Le Transport Optimal, un couteau suisse pour la Data Science

Pirmin Lemberger
Directeur Scientifique

15 avril 2019

onepoint.
beyond the obvious

Résumé

Dans La théorie du transport optimal plonge ses racines dans l'histoire des mathématiques avec un problème formulé au 18ème siècle par Gaspard Monge qui se demandait comment déplacer un tas de sable en fournissant le moindre effort possible. *Grace à une succession de progrès théoriques et, plus récemment, algorithmiques, cette théorie trouve aujourd'hui de nombreuses applications pratiques en data science. Elle permet tout à la fois de définir une notion géométrique et naturelle de distance entre deux distributions de probabilités et de « morpher » une distribution en une autre à moindre coût. Cet article est une introduction modérément technique au sujet et s'adresse prioritairement aux data scientist.*

1. Un outil polyvalent encore méconnu

Le terme de transport optimal (TO) convoque immanquablement l'association d'idée avec les problèmes qui préoccupent la SNCF ou la RATP et leurs clients. En vérité cette association n'est pas totalement infondée, même si les liens sont beaucoup plus étroits comme nous le verrons entre le TO et la Data Science qu'avec les questions ferroviaires.

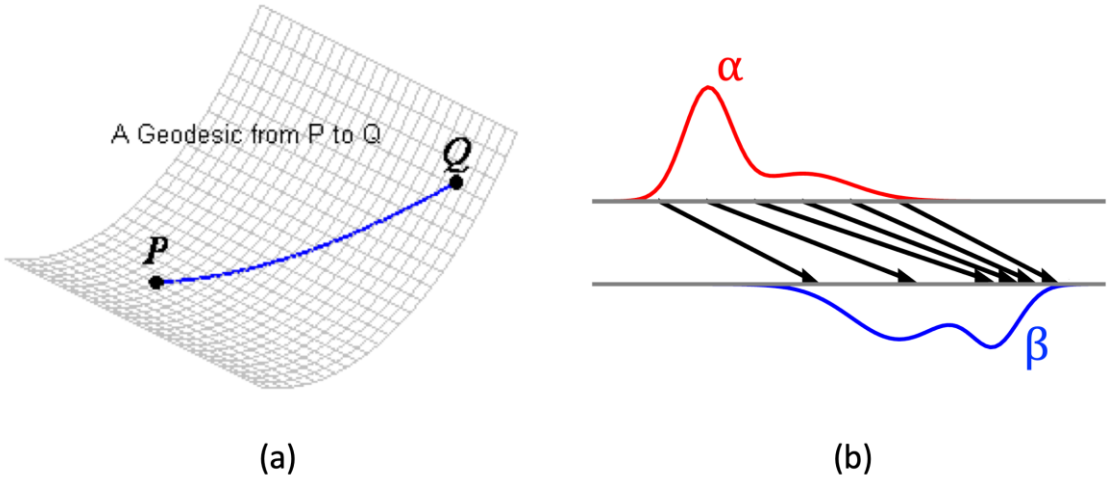


Figure 1 : (a) le plus court chemin entre deux points P et Q sur une surface est une géodésique. (b) Le TO permet de décrire une transformation qui transforme progressivement une distribution source α en une distribution cible β en déplaçant de petits morceaux de masse pour minimiser une certaine notion d'effort.

Cependant, commençons par le commencement. La question du **déplacement à moindre coût** d'un objet est un problème aussi ancien que la mécanique ou la géométrie. On peut se proposer par exemple de chercher le chemin le plus court qui relie deux points sur une surface (fig. 1(a)), on parle alors de géodésique. Plus complexe en revanche est le problème qui consiste à déplacer simultanément tous les grains d'un tas de sable pour en former un autre un peu plus loin dont on prescrit la forme (fig. 1(b)) et ceci tout en minimisant l'effort à produire. Il s'agit en l'occurrence de déplacer littéralement une infinité de grains en respectant à la fois la contrainte globale d'effort minimal et la contrainte de la forme prescrite du tas cible. Aussi surprenant que cela paraisse ce problème s'est avéré si ardu que pas moins de 150 ans se sont écoulés entre sa formulation originale par Gaspard **Monge** et une première esquisse de solution durant la seconde guerre mondiale par un mathématicien russe, Leonid **Kantorovitch**, alors expert en optimisation d'allocation de ressources.

Et lien avec la **Data Science** dans tout ça ? Patience, nous y viendrons. Mais avant cela un peu de théorie sera nécessaire pour y voir clair.

Les distributions de probabilités étant omniprésentes dans le **Machine Learning** (ML), qu'elles soient théoriques ou empiriques, qu'on admette simplement pour l'instant qu'il puisse être utile de savoir mesurer leur séparation ou, mieux encore, de pouvoir les transformer à moindre frais.

Pour le bénéfice des lecteurs les plus pressés voici sans attendre et sans explication une liste non exhaustive d'applications de la théorie du TO en ML qui, je l'espère, les convaincra qu'ils ne perdent pas leur temps avec des futilités académiques.

Le TO est utile, entre autres, dans les contextes pratiques et théoriques suivants :

- Le **traitement d'images** pour lequel il permet le transfert de couleurs ou de textures, le débruitage et l'augmentation de résolution ainsi que l'**imagerie médicale** [COT].

- La **classification de texte** pour laquelle il fournit des alternatives aux méthodes traditionnelles [DCL].

- La **classification multi-labels** pour laquelle il fournit une fonction de coût approprié à la comparaison de plusieurs histogrammes [LWL].

- Le problème de l'**adaptation de domaine** où il s'agit d'exploiter l'information disponible dans un train set pour construire un prédicteur qui sera appliqué à un test set échantillonné à partir d'une autre population [ODA, JDA, DJT]

- La **théorie de la généralisation** pour laquelle il offre une alternative aux mesures classiques de complexité (VC-dimension, complexité de Rademacher etc, ...) [DDD]

- L'élaboration de **modèles génératifs** performants comme les GAN [GAN].

- L'**inférence bayésienne** [AOT] pour lequel il offre une alternative aux simulations MCMC dans le calcul de la distribution à postériori.

Il n'est pas exclu par ailleurs que la théorie du TO soit en mesure d'apporter quelques lumières sur les capacités de généralisation encore mystérieuses des réseaux de neurones profonds [OTV].

La suite de cet article est organisée comme suit. La **section 2** introduit les principaux concepts du TO, notamment les formulations de Monge et de Kantorovich ainsi que la distance de Wasserstein. La présentation fait usage d'un minimum de formalisme et d'un maximum d'illustrations. La **section 3** décrit quelques applications pratiques du TO. La **section 4** fait un zoom sur une application phare du TO à la data science : l'adaptation de domaine non supervisée. Le problème consiste à construire un prédicteur destiné à un domaine cible différent de celui pour lequel on dispose de données d'entraînement, une situation que l'on rencontre couramment en pratique. La **section 5** présente l'une des avancées algorithmiques récente, l'algorithme de Sinkhorn qui a permet de calculer efficacement la distance de Wasserstein grâce à un procédé de régularisation entropique. La **section 6** propose quelques remarques de conclusion.

2. Vous reprendrez bien un peu de théorie ?

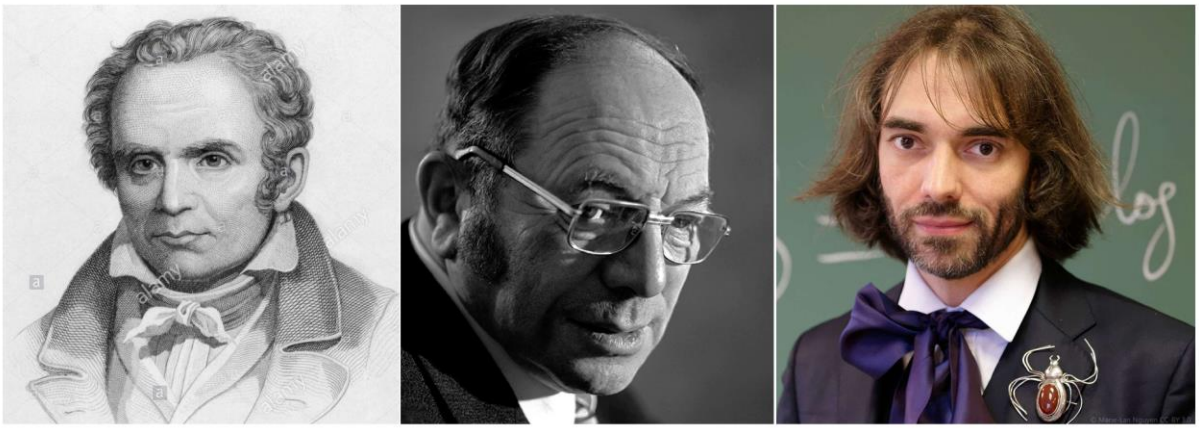


Figure 2 : Gaspard Monge [1746–1818], Leonid Kantorovitch [1912-1986] et Cédric Villani, 3 mathématiciens qui ont contribué à la théorie du transport optimal.

Quelques définitions

Ce paragraphe introduit les principaux concepts du TO en privilégiant les illustrations sur les définitions mathématiques rigoureuses. Nous nous inspirons librement de l'excellent ouvrage *Computational Optimal Transport* [COT].

L'idée intuitive d'une **distribution de probabilités** qui assigne des poids à des points (cas **discret**) ou à des régions de l'espace (cas **continu**) est illustrée sur la figure 3 pour les dimensions d=1 ou d=2. Cette distinction discret–continu sera traitée de manière cavalière dans la suite de cet article et nous ici faisons appel à l'intuition des lecteurs non mathématiciens et à l'indulgence de ceux qui sont plus aguerris aux rigueurs des définitions précises.

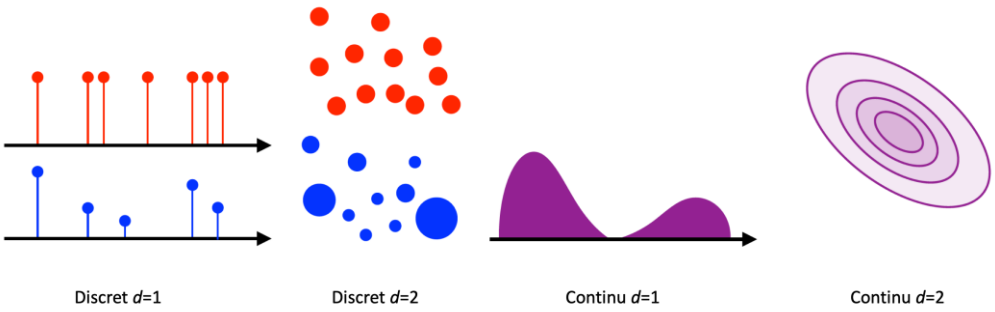


Figure 3 : Représentations intuitives des notions de distribution de probabilités discrètes et continues à d=1 et d=2 dimensions – source [AOT].

Très schématiquement, dans le cas discret une distribution peut s'écrire comme une somme pondérée $\mu = \sum_i a_i \delta_{x_i}$ de mesures δ_{x_i} ponctuelles alors que dans la cas continu μ possède une densité $\mu(\mathbf{x})$.

La formulation originale de Monge

Considérons à titre d'exemple deux distributions μ et ν continues à une dimension et supposons que nous cherchions une transformation $\mathbf{x} \rightarrow T(\mathbf{x})$ qui **transporte** la totalité de la masse de μ (un « tas de sable ») vers ν (un « trou ») comme l'illustre la figure 4. En raccourci on notera cette opération de transport de distribution : $\nu = T_{\#}\mu$.

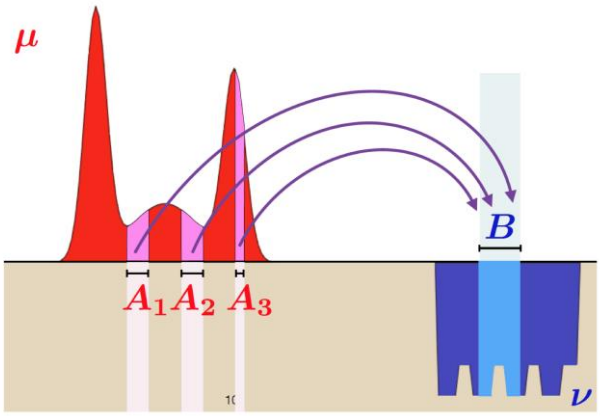


Figure 4 : La transformation $\mathbf{x} \rightarrow T(\mathbf{x})$ *transvase la masse de* μ *vers* ν. *La masse dans l'intervalle B provient de 3 petits intervalles* A_1, A_2 *et* A_3 – source [AOT].

Pour exprimer l'idée d'effort minimal il nous faut introduire une **fonction de coût** C qui mesure l'effort $C(\mathbf{x}, T(\mathbf{x}))$ à fournir pour transporter un grain de sable de la position $\mathbf{x}$ vers la position $T(\mathbf{x})$. L'effort total à produire est alors l'accumulation de ces efforts infinitésimaux, que l'on exprime avec une intégrale $\int C(\mathbf{x},T(\mathbf{x}))\, d\mu(\mathbf{x})$ pour le cas continu (figure 4) et avec une somme $\sum_i \mu(\mathbf{x}_i)C(\mathbf{x}_i,T(\mathbf{x}_i))$ pour la cas discret qui nous intéressera en pratique (figure 5).

En résumé, le problème du transport optimal formulé par **Monge** consiste donc à chercher une solution pour le problème suivant :

> Trouver une transformation T qui minimise $\sum_i \mu(\mathbf{x}_i)C(\mathbf{x}_i,T(\mathbf{x}_i))$ en respectant $\nu = T_{\#}\mu$

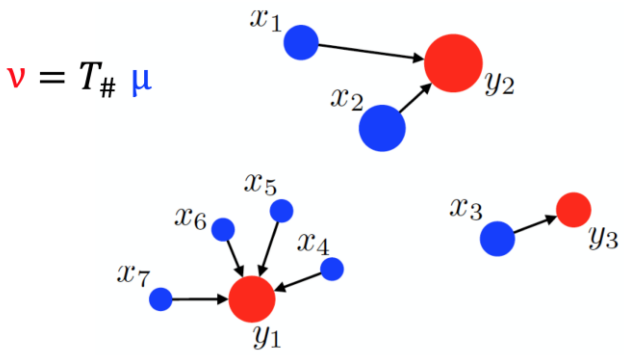


Figure 5 : Un exemple de transport d'une distribution discrète μ *supportée sur les points* $\mathbf{x}_i$ *vers une distribution* ν *supportée sur des points* $\mathbf{y}_j$. De telles distributions correspondent à des distributions empiriques associées à des échantillons d'observations – source [AOT].

Dans le cas discret, lorsque les deux distributions μ et ν sont réparties uniformément sur leur support le problème de Monge se réduit à un simple problème de combinatoire puisqu'il s'agit alors d'assigner un $\mathbf{y}_j$ à chaque $\mathbf{x}_i$, où $j = \sigma(i)$ est une permutation de i. Dans ce cas le problème possède toujours une solution. En revanche pour des distributions discrètes non-uniformes le problème ne possède pas nécessairement de solution.

La nature combinatoire du problème dans le cas discret et la complexité de la contrainte $\nu = T_{\#}\mu$ qui est **non convexe** font que le problème de Monge est très difficile.

La formulation relaxée de Kantorovich

Puisque le problème de Monge est trop ardu il faut nous résigner à en résoudre un plus simple ! C'est la stratégie proposée par Kantorovich, un mathématicien russe spécialiste de l'optimisation d'allocation de ressources durant la seconde guerre mondiale. Plutôt que d'exiger, comme le fait Monge, que la matière au point **x** d'une distribution α soit transportée de manière déterministe vers un point $T(\mathbf{x})$ dans la distribution cible β, Kantorovich propose d'autoriser la répartition de cette matière sur différents points dans la cible. La figure 6 illustre une nouvelle situation dans le cas discret et la figure 7 (a) donne un exemple du cas continu.

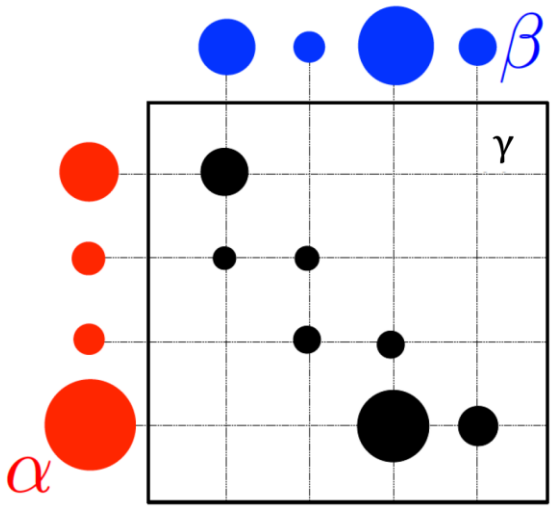


Figure 6 : un plan de transport γ entre deux distributions discrètes α et β. Le masse d'un point dans α peut être répartie sur plusieurs points dans β – source [AOT].

En termes un peu plus formels Kantorovich propose de chercher une distribution de **probabilités conjointes** $\gamma(\mathbf{x},\mathbf{y})$ qui indique la proportion de matière au voisinage de **x** dans α que l'on va transporter au voisinage de **y** dans β : c'est un **plan de transport**. Pour que le plan de transport γ soit compatible avec α et β il faut naturellement exiger les contraintes de marginalité $\sum_i \gamma(\mathbf{x}_i,\mathbf{y}_j) = \beta(\mathbf{y}_j)$ et $\sum_j \gamma(\mathbf{x}_i,\mathbf{y}_j) = \alpha(\mathbf{x}_i)$. Notons $U(\alpha,\beta)$ l'ensemble des plans de transport γ ainsi compatibles avec α et β.

Si $C(\mathbf{x},\mathbf{y})$ désigne comme précédemment le coût d'un transport de **x** vers **y**, le coût total $L_C(\alpha,\beta)$ pour transporter α vers β avec le plan γ vaudra $\sum_{ij} \gamma(\mathbf{x}_i,\mathbf{y}_j)\, C(\mathbf{x}_i,\mathbf{y}_j)$. Enfin, le coût de **transport optimal** de α vers β est le plus petit qu'on puisse réaliser :

$$L_C(\alpha,\beta) \equiv \text{minimum de } \sum_{ij} \gamma(\mathbf{x}_i,\mathbf{y}_j)\, C(\mathbf{x}_i,\mathbf{y}_j) \text{ parmi tous les plans } \gamma \text{ dans } U(\alpha,\beta)$$

Contrairement au problème de Monge, on constate que la formulation de Kantorovich est **symétrique** vis-à-vis des deux distributions α et β. C'est par ailleurs un problème de **programmation linéaire convexe**.

Un plan de transport γ de Kantorovich se réduit à un transport de Monge dans le cas particulier où la distribution $\gamma(\mathbf{x},\mathbf{y})$ est concentrée le long du graphe $\mathbf{x} \rightarrow T(\mathbf{x})$ comme l'illustre la figure 7 (b).

Dans certains cas on peut démontrer [OTA] que la solution du transport optimal de Kantorovich se réduit à une solution de type Monge. C'est la cas par exemple lorsque α et β sont toutes deux continues et que la fonction de coût $C(\mathbf{x},\mathbf{y}) = ||\mathbf{x} - \mathbf{y}||^2$ est le carré de la distance euclidienne entre **x** et **y**.

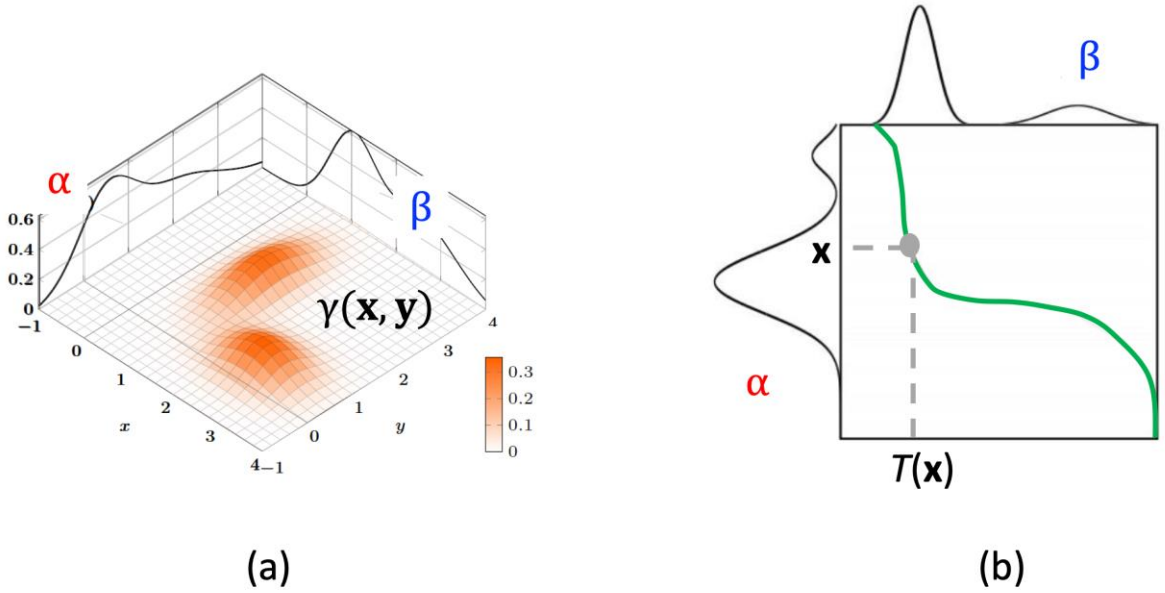


Figure 7 : (a) un plan de transport de Kantorovich γ dont les marginales α et β sont prescrites, (b) un transport de Monge correspond à un plan de transport $\gamma(\mathbf{x},\mathbf{y})$ concentré sur une courbe $\mathbf{x} \rightarrow T(\mathbf{x})$ – source [AOT].

La distance de Wasserstein

Nous possédons désormais tous les éléments nécessaires pour définir une notion de distance entre deux distributions α et β . Pour la fonction de coût de transport C on choisit $C(\mathbf{x},\mathbf{y}) = D^p(\mathbf{x},\mathbf{y})$ où $D(\mathbf{x},\mathbf{y})$ est une distance entre **x** et **y** et où $p \geq 1$. On définit alors la **distance de p-Wasserstein** entre α et β comme :

$$W_p(\alpha,\beta) \equiv [\, L_D{}^p(\alpha,\beta)\,]^{1/p} \quad \text{pour} \quad p \geq 1$$

De fait, on peut montrer que W_p vérifie effectivement les axiomes d'une distance, en particulier l'inégalité triangulaire : $W_p(\alpha,\gamma) \leq W_p(\alpha,\beta) + W_p(\beta,\gamma)$.

Voilà une belle définition, encore faut-il savoir la calculer étant données deux distributions de probabilité α et β. Dans la plupart des cas il faudra recourir à des **méthodes numériques**, nous y reviendrons dans la section 5.

Dans certains **cas spécifiques** on peut néanmoins calculer W_p explicitement. En voici quelques-uns, ce qui nous permettrons d'affermir notre intuition :

- Pour la distance entre deux distributions ponctuelles $\alpha = \delta_x$ et $\beta = \delta_y$ on vérifie immédiatement que $W_p(\delta_x, \delta_y) = D(x,y)$ qui n'est autre que la distance entre les supports de α et β. Elle n'est pas belle la vie ?

- Dans le cas p=2 si l'on définit α' et β' comme les versions centrées de moyennes nulles de α et β, qu'on définit $\mathbf{m}_\alpha$ et $\mathbf{m}_\beta$ comme les moyennes respectives de α et β alors on a la décomposition $W_2(\alpha, \beta)^2 = W_2(\alpha', \beta')^2 + ||\mathbf{m}_\alpha - \mathbf{m}_\beta||^2$.

- Dans le cas p=2 avec deux gaussiennes $\alpha = N(\mathbf{m}_\alpha, \Sigma_\alpha)$ et $\beta = N(\mathbf{m}_\beta, \Sigma_\beta)$ on a

  $W_2(\alpha, \beta)^2 = ||\mathbf{m}_\alpha - \mathbf{m}_\beta||^2 + B(\Sigma_\alpha, \Sigma_\beta)$ où B est une métrique entre matrices de covariances que l'on sait calculer explicitement[1].

En résumé : la distance de Wasserstein fournit un **outil géométrique** pour comparer des distributions de probabilités.

Une interprétation du TO comme un modèle de facturation

La valeur du transport optimal $L_C(\alpha,\beta)$ possède une interprétation particulièrement intuitive dans le cas où $\alpha = \sum_i a_i \delta_{\mathbf{x}i}$ et $\beta = \sum_i b_i \delta_{\mathbf{y}i}$ sont toutes deux discrètes.

Imaginons que a_i désigne une quantité de marchandise disponible dans un **entrepôt** n°i situé en $\mathbf{x}_i$ et que b_j soit une quantité de marchandise à livrer à une **usine** n°j située en $\mathbf{y}_j$. Imaginons par ailleurs que $C(\mathbf{x},\mathbf{y})$ représente un coût maximal qu'une entreprise est prête à payer pour transporter une unité de matière première de $\mathbf{x}$ vers $\mathbf{y}$. Enfin, considérons un **modèle de pricing** simple où l'on facture un prix f_i l'enlèvement d'une unité à partir de l'entrepôt n°i et où l'on facture un prix g_j pour la livraison d'une unité à l'usine n°j. Le prix total à payer pour transporter l'ensemble des marchandises de tous les entrepôts vers toutes les usines vaut donc :

$$\sum_i a_i f_i + \sum_j b_j g_j$$

On peut alors montrer[2] que la valeur $L_C(\alpha,\beta)$ définie précédemment coïncide avec le prix total maximal facturable si le modèle de pricing respecte la contrainte de coût imposée à savoir $f_i + g_j \leq C(\mathbf{x}_i,\mathbf{y}_j)$. En résumé :

$$L_C(\alpha,\beta) = \text{maximum de } \sum_i a_i f_i + \sum_j b_j g_j \text{ parmi les pricings tels que } f_i + g_j \leq C(\mathbf{x}_i,\mathbf{y}_j)$$

[1] Il s'agit de la métrique de Bures.

[2] Mathématiquement il s'agit de la formulation duale, qui est la maximisation d'un problème concave, du problème de Kantorovich qui est un problème de minimisation convexe.

3. Quelques applications du TO

Traitement et recherche d'images

L'une des applications les plus directes du TO consiste à comparer des histogrammes colorimétriques d'images (figure 8) pour créer des systèmes de recherche performants. Étant donné une image de référence, un tel système fournira une liste d'images proches au sens de la distance de Wasserstein entre les histogrammes associés.

Figure 8 : Deux images aux profils colorimétriques différents que l'on pourra comparer grâce au TO – source [AOT].

Dans le cadre du traitement d'image, une application élémentaire du TO consiste à **normaliser le contraste** d'une image en redistribuant l'histogramme des niveaux de gris vers une distribution qui accorde une importance égale à tous les niveaux de gris comme l'illustre la figure 9 :

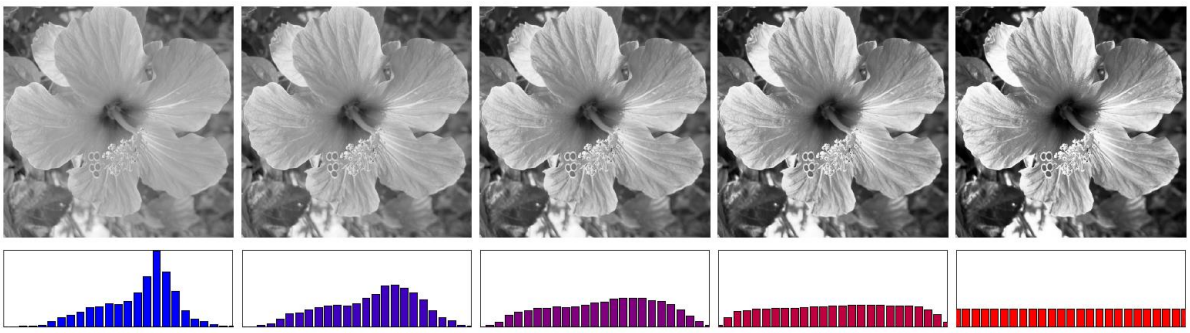


Figure 9 : Redistribution des niveaux de gris par TO pour créer une image de contraste maximal, la ligne du bas montre l'évolution de l'histogramme des niveaux de gris dans les images correspondantes – source [COT].

La classification de documents

L'idée de base pour appliquer le TO à la **classification de documents** est similaire à la technique classique du modèle génératif LDA (*Latent Dirichlet Allocation*) ou à la factorisation en matrice non négative (NMF) : chaque document d'un corpus est supposé être associé à un nombre limité de thèmes (politique, divertissement, actualité locale, environnement, publicité, …). A chacun de ces thèmes est associée une distribution de probabilités spécifique sur les mots. Étant donné un document il s'agit alors de retrouver le **mix de thèmes** qui le caractérise à partir de l'histogramme des fréquences des mots qu'il contient (figure 10).

Figure 10 : A chaque document est associé un histogramme qui décrit la fréquence des mots qu'il contient – source [AOT].

Là encore le TO fournit un outil qui permet de reformuler de manière très naturelle ce problème. A chaque document D_i d'un corpus de taille N on associe une distribution de probabilités β_i qui décrit la fréquence des mots (ou d'embeddings de mots) qu'il contient. L'objectif qu'on se propose est d'approximer chaque β_i par une combinaison linéaire d'un petit nombre K de distributions de base α_j. Plus formellement il s'agit de trouver ces distributions de base α_j et les coefficients Λ^i_j qui permettent de reconstruire au mieux les β_i … dans le sens d'une distance de Wasserstein minimale (en moyenne sur le corpus) :

Trouver des coeff Λ^i_j et des distributions α_j qui minimisent $\sum_{i=1}^{N} W(\beta_i, \sum_{j=1}^{K} \Lambda^j_i \alpha_j)$

Remarquons que les distributions de base α_j qui correspondent en quelques sorte aux thèmes principaux sont découverts par cette procédure.

La classification multilabel

Imaginons que nous souhaitions construire un modèle capable d'apprendre à dénombrer les objets présents dans une image. A chaque image $\mathbf{x}_i$ d'un train set utilisé pour entraîner un tel modèle il convient donc d'associer un vecteur y_i de labels qui dénombrent les objets qu'on y voit comme l'illustre la figure 11.

La **fonction de coût** utilisée pour comparer la prédiction $f_\theta(\mathbf{x}_i)$ du modèle avec le **multilabel** y_i du train set devra en particulier tenir compte de la proximité sémantique entre labels. Si par exemple $f_\theta(\mathbf{x}_i)$ = (2 « chiens », 3 « huskies ») et y_i = (3 « chiens », 1 « husky », 1 « loup ») il faudra considérer que la « distance » entre $f_\theta(\mathbf{x}_i)$ et y_i est faible. D'emblée on pressent tout l'intérêt dans cette situation du concept de distance de Wasserstein $W_C(f_\theta(\mathbf{x}_i), y_i)$. La métrique C(« mot A », « mot B ») qui permet de construire $W_C(.,.)$ est une notion de **distance sémantique** entre les termes susceptibles de désigner les objets dans un ensemble d'images. On peut la définir en utilisant une distance euclidienne entre 2 word embeddings par exemple [LWL].

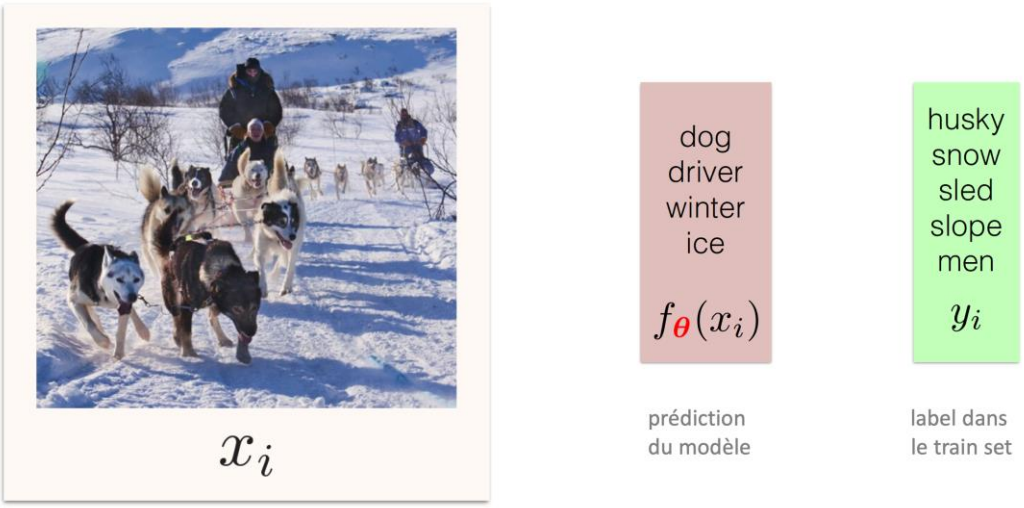


Figure 11 : Une classifieur multilabel associe à chaque image x_i un ensemble de labels $f_\theta(\mathbf{x}_i)$ qu'il s'agit de comparer aux labels y_i du train set – source [AOT].

En réalité nous avons passé sous silence un détail. Les histogrammes de fréquence $f_\theta(\mathbf{x}_i)$ et y_i ne sont pas à proprement parler des distributions de probabilités puisqu'ils ne sont pas normalisés si bien qu'on ne peut pas utiliser directement une distance de Wasserstein pour les comparer. Une légère adaptation du formalisme du TO est donc nécessaire pour l'appliquer à des mesures positives non normalisées [LWL].

Les modèles génératifs en grandes dimensions

Tout le monde a vu ces visages humains fictifs ultra-réalistes créés par des modèles génératifs récents. Techniquement il s'agit de construire une distribution de probabilités μ sur des images (disons de 1000 x 1000 x 3 pixels) dont les échantillons $\mathbf{x}_1$, $\mathbf{x}_2$, … ressemblent à des visages humains.

Figure 12 : Visages fictifs synthétisés par un modèle génératif d'images GAN – source [NVD].

Pour construire une distribution de probabilités μ dont les échantillons $\mathbf{x}\sim\mu$ « ressembleront » aux objets d'un ensemble d'entraînement $S = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_m\}$ on introduit généralement une distribution paramétrée μ_θ et l'on cherche les paramètres θ qui maximisent la vraisemblance $\mu_\theta(S) \equiv \mu_\theta(\mathbf{x}_1) \cdot \mu_\theta(\mathbf{x}_2) \cdot \ldots \cdot \mu_\theta(\mathbf{x}_m)$ ou, ce qui

est équivalent, qu'elle minimise $\sum_{i=1}^{m} - \log \mu_\theta(\mathbf{x}_i)$. Cette approche implique en particulier que $\mu_\theta(\mathbf{x}_i) > 0$ pour toutes les observations car $- \log 0 = \infty$. C'est le principe du **MLE** = maximum likelihood.

Notons au passage que maximiser cette vraisemblance revient à minimiser la **divergence de Kullback-Leibler** $K(S, \mu_\theta)$, entre la distribution empirique sur S et les prédictions μ_θ du modèle. C'est précisément cette notion qu'il s'agit de remplacer par une autre plus commode… que le lecteur aura certainement deviné !

Générer des objets à grandes dimensions comme des images est une affaire délicate. L'approche classique consistait jusqu'à récemment à compresser ces images **x** vers des description en basse dimension (<50) et à appliquer directement le MLE sur ces représentations compressées.

Une autre approche, plus récente, consiste à utiliser un espace de **variables latentes z**, de faible dimension (<50) que l'on va immerger dans l'espace original des images (de dimension 3'000'000) au moyen d'une fonction $\mathbf{z} \rightarrow \mathbf{x} = f_\theta(\mathbf{z})$ dont on va ajuster les paramètres θ comme l'illustre la figure 13 :

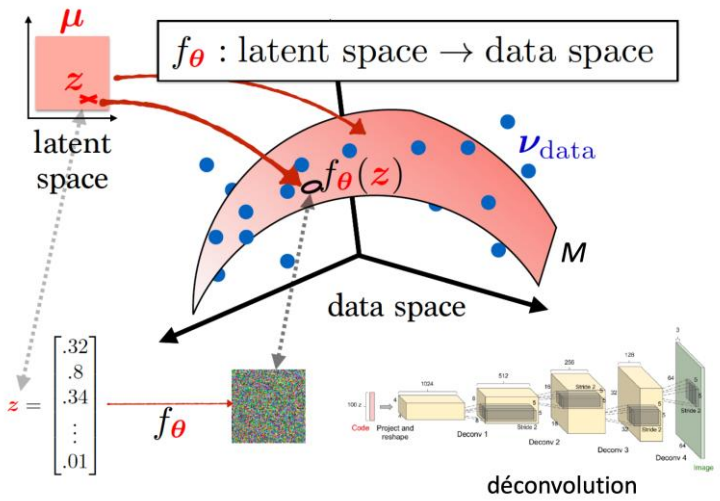


*Figure 13 : L'embedding d'un espace latent '**z**' de faible dimension dans l'espace original de grande dimension au moyen d'une fonction f_θ peut être implémentée par exemple par un réseau de déconvolution de paramètres θ* – source [AOT].

La surface incurvée rouge M dans la figure 13 symbolise l'**embedding** de l'espace latent à faible dimension dans l'espace original à grande dimension. La distribution μ dans l'espace latent '**z**' induit, via f_θ, une distribution, qu'on note $f_{\theta\,\#}\,\mu$, et dont le support coïncide avec M. Les points bleus sont les images dans $S = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_m\}$. Comment alors évaluer l'écart entre l'immersion M et les données S ? Plusieurs possibilités pour cela :

1. Utiliser la **KL-divergence** entre les données S et la distribution $f_{\theta\,\#}\,\mu$ sur M induite par l'embedding f_θ. Cette tentative est cependant immédiatement **vouée à l'échec** puisqu'en dehors de M cette distribution vaut zéro ce qui est interdit !

2. Les **GAN** proposent une méthode originale pour mesurer l'écart entre M et S. Imaginons que les points sur M soit étiquetés d'un label « *on* » et les points de S d'un label « *off* » et que nous entraînions ensuite un classifieur à prédire ces labels. Si le classifieur parvient à prédire ces labels avec une bonne précision cela signifie que M et S sont dissemblables. Le meilleur embedding f_θ sera donc celui qui parvient à duper ce classifieur avec le plus d'efficacité ! L'embedding f_θ et le classifieur sont tous deux des RN profonds dans les GAN.

3. Enfin, puisque la distance KL ne convient pas il faut trouver une autre notion, plus flexible. La distance de Wasserstein $W_p(S, f_{\theta\#}\mu)$ fait parfaitement l'affaire puisqu'elle n'exige rien des supports des mesures qu'elle compare. On peut alors chercher les paramètres θ qui minimise $W_p(S, f_{\theta\#}\mu)$. Effectuer ce calcul demande toutefois de savoir calculer non seulement la valeur mais également le **gradient** $\nabla_\theta W_p(S, f_{\theta\#}\mu)$, nous y reviendrons dans la section 5.

Un nouveau regard sur la généralisation en ML supervisé

Pour bien comprendre l'apport du TO au ML supervisé commençons par rappeler la formalisation du problème de l'apprentissage supervisé[3] [UML]. On suppose qu'un échantillon $S = \{(\mathbf{x}_1,y_1),\dots,(\mathbf{x}_m,y_m)\}$ est constitué par m tirages indépendants d'une même loi de probabilité conjointe μ sur des couples de features $\mathbf{x}$ et de labels y. On choisit une fonction de coût $L(y_{\text{prédiction}}, y_{\text{observé}})$ qui quantifie l'écart entre une prédiction $y_{\text{prédiction}}$ et une valeur observée $y_{\text{observé}}$. L'objectif est alors de trouver un prédicteur f parmi une classe H d'hypothèses (régression linéaire, SVM, arbre de décision, CNN, LSTM, etc…) qui fasse peu d'erreurs en moyenne sur μ (noté $E_\mu[.]$ ci-après). C'est une **approche discriminante** du ML : on ne se préoccupe pas d'approximer la distribution inconnue μ mais juste de trouver un bon prédicteur f.

Idéalement on souhaiterait bien sûr trouver $f \in H$ qui minimise l'erreur moyenne $\text{Err}_\mu[f] \equiv E_\mu[L(y,f(\mathbf{x}))]$ mais, hélas, μ est inconnue. En pratique on se résout donc à calculer l'erreur empirique $\text{Err}_S[f] \equiv 1/m \sum_{i=1}^{m} L(y_i,f(\mathbf{x}_i))$ sur l'échantillon S que l'on possède.

L'**erreur de généralisation** $G[f] \equiv |\text{Err}_\mu[f] - \text{Err}_S[f]|$ est l'écart entre les deux. Pour garantir que $\text{Err}_\mu[f]$ soit petite il faut donc trouver un prédicteur f tel que la somme $\text{Err}_\mu[f] + G[f]$ soit petite. La stratégie élémentaire de **minimisation du risque empirique** (MRE) ne fonctionne pas si l'erreur de généralisation $G[f]$ est grande (on fait face alors au **surapprentissage**), c'est donc elle qu'il faut parvenir à contrôler mathématiquement.

Le TO permet de formuler une version améliorée du MLE. Au lieu de chercher

$$\min_{f\in H} \text{Err}_S[f]$$

avec le risque de faire face au surapprentissage car on se focalise sur un seul train set S, on examine la pire erreur que l'on puisse faire en considérant toutes les distributions μ dont la distance de Wasserstein à la distribution empirique S n'est pas trop grande :

$$\min_{f\in H} \max_{\mu:W_p(\mu,S)} \text{Err}_\mu[f]$$

Il s'agit d'une nouvelle forme de **régularisation** plus robuste que la procédure MRE.

On peut montrer qu'elle offre des garanties sur l'erreur de généralisation. Dans beaucoup de situation f est effectivement calculable [APT].

[3] Cette formalisation s'appelle le Agnostic PAC Learning Model.

4. Focus sur l'adaptation de domaine non supervisée

Dans sa formulation élémentaire le machine learning supervisé présuppose que l'ensemble d'entraînement $S = \{(\mathbf{x}_1,y_1),\ldots,(\mathbf{x}_m,y_m)\}$ utilisé pour trouver un classifieur $f(\mathbf{x})$ est construit à partir d'une échantillonnage i.i.d[4] $(\mathbf{x}_i,y_i)$ selon une loi $\mu_{source}(\mathbf{x},y)$ et que les échantillons $(\mathbf{x}_i',y_i')$ de la cible sur lesquels on va utiliser f sont eux aussi distribués selon cette même loi μ_{source}. Dit autrement, on suppose que le processus génératif des observations est le même pour la source et la cible.

Cependant, les data scientists le savent bien, la réalité est généralement moins souriante que ce cas d'école. Bien souvent les échantillons de la cible sont distribués selon une loi μ_{cible} différente de μ_{source}.

> L'**adaptation de domaine** (AD) est un ensemble de techniques qui permettent de construire un prédicteur destiné à faire des prédictions sur une population cible différente de celle dont sont issus les échantillons d'entraînement.

On distingue deux variantes de l'AD. La première, l'**AD semi-supervisée**, présuppose que l'on dispose de quelques échantillons $(\mathbf{x}_j', y_j')$ de la population cible pour laquelle on connaît les labels y_j' et dont on va chercher à tirer profit pour trouver un « bon » classifieur f. L'**AD non-supervisée**, présuppose qu'on ne dispose d'aucune étiquette dans la population cible. C'est le cas qui nous intéresse ici et auquel on pourra appliquer des techniques de TO.

Des cas particuliers fréquents de DA non-supervisés sont identifiés par le termes suivants :

- Le **déséquilibre de classes** (« class imbalance »). C'est une situation que l'on rencontre par exemple dans le cadre d'une détection d'**anomalies**. Par définition, les anomalies dans la population cible sont rares. Dans l'ensemble d'entraînement en revanche on dispose parfois d'autant d'exemples normaux que d'anomalies. Les distributions marginales sur les labels sont donc très différentes dans les deux domaines : $\mu_{source}(y) \neq \mu_{cible}(y)$. Les distributions des features $\mathbf{x}$ conditionnées sur les labels y sont en revanche identiques $\mu_{source}(\mathbf{x}|y) = \mu_{cible}(\mathbf{x}|y)$.

- Le « **covariate shift** ». Dans ce cas la dépendance de l'étiquette y en fonction des features x est la même dans la cible et dans l'ensemble d'entraînement si bien que $\mu_{source}(y|\mathbf{x}) = \mu_{cible}(y|\mathbf{x})$. En revanche les distributions des features diffèrent $\mu_{source}(\mathbf{x}) \neq \mu_{cible}(\mathbf{x})$. La figure 14 illustre la situation.

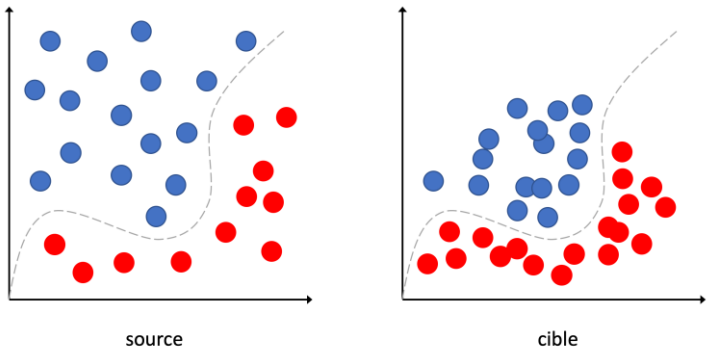


Figure 14 : Dans le cas d'un « covariate shift » les distributions de probabilité marginales des features $\mathbf{x}$ *diffèrent entre la source et la cible. La fonction de prédiction optimale* $f(\mathbf{x})$ *représentée par la courbe en pointillé est la même.*

[4] i.i.d = indépendantes et identiquement distribuées.

La réalité est souvent plus complexe et ne correspond alors à aucun des deux cas précédents. La situation que nous envisageons dans cette section 4 et à laquelle nous appliquerons différentes techniques basées sur le TO est celle dans laquelle on suppose que le domaine cible est obtenu par une **déformation** $\mathbf{x} \rightarrow T(\mathbf{x})$ inconnue du domaine source mais où les labels y rattachés aux points reliés par T sont identiques. La figure 15 illustre la situation.

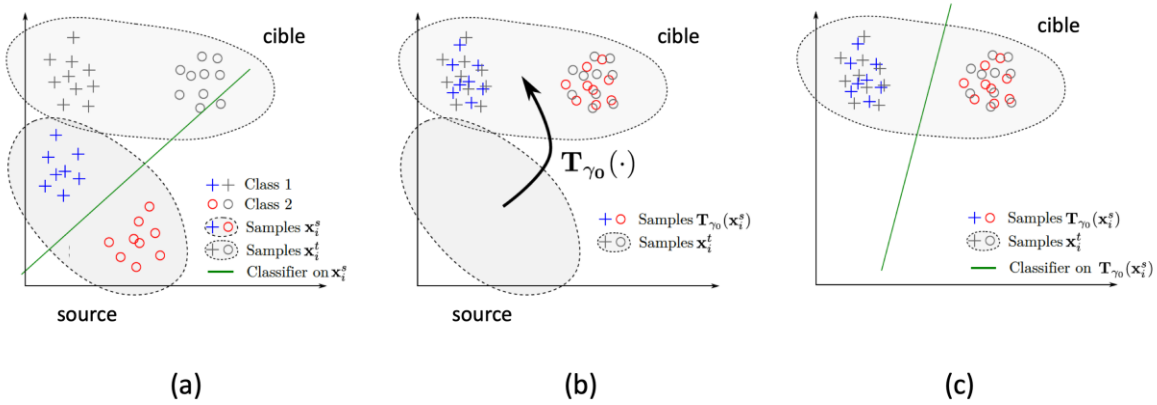


Figure 15 : (a) la distribution des feature dans le domaine source sur lequel on connaît les étiquettes est différente de la distribution des features dans le domaine cible, (b) on calcule un plan de TO γ_0 pour déplacer les points étiquetés dans le domaine cible (c) on entraîne un classifieur sur ces points étiquetés transportés – source [ODA].

Pour illustrer concrètement une telle situation imaginons que la source et la cible sont constituées d'images, les features **x** étant les intensités RVB des pixels de ces images et les y des étiquettes de classification. La figure 15 pourrait correspondre, par exemple, à une situation où les images sources auraient une dominante jaunâtre et ne seraient pas représentatives des images cibles qu'on supposerait équilibrées.

Plus formellement et en utilisant les notations de la section 2, la figure 15 décrit une situation où :

1. $\mu_{cible}(\mathbf{x}) = (T_{\#}\,\mu_{source})(\mathbf{x})$ qui signifie que la marginale $\mu_{cible}(\mathbf{x})$ s'obtient par transport de la marginale $\mu_{source}(\mathbf{x})$ avec un mapping T qu'il s'agit d'optimiser.

2. $\mu_{cible}(y\,|\,T(\mathbf{x})) = \mu_{source}(y\,|\,\mathbf{x})$ qui signifie que l'étiquetage reste inchangé sur les points mis en relation par T dans la source et dans la cible.

Dans les 3 sous-sections qui suivent nous envisagerons successivement 3 techniques basées sur le TO [ODA, JDA, DJT], de plus en plus élaborées, pour construire un prédicteur efficace sur la cible.

Adaptation du domaine des features avec le TO

De prime abord, construire un prédicteur $f(x)$ adapté à la cible au moyen du TO est très intuitif :

1. Trouver l'application T qui transporte l'espace de features décrit par $\mu_{source}(\mathbf{x})$ vers $\mu_{cible}(\mathbf{x}')$ avec le « **moins d'effort** » possible au sens TO du terme. Les distributions $\mu_{source}(\mathbf{x})$ et $\mu_{cible}(\mathbf{x}')$ jouent donc ici le rôle des distributions α et β de la section 2.

2. Utiliser ce mapping T pour transporter les observations de la source, munis de leurs labels, vers le domaine cible.

3. Entraîner un classifieur *f* sur les points étiquetés ainsi transportés dans le domaine cible.

Se pose la question de ce qu'on entend par « moindre effort » ou, en d'autres termes, quelle est le coût $C(\mathbf{x},\mathbf{x}')$ du transport d'un point $\mathbf{x}$ de la source vers la cible $\mathbf{x}'$? L'expérience [ODA] montre que le carré de la distance euclidienne $||\mathbf{x}-\mathbf{x}'||_2^2$ donne de bons résultats en pratique. C'est le cas qui définit la distance de Wasserstein W_2 avec p=2.

Comme nous l'avons expliqué dans la section 2 on ne cherche par un mapping T mais plutôt un **plan de transport optimal** γ. Une fois ce plan γ trouvé, reste encore à transporter les points $\mathbf{x}_j$ du domaine source vers la cible. Des formules plus ou moins explicites existent qui permettent de calculer le résultat de cette opération. Dans le cas où p=2 et lorsque μ_{source} et μ_{cible} correspondent toutes deux à des distributions empiriques[5] la formule est particulièrement simple. Si $\mathbf{X}_s$ désigne la matrice dont les lignes sont les vecteurs $\mathbf{x}_i$ du train set, $\mathbf{X}_s'$ la matrice des vecteurs transportés, n_t le nombre de points dans le domaine cible et $\gamma = (\gamma_{ij})$ avec $\gamma_{ij} = \gamma(\mathbf{x}_i, \mathbf{x}_j')$ est la matrice du plan de TO alors :

$$\mathbf{X}_s' = n_t\, \gamma^T\, \mathbf{X}_s$$

Pour l'instant notre plan de TO γ ne prend pas en compte l'information des labels disponibles dans la source. Intuitivement nous souhaiterions cependant privilégier les plans de transport γ pour lesquels chaque point de la cible ne reçoit que des contributions de points source ayant le *même* label. On peut implémenter cette contrainte en ajoutant au coût $C_{transport}[\gamma]$ du transport γ définit par :

$$C_{transport}[\gamma] = \sum_{ij} C(\mathbf{x}_i, \mathbf{x}_j')\, \gamma(\mathbf{x}_i, \mathbf{x}_j')$$

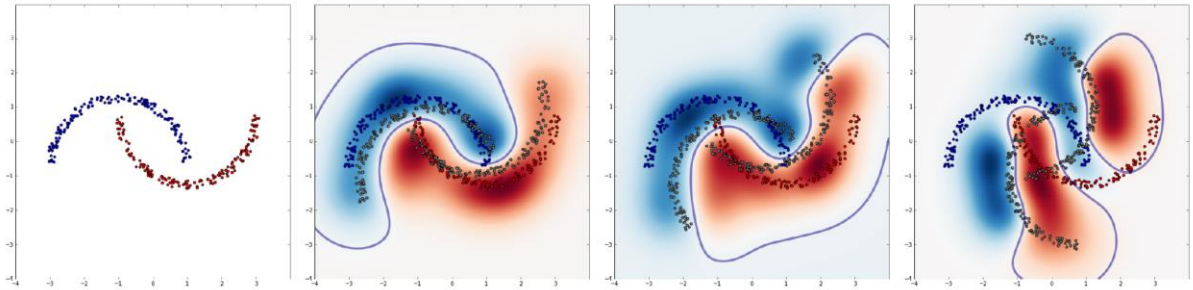
un terme $\Omega_c[\gamma]$ qui pénalise les plans γ qui transportent des points sources $\mathbf{x}_i$ avec beaucoup de labels y_i différents vers un même point cible $\mathbf{x}_j'$. Si $I_K = \{i_1, i_2, \ldots\}$ désigne les indices des points $\mathbf{x}_i$ de la source qui ont le label K et si l'on note $\gamma(I_K, \mathbf{x}_j')$ le vecteur formé des valeurs de $\gamma(\mathbf{x}_i, \mathbf{x}_j')$ lorsque i parcourt I_k on peut voir que le Ω_c suivant fait l'affaire[6] :

$$\Omega_c[\gamma] = \sum_j \sum_K ||\, \gamma(I_K, \mathbf{x}_j')\, ||_2$$

Minimiser la **fonction de coût total** $C_{total}[\gamma] = C_{transport}[\gamma] + \Omega_c[\gamma]$ conduit au résultat souhaité. Les auteurs dans [ODA] ont appliqué cette méthode aussi bien à des données synthétiques qu'à des data set réels et ont obtenus des résultats dépassant l'état de l'art (figure 16).

[5] Ce qui signifie que μ_{source} et μ_{cible} sont des sommes finies de distributions ponctuelles avec des poids identiques.

[6] Régularisation de groupe lasso



(a) source domain (b) rotation=20° (c) rotation=40° (d) rotation=90°

Target rotation angle	10°	20°	30°	40°	50°	70°	90°
SVM (no adapt.)	0	0.104	0.24	0.312	0.4	0.764	0.828
DASVM [9]	0	0	0.259	0.284	0.334	0.747	0.82
PBDA [22]	0	0.094	0.103	0.225	0.412	0.626	0.687
OT-exact	0	0.028	0.065	0.109	0.206	0.394	0.507
OT-IT	0	0.007	0.054	0.102	0.221	0.398	0.508
OT-GL	0	0	0	0.013	0.196	0.378	0.508
OT-Laplace	0	0	0.004	0.062	0.201	0.402	0.524

Figure 16 : (a) représente un domaine source constitué de deux demi-lunes, chacune étant associée à un label « rouge » ou « bleu ». (b),(c),(d) illustrent des domaines cibles obtenus par rotations successives d'angles croissants. La ligne en surbrillance du tableau compare le taux d'erreur moyen de la technique TO décrite ici lorsqu'elle est associée à un classifieur SVM avec d'autres méthodes d'AD. On constate que pour des angles de rotations compris entre 0 et 40° l'erreur est obtenue avec TO est presque nulle et inférieure à celles obtenues par d'autres méthodes (3 premières lignes) – source [ODA].

Apprentissage direct d'un classifieur avec le TO

La méthode que nous venons de décrire procède en deux étapes :

1. On cherche un plan de transport optimal γ qui transporte le domaine des features vers le domaine cible.

2. On entraîne un classifieur f sur le domaine cible en utilisant les données sources étiquetées transportées par γ.

La deuxième application du TO à l'AD propose d'apprendre le classifieur f en une seule étape. Pour cela, au lieu de comparer à l'aide du TO les deux distributions **marginales** $\mu_{source}(\mathbf{x})$ et $\mu_{cible}(\mathbf{x}')$ sur les features dans la source et la cible, on va comparer plutôt les distributions **conjointes** $\mu_{source}(\mathbf{x},y)$ et $\mu_{cible}(\mathbf{x}',y')$ sur les features *et* les labels dans les deux domaines. Utilisons l'acronyme *JDOT* (Joint Distribution OT) pour désigner cette approche. Souvenons-nous cependant que nous ne connaissons pas les labels sur la cible ! Ces labels inconnus sont précisément ce que doit prédire le classifieur f que nous souhaitons construire sur le domaine cible. Lorsque f est un bon classifieur on peut cependant approximer y_i', qui est inconnu, par sa prédiction $f(\mathbf{x}_i')$. Pour chaque classifieur candidat f on calcule alors la distance de Wasserstein (p=1 dans [JDA]) entre la distribution conjointe (empirique) sur la source :

$$\mu_{source}(\mathbf{x},y) = \sum_i \delta(\mathbf{x} - \mathbf{x}_i')\,\delta(y - y_i')$$

et la distribution conjointe (empirique) estimée avec f sur la cible :

$$\mu^{f}_{\text{cible}}(\mathbf{x},y) = \sum_j \delta(\mathbf{x} - \mathbf{x}_j') \, \delta(y - f(\mathbf{x}_j'))$$

Pour la fonction $C((\mathbf{x},y), (\mathbf{x}',y'))$ qui définit le coût du transport les auteurs de [JDA] suggèrent d'additionner la distance $d(\mathbf{x},\mathbf{x}')$ entre les features et une mesure $L(y,y')$ de l'écart entre les labels (comme l'entropie croisée par exemple).

$$C((\mathbf{x},y), (\mathbf{x}',y')) = d(\mathbf{x}, \mathbf{x}') + \lambda \, L(y, y')$$

Le plan de TO γ entre μ_{source} et μ^{f}_{cible} définit une distance $W_1(\mu_{\text{source}}, \mu^{f}_{\text{cible}})$ qui dépend donc de f. Du coup, le meilleur classifieur f sera celui qui la minimise. Le prix à payer pour obtenir f en **une seule étape** est donc la minimisation d'un coût de transport simultanément sur le plan γ et sur le classifieur f. Le classifieur ainsi obtenu jouit de propriétés théoriques qui garantissent sa qualité [JDA] chose qui n'était pas vraie dans l'approche du TO appliquée à l'AD des features uniquement.

Les expérimentations dans [JDA] démontrent par ailleurs la supériorité expérimentale de cette deuxième approche.

Le TO et le Deep Learning font bon ménage

Les deux méthodes décrites précédemment se heurtent à deux difficultés. La première est qu'elles ne permettent pas de **passer à l'échelle** car elles exigent de calculer un plan de transport exact γ de dimension $n_{\text{source}} \times n_{\text{cible}}$ où n_{source} et n_{cible} sont les nombres d'observations dans les deux domaines. La seconde difficulté tient au fait que le TO définit par γ opère directement sur les données brutes dans la source et dans la cible. Pour des objets complexes comme des images ces représentations en pixels sont peu adaptées à une comparaison utile. Il serait plus judicieux de faire opérer le TO sur des **représentations sémantiquement** plus riches.

L'architecture proposée dans [DJT] propose une solution à ces deux difficultés pour un classifieur d'images. Elle reprend le principe de *JDOT* qui consiste à minimiser le coût de transport entre les distributions conjointes à la fois sur le plan γ et sur le classifieur f. Mais plutôt que de faire opérer le TO directement sur les données brutes $\mathbf{x}$ et $\mathbf{x}'$, on va l'appliquer à une représentation sémantiquement plus riche $\mathbf{z}$ et $\mathbf{z}'$ que l'on extrait d'une couche profonde d'un CNN pour trouver des représentations $\mathbf{z} = g(\mathbf{x})$ et $\mathbf{z}' = g(\mathbf{x}')$ sémantiquement comparables. La figure 17 illustre cette architecture. Cette approche, que l'on désignera par le sigle *DeepJDOT*, cherche donc à minimiser le coût d'un TO γ entre la distribution empirique des observations sources $(g(\mathbf{x}_i), y_i)$ et celle de la cible $(g(\mathbf{x}_i'), f(g(\mathbf{x}_i')))$ en faisant varier à la fois le classifieur f et la représentation g des données.

En résumé :

- Dans *JDOT* on minimise le coût du transport en faisant varier simultanément le plan γ et le prédicteur f.
- Dans *DeepJDOT* on minimise le coût du transport en faisant varier simultanément le plan γ, le prédicteur f et l'embedding g.

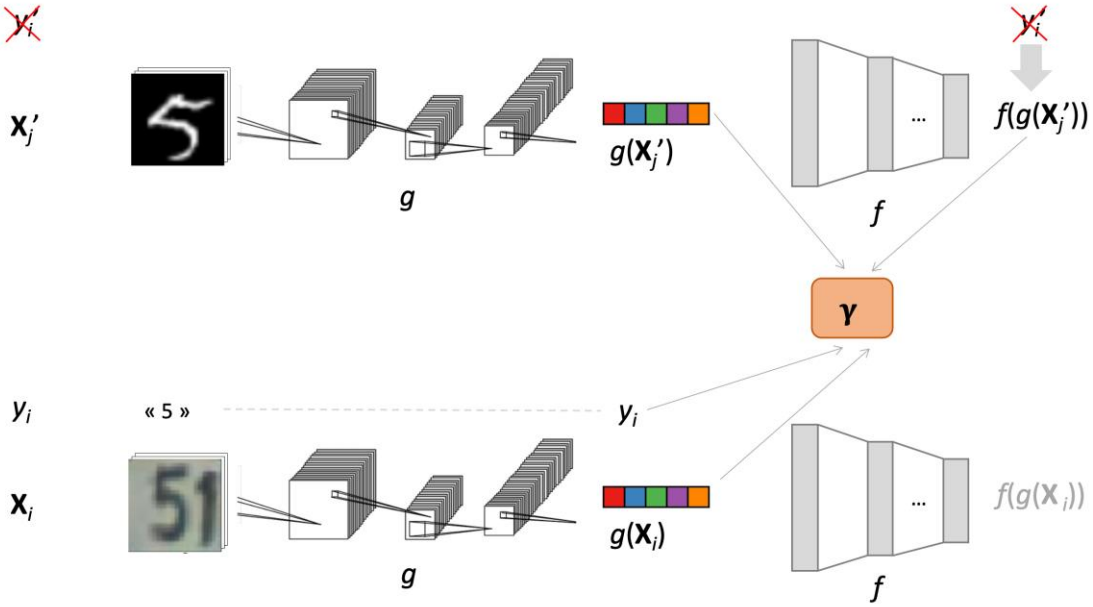


Figure 17 : L'architecture DeepJDOT cherche à minimiser le coût du transport, via un plan γ, de la distribution empirique basée sur les $(g(\mathbf{x}_i), y_i)$ *vers celle basée sur les* $(g(\mathbf{x}_i'), f(g(\mathbf{x}_j')))$. *Pour cela on fait varier simultanément le plan* γ, la représentation des données g et le classifieur f – d'après [DJT].

Pour résoudre le problème de la scalabilité *DeepJDOT* propose une approche par mini-batch. On sélectionne des batchs d'observations choisies au hasard dans la source et la cible. On calcule le minimum de la fonction de coût sur le triplet γ, f et g par itérations sur ces batchs en alternant deux étapes :

(1) Pour f et g fixés on cherche un plan γ approximatif (avec un algorithme de TO approprié).

(2) Pour un plan γ fixé on cherche f et g par DSG.

Les résultats expérimentaux obtenus avec *DeepJDOT* dépassent l'état de l'art dans des applications comme la classification de digits appartenant à des domaines différents.

5. Régularisation entropique et algorithme de Sinkhorn

La découverte récente d'algorithmes performants pour calculer un plan de TO explique le foisonnement d'applications mentionné dans l'introduction. Sans entrer dans des détails trop techniques [COT] cette section présente rapidement l'une des idées centrales de ces avancées : la **régularisation entropique** et l'**algorithme de Sinkhorn**.

Dans sa formulation élémentaire le calcul d'un plan optimal γ entre deux distributions α et β avec une fonction de coût C implique de chercher :

$$\text{minimum de } \sum_{ij} \gamma_{ij} C_{ij} \text{ parmi tous les } \gamma \text{ dans } U(\alpha,\beta)$$

où $U(\alpha,\beta)$ désigne l'ensemble des distributions conjointes γ ayant α, β pour marginales. La fonction à minimiser est linéaire de même que les contraintes d'appartenance de γ à $U(\alpha,\beta)$ comme on peut le voir facilement. Il s'agit par conséquent d'un problème de **programmation linéaire** (PL) pour lequel il existe des algorithmes classiques.

Dans un contexte data science on fait face à deux problèmes :

1. La **complexité** prohibitive des algorithmes de PL, de l'ordre de $O(n^3 \log n)$ où n désigne le nombre de points supports dans α et β, interdit leur utilisation avec des grands data sets.

2. En règle générale il n'existe **pas de solution unique** au problème. Les solutions optimales γ^* sont des points extrémaux de $U(\alpha,\beta)$ dont les bords sont « plats » comme le montre la figure 18. Pour cette raison γ^* est instable vis-à-vis de petites modifications de α, de β ou de C. Ceci empêche en particulier de calculer le gradient du coût de transport dans une DGS.

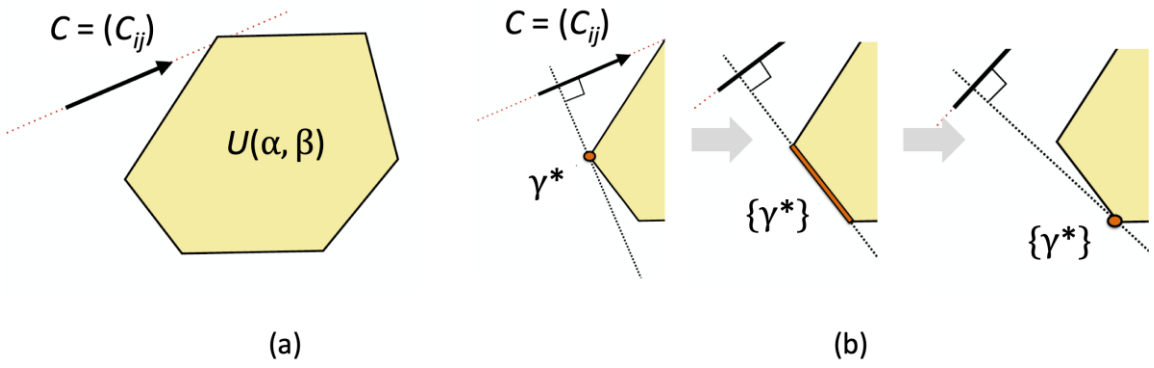


Figure 18 : La solution optimale γ^* est située sur le bord du domaine convexe $U(\alpha,\beta)$ qui est linéaire par morceaux. Elle varie donc de manière discontinue lorsque α, β ou C varient ce qui empêche le calcul d'un gradient du coût du TO dans une DSG – source [AOT].

Une solution au point 2 consiste à **régulariser** le problème en additionnant un petit terme supplémentaire convexe au coût de transport pour faire en sorte que les variations de la solution optimale γ^* soient plus

régulières. Un terme qui fait l'affaire est la négation de l'entropie[7] de $H[\gamma]$. Très schématiquement, $H[\gamma]$ est grande lorsque γ est très dispersée (ou floue si l'on préfère).

$$\text{minimum de } \sum_{ij} \gamma_{ij} C_{ij} - \varepsilon H[\gamma] \quad \text{parmi tous les } \gamma \text{ dans } U(\alpha,\beta) \qquad (*)$$

Plus ε augmente, plus on a intérêt à choisir un γ dispersé comme l'illustre la figure 19.

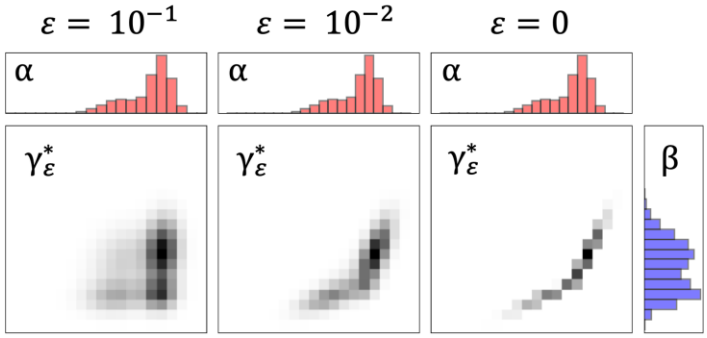


Figure 19 : Lorsque $\varepsilon = 0$ *la solution* γ^* est proche d'une courbe de Monge $\mathbf{x} \rightarrow T(\mathbf{x})$. A mesure que $\varepsilon > 0$ croit $\gamma^*(\varepsilon)$ devient de plus en plus « floue » – source [OTM].

Géométriquement, lorsque $\varepsilon > 0$ le point optimal γ^* sera un point intérieur à $U(\alpha,\beta)$ et non plus un point extrémal du bord comme l'illustre la figure 20.

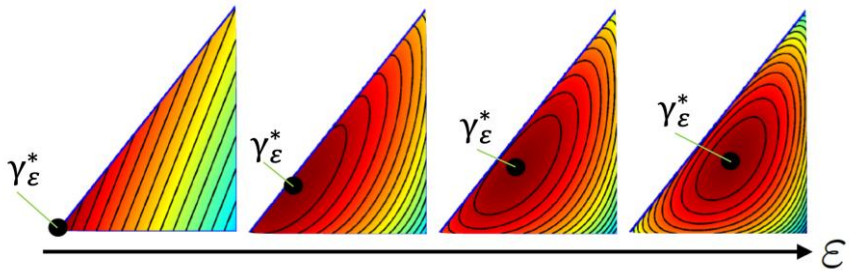


Figure 20 : Lorsque l'on addition un petit terme entropique $-\varepsilon H[\gamma]$ *au coût de transport, le plan de TO* γ^* *migre vers l'intérieur du domaine* $U(\alpha,\beta)$ – source [COT]

Le calcul du minimum (*) se fait de manière classique en annulant le gradient de la fonction de Lagrange construite à partir des contraintes $\gamma^*(\varepsilon) \in U(\alpha,\beta)$ [COT]. Le calcul est élémentaire et le résultat pour $\gamma^*(\varepsilon)$ s'exprime de manière compacte à l'aide de deux vecteurs u et v comme :

$$\gamma^*_{ij}(\varepsilon) = u^T_i K_{ij} v_j \quad \text{où} \quad K_{ij} \equiv \exp[-C_{ij}/\varepsilon]$$

[7] L'entropie de Shannon usuelle.

On trouve alors deux équations pour u et v en substituant l'expression précédente dans les équations qui définissent les contraintes $\gamma^*(\varepsilon) \in U(\alpha,\beta)$:

$$u_i = \alpha_i / (Kv)_i$$
$$v_j = \beta_j / (K^T u)_j$$

Ces deux équations non-linéaires pour u et v peuvent alors se résoudre approximativement par itérations successives :

$$u_i^{(n+1)} = \alpha_i / (Kv^{(n)})_i$$
$$v_j^{(n+1)} = \beta_j / (K^T u^{(n)})_j$$

C'est l'**algorithme de Sinkhorn** pour trouver le plan de TO optimal régularisé $\gamma^*(\varepsilon)$. Cet algorithme se prête bien à la parallélisation sur GPU. En pratique différentes astuces permettent de réduire la complexité de cette algorithme régularisé à $O(n \log n)$ ce qui le rend désormais utilisable pour calcul un TO sur de grands ensembles de données.

6. Les nouvelles mathématiques du Machine Learning ?

La notion de distance de Wasserstein qui caractérise le coût d'un TO entre deux distributions de probabilités se distingue d'autres notions de dissemblance en ce qu'elle permet une comparaison géométrique entre distributions dont les supports ne coïncident pas. Cette observation ainsi que l'ubiquité des distributions de probabilités en Machine Learning font du TO un outil polyvalent pour attaquer certains des problèmes réputés difficiles du Machine Learning. Rappelons-en quelques-uns :

- L'**adaptation de domaine** : Comment entraîner un algorithme d'apprentissage lorsque les données d'entraînement dont on dispose ne sont pas représentatives de la cible sur laquelle on souhaite déployer le prédicteur ?

- Les **garanties de généralisation** : Comment définir une procédure d'entraînement qui apporte des garanties sur les capacités de généralisation d'un algorithme ?

- Les **modèles génératifs** : Comment construire des modèles génératifs dans des espaces à très grandes dimensions par embedding d'un espace latent ?

- La **classification multilabel** : Comment définir une notion pertinente de métrique pour un modèle de classification multilabel basé sur une distance sémantique entre étiquettes ?

Il se pourrait même qu'un jour prochain la théorie du TO soit en mesure d'apporter quelques lumières sur les capacités de généralisation encore mystérieuses des réseaux de neurones profonds [OTV].

Longtemps cantonné à la recherche en mathématiques [VIL], le TO a fait aujourd'hui une percée dans les applications du Machine Learning, grâce notamment à la mise au point d'algorithmes performants. Le caractère un peu abstrait de la théorie de TO explique sans doute qu'elle reste encore méconnue dans les cercles des data scientists. Néanmoins il faut ici se réjouir du fait que les experts du TO aient développé une **tradition d'excellence** s'agissant de la qualité des articles qu'ils rédigent, en témoigne par exemple l'ouvrage remarquable « *Computational Optimal Transport* » [COT]. Les data scientist auraient donc tort d'ignorer ce domaine fascinant des mathématiques appliquées, d'autant plus que des API's [POT] existent pour passer de la théorie à la pratique.

Peut-on affirmer pour autant que la théorie du Transport Optimal constitue désormais les **nouvelles mathématiques** du Machine Learning comme d'aucuns le prétendent [NMD] ? Il est probablement prématuré pour l'affirmer car la recherche dans ce domaine est très active et n'a vraisemblablement encore livré ni la pleine mesure du potentiel du TO, ni toutes ses limitations.

L'air du temps en data science est plutôt au **concret** à tous crins, aux perfectionnements des outils et aux « hacks » plutôt qu'à la recherche de grandes idées. Une telle attitude est en partie justifiée par les succès pratiques incontestables du Deep Learning, une technique qui n'a certes pas attendu d'être assise sur des fondements théoriques robustes pour faire preuve de son utilité. Pourtant, venue de la nuit des temps, la théorie du TO illustre de manière très vivante que la quête des idées universelles n'est pas forcément antinomique avec l'efficacité. Bien au contraire, le coût, somme toute modeste, que l'on paie en **abstraction** pour la maîtriser est à l'origine même de la diversité des contextes où elle trouve ses applications.

Références

- [AOT] **Scalable bayes via barycenters of subset posteriors**, *S. Srivastava, V. Cevher, Q. Dinh, and D. Dunson*, Artificial Intelligence and Statistics, pages 912–920, 2015, http://proceedings.mlr.press/v38/srivastava15.html

- [OTM] **Optimal transport for machine learning**, *R. Flamary*], AG GDR ISIS, Sète – novembre 2017, https://remi.flamary.com/pres/OTML_ISIS_2017.pdf

- [OTA] **Optimal transport for applied mathematicians**, *F. Santambrogio* Birkhauser – 2015 http://www.math.toronto.edu/mccann/assignments/477/Santambrogio15.pdf

- [DCL] **Optimal transport for documents classification**, blog LumenAI, http://www.lumenai.fr/blog/optimal-transport-for-documents-classification

- [GAN] **Improving GANs Using Optimal Transport**, *T. Salimans, H. Zhang, A. Radford, D. Metaxas* – mars 2018 https://arxiv.org/abs/1803.05573

- [OTV] **An Optimal Transport View on Generalization**, *J. Zhang, T. Liu, D. Tao* – novembre 2018, arxiv.org/abs/1811.03270 [stat.ML]

- [NVD] **GAN 2.0: NVIDIA's Hyperrealistic Face Generator**, *Synced Review*, Medium – décembre 2018, https://medium.com/syncedreview/gan-2-0-nvidias-hyperrealistic-face-generator-e3439d33ebaf

- [COT] **Computational Optimal Transport**, *G. Peyré, M. Cuturi* – février 2019, arxiv.org/abs/1803.00567 [stat.ML]

- [DDD] **Data-Driven Distributionally Robust Optimization Using the Wasserstein Metric: Performance Guarantees and Tractable Reformulations**, *P. Mohajerin Esfahabi, D. Kuhn* – juin 2017, arxiv.org/abs/1505.05116 [math.OC]

- [DJT] **DeepJDOT: Deep Joint Distribution Optimal Transport for Unsupervised Domain Adaptation**, *B. Bhushan Damodaran, B. Kellenberger_, R_. Flamary, D. Tuia, N. Courty* – septembre 2018, arxiv.org/abs/1803.10081 [cs.CV]

- [JDA] **Joint distribution optimal transportation for domain adaptation**, *N. Courty, R. Flamary, A. Habrard, A. Rakotomamonjy* – NIPS 2017, arxiv.org/abs/1705.08848 [stat.ML]

- [LWL] **Learning with a Wasserstein Loss**, *C. Frogner, C. Zhang, H. Mobahi, M. Araya-Polo, T. Poggio* – décembre 2015, arxiv.org/abs/1506.05439 [cs.LG]

- [ODA] **Optimal Transport for Domain Adaptation**, *N. Courty, R. Flamary, Devis Tuia, A. Rakotomamonjy* – juin 2016, arxiv.org/abs/1507.00504 [cs.LG]

- [VIL] **Optimal transport, old and new**, *C. Villani*, Springer-Verlag Berlin and Heidelberg GmbH & Co – 2009 https://cedricvillani.org/wp-content/uploads/2012/08/preprint-1.pdf

- [APT] **A Primer on Optimal Transport**, *M. Cuturi, J. Solomon* – NIPS 2017 https://media.nips.cc/Conferences/NIPS2017/Eventmedia/nips-2017-marco-cuturi-tutorial.pdf

- [UML] **Understanding Machine Learning**, *S. Shalev-Shwartz, S. Ben-David*, Cambridge University Press – 2014 https://www.cs.huji.ac.il/~shais/UnderstandingMachineLearning/understanding-machine-learning-theory-algorithms.pdf

- [NMD] **Optimal Transport Theory the New Math for Deep Learning**, *C. E. Perez*, Medium – novembre 2018 https://medium.com/intuitionmachine/optimal-transport-theory-the-new-math-for-deep-learning-2520395fc183