Approfondissements d'Optimisation – M1 MA 2020/2021 – Université Paris-Saclay

Le transport optimal numérique

Le rapport (fichier .pdf) et le code (fichier `.ipynb` ou `.py`) sont à rendre avant vendredi `22 janvier à minuit` (heure de Paris).
Le sections 1-3 sont obligatoires. Vous devez choisir entre l'option A et B.

Situons notre contexte à Paris et prenons en considération l'ensemble des boulangeries qui produisent des pains au chocolat (le lecteur peut bien sur choisir la viennoiserie qui lui plaît le plus). Ces derniers doivent être livrés chaque matin aux cafés dans lesquels les clients pourront les goûter. La production et la consommation de pains au chocolat sont décrites par $\mu \in \mathbb{R}_+^N$ et $\nu \in \mathbb{R}_+^N$, respectivement (μ_i donne la quantité de pains au chocolat produite par la boulangerie i). Nous supposons que la quantité totale de la production et de la consommation est le même :

$$\sum_{i=1}^{N} \mu_i = \sum_{j=1}^{N} \nu_j = 1.$$

Le problème du transport optimal consiste alors à transporter la quantité de pains au chocolat produite par la boulangerie $i \in I$ ($I = \{1, \cdots, N\}$ est l'ensemble des boulangeries) à un café $j \in J$ ($J = \{1, \cdots, N\}$ est l'ensemble des cafés) de sorte que le coût de transport $(C)_{ij} := c(x_i, y_j)$, par exemple la distance entre la boulangerie et le café, soit minimal. On a noté x_i (y_j) la position de la boulangerie i (café j). Le problème du transport optimal (connu sous le nom de problème de *Monge-Kantorovich*) consiste à chercher un couplage optimal $\gamma \in \mathbb{R}_+^{N\times N}$ qui nous indique comment la masse en i est répartie sur chaque $j \in J$ (on peut bien imaginer que γ est un camion qui transporte les pains au chocolat de la boulangerie i au café j). Si le couplage optimal γ consiste à assigner le même café j à tous les pains au chocolat produits par la boulangerie i, alors on dit que le couplage est *déterministe* (c'est-à-dire la matrice γ est sparse).
Le problème de Monge-Kantorovich s'écrit sous la forme

$$\min \left\{ \sum_{i=1}^{N} \sum_{j=1}^{N} C_{ij}\gamma_{ij} \mid \gamma \in \Pi(\mu, \nu) \right\}, \tag{$\mathcal{MK}$}$$

où

$$\Pi(\mu, \nu) := \{\gamma \in \mathbb{R}_+^{N\times N} \mid \sum_{j=1}^{N} \gamma_{ij} = \mu_i,\ \forall i \in I \text{ et } \sum_{i=1}^{N} \gamma_{ij} = \nu_j,\ \forall j \in J\}.$$

On remarque que ($\mathcal{MK}$) est bien un problème d'optimisation sous contraintes : de positivité $\gamma_{ij} \geq 0\ \forall (i,j) \in I \times J$ et de marginales $\sum_{j=1}^{N} \gamma_{ij} = \mu_i$, $\sum_{i=1}^{N} \gamma_{ij} = \nu_j$. Le but de ce projet est d'étudier différents algorithmes pour résoudre ($\mathcal{MK}$) et les appliquer pour calculer l'énergie de repulsion entre les électrons.

1 Un problème de programmation linéaire (5 points)

Q1.1. On définit un vecteur $\boldsymbol{c} := (C_{11}, C_{12}, \cdots, C_{1N}, C_{21}, \cdots, C_{N1}, \cdots, C_{NN})$ et un vecteur $\boldsymbol{x} := (\gamma_{11}, \gamma_{12}, \cdots, \gamma_{1N}, \gamma_{21}, \cdots, \gamma_{N1}, \cdots, \gamma_{NN})$.
Soit $\boldsymbol{b}$ un vecteur de taille $2N$ $\boldsymbol{b} := (\mu_1, \cdots, \mu_N, \nu_1, \cdots, \nu_N)$. Determiner une matrice A de taille $2N \times N^2$ telle que le problème $(\mathcal{MK})$ peut s'écrire sous la forme

$$\min \{< \boldsymbol{c}, \boldsymbol{x} > \mid A\boldsymbol{x} = \boldsymbol{b},\ \boldsymbol{x} \geq 0\}. \qquad (\mathcal{P})$$

Q1.2. Programmer une fonction python

`solveOT(mu,nu,C)`

prenant en argument les distributions `mu` et `nu` et la matrice coût `C`. La fonction devra retourner une matrice `gamma` solution de $(\mathcal{MK})$. En particulier on veut résoudre $(\mathcal{MK})$ sous la forme $(\mathcal{P})$ en utilisant l'algorithme du simplex. On utilisera la fonction `scipy.optimize.linprog` avec `method='simplex'`.

Q1.3. Tester la fonction `solveOT` avec $\mu(x) = \chi_{[0,1]}(x)$, $\nu(y) = \exp(-10(y-0.5)^2)$ sur l'interval $[0,1]$ discretisé avec $N = 20$ points (maillage uniform) et $c(x,y) = |x-y|^2$ (ex. $\mu_i = \mu(x_i)$ où $x_i \in [0,1]$). Tracer sur une fenêtre graphique la solution γ. La solution obtenue est-elle déterministe ? (**Indication :** les deux distributions doivent avoir la même masse on pourra les normaliser de la façon suivante $\mu = \frac{\mu}{\sum_i \mu_i}$ et $\nu = \frac{\nu}{\sum_i \nu_i}$)

Q1.4. Montrer que le problème dual de $(\mathcal{MK})$ est donné par

$$\max \left\{ \sum_i u_i \mu_i + \sum_j v_j \nu_j \mid \boldsymbol{u}, \boldsymbol{v} \in \mathbb{R}^N\ u_i + v_j \leq C_{ij}\ \forall (i,j) \in I \times J \right\}, \qquad (\mathcal{MK}_d)$$

où les $\boldsymbol{u}, \boldsymbol{v}$ sont les variables duales (**Remarque :** $\boldsymbol{u}^\star, \boldsymbol{v}^\star$ optimales sont appelés potentiels de Kantorovich). Réécrire $(\mathcal{MK}_d)$ sous la forme de dual de $(\mathcal{P})$. Peut on appliquer l'algorithme du simplex ? Si oui programmer une fonction python `solveOTdual(mu,nu,C)` qui retournera les potentiels de Kantorovich $\boldsymbol{u}^\star, \boldsymbol{v}^\star$. On demandera à la fonction `solveOTdual` de retourner aussi la variable d'écart `s`. Reprendre les expériences effectuées pour le primal et tracer les potentiels dans la même fenêtre graphique. Verifier que $x_i^\star s_i = 0\ \forall i = 1, \cdots, N$ où $x^\star$ est la solution du problème primal en forme vectorielle.

Q1.5.[Permutation optimale] On considère maintenant le cas où $\mu_i = \nu_j = \frac{1}{N}\ \forall i, j$. On peut montrer qu'il existe une permutation optimale $\sigma^\star$ telle que

$$\gamma_{ij}^\star = \begin{cases} 1 & \text{si } j = \sigma^\star(i) \\ 0 & \text{sinon.} \end{cases}$$

Résoudre ce problème de transport optimal avec $N = 20$ et $\{x_i\}_{i\geq 0}$, $\{y_j\}_{j\geq 0}$ générés aléatoirement par la fonction `numpy.random.rand`. Tracer la solution $\gamma^\star$ et verifier graphiquement qu'on a obtenu une matrice de permutation. Soient maintenant $x_i, y_i \in \mathbb{R}^2$ et

$$C_{ij} = c(x_i, y_j) = |x_i^1 - y_j^1|^2 + |x_i^2 - y_j^2|^2.$$

Résoudre le problème de transport avec $N = 20$ et $\{x_i\}_{i\geq 0}$ générés par `numpy.random.rand` et $\{y_j\}_{j\geq 0}$ par `numpy.random.randn`. Tracer la solution γ. Représenter dans le plan xy les points x_i et y_j et afficher la ligne qui relie chaque point i au point $j = \sigma^\star(i)$.

2 Le problème dual et la fonction log-sum-exp (5 points)

Q2.1. On considère la fonction log-sum-exp $f : \mathbb{R}^N \to \mathbb{R}_+$ définie comme

$$f(x) = \log\left(\sum_{i=1}^{N} \exp(x_i)\right).$$

1. Montrer que f est convexe.
2. Montrer l'inégalité suivante

$$\max_i x_i \leq \log\left(\sum_{i=1}^{N} \exp(x_i)\right) \leq \max_i x_i + \log N$$

3. En déduire que $f_\varepsilon(x) = \varepsilon \log\left(\sum_{i=1}^{N} \exp(\frac{x_i}{\varepsilon})\right)$ converge vers $\max_i x_i$ quand $\varepsilon \to 0$.

Q2.2. Si on se focalise sur $(\mathcal{MK}_d)$ on peut remarquer que

$$v_j = \min_i C_{ij} - u_i \quad \forall j \in J.$$

En utilisant la fonction f_ε (**attention :** on a un min et pas un max) écrire une version régularisée de $(\mathcal{MK}_d)$ dans la seule variable $\boldsymbol{u}$ et sans contrainte.

Q2.3. Calculer le gradient du critère.
Programmer alors une fonction python `GradDualReg(mu,nu,C,eps,t)` prenant en argument les distributions `mu`, `nu`, la matrice coût `C`, le paramètre de regularisation `eps` pour f_ε et le pas `t` pour la méthode du gradient à pas fixe. La fonction devra retourner deux vecteurs `u`, `v` contenants les potentiels de Kantorovich (regularisés) obtenus en résolvant le problème régularisé avec la méthode du gradient à pas fixe.

Q2.4. Reprendre les experiences de la question **Q1.3.** et tracer les potentiels pour différentes valeurs de ϵ (ex. $\varepsilon = 1, 0.5, 0.1, 0.05, 0.01, 0.005$). Comparer les résultats avec ceux obtenus en utilisant le simplex et commenter.

3 La régularisation entropique (6 points)

Q3.1. On s'intéresse maintenant à la régularisation du problème primal en pénalisant la contrainte de positivité en ajoutant un terme d'entropie : on va minimiser le critère suivant

$$F(\gamma) := \sum_{ij} C_{ij}\gamma_{ij} + \varepsilon \operatorname{Ent}(\gamma), \tag{1}$$

où $\varepsilon > 0$ est un paramètre de régularisation et $\operatorname{Ent} : \mathbb{R}_+^{N\times N} \to \mathbb{R}$ est l'entropie qu'on définit comme

$$\operatorname{Ent}(\eta) = \sum_{ij} \eta_{ij}(\log(\eta_{ij}) - 1).$$

Montrer que $F(\gamma)$ est convexe et qu'on peut l'écrire sous la forme

$$F(\gamma) = \sum_{ij} \gamma_{ij}\left(\log\left(\frac{\gamma_{ij}}{\overline{\gamma}_{ij}}\right) - 1\right)$$

où $\overline{\gamma}_{ij} = \exp\left(\frac{-C_{ij}}{\varepsilon}\right)$.

Q3.2. Le problème $(\mathcal{MK})$ devient alors

$$\min\left\{F(\gamma) \mid \sum_j \gamma_{ij} = \mu_i,\ \sum_i \gamma_{ij} = \nu_j\right\}. \tag{H}$$

1. Calculer le Lagrangien du problème et les conditions d'optimalité.
2. En déduire que la solution $\gamma^\star$ peut s'écrire comme

   $$\gamma^\star_{ij} = a_i b_j \overline{\gamma}_{ij}\ \forall i, j,$$

   où $a_i = \exp(u_i/\varepsilon)$, $b_j = \exp(v_j/\varepsilon)$ et u_i, v_j sont les multiplicateurs de Lagrange.
3. En imposant que $\gamma^\star$ doit satisfaire les contraintes de marginales, montrer que a_i, b_j sont déterminés par les équations, connues sous le nom de équations de Bernstein-Schrödinger, suivantes

   $$a_i = \frac{\mu_i}{\sum_j b_j \overline{\gamma}_{ij}},\ b_j = \frac{\nu_j}{\sum_i a_i \overline{\gamma}_{ij}}.$$

4. Écrire le problème dual et commenter par rapport au problème régularisé avec la fonction log-sum-exp.

Q3.3. On peut appliquer un algorithme de descente de gradient pour résoudre le problème dual ? Si oui programmer une fonction `EntDual(mu,nu,C,eps)` prenant en argument les distributions `mu`, `nu`, la matrice de coût `C` et le paramètre de regularisation `eps`. La fonction retournera deux vecteurs `u` et `v` et une matrice `gamma` contenant $\gamma^\star$.

Q3.4. Tester la fonction `EntDual(mu,nu,C,eps)` en reprenant les expériences précédentes pour différentes valeurs de ϵ. Tracer $\gamma^\star$ pour ϵ le plus petit possible et comparer les résultats avec ceux obtenus en utilisant la méthode du simplex.

Remarque 1. *si ϵ est trop petit on peut avoir des erreur machines ou pas de convergence.*

Q3.5. Une méthode alternative consiste à construire deux suite $\{a^n\}$ et $\{b^n\}$ en utilisant les équations de Bernstein-Schrödinger. On obtient l'algorithme (connu sous le nom d'algorithme de Sinkhorn) suivant

$$a_i^{n+1} = \frac{\mu_i}{\sum_j b_j^n \overline{\gamma}_{ij}}, \quad b_j^{n+1} = \frac{\nu_j}{\sum_i a_i^{n+1} \overline{\gamma}_{ij}}. \tag{2}$$

Programmer alors une fonction Python `Sinkhorn(mu,nu,C,eps)` prenant en argument les distribution `mu`, `nu`, la matrice coût `C` et le paramètre de regularisation `eps`. La fonction retournera deux vecteurs `u` et `v` et une matrice `gamma` contenant $\gamma^\star$.

Q3.6. Reprendre les experiences précédentes et montrer numériquement que les potentiels $\boldsymbol{u}, \boldsymbol{v}$ obtenus avec Sinkhorn convergent vers les potentiels calculés en utilisant l'algorithme du simplex.

Option A : Le transport multi-marges et l'interaction électron-électron (7 points)

On traite maintenant le cas du problème du transport optimal multi-marges. Ne considérons plus uniquement les boulangeries et les cafés, mais prenons également en compte les hôtels,

les restaurants, etc. et cherchons un couplage $\gamma \in \otimes^K \mathbb{R}_+^N$ (γ est maintenant un tensor K dimensionnel et on notera les composantes comme $\gamma_{i_1,\cdots,i_K}$) nous indiquant la quantité de pains au chocolat envoyée par la boulangerie i_1 au café i_2, à l'hôtel i_3, etc. Cet exemple peut paraître simple, mais nous pouvons remarquer que le transport optimal multi-marges apparaît plus général que le TO classique, car il nous permet de modéliser l'interaction entre les boulangeries et tous les clients possibles, à savoir des cafés, des restaurants, des hôtels, etc. Avoir plusieurs marges augmente évidemment la difficulté du problème et sa résolution est une tâche délicate. On a des solutions présentant une structure surprenante et qui n'ont pas d'equivalent dans le cas du transport à deux marges : on peut avoir un couplage optimal déterministe qui se concentre sur des structures "fractals" ! ! !
Dans le cas où on prend comme coût l'intéraction de Coulomb

$$c(x_1, \cdots, x_K) = \sum_{i \neq j}^{K} w(|x_i - x_j|),$$

avec $w(x) = \dfrac{1}{x}$ et les K marginals $\mu^k(x_{i_k})$ $k = 1, \cdots, K$ sont toutes égales à $\rho(x_i)$ (qui denote la densité électronique : la probabilité de trouver un électron en x_i) alors le problème du transport multi-marge nous donne l'énergie de repulsion entre les électrons et le couplage optimal $\gamma^\star_{i_1,\cdots,i_K}$ la probabilité de trouver l'electron k à la position i_k.

Remarque 2. *Les marginals sont toutes égales à ρ car les electrons sont indistinguables .*

QA.1. Écrire le problème de Monge-Kantorovich et son dual pour le coût de Coulomb et $K = 3$. (les marginals sont maintenant toutes égales !)

QA.2. En procédant de la même façon que la section 3 écrire le problème de Monge-Kantorovich régularisé pour le coût de Coulomb et $K = 3$

1. Calculer le Lagrangien du problème et les conditions d'optimalité.
2. En déduire que la solution $\gamma^\star$ peut s'écrire comme

   $$\gamma^\star_{i_1,i_2,i_3} = a_{i_1} b_{i_2} c_{i_3} \overline{\gamma}_{i_1,i_2,i_3} \ \forall i_1, i_2, i_3,$$

   où $a_{i_1} = \exp(u_{i_1}/\varepsilon)$, $b_{i_2} = \exp(v_{i_2}/\varepsilon)$, $c_{i_3} = \exp(w_{i_3}/\varepsilon)$ et $u_{i_1}, v_{i_2}, w_{i_3}$ sont les multiplicateurs de Lagrange.
3. En imposant que $\gamma^\star$ doit satisfaire les contraintes de marginales, montrer que $a_{i_1} b_{i_2} c_{i_3}$ sont déterminés par les équations suivantes

   $$a_{i_1} = \frac{\rho_{i_1}}{\sum_{i_2,i_3} b_{i_2} c_{i_3} \overline{\gamma}_{i_1,i_2,i_3}},\ b_{i_2} = \frac{\rho_{i_2}}{\sum_{i_1,i_3} a_{i_1} c_{i_3} \overline{\gamma}_{i_1,i_2,i_3}},\ c_{i_3} = \frac{\rho_{i_3}}{\sum_{i_1,i_2} a_{i_1} b_{i_2} \overline{\gamma}_{i_1,i_2,i_3}}. \qquad (3)$$

4. Écrire l'algorithme de Sinkhorn pour ce problème.

On pourra decomposer $\overline{\gamma}_{i_1,i_2,i_3} = \tilde{\gamma}_{i_1,i_2} \tilde{\gamma}_{i_2,i_3} \tilde{\gamma}_{i_1,i_3}$ tel que les sommes dans (3) peuvent s'écrire comme des produits matrice-matrice : i.e.

$$\sum_{i_2,i_3} b_{i_2} c_{i_3} \overline{\gamma}_{i_1,i_2,i_3} = (\tilde{\gamma} \operatorname{diag}(b) \tilde{\gamma} \operatorname{diag}(c) \tilde{\gamma})_{i_1,i_1},$$

où $\tilde{\gamma}_{i_1,i_2} = \exp(-w(|x_{i_1} - y_{i_2}|)/\varepsilon)$ et $\operatorname{diag}(b)$ est une matrice diagonale (notation à la matlab).
QA.3. Programmer une fonction Python `Sinkhorn3(rho,E,eps)` prenant en argument la distribution `rho`, la matrice `E` (ayant composantes e_{ij} trouvées à la question précédente) et le

paramètre de regularisation `eps`. La fonction retournera trois vecteurs `u`, `v`, `w` et une matrice `gamma` contenant $\gamma^\star_{red}$ où

$$\gamma^\star_{red} = \sum_{i_3} \gamma^\star_{i_1,i_2,i_3}$$

Remarque 3. *Comme les electrons sont indistinguables et le coût de Coulomb est symétrique (on peut interchanger deux electrons et le potentiel de Coulomb ne change pas) alors le couplage optimal $\gamma^\star$ est aussi symétrique i.e. $\gamma^\star_{i_1,i_2,i_3} = \gamma^\star_{i_3,i_2,i_1}$. Pour visualiser le résultat (il s'agit d'un tensor en dimension !!) on peut alors se réduire à $\gamma^\star_{red} \in \mathbb{R}^{N\times N}$.*

.

QA.4.[Le cas $K = 2$] En utilisant l'algorithme du simplex et `Sinkhorn` (avec ε le plus petit possible) comparer les résultats obtenus avec le coût de Coulomb, $\rho = \chi_{[0,1]}$, $N = 30$. Tracer sur deux fenêtres graphiques les solutions obtenues et donner une interpretation physique. Reprendre la même experience mais en utilisant $c(x, y) = |x-y|^2$, La solution change ? Pourquoi ?

.

QA.5.[Le cas $K = 3$] Tester `Sinkhorn3` avec $\rho = \chi_{[0,1]}$ et $N = 30$. Tracer sur une fenêtre graphique la solution obtenue et donner une interpretation physique.

Bonus. Choisir le M2 d'optimisation et le cours de Transport Optimal :)

Option B : Wasserstein flot pour le problème de matching (7 points)

Le point clé de la théorie est le suivant : dans un espace compliqué on peut trouver le meilleur chemin (qui n'est pas forcément la ligne droite). Un tel chemin optimal s'appelle une **géodésique**. Le concept géométrique de géodésique est bien adapté à la description d'une « particule ponctuelle », astreinte à se déplacer au moindre coût d'un point à un autre sur une surface donnée. Pensons à la trajectoire d'un vol transatlantique, modélisée par le mouvement d'un point à la surface du globe terrestre.
Dans le cas où $c(x, y) = |x - y|^2$, le problème du transport est alors une distance géodésique, connue sous le nom de distance de Wasserstein, entre les distributions μ et ν :

$$\mathcal{W}_2^2(\mu, \nu) := (\mathcal{MK})$$

Dans cette section on s'intéresse à étudier et coder un algorithme de descente de gradient pour la distance de Wasserstein. En particulier on voudrait trouver les z_i $i = 1, \cdots, n$ qui minimisent l'énergie suivante

$$\mathcal{E}(z) = \mathcal{W}_2^2(\frac{1}{n}\sum_i \delta_{z_i}, \frac{1}{m}\sum_i \delta_{y_i}),$$

où les y_i sont donnés (on pourrait le générer de façon aléatoire comme à la question **Q1.5**). Dans la suite on remplacera $\mathcal{W}_2^2$ par

$$\mathcal{W}_\varepsilon(\mu, \nu) := \sum_{ij} C_{ij}\gamma_{ij} + \varepsilon \sum_{ij} \gamma_{ij}\left(\log\left(\frac{\gamma_{ij}}{\mu_i\nu_j}\right) - 1\right),$$

où C_{ij} est le coût quadratique.
On veut alors résoudre le problème suivant

$$\min_{z_i} \mathcal{E}(z) := \mathcal{W}_\varepsilon(\frac{1}{n}\sum_i \delta_{z_i}, \frac{1}{m}\sum_i \delta_{y_i}).$$

B.1. Montrer que le gradient de $\mathcal{E}(z)$ est donné par

$$(\nabla \mathcal{E}(z))_i = \mu_i z_i - \sum_j y_j \gamma_{ij},$$

où γ est le plan de transport optimal. Dans la suite on considérera le gradient normalisé

$$(\nabla \mathcal{E}(z))_i = z_i - \overline{y}_i,$$

où $\overline{y}_i := \dfrac{\sum_j y_j \gamma_{ij}}{\mu_i}$.

B.2. Tester Sinkhorn avec deux mesures générées comme à la question **Q1.5**, coder le gradient de $\mathcal{E}(z)$ et tracer le champ de gradient. (**Rmq : il faudra coder une fonction** `Sinkhorn` **permettant de résoudre un problème en 2D, voir question Q1.5**).

B.3. Programmer une fonction Python `GradWas(mu0,nu,tau,iter)` prenant en argument une distribution initiale `mu0`, la distribution donnée `nu`, le pas pour la descente de gradient `tau` et `iter` le nombre d'iterations. La fonction retournera un vecteur `z` contenant le positions z_i qui minimisent l'énergie.

B.4. Tester la fonction `GradWas(mu0,nu,tau)` en prennant $n = m = N = 100$, $\tau = 0.1$, $\mu_0 = \frac{1}{N}\sum_i z_i^0$ où `z0 = np.random.rand(2,N)-.5` , $\nu = \frac{1}{N}\sum_i y_i$ où `theta = 2*np.pi*np.random.rand(1,N)` `r = .8 + .2*np.random.rand(1,N), y = np.vstack((np.cos(theta)*r,np.sin(theta)*r))` et `iter=20` . Tracer le nuage de points z_i^k toutes les 5 iterations. Commenter les résultats.

B.5. Reprendre l'experience précédente avec $n = 100$ et $m = 200$.

B.6. Reprendre l'experience précédente et tracer le résultat final pour différentes valeurs (décroissantes) de ε. Commenter les résultats.

B.7. Reprendre l'exercice en replaçant $\mathcal{W}_\varepsilon$ avec la divergence de Sinkhorn

$$\tilde{\mathcal{W}}_\varepsilon(\mu,\nu) = \mathcal{W}_\varepsilon(\mu,\nu) - \frac{1}{2}\mathcal{W}_\varepsilon(\mu,\mu) - \frac{1}{2}\mathcal{W}_\varepsilon(\nu,\nu).$$

Bonus. Choisir le M2 d'optimisation et le cours de Transport Optimal :)