)$, i.e. $h(x,y,t)$ approximates the heat kernel on the manifold when t is small. The end goal is not to compute the full kernel as this has a prohibitive cost in memory for a large number of points. For the Sinkhorn algorithm, we only need to apply the kernel matrix to functions stored as vectors. This can also be achieved by solving the heat equation with that function as initial data, i.e. smoothing it through diffusion up to a small time τ. We then denote the initial function $u_0(x)$ and solve

$$\begin{cases} \frac{\partial u(x,t)}{\partial t} = \Delta_g u(x,t) \\ u(x,0^+) = u_0(x). \end{cases} \tag{4.6}$$

We discretize this equation in time using an implicit Euler scheme and perform S sub-steps so that the finite difference approximation stays accurate. It is crucial to rely on an implicit stepping scheme to obtain approximated kernels supported on the full domain, in order for Sinkhorn iterations to be well conditioned (as opposed to using an explicit Euler scheme which would break Sinkhorn's convergence). With $\delta t = \tau/S$, let $t_s = s\delta t, s \in [0,S]$, which yields $t_0 = 0$ and $t_S = \tau$. From Equation 4.6, we get

$$\frac{u(x,t+\delta t) - u(x,t)}{\delta t} = \Delta_g u(x,t+\delta t) \tag{4.7}$$

$$\Rightarrow (Id - \delta t \Delta_g)\, u(x,t+\delta t) = u(x,t). \tag{4.8}$$

We now discretize the equation in space. While Solomon et al. [Sol+15] discretize Δ_g using a cotangent Laplacian because they deal with triangular meshes, we prefer a weighted graph Laplacian parameterized by the metric weights $\boldsymbol{w}$ (detailed hereafter), since the probability distributions we consider are defined on graphs. The weighted adjacency matrix $\mathbf{W}$ is defined as $\mathbf{W}_{i,j} = \mathbf{W}_{j,i} = w_{i,j}$ where $w_{i,j}$ are the edge weights parameterizing the metric. It is symmetric and usually sparse, since $w_{i,j}$ is non-zero only for vertices that are connected, and 0 otherwise. The diagonal weighted degree matrix sums for each vertex the weights of its adjacent edges: $\mathbf{\Lambda} \overset{\text{def.}}{=} \operatorname{diag}(\boldsymbol{d})$, with $\boldsymbol{d}_i \overset{\text{def.}}{=} \sum_{j=1}^N w_{i,j} = \sum_{j\in\mathcal{N}(i)} w_{i,j}$. The negative semi-definite weighted graph Laplacian matrix is then defined as $\mathbf{L}_{\boldsymbol{w}} = \mathbf{W} - \mathbf{\Lambda}$.

Writing $\boldsymbol{u}_s$ the approximation of $u(x,t_s)$ at the N vertices x_i, Equation 4.8 gives

$$\left(\mathbf{Id} - \frac{\tau}{S}\mathbf{L}_{\boldsymbol{w}}\right) \boldsymbol{u}_{s+1} = \boldsymbol{u}_s, \tag{4.9}$$

which is a diffusion equation rather than a heat equation since it has variable diffusion coefficients (enclosed in $\mathbf{L}_{\boldsymbol{w}}$). Starting with the initial function $\boldsymbol{u}_0 = u_0(x)$,

one can iteratively obtain the final solution $\boldsymbol{u}_S = u(x, \tau)$, which approximates the diffusion of $u_0(x)$ after a short time τ:

$$\left(\mathbf{Id} - \frac{\tau}{S}\mathbf{L}_{\boldsymbol{w}}\right)^S \boldsymbol{u}_S = \boldsymbol{u}_0. \tag{4.10}$$

In order to get a kernel of the form $\mathbf{K} = \exp(-\mathbf{C}_{\boldsymbol{w}}/\varepsilon)$, Varadhan's formula (4.5) shows that the diffusion time should be chosen as $\tau = \varepsilon/4$. Therefore, applying the kernel $\mathbf{K}$ to a vector $\boldsymbol{v}$ is simply carried out by solving S sparse linear systems with the matrix $\mathbf{M} \overset{\text{def.}}{=} \mathbf{Id} - \frac{\varepsilon}{4S}\mathbf{L}_{\boldsymbol{w}}$:

$$\boldsymbol{u} = \mathbf{K}\boldsymbol{v} = \mathbf{M}^{-S}\boldsymbol{v} = \left(\mathbf{Id} - \frac{\varepsilon}{4S}\mathbf{L}_{\boldsymbol{w}}\right)^{-S} \boldsymbol{v}. \tag{4.11}$$

As stated in section 4.1, this method can be seen as choosing a cost of the form

$$\mathbf{C}_{\boldsymbol{w}} = -\varepsilon \log\left((\mathbf{Id} - \frac{\varepsilon}{4S}\mathbf{L}_{\boldsymbol{w}})^{-S}\right). \tag{4.12}$$

The chief advantage of the formula (4.11) to approximate a kernel evaluation is that the same matrix $\mathbf{M}$ is repeatedly used S times, which is itself repeated at each iteration of the Sinkhorn barycenter algorithm (2). Following Solomon et al. [Sol+15], a dramatic speed-up is thus obtained by pre-computing a sparse Cholesky decomposition of $\mathbf{M}$, which is sparse and positive definite. For instance, on a 2-D domain, the number of non-zero elements of such a factorization is of the order of N, so that each linear system resolution has linear complexity.

4.2.3 Inverse Problem Regularization

The metric learning problem is severely ill-posed and this difficulty is further increased by the fact that the corresponding optimization problem (4.3) is non-convex. These issues can be mitigated by introducing a regularization term $f(\boldsymbol{w})$.

We introduce two different regularizations: $f(\boldsymbol{w}) \overset{\text{def.}}{=} \lambda_c f_c(\boldsymbol{w}) + \lambda_s f_s(\boldsymbol{w})$. The term f_c forces the metric weights (similar to diffusion coefficients) to be close to 1: this controls how much the space becomes inhomogeneous and anisotropic, and impedes the weights from diverging to infinity during the optimization process. The term f_s constrains the weights to be spatially smooth, which reduces the number of local minima in the energy landscape. Since we carry out the numerical examples on graphs that are 2-D and 3-D grids, we use a smoothing regularization f_s that is specific to that case. This term must be adapted when dealing with general graphs.

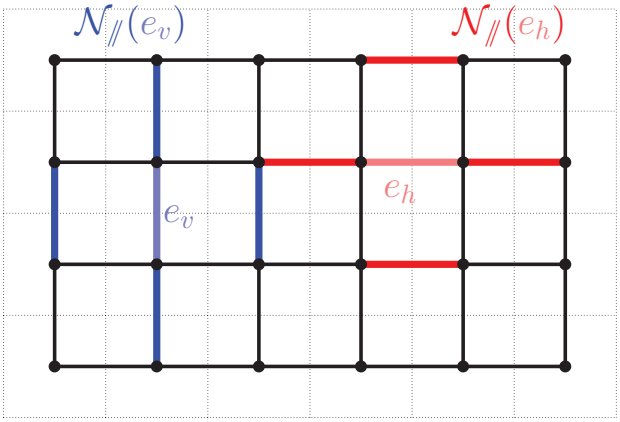


Figure 4.3: Smooth prior: for each vertical (e_v) and horizontal (e_h) edge, we minimize the squared sum of weight differences with its respective neighbors of the same orientation.

The first regularization is imposed by adding the following term to the energy functional, multiplied by a control coefficient λ_c:

$$f_c(\boldsymbol{w}) \overset{\text{def.}}{=} ||\boldsymbol{w} - \mathbb{1}||_2^2 \,. \tag{4.13}$$

To enforce the second prior, we add the following term to the functional, multiplied by a control coefficient λ_s:

$$f_s(\boldsymbol{w}) \overset{\text{def.}}{=} \sum_{e \in E} \left(\sum_{e' \in \mathcal{N}_{/\!/}(e)} (w_e - w_{e'}) \right)^2 , \tag{4.14}$$

with E the set of edges, and $\mathcal{N}_{/\!/}$ the set of neighbor edges of the same orientation, as illustrated in Figure 4.3 for the 2-D case.

We regularize separately horizontal and vertical edges to ensure that we recover an anisotropic metric. This is important for various applications, for example when dealing with color histograms, as MacAdam's ellipses reveal [Mac42].

The selection of the regularization parameters (λ_c, λ_s) and their impact on the recovered metric is discussed in section 4.4.

4.2.4 Implementation

In order to ensure positivity of the metric weights, problem (4.3) is solved after a log-domain change of variable $\boldsymbol{w} = e^{\boldsymbol{z}}$, and the optimization on $\boldsymbol{z}$ is achieved using the L-BGFS algorithm.

In order to evaluate the gradient, we use automatic differentiation (AD), which allows to compute the gradient of a function without having to know its explicit formula. It is based on the idea that any function evaluated on a computer is simply a sequence of arithmetic operations. Knowing the derivative of each operation, one can compute the derivative of the entire function by applying the chain rule recursively, with a computational cost that is only a small constant factor times the function evaluation itself [GW08]. Reverse accumulation (or reverse mode) AD [Gri12] is a particular way of computing a function's derivative, in which the gradient is recursively accumulated from the output variable to the input variable. As a result, the function needs to be completely evaluated (the forward pass), before computing its gradient (the backward pass). AD has become popular for learning algorithms, in particular for neural networks, where it is used for the back-propagation of errors through the net [Bay+18].

We implemented our method in Python, using the PyTorch framework which supports automatic differentiation [Pas+17] implemented as reverse mode. During the backward pass, the gradient evaluation requires the computation of the adjoint of the Jacobian of each elementary operation of the algorithm. The only non-trivial operation that is necessary to implement is the Jacobian of the matrix inversion.

Differentiating the matrix inversion Since we use a reverse accumulation AD framework, the gradient is computed from the end variable to the leaf variable (the one with respect to which we require the gradient). If a pipeline is of the form $y_i = f_i(y_{i-1})$, with $y_0 = x$ the leaf variable and $z = y_m$ the end variable (which in the case of optimization, is the value of the energy function), the gradient $\frac{\partial z}{\partial x}$ is recursively computed with

$$\frac{\partial z}{\partial y_i} = \frac{\partial z}{\partial y_{i+1}} \frac{\partial y_{i+1}}{\partial y_i}. \tag{4.15}$$

In the multivariate case, this extends to matrix multiplication of the adjoints (transpose for real matrices) of the Jacobians:

$$\left(\frac{\partial z}{\partial y_i}\right)^T = \left(\frac{\partial y_{i+1}}{\partial y_i}\right)^T \left(\frac{\partial z}{\partial y_{i+1}}\right)^T \tag{4.16}$$

$$= \left(\frac{\partial y_{i+1}}{\partial y_i}\right)^T \left(\frac{\partial y_{i+2}}{\partial y_{i+1}}\right)^T \cdots \left(\frac{\partial z}{\partial y_{m-1}}\right)^T. \tag{4.17}$$

It is not necessary to compute Jacobians $\frac{\partial y_{i+1}}{\partial y_i}$ fully, it suffices to be able to apply them on the input gradient $\frac{\partial z}{\partial y_{i+1}}$ to get the output gradient $\frac{\partial z}{\partial y_i}$.

Let us denote z the final energy value of the function we minimize (4.3), and $\tilde{\boldsymbol{v}} = \mathbf{M}^{-1}\boldsymbol{v}$ a vector $\boldsymbol{v}$ diffused after one step of the implicit Euler scheme presented in (4.11). The automatic differentiator computes the gradient $\boldsymbol{g} = \frac{\partial z}{\partial \tilde{\boldsymbol{v}}}$, and since both $\mathbf{M}$ and $\boldsymbol{v}$ can be metric-dependent, we need to compute $\frac{\partial z}{\partial \boldsymbol{v}}$ and $\frac{\partial z}{\partial \mathbf{M}}$. From $dz = \left\langle \frac{\partial z}{\partial \tilde{\boldsymbol{v}}}, d\tilde{\boldsymbol{v}} \right\rangle$, we have

$$\begin{aligned}
dz &= \left\langle \boldsymbol{g}, d\left(\mathbf{M}^{-1}\boldsymbol{v}\right) \right\rangle \\
&= \left\langle \boldsymbol{g}, d(\mathbf{M}^{-1})\boldsymbol{v} + \mathbf{M}^{-1}d\boldsymbol{v} \right\rangle \\
&= \left\langle \boldsymbol{g}, -\mathbf{M}^{-1}d\mathbf{M}\mathbf{M}^{-1}\boldsymbol{v} \right\rangle + \left\langle \boldsymbol{g}, \mathbf{M}^{-1}d\boldsymbol{v} \right\rangle \\
&= \left\langle -\left(\mathbf{M}^{-1}\right)^T \boldsymbol{g} \left(\mathbf{M}^{-1}\boldsymbol{v}\right)^T, d\mathbf{M} \right\rangle + \left\langle \left(\mathbf{M}^{-1}\right)^T \boldsymbol{g}, d\boldsymbol{v} \right\rangle \\
&= \left\langle -\left(\mathbf{M}^{-1}\boldsymbol{g}\right)\left(\mathbf{M}^{-1}\boldsymbol{v}\right)^T, d\mathbf{M} \right\rangle + \left\langle \mathbf{M}^{-1}\boldsymbol{g}, d\boldsymbol{v} \right\rangle.
\end{aligned}$$

Consequently, the two gradients of interest are:

$$\begin{aligned}
\boldsymbol{h} &\overset{\text{def.}}{=} \frac{\partial z}{\partial \boldsymbol{v}} = \mathbf{M}^{-1}\boldsymbol{g} \\
\mathbf{H} &\overset{\text{def.}}{=} \frac{\partial z}{\partial \mathbf{M}} = -\left(\mathbf{M}^{-1}\boldsymbol{g}\right)\left(\mathbf{M}^{-1}\boldsymbol{v}\right)^T = -\boldsymbol{h}\tilde{\boldsymbol{v}}^T.
\end{aligned}$$

The vector $\tilde{\boldsymbol{v}} = \mathbf{M}^{-1}\boldsymbol{v}$ has been computed in the forward pass and should be stored for reuse in the backward pass. We then only need to compute $\boldsymbol{h} = \mathbf{M}^{-1}\boldsymbol{g}$.

Since the automatic differentiation library we used (PyTorch v1.0.0) does not differentiate sparse tensors, we have to provide the gradient of the loss z with respect to the weights $\boldsymbol{w}$, which is the first variable that is not a sparse matrix when browsing the computational graph backwards (from the end variable to the leaf).

Recall that we built the matrix $\mathbf{M}$ from $w_{i,j}$ in the following manner:

$$\mathbf{W}_{i,j} \overset{\text{def.}}{=} w_{i,j} \tag{4.18}$$

$$\boldsymbol{d}_i \overset{\text{def.}}{=} \sum_{j=1}^{N} w_{i,j} = \sum_{j \in \mathcal{N}(i)} w_{i,j} \tag{4.19}$$

$$\boldsymbol{\Lambda} \overset{\text{def.}}{=} \operatorname{diag}(\boldsymbol{d}) \tag{4.20}$$

$$\mathbf{L} \overset{\text{def.}}{=} \mathbf{W} - \boldsymbol{\Lambda} \tag{4.21}$$

$$\mathbf{M} \overset{\text{def.}}{=} \mathbf{Id} - \frac{\varepsilon}{4S}\mathbf{L}. \tag{4.22}$$

Next, we have:

$$\frac{\partial z}{\partial w_{i,j}} = \sum_{k,l} \frac{\partial z}{\partial \mathbf{M}_{k,l}} \frac{\partial \mathbf{M}_{k,l}}{\partial w_{i,j}} = \sum_{k,l} \mathbf{H}_{k,l} \frac{\partial \mathbf{M}_{k,l}}{\partial w_{i,j}} = -\frac{\varepsilon}{4S} \sum_{k,l} \mathbf{H}_{k,l} \frac{\partial \mathbf{L}_{k,l}}{\partial w_{i,j}}. \tag{4.23}$$

We can rewrite (4.21) as

$$\mathbf{L}_{k,l} = w_{k,l} - \delta_{k,l} \sum_{m \in \mathcal{N}(k)} w_{k,m}, \tag{4.24}$$

where $\delta_{k,l}$ is the Kronecker delta.
Thus, if $k \neq l$,

$$\frac{\partial \mathbf{L}_{k,l}}{\partial w_{i,j}} = \frac{\partial}{\partial w_{i,j}}(w_{k,l}) = \begin{cases} 1 & \text{if } (k,l) = (i,j) \text{ or } (k,l) = (j,i) \\ 0 & \text{otherwise} \end{cases} \tag{4.25}$$

and, if $k = l$,

$$\frac{\partial \mathbf{L}_{k,l}}{\partial w_{i,j}} = \frac{\partial}{\partial w_{i,j}} \left(- \sum_{m \in \mathcal{N}(k)} w_{k,m} \right) = \begin{cases} -1 & \text{if } k = i \text{ or } k = j \\ 0 & \text{otherwise.} \end{cases} \tag{4.26}$$

Consequently, keeping only the non-zero terms in the sum (4.23), we obtain:

$$\frac{\partial z}{\partial w_{i,j}} = -\frac{\varepsilon}{4S}(\mathbf{H}_{i,j} + \mathbf{H}_{j,i} - \mathbf{H}_{i,i} - \mathbf{H}_{j,j}). \tag{4.27}$$

$$= -\frac{\varepsilon}{4S}(\boldsymbol{h}_i \tilde{\boldsymbol{v}}_j + \boldsymbol{h}_j \tilde{\boldsymbol{v}}_i - \boldsymbol{h}_i \tilde{\boldsymbol{v}}_i - \boldsymbol{h}_j \tilde{\boldsymbol{v}}_j) \tag{4.28}$$

$$= \frac{\varepsilon}{4S}(\boldsymbol{h}_i - \boldsymbol{h}_j)(\tilde{\boldsymbol{v}}_i - \tilde{\boldsymbol{v}}_j) \tag{4.29}$$

Differentiating the conjugate gradient We experimented with another way of differentiating the matrix inversion, which arises in the case where we solve the linear system with an iterative method such as the conjugate gradient (CG) instead of the

Cholesky factorization. In order to solve $\mathbf{M}\boldsymbol{u} = \boldsymbol{v}$, the CG algorithm only performs multiplications of the matrix $\mathbf{M}$ with vectors. Therefore, we only need to provide the gradient of the matrix multiplication, with respect to the metric weights $\boldsymbol{w}$, and let the automatic differentiator operate on the CG iterations. This technique is however associated with a high memory footprint even for small datasets. Indeed, intermediate variables are stored for each CG iteration, and one evaluation of the energy function performs L Sinkhorn iterations, each applying the kernel 4 times, each application requiring S calls to the CG algorithm. There are of course checkpointing techniques that keep a fraction of the results of the forward pass and recompute the other ones from these "snapshots" during the backward pass. However, these techniques trade memory for computation time, which can already be large for our applications (see 4.1).

4.3 Experiments

We first show the influence of the metric in an example of the direct problem. We then show a few synthetic examples in which the input sequence has been generated as a Wasserstein geodesic using a ground metric known beforehand. This ground truth metric is compared with the output of our algorithm. We then present an application to a task of learning color variations in image sequences.

In the following, an "interpolation" refers to a displacement interpolation, unless stated otherwise.

4.3.1 Direct problem

The direct problem in this setting is to obtain the displacement interpolation between two measures, under a given metric. We solve it with the Sinkhorn barycenter algorithm (2.1.4) using the metric-parameterized diffusion kernel (4.2). In order to grasp the role of the metric in optimal transport, we show in Figure 4.4 the results of an interpolation with a Euclidean metric, and with a simple non-Euclidean metric. In the former case, the mass travels in a straight line since the diffusion is homogeneous. In the latter case, the mass does not travel in a straight line, but avoids the central region, where the diffusion coefficient is low, i.e where it is more costly to travel.

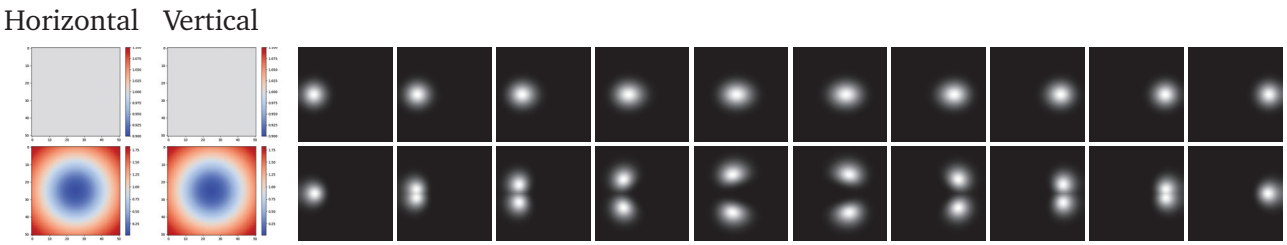


Figure 4.4: The direct problem: for a given metric (two leftmost columns: weights on horizontal and vertical edges), compute the displacement interpolation between two measures (other columns). This figure compares the interpolation between two measures, using a Euclidean metric (top row), and a non-Euclidean metric (bottom row). In the metric images, low values (in blue) indicate a low diffusion coefficient, whereas high values (in red) indicate a high diffusion coefficient.

4.3.2 Synthetic experiments in 2-D

As mentioned in 4.1, the proposed algorithm solves an inverse problem: given a sequence of measures representing a movement of mass, we aim at fitting a metric for which that sequence can be sufficiently well approached by a displacement interpolation between the first and last frame. We test the algorithm by applying it on different sequences of measures that are themselves geodesics generated using handcrafted metrics, and verify that the learned metric is close to the original one. In general, it is impossible to recover with high precision the exact same metric, because such an inverse problem is too ill-posed (many different metrics can generate the same interpolation sequence) and the energy non-convex. Moreover, regularization introduces a bias while helping to fight against this non-convexity. In view of this, we attempt to find a metric that shares the same large scale features as the original one.

We run three experiments with different handcrafted metrics, presented in Figure 4.5. The domain's graph we use here is a 2-D grid of size $n \times n$. The parameters used for these experiments are: a grid of size $N = 50^2$, $L = 50$ Sinkhorn iterations, an entropic regularization factor $\varepsilon = 1.2e - 2$, $S = 100$ sub-steps for the diffusion equation, 1000 L-BFGS iterations, and the metric regularization factor $\lambda_c = 0$. The other regularization factor is $\lambda_s = 0.03$ for the first two experiments and $\lambda_s = 1.0$ for the third one. Finally, each of the three experiments is tested with three different loss functions, and we display the result which is closest to the ground truth. The

different loss functions are the L^1 norm, the squared L^2 norm and the Kullback-Leibler divergence:

$$\mathcal{L}_1(\boldsymbol{p}, \boldsymbol{q}) \overset{\text{def.}}{=} ||\boldsymbol{p} - \boldsymbol{q}||_1 \,, \tag{4.30}$$

$$\mathcal{L}_2(\boldsymbol{p}, \boldsymbol{q}) \overset{\text{def.}}{=} ||\boldsymbol{p} - \boldsymbol{q}||_2^2 \,, \tag{4.31}$$

$$\mathcal{L}_{KL}(\boldsymbol{p}, \boldsymbol{q}) \overset{\text{def.}}{=} \mathbb{1}^T(\boldsymbol{p} \odot \log(\boldsymbol{p}/\boldsymbol{q}) - \boldsymbol{p} + \boldsymbol{q}), \tag{4.32}$$

with $\odot$ being the entry-wise multiplication. We will see that the best loss function varies depending on the data.

The metric $\boldsymbol{w}$ is located along either vertical or horizontal edges. We thus display two images each time, one for the horizontal and one for the vertical edges.

In the first experiment, we are able to reconstruct the input sequence, and retrieve the main zones of low diffusion (in blue), that deviate the mass from a straight trajectory. The loss $\mathcal{L}_1$ gives the best result.

In the second experiment, the original horizontal and vertical metric weights are different and this experiment shows that we are able to recover the distinct features of each metric i.e. the dark blue and dark red areas. The loss $\mathcal{L}_{KL}$ gives the best result.

In the third experiment, the original metric is composed of two obstacles, but only one of them is in the mass' trajectory. We observe that obstacles that are not approached by any mass are not recovered, which is expected, because the algorithm cannot find information in these areas. The loss $\mathcal{L}_2$ gives the best result.

4.3.3 Evaluation

Regularization. In order to evaluate the influence of the regularization, we compare the same experiment (the second one conducted in Figure 4.5), with one of the two regularizers (f_c and f_s). The first regularizer f_c effectively stabilizes the values around 1, but the recovered metric is noisy, with patterns that reflect over-fitting. The second regularizer f_s effectively produces a smooth metric, but we note that the metric values have drawn away from their initial value of 1. After experimenting with each one, we observed that while reconstruction errors are smaller with f_c (which is another sign of overfitting), the regularizer f_s produces more interpretable results, and allows the global metric scale to shift reasonably in order to adapt to the input sequence. Moreover, combining both generally does not significantly change the result compared to having only f_s. We thus concluded that the smoothing term

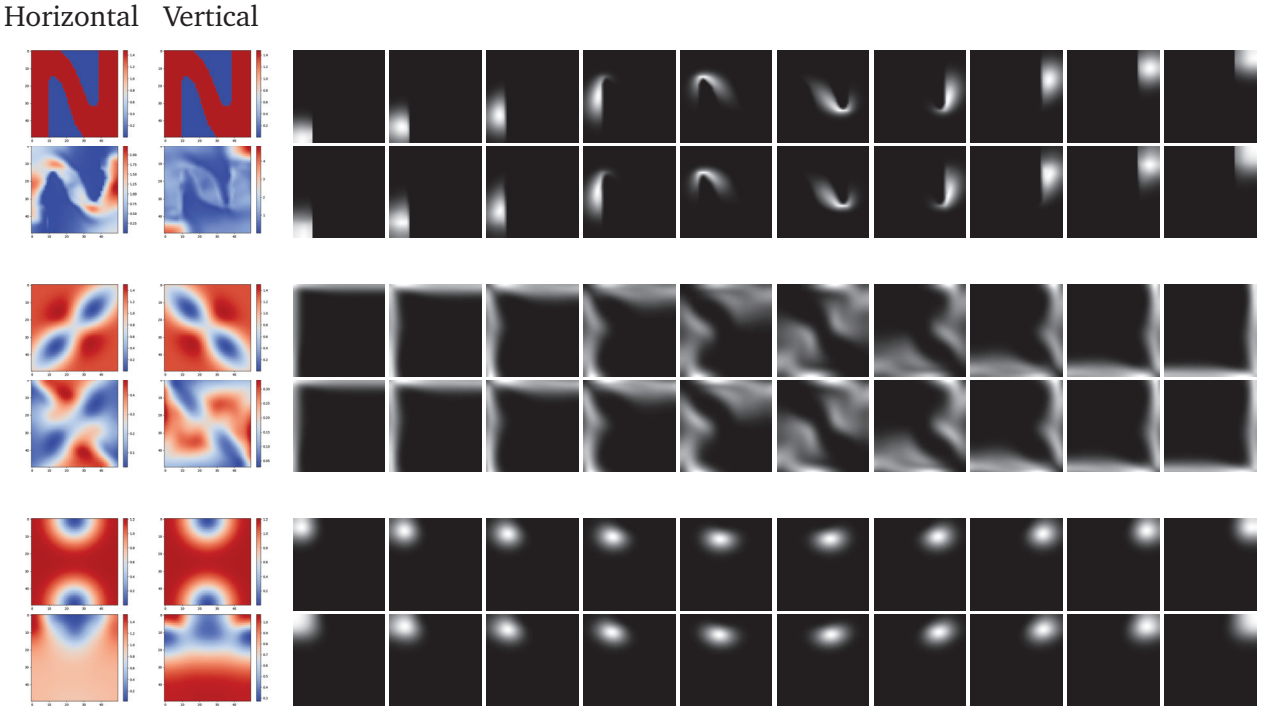


Figure 4.5: For each experiment (each pair of rows): The first row is the initial metric (two leftmost images) and the sequence of measures generated with it. The second row is the metric learned by our algorithm on the measure sequence above, and the reconstructed sequence. In the first experiment, the algorithm is able to recover the blue zones that are avoided by the mass, and red zones on the path it is taking. In the second experiment, the algorithm recovers the high (red) and low (blue) diffusion areas horizontally, as well as vertically. In the third experiment, we show an example of a metric detail not being recovered because mass is not traveling in that region.

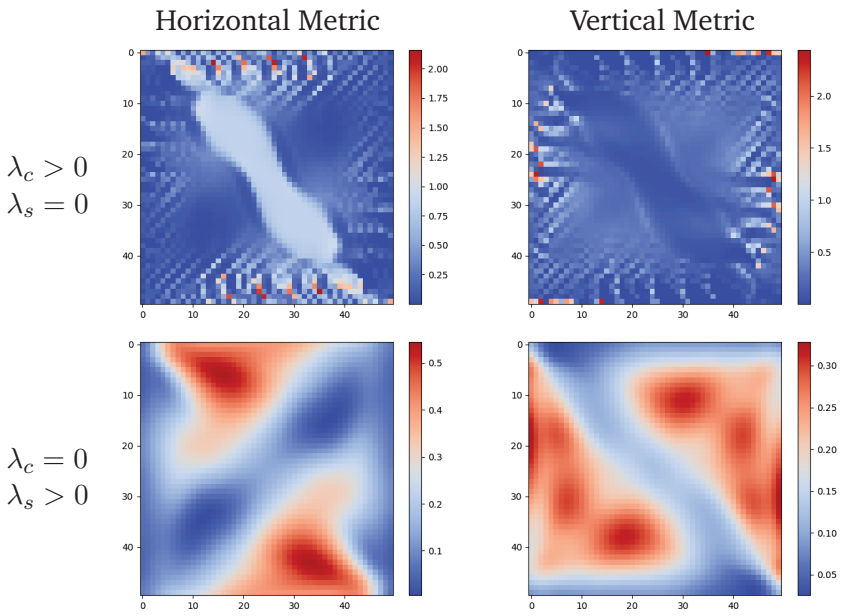


Figure 4.6: In this experiment, we show the effects of each regularizer (f_c and f_s) on the metric, using the second experiment presented in Figure 4.5. f_c constrains the weights to be close to 1, while f_s constrains them to be spatially smooth. When $\lambda_c = 0$ and $\lambda_s = 0$, a few metric weights diverge to infinity, leading to numerical errors. When $\lambda_c > 0$ and $\lambda_s > 0$, the result generally does not differ significantly from the case when $\lambda_c = 0$.

f_s also stabilizes the values and that we can set $\lambda_c = 0$ in most cases. Finally, tuning the λ_s parameter allows the user to specify the desired smoothing scale (max spatial frequency) in the final metric.

Initialization. Since the problem we are addressing is non-convex, the initialization of the metric weights is expected to have non negligible effects on the final result. In Figure 4.7, we present the end metric of the second experiment in Figure 4.5 with $\lambda_s = 0.3$, and for 3 different initializations: (1) constant initialization to 1, (2) random initialization in [0.3,3] uniformly in log scale, and (3) random initialization in [0.1,10] uniformly in log scale. We observe that the level of noise in (2) does not change the result significantly, but the one in (3) does. In (2), the initial noise did not impact the final result, because it has been smoothed out by the regularization. We conclude that the algorithm allows for some noise in the initialization, but too much noise can not be smoothed out by the regularizer, and impacts the reconstruction and the final metric significantly.

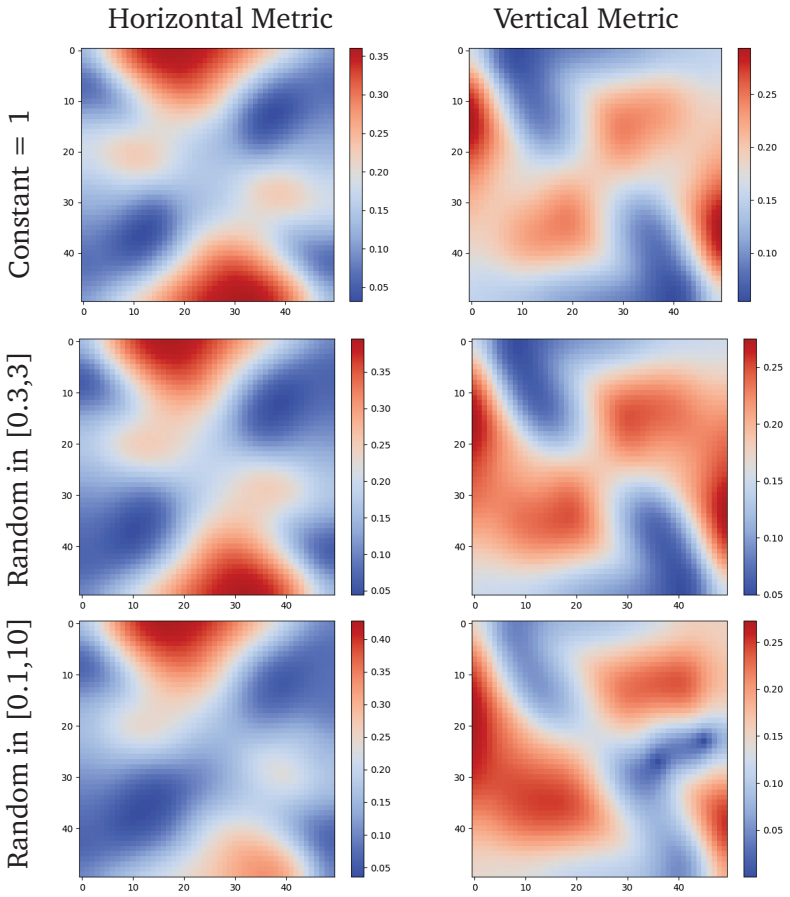


Figure 4.7: We present the final metric of the second experiment in Figure 4.5 for 3 different initializations: constant with weights equal to 1, random in log space in [0.3,3], and random in log space in [0.1,10]. The algorithm is robust to a half order of magnitude in the metric weights, but not to a full one.

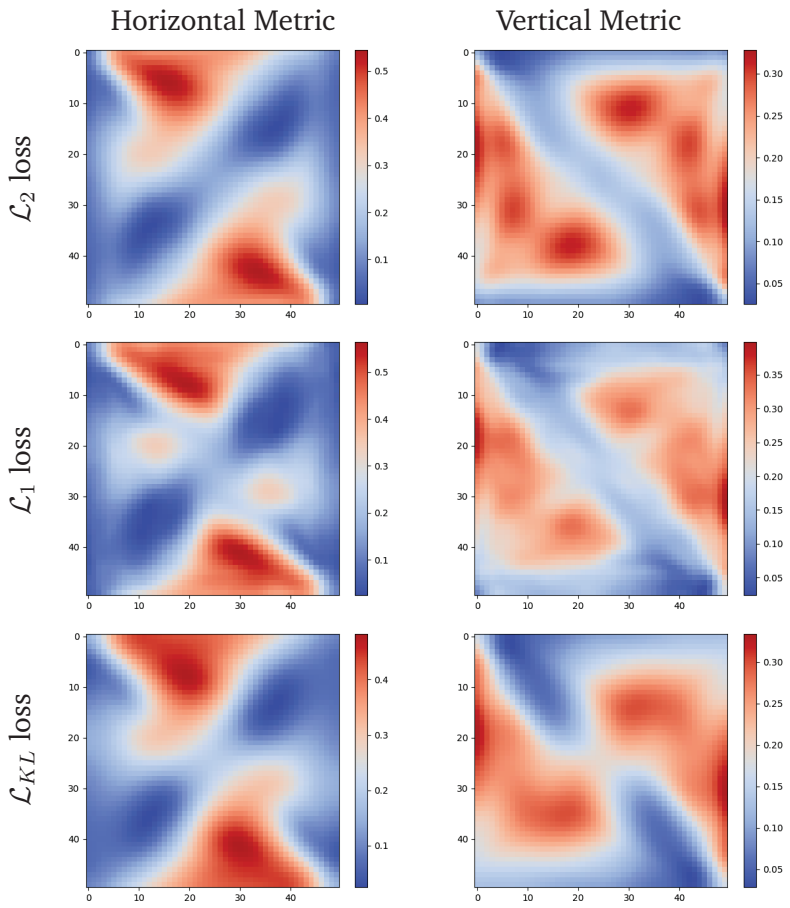


Figure 4.8: The loss function $\mathcal{L}$ influences the resulting metric. We present the metric learned during three experiments using the same parameters, but three different loss functions $\mathcal{L}_2$, $\mathcal{L}_1$ and $\mathcal{L}_{KL}$. The experiment is the second one described in Figure 4.5. One must choose the loss function depending on the application.

Loss function. The choice of the loss function $\mathcal{L}$ is left to the user, depending on what works best with their application. In Figure 4.8, we show three 2-D metrics learned on the second synthetic experiment of Figure 4.5, using the different loss functions (4.30),(4.31) and (4.32)

Diffusion equation The parameters ε and S need to be carefully set for solving the diffusion equation. Indeed, depending on their value, the formula (4.11) yields a kernel that is a better or worse approximation of the heat kernel, which directly impacts the accuracy of the displacement interpolations computed with it. We demonstrate these effects in 2-D, by interpolating between two Dirac masses across a 50^2 image. We plot the middle slice of the 2-D image as a 1-D function, for easier visualization. In Figure 4.9, we plot 10 steps of an interpolation in each subplot,

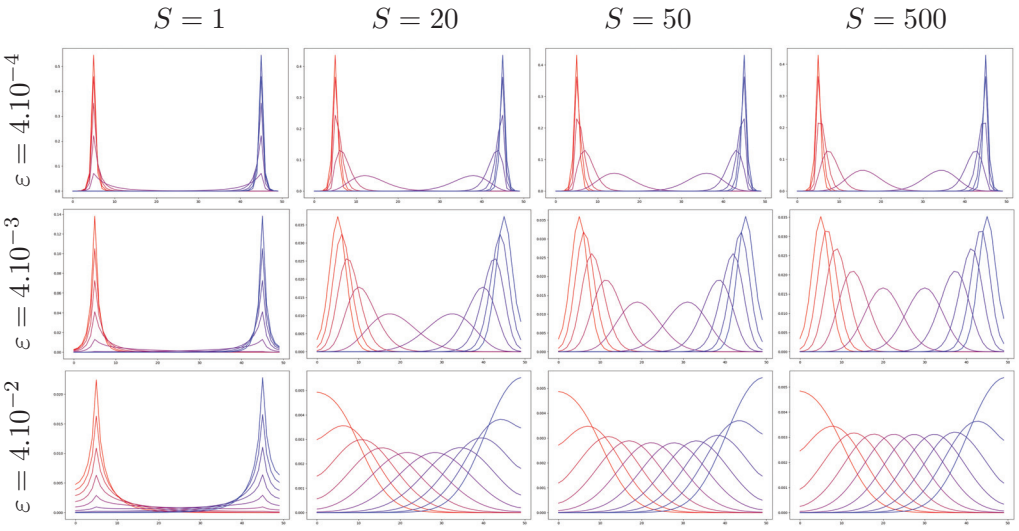


Figure 4.9: Influence of parameters ε (diffusion time) and S (number of time discretization sub-steps) on displacement interpolation, which are computed with 50 Sinkhorn iterations (sufficient for convergence in this case). Each plotted line is the 1-D middle slice of a 2-D image. We notice that there is a trade-off between the smoothness of interpolation, and the spacing equality between interpolants. An equal spacing translates a constant speed interpolation.

for different values of ε and S, with a Euclidean metric (all metric weights equal to 1).

The kernel we approximate is of the form $\exp(-d^p(x, y)/\varepsilon)$. Ideally, we would like to approximate a kernel that has $p = 2$. However, when the number of sub-steps S is small, the obtained kernel approximates one with $p = 1$, which leads to a bad interpolation where interpolants are not equally spaced (the interpolation is not of constant speed). When increasing S, it seems that the power p goes from 1 to 2, explaining the improvement of the interpolation.

The value of ε also impacts the kernel: the smaller it is, the worse the kernel approximation becomes. We thus observe a trade-off between having sharp interpolations, and having evenly spaced interpolants. It is important to note that memory footprint grows almost linearly with S (see next paragraph), since every intermediate vector in (4.11) is stored for the backward pass. In practice, we use either $\varepsilon = 4.10^{-2}$ and $S = 20$, or $\varepsilon = 1.2.10^{-3}$ and $S = 50$. With this level of smoothing, we set the number of Sinkhorn iterations to 50, which is generally enough for the Sinkhorn algorithm to converge.

Timing and memory In Table 4.1, we give the time and memory requirements of the proposed algorithm, depending on the problem size $N = n^d$ and S the number

d	n	N	S	$t_{500}(h)$	$Mem.(GB)$	$Threads$
2	50	2500	20	1.6	1.3	1
2	50	2500	100	7	4.7	1
2	100	10000	20	13	4	1
2	100	10000	100	60	16	1
3	16	4096	20	9	1.7	1
3	16	4096	50	25	3.3	1
3	16	4096	100	46	6.2	1
3	32	32768	20	110	10.9	8

Table 4.1: Time and memory requirements of the metric learning algorithm, with respect to problem size $N = n^d$ and S the number of sub-steps for solving the diffusion equation. "t_{500}" is the time taken to run 500 iterations of L-BFGS. "Mem." is the maximum resident memory (in GB) that the algorithm requires, and "Threads" is the number of threads it runs on.

of sub-steps to solve the diffusion equation. The entropic regularization factor ε (which is used here as a diffusion time) does not affect the runtime, if we keep the number of Sinkhorn iterations constant. However, if one requires a fixed precision in the computation of the transport, one needs to increase the number of Sinkhorn iterations when decreasing ε, as described in 2.1.2. We give the timings for 500 L-BFGS iterations, which in our use cases, was generally sufficient for the algorithm to converge.

This algorithm is difficult to parallelize because we need to solve a very large number of medium-size linear systems, which individually do not benefit from multi-threading. Giving more than one thread to the algorithm was only faster for $N = 32^3$. If instead we parallelize over input images (we generally have around 10), the memory footprint grows 10 times, which would become prohibitive very quickly.

4.3.4 Learning color evolutions

We now demonstrate an application of our algorithm that deals with color histograms in the RGB color space. Such histograms can be represented as discrete measures on a particular graph: a 3-D grid of size $N = n \times n \times n$ with a 6-connectivity. An important question in imaging and learning is which color space to use. The RGB space is simple to use, but variations in that space do not reflect variations of color perceived by the human eye. Other spaces, such as L*a*b* or L*u*v* have been designed to counteract this, and match variations in perception and space. Learning a ground color metric is a way to automatically fit the color space to the problem under

consideration. Note that the problem of color metric learning in psychophysics has a long history, starting with the idea of MacAdam's ellipses [Mac42], which introduces a Riemannian metric (corresponding to ellipses) to fit perceptual thresholds.

Given an input sequence of sunset images (Figure 4.10), we compute each input's color histogram, and use our algorithm to learn the metric for which the histogram sequence resembles an optimal transport of mass. Figure 4.11 shows the reconstruction of the input histogram sequence, at the end of the metric learning process. The final goal is to create a new sunset sequence from a pair of day/night images, by interpolating between them using the learned metric, and transferring the interpolated histograms onto the day image. Once a target histogram is known, transferring colors can be done via regularized OT, and we refer to the method of Solomon et al. [Sol+15].

All sequences presented hereafter contain around ten frames, but we only show five of them for brevity. We first perform two validation checks. The first check consists in interpolating between the first and last frames of the input sequence (the way it is done during the learning procedure), but transferring the interpolated histograms onto the first frame. See Figure 4.12. The second check consists in learning a metric on one part of the image, interpolating with the learned metric on the other part, and transferring the interpolated histograms onto the first frame of the other part, see Figure 4.13.

We now create a new sunset sequence from a pair of new day/night images, as described earlier. The image pair is extracted from the *country1* dataset (Figure 4.14), where we take the first and the last frame. We first learn a metric on the *seldovia2* dataset (Figure 4.15), with histograms of size 16^3, the $\mathcal{L}_2$ loss, 50 Sinkhorn iterations, 500 L-BFGS iterations, an entropic regularization of $\varepsilon = 0.004$, $S = 20$ sub-steps, and a metric regularizer parameter $\lambda_s = 1$. Next, we interpolate between the day and night histograms, using the learned metric, which is upsampled to 31^3 in order to decrease color quantization errors. Finally, we transfer each interpolated color histogram on the day frame.

We show in rows two to four of Figure 4.17 that the colors obtained with that interpolation are closer to the ground truth than with a linear interpolation or an OT interpolation with a Euclidean metric. We also note in the resulting histogram sequence (Figure 4.16) that the sunset colors are obtained because the mass does not travel in straight lines, by virtue of the learned metric.

Finally, in row five of Figure 4.17, we compare our result with a direct transfer of the *seldovia2* dataset on the day image of the *country1* dataset. A direct transfer

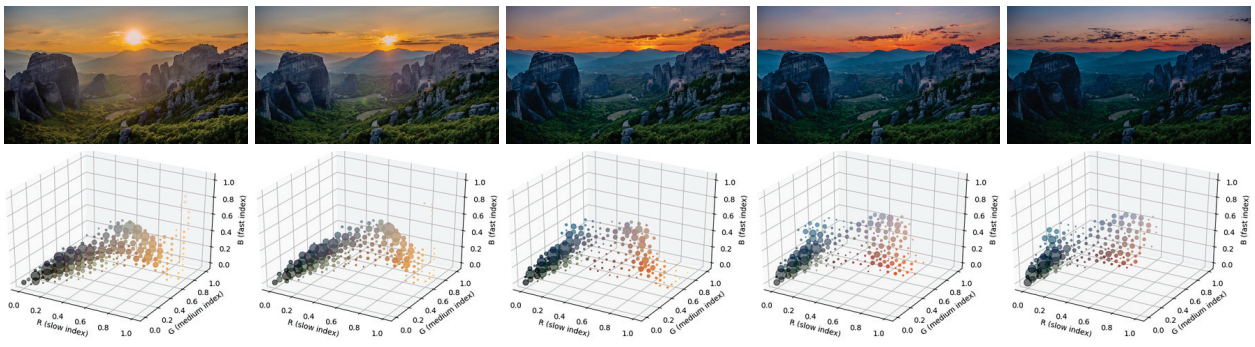


Figure 4.10: The *meteora2* dataset: images (first row) and color histograms (second row). Video courtesy of Alex Paul.

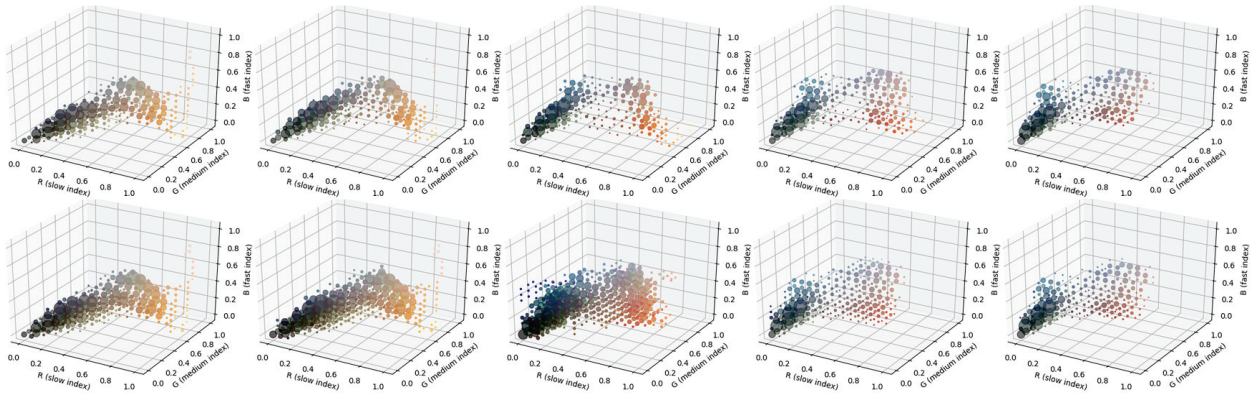


Figure 4.11: Comparison of the *meteora2* dataset (Figure 4.10) and its reconstruction using the metric learned from it, after 500 iterations of our algorithm. We notice that the reconstructions are more diffuse than the inputs, due to the entropic regularization.

also gives a plausible sunset sequence, however, the original colors of the target dataset (*country1*) are not preserved. Moreover, our method allows interpolating with an arbitrary number of frames, whereas the direct transfer can only produce the number of frames available in the source dataset.

4.4 Discussion

The problem we tackle is ill-posed since in general there is no way to find information where mass does not travel. This means that there are potentially many local minima in the energy landscape. Nevertheless, our regularization of the problem reduces the number of local minima and reduces the non-convexity by imposing spatially smooth metric weights, which also avoids over-fitting.

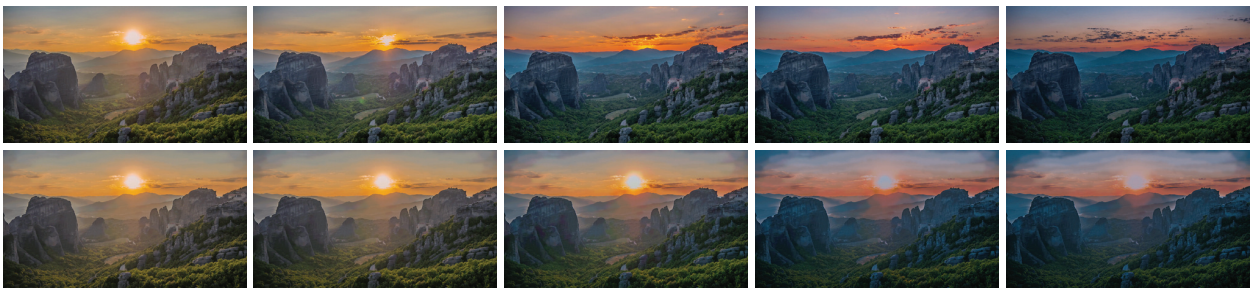


Figure 4.12: Preliminary experiment: we learn a metric on the *meteora2* dataset (top row), then reinterpolate color histograms between the first and last frames, and transfer each interpolation onto the first frame.

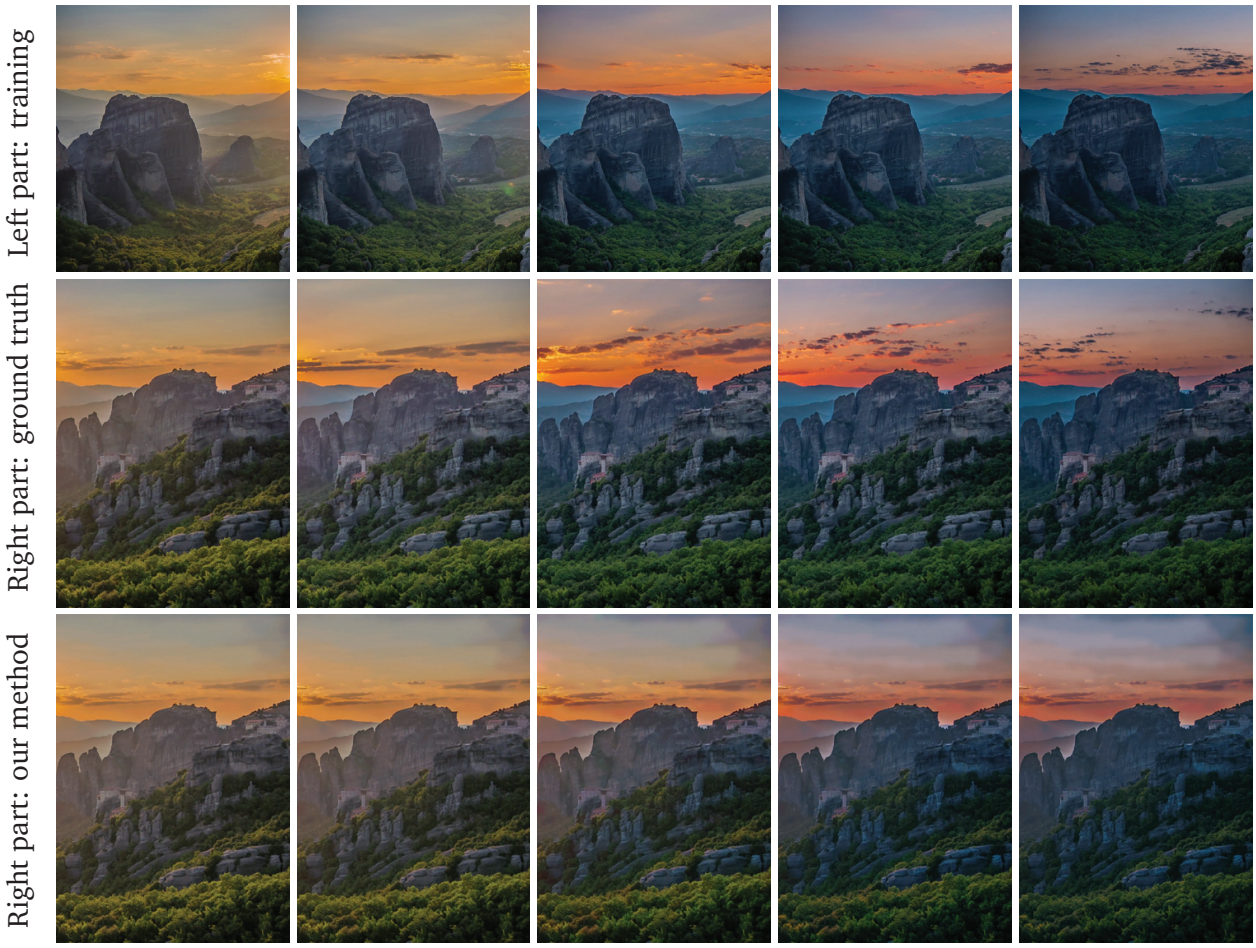


Figure 4.13: Preliminary experiment: we learn a metric on the left part of the sequence images (top row), then use that metric for interpolating between the first and last frames of the right part (middle row), and finally transfer interpolated histograms on the first frame of the right part. We recover (bottom row) colors similar to the original ones.

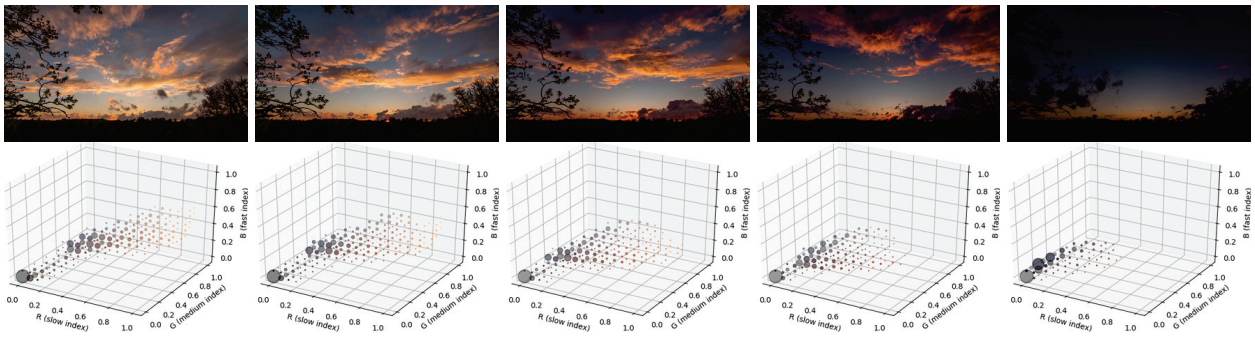


Figure 4.14: The *country1* dataset: images (first row) and color histograms (second row). Video courtesy of Quincy van den Boom

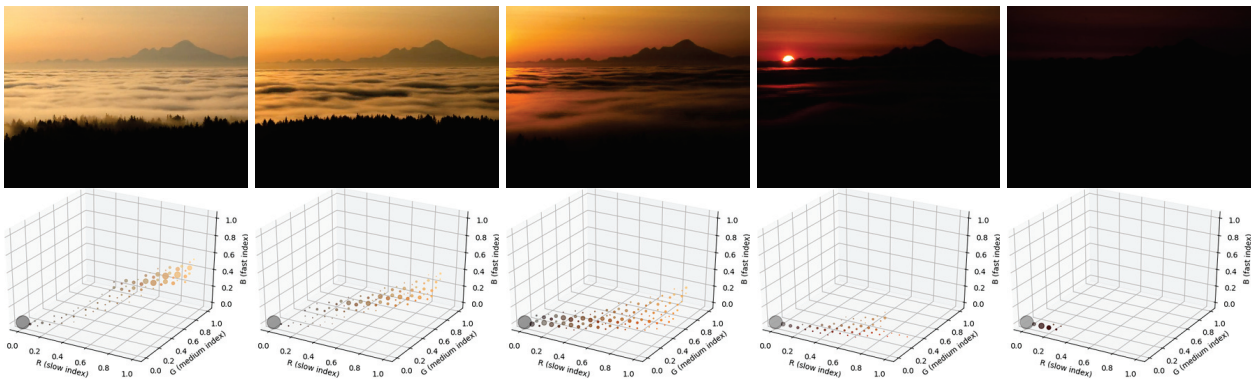


Figure 4.15: The *seldovia2* dataset: images (first row) and color histograms (second row). Video courtesy of Bretwood Higman

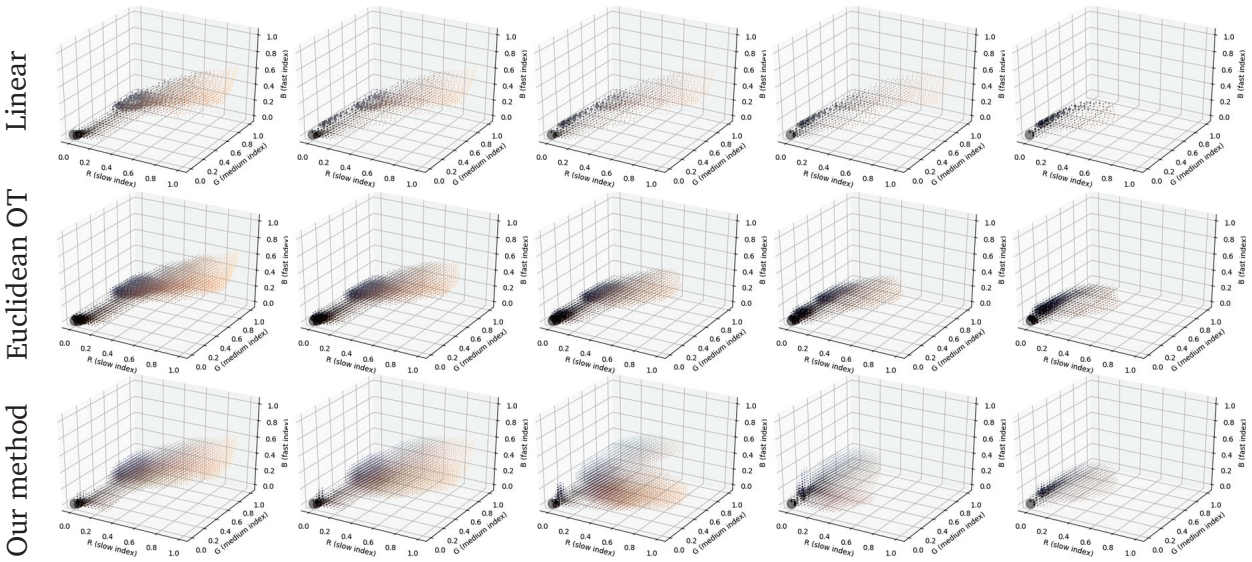


Figure 4.16: Interpolation between day and night histograms of the *country1* dataset (Figure 4.14) using: linear interpolation (1st row), OT with Euclidean metric (2nd row) and OT with the metric learned on the *seldovia2* (Figure 4.15) dataset (3rd row). Color transfers using these histograms are presented in Figure 4.17.

The fidelity of the approximation of the heat kernel (exponential of a *squared* distance) affects the quality of the displacement interpolations, as seen in Figure 4.9. This leads to a trade-off when choosing ε, between the smoothness of the interpolations, the regularity with which they are spaced out spatially, and the computational limits it involves (as S increases). Moreover, as pointed out in [Cra+13, Appendix A], low values for the parameter ε yield a distance that is closer to the graph distance (number of edges), than to the geodesic distance. This means that as ε decreases, the edge weights have less and less influence.

Although we managed to develop a tractable framework, as compared, for instance, to using a dense storage of the cost matrix, this algorithm remains computationally expensive for histograms with more than 10 000 points (see Table 4.1).

Learning different metric tensors in different areas of the domains is known in the literature as *multi-metric learning*, and in some way, our metric parameterization is close to that concept. Indeed, the weights on the graph can be seen as simple local metrics that make the space inhomogeneous. The difference between existing multi-metric learning approaches [WS09; Zha+09; RB11; Hau+12; Wan+12; Shi+14] and ours, is that they learn a few local metrics in the high-dimensional feature space, whereas we learn many of them by covering the low-dimensional ground space, and use OT to leverage them to the high-dimensional distribution space.

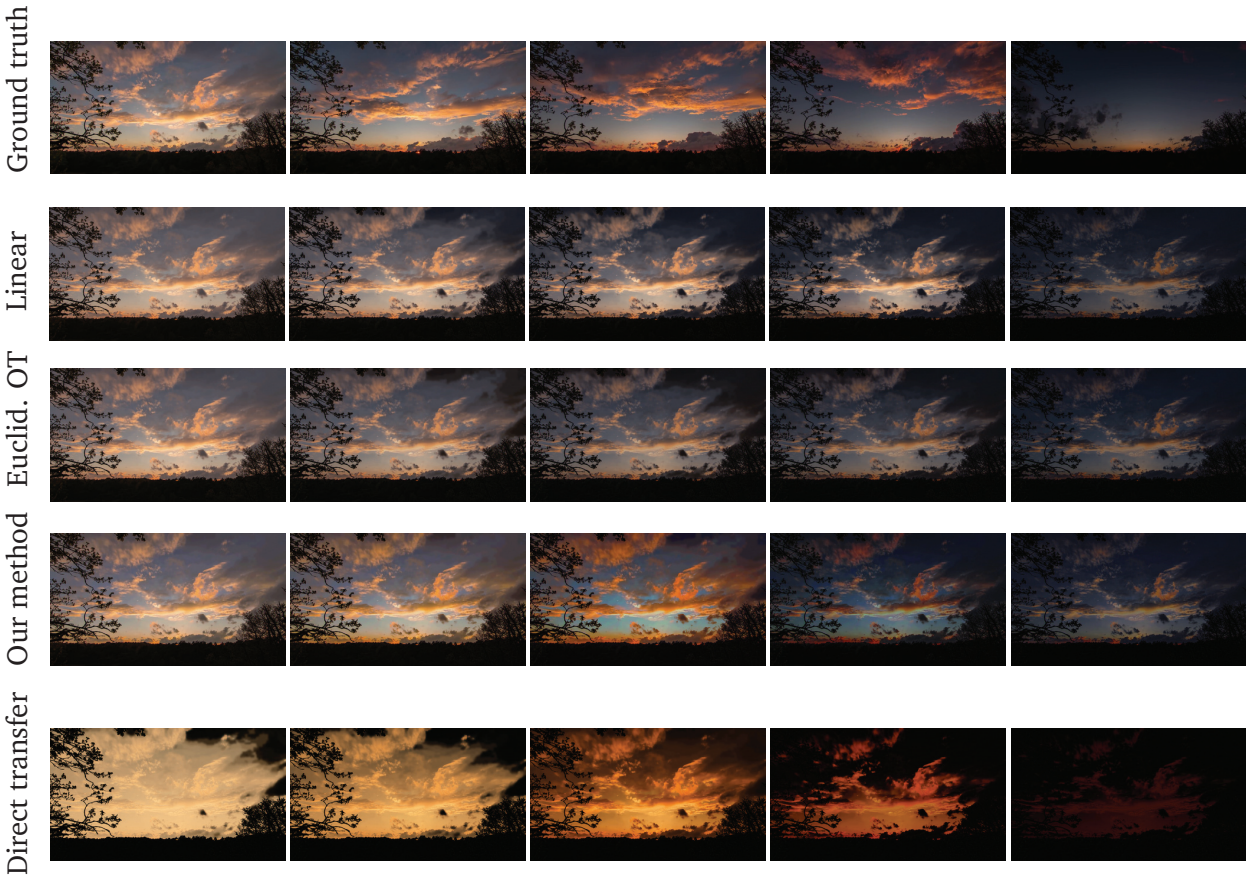


Figure 4.17: 1st row: ground truth, the *country1* dataset. Rows 2-4: color transfer of each interpolated histograms in Figure 4.16. When comparing with the ground truth, we see that our method recreates sunset-like colors, as opposed to the two other methods. Row 5: direct transfer of each frame of the *seldovia2* dataset on the first frame of the *country1* dataset. Our method is able to preserve the original colors of the day image, contrary to a direct transfer.

4.5 Conclusion

We have proposed a new method to learn the ground metric of optimal transport, as a geodesic distance on the graph supporting the data (pre-print [Hei+20]). We learn from observations of a mass displacement and aim to reconstruct them using displacement interpolations. We were able to turn a challenging task in terms of time and memory complexity into a tractable framework, using diffusion-based distance computations, regularized Wasserstein barycenters, and automatic differentiation. We demonstrated our method on toy examples, as well as on a color transfer application, where we learn the evolution of colors during a sunset, and use it to create a new sunset sequence. We finally discussed the limitations of the proposed method: our parametrization of the metric might be too simple, which limits the precision of geodesic distance approximations, which in turn impacts the interpolation, and adds a trade-off between having sharp and equally-spaced interpolations, and the computational effort it requires.

Future work. Multi-resolution strategies can be integrated into our pipeline to accelerate the linear system resolution and Sinkhorn algorithm (as in [GM17]).

For regular domains such as images and surfaces, it is possible to use a more precise approximation of a Riemannian metric as a field of tensors in place of a graph, as done for instance in [MD17], which in turn can be combined with triangulated meshes.

As we have seen in subsection 4.3.3, the power p of the ground distance for the cost function has a large influence on the displacement interpolation. It would be interesting to study more theoretically the influence of the number of sub-steps S on the quality of the heat kernel approximation, and its direct effect on displacement interpolations computed with that kernel.

In this framework, we restricted ourselves to learning from a single input sequence, but our method can be extended to take into account multiple sequences in order to learn a more robust and versatile metric.

Unbalanced optimal transport [Chi+16] could also be valuable to account for mass creation and elimination during the mass interpolation, which is important for some applications in chemistry or biology, and might be beneficial for the application on color variation.

5 Conclusion

In this dissertation, we have presented two frameworks to solve inverse problems involving OT. They are both machine learning problems applied to the Wasserstein geometry. The methods we developed are based on the entropic regularization of the transport, which offers fast iterative algorithms (Sinkhorn) to approximate OT quantities such as distances, transport plans and barycenters. Moreover, these approximated quantities are smooth with respect to their input parameters, and we applied automatic differentiation to compute their Jacobians. We focused on the development of efficient algorithms to tackle the time and memory complexity issues associated with the sizes of the distributions we considered.

The first framework we proposed is a non-linear dictionary learning method that makes full use of the Wasserstein geometry. It optimizes a dictionary and codes to represent a set of input distributions in the most faithful way possible, according to a certain fidelity criterion. The reconstruction of input distributions is carried out with Wasserstein barycenters weighted by the optimized codes. We demonstrated this method on applications such as the representation of a cardiac sequence by its key moments, and the clustering of facial expressions, and of literature works by their lexical fields.

The second framework consists in learning the ground metric of OT when it is constrained to be a geodesic distance on a graph. The metric parameters are diffusion weights on the edges of the graph, and we compute geodesic distances with a diffusion equation. The data we learn from are time series of evolving density, which we aim to model with displacement interpolations. In essence, we optimize the ground metric so that the displacement interpolation computed with it, from the first frame to the last, approximates as closely as possible the complete input series. We applied this framework to the learning and transfer of color evolutions in sunset sequences.

Inverse problems in optimal transport are generally non-linear, because of the Riemannian structure of the Wasserstein space. Additionally, they are rarely convex, which makes them difficult to solve. Regularizing the OT problem is an effective way to obtain smooth quantities that we can differentiate. In particular, the entropic

regularization yields fast algorithms that allow the resolution of medium and large-scale problems in reasonable time. Some inverse problems are more ill-posed than others and may require additional regularization to reduce the number of local minima in the energy landscape. This was the case in our ground metric learning algorithm.

5.1 Future Work

One project I worked on at the beginning of my thesis was the empirical study of barycenters outside of the simplex defined by the data, i.e. barycenters with negative weights. For example, displacement interpolations can be computed with Wasserstein barycenters with weights $[t, 1-t], t \in [0, 1]$. In a Eulerian setting, when interpolating from one histogram to another, the mass moves in straight lines if the metric is squared Euclidean, and when extrapolating outside of $[0, 1]$, the mass seems to continue along its previous trajectory. We observed that for weights in $[-1, 2]$, the computations were stable, but outside of that range, numerical instabilities appear. A study of the theoretical soundness of such weight extrapolations would be important. Being able to extrapolate a displacement interpolation could be beneficial for modeling prediction problems, as done for example with Kalman filters, which can recover posterior or anterior data such that it is coherent with the observed model of evolution.

In the Wasserstein faces application presented in 3.4.2, we noticed that the atoms recovered are generally sharper than the reconstructions, probably to counteract the blur introduced when combining them using regularized Wasserstein barycenters. This suggests a potential deconvolution ability of this algorithm. Atoms recovered using the $\mathcal{L}_1$, $\mathcal{L}_2$, or $\mathcal{L}_{KL}$ loss functions are visually distorted compared to the input images, whereas those obtained with the Wasserstein loss are closer to realistic images. Therefore, we could imagine a deconvolution problem using the regularized Wasserstein distance as the fitting loss.

The sliced Wasserstein distance briefly introduced in 2.1.5 is also promising for solving regression problems with OT, in particular for distributions with high-dimensional support. Indeed, the gradient and Hessian of that distance with respect to the mass positions are computable in closed form, and they have already been used to compute barycenters in Bonneel et al. [Bon+15]. The authors compare sliced barycenters (for Lagrangian distributions), Radon barycenters (for Eulerian distributions) and real Wasserstein barycenters of 2-D shapes, and they show that

the former two are not well suited for applications where a faithful reconstruction is sought, as they present significant artifacts. However, they demonstrate that this approach is still applicable for tasks such as texture mixing and color harmonization. This can be explained by the fact that for those applications, the final result is not the histogram itself but another quantity that is computed from it, therefore they are more robust to noise in the histogram. We can therefore imagine that the sliced paradigm would be appropriate for other such applications.

One drawback of the Eulerian discretization is that it does not scale well in higher dimension, since it requires $N = n^{\mathrm{d}}$ values per histogram. Moreover, with the curse of dimensionality, high-dimensional densities are often sparse as objects get further away from each other with increasing dimension. Therefore, a Lagrangian discretization would be more suitable for solving inverse problems involving high dimensional histograms.

OT is getting popular for natural language processing (NLP) applications, because it allows for example to compare documents semantically through their histogram of words. More complex constructions have recently been proposed, such as the hierarchical framework of Yurochkin et al. [Yur+19] based on LDA (see 2.1.6), which represents documents as histograms of topics and topics as histograms of words. Both histogram distances and topic distances are modeled as Wasserstein distances, with topic distances and word distances as respective ground metrics. This kind of architecture greatly reduces the size of histograms, which opens the door for new OT-based NLP applications on large datasets.

As the tools of optimal transport are improved and continue to spread, we are confident they will help solve many more problems, direct and inverse.

Bibliography

[Abr+17] I. Abraham, R. Abraham, M. Bergounioux, and G. Carlier. “Tomographic Reconstruction from a Few Views: A Multi-Marginal Optimal Transport Approach”. en. In: *Applied Mathematics & Optimization* 75.1 (Feb. 2017), pp. 55–73 (cit. on p. 33).

[Adl+17] Jonas Adler, Axel Ringh, Ozan Öktem, and Johan Karlsson. “Learning to Solve Inverse Problems Using Wasserstein Loss”. en. In: *arXiv:1710.10898 [cs, math]* (Oct. 2017). arXiv: 1710.10898 [cs, math] (cit. on p. 33).

[AC11] Martial Agueh and Guillaume Carlier. “Barycenters in the Wasserstein Space”. en. In: *SIAM Journal on Mathematical Analysis* 43.2 (Jan. 2011), pp. 904–924 (cit. on p. 25).

[Aha+06] M. Aharon, M. Elad, and A. Bruckstein. “K-SVD: An Algorithm for Designing Overcomplete Dictionaries for Sparse Representation”. In: *IEEE Transactions on Signal Processing* 54.11 (Nov. 2006), pp. 4311–4322 (cit. on p. 35).

[Aif+10] Niki Aifanti, Christos Papachristou, and Anastasios Delopoulos. “The MUG Facial Expression Database”. In: *Image Analysis for Multimedia Interactive Services (WIAMIS), 2010 11th International Workshop On*. IEEE, 2010, pp. 1–4 (cit. on pp. 58, 60).

[Alt+17] Jason Altschuler, Jonathan Weed, and Philippe Rigollet. “Near-Linear Time Approximation Algorithms for Optimal Transport via Sinkhorn Iteration”. In: *arXiv preprint arXiv:1705.09634* (2017). arXiv: 1705.09634 (cit. on p. 24).

[Alt+18] Jason Altschuler, Francis Bach, Alessandro Rudi, and Jonathan Weed. “Massively Scalable Sinkhorn Distances via the Nyström Method”. en. In: *arXiv:1812.05189 [cs, math, stat]* (Dec. 2018). arXiv: 1812.05189 [cs, math, stat] (cit. on pp. 24, 64).

[Arj+17] Martin Arjovsky, Soumith Chintala, and Léon Bottou. “Wasserstein GAN”. en. In: *arXiv:1701.07875 [cs, stat]* (Jan. 2017). arXiv: 1701.07875 [cs, stat] (cit. on pp. 29, 31).

[Aur+98] F. Aurenhammer, F. Hoffmann, and B. Aronov. “Minkowski-Type Theorems and Least-Squares Clustering”. en. In: *Algorithmica* 20.1 (Jan. 1998), pp. 61–76 (cit. on p. 29).

[Bas+06] Federico Bassetti, Antonella Bodini, and Eugenio Regazzini. “On Minimum Kantorovich Distance Estimators”. en. In: *Statistics & Probability Letters* 76.12 (July 2006), pp. 1298–1302 (cit. on p. 31).

[Bay+18] Atilim Gunes Baydin, Barak A. Pearlmutter, Alexey Andreyevich Radul, and Jeffrey Mark Siskind. “Automatic Differentiation in Machine Learning: A Survey”. In: *Journal of Machine Learning Research* 18.153 (2018), pp. 1–43 (cit. on p. 74).

[Bel+15] Aurélien Bellet, Amaury Habrard, and Marc Sebban. *Metric Learning*. English. Synthesis Digital Library of Engineering and Computer Science. San Rafael, California (1537 Fourth Street, San Rafael, CA 94901 USA): Morgan & Claypool, 2015 (cit. on pp. 38, 39).

[Ben03] Jean-David Benamou. “Numerical Resolution of an “Unbalanced” Mass Transport Problem”. en. In: *ESAIM: Mathematical Modelling and Numerical Analysis* 37.5 (Sept. 2003), pp. 851–868 (cit. on p. 55).

[BB00] Jean-David Benamou and Yann Brenier. “A Computational Fluid Mechanics Solution to the Monge-Kantorovich Mass Transfer Problem”. en. In: *Numerische Mathematik* 84.3 (Jan. 2000), pp. 375–393 (cit. on p. 29).

[Ben+15] Jean-David Benamou, Guillaume Carlier, Marco Cuturi, Luca Nenna, and Gabriel Peyré. “Iterative Bregman Projections for Regularized Transportation Problems”. In: *SIAM Journal on Scientific Computing* 37.2 (2015), A1111–A1138 (cit. on pp. 21, 23, 26).

[Ben+16] Jean-David Benamou, Guillaume Carlier, and Maxime Laborde. “An Augmented Lagrangian Approach to Wasserstein Gradient Flows and Applications”. en. In: *ESAIM: Proceedings and Surveys* 54 (June 2016). Ed. by Bertram Düring, Carola-Bibiane Schönlieb, and Marie-Therese Wolfram, pp. 1–17 (cit. on p. 31).

[Ben+10] Fethallah Benmansour, Guillaume Carlier, Gabriel Peyré, and Filippo Santambrogio. “Derivatives with Respect to Metrics and Applications: Subgradient Marching Algorithm”. In: *Numerische Mathematik* 116.3 (2010), pp. 357–381 (cit. on p. 41).

[Ber+17] Espen Bernton, Pierre E. Jacob, Mathieu Gerber, and Christian P. Robert. “Inference in Generative Models Using the Wasserstein Distance”. In: *arXiv preprint arXiv:1701.05146* (2017). arXiv: 1701.05146 (cit. on p. 31).

[Ber81] Dimitri P. Bertsekas. “A New Algorithm for the Assignment Problem”. en. In: *Mathematical Programming* 21.1 (Dec. 1981), pp. 152–171 (cit. on p. 27).

[Big+17] Jérémie Bigot, Raúl Gouet, Thierry Klein, and Alfredo López. “Geodesic PCA in the Wasserstein Space by Convex PCA”. EN. In: *Annales de l’Institut Henri Poincaré, Probabilités et Statistiques* 53.1 (Feb. 2017), pp. 1–26 (cit. on p. 37).

[Ble03] David M Blei. “Latent Dirichlet Allocation”. en. In: (2003), p. 30 (cit. on p. 33).

[Blo+18] Mathieu Blondel, Vivien Seguy, and Antoine Rolet. “Smooth and Sparse Optimal Transport”. en. In: *arXiv:1710.06276 [cs, stat]* (Feb. 2018). arXiv: 1710.06276 [cs, stat] (cit. on pp. 17, 30).

[Boi+15] Emmanuel Boissard, Thibaut Le Gouic, and Jean-Michel Loubes. “Distribution’s Template Estimate with Wasserstein Metrics”. en. In: *Bernoulli* 21.2 (May 2015), pp. 740–759 (cit. on p. 37).

[BC19] Nicolas Bonneel and David Coeurjolly. “SPOT: Sliced Partial Optimal Transport”. en. In: *ACM Transactions on Graphics* 38.4 (July 2019), pp. 1–13 (cit. on pp. 29, 30).

[Bon+11] Nicolas Bonneel, Michiel Van De Panne, Sylvain Paris, and Wolfgang Heidrich. “Displacement Interpolation Using Lagrangian Mass Transport”. In: *ACM Transactions on Graphics (TOG)*. Vol. 30. ACM, 2011, p. 158 (cit. on pp. 16, 21, 30).

[Bon+15] Nicolas Bonneel, Julien Rabin, Gabriel Peyré, and Hanspeter Pfister. “Sliced and Radon Wasserstein Barycenters of Measures”. en. In: *Journal of Mathematical Imaging and Vision* 51.1 (Jan. 2015), pp. 22–45 (cit. on pp. 16, 29–32, 94).

[Bon+16] Nicolas Bonneel, Gabriel Peyré, and Marco Cuturi. “Wasserstein Barycentric Coordinates: Histogram Regression Using Optimal Transport”. en. In: *ACM Transactions on Graphics* 35.4 (July 2016), pp. 1–10 (cit. on pp. 30–32, 34, 45, 46, 63, 66).

[Bou+17] Olivier Bousquet, Sylvain Gelly, Ilya Tolstikhin, Carl-Johann Simon-Gabriel, and Bernhard Schoelkopf. “From Optimal Transport to Generative Modeling: The VEGAN Cookbook”. en. In: *arXiv:1705.07642 [stat]* (May 2017). arXiv: 1705.07642 [stat] (cit. on p. 29).

[Bre67] L M Bregman. “The Relaxation Method of Finding the Common Point of Convex Sets and Its Application to the Solution of Problems in Convex Programming”. en. In: (1967), p. 18 (cit. on p. 18).

[Bre+03] Y. Brenier, U. Frisch, M. Hénon, et al. “Reconstruction of the Early Universe as a Convex Optimization Problem: Early Universe as a Convex Optimization Problem”. en. In: *Monthly Notices of the Royal Astronomical Society* 346.2 (Dec. 2003), pp. 501–524 (cit. on p. 29).

[Bre91] Yann Brenier. “Polar Factorization and Monotone Rearrangement of Vector-Valued Functions”. en. In: *Communications on Pure and Applied Mathematics* 44.4 (June 1991), pp. 375–417 (cit. on p. 13).

[Bri+08] Justin Brickell, Inderjit S. Dhillon, Suvrit Sra, and Joel A. Tropp. “The Metric Nearness Problem”. en. In: *SIAM Journal on Matrix Analysis and Applications* 30.1 (Jan. 2008), pp. 375–396 (cit. on p. 40).

[Bur+12] M. Burger, M. Franek, and C.-B. Schonlieb. “Regularized Regression and Density Estimation Based on Optimal Transport”. en. In: *Applied Mathematics Research eXpress* (Mar. 2012) (cit. on p. 33).

[But+04] G. Buttazzo, A. Davini, I. Fragalà, and F. Macià. “Optimal Riemannian Distances Preventing Mass Transfer”. en. In: *Journal für die reine und angewandte Mathematik (Crelles Journal)* 2004.575 (Jan. 2004) (cit. on p. 41).

[Car+15] Guillaume Carlier, Adam Oberman, and Edouard Oudet. “Numerical Methods for Matching for Teams and Wasserstein Barycenters”. In: *ESAIM: Mathematical Modelling and Numerical Analysis* 49.6 (2015), pp. 1621–1642 (cit. on p. 27).

[Car+17] Mathieu Carrière, Marco Cuturi, and Steve Oudot. “Sliced Wasserstein Kernel for Persistence Diagrams”. en. In: *arXiv:1706.03358 [cs, math, stat]* (Nov. 2017). arXiv: 1706.03358 [cs, math, stat] (cit. on p. 29).

[Caz+17] Elsa Cazelles, Vivien Seguy, Jérémie Bigot, Marco Cuturi, and Nicolas Papadakis. “Log-PCA versus Geodesic PCA of Histograms in the Wasserstein Space”. In: *arXiv preprint arXiv:1708.08143* (2017). arXiv: 1708.08143 (cit. on p. 37).

[CK18] Deeparnab Chakrabarty and Sanjeev Khanna. “Better and Simpler Error Analysis of the Sinkhorn-Knopp Algorithm for Matrix Scaling”. en. In: *arXiv:1801.02790 [cs]* (Feb. 2018). arXiv: 1801.02790 [cs] (cit. on p. 24).

[Che+09] Gal Chechik, Uri Shalit, Varun Sharma, and Samy Bengio. “An Online Algorithm for Large Scale Image Similarity Learning”. en. In: *Advances in Neural Information Processing Systems* (2009), p. 9 (cit. on p. 39).

[Chi+16] Lenaic Chizat, Gabriel Peyré, Bernhard Schmitzer, and François-Xavier Vialard. “Scaling Algorithms for Unbalanced Transport Problems”. In: *arXiv preprint arXiv:1607.05816* (2016). arXiv: 1607.05816 (cit. on pp. 50, 51, 55, 92).

[Cho+05] S. Chopra, R. Hadsell, and Y. LeCun. “Learning a Similarity Metric Discriminatively, with Application to Face Verification”. en. In: *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR’05)*. Vol. 1. San Diego, CA, USA: IEEE, 2005, pp. 539–546 (cit. on p. 39).

[Cla+18] Sebastian Claici, Edward Chien, and Justin Solomon. “Stochastic Wasserstein Barycenters”. en. In: *arXiv:1802.05757 [cs, math, stat]* (Feb. 2018). arXiv: 1802.05757 [cs, math, stat] (cit. on pp. 26, 27).

[Cou+14] Nicolas Courty, Rémi Flamary, and Devis Tuia. “Domain Adaptation with Regularized Optimal Transport”. en. In: *Machine Learning and Knowledge Discovery in Databases*. Vol. 8724. Berlin, Heidelberg: Springer Berlin Heidelberg, 2014, pp. 274–289 (cit. on p. 31).

[Cou+16] Nicolas Courty, Remi Flamary, Devis Tuia, and Alain Rakotomamonjy. “Optimal Transport for Domain Adaptation”. en. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 39.9 (2016), pp. 1853–1865 (cit. on p. 31).

[Cou+17] Nicolas Courty, Rémi Flamary, and Mélanie Ducoffe. “Learning Wasserstein Embeddings”. In: *arXiv:1710.07457 [cs, stat]* (Oct. 2017). arXiv: 1710.07457 [cs, stat] (cit. on pp. 16, 31).

[Cra+13] Keenan Crane, Clarisse Weischedel, and Max Wardetzky. “Geodesics in Heat: A New Approach to Computing Distance Based on Heat Flow”. In: *ACM Transactions on Graphics (TOG)* 32.5 (2013), p. 152 (cit. on p. 90).

[Cut13] Marco Cuturi. “Sinkhorn Distances: Lightspeed Computation of Optimal Transport”. In: *Advances in Neural Information Processing Systems*. 2013, pp. 2292–2300 (cit. on pp. 16, 21, 31).

[CA14] Marco Cuturi and David Avis. “Ground Metric Learning”. In: *Journal of Machine Learning Research* 15.1 (2014), pp. 533–564 (cit. on pp. 39–41).

[CD14] Marco Cuturi and Arnaud Doucet. “Fast Computation of Wasserstein Barycenters”. In: *International Conference on Machine Learning*. 2014, pp. 685–693 (cit. on pp. 26, 46).

[CP15] Marco Cuturi and Gabriel Peyré. “A Smoothed Dual Approach for Variational Wasserstein Problems”. en. In: *arXiv:1503.02533 [math, stat]* (Mar. 2015). arXiv: 1503.02533 [math, stat] (cit. on pp. 27, 31).

[CP18] Marco Cuturi and Gabriel Peyré. “Semi-Dual Regularized Optimal Transport”. en. In: *SIAM Review* 60.4 (Jan. 2018), pp. 941–965. arXiv: 1811.05527 (cit. on p. 27).

[D’a+05] Alexandre D’aspremont, Laurent E Ghaoui, Michael I Jordan, and Gert R Lanckriet. “A Direct Formulation for Sparse PCA Using Semidefinite Programming”. en. In: (2005), p. 8 (cit. on p. 35).

[de +12] Fernando de Goes, Katherine Breeden, Victor Ostromoukhov, and Mathieu Desbrun. “Blue Noise through Optimal Transport”. en. In: *ACM Transactions on Graphics* 31.6 (Nov. 2012), p. 1 (cit. on p. 32).

[Del04] Julie Delon. “Midway Image Equalization”. en. In: *Journal of Mathematical Imaging and Vision* 21.2 (Sept. 2004), pp. 119–134 (cit. on p. 30).

[Del+10] Julie Delon, Julien Salomon, and Andrei Sobolevskii. “Fast Transport Optimization for Monge Costs on the Circle”. en. In: *SIAM Journal on Applied Mathematics* 70.7 (Jan. 2010), pp. 2239–2258. arXiv: 0902.3527 (cit. on p. 29).

[Des+18a] Ishan Deshpande, Ziyu Zhang, and Alexander Schwing. “Generative Modeling Using the Sliced Wasserstein Distance”. en. In: *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*. Salt Lake City, UT: IEEE, June 2018, pp. 3483–3491 (cit. on pp. 29, 31).

[Des+18b] Arnaud Dessein, Nicolas Papadakis, and Jean-Luc Rouas. “Regularized Optimal Transport and the Rot Mover’s Distance”. en. In: (2018), p. 53 (cit. on p. 17).

[Dog+19] Pierre Dognin, Igor Melnyk, Youssef Mroueh, et al. “Wasserstein Barycenter Model Ensembling”. en. In: *arXiv:1902.04999 [cs, stat]* (Feb. 2019). arXiv: 1902.04999 [cs, stat] (cit. on p. 34).

[Dup+16] Arnaud Dupuy, Alfred Galichon, and Yifei Sun. “Estimating Matching Affinity Matrix under Low-Rank Constraints”. en. In: *arXiv:1612.09585 [stat]* (Dec. 2016). arXiv: 1612.09585 [stat] (cit. on pp. 18, 40, 41).

[Dvu+18] Pavel Dvurechensky, Alexander Gasnikov, and Alexey Kroshnin. “Computational Optimal Transport: Complexity by Accelerated Gradient Descent Is Better Than by Sinkhorn’s Algorithm”. en. In: *arXiv:1802.04367 [cs, math]* (Feb. 2018). arXiv: 1802.04367 [cs, math] (cit. on pp. 17, 24).

[Eng+16] Bjorn Engquist, Brittany D. Froese, and Yunan Yang. “Optimal Transport for Seismic Full Waveform Inversion”. en. In: *arXiv:1602.01540 [physics]* (Apr. 2016). arXiv: 1602.01540 [physics] (cit. on p. 33).

[ES18] Montacer Essid and Justin Solomon. “Quadratically-Regularized Optimal Transport on Graphs”. en. In: *arXiv:1704.08200 [cs, math]* (Mar. 2018). arXiv: 1704.08200 [cs, math] (cit. on p. 17).

[Fer+13] Sira Ferradans, Nicolas Papadakis, Gabriel Peyré, and Jean-François Aujol. “Regularized Discrete Optimal Transport”. en. In: *Scale Space and Variational Methods in Computer Vision 2013* (July 2013). arXiv: 1307.5551 (cit. on pp. 17, 30).

[Fey+17] Jean Feydy, Benjamin Charlier, François-Xavier Vialard, and Gabriel Peyré. “Optimal Transport for Diffeomorphic Registration”. en. In: *arXiv:1706.05218 [math]* (June 2017). arXiv: 1706.05218 [math] (cit. on p. 31).

[Fey+19] Jean Feydy, Pierre Roussillon, Alain Trouvé, and Pietro Gori. “Fast and Scalable Optimal Transport for Brain Tractograms”. en. In: (2019), p. 10 (cit. on p. 18).

[Fis36] R. A. Fisher. “The Use of Multiple Measurements in Taxonomic Problems”. en. In: *Annals of Eugenics* 7.2 (Sept. 1936), pp. 179–188 (cit. on p. 40).

[Fla+16] Rémi Flamary, Cédric Févotte, Nicolas Courty, and Valentin Emiya. “Optimal Spectral Transportation with Application to Music Transcription”. en. In: (2016), p. 9 (cit. on p. 37).

[Fla+18] Rémi Flamary, Marco Cuturi, Nicolas Courty, and Alain Rakotomamonjy. “Wasserstein Discriminant Analysis”. en. In: *Machine Learning* 107.12 (Dec. 2018), pp. 1923–1945 (cit. on pp. 31, 40).

[Fle+04] P.T. Fletcher, C. Lu, S.M. Pizer, and S. Joshi. “Principal Geodesic Analysis for the Study of Nonlinear Statistics of Shape”. en. In: *IEEE Transactions on Medical Imaging* 23.8 (Aug. 2004), pp. 995–1005 (cit. on p. 36).

[Fro+15] Charlie Frogner, Chiyuan Zhang, Hossein Mobahi, Mauricio Araya, and Tomaso A Poggio. “Learning with a Wasserstein Loss”. en. In: *Advances in Neural Information Processing Systems* (2015), p. 9 (cit. on p. 33).

[GM18] Thomas O. Gallouët and Quentin Mérigot. “A Lagrangian Scheme à La Brenier for the Incompressible Euler Equations”. en. In: *Foundations of Computational Mathematics* 18.4 (Aug. 2018), pp. 835–865 (cit. on pp. 29, 31).

[Gan+13] Mehrdad J. Gangeh, Ali Ghodsi, and Mohamed S. Kamel. “Kernelized Supervised Dictionary Learning”. en. In: *IEEE Transactions on Signal Processing* 61.19 (Oct. 2013), pp. 4753–4767. arXiv: 1207.2488 (cit. on p. 36).

[Gen+16] Aude Genevay, Marco Cuturi, Gabriel Peyré, and Francis Bach. “Stochastic Optimization for Large-Scale Optimal Transport”. en. In: (2016), p. 9 (cit. on p. 29).

[Gen+17] Aude Genevay, Gabriel Peyré, and Marco Cuturi. “Learning Generative Models with Sinkhorn Divergences”. en. In: *arXiv:1706.00292 [stat]* (June 2017). arXiv: 1706.00292 [stat] (cit. on pp. 29, 31, 46, 66).

[GM17] Samuel Gerber and Mauro Maggioni. “Multiscale Strategies for Computing Optimal Transport”. In: *arXiv preprint arXiv:1708.02469* (2017). arXiv: 1708.02469 (cit. on pp. 31, 64, 92).

[GW08] A. Griewank and A. Walther. *Evaluating Derivatives*. Other Titles in Applied Mathematics. Society for Industrial and Applied Mathematics, Jan. 2008 (cit. on pp. 43, 74).

[Gri12] Andreas Griewank. "Who Invented the Reverse Mode of Differentiation?" en. In: *Documenta Mathematica* (2012), p. 12 (cit. on p. 74).

[Hak+04] Steven Haker, Lei Zhu, Allen Tannenbaum, and Sigurd Angenent. "Optimal Mass Transport for Registration and Warping". en. In: *International Journal of Computer Vision* 60.3 (Dec. 2004), pp. 225–240 (cit. on p. 31).

[Har+11] Mehrtash T. Harandi, Conrad Sanderson, Sareh Shirazi, and Brian C. Lovell. "Graph Embedding Discriminant Analysis on Grassmannian Manifolds for Improved Image Set Matching". en. In: *CVPR 2011*. Colorado Springs, CO, USA: IEEE, June 2011, pp. 2705–2712 (cit. on p. 36).

[Hau+12] Søren Hauberg, Oren Freifeld, and Michael J Black. "A Geometric Take on Metric Learning". en. In: (2012), p. 9 (cit. on p. 90).

[Hei+20] Matthieu Heitz, Nicolas Bonneel, David Coeurjolly, Marco Cuturi, and Gabriel Peyré. "Ground Metric Learning on Graphs". en. In: *arXiv:1911.03117 [cs, math, stat]* (Oct. 2020). arXiv: 1911.03117 [cs, math, stat] (cit. on p. 92).

[Hit41] Frank L. Hitchcock. "The Distribution of a Product from Several Sources to Numerous Localities". en. In: *Journal of Mathematics and Physics* 20.1-4 (1941), pp. 224–230 (cit. on p. 16).

[Hua+16] Gao Huang, Chuan Guo, Matt J. Kusner, et al. "Supervised Word Mover's Distance". In: *Advances in Neural Information Processing Systems*. 2016, pp. 4862–4870 (cit. on pp. 33, 40, 41).

[HO00] A. Hyvärinen and E. Oja. "Independent Component Analysis: Algorithms and Applications". en. In: *Neural Networks* 13.4-5 (June 2000), pp. 411–430 (cit. on p. 35).

[IP11] Piotr Indyk and Eric Price. "K-Median Clustering, Model-Based Compressive Sensing, and Sparse Recovery for Earth Mover Distance". en. In: *arXiv:1104.4674 [cs, math]* (Apr. 2011). arXiv: 1104.4674 [cs, math] (cit. on p. 33).

[Jay+13] Sadeep Jayasumana, Richard Hartley, Mathieu Salzmann, Hongdong Li, and Mehrtash Harandi. "Kernel Methods on the Riemannian Manifold of Symmetric Positive Definite Matrices". en. In: *2013 IEEE Conference on Computer Vision and Pattern Recognition*. Portland, OR, USA: IEEE, June 2013, pp. 73–80 (cit. on p. 36).

[Kan42] Leonid Kantorovich. "On the Translocation of Masses (in Russian)". In: *Doklady Akademii Nauk SSSR* 37.7-8 (1942), pp. 227–229 (cit. on pp. 11, 16).

[KR17] Johan Karlsson and Axel Ringh. "Generalized Sinkhorn Iterations for Regularizing Inverse Problems Using Optimal Mass Transport". en. In: *SIAM Journal on Imaging Sciences* 10.4 (Jan. 2017), pp. 1935–1962. arXiv: 1612.02273 (cit. on p. 33).

[Ked+12] Dor Kedem, Stephen Tyree, Fei Sha, Gert R Lanckriet, and Kilian Q Weinberger. "Non-Linear Metric Learning". en. In: *Neural Information Processing Systems (NIPS)* (2012), p. 9 (cit. on p. 39).

[KK12] Z. Király and P. Kovács. "Efficient Implementations of Minimum-Cost Flow Algorithms". en. In: *arXiv:1207.6381 [cs]* (July 2012). arXiv: 1207.6381 [cs] (cit. on p. 16).

[Kir11] Andreas Kirsch. *An Introduction to the Mathematical Theory of Inverse Problems*. en. 2. ed. Applied Mathematical Sciences 120. New York, NY: Springer, 2011 (cit. on pp. 6, 7).

[Kol+16] Soheil Kolouri, Yang Zou, and Gustavo K. Rohde. "Sliced Wasserstein Kernels for Probability Distributions". en. In: *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. Las Vegas, NV, USA: IEEE, June 2016, pp. 5258–5267 (cit. on p. 29).

[KY94] J.J. Kosowsky and A.L. Yuille. "The Invisible Hand Algorithm: Solving the Assignment Problem with Statistical Physics". en. In: *Neural Networks* 7.3 (Jan. 1994), pp. 477–490 (cit. on p. 18).

[KT17] Max Kuang and Esteban G. Tabak. "Preconditioning of Optimal Transport". en. In: *SIAM Journal on Scientific Computing* 39.4 (Jan. 2017), A1793–A1810 (cit. on p. 30).

[Kuh55] Harold W. Kuhn. "The Hungarian Method for the Assignment Problem". In: *Naval research logistics quarterly* 2.1-2 (1955), pp. 83–97 (cit. on p. 27).

[Kul13] Brian Kulis. "Metric Learning: A Survey". en. In: *Foundations and Trends® in Machine Learning* 5.4 (2013), pp. 287–364 (cit. on p. 38).

[Kus+15] Matt J Kusner, Yu Sun, Nicholas I Kolkin, and Kilian Q Weinberger. "From Word Embeddings To Document Distances". en. In: (2015), p. 10 (cit. on pp. 33, 40).

[Lav+18] Hugo Lavenant, Sebastian Claici, Edward Chien, and Justin Solomon. "Dynamical Optimal Transport on Discrete Surfaces". en. In: *ACM Transactions on Graphics* 37.6 (Dec. 2018), pp. 1–16. arXiv: 1809.07083 (cit. on pp. 30, 32).

[LS99] Daniel D. Lee and H. Sebastian Seung. "Learning the Parts of Objects by Non-Negative Matrix Factorization". en. In: *Nature* 401.6755 (Oct. 1999), pp. 788–791 (cit. on p. 35).

[Lee+07] Honglak Lee, Alexis Battle, Rajat Raina, and Andrew Y Ng. "Efficient Sparse Coding Algorithms". en. In: (2007), p. 8 (cit. on p. 35).

[Lee+09] Hyekyoung Lee, Andrzej Cichocki, and Seungjin Choi. "Kernel Nonnegative Matrix Factorization for Spectral EEG Feature Extraction". en. In: *Neurocomputing* 72.13-15 (Aug. 2009), pp. 3182–3190 (cit. on p. 36).

[Lel+14] Jan Lellmann, Dirk A. Lorenz, Carola Schönlieb, and Tuomo Valkonen. "Imaging with Kantorovich-Rubinstein Discrepancy". en. In: *arXiv:1407.0221 [cs, math]* (July 2014). arXiv: 1407.0221 [cs, math] (cit. on p. 31).

[Lév15] Bruno Lévy. “A Numerical Algorithm for L2 Semi-Discrete Optimal Transport in 3D”. en. In: *ESAIM: Mathematical Modelling and Numerical Analysis* 49.6 (Nov. 2015), pp. 1693–1715 (cit. on pp. 29, 30).

[LS17] Bruno Lévy and Erica Schwindt. “Notions of Optimal Transport Theory and How to Implement Them on a Computer”. en. In: *arXiv:1710.02634 [math]* (Oct. 2017), pp. 135–148. arXiv: 1710.02634 [math] (cit. on p. 32).

[LO07] Haibin Ling and Kazunori Okada. “An Efficient Earth Mover’s Distance Algorithm for Robust Histogram Comparison”. en. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 29.5 (May 2007), pp. 840–853 (cit. on p. 29).

[Lui+18] Giulia Luise, Alessandro Rudi, Massimiliano Pontil, and Carlo Ciliberto. “Differential Properties of Sinkhorn Approximation for Learning with Wasserstein Distance”. en. In: (2018), p. 12 (cit. on p. 27).

[Mac42] David L. MacAdam. “Visual Sensitivities to Color Differences in Daylight”. en. In: *Journal of the Optical Society of America* 32.5 (May 1942), p. 247 (cit. on pp. 73, 86).

[Mai+09] Julien Mairal, Jean Ponce, Guillermo Sapiro, Andrew Zisserman, and Francis R Bach. “Supervised Dictionary Learning”. en. In: (2009), p. 8 (cit. on p. 36).

[Mai+10] Julien Mairal, Francis Bach, Jean Ponce, and Guillermo Sapiro. “Online Learning for Matrix Factorization and Sparse Coding”. en. In: *Journal of Machine Learning Research* 11.Jan (2010), pp. 19–60 (cit. on p. 36).

[McC97] Robert J. McCann. “A Convexity Principle for Interacting Gases”. en. In: *Advances in Mathematics* 128.1 (June 1997), pp. 153–179 (cit. on p. 25).

[Mér11] Quentin Mérigot. “A Multiscale Approach to Optimal Transport”. en. In: *Computer Graphics Forum* 30.5 (Aug. 2011), pp. 1583–1592 (cit. on p. 29).

[Mét+16] L Métivier, R Brossier, Q Mérigot, E Oudet, and J Virieux. “An Optimal Transport Approach for Seismic Tomography: Application to 3D Full Waveform Inversion”. en. In: *Inverse Problems* 32.11 (Nov. 2016), p. 115008 (cit. on p. 33).

[Mey+18] Jocelyn Meyron, Quentin Mérigot, and Boris Thibert. “Light in Power: A General and Parameter-Free Algorithm for Caustic Design”. en. In: *ACM Transactions on Graphics* 37.6 (Dec. 2018), pp. 1–13. arXiv: 1708.04820 (cit. on pp. 29, 31).

[Mik+13] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. “Distributed Representations of Words and Phrases and Their Compositionality”. en. In: (2013), p. 9 (cit. on pp. 33, 40).

[MS02] Paul Milgrom and Ilya Segal. “Envelope Theorems for Arbitrary Choice Sets”. en. In: *Econometrica* 70.2 (Mar. 2002), pp. 583–601 (cit. on p. 46).

[MD17] Jean-Marie Mirebeau and Johann Dreo. “Automatic Differentiation of Non-Holonomic Fast Marching for Computing Most Threatening Trajectories under Sensors Surveillance”. en. In: *arXiv:1704.03782 [math]* (Apr. 2017). arXiv: 1704.03782 [math] (cit. on pp. 41, 92).

[Mon81] Gaspard Monge. “Mémoire Sur La Théorie Des Déblais et Des Remblais”. In: *Histoire de l’Académie Royale des Sciences de Paris* (1781) (cit. on pp. 2, 6, 10).

[Mon+16] Grégoire Montavon, Klaus-Robert Müller, and Marco Cuturi. “Wasserstein Training of Restricted Boltzmann Machines”. en. In: (2016), p. 9 (cit. on p. 31).

[Nes83] Yurii Nesterov. “A Method for Unconstrained Convex Minimization Problem with the Rate of Convergence O (1/K̂2)”. In: *Doklady AN USSR*. Vol. 269. 1983, pp. 543–547 (cit. on p. 54).

[Ngu+12] Hien Van Nguyen, Vishal M Patel, Nasser M Nasrabadi, and Rama Chellappa. “Kernel Dictionary Learning”. en. In: (2012), p. 4 (cit. on p. 36).

[OF97] Bruno A. Olshausen and David J. Field. “Sparse Coding with an Overcomplete Basis Set: A Strategy Employed by V1?” en. In: *Vision Research* 37.23 (Dec. 1997), pp. 3311–3325 (cit. on p. 35).

[Pap+11] N Papadakis, E Provenzi, and V Caselles. “A Variational Model for Histogram Transfer of Color Images”. en. In: *IEEE Transactions on Image Processing* 20.6 (June 2011), pp. 1682–1695 (cit. on p. 30).

[Pap+14] Nicolas Papadakis, Gabriel Peyré, and Edouard Oudet. “Optimal Transport with Proximal Splitting”. en. In: *SIAM Journal on Imaging Sciences* 7.1 (Jan. 2014), pp. 212–238. arXiv: 1304.5784 (cit. on p. 29).

[Pas+17] Adam Paszke, Sam Gross, Soumith Chintala, et al. “Automatic Differentiation in PyTorch”. en. In: (2017), p. 4 (cit. on pp. 46, 74).

[Pea01] Karl Pearson. “On Lines and Planes of Closest Fit to Systems of Points in Space”. In: *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science* 2.11 (1901), pp. 559–572 (cit. on p. 35).

[PB16] Ofir Pele and Yakir Ben-Aliz. “Interpolated Discretized Embedding of Single Vectors and Vector Pairs for Classification, Metric Learning and Distance Approximation”. en. In: *arXiv:1608.02484 [cs]* (Aug. 2016). arXiv: 1608.02484 [cs] (cit. on p. 39).

[Pen+14] Jeffrey Pennington, Richard Socher, and Christopher D. Manning. “Glove: Global Vectors for Word Representation.” In: *EMNLP*. Vol. 14. 2014, pp. 1532–1543 (cit. on pp. 33, 59).

[Per+16] Michaël Perrot, Nicolas Courty, Rémi Flamary, and Amaury Habrard. “Mapping Estimation for Discrete Optimal Transport”. en. In: (2016), p. 9 (cit. on p. 31).

[Pey15] Gabriel Peyré. “Entropic Wasserstein Gradient Flows”. en. In: *arXiv:1502.06216 [math]* (Feb. 2015). arXiv: 1502.06216 [math] (cit. on p. 31).

[PC18] Gabriel Peyré and Marco Cuturi. *Computational Optimal Transport*. Now Publishers, Inc., 2018 (cit. on pp. v, 13, 16, 18, 19, 28, 30).

[Pey+16] Gabriel Peyré, Lenaïc Chizat, François-Xavier Vialard, and Justin Solomon. “Quantum Optimal Transport for Tensor Field Processing”. en. In: *arXiv:1612.08731 [cs]* (Dec. 2016). arXiv: 1612.08731 [cs] (cit. on p. 54).

[PK07] F. Pitié and A. Kokaram. “The Linear Monge-Kantorovitch Linear Colour Mapping for Example-Based Colour Transfer”. en. In: *IET 4th European Conference on Visual Media Production (CVMP 2007)*. Vol. 2007. London, UK: IEE, 2007, pp. 23–23 (cit. on p. 30).

[Pit+05] F. Pitié, A.C. Kokaram, and R. Dahyot. “N-Dimensional Probability Density Function Transfer and Its Application to Color Transfer”. en. In: *Tenth IEEE International Conference on Computer Vision (ICCV'05) Volume 1*. Beijing, China: IEEE, 2005, 1434–1439 Vol. 2 (cit. on p. 29).

[Pol64] B.T. Polyak. “Some Methods of Speeding up the Convergence of Iteration Methods”. en. In: *USSR Computational Mathematics and Mathematical Physics* 4.5 (Jan. 1964), pp. 1–17 (cit. on p. 54).

[Qua18] Kent Quanrud. “Approximating Optimal Transport with Linear Programs”. en. In: *arXiv:1810.05957 [cs]* (Oct. 2018). arXiv: 1810.05957 [cs] (cit. on p. 16).

[RP15] Julien Rabin and Nicolas Papadakis. “Convex Color Image Segmentation with Optimal Transport Distances”. en. In: *arXiv:1503.01986 [cs]* (Mar. 2015). arXiv: 1503.01986 [cs] (cit. on p. 31).

[Rab+10] Julien Rabin, Julie Delon, and Yann Gousseau. “Regularization of Transportation Maps for Color and Contrast Transfer”. en. In: *2010 IEEE International Conference on Image Processing*. Hong Kong, Hong Kong: IEEE, Sept. 2010, pp. 1933–1936 (cit. on pp. 17, 30).

[Rab+11] Julien Rabin, Gabriel Peyré, Julie Delon, and Marc Bernot. “Wasserstein Barycenter and Its Application to Texture Mixing”. en. In: *Scale Space and Variational Methods in Computer Vision*. Ed. by Alfred M. Bruckstein, Bart M. ter Haar Romeny, Alexander M. Bronstein, and Michael M. Bronstein. Vol. 6667. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011, pp. 435–446 (cit. on pp. 16, 29, 31).

[RB11] D. Ramanan and S. Baker. “Local Distance Functions: A Taxonomy, New Algorithms, and an Evaluation”. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 33.4 (Apr. 2011), pp. 794–806 (cit. on p. 90).

[Ric+13] E. Ricci, G. Zen, N. Sebe, and S. Messelodi. “A Prototype Learning Framework Using EMD: Application to Complex Scenes Analysis”. en. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 35.3 (Mar. 2013), pp. 513–526 (cit. on p. 37).

[Rol+16] Antoine Rolet, Marco Cuturi, and Gabriel Peyré. “Fast Dictionary Learning with a Smoothed Wasserstein Loss”. In: *Proceedings of the 19th International Conference on Artificial Intelligence and Statistics*. Ed. by Arthur Gretton and Christian C. Robert. Vol. 51. Proceedings of Machine Learning Research. Cadiz, Spain: PMLR, May 2016, pp. 630–638 (cit. on pp. 31, 33, 37).

[Rub+08] Ron Rubinstein, Michael Zibulevsky, and Michael Elad. “Efficient Implementation of the K-SVD Algorithm Using Batch Orthogonal Matching Pursuit”. In: *Cs Technion* 40.8 (2008), pp. 1–15 (cit. on pp. 35, 58, 60).

[Rub+00] Yossi Rubner, Carlo Tomasi, and Leonidas J Guibas. “The Earth Mover’s Distance as a Metric for Image Retrieval”. en. In: *International Journal of Computer Vision* (2000), p. 23 (cit. on pp. 13, 30, 65).

[SM86] Gerard Salton and Michael J McGill. “Introduction to Modern Information Retrieval”. In: (1986) (cit. on p. 59).

[SL11] Roman Sandler and Michael Lindenbaum. “Nonnegative Matrix Factorization with Earth Mover’s Distance Metric for Image Analysis”. en. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 33.8 (Aug. 2011), pp. 1590–1602 (cit. on pp. 31, 37, 40).

[San15] Filippo Santambrogio. *Optimal Transport for Applied Mathematicians*. Springer, 2015 (cit. on pp. 13, 30).

[Sch+19] Geoffrey Schiebinger, Jian Shu, Marcin Tabaka, et al. “Optimal-Transport Analysis of Single-Cell Gene Expression Identifies Developmental Trajectories in Reprogramming”. en. In: *Cell* 176.4 (Feb. 2019), 928–943.e22 (cit. on p. 41).

[Sch+17] Morgan A Schmitz, Matthieu Heitz, Nicolas Bonneel, et al. “Optimal Transport-Based Dictionary Learning and Its Application to Euclid-like Point Spread Function Representation”. In: *Wavelets and Sparsity XVII*. San Diego, United States: SPIE, Aug. 2017 (cit. on p. 43).

[Sch+18] Morgan A Schmitz, Matthieu Heitz, Nicolas Bonneel, et al. “Wasserstein Dictionary Learning: Optimal Transport-Based Unsupervised Non-Linear Dictionary Learning”. In: *SIAM Journal on Imaging Sciences* (2018) (cit. on pp. 34, 43, 47, 49, 50, 55, 56).

[Sch16] Bernhard Schmitzer. “Stabilized Sparse Scaling Algorithms for Entropy Regularized Transport Problems”. In: *arXiv:1610.06519 [cs, math]* (Oct. 2016). arXiv: 1610.06519 [cs, math] (cit. on pp. 21, 31, 50–52).

[SS13] Bernhard Schmitzer and Christoph Schnörr. “Object Segmentation by Shape Matching with Wasserstein Modes”. en. In: *Energy Minimization Methods in Computer Vision and Pattern Recognition*. Vol. 8081. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, pp. 123–136 (cit. on pp. 31, 32).

[Sch+97] Bernhard Schölkopf, Alexander Smola, and Klaus-Robert Müller. “Kernel Principal Component Analysis”. en. In: *Artificial Neural Networks — ICANN’97*. Ed. by Gerhard Goos, Juris Hartmanis, Jan van Leeuwen, et al. Vol. 1327. Berlin, Heidelberg: Springer Berlin Heidelberg, 1997, pp. 583–588 (cit. on p. 36).

[SC15] Vivien Seguy and Marco Cuturi. “Principal Geodesic Analysis for Probability Measures under the Optimal Transport Metric”. In: *Advances in Neural Information Processing Systems*. 2015, pp. 3312–3320 (cit. on pp. 37, 56).

[Seg+18] Vivien Seguy, Bharath Bhushan Damodaran, Rémi Flamary, et al. “Large-Scale Optimal Transport and Mapping Estimation”. en. In: *arXiv:1711.02283 [stat]* (Feb. 2018). arXiv: 1711.02283 [stat] (cit. on p. 31).

[Shi+14] Yuan Shi, Aurelien Bellet, and Fei Sha. “Sparse Compositional Metric Learning”. en. In: (2014), p. 7 (cit. on p. 90).

[SJ08] Sameer Shirdhonkar and David W. Jacobs. “Approximate Earth Mover’s Distance in Linear Time”. en. In: *2008 IEEE Conference on Computer Vision and Pattern Recognition*. Anchorage, AK, USA: IEEE, June 2008, pp. 1–8 (cit. on p. 16).

[SF18] Effrosyni Simou and Pascal Frossard. “Graph Signal Representation with Wasserstein Barycenters”. en. In: *arXiv:1812.05517 [eess]* (Dec. 2018). arXiv: 1812.05517 [eess] (cit. on pp. 34, 37).

[SK67] Richard Sinkhorn and Paul Knopp. “Concerning Nonnegative Matrices and Doubly Stochastic Matrices”. en. In: *Pacific Journal of Mathematics* 21.2 (May 1967), pp. 343–348 (cit. on p. 18).

[SR06] Paris Smaragdis and Bhiksha Raj. “A Probabilistic Latent Variable Model for Acoustic Modeling”. en. In: (2006), p. 6 (cit. on pp. 35, 37).

[Sol+14] Justin Solomon, Raif Rustamov, Leonidas Guibas, and Adrian Butscher. “Earth Mover’s Distances on Discrete Surfaces”. en. In: *ACM Transactions on Graphics* 33.4 (July 2014), pp. 1–12 (cit. on p. 32).

[Sol+15] Justin Solomon, Fernando de Goes, Gabriel Peyré, et al. “Convolutional Wasserstein Distances: Efficient Optimal Transportation on Geometric Domains”. In: *ACM Trans. Graph.* 34.4 (July 2015), 66:1–66:11 (cit. on pp. 23, 30, 47, 67, 68, 70–72, 86).

[Sta+17] Matthew Staib, Sebastian Claici, Justin Solomon, and Stefanie Jegelka. “Parallel Streaming Wasserstein Barycenters”. In: *arXiv:1705.07443 [cs, math, stat]* (May 2017). arXiv: 1705.07443 [cs, math, stat] (cit. on p. 27).

[SW19] Bing Su and Ying Wu. “Learning Distance for Sequences by Learning a Ground Metric”. en. In: (2019), p. 11 (cit. on p. 40).

[Tar+16] Guillaume Tartavel, Gabriel Peyré, and Yann Gousseau. “Wasserstein Loss for Image Synthesis and Restoration”. en. In: *SIAM Journal on Imaging Sciences* 9.4 (Jan. 2016), pp. 1726–1755 (cit. on p. 31).

[Tea+16] The Theano Development Team, Rami Al-Rfou, Guillaume Alain, et al. “Theano: A Python Framework for Fast Computation of Mathematical Expressions”. In: *arXiv:1605.02688 [cs]* (May 2016). arXiv: 1605.02688 [cs] (cit. on p. 46).

[Tha+14] Dorina Thanou, David I Shuman, and Pascal Frossard. “Learning Parametric Dictionaries for Signals on Graphs”. en. In: *IEEE Transactions on Signal Processing* 62.15 (Aug. 2014), pp. 3849–3862 (cit. on p. 36).

[Thi+17] Alexis Thibault, Lénaïc Chizat, Charles Dossal, and Nicolas Papadakis. “Overrelaxed Sinkhorn-Knopp Algorithm for Regularized Optimal Transport”. In: *arXiv preprint arXiv:1711.01851* (2017). arXiv: 1711.01851 (cit. on p. 64).

[Tol30] AN Tolstoı. “Metody Nakhozhdeniya Naimen’shego Summovogo Kilometrazha Pri Planirovanii Perevozok v Prostranstve [Russian; Methods of Finding the Minimal Total Kilometrage in Cargo-Transportation Planning in Space]”. In: *Planirovanie Perevozok, Sbornik pervyı [Russian; Transportation Planning, Volume I], Transpechat’NKPS [TransPress of the National Commissariat of Transportation], Moscow* (1930), pp. 23–55 (cit. on p. 16).

[TL07] Lorenzo Torresani and Kuang-chih Lee. “Large Margin Component Analysis”. en. In: *Advances in neural information processing systems* (2007), p. 8 (cit. on p. 39).

[TP91] Matthew Turk and Alex Pentland. “Eigenfaces for Recognition”. In: *Journal of Cognitive Neuroscience* 3.1 (1991), pp. 71–86 (cit. on pp. 58, 60).

[Uza14] Hirofumi Uzawa. “The Kuhn-Tucker Theorem in Concave Programming”. en. In: *Traces and Emergence of Nonlinear Programming*. Ed. by Giorgio Giorgi and Tinne Hoff Kjeldsen. Basel: Springer, 2014, pp. 307–312 (cit. on p. 17).

[Var67] Sathamangalam R. Srinivasa Varadhan. “On the Behavior of the Fundamental Solution of the Heat Equation with Variable Coefficients”. In: *Communications on Pure and Applied Mathematics* 20.2 (1967), pp. 431–455 (cit. on pp. 67, 70).

[Vil03] Cédric Villani. *Topics in Optimal Transportation*. en. American Mathematical Soc., 2003 (cit. on p. 13).

[Vil09] Cédric Villani. *Optimal Transport: Old and New*. en. Grundlehren Der Mathematischen Wissenschaften 338. Berlin: Springer, 2009 (cit. on p. 13).

[WG12] Fan Wang and Leonidas J. Guibas. “Supervised Earth Mover’s Distance Learning and Its Computer Vision Applications”. en. In: *Computer Vision – ECCV 2012*. Vol. 7572. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, pp. 442–455 (cit. on pp. 40, 41).

[Wan+11] Jun Wang, Huyen T Do, Adam Woznica, and Alexandros Kalousis. “Metric Learning with Multiple Kernels”. en. In: *Advances in Neural Information Processing Systems* (2011), p. 9 (cit. on p. 39).

[Wan+12] Jun Wang, Alexandros Kalousis, and Adam Woznica. “Parametric Local Metric Learning for Nearest Neighbor Classification”. en. In: (2012), p. 9 (cit. on pp. 36, 90).

[Wan+13] Wei Wang, Dejan Slepčev, Saurav Basu, John A. Ozolek, and Gustavo K. Rohde. “A Linear Optimal Transportation Framework for Quantifying and Visualizing Variations in Sets of Images”. In: *International journal of computer vision* 101.2 (2013), pp. 254–269 (cit. on p. 37).

[WS08] Kilian Q. Weinberger and Lawrence K. Saul. “Fast Solvers and Efficient Implementations for Distance Metric Learning”. en. In: *Proceedings of the 25th International Conference on Machine Learning - ICML ’08*. Helsinki, Finland: ACM Press, 2008, pp. 1160–1167 (cit. on pp. 40, 41).

[WS09] Kilian Q Weinberger and Lawrence K Saul. “Distance Metric Learning for Large Margin Nearest Neighbor Classification”. en. In: (2009), p. 38 (cit. on p. 90).

[Wei+06] Kilian Q Weinberger, John Blitzer, and Lawrence K Saul. “Distance Metric Learning for Large Margin Nearest Neighbor Classification”. en. In: *Advances in neural information processing systems* (2006), p. 8 (cit. on p. 39).

[Xie+13] Yuchen Xie, Jeffrey Ho, and Baba Vemuri. “On A Nonlinear Generalization of Sparse Coding and Dictionary Learning”. en. In: (2013), p. 9 (cit. on p. 36).

[Xie+18] Yujia Xie, Xiangfeng Wang, Ruijia Wang, and Hongyuan Zha. “A Fast Proximal Point Method for Computing Exact Wasserstein Distance”. en. In: *arXiv:1802.04307 [cs, stat]* (Feb. 2018). arXiv: 1802.04307 [cs, stat] (cit. on pp. 27, 31, 64).

[Xin+03] Eric P Xing, Michael I Jordan, Stuart J Russell, and Andrew Y Ng. “Distance Metric Learning with Application to Clustering with Side-Information”. en. In: *Advances in Neural Information Processing Systems*. 2003, p. 8 (cit. on p. 38).

[Xu+18a] Hongteng Xu, Wenlin Wang, Wei Liu, and Lawrence Carin. “Distilled Wasserstein Learning for Word Embedding and Topic Modeling”. en. In: (2018), p. 10 (cit. on pp. 33, 34, 37, 40).

[Xu+19] Hongteng Xu, Dixin Luo, Hongyuan Zha, and Lawrence Carin. “Gromov-Wasserstein Learning for Graph Matching and Node Embedding”. en. In: *arXiv:1901.06003 [cs, stat]* (Jan. 2019). arXiv: 1901.06003 [cs, stat] (cit. on p. 40).

[Xu+18b] Jie Xu, Lei Luo, Cheng Deng, and Heng Huang. “Multi-Level Metric Learning via Smoothed Wasserstein Distance”. en. In: *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence*. Stockholm, Sweden: International Joint Conferences on Artificial Intelligence Organization, July 2018, pp. 2919–2925 (cit. on pp. 40, 41).

[Xu+17] Y. Xu, Z. Li, J. Yang, and D. Zhang. “A Survey of Dictionary Learning Algorithms for Face Recognition”. In: *IEEE Access* 5 (2017), pp. 8502–8514 (cit. on p. 36).

[YC16] Fang Yang and Laurent D. Cohen. “Geodesic Distance and Curves Through Isotropic and Anisotropic Heat Equations on Images and Surfaces”. en. In: *Journal of Mathematical Imaging and Vision* 55.2 (June 2016), pp. 210–228 (cit. on p. 65).

[Yan+15] Wei Yang, Luhui Xu, Xiaopan Chen, Fengbin Zheng, and Yang Liu. “Chi-Squared Distance Metric Learning for Histogram Data”. en. In: *Mathematical Problems in Engineering* 2015 (2015), pp. 1–12 (cit. on p. 39).

[Ye+17] Jianbo Ye, Panruo Wu, James Z. Wang, and Jia Li. “Fast Discrete Distribution Clustering Using Wasserstein Barycenter with Sparse Support”. In: *IEEE Transactions on Signal Processing* 65.9 (2017), pp. 2317–2332 (cit. on p. 26).

[Yur+19] Mikhail Yurochkin, Sebastian Claici, Edward Chien, Farzaneh Mirzazadeh, and Justin Solomon. “Hierarchical Optimal Transport for Document Representation”. en. In: *arXiv:1906.10827 [cs, stat]* (June 2019). arXiv: 1906.10827 [cs, stat] (cit. on pp. 33, 34, 95).

[Zen+14] G. Zen, E. Ricci, and N. Sebe. “Simultaneous Ground Metric Learning and Matrix Factorization with Earth Mover’s Distance”. In: *2014 22nd International Conference on Pattern Recognition*. Aug. 2014, pp. 3690–3695 (cit. on pp. 31, 37, 40, 41).

[Zha+09] De-Chuan Zhan, Ming Li, Yu-Feng Li, and Zhi-Hua Zhou. “Learning Instance Specific Distances Using Metric Propagation”. en. In: *Proceedings of the 26th Annual International Conference on Machine Learning - ICML ’09*. Montreal, Quebec, Canada: ACM Press, 2009, pp. 1–8 (cit. on p. 90).

[ZP01] Michael Zibulevsky and Barak A Pearlmutter. “Blind Source Separation by Sparse Decomposition in a Signal Dictionary”. en. In: (2001), p. 20 (cit. on p. 35).

List of Algorithms

List of Figures