Experiment in Taxi Routes in Porto

Nafissa BENALI
Université Paris-Saclay
Saclay, France
nafissa.benali@universite-paris-saclay.fr

Laura FUENTES
Université Paris-Saclay
Saclay, France
laura.fuentes-vicente@universite-paris-saclay.fr

Rita MAATOUK
Université Paris-Saclay
Saclay, France
rita.maatouk@universite-paris-saclay.fr

1 INTRODUCTION

Ce rapport présente le travail de synthèse que nous avons réalisé pour le rendu de projet final de l'Unité d'Enseignement (UE) de Modèles Graphiques de Centrale-Supélec. Il aura été réalisé au cours du cursus de Master 2 de Mathématiques et Intelligence Artificielle de l'Institut Mathématiques d'Orsay (IMO) de l'Université Paris-Saclay.

Nous ferons l'étude d'un article de [Alp Kucukelbir] dans lequel est proposé une méthode d'automatisation de l'inférence variationnelle pour des modèles probabilistes complexes. Cela permet en théorie d'appliquer l'inférence variationnelle sur des grands jeux de données de manière plus efficace. L'objectif présenté par cet article est donc d'accélerer l'inférence variationnelle.

Tout d'abord nous ferons la présentation des méthodes théoriques abordées dans l'article, ensuite nous développerons l'implémentation de code que nous avons réalisé. Nous présenterons alors les résultats que nous aurons pu réunir avec comme étude de cas le jeu de données présentant des trajectoires de taxi à Porto en 2014.

2 CONTEXTE THÉORIQUE

2.1 Littérature

Un modèle probabiliste représente la distribution jointe $p(\theta, x)$ des variables latentes $\theta \in \mathbb{R}^K$ et des variables observées $x = x_{1:N}$.

Nous allons expliquer l'inférence variationnelle (VI), une méthode permettant d'estimer les distributions de probabilité en utilisant des techniques d'optimisation [David M Blei and McAuliffe] [Wainwright and Jordan]. L'inférence sur les variables latentes se fait à travers le posterior, qui est la distribution conditionnelle des variables latentes sachant les observations, notée $p(\theta|x)$. Nous notons $q(\theta, \phi)$ la densité variationnelle, dont nous détaillerons les spécificités dans la suite.

Le concept de l'inférence variationnelle est d'approximer $p(\theta|x)$ par $q(\theta, \phi^*)$ où,

$$\phi^* = \underset{\phi}{\operatorname{argmin}}\ KL(q(\theta;\phi)||p(\theta|x)).$$

Le critère à optimiser est la divergence de Kullback-Leibler. Cette divergence permet de mesurer l'écart entre deux distributions en évaluant la différence relative entre les probabilités assignées par les deux distributions à chaque événement possible. Elle est définie comme suit pour deux distributions de probabilité P et Q sur le même espace de probabilité : $D_{KL}(P\|Q) = \sum_x P(x) \log\left(\frac{P(x)}{Q(x)}\right)$ L'optimisation de la divergence de Kullback-Leibler relève d'un problème sous la contrainte $\text{supp}(q(\theta;\phi)) \subseteq \text{supp}(p(\theta|\mathbf{x}))$. Cependant, ce problème d'optimisation est défini en fonction du posterior $p(\theta|x)$ qui est inconnu en pratique. On peut transformer le problème de minimisation en maximisation [Michael I Jordan and Saul] [Bishop] On a :

$$\begin{aligned}
KL(q(\theta;\phi)||p(\theta|x)) =& \mathbb{E}_q[\log(q(\theta;\phi))] - \mathbb{E}_q[\log(p(\theta|x))] \\
=& \mathbb{E}_q[\log(q(\theta;\phi))] - \mathbb{E}_q[\log(\frac{p(\theta,x)}{p(x)})] \\
=& \mathbb{E}_q[\log(q(\theta;\phi))] - \mathbb{E}_q[\log(p(\theta,x)) - \log(p(x))] \\
=& \mathbb{E}_q[\log(q(\theta;\phi))] - \mathbb{E}_q[\log(p(\theta,x))] + \log(p(x))
\end{aligned}$$

Ainsi, minimiser la divergence KL revient à maximiser

$$\mathcal{L}(\phi) = \mathbb{E}_{q(\theta)}[\log(p(x,\theta))] - \mathbb{E}_{q(\theta)}[\log(q(\theta;\phi))]$$

$\mathcal{L}(\phi)$ est ce qu'on appelle l'ELBO ("Evidence Lower Bound"). Le premier terme est l'espérance du log de la densité jointe des données x et des paramètres θ. Le deuxième terme est l'entropie de la densité variationnelle $q(\theta;\phi)$:

$$\mathcal{L}(\phi) = \mathbb{E}_{q(\theta)}[\log(p(x,\theta))] + H(q(\theta;\phi))$$

$\mathcal{L}(\phi)$ est une fonction que nous pouvons évaluer car elle utilise des termes connus.

Le problème d'optimisation de l'ELBO hérite des contraintes liées à l'optimisation de la divergence KL, et veut retrouver une approximation dont le support est inclus dans celui de $p(\theta|x)$:

$$\phi^* = \underset{\phi}{\operatorname{argmax}}\mathcal{L}(\phi)$$

tel que $\text{supp}(q(\theta;\phi)) \subseteq \text{supp}(p(\theta|\mathbf{x}))$.

Notons que le support du posterior peut être également inconnu. Pour pallier à cela, on peut supposer que le support du posterior est équivalent au support du prior $p(\theta)$. Même si nous sommes confrontés à un problème d'optimisation avec contraintes,il existe de nombreuses méthodes permettant la résolution de ce problème dans la littérature. Cependant, ces démarches peuvent être complexes et demandent du temps car elles requierent des étapes préliminaires au calcul de l'approximation, comme par exemple de choisir une forme pour $q(\theta;\phi)$ qui vérifie la contrainte, calculer les espérances dans le ELBO, et décider d'une stratégie pour maximiser l'ELBO.

Dans l'article que nous présentons, les auteurs proposent une nouvelle méthode de l'Inférence Variationnelle, nommée ADVI, visant à maximiser et automatiser le processus d'inférence variationnelle.

2.2 ADVI

Nous considérons un modèle de probabilité différentiable $p(x, \theta)$. On rappelle que le problème d'optimisation en inférence variationnelle s'écrit :

$$\phi^* = \underset{\phi}{\text{argmax}}\ \mathcal{L}(\phi)$$

tel que $\text{supp}(q(\theta;\phi)) \subseteq \text{supp}(p(\theta|\mathbf{x}))$. Avec

$$\mathcal{L}(\phi) = \mathbb{E}_{q(\theta)}[\log(p(x,\theta))] + H(q(\theta;\phi))$$

Pour se ramener à un problème d'optimisation sans contraintes, les auteurs proposent de transformer les variables latentes θ en des variables ξ dans un espace de coordonnées réelles.

La transformation $T : \text{supp}(p(\theta)) \to \mathbb{R}^K$ est définie par $\xi = T(\theta)$, où T est une fonction bijective differentiable. Par conséquent, la transformation de la fonction de densité jointe est :

$$p(x,\xi) = p(x, T^{-1}(\xi)) \cdot |\det J_{T^{-1}}(\xi)|$$

La fonction objectif $\mathcal{L}(\phi)$ devient alors :

$$\begin{aligned}\mathcal{L}(\phi) =& \mathbb{E}_{q(\xi;\phi)}[\log(p(x,\xi))] + H(q(\xi;\phi))\\ =& \mathbb{E}_{q(\xi;\phi)}[\log(p(x,T^{-1}(\xi)) \cdot |\det J_{T^{-1}}(\xi)|)] + H(q(\xi;\phi))\\ =& \mathbb{E}_{q(\xi;\phi)}[\log(p(x,T^{-1}(\xi))) + \log(|\det J_{T^{-1}}(\xi)|)] + H(q(\xi;\phi))\end{aligned}$$

On traite alors, avec cette transformation, d'un problème sans contraintes :

$$\phi^* = \underset{\phi}{\text{argmax}}\ \mathcal{L}(\phi)$$

Après la transformation, les variables latentes ξ ont un support dans l'espace des coordonnées réelles $\mathbb{R}^K$. Nous avons beaucoup de choix d'approximation variationnelle $q(\xi;\phi)$ dans cet espace.

Les auteurs ont présenté deux options :

(1) Mean-field Gaussian :
$q(\xi;\phi)$=Normal$(\xi;\mu,$ diag (σ^2) $)$=$\prod_{k=1}^K$ Normal$(\xi_k;\mu_k,\sigma_k^2)$, où le vecteur $\phi = (\mu_1, .., \mu_K, \sigma_1^2, ..., \sigma_K^2)$ regroupe la moyenne et la variance de chaque facteur gaussien. On pose $\omega = \log(\sigma)$

(2) Full-rank Gaussian :
$q(\xi;\phi)$=Normal$(\xi;\mu,\sum)$, où le vecteur $\phi = (\mu, \sum)$ regroupe la moyenne et la matrice de covariance. Pour garantir que $\sum$ reste toujours semi-définie positive, nous reparamétrons la matrice de covariance en utilisant une factorisation de Cholesky, $\sum = LL^T$. Le full-rank Gaussian généralise la mean-field Gaussian [Seeger].

Nous pouvons remarquer que l'ELBO comprend un terme non calculable : $\mathbb{E}_{q(\xi;\phi)}[\log(p(x,T^{-1}(\xi)))]$ en raison de la présence de deux paramètres inconnus : ξ et ϕ. Les auteurs développent un algorithme de montée de gradient stochastique qui utilise :

— l'intégration MC pour approximer les espérances non calculables, $q(\xi;\phi)$

— la différenciation automatique pour calculer les gradients.

Notons que nous ne pouvons pas utiliser directement la différenciation automatique sur l'ELBO à cause de la présence du terme d'espérance non calculable. Cependant, nous pouvons différencier automatiquement les fonctions à l'intérieur de cette espérance. En d'autres termes, pour appliquer la différenciation automatique, nous voulons pousser l'opération de gradient à l'intérieur de l'espérance. À cette fin, nous utilisons une dernière transformation : la standarisation elliptique $\eta = S_\phi(\xi)$ [Härdle and Simar]. Cette dernière transformation, S_ϕ, convertit l'approximation variationnelle gaussienne $q(\xi;\phi)$ en une gaussienne centrée réduite $q(\eta)$. Cette nouvelle variable η utilise θ et ξ, ce qui permet d'une part, de calculer les derivées partielles par la règle de la chaîne, puis d'autre part, de se ramener à une espérance plus facile à calculer. Ainsi le calcul des gradients est possible.

(1) mean-field Gaussian : $\eta = S_\phi(\xi) = diag(exp(\omega))^{-1}(\xi - \mu)$

(2) full-rank Gaussian : $\eta = S_\phi(\xi) = L^{-1}(\xi - \mu)$

On obtient alors :

$$\begin{aligned}\mathcal{L}(\phi) = \mathbb{E}_{\mathcal{N}(\eta;0,I)}[\log p(x, T^{-1}(S_\phi^{-1}(\eta))) + \log(|\det J_{T^{-1}}(S_\phi^{-1}(\eta))|)]\\ + H(q(\xi;\phi))\end{aligned}$$

Il ne nous reste alors qu'à déterminer les paramètres de la distribution variationelle $q(\eta) = \mathcal{N}(\mu, diag(e^\omega)^2)$ (mean field Gaussian). Les auteurs proposent un algorithme itératif que l'on retrouvera en [Annexe A.2].

2.3 Analyse en composantes principales probabilistique (PPCA)

Tout comme la PCA, l'objectif de la PPCA est de trouver une représentation des données de dimension réduite [Tipping and Bishop]. Le modèle posé considère que l'ensemble des observations est généré par la relation $x = wz + \sigma$, où w et z sont des paramètres latents et σ est l'écart-type du bruit. Cette relation implique que la distribution des données x suit une distribution normale $\mathcal{N}(wz, \sigma^2)$. Ainsi, étant donné un ensemble d'observations x et un ensemble de paramètres $\theta = (z, w, \sigma)$, l'objectif de la PPCA peut être réexprimé comme la détermination de la distribution postérieure $p(\theta|x)$. Cette distribution postérieure permet de déterminer les valeurs optimales des paramètres θ, ce qui facilite la réduction de dimension et la reconstruction des observations [Bhattacharyya].

Modélisation de la vraisemblance. Considérons un jeu de données de dimension (N, D), tel que $\mathbf{x} = (x_i)_{0\le i\le N}$ avec $x_i \in \mathbb{R}^D$ pour tout i. Soit $M < D$ la dimension du sous-espace sur lequel on veut projeter nos données. On définit θ le vecteur de paramètres du modèle tel que $\theta = (w, z, \sigma, \alpha)$. On rappelle que $p(x,\theta) = p(x|\theta) \times p(\theta)$.

Nous supposons que $x|\theta \sim \mathcal{N}(w^T z, \sigma^2)$. L'objectif est d'évaluer la probabilité que chaque observation soit bien distribuée selon la loi de $x|\theta$. D'autre part, $p(\theta) = p(w,z,\sigma,\alpha) = p(w|\alpha) \times p(\alpha) \times p(\sigma) \times p(z)$. Les paramètres sont indépendants sauf w et α. α est un paramètre introduit lors de la réduction de dimension pour déterminer la dimension du sous-espace sur lequel on projette nos données.

Nous faisons alors une série d'hypothèses permettant l'estimation des paramètres du modèle :

— $x|\theta \sim \mathcal{N}(w^T z, \sigma^2)$

— $x|w,z,\sigma \sim \mathcal{N}(x; wz, \sigma\mathbf{I})$

— $w|\alpha \sim \mathcal{N}(w; \mathbf{0}, \sigma\text{diag}(\alpha))$

— $\alpha \sim \text{InvGamma}(\alpha; 1, 1)$

En maximisant la densité à posteriori, nous estimons les paramètres du modèle et effectuons une projection sur un espace de dimension inférieure.

3 IMPLEMENTATIONS ET RÉSULTATS

Les auteurs offrent diverses implémentations d'ADVI, démontrant la souplesse de la technique présentée. Dans cette étude, nous nous focaliserons sur l'exemple "Million Taxi Trajectories". Ils fournissent un jeu de données contenant les trajectoires des taxis à Porto pour l'année 2014 [PKDD]. L'implémentation que nous avons réalisée de ce problème est à retrouver à l'adresse suivante : https://github.com/laufuentes/ADVI-taxi-.
Ce jeu de données se compose de coordonnées (x, y) enregistrées toutes les 15 secondes pour chaque trajectoire de taxi. L'objectif principal est d'analyser les motifs intrinsèques au trafic afin de créer des clusters de trajectoires. L'approche présentée dans l'article peut être divisée en trois étapes successives : une interpolation des données, une analyse en composantes principales probabiliste (PPCA), et enfin un modèle de mélange basé sur la PPCA. Nous détaillerons, à chaque étape, les parties d'implémentation que nous avons réalisées nous-même. Il est important de préciser que dû à un manque de ressources computationnelles nous avons fait le choix de réduire la taille des données.

3.1 Interpolation

Les coordonnées des trajectoires sont enregistrées toutes les 15 secondes, ce qui implique que la longueur des trajets varie en fonction de leur durée. Afin d'éviter tout biais dans la distribution des clusters en fonction de la longueur des trajets, les auteurs recommandent cette étape préliminaire d'interpolation. En remarquant que la durée moyenne des trajets est de 13 minutes, ce qui correspond à une trajectoire de 50 coordonnées. C'est pour cela que les auteurs suggèrent une interpolation à 50 points. Cette approche normalise la longueur des trajets et rend le clustering indépendant de la durée du trajet. L'interpolation aura été réalisée à l'aide de la fonction 'np.interp1d' de numpy.

3.2 Réduction de dimension (PPCA)

Après l'interpolation des trajectoires, une étape cruciale consiste à effectuer une réduction de dimension. Son objectif est d'identifier une représentation des données de dimension inférieure, permettant ainsi de mettre en lumière les structures cachées dans les trajectoires.

Dans le cadre de l'implémentation d'ADVI, une généralisation bayésienne de la PCA, appelée analyse en composantes principales probabiliste (PPCA) [Tipping and Bishop] est employée. Dans l'article, elle est utilisée avec Détermination de la Pertinence Automatique (ARD). Pour des raisons de manque de ressources computationnelles, on a fait le choix de l'employer sans ARD. Étant donné que le terme $p(\theta|x)$ n'a pas de formule explicite calculable, nous cherchons maintenant à déterminer une distribution q pour approcher l'à posteriori $p(\theta|x)$ et enfin, optimiser θ. Nous déploierons ainsi ADVI pour déterminer les paramètres μ et ω (mean-field Gaussian) de la distribution variationnelle $q(v) = \mathcal{N}(\mu, diag(e^{\omega})^2)$.

Mise en place d'ADVI. Le cœur du problème consiste à mettre à jour μ et ω itérativement en utilisant les gradients suivants :

$$\begin{aligned}\nabla_{\mu}\mathcal{L} &= \mathbb{E}_{\mathcal{N}(0,I)}[\nabla_{\theta} log(p(x,\theta))\nabla_{\xi}T^{-1}(\xi)] + \nabla_{\xi} log(|J_{T^{-1}}|)] \\ &= \mathbb{E}_{\mathcal{N}(0,I)}[f_1(x,\theta,\xi)] \\ \nabla_{\omega}\mathcal{L} &= \mathbb{E}_{\mathcal{N}(0,I)}[\nabla_{\theta} log(p(x,\theta))\nabla_{\xi}T^{-1}(\xi)] + \nabla_{\xi} log(|J_{T^{-1}}|)\eta^T \mathrm{diag}(e^{\omega})] + 1 \\ &= \mathbb{E}_{\mathcal{N}(0,I)}[f_2(x,\theta,\xi)]\end{aligned}$$

Le calcul des gradients repose sur le calcul des espérances $\mathbb{E}_{\mathcal{N}(0,I)}[f_i(x,\theta,\xi)]\ \forall i \in \{1,2\}$ par intégration Monte Carlo : $\mathbb{E}_{\mathcal{N}(0,I)}[f_i(x,\theta,\xi)] \approx \frac{1}{M}\sum_{m=0}^{M} f_i(x,\theta,\xi)\ \forall i \in \{1,2\}$

L'objet devient ainsi de calculer $f_i(x,\theta,\xi), \forall i \in \{1,2\}$.
Calculons d'abord le terme $log(p(x,\theta_m))$, pour $m \in \{1,...,M\}$.
On tire ainsi $\eta_m \sim \mathcal{N}(0,I)$, puis on applique la standardisation elliptique pour obtenir $\xi_m = S_{\phi}(\eta_m) = \frac{(\eta_m - \mu)}{\omega}$. Ensuite, on extrait $\theta_m = T^{-1}(\xi_m)$. Cette transformation T permet d'ajuster θ_m au support de σ. Son inverse nous permet ainsi d'extraire des échantillons des priors établis dans le cadre de la PPCA : z_m, w_m et σ_m. Bien que cette transformation ne soit pas spécifiée dans l'article, nous avons implémenté l'exponentielle sur une partie de θ_m afin de le restreindre au support de $\sigma \in \mathbb{R}^+$. Enfin, nous utilisons ces échantillons pour calculer la probabilité jointe $log(p(x,\theta_m))$.
Les gradients $(\nabla_{\theta_m}, \nabla_{\xi_m})$ sont calculés par différentiation automatique. Dernièrement, les termes de step-size sont calculés pour chacun des gradients à partir du learning rate spécifié en input et la formule suivante :

$$\begin{aligned}\rho_k^{(i)} &= \eta \times i^{-1/2+\epsilon} \times (\tau + \sqrt{s_k^{(i)}}) \\ s_k^{(i)} &= \alpha g_k^{2(i)} + (1-\alpha) s_k^{(i-1)}\end{aligned}$$

On repète M fois ce processus pour approcher les espérances par Monte Carlo. On a finalement tous les ingrédients pour mettre à jour μ et ω. L'algorithme ADVI converge lorsque le changement de l'ELBO entre deux itérations consécutives est inférieur à un seuil [Figure 1].

Projection dans le sous-espace. Une fois μ et ω déterminés, la projection dans le sous-espace de plus petite dimension peut être effectuée en tirant ξ selon la distribution variationelle approchant $p(\theta|x)$, $\xi^* \sim q(\mu^*, e^{\omega *}) = \mathcal{N}(\mu, e^{\omega})$. Avec $x \sim wz + \sigma$, la représentation de x dans la plus petite dimension est obtenue en calculant z comme suit : $z = (w^T w)^{-1}(w^T(x-\sigma))$.

Dans cette partie, nous avons réalisé l'implémentation complète de l'algorithme ADVI. Les distributions de densité ont été obtenues grâce à TensorFlow Probability, tandis que la différentiation automatique a été réalisée avec 'tf.GradientTape' de TensorFlow.

3.3 Clustering des trajectoires (BGMM)

On rappelle qu'un modèle de mélange est un outil essentiel pour le clustering. Il permet de regrouper des données similaires ensemble en modélisant des distributions de probabilité complexes. Les observations sont attribuées à des clusters en fonction de leur probabilité d'appartenance à chaque composante du modèle. Les paramètres du modèle sont ensuite estimés itérativement jusqu'à

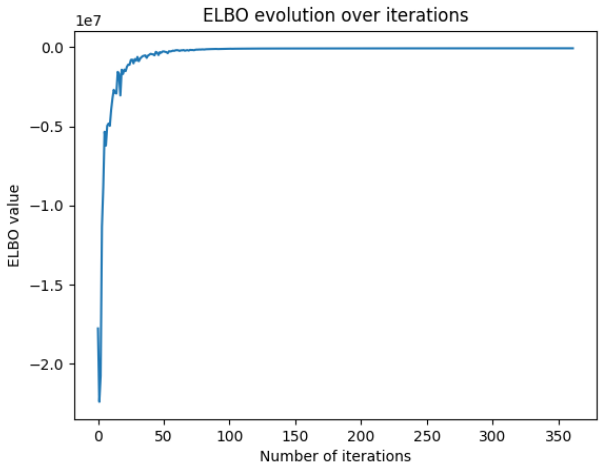

Figure 1 : Évolution de l'ELBO au cours des itérations

convergence. Ce processus permet d'identifier des clusters dans les données, ce qui facilite l'analyse et la compréhension des structures sous-jacentes.

Le modèle de mélange gaussien bayésien (Bayesian Gaussian Mixture Model) fonctionne en intégrant plusieurs processus de génération de données dans un seul modèle. Pour trouver l'appartenance à chaque cluster, le modèle estime la probabilité de chaque observation $z_i|\theta$, d'appartenir au cluster C_k, c'est-à-dire : $p(z \in C_k|\theta)$. Dans notre implémentation, nous avons utilisé une version déjà existante sur sklearn, 'BayesianGaussianMixture' [Bruno Nicenboim and Vasishth].

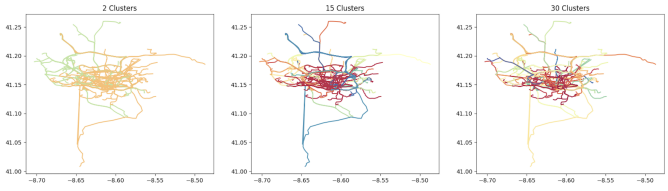

Figure 2 : Groupement des trajectoires pour différents nombres de clusters

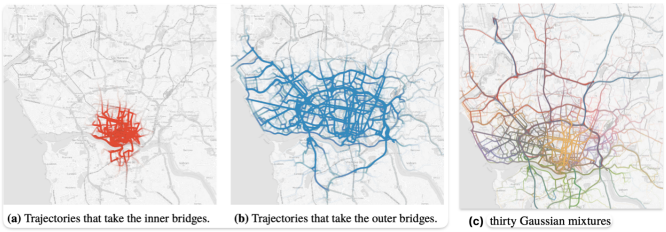

Figure 3 : Groupement des trajectoires pour 2 et 30 clusters [Extrait de l'article [Alp Kucukelbir]]

Lorsque nous comparons nos résultats [Figure 2] avec ceux de l'article [Figure 3], nous observons des différences notables. Des explications pour ces différences peuvent être trouvées dans notre méthode de reproduction des résultats. Nous avons dû réduire considérablement la dimension du jeu de données, ce qui a probablement entraîné une perte d'information. De plus, nous n'avons pas utilisé ARD ni sup-PPCA, ce qui pourrait également influencer les résultats. Enfin, le choix du learning rate et les transformations appliquées n'ont pas été précisés. Nous n'avons donc pas de moyen de savoir si notre choix pour le learning rate et la transformation est pertinent, ce qui pourrait créer des différences avec la figure de l'article [Figure 3].

4 CONCLUSION

En conclusion, l'implémentation de l'algorithme ADVI nous a permis d'effectuer un clustering sur un problème sans contraintes, démontrant ainsi son potentiel dans divers contextes grâce à son utilisation de l'inférence variationnelle. L'article présente également une gamme variée d'applications pour cet algorithme. Cependant, notre analyse a révélé plusieurs difficultés. Tout d'abord, l'absence de vérité de terrain rend difficile l'évaluation de la qualité du clustering produit. De plus, la compréhension de Stan, l'outil utilisé pour implémenter l'algorithme, s'est avérée être un défi. Le problème de reproductibilité a également été soulevé en raison du manque de paramètres dans l'article, ce qui a entravé notre capacité à reproduire les résultats. De plus, la partie application de l'article n'a pas été suffisamment détaillée, ne fournissant pas une explication claire des transformations effectuées dans chaque cas de figure. En outre, les contraintes computationnelles nous ont contraints à réduire considérablement les dimensions du jeu de données. Enfin, nous avons constaté que l'algorithme est sensible à la valeur du learning rate, ce qui nécessite une exploration minutieuse des paramètres. Malgré ces défis, cette expérience nous a permis de mieux comprendre les avantages et les limites de l'algorithme ADVI dans le contexte du clustering.

BIBLIOGRAPHIE

[Alp Kucukelbir] Alp Kucukelbir, Dustin Tran, R. R. A. G. D. M. B. Automatic differentiation variational inference.
[Bhattacharyya] Bhattacharyya, S. Probabilistic view of principal component analysis.
[Bishop] Bishop, C. M. Pattern recognition and machine learning.
[Bruno Nicenboim and Vasishth] Bruno Nicenboim, D. S. and Vasishth, S. An introduction to bayesian data analysis for cognitive science.
[David M Blei and McAuliffe] David M Blei, A. K. and McAuliffe, J. D. Variational inference : a review for statisticians.
[Härdle and Simar] Härdle, W. and Simar, L. Applied multivariate statistical analysis.
[Michael I Jordan and Saul] Michael I Jordan, Zoubin Ghahramani, T. S. J. and Saul, L. K. An introduction to variational methods for graphical models. machine learning, 37(2) :183–233.
[PKDD] PKDD, E. Taxi service trajectory - prediction challenge.
[Seeger] Seeger, M. Gaussian covariance and scalable variational inference.
[Tipping and Bishop] Tipping, M. E. and Bishop, C. M. Probabilistic principal component analysis. *Journal of the Royal Statistical Society Series B : Statistical Methodology*, 61(3) :611–622.
[Wainwright and Jordan] Wainwright, M. J. and Jordan, M. I. Graphical models, exponential families, and variational inference.

A ANNEXE

A.1 Jeu de données

Le jeu de données [PKDD] décrit avec précision les trajectoires effectuées par les 442 taxis opérant dans la ville de Porto, au Portugal. Chaque entrée du jeu de données contient deux champs principaux : TRIP_ID et POLYLINE. Le champ TRIP_ID est une chaîne de caractères qui contient un identifiant unique pour chaque trajet effectué par un taxi. Le champ POLYLINE est également une chaîne de caractères, mais il contient une liste de coordonnées GPS représentées sous forme de chaîne. Ces coordonnées sont enregistrées toutes les 15 secondes pendant le trajet, et la première paire de coordonnées représente le point de départ du trajet tandis que la dernière paire représente sa destination.

A.2 Algorithme

Algorithm 1 Inference Variationnelle par Différenciation Automatique (ADVI)

Entrée : Jeu de données $x = x_{1:N}$, modèle $p(x, \theta)$.
Configurer le compteur d'itérations $i = 1$.
Initialiser $\mu^{(1)} = 0$.
Initialiser $\omega^{(1)} = 0$ (mean-field) ou $L^{(1)} = I$ (full-rank).
Déterminer η par recherche sur des valeurs finies.
while *Le changement dans l'ELBO est supérieur à un certain seuil* **do**
  Tirer M échantillons $\eta_m \sim \mathcal{N}(0, I)$.
  Approximer $\nabla_\mu \mathcal{L}$ par intégration MC.
  Approximer $\nabla_\omega \mathcal{L}$ ou $\nabla_L \mathcal{L}$ par intégration MC.
  Calculer le pas de mise à jour $\rho(i)$.
  Mettre à jour $\mu^{(i+1)} = \mu^{(i)} + \text{diag}(\rho^{(i)})\nabla_\mu \mathcal{L}$.
  Mettre à jour $\omega^{(i+1)} = \omega^{(i)} + \text{diag}(\rho^{(i)})\nabla_\omega \mathcal{L}$.
  ou $L^{(i+1)} = L^{(i)} + \text{diag}(\rho^{(i)})\nabla_L \mathcal{L}$
  Incrémenter le compteur d'itérations
end while
Retourner $\mu^\star = \mu^{(i)}$, $\omega^\star = \omega^{(i)}$ ou $L^\star = L^{(i)}$