Université de Montréal

Apprentissage statistique avec le processus ponctuel déterminantal

par

Sergio Vicente

Département de mathématiques et de statistique
Faculté des arts et des sciences

Thèse présentée en vue de l'obtention du grade de
Philosophiæ Doctor (Ph.D.)
en statistique

8 février 2021

Résumé

Cette thèse aborde le processus ponctuel déterminantal, un modèle probabiliste qui capture la répulsion entre les points d'un certain espace. Celle-ci est déterminée par une matrice de similarité, la matrice noyau du processus, qui spécifie quels points sont les plus similaires et donc moins susceptibles de figurer dans un même sous-ensemble. Contrairement à la sélection aléatoire uniforme, ce processus ponctuel privilégie les sous-ensembles qui contiennent des points diversifiés et hétérogènes. La notion de diversité acquiert une importante grandissante au sein de sciences comme la médecine, la sociologie, les sciences forensiques et les sciences comportementales. Le processus ponctuel déterminantal offre donc une alternative aux traditionnelles méthodes d'échantillonnage en tenant compte de la diversité des éléments choisis. Actuellement, il est déjà très utilisé en apprentissage automatique comme modèle de sélection de sous-ensembles. Son application en statistique est illustrée par trois articles. Le premier article aborde le partitionnement de données effectué par un algorithme répété un grand nombre de fois sur les mêmes données, le partitionnement par consensus. On montre qu'en utilisant le processus ponctuel déterminantal pour sélectionner les points initiaux de l'algorithme, la partition de données finale a une qualité supérieure à celle que l'on obtient en sélectionnant les points de façon uniforme. Le deuxième article étend la méthodologie du premier article aux données ayant un grand nombre d'observations. Ce cas impose un effort computationnel additionnel, étant donné que la sélection de points par le processus ponctuel déterminantal passe par la décomposition spectrale de la matrice de similarité qui, dans ce cas-ci, est de grande taille. On présente deux approches différentes pour résoudre ce problème. On montre que les résultats obtenus par ces deux approches sont meilleurs que ceux obtenus avec un partitionnement de données basé sur une sélection uniforme de points. Le troisième article présente le problème de sélection de variables en régression linéaire et logistique face à un nombre élevé de covariables par une approche bayésienne. La sélection de variables est faite en recourant aux méthodes de Monte Carlo par chaînes de Markov, en utilisant l'algorithme de Metropolis-Hastings. On montre qu'en choisissant le processus ponctuel déterminantal comme loi *a priori* de l'espace des modèles, le sous-ensemble final de variables est meilleur que celui que l'on obtient avec une loi *a priori* uniforme.

Mots-clé : Algorithme de Lanczos ; approximation de Laplace ; décomposition en éléments propres ; exclusion mutuelle probabiliste ; méthodes à noyaux ; méthode des k plus proches voisins ; modèle graphique déterminantal ; prédiction *a posteriori* ; regroupement de données.

Abstract

This thesis presents the determinantal point process, a probabilistic model that captures repulsion between points of a certain space. This repulsion is encompassed by a similarity matrix, the kernel matrix, which selects which points are more similar and then less likely to appear in the same subset. This point process gives more weight to subsets characterized by a larger diversity of its elements, which is not the case with the traditional uniform random sampling. Diversity has become a key concept in domains such as medicine, sociology, forensic sciences and behavioral sciences. The determinantal point process is considered a promising alternative to traditional sampling methods, since it takes into account the diversity of selected elements. It is already actively used in machine learning as a subset selection method. Its application in statistics is illustrated with three papers. The first paper presents the consensus clustering, which consists in running a clustering algorithm on the same data, a large number of times. To sample the initials points of the algorithm, we propose the determinantal point process as a sampling method instead of a uniform random sampling and show that the former option produces better clustering results. The second paper extends the methodology developed in the first paper to large-data. Such datasets impose a computational burden since sampling with the determinantal point process is based on the spectral decomposition of the large kernel matrix. We introduce two methods to deal with this issue. These methods also produce better clustering results than consensus clustering based on a uniform sampling of initial points. The third paper addresses the problem of variable selection for the linear model and the logistic regression, when the number of predictors is large. A Bayesian approach is adopted, using Markov Chain Monte Carlo methods with Metropolis-Hasting algorithm. We show that setting the determinantal point process as the prior distribution for the model space selects a better final model than the model selected by a uniform prior on the model space.

Keywords : Data grouping; determinantal graphical model; eigendecomposition; k-nearest neighbors method; kernel-based methods; Lanczos algorithm; Laplace's approximation ; posterior prediction; probabilistic mutual exclusion.

Table des matières

Liste des tableaux

Table des figures

Liste des sigles et des abréviations

ARI	Indice de Rand ajusté, de l'anglais *Adjusted Rand Index*
ARPACK	Paquet logiciel d'Arnoldi, de l'anglais *Arnoldi Package*
CPU	Processeur, de l'anglais *Central Processing Unit*
DNA	Acide désoxyribonucléique, de l'anglais *Deoxyribonucleic Acid*
DPP	Processus Ponctuel Déterminantal, de l'anglais *Determinantal Point Process*
GB	Gigaoctet, de l'anglais *Gigabyte*
GHz	Gigahertz
KL	Kullback-Leibler
KVI	Indice de validation basé sur un noyau, de l'anglais *Kernel-based Validation Index*
LASSO	LASSO, de l'anglais *Least Absolute Shrinkage And Selection Operator*

MAP	Maximum *a posteriori*
MCMC	Monte Carlo par chaînes de Markov, de l'anglais *Markov Chain Monte Carlo*
MH	Metropolis-Hastings
MNIST	MNIST, de l'anglais *Modified National Institute of Standards and Technology*
NNGP	Processus Gaussien des plus proches voisins, de l'anglais *Nearest Neighbor Gaussian Process*
PAM	Partitionnement autour de médoïdes, de l'anglais *Partitioning Around Medoids*
PC	Ordinateur personnel, de l'anglais *Personal Computer*
RAM	Mémoire vive, de l'anglais *Random-access Memory*
RBF	Fonction de base radiale, de l'anglais *Radial Basis Function*
RMSE	Racine carrée de l'erreur quadratique moyenne, de l'anglais *Root Mean Square Error*
RN	Vrai nombre de groupes

Dédicace

À ma soeur Ana, que j'aime de tout mon coeur.

Remerciements

L'aboutissement de mon doctorat, marqué par la fin de ma thèse, permet de clore un chapitre important de ma vie. Il y a dix ans en arrière, je n'aurais jamais pensé que je me retrouverais au Canada un jour pour y faire un doctorat. Le cheminement au sein de la recherche a été un chemin parsemé d'embûches, avec beaucoup de hauts et de bas, des moments de joie, des moments de frustration, bref, tout un amalgame de sentiments mélangés. Au final, j'en retire beaucoup de bénéfices et d'éléments positifs. J'ai énormément appris et Dieu merci, ce n'est pas la fin de l'histoire car je continuerai à apprendre chaque jour davantage ! Le savoir n'est pas un ensemble fini et il me sera donc toujours possible de repousser les frontières de mon savoir. Le doctorat a suscité en moi un énorme sentiment de respect et d'humilité qui sera toujours présent. Sans le concours de certaines personnes, je n'aurais jamais réussi à remettre cette thèse et venir à bout de mon doctorat. Je tiens donc tout particulièrement à les remercier !

En tout premier lieu, je voudrais remercier mon directeur de recherche, Alejandro Murua, pour son énorme expertise, sa minutie et son incroyable talent pour faire la révision des articles que je présente dans cette thèse ! Je lui tire vraiment mon chapeau et ne le remercierai jamais assez pour son soutien crucial dans la partie finale de ma thèse ! Merci Alejandro ! Je tiens aussi à remercier du fond du coeur le responsable informatique du département, qui m'a sauvé la vie à plusieurs reprises ! Une grosse partie de ma recherche repose sur l'utilisation intensive des ordinateurs du département et son aide a été précieuse quand je me retrouvais face à des problèmes informatiques. Un gros merci également aux membres du personnel administratif du département, dont le soutien et la tendresse ont su me réconforter à plusieurs reprises. Je m'adresse tout particulièrement à Églantine Hontanx, Marie-Claude Turmel, Julie Colette, Anne-Marie Dupuis et ma douce amie Lise Imbeault, sans oublier de mentionner notre chère directrice actuelle du département Marlène Frigon. Même si certaines de ces personnes ne sont plus au département, elles y restent associées dans mon coeur. Et pour fermer la boucle en ce qui concerne le département de mathématiques et statistique, je remercie chaleureusement tous mes étudiants.

Beaucoup de personnes à l'extérieur de l'Université de Montréal ont également joué un rôle crucial dans ma vie de doctorant. Je commencerai par remercier toute l'équipe de l'Institut de recherche du Centre Universitaire de Santé McGill, avec qui j'ai collaboré à de nombreuses reprises et envers qui j'ai fini par développer un énorme attachement ! J'aimerais remercier plus particulièrement Bertand Lebouché, David Lessard, Kim Engler, Isabelle Toupin et Chantal Burelle. Merci pour votre soutien et vos encouragements à tous les niveaux ! Sans vous, je ne serais pas ce que je suis aujourd'hui ! Je n'oublie évidemment pas ma chère et tendre Lina del Balso, qui fait un travail incroyable en tant qu'infirmière, avec des qualités humaines hors pair !

Et finalement, si je les laisse pour la fin, ce n'est pas parce que je les aime moins. On dit toujours que l'on garde le meilleur pour la fin et c'est ce que je veux faire. Les paroles les plus marquantes et les plus touchantes dans un texte sont celles de la fin. Je veux donc remercier mes précieux amis qui m'ont apporté beaucoup d'amour et qui ont toujours cru en moi. Les moments passés avec vous ont permis de combler mes moments de tristesse et de frustration. Merci Floriane, Isabelle, Élodie, Ophélie, Aurélie, Ima et Sonia. Un gros merci aussi à mes amis péruviens Rossio, Natalia et Claudio. Merci égalament à Philippe, Janie, Kenza, Louis-Xavier et Annie. Un gros merci également à ma directrice de maîtrise et amie proche Isabel, sans qui je n'aurais jamais eu l'idée de faire un doctorat. Pour le mot de la fin, le trio que j'aime le plus au monde : papa, maman et ma soeur.... Merci du fond de mon âme pour votre amour inconditionnel, votre soutien constant et d'avoir toujours continué à croire en moi dans les moments difficiles.... Je vous aime tendrement et à l'infini....

Introduction

La sélection de sous-ensembles d'éléments à partir d'un ensemble plus vaste est une étape que l'on retrouve dans la plupart des méthodes statistiques. Que ce soit de l'échantillonnage à des fins d'inférence statistique, une sélection de points aléatoires pour initialiser un algorithme ou des simulations pour calculer la valeur approximative d'une intégrale, toutes ces méthodes dépendent largement des sous-ensembles sélectionnés. Bien que des techniques de sélection sophistiquées existent au sein de la statistique, comme l'échantillonnage par grappes, l'échantillonnage stratifié ou l'échantillonnage systématique, l'échantillonnage aléatoire simple est la façon la plus courante de sélectionner un sous-ensemble d'éléments. Bien que très répandue, cette méthode de sélection a pourtant ses limitations lorsqu'il s'agit de répondre à certains besoins particuliers. Dans le domaine des essais cliniques, par exemple, la notion de diversité occupe une place de plus en plus importante. Selon Clark *et al.* (2019), la diversité est cruciale pour assurer que les participants de l'essai soient suffisamment représentatifs da la population qui prendra le traitement par la suite. La diversité des éléments assure aussi une généralisation des résultats à une plus grande population. Les essais cliniques randomisés classiques se caractérisent par un ensemble de participants très homogènes et moins diversifiés, limitant la généralisation des résultats à une population plus restreinte. La notion de diversité est également très présente dans des domaines comme le développement et l'apprentissage (Legare, 2017) ou dans les sciences forensiques (Wagstaff et LaPorte, 2018).

Le processus ponctuel déterminantal (ou DPP, de l'anglais *Determinantal Point Process*) est un processus ponctuel utilisé pour modéliser la répulsion entre points de façon probabiliste, qui tire ses origines de la physique. On retrouve sa principale utilisation en apprentissage automatique (Kulesza et Taskar, 2012), comme méthode de sélection de sous-ensembles basés sur la diversité des éléments sélectionnés. Si l'on considère un ensemble fini d'éléments $\mathcal{S} = \{1, 2, \ldots, n\}$ et une matrice symétrique L semi-définie positive qui résume les similarités entre chaque paire d'éléments de $\mathcal{S}$, le DPP est une distribution de probabilité définie sur l'ensemble $2^{\mathcal{S}}$ qui attribue une probabilité plus élevée aux sous-ensembles de $2^{\mathcal{S}}$ caractérisés par une plus grande diversité des éléments qui les composent. Cette probabilité est proportionnelle au déterminant des sous-matrices de L indexées par les éléments de chaque sous-ensemble. Pour le choix de la matrice de similarité L, plusieurs choix sont possibles.

Notre attention s'est portée sur deux choix en particulier : les matrices construites par les méthodes à noyaux et les matrices de corrélations. Le premier choix est motivé par l'utilisation du produit scalaire entre deux vecteurs comme mesure de similarité, typique des méthodes à noyaux. Une matrice de similarité peut donc être construite à partir du produit scalaire entre les vecteurs d'observations correspondants à chaque élément de $\mathcal{S}$. Quant à la matrice de corrélations, il s'agit d'un choix naturel en statistique comme mesure de similarité.

Le partitionnement de données est une des méthodes statistiques où la sélection de points constitue un élément clé. On retrouve ici l'algorithme de partitionnement des k-médoïdes (Rdusseeun et Kaufman, 1987) et l'algorithme des k-moyennes (Lloyd, 1982). Traditionnellement, ces algorithmes déterminent une partition optimale des données en exécutant une seule fois les étapes qui les composent, à partir des points sélectionnés au départ. Cette façon de procéder a comme désavantage d'être sensible aux points sélectionnés. Deux algorithmes différents peuvent donc conduire à deux partitions optimales complètement différentes. Monti *et al.* (2003) introduisent la technique faisant un partitionnement de données par consensus qui consiste à exécuter le même algorithme de partitionnement plusieurs fois. La partition optimale est ensuite obtenue par agrément entre les diverses exécutions de l'algorithme. Vega-Pons et Ruiz-Shulcloper (2011) réfèrent que cette façon de procéder augmente la robustesse et la consistence des résultats du partitionnement. Une sensibilité plus faible face à la sélection initiale de points est également pointée par les auteurs. Étant donné que la majeure partie des algorithmes de partitionnement sont basés sur une sélection uniforme de points, le DPP est proposé comme méthode de sélection de sous-ensembles d'éléments, comme alternative à l'échantillonnage basé sur la distribution uniforme. On montre que l'utilisation du DPP comme méthode de sélection initiale de points augmente la qualité des résultats obtenus avec le partitionnement par consensus, par rapport à une sélection de points uniforme. L'algorithme d'échantillonnage de Hough *et al.* (2006) et Kulesza et Taskar (2012) est utilisé pour obtenir des sous-ensembles de points provenant d'un DPP. Cet algorithme est basé sur la décomposition spectrale de la matrice L, ce qui impose un défi computationnel lorsque la taille de celle-ci est très grande. La complexité de cette décomposition est d'ordre $O\left(n^3\right)$. Pour remédier à ce problème, on propose deux méthodes qui permettent de faire un partitionnement de données par consensus, en présence de données avec un grand nombre d'observations. La première est basée sur le NNGP (de l'anglais, *Nearest Neighbor Gaussian Process*) de Datta *et al.* (2016) et la seconde est basée sur l'échantillonnage de petites sous-matrices à partir de la matrice L originale et de grande taille.

La sélection aléatoire de points joue également un rôle en analyse de régression, particulièrement dans la problématique de sélection de variables. Le but est de sélectionner un sous-ensemble de covariables parmi un ensemble de variables, de façon à pouvoir expliquer une variable réponse d'intérêt. Lorsque le nombre de variables est très élevé, il devient impraticable et inefficace d'évaluer tous les sous-ensembles de covariables possibles. Les méthodes

MCMC (de l'anglais, *Markov Chain Monte Carlo*) sont donc suggérées par O'Hagan et Forster (2004), de façon à obtenir une chaîne de modèles qui représente un échantillon, du moins approximativement, de la distribution *a posteriori* de l'espace de tous les modèles possibles. Plusieurs algorithmes sont possibles pour obtenir une telle chaîne, dont l'algorithme de MH (de l'anglais, *Metropolis-Hastings*). Cet algorithme itératif permet de construire la chaîne au moyen de probabilités de transition, par sélection aléatoire d'un modèle candidat à ajouter ou non à chaque étape de la chaîne. L'approche naïve consiste à adopter une distribution uniforme pour la sélection de modèles candidats. Zanella (2019) démontre que le choix d'une distribution non informative pour définir les transitions de la chaîne, comme la distribution uniforme, ralentit le temps de mélange des chaînes de Markov. On propose donc d'utiliser le DPP comme distribution *a priori* de l'espace de tous les modèles et pour définir les probabilités de transition de la chaîne. On montre que les sous-ensembles de covariables sélectionnés avec le DPP sont plus proches de l'ensemble des covariables du vrai modèle (en termes du score F_1 [1]) que ceux obtenus avec la distribution uniforme comme distribution *a priori* sur l'espace des modèles et pour les probabilités de transition. La matrice L de similarité utilisée est la matrice de corrélations partielles entre les variables et on aborde le cas de la régression linéaire multiple et le cas de la régression logistique.

Dans le Chapitre 1, on présente les notions préliminaires qui serviront de base à la thèse. La Section 1.1 présente le DPP ainsi que ses principales propriétés, la Section 1.2 introduit les méthodes à noyaux qui servent de base à la construction de la matrice de similarité du DPP et la Section 1.3 présente quelques applications du DPP dans d'autres domaines. Le Chapitre 2 présente la méthodologie du partitionnement de données par consensus avec le DPP. Le Chapitre 3 détaille les deux approches adoptées pour résoudre le problème computationnel du partitionnement de données avec DPP pour des données avec un grand nombre d'observations. Le Chapitre 4 aborde la sélection de variables en régression linéaire et logistique dans un contexte de grande dimension, en utilisant le DPP comme distribution *a priori* de l'espace de tous les modèles et pour définir les probabilités de transition des chaînes de Markov.

1. Le score F_1 est une mesure utilisée pour évaluer la précision de classificateurs binaires (Chekouo et Murua, 2018). Il sera introduit et défini dans la Section 4.5.1

Bibliographie

T. Chekouo et A. Murua : High-dimensional variable selection with th eplaid mixture model for clustering. *Computational Statistics*, 33(3):1475–1496, 2018.

Luther T Clark, Laurence Watkins, Ileana L Piña, Mary Elmer, Ola Akinboboye, Millicent Gorham, Brenda Jamerson, Cassandra McCullough, Christine Pierre, Adam B Polis *et al.* : Increasing diversity in clinical trials : overcoming critical barriers. *Current Problems in Cardiology*, 44(5):148–172, 2019.

Abhirup Datta, Sudipto Banerjee, Andrew O Finley et Alan E Gelfand : On nearest-neighbor gaussian process models for massive spatial data. *Wiley Interdisciplinary Reviews : Computational Statistics*, 8(5):162–171, 2016.

J Ben Hough, Manjunath Krishnapur, Yuval Peres, Bálint Virág *et al.* : Determinantal processes and independence. *Probability surveys*, 3:206–229, 2006.

Alex Kulesza et Ben Taskar : Determinantal point processes for machine learning. *arXiv preprint arXiv :1207.6083*, 2012.

Cristine H Legare : Cumulative cultural learning - Development and diversity. *Proceedings of the National Academy of Sciences*, 114(30):7877–7883, 2017.

Stuart P. Lloyd : Least squares quantization in pcm. *IEEE Trans. Inf. Theory*, 28:129–136, 1982.

Stefano Monti, Pablo Tamayo, Jill Mesirov et Todd Golub : Consensus clustering : a resampling-based method for class discovery and visualization of gene expression microarray data. *Machine learning*, 52(1-2):91–118, 2003.

Anthony O'Hagan et Jonathan J Forster : *Kendall's advanced theory of statistics, volume 2B : Bayesian inference*, volume 2. Arnold, 2004.

Leonard Kaufman Peter J Rdusseeun et PJ Kaufman : Clustering by means of medoids. 1987.

Sandro Vega-Pons et José Ruiz-Shulcloper : A survey of clustering ensemble algorithms. *International Journal of Pattern Recognition and Artificial Intelligence*, 25 (03):337–372, 2011.

Iris R Wagstaff et Gerald LaPorte : The importance of diversity and inclusion in the forensic sciences. *National Institute of Justice Journal*, 279:81–91, 2018.

Giacomo Zanella : Informed proposals for local mcmc in discrete spaces. *Journal of the American Statistical Association*, pages 1–27, 2019.

Chapitre 1

Notions préliminaires

1.1. Le processus ponctuel déterminantal

Définition 1.1.1. *Soit un ensemble fini de taille $n \geq 2$, $\mathcal{S} = \{x_1, \ldots, x_n\} \subset \mathbb{R}^p$, où p est la dimension des vecteurs $x_i \in \mathcal{S}, i = 1, \ldots, n$, et considérons la variable aléatoire $\boldsymbol{Y}$ qui représente un sous-ensemble sélectionné aléatoirement de l'ensemble puissance $2^{\mathcal{S}}$. Un processus ponctuel déterminantal de matrice noyau L est une mesure de probabilité sur $2^{\mathcal{S}}$ si, pour tout $Y \in 2^{\mathcal{S}}$, sa fonction de masse de probabilité est donnée par*

$$P(\boldsymbol{Y} = Y) = \det(L_Y)/\det(L + I_n). \tag{1}$$

L est une matrice de taille $n \times n$, réelle, symétrique et semi-définie positive désignée par matrice noyau du processus, L_Y est la sous-matrice de L indexée par les lignes et colonnes de Y, i.e., $L_Y = [L_{ij}]_{i,j \in Y}$ et I_n est la matrice identité de taille $n \times n$. On dit alors que $\boldsymbol{Y} \sim DPP_{\mathcal{S}}(L)$.

Kulesza et Taskar (2012) démontrent que $\det(L + I_n) = \sum_{Y \subset 2^{\mathcal{S}}} \det(L_Y)$ et donc, l'expression (1) définit effectivement une fonction de probabilité sur l'ensemble $2^{\mathcal{S}}$. Plusieurs auteurs (Kulesza et Taskar, 2012; Affandi *et al.*, 2012) préfèrent définir le DPP avec une matrice noyau marginale qui permet d'obtenir la probabilité que certains éléments de $\mathcal{S}$ fassent partie de $\boldsymbol{Y}$. Dans ce chapitre, le fait qu'une matrice A soit semi-définie positive sera noté $A \succeq 0$.

Définition 1.1.2. *Soit un ensemble fini de taille $n \geq 2$, $\mathcal{S} = \{x_1, \ldots, x_n\} \subset \mathbb{R}^p$, où p est la dimension des vecteurs $x_i \in \mathcal{S}, i = 1, \ldots, n$, et considérons la variable aléatoire $\boldsymbol{Y}$ qui représente un sous-ensemble sélectionné aléatoirement de l'ensemble puissance $2^{\mathcal{S}}$. Un processus ponctuel déterminantal de matrice noyau marginale $\mathcal{K}$ est une mesure de probabilité sur $2^{\mathcal{S}}$ si, pour tout $A \subseteq \mathcal{S}$, sa fonction de masse de probabilité est donnée par*

$$P\left(A \subseteq \boldsymbol{Y}\right) = \det(\mathcal{K}_A),$$

où $\mathcal{K}$ est une matrice de taille $n \times n$, réelle, symétrique et semi-définie positive, désignée par matrice noyau marginale du processus et $\mathcal{K}_A$ est la sous-matrice de $\mathcal{K}$ indexée par les lignes et colonnes de A, i.e., $\mathcal{K}_A = [\mathcal{K}_{ij}]_{i,j\in A}$.

Kulesza et Taskar (2012) établissent la relation entre les matrices L and $\mathcal{K}$ des Définitions 1.1.1 et 1.1.2 par le théorème suivant :

Théorème 1.1.3. *Si $\boldsymbol{Y} \sim DPP_{\mathcal{S}}(L)$, alors,*

$$\mathcal{K} = L(L + I_n)^{-1} = I_n - (L + I_n)^{-1}.$$

La principale caractéristique du DPP est de pouvoir modéliser la corrélation négative entre deux éléments $x_i, x_j \in \mathcal{S}$. L'intensité de cette corrélation négative est capturée par la matrice noyau du processus, qui définit une mesure de similarité entre chaque paire d'éléments. Cela implique que deux éléments de $\mathcal{S}$ très similaires auront une plus faible probabilité de figurer dans le même sous-ensemble sélectionné dans $\mathcal{S}$. Le DPP privilégie donc des sous-ensembles caractérisés par une plus grande diversité parmi les éléments sélectionnés. Considérons par exemple la matrice noyau marginale $\mathcal{K}$ et $A = \{x_i,x_j\}$. Alors,

$$\begin{aligned} P\left(A \subseteq \boldsymbol{Y}\right) &= \begin{vmatrix} k_{ii} & k_{ij} \\ k_{ij} & k_{jj} \end{vmatrix} \\ &= k_{ii}k_{jj} - k_{ij}^2 \\ &= P\left(x_i \subseteq \boldsymbol{Y}\right) P\left(x_j \subseteq \boldsymbol{Y}\right) - k_{ij}^2. \end{aligned} \tag{2}$$

L'expression (2) montre que les coefficients hors de la diagonale principale de $\mathcal{K}_A$ définissent la corrélation négative entre x_i et x_j. Plus la valeur de k_{ij} est élevée, plus basse sera la probabilité que x_i et x_j figurent dans le même sous-ensemble sélectionné dans $\mathcal{S}$. En considérant que la matrice noyau du DPP représente la similarité entre chaque paire d'éléments de $\mathcal{S}$, deux éléments similaires auront une probabilité plus faible d'être sélectionnés ensemble.

La fonction de masse présentée en Définition 1.1.1 a également une interprétation géométrique. Étant donné que la matrice L est semi-définie positive, elle admet une décomposition $L = B^T B$, où B est une matrice de taille $m \times n$. Si l'on désigne la colonne j de B par $B_{\bullet j}$, alors

$$P\left(\boldsymbol{Y} = Y\right) \propto \det\left(L_Y\right) = \mathrm{Vol}^2\left(\left\{B_{\bullet j}\right\}_{j\in Y}\right), \tag{3}$$

où $\mathrm{Vol}\Big(\left\{B_{\bullet j}\right\}_{j\in Y}\Big)$ représente le volume de l'hyper-parallélépipède engendré par les colonnes de B indexées par les éléments de Y. Chaque colonne de B peut donc être interprétée

comme un vecteur de caractéristiques qui décrit chaque élément de $\mathcal{S}$. La matrice L décrit la similarité entre chacun de ces vecteurs au moyen d'un produit scalaire. Une sélection d'éléments de $\mathcal{S}$ très similaires engendrera un hyper-parallélépipède dont le volume sera très petit. Au contraire, si les éléments sont dissimilaires (et donc suffisamment diversifiés), le volume de l'hyper-parallélépipède sera plus élevé.

Même si les Définitions 1.1.1 et 1.1.2 sont équivalentes, beaucoup d'auteurs préfèrent utiliser la Définition 1.1.1 (Kulesza et Taskar, 2012; Kang, 2013; Affandi *et al.*, 2014), étant donné qu'elle offre une structure plus flexible. En effet,

i) La Définition 1.1.1 offre la possibilité de modéliser directement la probabilité individuelle de sélectionner chaque sous-ensemble de $\mathcal{S}$. La Définition 1.1.2 ne permet pas de calculer directement cette probabilité : elle calcule la probabilité que les éléments qui figurent dans $A \subseteq \mathcal{S}$ apparaissent dans le même sous-ensemble $\boldsymbol{Y}$ sélectionné aléatoirement dans $\mathcal{S}$ (Rising, 2013).

ii) La Définition 1.1.1 impose moins de restrictions sur la matrice noyau correspondante que la Définition 1.1.2. Tandis que n'importe quelle matrice semi-définie positive peut être choisie comme matrice noyau L ($L \succeq 0$), le choix de la matrice noyau marginale $\mathcal{K}$ est plus limité. Toute matrice noyau marginale doit satisfaire $\mathcal{K} \succeq 0$ et aussi $I_n - \mathcal{K} \succeq 0$. Cela équivaut à dire que $\mathcal{K}$ doit satisfaire $0 \preceq \mathcal{K} \preceq I_n$ (Rising, 2013).

Le point ii) découle directement de la Définition 1.1.2. Celle-ci impose une restriction sur tous les mineurs principaux de $\mathcal{K}$, désignés par $\det(\mathcal{K}_A)$. Étant donné que $P(A \subseteq \boldsymbol{Y}) = \det(\mathcal{K}_A)$ est une mesure de probabilité, on a

$$0 \leq \det(\mathcal{K}_A) \leq 1, \tag{4}$$

pour tout $A \subseteq \mathcal{S}$. La condition (4) implique directement le fait que $\mathcal{K}$ doit satisfaire $\mathcal{K} \succeq 0$. Si, en particulier, A est un singleton, i.e. $A = \{x_i\},\ i = 1, \ldots, n$, on a

$$P(x_i \in \boldsymbol{Y}) = \mathcal{K}_{ii},$$

ce qui montre que les coefficients de la diagonale de $\mathcal{K}$ donnent les probabilités que chaque élément de $\mathcal{S}$ fasse partie d'un sous-ensemble $\boldsymbol{Y}$ sélectionné aléatoirement. Ceci implique donc que $0 \leq \mathcal{K}_{ii} \leq 1$. Le théorème suivant, démontré dans Kulesza et Taskar (2012), est utile pour établir que $\mathcal{K}$ doit également satisfaire la condition $I_n - \mathcal{K} \succeq 0$ évoquée au point ii) :

Théorème 1.1.4. *Si $\boldsymbol{Y}$ est distribué selon un DPP avec matrice noyau marginale $\mathcal{K}$, alors $\mathcal{S} - \boldsymbol{Y}$ est aussi distribué selon un DPP, avec matrice noyau marginale $\overline{\mathcal{K}} = I_n - \mathcal{K}$.*

Par conséquent, comme on a

$$P(A \cap \boldsymbol{Y} = \emptyset) = \det(\overline{\mathcal{K}}_A) = \det(I_n - \mathcal{K}_A),$$

pour tout $A \subseteq \mathcal{S}$, cette mesure de probabilité entraîne la condition

$$0 \leq \det(I_n - \mathcal{K}_A) \leq 1,$$

et donc on que $I_n - \mathcal{K} \succeq 0$, ce qui équivaut à dire que $\mathcal{K} \preceq I_n$.

La contrainte $0 \preceq \mathcal{K} \preceq I_n$ discutée au point ii) a aussi un impact sur les valeurs propres de $\mathcal{K}$. Selon le Théorème de Weyl que l'on trouve en page 239 de Horn et Johnson (2012), on démontre la proposition suivante :

Proposition 1. *Considérons deux matrices symétriques A et B. Si $A \succeq B$, alors*

$$\lambda_i(A) \geq \lambda_i(B),$$

pour tout i, où $\lambda_i(\cdot)$ représente la i-ème plus grande valeur propre.

La démonstration de la Proposition 1 se trouve en appendice à la fin de ce chapitre.

Comme $\mathcal{K}$ et I_n sont des matrices symétriques, on a que

$$\lambda_i(\mathcal{K}) \leq \lambda_i(I_n), \quad i = 1, \ldots, n.$$

Pour la matrice identité, on a $\lambda_i(I_n) = 1$, pour tout i, et comme on a $\mathcal{K} \succeq 0$, ses valeurs propres satisfont $0 \leq \lambda_i(\mathcal{K}) \leq 1$, pour tout i.

La Définition 1.1.1 impose aussi des restrictions sur tous les mineurs principaux de la matrice noyau L, désignés par $\det(L_Y)$. Cependant, ces restrictions sont moins fortes que celles imposées sur $\mathcal{K}$. En effet, comme $P(\boldsymbol{Y} = Y) \propto \det(L_Y)$ représente une mesure de probabilité, on a $\det(L_Y) \geq 0$, pour tout $Y \subseteq \mathcal{S}$. Puisque $P(\boldsymbol{Y} = Y) \propto \det(L_Y)$, il n'est pas nécessaire que $\det(L_Y) \leq 1$. Cela implique donc que $L \succeq 0$ et donc, tout matrice semi-définie positive est candidate pour L, avec valeurs propres telles que $\lambda_i(L) \geq 0$, pour tout i.

Il est quand même important de préciser que même si les Définitions 1.1.1 et 1.1.2 sont équivalentes et même si le Théorème 1.1.3 indique qu'il existe une correspondance entre L et $\mathcal{K}$, cela n'est pas toujours le cas. Comme on a $L \succeq 0$, on a $\lambda_i(L) \geq 0, \quad i = 1, \ldots, n$. De l'algèbre linéaire, on sait que si A est une matrice réelle de taille $n \times n$ et de valeurs

propres $\lambda_1(A), \lambda_2(A), \ldots, \lambda_n(A)$, en incluant leur multiplicité algébrique, alors la matrice réelle $aA + bI_n$, de taille $n \times n$, a pour valeurs propres $a\lambda_1(A) + b, a\lambda_2(A) + b, \ldots, a\lambda_n(A) + b$, avec a,$b \in \mathbb{R}$. La matrice $L + I_n$ du Théorème 1.1.3 est telle que

$$\lambda_i(L + I_n) \geq 1, \quad i = 1, \ldots, n,$$

ce qui rend cette matrice inversible. Ceci garantit que toute matrice noyau L a une unique matrice noyau marginale $\mathcal{K}$ qui lui correspond. La condition réciproque n'est pas forcément vraie. En effet, si l'on inverse l'équation du Théorème 1.1.3, on obtient

$$L = \mathcal{K}(I_n - \mathcal{K})^{-1},$$

et par conséquent, si $\mathcal{K}$ a pour valeurs propres $\lambda_1(\mathcal{K}), \lambda_2(\mathcal{K}), \ldots, \lambda_n(\mathcal{K})$, alors les valeurs propres de $I_n - \mathcal{K}$ sont données par $1 - \lambda_1(\mathcal{K}), 1 - \lambda_2(\mathcal{K}), \ldots, 1 - \lambda_n(\mathcal{K})$. Comme on a $0 \leq \lambda_i(\mathcal{K}) \leq 1$ pour tout i, s'il existe un i tel que $\lambda_i(\mathcal{K}) = 1$, on a $\lambda_i(I_n - \mathcal{K}) = 0$ rendant la matrice $I_n - \mathcal{K}$ singulière. Pour toute matrice noyau marginale $\mathcal{K}$ satisfaisant $0 \preceq \mathcal{K} \preceq I_n$, il n'existe pas de matrice noyau L correspondante lorsque $\lambda_i(\mathcal{K}) = 1$, pour au moins un i.

Pour toutes les raisons exposées dans cette section, on utilise donc la Définition 1.1.1 du DPP tout au long de cette thèse.

Il est également possible de générer des échantillons du DPP. Pour cela, Hough *et al.* (2006) et Kulesza et Taskar (2012) ont créé un algorithme efficace pour obtenir de tels échantillons. Cet algorithme est basé sur le théorème suivant, démontré par les auteurs :

Théorème 1.1.5. *Soient $\lambda_1(L), \lambda_2(L), \ldots, \lambda_n(L)$ les valeurs propres de la matrice noyau L. La variable aléatoire* $\text{card}(\boldsymbol{Y})$*, représentant le nombre d'éléments dans $\boldsymbol{Y}$, est distribuée selon le nombre de succès en n essais indépendants de Bernoulli, où l'essai $i = 1, \ldots, n$ est un succès avec probabilité $\lambda_i(L)/(\lambda_i(L) + 1)$. De plus, on a*

$$\begin{aligned} E\left[\text{card}(\boldsymbol{Y})\right] &= \sum_{i=1}^{n} \lambda_i(L)/(\lambda_i(L) + 1) = tr(\mathcal{K}) \\ Var\left[\text{card}(\boldsymbol{Y})\right] &= \sum_{i=1}^{n} \lambda_i(L)/\left((\lambda_i(L) + 1)^2\right). \end{aligned}$$

Les auteurs démontrent donc que si $L = \sum_{i=1}^{n} \lambda_i(L) v_i v_i^T$ est une décomposition spectrale de L, alors, l'algorithme basé sur le Théorème 1.1.5 génère des échantillons de $\boldsymbol{Y} \sim DPP_{\mathcal{S}}(L)$. Les principales étapes de l'algorithme sont les suivantes :

(1) Obtenir la décomposition spectrale de L, $L = \sum_{i=1}^{n} \lambda_i(L) v_i v_i^T$;

(2) Construire $J \subseteq \{1, 2, \ldots, n\}$ et le sous-ensemble de vecteurs propres $V_J = \{v_i : i \in J\}$, où v_i est sélectionné avec probabilité $\lambda_i(L)/(\lambda_i(L) + 1)$;

(3) Pour $k = 1,2, \ldots, |V_J|$, répéter les étapes suivantes :

   (a) sélectionner $i_k \in \{1, 2, \ldots, n\}$ avec probabilité $P_{i_k} = 1/|V_J| \sum_{v \in V_J} \left(v^T e_{i_k}\right)^2$, où e_{i_k} représente le vecteur correspondant de la base canonique de $\mathbb{R}^n$;

   (b) calculer $V_{J\perp}$, qui représente une base orthonormale du sous-espace de V_J orthogonal à e_{i_k} ;

   (c) mettre à jour V_J par $V_J = V_{J\perp}$;

(4) Construire $Y = \{i_k, k = 1, 2, \ldots, |V_J|\}$, réalisation de la variable aléatoire $\boldsymbol{Y}$.

L'algorithme de Hough *et al.* (2006) et Kulesza et Taskar (2012) est utilisé tout au long de cette thèse. Il sera entièrement implémenté et codé avec le langage de programmation Julia, version 1.1.1.

1.2. L'astuce du noyau

On a vu que toute matrice semi-définie positive pouvait être choisie comme matrice noyau L du DPP. Pour cette thèse, un choix bien particulier a été adopté comme matrice noyau, en tirant profit de la décomposition $L = B^T B$ sous-jacente à l'interprétation géométrique du DPP donnée par (3). Étant donné que le produit scalaire est une mesure usuelle de similarité entre deux vecteurs, cela établit une relation directe avec les méthodes à noyaux, largement utilisées en apprentissage automatique (Howley et Madden, 2006). Les noyaux sont utilisés comme mesures de similarité entre éléments d'un certain espace. Cette similarité est établie en transformant les données de l'espace original dans un espace de dimension supérieure, de façon à établir des relations linéaires dans ce nouvel espace. La mesure de similarité la plus communément utilisée par les méthodes à noyaux est le produit scalaire. L'avantage de cette transformation est de pouvoir exprimer des relations linéaires dans un espace transformé quand elles sont non linéaires dans l'espace original.

Définition 1.2.1. *Soit un ensemble fini de taille $n \geq 2$, $\mathcal{S} = \{x_1, \ldots, x_n\} \subset \mathbb{R}^p$, où p est la dimension des vecteurs $x_i \in \mathcal{S}, i = 1, \ldots, n$, et considérons un espace $\mathcal{H}$ de redescription de Hilbert. En transformant $x_i \in \mathcal{S}$, $i = 1, \ldots, n$, au moyen d'une transformation $\boldsymbol{\Phi} : \mathcal{S} \to \mathcal{H}$, on obtient l'ensemble*

$$\boldsymbol{\Phi}(\mathcal{S}) = \{\boldsymbol{\Phi}(x_1), \boldsymbol{\Phi}(x_2), \ldots, \boldsymbol{\Phi}(x_n)\},$$

où $\boldsymbol{\Phi}(x_i) = (\phi_1(x_i), \phi_2(x_i), \ldots)$.

La Définition 1.2.1 montre que chaque élément de $\mathcal{S}$ est transformé dans un espace de grande dimension, possiblement infini. Cette particularité peut rendre impraticable la manipulation directe des éléments transformés. En réponse à ce problème, les méthodes à noyaux préfèrent travailler avec des mesures de similarité entre les éléments de l'ensemble $\mathbf{\Phi}(\mathcal{S})$, sans utiliser explicitement les éléments transformés $\mathbf{\Phi}(x_i),\ i = 1,\ldots,n$ (Schölkopf *et al.*, 2004). Les similarités entre les éléments de $\mathbf{\Phi}(\mathcal{S})$ sont ensuite utilisées dans les algorithmes d'apprentissage automatique. Étant donné que l'espace de redescription est un espace de Hilbert, le produit scalaire est la mesure de similarité la plus commune. Ce produit scalaire entre les éléments de $\mathbf{\Phi}(\mathcal{S})$ peut être calculé directement à partir des éléments de l'espace original $\mathcal{S}$ en utilisant une fonction spéciale, la fonction noyau.

Définition 1.2.2. *Soit un ensemble fini de taille $n \geq 2$, $\mathcal{S} = \{x_1,\ldots,x_n\} \subset \mathbb{R}^p$, où p est la dimension des vecteurs $x_i \in \mathcal{S}, i = 1,\ldots,n$. Considérons un espace $\mathcal{H}$ de redescription de Hilbert muni du produit scalaire $\langle\cdot,\cdot\rangle_{\mathcal{H}} : \mathcal{H}\times\mathcal{H} \to \mathbb{R}$. Une fonction noyau $\kappa : \mathcal{S}\times\mathcal{S} \to \mathbb{R}$ est une fonction telle que*

$$\kappa(x_i,x_j) = \langle \mathbf{\Phi}(x_i), \mathbf{\Phi}(x_j)\rangle_{\mathcal{H}},\ i,j = 1,\ldots,n,$$

pour une certaine transformation $\mathbf{\Phi} : \mathcal{S} \to \mathcal{H}$.

La fonction noyau permet donc d'obtenir les produits scalaires entre les éléments de $\mathbf{\Phi}(\mathcal{S})$ directement à partir des éléments de l'espace original $\mathcal{S}$ sans devoir calculer explicitement les coordonnées des éléments transformés $\mathbf{\Phi}(x_i) = (\phi_1(x_i), \phi_2(x_i), \ldots),\ 1,\ldots,n$. Cette particularité est connue sous le nom d'astuce du noyau (Aizerman, 1964). La définition 1.2.2 soulève cependant un problème : toute fonction noyau κ doit impliquer obligatoirement l'existence d'une transformation $\mathbf{\Phi} : \mathcal{S} \to \mathcal{H}$. Le théorème suivant, connu sous le nom de Théorème de Mercer (Vapnik, 1995), définit les fonctions κ pouvant être considérées comme fonctions noyau :

Théorème 1.2.3. *(Mercer) Soit $\mathcal{X} \subset \mathbb{R}^p$ un sous-ensemble compact et soit $L^2(\mathcal{X})$ l'espace de Hilbert des fonctions de carré intégrable définies en $\mathcal{X} \to \mathbb{R}$. Soit $\kappa : \mathcal{S}\times\mathcal{S} \to \mathbb{R}$ une fonction symétrique continue qui satisfait*

$$\sum_{i=1}^{n}\sum_{j=1}^{n} c_i c_j \kappa(x_i,x_j) \geq 0,$$

pour tout $\{x_i\}_{i=1}^n \subset \mathcal{X}$ et $\{c_i\}_{i=1}^n \subset \mathbb{R}^n$, tel que l'opérateur intégral compact $L_\kappa : L^2(\mathcal{X}) \to L^2(\mathcal{X})$ défini par

$$L_\kappa f(x_i) = \int_{\mathcal{X}} \kappa(x_i,x_j) f(x_j) dx_j$$

soit non-négatif, i.e., pour toute fonction $f \in L^2(\mathcal{X})$,

$$\int_{\mathcal{X}\times\mathcal{X}} \kappa(x_i, x_j) f(x_i) f(x_j) dx_i dx_j \geq 0.$$

Alors, κ *admet une expansion en série uniformément convergente, de termes donnés par les valeurs propres non-négatives de* L_κ*, définies par* $\{\lambda_r\}_{r=1}^{+\infty}$ *et qui satisfont* $\sum_{r=1}^{+\infty} \lambda_r < +\infty$*, et leur fonctions propres associées, définies par* $\{\psi_r\}_{r=1}^{+\infty}$*, qui forment une base orthonormale de* $L^2(\mathcal{X})$*. Cela équivaut à dire que*

$$\kappa(x_i, x_j) = \sum_{r=1}^{+\infty} \lambda_r \psi_r(x_i) \psi_r(x_j),$$

pour tout $x_i, x_j \in \mathbb{R}^p$*.*

Le théorème de Mercer définit donc quelles fonctions peuvent être prises comme fonction noyau. Si une fonction satisfait le théorème de Mercer, alors cela implique qu'il existe toujours une transformation $\mathbf{\Phi} : \mathcal{S} \to \mathcal{H}$ que l'on peut définir de la façon suivante :

$$\mathbf{\Phi}(x_i) = \left(\sqrt{\lambda_1}\psi_1(x_i), \sqrt{\lambda_2}\psi_2(x_i), \dots \right), \ i = 1, \dots, n.$$

On dit alors que la fonction κ est un noyau de Mercer. Les noyaux de Mercer les plus courants sont les suivants :

i) Le noyau polynomial de degré $d \in \mathbb{N}^+$: $\kappa(x_i, x_j) = \left(x_i^T x_j + c\right)^d, \ c \geq 0$;

ii) Le noyau Gaussien : $\kappa(x_i, x_j) = \exp\left(-\left\|x_i - x_j\right\|^2 / (2\sigma^2)\right), \ \sigma^2 > 0$;

iii) Le noyau sigmoïde : $\kappa(x_i, x_j) = \tanh\left(a\left(x_i^T x_j\right) + b\right), \ a > 0, b < 0$;

iv) Le noyau α-stable (Bilodeau et Nangue, 2017) : $\kappa(x_i, x_j) = \exp\left(-\beta^\alpha \left\|x_i - x_j\right\|^\alpha\right)$, $\alpha \in (0,2], \beta > 0$. Le cas $\alpha = 2$ correspond au noyau Gaussien et le cas $\alpha = 1$ correspond au noyau de Cauchy ou de Laplace.

Les noyaux de Mercer ont la propriété particulière d'être des noyaux semi-définis positifs :

Définition 1.2.4. *Une fonction* $\kappa : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ *est une fonction noyau semi-définie positive si et seulement si elle est symétrique, i.e.,*

$$\kappa(x_i, x_j) = \kappa(x_j, x_i),$$

pour tout $x_i, x_j \in \mathcal{X}$*, et aussi*

$$\sum_{i=1}^{n} \sum_{j=1}^{n} c_i c_j \kappa(x_i, x_j) \geq 0, \tag{5}$$

pour tout $\{x_i\}_{i=1}^n \subset \mathcal{X}$ *et* $\{c_i\}_{i=1}^n \in \mathbb{R}^n$*, quel que soit* $n \in \mathbb{N}$*.*

On retrouve la condition (5) dans le Théorème 1.2.3. Ceci assure que toute matrice de taille $n \times n$ définie par $K = [\kappa(x_i, x_j)]_{i,j=1}^n$ est semi-définie positive. En effet, si l'on considère (5) et que l'on définit $c^T = (c_1, c_2, \ldots, c_n)$, on a :

$$\sum_{i=1}^{n}\sum_{j=1}^{n} c_i c_j \kappa(x_i, x_j) \geq 0 \quad \Leftrightarrow \quad c^T K c \geq 0,$$

qui est la définition d'une matrice K semi-définie positive.

Les noyaux de Mercer sont donc de parfaits candidats pour construire la matrice noyau L du DPP, présentée dans la Définition 1.1.1 et adoptée dans cette thèse. En effet, les noyaux de Mercer représentent des mesures de similarité entre vecteurs, où une valeur plus élevée signifie une plus grande similarité dans l'espace de redescription $\mathcal{H}$. On a donc

$$L = [\kappa(x_i, x_j)]_{i,j=1}^n .$$

1.3. Applications du processus ponctuel déterminantal

Dans cette section, on illustre quelques applications du DPP trouvées dans la littérature. On retrouve le DPP dans les domaines suivants : théorie des probabilités, matrices aléatoires, analyse combinatoire, apprentissage automatique et modélisation de réseaux sans fil, entre autres.

L'utilisation du DPP en théorie des probabilités apparaît par exemple dans le domaine des marches aléatoires sans croisement (Karlin *et al.*, 1959). Johansson (2004) démontre que si l'on considère un ensemble de marches aléatoires simples, indépendantes, symétriques et qui ne se croisent jamais, l'ensemble formé par les positions de chaque marche à un certain moment t est un sous-ensemble de $\mathbb{Z}$ distribué selon un DPP. Une autre contribution en théorie des probabilités est due à Borodin (2008) qui démontre que si l'on définit une chaîne de Markov sans cycles sur un espace discret, on peut considérer celle-ci comme un DPP. En particulier, soit une mesure de probabilité $\mathcal{P}$ sur un espace discret $\mathcal{X}$ et soit une chaîne de Markov avec distribution initiale $\mathcal{P}$ et matrice de transition T. Considérons une certaine trajectoire comme étant un sous-ensemble de $\mathcal{X}$ avec mesure de probabilité définie par la chaîne de Markov. Cette mesure de probabilité sur $2^{\mathcal{X}}$ est un DPP. Les arbres aléatoires ont aussi un lien étroit avec le DPP. Hough *et al.* (2009) montrent que si l'on considère un graphe avec N arêtes et que l'on choisit un arbre couvrant aléatoire (Burton et Pemantle, 1993) parmi l'ensemble de tous les arbres couvrants, les arêtes de l'arbre sélectionné forment un sous-ensemble des arêtes du graphe qui se distribue selon un DPP.

En théorie des matrices aléatoires, on retrouve une littérature abondante dans le domaine (Mehta et Gaudin, 1960; Evans et Gottlieb, 2009; Johansson, 2005), principalement dans la détermination de la densité des valeurs propres d'une matrice aléatoire. Les principales contributions sont dues à Mehta et Gaudin (1960) et Ginibre (1965). Les auteurs démontrent

que les valeurs propres d'une matrice aléatoire M, dont les coefficients sont obtenus à partir d'une distribution Normale complexe, se distribuent selon un DPP. Les valeurs propres de M forment un sous-ensemble du plan complexe. Plus récemment, Adhikari *et al.* (2016) démontrent que cette propriété se maintient dans le cas du produit de matrices aléatoires. Les valeurs propres d'une matrice issue du produit de plusieurs matrices aléatoires, dont les coefficients sont générés d'une distribution Normale complexe, se distribuent aussi selon un DPP. Pour d'autres exemples de matrices aléatoires dont les valeurs propres sont distribuées selon un DPP, Soshnikov (2000) est une excellente référence.

L'analyse combinatoire utilise également le DPP dans la résolution de certains problèmes. Le plus fameux d'entre eux est le diamant aztèque (Johansson *et al.*, 2005). Dans ce problème, la moitié des carrés du diamant sont colorés et les autres sont laissés en blanc. On recouvre alors complètement le diamant avec des dominos, en les plaçant horizontalement et verticalement. Supposons qu'il y a N façons possibles de recouvrir ce diamant avec les dominos. Considérons l'expérience qui consiste à choisir au hasard un des recouvrements possibles parmi les N recouvrements. On construit un sous-ensemble avec les carrés colorés du diamant sélectionné qui se trouvent soit sur la moité gauche d'un domino placé horizontalement, soit sur la moitié inférieure d'un domino placé verticalement. Ce sous-ensemble se distribue selon un DPP. Chhita *et al.* (2015) considèrent également l'expérience où différents poids sont attribués aux dominos horizontaux et verticaux. Le DPP est également utilisé comme distribution du sous-ensemble de points colorés.

L'apprentissage automatique est sans aucun doute le domaine de prédilection du DPP. Deux références centrales dans le domaine sont Kulesza et Taskar (2012) et Lavancier *et al.* (2015). L'application principale du DPP consiste à extraire des éléments à partir d'un corpus en basant sur la diversité des éléments sélectionnés. Kulesza et Taskar (2012) donnent l'exemple de la technique de résumé de documents. Les auteurs établissent un algorithme basé sur le DPP qui permet d'extraire les parties-clé du document afin de construire un court texte qui résume les idées principales du document. Les auteurs élargissent également le champ d'action du DPP aux moteurs de recherche d'images. En attribuant un score de pertinence aux diverses images présentes sur un certain moteur de recherche, ils définissent une mesure de similarité entre images et utilisent le DPP pour sélectionner les images les plus pertinentes et diversifiées lors d'une recherche par mots-clé. L'algorithme d'extraction est perfectionné par Wang et Chan (2019), qui combinent le DPP avec de l'apprentissage par renforcement. Selon Belhadji *et al.* (2018), un des problèmes récurrents en apprentissage automatique est la réduction de la dimensionnalité d'un pipeline d'apprentissage automatique. Les méthodes usuelles utilisées sont l'analyse en composantes principales ou la sélection de caractéristiques. Les auteurs proposent une méthode d'extraction de variables basée sur le DPP comme alternative à ces deux méthodes et démontrent que la méthode basée sur le DPP est supérieure aux deux autres sous un certain critère d'optimalité.

Finalement, dans le domaine de la modélisation de réseaux sans fil, on trouve la contribution de Miyoshi et Shirai (2014). La performance des réseaux sans fil est fortement liée à la répartition spatiale des senseurs qui peut être capturée par un processus ponctuel spatial. La pratique courante est de supposer que la distribution spatiale des senseurs suit un processus ponctuel de Poisson homogène. Les auteurs proposent d'utiliser le DPP pour modéliser cette distribution spatiale, afin de favoriser la répulsion entre les différentes stations et maximiser la couverture du réseau. D'autres recherches importantes dans le domaine sont également rapportées par Deng *et al.* (2014) et Torrisi et Leonardi (2014).

Appendice

Pour démontrer la Proposition 1, commençons par énoncer le Théorème de Weyl, que l'on trouve en page 239 de Horn et Johnson (2012), dans le cas de matrices réelles :

Théorème 1.3.1. *(Weyl) Soient A et B deux matrices réelles de taille $n \times n$ et considérons les valeurs propres de A, B et $A+B$ données par $\{\lambda_j(A)\}_{j=1}^n$, $\{\lambda_j(B)\}_{j=1}^n$ et $\{\lambda_j(A+B)\}_{j=1}^n$, respectivement, où chaque ensemble de valeurs propres est ordonné algébriquement, i.e.,*

$$\lambda_{\min} = \lambda_1 \leq \lambda_2 \leq \cdots \leq \lambda_{n-1} \leq \lambda_n = \lambda_{\max}.$$

Alors,

$$\lambda_j(A+B) \leq \lambda_{j+\ell}(A) + \lambda_{n-\ell}(B), \quad \ell = 0, 1, \ldots, n-j, \tag{6}$$

pour tout $j = 1, \ldots, n$.

Si on prend $B = -B$ dans l'inéquation (6), on obtient, pour $\ell = 0, 1, \ldots, n-j$ et tout $j = 1, \ldots, n$,

$$\begin{aligned} \lambda_j(A-B) &\leq \lambda_{j+\ell}(A) + \lambda_{n-\ell}(-B) \Leftrightarrow \\ \Leftrightarrow \lambda_j(A-B) &\leq \lambda_{j+\ell}(A) - \lambda_{1+\ell}(B), \end{aligned} \tag{7}$$

en utilisant les propriétés des valeurs propres.

Comme on a $A \succeq B$ selon la Proposition 1, cela équivaut à dire que $A - B \succeq 0$, ce qui implique que $\lambda_j(A-B) \geq 0$, $j = 1, \ldots, n$. En reprenant (7), on a

$$0 \leq \lambda_j(A-B) \leq \lambda_{j+\ell}(A) - \lambda_{1+\ell}(B), \quad \ell = 0, 1, \ldots, n-j,$$

pour tout $j = 1, \ldots, n$. Et donc, par transitivité,

$$\begin{aligned} \lambda_{j+\ell}(A) - \lambda_{1+\ell}(B) &\geq 0 \Leftrightarrow \\ \Leftrightarrow \lambda_{j+\ell}(A) &\geq \lambda_{1+\ell}(B). \end{aligned} \tag{8}$$

En prenant en particulier $j = 1$ en (8), on obtient

$$\lambda_{1+\ell}(A) \geq \lambda_{1+\ell}(B), \quad \ell = 0, 1, \ldots, n-1. \tag{9}$$

Finalement, en posant $i = 1+\ell$ en (9), on obtient l'inégalité de la Proposition 1 à démontrer :

$$\lambda_i(A) \geq \lambda_i(B), \quad i = 1, \ldots, n.$$

Bibliographie

Kartick Adhikari, Nanda Kishore Reddy, Tulasi Ram Reddy, Koushik Saha *et al.* : Determinantal point processes in the plane from products of random matrices. *In Annales de l'Institut Henri Poincaré, Probabilités et Statistiques*, volume 52, pages 16–46. Institut Henri Poincaré, 2016.

Raja Hafiz Affandi, Emily Fox, Ryan Adams et Ben Taskar : Learning the parameters of determinantal point process kernels. *In International Conference on Machine Learning*, pages 1224–1232, 2014.

Raja Hafiz Affandi, Alex Kulesza et Emily B Fox : Markov determinantal point processes. *arXiv preprint arXiv :1210.4850*, 2012.

Mark A Aizerman : Theoretical foundations of the potential function method in pattern recognition learning. *Automation and remote control*, 25:821–837, 1964.

Ayoub Belhadji, Rémi Bardenet et Pierre Chainais : A determinantal point process for column subset selection. *arXiv preprint arXiv :1812.09771*, 2018.

Martin Bilodeau et Aurélien Guetsop Nangue : Tests of mutual or serial independence of random vectors with applications. *The Journal of Machine Learning Research*, 18 (1):2518–2557, 2017.

Alexei Borodin : Loop-free markov chains as determinantal point processes. *In Annales de l'IHP Probabilités et statistiques*, volume 44, pages 19–28, 2008.

Robert Burton et Robin Pemantle : Local characteristics, entropy and limit theorems for spanning trees and domino tilings via transfer-impedances. *The Annals of Probability*, pages 1329–1371, 1993.

Sunil Chhita, Kurt Johansson, Benjamin Young *et al.* : Asymptotic domino statistics in the aztec diamond. *The Annals of Applied Probability*, 25(3):1232–1278, 2015.

Na Deng, Wuyang Zhou et Martin Haenggi : The ginibre point process as a model for wireless networks with repulsion. *IEEE Transactions on Wireless Communications*, 14 (1):107–121, 2014.

Steven N Evans et Alex Gottlieb : Hyperdeterminantal point processes. *Metrika*, 69 (2-3):85–99, 2009.

Jean Ginibre : Statistical ensembles of complex, quaternion, and real matrices. *Journal of Mathematical Physics*, 6(3):440–449, 1965.

Roger A. Horn et Charles R. Johnson : *Matrix Analysis*. Cambridge University Press, USA, 2nd édition, 2012. ISBN 0521548233.

J Ben Hough, Manjunath Krishnapur, Yuval Peres, Bálint Virág *et al.* : Determinantal processes and independence. *Probability surveys*, 3:206–229, 2006.

John Ben Hough, Manjunath Krishnapur, Yuval Peres *et al.* : *Zeros of Gaussian analytic functions and determinantal point processes*, volume 51. American Mathematical Soc.,

2009.

Tom Howley et Michael G Madden : An evolutionary approach to automatic kernel construction. *In International Conference on Artificial Neural Networks*, pages 417–426. Springer, 2006.

K Johansson : Determinantal processes with number variance saturation, 2001 commun. *Math. Phys*, 252:1–3, 2004.

Kurt Johansson : Random matrices and determinantal processes. *arXiv preprint math-ph/0510038*, 2005.

Kurt Johansson *et al.* : The arctic circle boundary and the airy process. *The Annals of Probability*, 33(1):1–30, 2005.

Byungkon Kang : Fast Determinantal Point Process Sampling with Application to Clustering. *In* C.J.C. Burges, L. Bottou, M. Welling, Z. Ghahramani et K.Q. Weinberger, éditeurs : *Advances in Neural Information Processing Systems 26*, pages 2319–2327. Curran Associates, Inc., 2013.

Samuel Karlin, James McGregor *et al.* : Coincidence probabilities. *Pacific Journal of Mathematics*, 9(4):1141–1164, 1959.

Alex Kulesza et Ben Taskar : Determinantal point processes for machine learning. *arXiv preprint arXiv :1207.6083*, 2012.

Frédéric Lavancier, Jesper Møller et Ege Rubak : Determinantal point process models and statistical inference. *Journal of the Royal Statistical Society : Series B : Statistical Methodology*, pages 853–877, 2015.

Madan Lal Mehta et Michel Gaudin : On the density of eigenvalues of a random matrix. *Nuclear Physics*, 18:420–427, 1960.

Naoto Miyoshi et Tomoyuki Shirai : A cellular network model with ginibre configured base stations. *Advances in Applied Probability*, 46(3):832–845, 2014.

J.K. Rising : *Advances in the Theory of Determinantal Point Processes*. University of Pennsylvania, 2013.

Bernhard Schölkopf, Koji Tsuda et Jean-Philippe Vert : *Kernel methods in computational biology*. MIT press, 2004.

Alexander Soshnikov : Determinantal random point fields. *Russian Mathematical Surveys*, 55(5):923, 2000.

Giovanni Luca Torrisi et Emilio Leonardi : Large deviations of the interference in the ginibre network model. *Stochastic Systems*, 4(1):173–205, 2014.

Vladimir N. Vapnik : *The Nature of Statistical Learning Theory*. Springer-Verlag, Berlin, Heidelberg, 1995. ISBN 0387945598.

Qingzhong Wang et Antoni B Chan : Towards diverse and accurate image captions via reinforcing determinantal point process. *arXiv preprint arXiv :1908.04919*, 2019.

Chapitre 2

Determinantal consensus clustering

par

Sergio Vicente[1] et Alejandro Murua[2]

([1]) Département de Mathématiques et de Statistique, Université de Montréal.
E-mail : vicentes@dms.umontreal.ca

([2]) Département de Mathématiques et de Statistique, Université de Montréal.
E-mail : murua@dms.umontreal.ca

Cet article sera soumis dans Advances in Data Analysis and Classification.

Résumé. La réinitialisation aléatoire d'un certain algorithme génère plusieurs partitions qui produisent un partitionnement par consensus. Les méthodes d'apprentissage ensemblistes, comme le partitionnement par consensus, sont reconnues comme des méthodes de partitionnement plus robustes qu'une simple exécution d'un algorithme de partitionnement. On propose le processus ponctuel déterminantal ou DPP pour la réinitialisation d'algorithmes de partitionnement basés sur un ensemble initial de centroïdes, comme les k-médoïdes ou les k-moyennes. La relation entre le DPP et les méthodes à noyaux permet son utilisation pour décrire et quantifier la similarité entre objets. Le DPP privilégie la diversité des centroïdes des sous-ensembles. Par conséquent, les sous-ensembles contenant beaucoup de points similaires seront moins susceptibles d'être générés que ceux qui contiennent des points très dissimilaires. L'approche classique la plus courante pour sélectionner les centroïdes est la sélection aléatoire uniforme. Au moyen de simulations exhaustives, on montre que cette dernière ne parvient pas à assurer la diversité et une bonne couverture de toutes les facettes des données, contrairement au DPP. Ces deux propriétés sont cruciales pour garantir la performance du DPP avec de petits ensembles. Les simulations de données artificielles et l'application à des données réelles montrent que le partitionnement par consensus déterminantal surpasse les algorithmes classiques comme le partitionnement par consensus basé sur les k-médoïdes ou les k-moyennes. Ces deux algorithmes reposent sur une sélection aléatoire uniforme de centroïdes.
Mots clés : Classification ; diagramme de Voronoï ; fonction de base radiale ; indice de validation basé sur un noyau ; noyau de Mercer ; partitionnement autour de médoïdes ; répulsion.

Abstract. Random restart of a given algorithm produces many partitions to yield a consensus clustering. Ensemble methods such as consensus clustering have been recognized as more robust approaches for data clustering than single clustering algorithms. We propose the use of determinantal point processes or DPP for the random restart of clustering algorithms based on initial sets of center points, such as k-medoids or k-means. The relation between DPP and kernel-based methods makes DPPs suitable to describe and quantify similarity between objects. DPPs favor diversity of the center points within subsets. So, subsets with more similar points have less chances of being generated than subsets with very distinct points. The current and most popular sampling technique is sampling center points uniformly at random. We show through extensive simulations that, contrary to DPP, this technique fails both to ensure diversity, and to obtain a good coverage of all data facets. These two properties of DPP are key to make DPPs achieve good performance with small ensembles. Simulations with artificial datasets and applications to real datasets show that determinantal consensus clustering outperforms classical algorithms such as k-medoids and k-means consensus clusterings which are based on uniform random sampling of center points.
Keywords: Classification; kernel-based validation index; Mercer kernel; partitioning about medoids; radial basis function; repulsion; Voronoi diagram.

2.1. Introduction

A classical core procedure in fields such as biology, psychology, medicine, marketing, computer vision and remote sensing is to group elements based on similar features (cluster analysis) to provide a framework for learning, as reported by Jain and Dubes (1988). Some clustering techniques, such as the standard k-means algorithm or the partitioning around medoids algorithm, are characterized by an initial choice of a subset of random points. We find the same type of initial choice in some classification techniques, such as neural networks or machine learning. Selecting a subset of points simply at random does not take into account the diversity among the selected points, because this sort of sampling mechanism gives to every point an equal probability of being selected. Many similar points may be chosen simultaneously, conveying much redundancy and little representability of the data. In some domains of research, the diversity of the selected points is a major concern. Ensuring diversity, so as to obtain a good coverage of all data facets, would certainly fail when doing simple random sampling. In contrast, determinantal point processes, or DPPs for short, induce negative correlations between similar points (Borodin and Olshanski, 2000). Kulesza and Taskar (2012) emphasize that the strength of those negative correlations is to assign higher probability to sets of points that are more diverse. Consequently, similar points have less chance of appearing together. This property has established DPPs use in machine learning as models for subset selection (Hafiz Affandi et al., 2013).

The origins of DPP date back to Macchi (1975) in quantum physics, where it is known as the fermion process. The name *Determinantal Point Process* was established in Mathematics by Borodin and Olshanski (2000). It also arises in studies of non-intersecting random paths (Daley and Vere Jones, 2003), random spanning trees (Borodin and Soshnikov, 2003), and eigenvalues of random matrices (Ben Hough et al., 2006). Currently, Kulesza and Taskar (2012) and Lavancier et al. (2015) represent central references concerning DPP.

Traditionally, algorithms and methods of cluster analysis are gathered in two families of techniques: hierarchical techniques and partitioning or non-hierarchical techniques. Hierarchical techniques include agglomerative clustering with single linkage, (Florek et al., 1951), and Hidden Markov Models agglomerative clustering, (Smyth, 1997). Partitioning techniques include the partitioning around medoids (PAM) algorithm, (Kaufmann and Rousseeuw, 1987), and the k-means algorithm, (Lloyd, 1982). Recently, and due to the contribution of advanced computational methods, additional families of clustering techniques may be considered (Han et al., 2011). Among others, one finds probabilistic model-based techniques, which assume that each observed cluster represents a sample drawn from a specific probability distribution and, consequently, that the overall distribution of the data consists in a mixture of several distributions (Banfield and Raftery, 1993; Fraley and Raftery, 1998); density-based techniques, which model clusters as dense regions of objects in space (with

respect to a local density measure) separated by sparse or low-density regions (Ester et al., 1996; Ankerst et al., 1999; Stuetzle and Nugent, 2010; Stuetzle, 2003); and grid-based techniques, which split the space into a finite number of cells that establishes a grid structure, where all the clustering process is performed (Wang et al., 1997; Hinneburg and Keim, 1999).

The majority of the clustering methods seeks to obtain a single and individual optimal partition of the data, according to some internal clustering criterion, based on the principle of maximizing both within-cluster similarity and between-cluster dissimilarity. However, as stressed by Vega-Pons and Ruiz-Shulcloper (2011), if different clustering techniques are applied to the same data, they can produce very different clustering results, due in part to a lack of an external objective and impartial criterion. The techniques' dependency on the initial choice of points can also explain those differences. In order to improve the quality and robustness of clustering results, Blatt et al. (1996) and Blatt et al. (1997) introduced a cluster-membership probabilistic framework for clustering based on physical properties of ferromagnetic models. Later, Strehl and Ghosh (2002) formalized this approach, defining the *cluster ensembles* framework whose main objective is to combine different clustering results into a single consolidated clustering. Vega-Pons and Ruiz-Shulcloper (2011) established four desirable properties that should be present in the results of any cluster ensemble method. These are (i) robustness, so that the single consolidated clustering must have better average performance than single and individual clustering algorithms; (ii) consistency, in the sense that the single consolidated clustering should produce similar results to those of all combined individual clustering algorithms; (iii) novelty, that is, any cluster ensemble method should produce clustering solutions usually not attainable by single clustering algorithms; and (iv) stability, in the sense that the results of the single consolidated clustering should have lower sensitivity to noise, outliers and initial conditions. One of the most well-known cluster ensembles methods was introduced by Monti et al. (2003), in genomic studies and gene expression data, inspired by resampling and cross-validation techniques such as bootstrapping.

Consensus clustering is defined as a method meant for attaining a single consolidated clustering from multiple runs of the same clustering algorithm. The obtained single consolidated clustering, built over some agreement among the several runs, represents a partition of data. Although not required, multiple runs of the algorithm could be initialized with a random restart.

In this paper, we focus on partitioning techniques with random initial conditions. We explore the determinantal point process presented by Ben Hough et al. (2006) and Kulesza and Taskar (2012) for sampling the initial cluster centers. The use of the determinantal point process implies the choice of a real, symmetric and positive semidefinite matrix that measures similarity between all elements. The properties of this type of matrices open a connection with the well-known kernel-based methods, which have been widely used in pattern analysis, classification and clustering (Howley and Madden, 2006). One of the most popular kernels,

and the one we use in this paper, is the Radial Basis Function, also known as the Gaussian kernel.

The paper is organized as follows: in Section 2.2, we introduce some basic notation, and describe the basic ideas related to consensus clustering. In Section 2.3, we present the basic properties of determinantal point processes, and put them into context as a sampling method to generate cluster centers in a partitioning clustering algorithm. Since the choice of a proper kernel is central in the construction of determinantal point processes, kernel-based methods are also introduced in this section. In Section 2.4, we introduce our proposed methodology for determinantal consensus clustering, or *consensus DPP,* for short. In Section 2.5, we perform an extensive simulation in order to evaluate the performance of consensus DPP. The performance of the proposed algorithm on real datasets is presented in Section 2.6. A comparison with other partitioning methods is also shown in these last two sections. We conclude with a few thoughts and a discussion in Section 2.7.

2.2. Consensus clustering

Throughout the paper the data will be denoted by $\mathcal{S} = \{x_1, \ldots, x_n\} \subset \mathbb{R}^p$, where x_i represents a p-dimensional vector, for $i = 1, \ldots, n$ and $n \geq 2$. Cluster analysis consists in a range of algorithms and methods that divide a discrete set of elements into several subsets, or clusters, sharing some common features or properties. The process of division into subsets follows two criteria: (a) Division is exclusive, i.e., subsets do not overlap, forming a partition $\{C_1, C_2, \ldots, C_K\}$ of $\mathcal{S}$. That is $\mathcal{S} = \bigcup_{i=1}^{K} C_i$, and $C_i \cap C_j = \emptyset$ whenever $i \neq j$, $i,j = 1, \ldots, K$. (b) Division is intrinsic or unsupervised, i.e., the division is based only on a proximity matrix, rather than using category labels denoting an a priori partition.

Consider a particular partitioning clustering technique run R times on the data $\mathcal{S}$. The agreement among the several runs of the algorithm is based on the *consensus matrix* C. This is a $n \times n$ symmetric matrix whose entries $\{C_{ij},\ i,j = 1, \ldots, n\}$ represent the proportion of runs in which elements x_i and x_j of $\mathcal{S}$ fall in the same cluster. Let r represent a specific run of the clustering algorithm, $r = 1, \ldots, R$, and let C_r be the associated $n \times n$ symmetric binary matrix with entries

$$c_{ij}^r = \begin{cases} 1, & \text{if } x_i \text{ and } x_j \text{ belong to the same cluster;} \\ 0, & \text{otherwise,} \end{cases} \tag{10}$$

for $i,j = 1, \ldots, n$. The consensus matrix C is then the $n \times n$ symmetric matrix with entries defined by

$$C_{ij} = \sum_{r=1}^{R} c_{ij}^r / R, \tag{11}$$

for $i,j = 1, \ldots, n$. The entry C_{ij} is known as *consensus index*. Obviously, the diagonal entries are given by $C_{ii} = 1$, for $i = 1, \ldots, n$.

Clusters can also be defined using graph theory by considering the graph (V,E) whose vertices are given by the elements $x_i \in \mathcal{S}$. The set of edges E is defined by connecting each pair of elements sharing some common features or properties. A clustering configuration with K clusters consists in an undirected graph with K connected components. Murua et al. (2008) established that the single consolidated clustering with K final clusters obtained by consensus clustering represent the K connected components of the *consensus graph*. That is the graph over the observations with an edge between any pair of elements that belong to the same cluster in the majority of the configurations. The majority concept is tied to the consensus index defined in (11).

2.3. The determinantal point process

Keeping the notation from the previous section, a DPP is a probability measure on $2^{\mathcal{S}}$ that assigns probability

$$P(Y) = \det(L_Y)/\det(L + I_n), \tag{12}$$

to any subset $Y \in 2^{\mathcal{S}}$, where L is a $n \times n$ real, symmetric and positive semidefinite matrix that measures similarity between all pairs of elements of $\mathcal{S}$; L_Y is the principal submatrix of L whose rows and columns are indexed by Y, i.e., $L_Y = (L_{ij})_{i,j\in Y}$; and I_n is the $n \times n$ identity matrix. If $\boldsymbol{Y}$ is the random variable that represents the subset selected from $2^{\mathcal{S}}$, then we write $\boldsymbol{Y} \sim DPP_{\mathcal{S}}(L)$ for the corresponding determinantal process. The matrix L is known as the kernel matrix of the DPP (Kulesza and Taskar, 2012; Kang, 2013; Hafiz Affandi et al., 2014). It can be shown (Kulesza and Taskar, 2012) that $\det(L+I_n) = \sum_{Y\subset 2^{\mathcal{S}}} \det(L_Y)$, hence (12) does indeed define a probability mass function over all subsets in $2^{\mathcal{S}}$. This definition states restrictions on all the principal minors of the kernel matrix L, denoted by $\det(L_Y)$. Indeed, as $P(\boldsymbol{Y} = Y) \propto \det(L_Y)$ represents a probability measure, we have $\det(L_Y) \geq 0$, for any $Y \subseteq \mathcal{S}$. This implies that any symmetric positive semidefinite matrix can be taken as kernel matrix L, where its eigenvalues are such that $\lambda_i(L) \geq 0$, for every $i = 1, \ldots, n$.

2.3.1. Relation between kernel-based methods and DPP

Determinants have a well-known geometric interpretation. Because L is positive semidefinite, it can be decomposed as $L = B^T B$, where B is a $m \times n$ matrix. Denoting the columns of B by B_i, for $i = 1, \ldots, n$, we have

$$P(\boldsymbol{Y} = Y) \propto \det(L_Y) = \mathrm{Vol}^2\left(\{B_i\}_{i\in Y}\right), \tag{13}$$

where Vol^2 represents the squared volume of the parallelepiped spanned by the columns of B corresponding to elements in Y. The columns of B can be interpreted as feature vectors describing the elements of $\mathcal{S}$ and, therefore, L measures similarity using dot products between feature vectors. As the dot product is the most natural similarity measure between vectors,

it establishes a connection with the well-known kernel-based methods. Kernels are widely used to describe and quantify how any two objects are related. They have been widely used in pattern analysis, like classification and clustering (Howley and Madden, 2006). By (13), we can see that the probability assigned by a DPP to a subset Y is related to the volume spanned by its associated feature vectors: sets composed of very diverse elements have higher probabilities, because their feature vectors are more orthogonal, and span larger volumes.

Considering clustering analysis, Jain and Dubes (1988) established that, for datasets with ellipsoidal clustered structures, "sum-of-squares" based methods have proved to be effective. However, if the frontiers that separate clusters are non-quadratic, these methods will fail to generate an effective clustering configuration. One of the several approaches to deal with this problem consists in nonlinearly transforming the data into a high-dimensional feature space, so that clustering analysis can be conducted in this feature space, constructing an optimal separating hyperplane (Vapnik, 1995; Girolami, 2002; Murua et al., 2008). Let $\mathcal{H}$ be an embedding Hilbert space, and consider a mapping $\mathbf{\Phi} : \mathcal{S} \to \mathcal{H}$. The set containing all the transformed elements of $\mathcal{S}$ is represented by

$$\mathbf{\Phi}(\mathcal{S}) := \{\mathbf{\Phi}(x_1), \ldots, \mathbf{\Phi}(x_n)\}.$$

Each x_i is mapped into a high-dimensional Hilbert space $\mathcal{H}$ with coordinates $\mathbf{\Phi}(x_i) = (\phi_1(x_i),\phi_2(x_i),\ldots)$, for $i = 1,\ldots,n$. Because the feature space $\mathcal{H}$ may be of high and possibly infinite dimension, working directly with the transformed data is an unrealistic option. Kernel-based methods calculate a similarity measure between each pair of elements on the feature space $\mathcal{H}$, to afterwards use algorithms that only need the value of this measure (Schölkopf et al., 2004). Since the feature space is a Hilbert space, the inner product $\langle\mathbf{\Phi}(x_i),\mathbf{\Phi}(x_j)\rangle_{\mathcal{H}}$ is the obvious and simplest similarity measure to conceive. The algorithms of kernel-based methods are said to employ kernel functions, since the pairwise inner products $\langle\mathbf{\Phi}(x_i),\mathbf{\Phi}(x_j)\rangle_{\mathcal{H}}$ can be computed directly from the original data using an appropriate kernel function $\kappa : \mathcal{S} \times \mathcal{S} \to \mathbb{R}$. That is,

$$\kappa(x_i,x_j) = \langle\mathbf{\Phi}(x_i),\mathbf{\Phi}(x_j)\rangle_{\mathcal{H}}, \tag{14}$$

for $i,j = 1,\ldots,n$. The kernel function is then able to represent the inner products in $\mathcal{H}$ in the original space $\mathcal{S}$. Consequently, kernel-based methods replace the inner products with the kernel function, a fact known as the *kernel trick*. However, (14) raises the issue of which type of kernel functions are allowed.

Mercer's Theorem (Vapnik, 1995) tells us whether or not a function κ is actually an inner product $\langle\mathbf{\Phi}(x_i),\mathbf{\Phi}(x_j)\rangle$ in some space $\mathcal{H}$. Assume that all possible data $\mathcal{S}$ live in a compact subset $\mathcal{X}$ of $\mathbb{R}^p$. Suppose $\kappa : \mathcal{S} \times \mathcal{S} \to \mathbb{R}$ is a continuous symmetric function satisfying

$$\sum_{i=1}^{n}\sum_{j=1}^{n} a_i a_j \kappa(x_i,x_j) \geq 0,$$

for any finite set of points $\{x_i\}_{i=1}^n$ in $\mathcal{X}$ and real numbers $\{a_i\}_{i=1}^n$. Furthermore, suppose that

$$\int_{\mathcal{X}\times\mathcal{X}} \kappa(x_i, x_j) f(x_i) f(x_j) dx_i dx_j \geq 0,$$

for all squared-integrable functions $f(\cdot)$ on $\mathcal{X}$. Then, κ can be expanded in a uniformly convergent series in terms of a unique enumerable set of non-negative eigenvalues $\{\lambda_r\}_{r=1}^{+\infty}$, and associated squared-integrable orthogonal eigenfunctions $\{\psi_r\}_{r=1}^{+\infty}$,

$$\kappa(x_i, x_j) = \sum_{r=1}^{+\infty} \lambda_r \psi_r(x_i) \psi_r(x_j).$$

Recalling (14), if a function κ satisfies Mercer's Theorem, we can define a feature map $\boldsymbol{\Phi} : \mathcal{S} \to \mathcal{H}$ as follows:

$$\boldsymbol{\Phi}(x_i) = \left(\sqrt{\lambda_1}\psi_1(x_i), \sqrt{\lambda_2}\psi_2(x_i), \dots \right),$$

for $i = 1, \dots, n$. In this case, we say that the kernel function κ is a *Mercer kernel*. A particular property of Mercer kernels is that they also are positive semidefinite kernels. This ensures that the $n \times n$ matrix defined by

$$\boldsymbol{K} = (\kappa(x_i, x_j))_{i,j=1}^n = (\langle \boldsymbol{\Phi}(x_i), \boldsymbol{\Phi}(x_j) \rangle_{\mathcal{H}})_{i,j=1}^n ,$$

is positive semidefinite. $\boldsymbol{K}$ is also a Gram matrix (Horn and Johnson, 2012).

Kernel functions are often considered measures of similarity, since a higher kernel value represents a higher correlation in the associated Hilbert space. Therefore, for the purpose of cluster analysis and the choice of the similarity matrix L for the DPP established in (12), Mercer kernels are perfect candidates, so that

$$L = [\kappa(x_i, x_j)]_{i,j=1}^n . \tag{15}$$

2.3.2. Choice of kernel: the radial basis function kernel

The choice of the most appropriate kernel function is a critical step in the application of any kernel-based method. However, as pointed by Howley and Madden (2006), there is no rule or consensus about the choice of the most suitable kernel function for a particular problem. Ideally, the suitable kernel function is chosen according to prior knowledge of the problem domain (Howley and Madden, 2006; Lanckriet et al., 2004), which is rarely observable in practice. In the absence of expert knowledge, a common choice is the *Radial Basis Function* (RBF) kernel (or *Gaussian kernel*):

$$\kappa(x_i, x_j) = \exp\left(-\|x_i - x_j\|^2 / (2\sigma^2) \right), \tag{16}$$

where the scale parameter σ, known as the bandwidth of the kernel, represents the relative spread of the distances $\|x_i - x_j\|$. Here, the distance $\|x_i - x_j\|$ represents the Euclidean distance between x_i and x_j, a common choice for the RBF, which we will also follow. The

RBF is a Mercer kernel, as presented in Section 2.3.1, and details concerning its expansion in terms of non-negative eigenvalues $\{\lambda_r\}_{r=1}^{+\infty}$ and associated eigenfunctions $\{\psi_r\}_{r=1}^{+\infty}$ can be found in Fasshauer (2011). This particular kernel has been extensively used in many studies, due to its appealing mathematical properties, as mentioned by Girolami (2002). A particular property of the Gaussian kernel is that it is positive and bounded from above by one, making it directly interpretable as a scaled measure of similarity between x_i and x_j.

For the purpose of this paper, we choose the Gaussian kernel defined in (16) for building the similarity matrix L of the DPP with (15). The computation of the RBF kernel requires the estimation of the bandwidth parameter σ. As pointed by Murua and Wicker (2014), most of the literature considers σ as a parameter that can be estimated by observed data. Inspired by Blatt et al. (1996) and Blatt et al. (1997), we estimate σ^2 by the average of all pairwise squared Euclidean distances, i.e.,

$$\widehat{\sigma}^2 = 2\sum_{i<j} \|(x_i - x_j)\|^2 / \Big(n(n-1)\Big). \tag{17}$$

The authors justify the use of the average to estimate σ^2 based on local structure of the data and identification of high-density regions in the data space. Other methods of estimating the bandwidth parameter can be found in the literature. Murua et al. (2008) do not consider σ as fixed and propose an adaptive bandwidth selection procedure, where σ depends on the data points. They explore the relationship between Potts model and kernel density estimation, building an algorithm based on Markov Chain Monte Carlo methods to obtain a Bayesian estimate of σ. Murua and Wicker (2014) consider σ as fixed and obtain its Bayesian estimate based on the Wang-Landau algorithm (Wang and Landau, 2001). Sejdinovic et al. (2013) refer the median of the pairwise Euclidean distances as a common choice. However, Chaudhuri et al. (2017) demonstrate that the use of the average and the median of Euclidean distances to estimate σ produce similar clustering results for the majority of situations. They justify the use of the average distances by its simplicity and fast computation even when the dataset is large. We decided to use the average for the same reasons.
To explore the sensitivity of the clustering configuration to the parameter σ, we decided to introduce a tuning parameter $s > 0$, which will be estimated heuristically by simulation in Section 2.4.2. With the bandwidth estimate given by (17) and the tuning parameter s, the RBF kernel in (16) will be adjusted to

$$\kappa(x_i,x_j) = \exp\left(-\|(x_i - x_j)\|^2/(2s\widehat{\sigma}^2)\right). \tag{18}$$

2.4. Consensus DPP

In this section we develop a partitioning clustering algorithm that will be run R times over the set $\mathcal{S}$, in order to obtain a consolidated clustering configuration by consensus clustering. To build a consensus clustering, any partitioning clustering method can be chosen. We

propose the use of determinantal point processes as the partition generating algorithm. The algorithm is also based on a Voronoi diagram as described next.

Voronoi diagrams support many clustering techniques (Aurenhammer, 1991), such as the $k-$means and k-medoids algorithms, for example. A Voronoi diagram refers to a partition of the space into several cells or regions, based on a subset of elements that are called *generator points*, or simply *generators*. Each cell includes only one generator and all the space points that are closer to that generator than to any other generator. For a formal definition of Voronoi diagram, see for example (Okabe et al., 2000).

Let $\mathcal{P} = \{p_1, \ldots, p_K\} \subseteq \mathcal{S}$ be the generator set of the Voronoi diagram. The p-dimensional Voronoi polyhedron associated with p_i, $i = 1, \ldots, K$ is the region defined by

$$V(p_i) = \{x \in \mathcal{S} : \|x - p_i\| < \|x - p_j\|, \text{ for all } j \neq i, j = 1, \ldots, K\}.$$

The set $\mathcal{V}(\mathcal{P}) = \{V(p_1), \ldots, V(p_K)\}$ is said to be the p-dimensional Voronoi diagram generated by $\mathcal{P}$. We call p_i the generator point or generator of the ith Voronoi polyhedron. The Voronoi diagram is a partition of the data $\mathcal{S}$, and hence a clustering of the data. In order to obtain a Voronoi diagram, one needs to select the set of generators. We proposed using a determinantal point process (DPP) rather than a classical random sampling for this step. A DPP intents to capture and model negative correlations between the elements of $\mathcal{S}$ (Hafiz Affandi et al., 2013), so that the inclusion of one element makes the inclusion of other similar elements less likely. We conjecture that sampling from a DPP for Voronoi generators is more efficient than sampling generator points uniformly at random as it is usually done in PAM. Our experiments in Section 2.4.2 corroborate this belief.

Ben Hough et al. (2006) and Kulesza and Taskar (2012) present an efficient scheme to sample from a DPP. The algorithm is based on the following observations. Let $L = \sum_{i=1}^{n} \lambda_i(L) v_i v_i^T$ be an orthonormal eigendecomposition of L. For any set of indexes $J \subseteq \{1,2,\ldots,n\}$, define the subset of eigenvectors $V_J = \{v_i : i \in J\}$, and the associated matrix $\mathcal{K}_J = \sum_{i \in J} v_i v_i^T$. It can be shown that the matrix $\mathcal{K}_J$ defines a so-called *elementary* DPP which we denote by $\text{DPP}(\mathcal{K}_J)$. It turns out that the $\text{DPP}_{\mathcal{S}}(L)$ is a mixture of all elementary DPP given by the index sets J. That is

$$\text{DPP}_{\mathcal{S}}(L) = \left[\sum_{J} \text{DPP}(\mathcal{K}_J) \prod_{i \in J} \lambda_i(L)\right] / \det(L + I_n)$$

The mixture weight of $\text{DPP}(\mathcal{K}_J)$ is given by the product of the eigenvalues $\lambda_i(L)$ corresponding to the eigenvectors $\boldsymbol{v}_j \in V_J$, normalized by $\det(L + I_n) = \prod_{i=1}^{n} [\lambda_i(L) + 1]$. Sampling can be realized by first selecting an elementary DPP, $\text{DPP}(\mathcal{K}_J)$, with probability equal to its mixture component weight, and then, in a second step, sampling $\boldsymbol{Y} \sim \text{DPP}(\mathcal{K}_J)$. In particular, it can be shown that in this case, necessarily $\text{card}(\boldsymbol{Y}) = \text{rank}(\mathcal{K}_J)$.

For the purpose of consensus clustering and the construction of the consensus matrix with entries defined by (11), we will consider R runs of the sampling algorithm of Ben Hough et al. (2006) and Kulesza and Taskar (2012). The sampling of the R sets is done from the DPP with associated kernel matrix L constructed with the RBF kernel in (18). This yields R generator sets $\{\mathcal{P}_r\}_{r=1}^{R}$. For each generator set, we construct a p-dimensional Voronoi diagram $\mathcal{V}(\mathcal{P}_r)$ based on the similarities given by L. The binary matrix C_r with entries given by (10) has an entry $c_{ij}^r = 1$ if and only if the points x_i and x_j fall in the same Voronoi cell, $r = 1, \ldots, R$. The consensus matrix C, constructed with the consensus indexes defined by (11), is finally given by the average of all the matrices C_r over the R runs, $r = 1, \ldots, R$.

The consensus matrix represents the proportion of runs in which two elements x_i and x_j of $\mathcal{S}$ belong to the same cluster. The consolidated clustering configuration is obtained by a thresholding procedure. According to Blatt et al. (1996), if $C_{ij} \geq \theta$, with $0 \leq \theta \leq 1$, points x_i and x_j are defined as "friends" and then included in the same final cluster. Moreover, all mutual friends (including friends of friends, etc.) are assigned to the same cluster. It can be shown that this is equivalent to finding the connected components of the consensus graph introduced in Section 2.2 (Murua et al., 2008).

The choice of the threshold is not an easy task. Although a value of $\theta = 0.5$ makes sense most of the time, it might not be the optimal choice. In fact, Murua and Wicker (2014) shows that choosing a fixed and unique threshold does not necessarily give the best clustering results. Changing the threshold yields different clustering results. Many of those clusterings are worth exploring. Murua and Wicker (2014) consider all threshold values from the set of all different observed consensus indexes C_{ij} (see in (11)). If there are t different consensus indexes, we will have a collection of t thresholds $\theta_1, \theta_2, \ldots, \theta_t$. For each threshold θ_i, $i = 1, \ldots, t$, a consolidated clustering configuration with $K(\theta_i)$ clusters is obtained. If $\theta_i = 0$, we obtain a graph with $K(0) = 1$ cluster, that is, $\mathcal{S}$. If $\theta_i = 1$, we obtain a graph with $K(1) = n$ clusters; that is, each element of $\mathcal{S}$ is an isolated point and form a singleton cluster of size one. In general, clustering configurations with one cluster or n clusters are of no interest. Therefore, thresholds θ_i that are too low or too large are not relevant. We adopt a mixed strategy between choosing a predetermined fixed threshold (Blatt et al., 1996) and studying a sequence of interesting thresholds (Murua and Wicker, 2014). We consider a sequence of t predetermined thresholds $\theta_1, \theta_2, \ldots, \theta_t$ that are above a certain minimum threshold τ. The value of $\tau = 0.6$ has been determined through simulations. These are reported in Section 2.4.2.

Moreover, we are not interested in a clustering configuration with too many small clusters. We impose a minimal size for each cluster, accepting only clustering configurations with cluster sizes larger than that minimal value. For the establishment of the minimal size, we decided to take a classical approach, inspired by the "square-root choice" for the number of bins of a histogram, $\sqrt{n}$. However, we also consider a more general case that eliminates all

clusters with less than n^a elements for a predetermined power $a \in (0,1)$. The optimal value of the power a depends on various considerations such as the data size, the data dimension, and the number of clusters. We have studied it through the simulations reported in Section 2.4.2. In summary, we examine all the t consolidated clustering configurations obtained with all the different considered thresholds $\theta_1, \theta_2, \ldots, \theta_t$. If one configuration does not satisfy the minimal cluster size criterion, we merge each small cluster with its closest "large" cluster, according to the following procedure, inspired by single linkage: select the component $\mathcal{V}$ that has the smallest cluster size $< n^a$; find the pair of indexes $(i^*, j^*) \in \{1, \ldots, n\}$ that satisfies $C_{i^*j^*} = \max\{C_{ij} : x_i \in \mathcal{V}, x_j \notin \mathcal{V}\}$; merge the component $\mathcal{V}$ to the component that includes $\boldsymbol{x_{j^*}}$; repeat the merging procedure until there are no more connected components with cluster size smaller than n^a. Other linkage merging criteria are possible, such as average linkage or minimax linkage (Ao et al., 2005; Bien and Tibshirani, 2011). However, in our experiments these two merging linkage criteria perform similarly to single linkage merging.

As mentioned above, the choice of the power a, the minimum threshold τ, and the number R of runs of our clustering algorithm, have been determined via simulation (see Section 2.4.2).

2.4.1. Choosing an optimal clustering

Following our mixed strategy for the thresholding procedure, we end up with a set of consolidated clustering configurations that meet our minimal cluster size criterion. One question remains: which configuration to keep as final clustering configuration. The answer depends on the criterion chosen to measure the adequacy of the clustering configuration. We use kernel-based measures that depend only on the kernel matrix L in order to be computed.

Consider the RBF kernel $\kappa(x_i,x_j)$ defined by (16). Compute the mean of the transformed data $\boldsymbol{\Phi}(\mathcal{S})$, $\bar{\boldsymbol{\Phi}} = \sum_{i=1}^n \boldsymbol{\Phi}(x_i)/n$. The *mean scattering* induced by the kernel on the data (Vert et al., 2004; Fan et al., 2010) is defined as

$$V_{\mathcal{S}} \quad = \quad \sum_{i=1}^{n} \frac{\|\boldsymbol{\Phi}(x_i) - \bar{\boldsymbol{\Phi}}\|}{n} \quad = \quad \frac{1}{n}\sum_{i=1}^{n} \left\{ \kappa(x_i,x_i) - \sum_{j=1}^{n} \frac{2\,\kappa(x_i,x_j)}{n} + \sum_{j,\ell=1}^{n} \frac{\kappa(x_j,x_\ell)}{n^2} \right\}^{\frac{1}{2}}.$$

Let $\mathcal{V} = \{\mathcal{V}_1, \ldots, \mathcal{V}_K\}$ be a cluster configuration with K clusters. Also, let n_k be the size of cluster $\mathcal{V}_k$, and consider its center in the transformed space $\bar{\boldsymbol{\Phi}}_k = \sum_{i \in J_k} \boldsymbol{\Phi}(x_i)/n_k$, where $J_k = \{i : x_i \in \mathcal{V}_k\}$, $k = 1, \ldots, K$. As with the mean scattering, Fan et al. (2010) define similarly, the kernel induced scattering within each cluster $\mathcal{V}_k$, $W_{\mathcal{V}_k} = \sum_{i \in J_k} \|\boldsymbol{\Phi}(x_i) - \bar{\boldsymbol{\Phi}}_k\|/n_k$. That is,

$$W_{\mathcal{V}_k} = \frac{1}{n_k} \sum_{i \in J_k} \left\{ \kappa(x_i,x_i) - \sum_{j \in J_k} \frac{2\,\kappa(x_i,x_j)}{n_k} + \sum_{j,\ell \in J_k} \frac{\kappa(x_j,x_\ell)}{n_k^2} \right\}^{\frac{1}{2}}.$$

The average within cluster scattering associated with clustering configuration $\mathcal{V}$ is defined as

$$W_{\mathcal{V}} = \sum_{k=1}^{K} W_{\mathcal{V}_k}/(K\,V_{\mathcal{S}}).$$

To measure the total scattering between clusters, one considers the distance between cluster means in the transformed space, $B(\mathcal{V}_i, \mathcal{V}_j) = \|\bar{\mathbf{\Phi}}_i - \bar{\mathbf{\Phi}}_j\|$, that is,

$$B^2(\mathcal{V}_i,\mathcal{V}_j) = \Big[B(\mathcal{V}_i,\mathcal{V}_j)\Big]^2 = \sum_{i,j\in J_i} \frac{\kappa(x_i,x_j)}{n_i^2} - \sum_{i\in J_i}\sum_{j\in J_j} \frac{\kappa(x_i,x_j)}{n_i n_j} + \sum_{i,j\in J_j} \frac{\kappa(x_i,x_j)}{n_j^2},$$

and $B_{\mathcal{V}} = \sum_{i=2}^{K}\sum_{j=1}^{i-1} n_i n_j B(\mathcal{V}_i,\mathcal{V}_j)/\sum_{i=2}^{K}\sum_{j=1}^{i-1} n_i n_j$. A simple measure of quality of the clustering configuration may be easily derived from the similarity ratio introduced by Chen et al. (2002),

$$SR_{\mathcal{V}} = 1 - \Big[n/(n-1)\Big]\Big[W_{\mathcal{V}}/(W_{\mathcal{V}} + B_{\mathcal{V}})\Big]. \tag{19}$$

Because a good clustering configuration must have a small within variance and a large between variance, the larger $SR_{\mathcal{V}}$, the better the clustering $\mathcal{V}$.

An alternative measure, *the kernel-based validation index,* developed by Fan et al. (2010) is based on

$$\tilde{B}_{\mathcal{V}} = (B_{\max}/B_{\min}) \sum_{i=1}^{K}\sum_{j=1}^{K}\Big[B^2(\mathcal{V}_i,\mathcal{V}_j)\Big]^{-1},$$

where $B_{\max} = \max_{(i,j)} B^2(\mathcal{V}_i,\mathcal{V}_j)$, and $B_{\min} = \min_{(i,j)} B^2(\mathcal{V}_i,\mathcal{V}_j)$. The measure is given by

$$KVI_{\mathcal{V}} = \alpha W_{\mathcal{V}} + \tilde{B}_{\mathcal{V}}, \tag{20}$$

where α is a tuning parameter that the authors set to the value of $\tilde{B}_{\mathcal{V}}$ associated with the largest clustering size K among those clustering configurations being considered. The optimal clustering configuration among the set of all retained clustering configurations is the one that minimizes (20).

2.4.2. Setting appropriate consensus DPP parameters

In this section, we will conduct simulations to choose the tuning parameters of the proposed clustering method, these are (i) the tuning parameter a for the minimal size of clusters n^a; (ii) the number R of sufficient runs of the clustering algorithm; (iii) the inferior limit τ for the threshold range to obtain the consolidated clustering configurations; and (iv) the tuning parameter s in (18) for the sensitivity of the bandwidth parameter $\hat{\sigma}^2$.

Data generation. The simulated data were generated with the algorithm of Melnykov et al. (2012). This generates datasets from p-variate Gaussian mixtures with K components, $\sum_{k=1}^{K} \pi_k \phi_p(\cdot;\mu_k, V_k)$, where ϕ_p denotes the p-variate normal density. The mean vectors of the components, $\{\mu_1,\ldots,\mu_K\}$, are obtained as k independent realizations from a uniform p-variate unit hypercube. The covariance matrix of each component, V_k, is obtained as a

realization from the p-variate standard Wishart distribution with $p+1$ degrees of freedom. The mixing proportions π_k are generated from a Dirichlet distribution on the standard $K-1$ simplex, so that $\sum_{k=1}^{K} \pi_k = 1$. The number of elements generated from each component is obtained from the multinomial distribution based on the mixing proportions.

The algorithm allows to control the *pairwise overlap* between two components, which measures the interaction between components, and controls the clustering complexity of datasets simulated from the mixtures. For the purposes of our study, we focus on low pairwise overlap cases, as the notion of clustering itself becomes less meaningful as the overlap degree between clusters becomes large. According to Melnykov et al. (2012), an average overlap of 0.4 is considered extreme while an average overlap of 0.001 is considered very low. We fix a maximum pairwise overlap of 0.01 between any two components. The values of the pairwise overlap were then generated uniformly at random taking into account this constraint. Moreover, the data were obtained from mixtures with ellipsoidal covariance matrices, and unequal number of elements per component.

We studied the effect of three variables on the clustering results. These are the number of observations per dataset $n \in \{150 \text{ (low)}, 500 \text{ (medium)}, 1500 \text{ (large)}\}$; the number of variables per dataset p: low ($2 \leq p \leq 7$), medium ($8 \leq p \leq 12$) and large ($13 \leq p \leq 20$); and the number of components or clusters per dataset G: low ($2 \leq K \leq 5$), medium ($6 \leq K \leq 10$) and large ($11 \leq K \leq 20$). The values of p and G were chosen randomly in each case once the level (low, medium or large) was chosen. The three categories generate a 3^3 factorial design with 27 experimental conditions. To simulate data for this factorial design, we ensured that no cluster with a small number of elements was present in the simulated datasets. This was done because, according to the merging procedure described in Section 2.4, small clusters would be inevitably merged with a larger cluster. This consideration results in simulated data with more or less balanced clusters. However, as it is impossible to simulate a dataset with a low number of observations ($n = 150$) and a large number of clusters ($11 \leq K \leq 20$) that contains no cluster with a small number of elements, this case has been excluded from our analysis. Therefore, we only consider 24 experimental conditions of the 3^3 factorial design, generating 10 datasets per condition, obtaining a total of 240 datasets. We will refer to the 24 experimental conditions as the 24 experimental *scenarios,* or scenarios, for short.

Measuring the quality of the clustering. In order to measure the quality of the clustering results we use the Adjusted Rand Index (ARI), which is a common measure of goodness-of-fit in the clustering literature (Yeung et al., 2001; Murua and Wicker, 2014). The ARI was first introduced by Rand (1971) and later adjusted for randomness by Hubert and Arabie (1985). It is a measure of agreement between two clustering configurations. The original Rand Index counts the proportion of elements that are either in the same clusters in both clustering configurations or in different clusters in both configurations. The adjusted

version of the Rand Index corrected the calculus of the proportion, so that its expected value is zero when the clustering configurations are random. The larger the ARI, the more similar the two configurations are, with the maximum ARI score of 1.0 indicating a perfect match.

We will also use the relative difference in the estimated number of clusters and the true number of clusters (RN) presented by Muñoz and Murua (2018):

$$\mathrm{RN} = (\sqrt{\widehat{G}} - \sqrt{G})/\sqrt{G},$$

where $\widehat{G}$ is the estimated number of clusters and G is the true number of clusters. Here, small absolute values of RN are preferred and, therefore, the absolute value of RN will be used as another measure of clustering quality.

Results. For each of the ten replicas associated with one of the 24 scenarios, we run consensus DPP with $R = 1000$, and $a \in \{1/4, 1/3, 1/2, 2/3\}$. To obtain the consolidated clustering configurations with the thresholding procedure, we set the threshold to a unique fixed value in $\{0.4, 0.5, 0.6, 0.7, 0.8, 0.9\}$. The quality of each clustering configuration associated with each threshold was assessed with the ARI criterion, after the first 10, 50, 100, 200, 300, 500, 700, 900 and 1000 runs. As each case is evaluated on the ten replica datasets of a particular scenario, the ARIs were averaged over the ten runs. For the choice of the optimal clustering, we used the similarity ratio and the kernel-based validation index, defined in (19) and (20), respectively. After comparing the results, we decided to only keep the kernel-based validation index since it provides better results.

From the results (not shown here) we observe a relative stability or the ARI mean values for most of the cases. Also, setting $a = 1/2$ for the minimal cluster size criterion of the merging procedure seems a reasonable choice, which corresponds to the classic choice of $\sqrt{n}$. With respect to the number of runs required to achieve a good clustering fit, the simulations show that a number of runs between 50 and 200 is adequate. As the best results are obtained most of the times with 200 runs, we adopt $R = 200$. The choice of the inferior limit τ for the range of thresholds depends slightly on the experimental scenario. The optimal values of τ are larger for datasets with a large number of clusters. In this scenario, values of 0.7 to 0.9 are preferred. For the other cases, the optimal value hovers around 0.5. In order to recommend a unique value, we took the average among the optimal values of τ for each scenario. This yielded $\tau = 0.6$.

To evaluate the sensitivity of the bandwidth parameter $\hat{\sigma}^2$ through the tuning parameter s in (18), we evaluate the ARI criterion as above for every value of $s \in \{0.5, 0.75, 1, 1.25, 1.5, 2\}$. Following the above observations on the optimal values for a, R and τ, for this experiment, we set $a = 0.5$, $R = 200$, and $\tau = 0.6$. The simulation results clearly show that the value of s has almost no effect on the clustering configurations, nor the corresponding ARI values. Therefore, we decided to fix $s = 1$ in (18).

2.5. Comparison performance between consensus DPP and PAM

As the choices for a, R, τ and s have been fixed, we can now assess the performance of our clustering algorithm. For the evaluation, we use the procedure to simulate data described in the previous section. As a reference, and for comparison purposes, our consensus clustering methodology will not only be applied to clustering configurations generated with DPP, but also to clustering configurations generated with the well-known Partitioning Around Medoids (PAM) method (Kaufmann and Rousseeuw, 1987). The PAM method is a classical partitioning technique for clustering that chooses the data point centers of the Voronoi cells by simple random sampling. As DPP selects data points centers based on diversity, our goal here is to study how the quality of the clustering configurations depends on diversity at sampling centroids.

To sample points at random in the PAM procedure, we first sample a number k of Voronoi cells uniformly at random from a finite set of integers $\{1, \ldots, K_{\max}\}$. Then, we sampled uniformly at random k points from the dataset. Although, the sampling is uniform over the subset sizes, and over subsets of the same size, this sampling technique is not really uniform; it favors very large and very small subsets over moderately sized subsets. In fact, the probability of choosing a subset of k points $\{x_{i_1}, \ldots, x_{i_k}\}$ following this sampling technique is $\left(K_{\max}\binom{n}{k}\right)^{-1}$. Despite this fact, we will refer to this sampling as uniform random sampling.

First, we compare the ARI trajectories of consensus DPP with those of PAM as a function on the number of runs R. Figure 2.1 displays these trajectories considering all experimental scenarios described in the previous section. The trajectories globally reflect the performance of each experimental condition involving the three levels of variables (low, medium and large) and the three levels of clusters (low, medium and large). Observe that due to the diversity in the sampling of data points center, DPP has a jump-start like behavior, requiring very few runs to achieve good clustering configurations. PAM, on the contrary, improves slowly its clustering configurations, and sometimes, even after 1000 runs, it never catches up with consensus DPP.

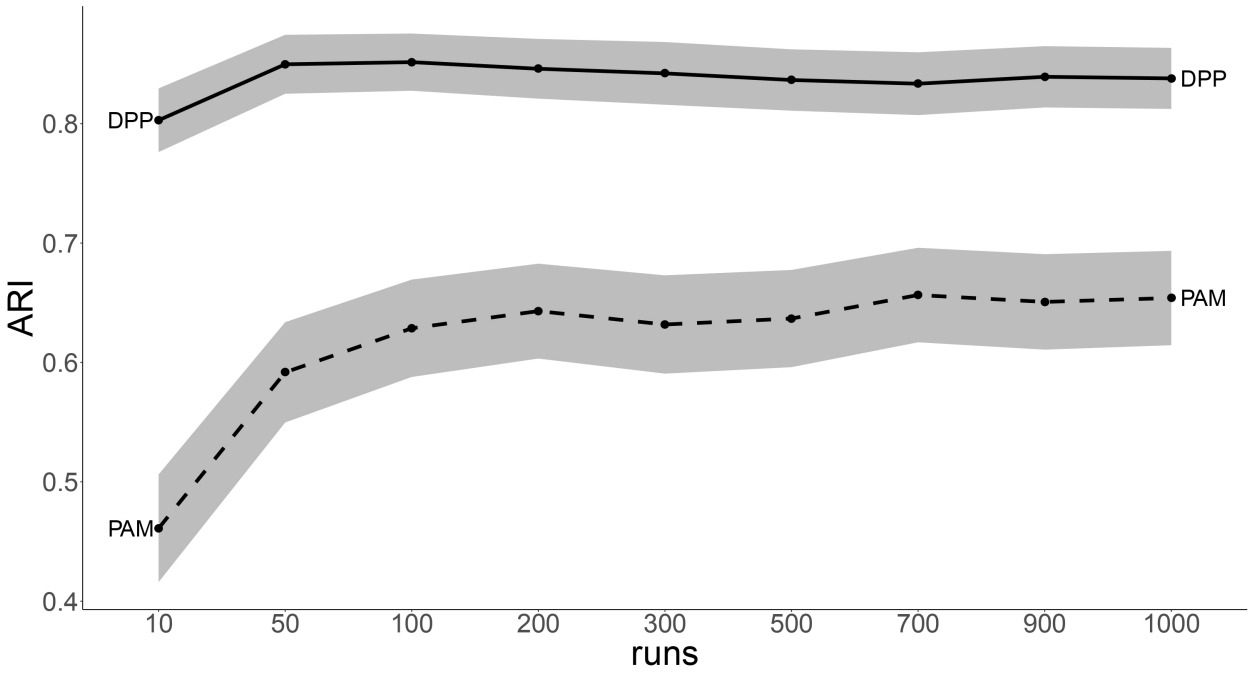
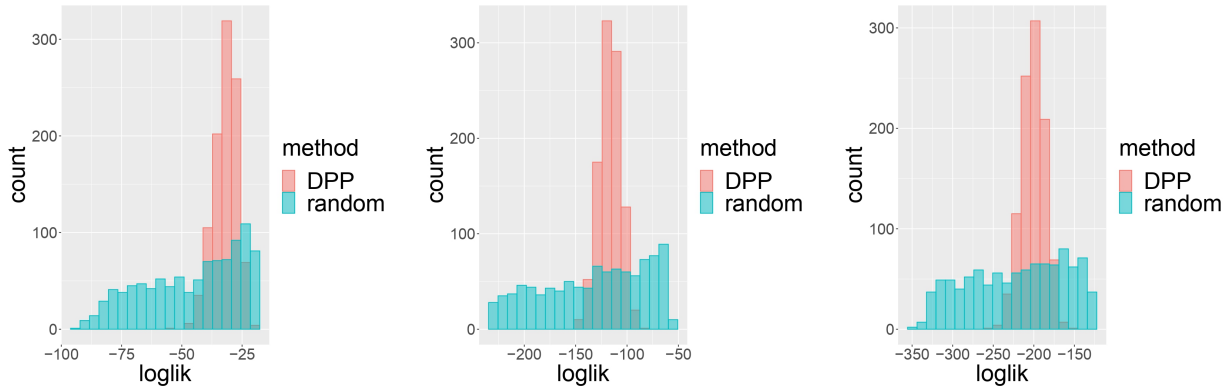


Figure 2.1 – ARI mean trajectories for DPP and PAM as a function of the number of runs R. The regions encompassing the trajectories are the envelops of the corresponding 95% confidence intervals associated with each mean ARI value of the run R.

We can see clear benefits of using DPP as a sampling method for the initial points needed to construct the Voronoi diagrams that define the clustering configurations. However, consensus DPP and PAM yield similar results when the variable dimensions are low. The dimension p seems to play a crucial role in the potential of DPPs to sample with more diversity: there are more possibilities of distinguishing two vectors x_i,$x_j \in \mathbb{R}^p$ in the transformed space $\mathcal{H}$ when the dimension p is already large, since the two vectors are more likely to be projected in very different places. In this case, DPP will sample these two points together more often than uniform random sampling, which will sample any pair of points with the same probability. When p is small, the two vectors have less potential of being very different, and using DPP or PAM should give similar results.

To complement the differences between DPP and PAM, we show in Figure 2.2 typical histograms of the logarithm of the probability mass function of the DPP, given by (12), for 1000 sampled random subsets, using either the DPP sampling algorithm of Ben Hough et al. (2006) and Kulesza and Taskar (2012), or the uniform random sampling of PAM, for three simulated datasets selected among the 24 experimental scenarios.



(a) First simulated dataset **(b)** Second simulated dataset **(c)** Third simulated dataset

Figure 2.2 – Typical histograms of the logarithm of the probability mass function (loglik), using DPP (light bars) and simple random sampling (dark bars), for three simulated datasets.

The histograms clearly show that DPP selects random subsets with higher and less dispersed probability mass values (likelihood) than uniform random sampling. This explains the observed lower dispersion of the ARI values when sampling is performed with DPP. The higher likelihood of the random subsets sampled by DPP confirms the higher diversity of those subsets. Instead, subsets sampled as in PAM can be highly or poorly diverse: in fact, the associated histograms show very high dispersion in terms of diversity. DPP tends to select points that maintain a high level of diversity at each sampling, proving to be more consistent and stable than uniform random sampling in terms of ensuring the heterogeneity of the elements forming the subset.

Next, we fix $R = 200$ as suggested by the study of the previous section, and compare the performance of consensus DPP and PAM. We have already noted the superiority of consensus DPP when looking at ARI as a measure of clustering quality. But, this time we apply our mixed strategy presented in Section 2.4. That is, we consider a sequence of thresholds $\theta_1, \theta_2, \ldots, \theta_t$ that are above the inferior limit τ. We recall that adopting the strategy of considering a range of thresholds results in a collection of consolidated clustering configurations for each datasets in each scenario. For the choice of the optimal clustering configuration, we use the kernel-based validation index KVI_ν in (20). To measure the goodness-of-fit of the optimal clustering configuration we use the ARI and RN measures described earlier. Table 2.1 displays the ARI means and standard deviations over all 24 scenarios, while Table 2.2 displays the RN means and standard deviations over all 24 scenarios.

Sample size	Method	Number of clusters	Number of variables		
			Low	Medium	Large
$n = 150$	DPP	Low	0.95 (0.07)	0.92 (0.16)	0.90 (0.15)
	PAM		0.90 (0.14)	0.93 (0.13)	0.74 (0.22)
	DPP	Medium	0.92 (0.09)	0.94 (0.07)	0.85 (0.10)
	PAM		0.98 (0.02)	0.83 (0.14)	0.65 (0.14)
$n = 500$	DPP	Low	0.95 (0.11)	0.95 (0.10)	0.92 (0.09)
	PAM		0.77 (0.22)	0.86 (0.19)	0.83 (0.18)
	DPP	Medium	0.95 (0.08)	0.98 (0.03)	0.91 (0.11)
	PAM		0.96 (0.08)	0.90 (0.11)	0.71 (0.34)
	DPP	Large	0.96 (0.05)	0.96 (0.03)	0.98 (0.02)
	PAM		0.99 (0.02)	0.87 (0.10)	0.70 (0.15)
$n = 1500$	DPP	Low	0.91 (0.08)	0.89 (0.11)	0.90 (0.10)
	PAM		0.66 (0.26)	0.74 (0.18)	0.76 (0.26)
	DPP	Medium	0.96 (0.04)	0.99 (0.01)	0.88 (0.15)
	PAM		0.98 (0.02)	0.99 (0.01)	0.96 (0.06)
	DPP	Large	0.97 (0.04)	0.96 (0.05)	0.96 (0.02)
	PAM		0.93 (0.12)	0.90 (0.17)	0.69 (0.35)

Table 2.1 – ARI means and standard deviations (within parentheses) over all 24 scenarios with consensus DPP and PAM.

Sample size	Method	Number of clusters	Number of variables		
			Low	Medium	Large
$n = 150$	DPP	Low	0.03 (0.06)	0.04 (0.08)	0.04 (0.06)
	PAM		0.04 (0.08)	0.03 (0.08)	0.09 (0.12)
	DPP	Medium	0.03 (0.05)	0.02 (0.03)	0.04 (0.04)
	PAM		0.00 (0.00)	0.08 (0.08)	0.17 (0.07)
$n = 500$	DPP	Low	0.05 (0.13)	0.04 (0.13)	0.04 (0.08)
	PAM		0.23 (0.21)	0.08 (0.11)	0.13 (0.17)
	DPP	Medium	0.03 (0.06)	0.02 (0.05)	0.04 (0.07)
	PAM		0.02 (0.06)	0.05 (0.06)	0.16 (0.20)
	DPP	Large	0.02 (0.02)	0.02 (0.03)	0.01 (0.01)
	PAM		0.01 (0.01)	0.06 (0.06)	0.14 (0.06)
$n = 1500$	DPP	Low	0.11 (0.13)	0.16 (0.21)	0.18 (0.22)
	PAM		0.50 (0.40)	0.37 (0.28)	0.44 (0.50)
	DPP	Medium	0.02 (0.04)	0.00 (0.00)	0.16 (0.21)
	PAM		0.01 (0.03)	0.00 (0.00)	0.03 (0.08)
	DPP	Large	0.02 (0.02)	0.02 (0.03)	0.01 (0.03)
	PAM		0.03 (0.06)	0.04 (0.07)	0.14 (0.16)

Table 2.2 – RN means and standard deviations (within parentheses) over all 24 scenarios with consensus DPP and PAM.

As noted earlier, there is a clear advantage of using DPP over uniform random sampling for the initial points needed to construct the Voronoi diagrams. This is particularly true when the number of variables is moderate to high. When the number of variables is low, the results yielded by DPP and PAM are similar. Another interesting advantage that we observe is that sampling with DPP contributes to reducing the dispersion of the ARI scores, and then produces more stable optimal clustering configurations. Turning now to the RN means of Table 2.2, the same conclusion applies: sampling with DPP yields better results. The optimal clustering configurations yielded by consensus DPP are associated with estimated number of clusters closer to the true number of clusters than those yielded by PAM. Observe as well, that DPP contributes to the reduction of the variability of the RN values for almost all the cases, as it was already the case with ARI.

2.6. Application to real data

In this section we proceed to evaluate the performance of consensus DPP versus PAM on real datasets. The datasets were obtained from the *UCI Machine Learning Repository* (Dua and Graff, 2017) and *OpenML* website (Vanschoren et al., 2013), two well known databases

in the Machine Learning community for clustering and classification problems. Table 2.3 shows the selected real datasets and some of their features: n = number of observations, K = number of clusters, p = number of variables (i.e., data dimension).

Dataset	n	K	p
Iris	150	3	4
OliveOil	572	9	8
Ecoli	327	5	7
Bank	1372	2	4
Colposcopy	287	3	62
Forest	198	4	27
Breast	569	2	30
Synthetic	600	6	60
Lung cancer	181	2	12533
Yeast Cycle	384	5	17

Table 2.3 – Selected datasets from the UCI and openML repositories

Following the recommendations in (Bicego and Baldo, 2016; Xuan et al., 2013), and due to its strongly unbalanced nature, the Ecoli dataset was transformed using the Box-Cox transformation procedure. Moreover, the original dataset contains $n = 336$ observations with $K = 8$ clusters, but two clusters have only 2 observations, and a third cluster has only 5 observations. These clusters were removed from the data. The Breast and Lung Cancer datasets were also transformed using the Box-Cox transformation. We note that transforming the data is a common procedure for DNA microarray data (Thygesen and Zwinderman, 2004). The Bank dataset has only two clusters, even though it contains $n = 1372$ observations. So using $\sqrt{n} \approx 37$ as a minimal cluster size in the cluster merging stage of the consensus procedure is not optimal. We note that our experiments to select the appropriate parameters for consensus DPP hinted at larger values of the power a when the number of clusters is small. Hence, for these data, we used $n^{2/3} \approx 124$ as the minimal cluster size.

For each dataset, we performed $R = 200$ runs of consensus DPP. The procedure was repeated ten times. For the choice of the optimal clustering configuration, we use the kernel-based validation index $\mathrm{KVI}_{\mathcal{V}}$ defined in (20). To measure the goodness-of-fit of the optimal clustering configuration we use the ARI and RN measures. We compare the consensus DPP results to those of two traditional clustering algorithms: PAM and k-means. Our goal is to show the advantages of the DPP diversity at sampling centroids on the quality of clustering configurations.

The PAM algorithm was already used and mentioned in the study with the simulated datasets of Section 2.5. The k-means algorithm was proposed by Stuart Lloyd in 1957, and later published in (Lloyd, 1982). It starts with an initial set of k means, representing k clusters. It assigns each observation to the corresponding Voronoi cell or cluster given by the

corresponding closer mean among the k means. Once all observations are assigned, the mean vectors of all Voronoi cells are updated, and the process is repeated until there is no change in the means. However, as argued by Celebi et al. (2013), the popular methods for choosing the initial set of k means, such as Forgy (Forgy, 1965), Random Partition (Pena et al., 1999) and Maximin methods (Gonzalez, 1985; Katsavounidis et al., 1994), result often in cluster configurations with a low clustering quality. For that reason we decided to work with the k-means++ algorithm of Arthur and Vassilvitskii (2007), a popular choice mentioned by several authors (Capó et al., 2017; Fränti and Sieranoja, 2019) that avoids the poor quality results of the traditional methods for choosing the initial means. It is based on a simple probabilistic technique. For the consensus clustering with k-means, we proceed as follows: we first sample a number k of Voronoi cells uniformly at random from a finite set of integers $\{1, \ldots, K_{\max}\}$. Then we run k-means++ to obtain an initial set of k means. This step consists of (i) selecting the first center at random from the dataset $\mathcal{S}$; and (ii) repeating the following two steps until a subset of k centers has been sampled: (a) for each $x \in \mathcal{S}$, we compute $dc2(x)$ the square of the Euclidean distance between x and the closest center among those already sampled; (b) a new center is sampled with probability $dc2(x)/\sum_{x' \in \mathcal{S}} dc2(x')$. Once the k centers have been chosen, we proceed as in the standard k-means algorithm described above. We repeat the selection of k initial means $R = 200$ times, just as we do with DPP and PAM, to afterwards apply our consensus clustering methodology to the clustering configurations. As we did with consensus DPP, the optimal cluster configurations from PAM and k-means were chosen using the kernel-based validation index $\mathrm{KVI}_{\mathcal{V}}$ criterion defined in (20). The whole procedure was repeated ten times.

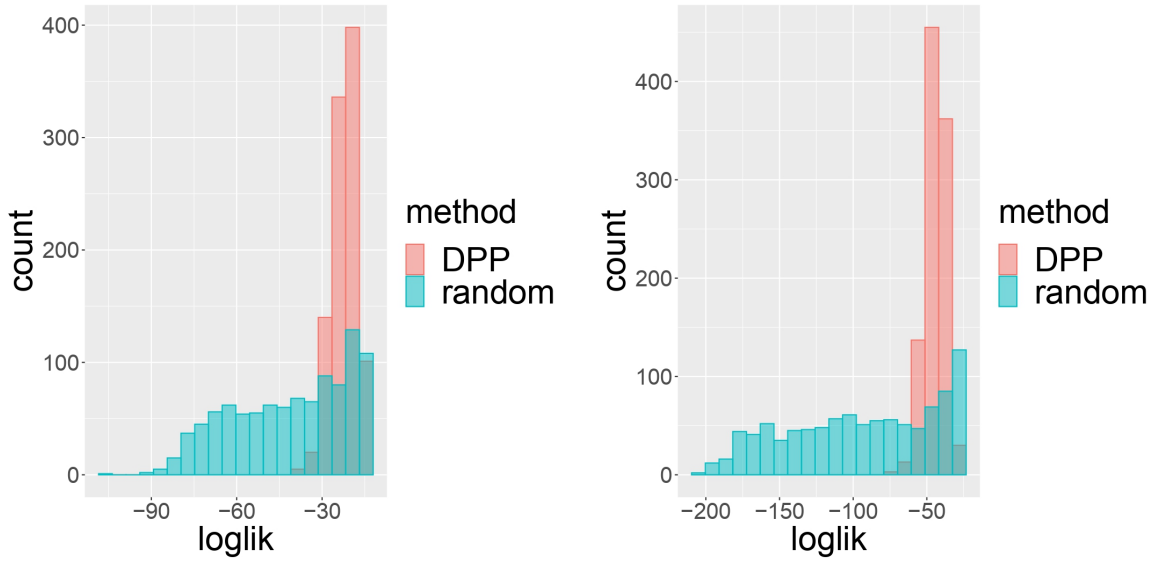
Table 2.4 displays the ARI and RN means and standard deviations obtained by applying consensus DPP, PAM and k-means consensus clustering to the datasets of Table 2.3.

Dataset	Measure	DPP	PAM	k-means
Iris	ARI	0.91 (0.03)	0.83 (0.09)	0.66 (0.05)
	RN	0.03 (0.07)	0.06 (0.08)	0.02 (0.05)
OliveOil	ARI	0.72 (0.06)	0.60 (0.11)	0.68 (0.08)
	RN	0.12 (0.05)	0.10 (0.07)	0.11 (0.03)
Ecoli	ARI	0.76 (0.02)	0.66 (0.09)	0.71 (0.07)
	RN	0.05 (0.06)	0.14 (0.08)	0.04 (0.05)
Bank	ARI	0.66 (0.09)	0.53 (0.19)	0.50 (0.10)
	RN	0.13 (0.12)	0.25 (0.19)	0.24 (0.15)
Colposcopy	ARI	0.44 (0.09)	0.42 (0.14)	0.35 (0.11)
	RN	0.21 (0.09)	0.15 (0.10)	0.16 (0.14)
Forest	ARI	0.86 (0.05)	0.70 (0.01)	0.72 (0.22)
	RN	0.01 (0.04)	0.13 (0.00)	0.08 (0.12)
Breast	ARI	0.61 (0.05)	0.50 (0.13)	0.61 (0.13)
	RN	0.09 (0.12)	0.13 (0.12)	0.07 (0.11)
Synthetic	ARI	0.69 (0.02)	0.66 (0.04)	0.64 (0.01)
	RN	0.09 (0.10)	0.16 (0.08)	0.13 (0.07)
Lung cancer	ARI	0.89 (0.14)	0.84 (0.34)	0.65 (0.42)
	RN	0.09 (0.12)	0.00 (0.00)	0.02 (0.07)
Yeast Cycle	ARI	0.47 (0.004)	0.47 (0.03)	0.44 (0.05)
	RN	0.08 (0.06)	0.10 (0.04)	0.03 (0.05)

Table 2.4 – ARI and RN means and standard deviations (within parentheses) over ten runs associated with consensus DPP, PAM, and k-means.

We observe that consensus DPP yields higher ARI values than the two other methods, and contributes to reducing the variability of this measure, as well. That is, consensus DPP produces more stable and better clustering configurations. The RN results are more balanced in the sense that there is no major difference between the three methods. This means that all three methods hinted at reasonable number of clusters, but not all got good clustering configurations.

As we also did for the simulated data in Section 2.5, we present in Figure 2.3 two typical histograms of the logarithm of the probability mass function of the DPP, given by (12). The datasets in the figure are *Iris* and *Synthetic* (see Table 2.3).



(a) Iris dataset **(b)** Synthetic dataset

Figure 2.3 – Typical histograms of the logarithm of the probability mass function (loglik) using DPP and uniform random sampling, for two real datasets.

As with simulated datasets, we observe that while subsets sampled at random as in PAM result in histograms with a very high dispersion in terms of diversity, DPP tends to select points that maintain a high level of diversity in each sample. DPP seems to be more consistent and stable than uniform random sampling in ensuring the heterogeneity of the elements forming the subsets.

2.7. Conclusions

We explored the potential of determinantal point processes as a sampling method for initializing each run of a consensus clustering algorithm. As a probabilistic model of repulsion, it favors diversity within subsets of points. This is in contrast to uniform random sampling, which gives to every point an equal probability of being selected. Extended simulations showed that, when compared to uniform random sampling, the use of DPPs to generate initial subsets of points results in final clustering configurations with higher and less dispersed quality scores. Applications to real datasets confirm these conclusions draw from simulations.

By using DPPs to generate center point subsets for clustering, the consensus clustering does not require a large number of sampled partitions to ensure a high goodness-of-fit score (e.g., ARI) in the final clustering configurations. In fact, a moderate number of ensemble partitions of about 100 or 200 is sufficient. In contrast, uniform random sampling generally requires a larger number of sampled partitions to reach ARI mean values comparable to

determinantal consensus clustering. For the choice of the final clustering configuration among several candidates, the kernel-based validation index of Fan et al. (2010) has proven to be a good option, outperforming other indexes.

Selecting an appropriate threshold during the merging procedure of the determinantal consensus algorithm is essential. Our simulations show that a good strategy consists in choosing a small subset of diverse thresholds among all the possible threshold values given by the observed consensus indexes. The main advantage of this strategy is to speed up the computations, while preserving the properties associated with keeping all threshold values from the set of all different observed consensus indexes (Murua and Wicker, 2014). Retaining thresholds above 0.6 was adopted as a general choice.

A variety of interesting questions remain for future research: (i) To extend the determinantal consensus clustering to datasets with both continuous and categorical variables, inducing the choice of a proper measure of distance, necessary for the construction of the kernel matrix. In general, for continuous random variables, the Euclidean or Mahalanobis-like distances perform well. For categorical data, Lin's pairwise similarity measure (Lin, 1998) is an attractive alternative to the usual Hamming distance. (ii) To study the effect of multivariate outliers on the mean and dispersion of quality scores of clustering configurations yielded by the determinantal consensus clustering. (iii) To adapt the determinantal consensus clustering to the case of very large datasets. The bottleneck of the method is the eigendecomposition of the kernel matrix. This is a central step for obtaining an initial random subset of points with the determinantal point process. The computational complexity of the eigendecomposition of a $n \times n$ symmetric matrix is $O(n^3)$. As n grows larger, the computation of the matrix spectral decomposition becomes expensive. We explore approximative ways to overcome this challenge with sparse matrix approximation to the kernel matrix. This is the topic we cover in a sequel paper on determinantal consensus clustering.

Bibliography

Ankerst, M., Breunig, M.M., Kriegel, H.P., Sander, J., 1999. Optics: ordering points to identify the clustering structure. ACM Sigmod record 28, 49–60.

Ao, S.I., Yip, K., Ng, M., Cheung, D., Fong, P., Melhado, I., Sham, P., 2005. Clustag: Hierarchical clustering and graph methods for selecting tag snps. Bioinformatics (Oxford, England) 21, 1735–6. doi:10.1093/bioinformatics/bti201.

Arthur, D., Vassilvitskii, S., 2007. K-means++: The advantages of careful seeding, in: Proceedings of the Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms, Society for Industrial and Applied Mathematics, USA. pp. 1027–1035.

Aurenhammer, F., 1991. Voronoi diagrams—a survey of a fundamental geometric data structure. ACM Comput. Surv. 23, 345–405. doi:10.1145/116873.116880.

Banfield, J.D., Raftery, A.E., 1993. Model-based Gaussian and non-Gaussian clustering. Biometrics 49, 803–821.

Ben Hough, J., Krishnapur, M., Peres, Y., Virág, B., 2006. Determinantal processes and independence. Probability Surveys [electronic only] 3, 206–229.

Bicego, M., Baldo, S., 2016. Properties of the box–cox transformation for pattern classification. Neurocomputing 218, 390–400.

Bien, J., Tibshirani, R., 2011. Hierarchical clustering with prototypes via minimax linkage. Journal of the American Statistical Association 106, 1075–1084. doi:10.1198/jasa.2011.tm10183.

Blatt, M., Wiseman, S., Domany, E., 1996. Superparamagnetic clustering of data. Phys. Rev. Lett. 76, 3251–3254.

Blatt, M., Wiseman, S., Domany, E., 1997. Data clustering using a model granular magnet. Neural Comput. 9, 1805–1842.

Borodin, A., Olshanski, G., 2000. Distributions on Partitions, Point Processes, and the Hypergeometric Kernel. Communications in Mathematical Physics 211, 335–358. doi:10.1007/s002200050815, arXiv:math/9904010.

Borodin, A., Soshnikov, A., 2003. Janossy densities determinantal ensembles. Journal of Statistical Physics 113, 595–610. doi:10.1023/A:1026025003309.

Capó, M., Pérez, A., Lozano, J.A., 2017. An efficient approximation to the k-means clustering for massive data. Knowledge-Based Systems 117, 56–69.

Celebi, M.E., Kingravi, H.A., Vela, P.A., 2013. A comparative study of efficient initialization methods for the k-means clustering algorithm. Expert Systems with Applications 40, 200 – 210. URL: http://www.sciencedirect.com/science/article/pii/S0957417412008767, doi:https://doi.org/10.1016/j.eswa.2012.07.021.

Chaudhuri, A., Kakde, D., Sadek, C., Gonzalez, L., Kong, S., 2017. The mean and median criteria for kernel bandwidth selection for support vector data description, in: 2017 IEEE International Conference on Data Mining Workshops (ICDMW), IEEE. pp. 842–849.

Chen, G., Jaradat, S.A., Banerjee, N., Tanaka, T.S., Ko, M.S.H., Zhang, M.Q., 2002. Evaluation and comparison of clustering algorithms in analyzing es cell gene expression data. Statistica Sinica , 241–262.

Daley, D., Vere Jones, D., 2003. An introduction to the theory of point processes. Volume I: Elementary theory and methods. volume 1. 2 ed., Springer.

Dua, D., Graff, C., 2017. UCI machine learning repository. URL: http://archive.ics.uci.edu/ml.

Ester, M., Kriegel, H.P., Sander, J., Xu, X., et al., 1996. A density-based algorithm for discovering clusters in large spatial databases with noise., in: Kdd, pp. 226–231.

Fan, Z., Jiang, X., Xu, B., Jiang, Z., 2010. An automatic index validity for clustering, in: Tan, Y., Shi, Y., Tan, K.C. (Eds.), Advances in Swarm Intelligence, Springer Berlin

Heidelberg, Berlin, Heidelberg. pp. 359–366.

Fasshauer, G., 2011. Positive definite kernels: Past, present and future. Dolomite Res. Notes Approx. 4.

Florek, K., Łukaszewicz, J., Perkal, J., Steinhaus, H., Zubrzycki, S., 1951. Sur la liaison et la division des points d'un ensemble fini, in: Colloquium mathematicum, pp. 282–285.

Forgy, E.W., 1965. Cluster analysis of multivariate data: efficiency versus interpretability of classifications. biometrics 21, 768–769.

Fraley, C., Raftery, A.E., 1998. How many clusters? which clustering method? answers via model-based cluster analysis. Computer Journal 41, 578–588.

Fränti, P., Sieranoja, S., 2019. How much can k-means be improved by using better initialization and repeats? Pattern Recognition 93, 95–112.

Girolami, M., 2002. Mercer kernel-based clustering in feature space. IEEE Transactions on Neural Networks 13, 780–784. doi:10.1109/TNN.2002.1000150.

Gonzalez, T.F., 1985. Clustering to minimize the maximum intercluster distance. Theoretical computer science 38, 293–306.

Hafiz Affandi, R., Fox, E.B., Adams, R.P., Taskar, B., 2014. Learning the Parameters of Determinantal Point Process Kernels. ArXiv e-prints arXiv:1402.4862.

Hafiz Affandi, R., Fox, E.B., Taskar, B., 2013. Approximate Inference in Continuous Determinantal Point Processes. ArXiv e-prints arXiv:1311.2971.

Han, J., Kamber, M., Pei, J., 2011. Data Mining: Concepts and Techniques. 3rd ed., Morgan Kaufmann Publishers Inc., San Francisco, CA, USA.

Hinneburg, A., Keim, D.A., 1999. Optimal grid-clustering: Towards breaking the curse of dimensionality in high-dimensional clustering, in: 25th International Conference on Very Large Databases, pp. 506–517.

Horn, R.A., Johnson, C.R., 2012. Matrix Analysis. 2nd ed., Cambridge University Press, USA.

Howley, T., Madden, M.G., 2006. An evolutionary approach to automatic kernel construction, in: Kollias, S., Stafylopatis, A., Duch, W., Oja, E. (Eds.), Artificial Neural Networks – ICANN 2006, Springer Berlin Heidelberg, Berlin, Heidelberg. pp. 417–426.

Hubert, L., Arabie, P., 1985. Comparing partitions. Journal of Classification 2, 193–218. URL: https://doi.org/10.1007/BF01908075, doi:10.1007/BF01908075.

Jain, A., Dubes, R., 1988. Algorithms for Clustering Data. Prentice-Hall, Inc., Upper Saddle River, NJ, USA.

Kang, B., 2013. Fast Determinantal Point Process Sampling with Application to Clustering, in: Burges, C., Bottou, L., Welling, M., Ghahramani, Z., Weinberger, K. (Eds.), Advances in Neural Information Processing Systems 26. Curran Associates, Inc., pp. 2319–2327.

Katsavounidis, I., Kuo, C.C.J., Zhang, Z., 1994. A new initialization technique for generalized lloyd iteration. IEEE Signal processing letters 1, 144–146.

Kaufmann, L., Rousseeuw, P., 1987. Clustering by means of medoids. Data Analysis based on the L1-Norm and Related Methods , 405–416.

Kulesza, A., Taskar, B., 2012. Determinantal point processes for machine learning. ArXiv e-prints arXiv:1207.6083.

Lanckriet, G.R.G., Cristianini, N., Bartlett, P., Ghaoui, L.E., Jordan, M.I., 2004. Learning the kernel matrix with semidefinite programming. J. Mach. Learn. Res. 5, 27–72.

Lavancier, F., Møller, J., Rubak, E., 2015. Determinantal point process models and statistical inference. Journal of the Royal Statistical Society: Series B (Statistical Methodology) 77, 853–877.

Lin, D., 1998. An information-theoretic definition of similarity, in: Proceedings of the 15th International Conference on Machine Learning, Morgan Kaufmann, San Francisco, CA, pp. 296–304.

Lloyd, S.P., 1982. Least squares quantization in pcm. IEEE Trans. Inf. Theory 28, 129–136.

Macchi, O., 1975. The Coincidence Approach to Stochastic Point Processes. Advances in Applied Probability 7, 83–122.

Melnykov, V., Chen, W.C., Maitra, R., 2012. Mixsim: An r package for simulating data to study performance of clustering algorithms. Journal of Statistical Software, Articles 51, 1–25. URL: https://www.jstatsoft.org/v051/i12, doi:10.18637/jss.v051.i12.

Monti, S., Tamayo, P., Mesirov, J., Golub, T., 2003. Consensus clustering: A resampling-based method for class discovery and visualization of gene expression microarray data. Machine Learning 52, 91–118. doi:10.1023/A:1023949509487.

Muñoz, J., Murua, A., 2018. Building cancer prognosis systems with survival function clusters. Statistical Analysis and Data Mining: The ASA Data Science Journal 11, 98–110. URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/sam.11373, doi:10.1002/sam.11373, arXiv:https://onlinelibrary.wiley.com/doi/pdf/10.1002/sam.11373.

Murua, A., Stanberry, L., Stuetzle, W., 2008. On potts model clustering, kernel k-means, and density estimation. Journal of Computational and Graphical Statistics 17, 629–658. doi:10.1198/106186008X318855.

Murua, A., Wicker, N., 2014. The Conditional-Potts Clustering Model. Journal of Computational and Graphical Statistics 23, 717–739. doi:10.1080/10618600.2013.837828.

Okabe, A., Boots, B., Sugihara, K., Chiu, S.N., 2000. Spatial Tessellations: Concepts and Applications of Voronoi Diagrams. Series in Probability and Statistics. 2nd ed. ed., John Wiley and Sons, Inc.

Pena, J.M., Lozano, J.A., Larranaga, P., 1999. An empirical comparison of four initialization methods for the k-means algorithm. Pattern recognition letters 20, 1027–1040.

Rand, W.M., 1971. Objective criteria for the evaluation of clustering methods. Journal of the American Statistical Association 66, 846–850. URL: http://www.jstor.org/stable/2284239.

Schölkopf, B., Tsuda, K., Vert, J.P., 2004. Kernel methods in computational biology. MIT Press, Cambridge, Mass.

Sejdinovic, D., Sriperumbudur, B., Gretton, A., Fukumizu, K., 2013. Equivalence of distance-based and rkhs-based statistics in hypothesis testing. The Annals of Statistics , 2263–2291.

Smyth, P., 1997. Clustering sequences with hidden markov models, in: Advances in neural information processing systems, pp. 648–654.

Strehl, A., Ghosh, J., 2002. Cluster ensembles — a knowledge reuse framework for combining multiple partitions. J. Mach. Learn. Res. 3, 583–617. doi:10.1162/153244303321897735.

Stuetzle, W., 2003. Estimating the cluster tree of a density by analyzing the minimal spanning tree of a sample. Journal of Classification 20, 25–47.

Stuetzle, W., Nugent, R., 2010. A generalized single linkage method for estimating the cluster tree of a density. Journal of Computational and Graphical Statistics 19, 397–418.

Thygesen, H.H., Zwinderman, A.H., 2004. Comparing transformation methods for dna microarray data. BMC bioinformatics 5, 77.

Vanschoren, J., van Rijn, J.N., Bischl, B., Torgo, L., 2013. Openml: Networked science in machine learning. SIGKDD Explorations 15, 49–60. URL: http://doi.acm.org/10.1145/2641190.2641198, doi:10.1145/2641190.2641198.

Vapnik, V.N., 1995. The Nature of Statistical Learning Theory. Springer-Verlag, Berlin, Heidelberg.

Vega-Pons, S., Ruiz-Shulcloper, J., 2011. A survey of clustering ensemble algorithms. International Journal of Pattern Recognition and Artificial Intelligence 25, 337–372. doi:10.1142/S0218001411008683.

Vert, J.P., Tsuda, K., Schölkopf, B., 2004. A primer on kernel methods. Kernel methods in computational biology 47, 35–70.

Wang, F., Landau, D.P., 2001. Efficient, multiple-range random walk algorithm to calculate the density of states. Physical review letters 86, 2050.

Wang, W., Yang, J., Muntz, R., et al., 1997. Sting: A statistical information grid approach to spatial data mining, in: VLDB, pp. 186–195.

Xuan, L., Zhigang, C., Fan, Y., 2013. Exploring of clustering algorithm on class-imbalanced data, in: 8th International Conference on Computer Science and Education, ICCSE 2013, pp. 89–93. doi:10.1109/ICCSE.2013.6553890.

Yeung, K.Y., Fraley, C., Murua, A., Raftery, A.E., Ruzzo, W.L., 2001. Model-based clustering and data transformations for gene expression data. Bioinformatics 17, 977–987.

Chapitre 3

Large-data determinantal clustering

par

Sergio Vicente[1] et Alejandro Murua[2]

([1]) Département de Mathématiques et de Statistique, Université de Montréal.
E-mail : vicentes@dms.umontreal.ca

([2]) Département de Mathématiques et de Statistique, Université de Montréal.
E-mail : murua@dms.umontreal.ca

Cet article sera soumis dans Journal of Computational and Graphical Statistics.

Résumé. Le partitionnement par consensus déterminantal est une méthode ensembliste offrant une alternative prometteuse et attractive par rapport au partitionnement des k-médoïdes et des k-moyennes. Comme il se base sur l'échantillonnage par le processus ponctuel déterminantal ou DPP, les points très similaires auront une probabilité plus petite d'être sélectionnés simultanément comme centroïdes . Il favorise des sous-ensembles de points diversifiés. L'algorithme d'échantillonnage du processus ponctuel déterminantal exige la décomposition d'une matrice de Gram en éléments propres. Cette tâche devient intense d'un point de vue computationnel lorsque la taille des données est très grande. Cela pose un problème additionnel dans le partitionnement par consensus étant donné qu'un certain algorithme de partitionnement est exécuté plusieurs fois de façon à produire un partitionnement final consolidé. On propose deux alternatives efficaces pour pouvoir exécuter le partitionnement par consensus sur des données de grande taille. Ces méthodes sont basées sur l'échantillonnage au moyen du DPP, à partir de matrices noyau creuses et petites, dont la distribution des valeurs propres est proche de celle de la matrice de Gram originale.
Mots clés : Classification ; décomposition spectrale ; divergence de Kullback-Leibler ; Factorisation de Cholesky ; k plus proches voisins ; processus Gaussien des plus proches voisins ; sparsité.

Abstract. Determinantal consensus clustering is a promising and attractive alternative to partitioning about medoids and k-means for ensemble clustering. Based on a determinantal point process or DPP sampling, it ensures that subsets of similar points are less likely to be selected as centroids. It favors more diverse subsets of points. The sampling algorithm of the determinantal point process requires the eigendecomposition of a Gram matrix. This becomes computationally intensive when the data size is very large. This is particularly an issue in consensus clustering, where a given clustering algorithm is run several times in order to produce a final consolidated clustering. We propose two efficient alternatives to carry out determinantal consensus clustering on large datasets. They consist in DPP sampling based on sparse and small kernel matrices whose eigenvalue distributions are close to that of the original Gram matrix.
Keywords: Cholesky factorization; classification; k-nearest neighbors; Kullback-Leibler divergence; nearest-neighbors Gaussian process; sparseness; spectral decomposition.

3.1. Introduction

Cluster analysis is a classical procedure to aggregate elements according to their similarity. Among the most popular methods to do clustering, we find the well-known k-means (Lloyd, 1982) and the partitioning around medoids (PAM) (Kaufmann and Rousseeuw, 1987) algorithms. These two procedures have the particularity of building data partitions from an initial choice of random points called centroids. These points are usually sampled uniformly at random from the set of all datapoints. Since each point has the same probability of being chosen, the initial sample may end up with a set of points containing many similar points that carry the same type of information. That is, the initial sample might not represent the diversity present in the data. This might affect the effectiveness of the clustering.

Nowadays, the diversity in the selected elements is a major concern in some domains of research, like clinical trials (Clark et al., 2019), forensic sciences (Wagstaff and LaPorte, 2018) or educational development (Szelei et al., 2019). Adopting uniform random sampling as a sampling mechanism can result in sets of elements with a poor coverage of all the facets of a population under study. Determinantal point processes, or DPPs for short, introduced by Borodin and Olshanski (2000), can address this problem. DPPs model negative correlations between points so that similar elements have less chances of being simultaneously sampled. The negative correlations are captured by the so-called kernel or Gram matrix (Kulesza and Taskar, 2012), a matrix whose entries represent a measure of similarity between pair of points. DPPs have already been adopted in machine learning as models for subset selection (Hafiz Affandi et al., 2013; Gartrell et al., 2018; Shah and Ghahramani, 2013; Gillenwater et al., 2012; Mariet et al., 2019).

The origins of DPPs can be found in quantum physics (Macchi, 1975). Known first as *fermion processes,* they model the distribution of fermion systems at thermal equilibrium. Much later, Borodin and Olshanski (2000) introduced the now accepted *Determinantal Point Process* terminolgy in the mathematics community. DPPs have also been applied to problems dealing with nonintersecting random paths (Daley and Vere Jones, 2003), random spanning trees (Borodin and Soshnikov, 2003), and the study of eigenvalues of random matrices (Ben Hough et al., 2006).

Clustering algorithms like k-means and the partitioning around medoids result in single partition of data, seeking to maximize intra-cluster similarity and inter-cluster dissimilarity. However, the clustering results of two different algorithms can be very different. The lack of an external objective and impartial criterion can explain those differences (Vega-Pons and Ruiz-Shulcloper, 2011). The dependence on the initial choice of centroids is also an important factor. Blatt et al. (1996) and Blatt et al. (1997) proposed a new approach to improve the quality and robustness of clustering results. The approach was later formalized by Strehl and Ghosh (2002), where the notion of *cluster ensembles* is introduced. Cluster

ensembles combine different data partitions into a single consolidated clustering. A particular cluster ensemble method was later introduced in Monti et al. (2003), the so-called *consensus clustering*. This method consists in performing multiple runs of the same clustering algorithm on the same data, to produce a single clustering configuration by agreement among all the runs.

Most often, the particular clustering algorithm to be run several times for consensus implies random initial conditions or centroids. Recently Vicente and Murua (2020) introduced one such method, the *determinantal consensus clustering* or *consensus DPP* procedure. This generates centroids through a DPP process. The similarity or distance between the datapoints is incorporated in the similarity matrix, also known as the kernel or Gram matrix, which constitutes the core of the DPP process. Moreover, the link between similarity matrices and kernel methods for statistical or machine learning makes the method very flexible and effective at discovering data clusters. The diversity within centroids is automatically inherited in the DPP sampling. The DPP "diversity at sampling" property has shown to greatly improve the consensus clustering results (Vicente and Murua, 2020).

In practice, the centroids are drawn using the DPP sampling algorithm described in Ben Hough et al. (2006) and Kulesza and Taskar (2012). This algorithm is based on computing the spectral decomposition of the data similarity or kernel matrix. Unfortunately, when the data size n is very large, the eigendecomposition becomes a computational burden. The computational complexity of the eigendecomposition of a $n \times n$ symmetric matrix is of order $O(n^3)$. However, to sample the centroid points, we might not need to compute all eigenvalues and eigenvectors of the kernel matrix. In fact, the probability of selecting each datapoint depends on a corresponding eigenvalue. Datapoints associated with relatively very small eigenvalues are selected with very low probability. Therefore, the key to reduce the computational burden induced by large datasets is the extraction of the largest eigenvalues of the associated kernel matrix. This raises the necessity to deliver algorithms able to extract such a subset of eigenvalues. One of the most used algorithms to extract the largest eigenvalues is the Lanczos algorithm (Lanczos, 1950). Due to his proven numerical instability, many variations of it have been proposed. A popular variation of the Lanczos algorithm is the *implicitly restarted Lanczos method* (Calvetti et al., 1994) which we adopte in this paper.

The Lanczos algorithm have been specially developed for large sparse and symmetric matrices. Hence, to be able to perform determinantal consensus clustering on large datasets, we need to find good sparse approximations of the often dense *original* kernel matrix. We propose two sound approaches: one approach based on the Nearest Neighbor Gaussian Process of Datta et al. (2016), and another approach based on a random sampling of small submatrices from the dense kernel matrix. This latter approach may be seen as a kind of divide and conquer approach. Although we show that these approaches offer good approximations to the eigenvalue distribution of the original kernel matrix, our goal is rather to introduce

alternative efficient DPP sampling models for large datasets that inherit the data diversity expressed in the original kernel matrix.

The paper is organized as follows: in Section 3.2, we summarize the consensus clustering and recall the basic characteristics of determinantal point processes. The determinantal consensus clustering is also summarized in this section. In Section 3.3, we introduce the problem of using the determinantal point process in the context of large datasets, and present two approaches to address this issue. In Section 3.4, we evaluate the two approaches presented in Section 3.3 through large dataset simulations; here we also illustrate the concept of diversity introduced by the determinantal point process. A performance comparison between our two approaches and two other competing methods on large real datasets is shown in Section 3.5. We conclude with a few thoughts and a discussion in Section 3.6.

3.2. Determinantal consensus clustering

3.2.1. Consensus clustering

Throughout the paper, the data will be denoted by $\mathcal{S} = \{x_1, \ldots, x_n\} \subset \mathbb{R}^p$, where x_i represents a p-dimensional vector, for $i = 1, \ldots, n$ and $n \geq 2$. Consider a particular clustering algorithm run R times on the same data $\mathcal{S}$. The agreement among the several runs of the algorithm is based on the *consensus matrix* C. This is a $n \times n$ symmetric matrix whose entries $\{C_{ij},\ i, j = 1, \ldots, n\}$ represent the proportion of runs in which elements x_i and x_j of $\mathcal{S}$ fall in the same cluster. Let r represent a specific run of the clustering algorithm, $r = 1, \ldots, R$, and let C_r be the associated $n \times n$ symmetric binary matrix with entries $c^r_{ij} = 1$ if x_i and x_j belong to the same cluster in the rth run of the algorithm, and $c^r_{ij} = 0$, otherwise, $i, j = 1, \ldots, n$. The components of the consensus matrix C are given by $C_{ij} = \sum_{r=1}^{R} c^r_{ij}/R$, $i, j = 1, \ldots, n$. The entry C_{ij} is known as *consensus index*. The diagonal entries are given by $C_{ii} = 1$, for $i = 1, \ldots, n$.

Our interest is to extend the determinantal consensus clustering (or consensus DPP) algorithm of Vicente and Murua (2020) to large datasets. Consensus DPP is a modified version of Partitioning around medoids (PAM) algorithm (Kaufmann and Rousseeuw, 1987) in which the center points are sampled with a determinantal point process (DPP); more details on DPP are shown in the next section. Each run starts with a Voronoi diagram (Aurenhammer, 1991) on the set $\mathcal{S}$. This partitions the space into several cells or regions based on a random subset of *generator points*. After R runs of the algorithm, we obtain R partitions associated with the R Voronoi diagrams. The consensus matrix C is computed from these partitions. Let $\theta \in [0,1]$ be a proportion threshold. According to Blatt et al. (1996), if $C_{ij} \geq \theta$, points x_i and x_j are defined as "friends" and are included in the same consensus cluster. Moreover, all mutual friends (including friends of friends, etc.) are assigned to the same cluster. To

select an appropriate threshold value, we follow Murua and Wicker (2014) and consider all threshold values given by the set of all different observed consensus indexes C_{ij}. If there are t different consensus indexes, we will have a collection of t thresholds $\theta_1, \theta_2, \ldots, \theta_t$. For each threshold $\theta_i,\ i = 1, \ldots, t$, a consensus clustering configuration with $K(\theta_i)$ clusters is obtained. If $\theta_i = 0$, we obtain a configuration with $K(0) = 1$ cluster, that is, a single cluster equal to $\mathcal{S}$. If $\theta_i = 1$, we obtain a configuration with $K(1) = n$ singleton clusters; that is, each element of $\mathcal{S}$ forms a singleton cluster. In general, clustering configurations with one cluster or n clusters are of no interest. Therefore, thresholds θ_i that are too low or too large are not relevant. This observation leads to a more efficient procedure (Vicente and Murua, 2020) where only a predetermined sequence of t thresholds $\tau < \theta_1 < \theta_2 < \cdots < \theta_t$, bounded from below by a certain minimum threshold τ, are considered. This approach is particularly useful in the context of large datasets, since reducing the range of considered thresholds can reduce the computational burden induced by the high number of elements.

Moreover, we are not interested in a clustering configuration with too many small clusters. Only clustering configurations with cluster sizes larger than a minimal value are admissible (that is, accepted). Vicente and Murua (2020) established through extensive simulations that the "square-root choice" (for the number of bins of a histogram) $\sqrt{n}$ is an adequate value for the minimum cluster size. Each one of the t consensus clustering configurations obtained with the t thresholds $\{\theta_i\}$ are examined to verify that they are admissible. For every non-admissible consensus clustering configuration we merge each small cluster with its closest "large" cluster, according to the following procedure, inspired by single linkage: (i) select the cluster $\mathcal{V}$ that has the smallest cluster size $< \sqrt{n}$; (ii) find the pair of indexes $(i^*, j^*) \in \{1, \ldots, n\}$ that satisfies $C_{i^*j^*} = \max\{C_{ij} : x_i \in \mathcal{V}, x_j \notin \mathcal{V}\}$; (iii) merge the cluster $\mathcal{V}$ to the cluster that includes x_{j^*}; (iv) repeat the merging procedure until there are no more clusters with size smaller than $\sqrt{n}$. Once all t consensus clustering configurations are made admissible, we proceed to select the *final consensus clustering* as the consensus clustering configuration among these t configurations that minimizes the kernel-based validation index of Fan et al. (2010). This index was conceived after the studies of Girolami (2002) to choose the optimal final cluster among several possible partitions. It can be seen as an index that combines modified extensions of the between and within variances to kernel-based methods. See Fan et al. (2010) or Vicente and Murua (2020) for further details.

3.2.2. The determinantal point process

Let L be a $n \times n$ real symmetric and positive semidefinite matrix that measures similarity between all pairs of elements of $\mathcal{S}$. We denote by $L_Y = (L_{ij})_{i,j \in Y}$ the principal submatrix of L whose rows and columns are indexed by the subset $Y \subseteq \mathcal{S}$. A determinantal point process,

DPP for short, is a probability measure on $2^{\mathcal{S}}$ that assigns probability

$$P(Y) = \det(L_Y)/\det(L + I_n), \tag{21}$$

to any subset $Y \in 2^{\mathcal{S}}$, where I_n is the identity matrix of dimension $n \times n$. We write $\boldsymbol{Y} \sim DPP_{\mathcal{S}}(L)$ for the corresponding determinantal process.

The matrix L is known as the kernel matrix of the DPP (Kulesza and Taskar, 2012; Kang, 2013; Hafiz Affandi et al., 2014). It can be shown (Kulesza and Taskar, 2012) that $\det(L+I_n) = \sum_{Y \subset 2^{\mathcal{S}}} \det(L_Y)$, hence (21) does indeed define a probability mass function over all subsets in $2^{\mathcal{S}}$. This definition states restrictions on all the principal minors of the kernel matrix L, denoted by $\det(L_Y)$. Indeed, as $P(\boldsymbol{Y} = Y) \propto \det(L_Y)$ represents a probability measure, we have $\det(L_Y) \geq 0$, for any $Y \subseteq \mathcal{S}$.

Any symmetric positive semidefinite matrix L may be a kernel matrix of a DPP. For the construction of the similarity matrix L in (21), we use a suitable Mercer Kernel (Girolami, 2002) as indicated in Vicente and Murua (2020). The choice of an appropriate kernel is a critical step in the application of any kernel-based method. However, as pointed out by Howley and Madden (2006), there is no rule nor consensus about its choice. Ideally, the kernel must be chosen according to prior knowledge of the problem domain (Howley and Madden, 2006; Lanckriet et al., 2004), but this practice is rarely observed. In the absence of expert knowledge, a common choice is the *Radial Basis Function* (RBF) kernel (or *Gaussian kernel*):

$$L = \Big(\exp\Big\{-\|x_i - x_j\|^2/(2\sigma^2)\Big\}\Big)_{i,j=1}^{n},$$

where the scale parameter σ, known as the kernel's bandwidth, represents the relative spread of the Euclidean distances $\|x_i - x_j\|$ between any two points x_i and x_j. Due to its appealing mathematical properties, this particular kernel has been extensively used in many studies. A particular property of the Gaussian kernel is that it is positive and bounded from above by one, making it directly interpretable as a scaled measure of similarity between any given pair of points.

The computation of the RBF kernel requires the estimation of the bandwidth parameter σ. As pointed by Murua and Wicker (2014), most of the literature considers σ as a parameter that can be estimated by observed data. Inspired by Blatt et al. (1996) and Blatt et al. (1997), we estimate σ^2 by the average of all pairwise and squared Euclidean distances, i.e., $\widehat{\sigma}^2 = 2\sum_{i<j} \|(x_i - x_j)\|^2/\Big(n(n-1)\Big)$. Other choices of estimating σ^2 are referred in Vicente and Murua (2020). We chose the average for its simplicity and fast computation even when the dataset is large.

3.3. The case of large datasets

Ben Hough et al. (2006) and Kulesza and Taskar (2012) present an efficient scheme to sample from a DPP. The algorithm is based on the following observations. Let $L = \sum_{i=1}^{n} \lambda_i(L) v_i v_i^T$ be an orthonormal eigendecomposition of L, where $\lambda_1(L) \geq \lambda_2(L) \geq \cdots \geq \lambda_n(L) \geq 0$ are the eigenvalues of L, and $\{v_i : i = 1, \ldots, n\}$ are the eigenvectors of L. For any set of indexes $J \subseteq \{1,2,\ldots,n\}$, define the subset of eigenvectors $V_J = \{v_i : i \in J\}$, and the associated matrix $\mathcal{K}_J = \sum_{i \in J} v_i v_i^T$. It can be shown that the matrix $\mathcal{K}_J$ defines a so-called *elementary* DPP which we denote by $\mathrm{DPP}(\mathcal{K}_J)$. It turns out that the $\mathrm{DPP}_{\mathcal{S}}(L)$ is a mixture of all elementary DPP given by the index sets J. That is

$$\mathrm{DPP}_{\mathcal{S}}(L) = \sum_{J} \mathrm{DPP}(\mathcal{K}_J) \left[\prod_{i \in J} \lambda_i(L) \right] / \det(L + I_n).$$

The mixture weight of $\mathrm{DPP}(\mathcal{K}_J)$ is given by the product of the eigenvalues $\lambda_i(L)$ corresponding to the eigenvectors $v_i \in V_J$, normalized by $\det(L + I_n) = \prod_{i=1}^{n} [\lambda_i(L) + 1]$. Sampling of a subset $\boldsymbol{Y} \sim \mathrm{DPP}(L)$ can be realized by first selecting an elementary DPP, $\mathrm{DPP}(\mathcal{K}_J)$, with probability equal to its mixture component weight, and then, in a second step, sampling a subset from $\mathrm{DPP}(\mathcal{K}_J)$. Moreover, it can be shown that the expected value and variance of the number of elements in $\boldsymbol{Y}$, $\mathrm{card}(\boldsymbol{Y})$, are given by

$$\mathbb{E}[\mathrm{card}(\boldsymbol{Y})] = \sum_{i=1}^{n} \frac{\lambda_i(L)}{\lambda_i(L)+1} \; ; \; \mathrm{Var}[\mathrm{card}(\boldsymbol{Y})] = \sum_{i=1}^{n} \frac{\lambda_i(L)}{(\lambda_i(L)+1)^2}.$$

It is well known that the computational complexity of obtaining the eigendecomposition of a $n \times n$ symmetric matrix is of order $O(n^3)$ and, as n grows larger, the computation of the eigenvalues and eigenvectors becomes expensive. Even the storage of the matrix is limited by the memory required.

The sampling algorithm based on DPP starts with a subset of the eigenvectors of the kernel matrix, selected at random, where the probability of selecting each eigenvector depends on its associated eigenvalue. This fact suggests directly that it is unnecessary to compute all the eigenvalues, as eigenvectors with low associated eigenvalues are selected with low probability. This is particularly useful in the case of large matrices: computing only the largest eigenvalues can substantially reduce the computational burden of obtaining all the eigenvalues. The literature points to many references of well-known algorithms that can extract the t largest (or smallest) eigenvalues, with their associated eigenvectors, of a $n \times n$ Hermitian matrix, where usually, we have $t \ll n$. One of the most classical and used algorithms is the Lanczos algorithm (Lanczos, 1950). Despite its popularity and computational efficiency, the algorithm was proven to be numerically instable, due in part to the loss of orthogonality of the computed Krylov subspace basis vectors generated. Since then, many

efforts have been made to solve this issue, like Cullum (1978), Parlett and Nour-Omid (1989) or Grimes et al. (1994). Many of the variations of the Lanczos algorithm propose a restart after a certain number of iterations. One of the most popular restarted variations of the algorithm is the implicitly restarted Lanczos method, proposed by Calvetti et al. (1994), which is implemented in ARPACK (Lehoucq et al., 1998), motivating then our preference for this particular variation.

The Lanczos algorithm and its implicitly restarted variation were specially developed for large sparse symmetric matrices. Consequently, our first step before implementing the implicitly restarted Lanczos method is to find a good approximation of the kernel matrix L by a sparse matrix. We decide to address this question by following two approaches: one approach based on the Nearest Neighbor Gaussian Process Datta et al. (2016) and another approach based on random sampling of small submatrices from the dense kernel matrix L.

3.3.1. The Nearest Neighbor Gaussian Process

The Nearest Neighbor Gaussian Process (NNGP) was developed by Datta et al. (2016) and extended by Finley et al. (2019), to obtain a sparse approximation of the kernel matrix L, say $\widetilde{L}$, to which the implicitly restarted Lanczos method can be applied to obtain its largest eigenvalues.

Finley et al. (2019) showed that the covariance matrix W of a Gaussian process can be expressed through a specific Cholesky decomposition:

$$W = (I_n - A)^{-1} D (I_n - A)^{-T}, \tag{22}$$

where A is a $n \times n$ strictly lower-triangular matrix and D is a $n \times n$ diagonal matrix; here $(\cdot)^{-T}$ stands for the inverse of the transposed matrix. In order to define properly the matrices A and D we need to introduce the following notation. For any $n \times n$ matrix M, and a subset of indices $J \subseteq \{1, \ldots, n\}$, we will write $M_{k,J} = (M_{kj})_{j \in J}$ for the card(J)-dimensional vector formed by the corresponding components of the row k of M, $k = 1, \ldots, n$. We define similarly, $M_{J,k}$. Also, we write $[i_1 : i_2]$ for the set $J = \{j : i_1 \leq j \leq i_2\}$.

Having introduced the notation, we can write the ith row of A, $A_{i\bullet}$ as $A_{i,[1:i-1]} = W^{-1}_{[1:i-1]} W_{[1:i-1],i}$, for $i = 2, \ldots, n$, and $A_{i,[i:n]} = 0$, for $i = 1, \ldots, n$. where $W_{[1:k]}$ represents the leading principal submatrix of order k of the matrix W. The diagonal entries D_{ii} of D are such that $D_{11} = W_{11}$ and $D_{ii} = W_{ii} - W_{i,[1:i-1]} A^T_{i,[1:i-1]}$, for $i = 2, \ldots, n$.

Note that these equations are the linear equations that define the matrices A and D. These equations need to be solve for A and D in order to obtain the decomposition given by the expression in (22). Unfortunately, the computation of $A_{i\bullet}$ still takes $O(n^3)$ floating point operations, specially for high values of i closer to n, which increases the dimension of W_{i-1}. Despite this shortcoming, the authors mention that this specific decomposition highlights where the sparseness can be exploited: setting to zero some elements in the lower triangular

part of A. This is achieved by limiting the number of nonzero elements in each row of A to a maximum of m elements.

Let $N_i \subseteq \{1, \ldots, n\}$ be the set of indices $j < i$ for which $A_{i,j} \neq 0$. We constraint N_i to have at most m indices. In this latter case, all elements of the ith row $A_{i\bullet}$ of A are zero, except for the elements $A_{i,N_i} = W_{N_i}^{-1} W_{N_i,i}$, for $i = 2, \ldots, n$, where W_{N_i} is the principal submatrix of W whose rows and columns are indexed by N_i. For the diagonal entries, we have $D_{11} = W_{11}$ and $D_{ii} = W_{ii} - W_{i,N_i} A_{i,N_i}^T$, for $i = 2, \ldots, n$.

These latter equations form a linear system of size at most $m \times m$, with $m = \max_i (\text{card}(N_i))$. This new system can be solved for A and D in $O(nm^3)$ floating points operations. Using these solutions gives rise to the sparse approximation to the precision matrix W^{-1}

$$\widetilde{W}^{-1} = (I_n - A)^T D^{-1} (I_n - A). \tag{23}$$

The inverse of $\widetilde{W}^{-1}$ is an approximation to W. Datta et al. (2016) show in the context of spatial Gaussian processes, that $\widetilde{W}^{-1}$ has at most $nm(m+1)/2$ nonzero entries. Thus, $\widetilde{W}^{-1}$ is sparse provided that $m \ll n$. We can apply this result to develop an efficient determinantal consensus clustering (see Section 3.2) when the data size n is large. Recall that the kernel matrix L is a real symmetric positive semidefinite matrix. When L is also positive definite, it can be seen as a covariance matrix. This is what we assume from now on. Therefore, it can be approximated using the NNGP approach by a matrix $\widetilde{L}$ whose inverse $\widetilde{L}^{-1}$ is sparse.

For each $i \in \{2, \ldots, n\}$ consider the distances $d_{ij} = \|x_i - x_j\|$ for $j < i$. Let $d_{i(1)} \leq d_{i(2)} \leq \cdots \leq d_{i(i-1)}$ be the corresponding sequence of ordered distances. In our model we set $N_i = \{j : j < i, d_{ij} \leq d_{i(m)}\}$, for $i > m$; and set $N_i = \{1, \ldots, i-1\}$ for $i \leq m$. Let $\check{L}$ be the m-nearest-neighbor matrix whose row entries are given by $\check{L}_{ii} = L_{ii}$, $\check{L}_{ij} = L_{ij}$ if $j \in N_i$, and $\check{L}_{ij} = 0$, otherwise. We would like to stress here that the matrix $\widetilde{L}$ based on the neighborhoods $\{N_i\}_{i=1}^n$ is not the same as the m-nearest-neighbor matrix $\check{L}$. We have adopted $\widetilde{L}$ instead of $\check{L}$ for several reasons. First, $\widetilde{L}$ is a dense approximation of L, whose inverse $\widetilde{L}^{-1}$ is sparse. Second, $\widetilde{L}^{-1}$ is a sparse matrix with $O(nm^2)$ nonzero entries. On the other hand, $\check{L}$ is a sparse matrix with $O(nm_{nn})$ nonzero entries, where m_{nn} is the number of nearest neighbors. Therefore, for a fixed level of sparseness, building $\widetilde{L}^{-1}$ requires many fewer nearest neighbors $m = O(\sqrt{m_{nn}})$ than building $\check{L}$. Finally, in Section 3.4, we compute both the Frobenius distances (Horn and Johnson, 2012) $\|L - \widetilde{L}\|_F$, and $\|L - \check{L}\|_F$, and show that the former is always smaller than the latter, for all our simulated data. Moreover, in terms of the symmetrized Kullback-Leibler divergence (Kullback and Leibler, 1951), the distribution of the eigenvalues of $\widetilde{L}$ is closer to the distribution of the eigenvalues of L than the distribution of the eigenvalues of $\check{L}$.

As our primary goal is to obtain the $t \ll n$ largest eigenvalues of the kernel matrix L, we start with the construction of the sparse matrix $\widetilde{L}^{-1}$ using the formula in (23), and

the sparse computation form above. We then apply the Lanczos algorithm to extract the t smallest eigenvalues of $\widetilde{L}^{-1}$, and their associated eigenvectors. By inverting the eigenvalues, we obtain the t largest eigenvalues of $\widetilde{L}$; the eigenvectors of $\widetilde{L}$ are the same as those of $\widetilde{L}^{-1}$. With the eigenvalues and eigenvectors in hand, we can proceed as usual with the determinantal consensus clustering of Section 3.2. We stress here that the actual DPP used for the determinantal consensus clustering after this construction is $DPP_{\mathcal{S}}(\widetilde{L})$, and not $DPP_{\mathcal{S}}(L)$. In pratice, L is chosen only to give us a measure of similarity between data points.

3.3.2. Approach based on random sampling of small submatrices from L

In this section, we consider another approach to deal with large datasets and kernel matrices. In this approach we combine dimension reduction techniques and the advantages of working with sparse matrices. Let $L^{(1)}, \ldots, L^{(M)}$ denote M $r \times r$ submatrices sampled uniformly at random and without replacement from L, where $r < n$ (ideally, $r \ll n$). The idea is to use these submatrices as proxies for L in the DPP sampling.

By generating a sufficiently large number M of matrices, we expect to cover the set of eigenvectors and eigenvalues of L with the smaller sets of eigenvectors and eigenvalues of the matrices collection $\{L^{(i)} : i = 1, \ldots, M\}$. This might be the case if the data are well separated so that, although L might be a *dense* matrix, its hidden structure might be *sparse*. That is, many entries in L might be small. We could also achieve sparseness by thresholding the elements of L. However, the following approach yielded better results in our experiments (not shown here), and hence, it is the second approach adopted (the first having been described in the previous section).

We apply the following iterative methodology to the set of submatrices, for a number N of times. For $k = 1, \ldots, N$:

(1) Select an index i_k from $\{1, 2, \ldots, M\}$ at random (with replacement), and consider the submatrix $L^{(i_k)}$.

(2) Build a sparse approximation $\widehat{L}^{(i_k)}$ of the submatrix $L^{(i_k)}$ by considering the k-nearest neighbors of each point associated with the rows of the submatrix; that is, $\widehat{L}_{ij}^{(i_k)} = L_{ij}^{(i_k)}$ if x_j is one of the k-nearest-neighbors of x_i or if x_i is one of the the k-nearest-neighbors of x_j; $\widehat{L}_{ij}^{(i_k)} = 0$, otherwise.

(3) Generate a subset sample Y_{i_k} from a DPP($\widehat{L}^{(i_k)}$) based only on the t largest eigenvalues extracted with the Lanczos algorithm.

(4) Find the Voronoi cells of the n data points based on the sampled Y_{i_k} center points. This generates the kth partition of the algorithm.

At the end of this procedure, we apply the determinantal consensus clustering of Section 3.2 to the set of N partitions obtained.

The number M of submatrices to be sampled must be chosen so that we get benefits from using the submatrices to sample the generator sets through DPP rather using the whole kernel matrix L. We know that the computational complexity of obtaining the eigendecomposition of the $n \times n$ kernel matrix is $O(n^3)$ operations. On the other hand, the eigendecompositions of the M $r \times r$ submatrices requires $O(Mr^3)$ operations. To obtain computational gains from the sampled submatrices, we must guarantee that $Mr^3 < n^3$, that is, $M < (r/n)^{-3}$. The quantity $\gamma = r/n$ is the proportion of points considered in the submatrices. Since we would like to take full advantage of both the dimension reduction and speed, we should work with values of $M \ll \gamma^{-3}$. In our experiments, we set $M = \lfloor \gamma^{-3}/2 \rfloor$, where $\lfloor x \rfloor$ stands for the floor function.

3.4. Experiments with large datasets

In this section, we apply both approaches presented in Section 3.3 to moderately large datasets in order to compare their results. The comparison and evaluation of the results are based on simulations. Because the data size depends on the nature of the problem, there is no clear definition of what is considered a "large dataset". Here, we consider two size values as large sizes: $n \in \{1000, 10000\}$. These values were chosen so as to obtain results that can be applied to moderetaly large real datasets, and to be able to explore the effect on the results of using sparse kernel matrices instead of the original kernel matrices for determinantal consensus clustering. Moreover, the chosen data sizes keep the computation time within reasonable elapsed times for our computer resources. Our choices for data sizes do not necessarily constitute what is known as *big data*, a term popularized by Mashey (1999) to describe datasets with sizes that go beyond the ability of commonly used software tools to capture, curate, manage, and process data within a tolerable elapsed time (Snijders et al., 2012). As Bonner et al. (2017) emphasize, the processing of large datasets does not have to involve big data.

3.4.1. Large datasets with $n = 1000$ observations

Data generation. Following Vicente and Murua (2020), we created nine experimental conditions or scenarios to generate datasets with $n = 1000$ observations. Each dataset was generated with the algorithm of Melnykov et al. (2012), which draws datasets from p-variate finite Gaussian mixtures. These have the form $\sum_{k=1}^{K} \pi_k \phi_p(\cdot; \mu_k, V_k)$, where K is the number of Gaussian components, ϕ_p denotes the p-variate Gaussian density, $\{\mu_1, \ldots, \mu_K\}$, are the component means, and $\{V_1, \ldots, V_K\}$ are the covariance matrices of the components. The means constitute K independent realizations of a uniform p-variate distribution on the

p-dimensional unit hypercube; the covariance matrices are K independent realizations of a p-variate standard Wishart distribution with $p+1$ degrees of freedom; the mixing proportions π_k are drawn from a Dirichlet distribution, so that $\sum_{k=1}^{K} \pi_k = 1$. The number of data points per component is a draw from the multinomial distribution based on the mixing proportions.

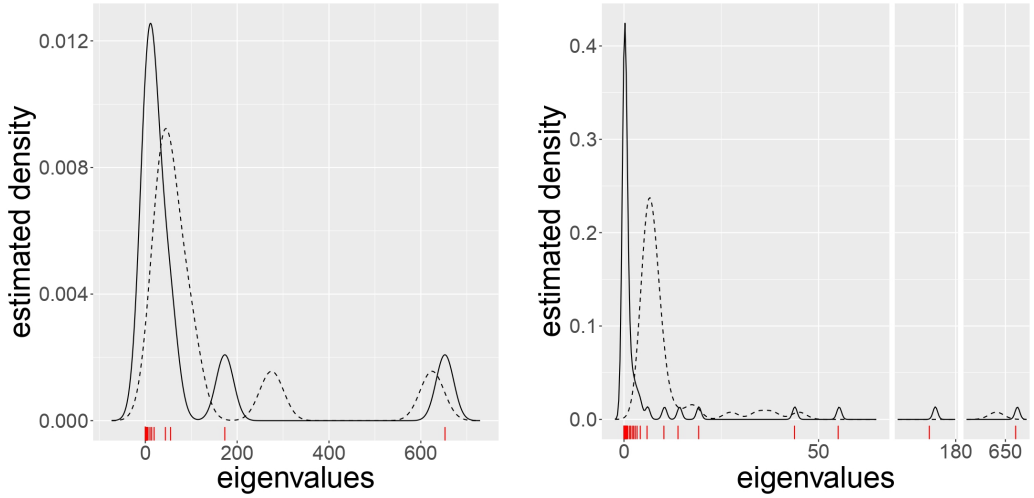
Melnykov et al. (2012) introduce the concept of *pairwise overlap* to generate datasets with the algorithm. It represents the degree of interaction between components and defines the clustering complexity of datasets. Melnykov et al. (2012) define a range of 0.001 to 0.4 for an average pairwise overlap between components, representing a very low and a very high overlap degree, respectively. We follow the values suggested by the authors, and choose, for every dataset generated, a random average pairwise overlap between 0.001 and 0.01. This range of values keep the overlap degree at a low level so that performing clustering on the data makes sense. All datasets were generated from mixtures with ellipsoidal covariance matrices; the simulated components do not necessarily contain the same number of elements.

To obtain the nine simulated datasets with $n = 1000$ observations, we consider $p \in \{5, 10, 18\}$ variables, and $K \in \{4, 8, 15\}$ components. These values correspond to three different levels (low, medium, large) for p and K. We also ensured that no cluster with size less than $\sqrt{n}$ is present among each simulated dataset, because otherwise, following the procedure described in Section 3.2.1, small clusters will be inevitably merged with larger clusters. We applied the two approaches presented in Section 3.3 to each simulated dataset.

Results with approach based on NNGP. In order to study the effect of sparseness in the approximation presented in Section 3.3.1, we set different values for m, the maximum number of nonzero elements in each row of the matrix A. Each value ensures four levels of sparseness (or total percentage of zeros) for the matrix $\widetilde{L}^{-1}$: $20\%, 40\%, 60\%$ and 80%. The first largest t eigenvalues and corresponding eigenvectors of $\widetilde{L}^{-1}$ were extracted with the Lanczos algorithm, for $t \in \{10, 25, 50\}$. The consensus DPP of Section 3.2 was applied to obtain 200 partitions, as recommended in (Vicente and Murua, 2020). The whole procedure was repeated ten times. Considering the nine simulated scenarios, the experiment consists of nine plots with ten repeated measures from a 4×3 factorial design given by the combination of sparseness and number of extracted eigenvalues. For each single plot, we note that all 120 observations are dependent, since the same data were used for all combinations.

Figure 3.1 shows density estimates of the eigenvalues distribution obtained with the NNGP approximation matrix $\widetilde{L}$ (solid line). We considered two combinations of sparseness levels and number of eigenvalues: $(20\%, t = 10)$ and $(60\%, t = 50)$, respectively. The figure presents plots from one dataset, since they depict typical plots obtained with all datasets. The tick-marks in the horizontal axis locate all eigenvalues of the original kernel matrix L. We can see that the density estimations obtained from $\widetilde{L}$ concentrate correctly around the true eigenvalues of the kernel matrix L. For comparison purposes, we also display

in Figure 3.1 the density estimates of the set of eigenvalues from the m_{nn}-nearest-neighbor matrix $\check{L}$ (dashed line) described in Section 3.3.1. Unlike the NNGP approach, the eigenvalue density estimations associated with $\check{L}$ do not quite concentrate on the set of true eigenvalues. The plots corroborate the arguments of Section 3.3.1 concerning our preference for $\widetilde{L}$ instead of $\check{L}$.



(a) Sparseness=20% ; t=10 eigenvalues **(b)** Sparseness=60% ; t=50 eigenvalues

Figure 3.1 – Kernel density estimates of the eigenvalue distribution from the NNGP approximation $\widetilde{L}$ (solid line), and the m_{nn}-nearest-neighbor matrix $\check{L}$ (dashed line). The plots are associated with two sparseness-eigenvalue conditions among the twelve experimental conditions for a given dataset. The tick-marks indicate the eigenvalues of L.

Following the discussion of Section 3.3, we also computed the Frobenius distances between the sparse approximation matrices, $\widetilde{L}$ and $\check{L}$, and the original dense matrix L. Table 3.1 shows the mean and standard deviation of Frobenius distances considering all nine data scenarios, as a function of the level of sparseness. The mean distance $\|L - \widetilde{L}\|_F$ is always much inferior to the distance $\|L - \check{L}\|_F$. At first, the distances $\|L - \widetilde{L}\|_F$ decrease with sparseness until a level of at least 60% sparseness is reached. However, the distances of L to $\check{L}$ monotonically increase with sparseness. The approximation based on NNGP is then the preferred choice for extracting eigenvalues. These results corroborate the nice concentration of the eigenvalue distributions of the NNGP approach about the eigenvalues of L in Figure 3.1.

Approximation matrix	Sparseness level			
	20%	40%	60%	80%
$\widetilde{L}$	11.31 (16.49)	6.80 (4.55)	8.04 (5.70)	66.58 (77.14)
$\check{L}$	200.17 (17.41)	315.89 (16.27)	421.82 (8.66)	527.88 (8.18)

Table 3.1 – Mean and standard deviation (within parentheses) of Frobenius distances.

To have an objective measure of the resemblance between the two sets of eigenvalues from $\widetilde{L}$ and L, we computed the symmetrized Kulback-Leibler (KL) divergence Kullback and Leibler (1951) between corresponding density estimates of the two eigenvalue distributions. The densities were estimated using a Gaussian kernel density estimator (Silverman, 1986). Table 3.2 reports the mean and standard deviation of the KL divergences over all nine data scenarios for each combination of sparseness and number of eigenvalues extracted. There does not appear to be any significant difference between all the cases, except for $t = 10$ eigenvalues. Better results are obtained when a larger but still very moderate number of eigenvalues is estimated. The divergences reported in Table 3.2 are very small, showing that there is pretty good similarity between the eigenvalue distributions.

Number of	Sparseness level			
eigenvalues	20%	40%	60%	80%
10	0.01 (0.00)	0.36 (0.98)	0.36 (0.97)	0.35 (0.96)
25	0.01 (0.00)	0.01 (0.00)	0.01 (0.00)	0.01 (0.00)
50	0.01 (0.00)	0.01 (0.00)	0.01 (0.00)	0.01 (0.00)

Table 3.2 – Mean and standard deviation (within parentheses) of Kullback-Leibler divergences.

We also report and compare the elapsed time in seconds for calculating each set of eigenvalues. Table 3.3 displays the means and standard deviations of elapsed times over all nine data scenarios for each combination of sparseness and number of eigenvalues extracted. We can see that, as expected, the average computation time decreases with sparseness, and increases with the number of eigenvalues extracted. In fact, a linear regression (not shown here) of the elapsed time as a function of sparseness and number of eigenvalues extracted yields a coefficient of determination of 0.97, clearly indicating a linear growth of the computational time with both sparseness and the number of eigenvalues to be estimated. The same statistics computed from the original kernel matrix L, from which all eigenvalues must be extracted, yield a mean of 0.27 seconds with a standard deviation of 0.06. Extracting only a few eigenvalues with the Lanczos algorithm reduces significantly the computation time. The elapsed times were computed while running a Julia v1.1.1 script (Bezanson et al., 2017) on a PC with an Intel-Core i5-4460 CPU running at 3.20GHz with 16GB of RAM.

Number of	Sparseness level			
eigenvalues	20%	40%	60%	80%
10	0.10 (0.00)	0.07 (0.00)	0.06 (0.01)	0.04 (0.01)
25	0.12 (0.01)	0.09 (0.00)	0.08 (0.01)	0.05 (0.01)
50	0.14 (0.01)	0.12 (0.01)	0.11 (0.01)	0.07 (0.01)

Table 3.3 – Means and standard deviations (within parentheses) of elapsed times in seconds for eigenvalues calculation.

To obtain the optimal clustering configuration for each dataset, we used the determinantal consensus clustering described in Section 3.2, meeting the minimal cluster size criterion of $\sqrt{n}$ and using the kernel-based validation index of Fan et al. (2010). The quality of the chosen optimal clustering configuration has been assessed with the adjusted Rand index or ARI (Rand, 1971; Hubert and Arabie, 1985). Among the many known measures of goodness-of-fit that can be found in the literature, the ARI is one of the most common criteria. The original Rand index counts the proportion of elements that are either in the same clusters in both clustering configurations or in different clusters in both configurations. The adjusted version of the Rand index corrected the calculus of the proportion, so that its expected value is zero when the clustering configurations are random. The larger the ARI, the more similar the two configurations are, with the maximum ARI score of 1.0 indicating a perfect match. Table 3.4 displays the ARI means and standard deviations over all nine scenarios and replicas for all twelve combinations of sparseness and number of eigenvalues extracted.

Sparseness level	$t = 10$	$t = 25$	$t = 50$
20%	0.94 (0.06)	0.95 (0.06)	0.95 (0.06)
40%	0.93 (0.05)	0.95 (0.05)	0.95 (0.06)
60%	0.94 (0.05)	0.95 (0.06)	0.95 (0.05)
80%	0.93 (0.06)	0.94 (0.05)	0.95 (0.05)

Table 3.4 – Global ARI means and standard deviations (within parentheses) yielded by consensus DPP with the sparse kernel matrices for the twelve combinations of sparseness and number of eigenvalues extracted. Each mean and standard deviation was computed from ninety datasets.

For comparison purposes, we computed the ARI statistics obtained by applying consensus DPP with the original dense kernel matrix L, from which we extracted all the eigenvalues and eigenvectors. This yielded an ARI mean of 0.91, and a standard deviation of 0.08. We did the same with the consensus clustering methodology applied to partitions generated with the well-known Partitioning Around Medoids (PAM) algorithm (Kaufmann and Rousseeuw,

1987). The PAM algorithm is a classical partitioning technique for clustering. It chooses the data center points of the Voronoi cells by uniform random sampling. Because DPP selects center points based on diversity, our goal here is to show how much the quality of the clustering configurations is affected by the lack of diversity at the moment of sampling centroids. PAM yielded a much lower ARI mean of 0.86, and a standard deviation of 0.14.

We can see that the quality of the clustering results associated with the NNGP approach is better in terms of ARI than the ones yielded using the whole dense kernel matrix L. This conclusion holds regardless of the sparseness level and number of eigenvalues extracted. The results are also more stable, considering the standard deviations. Taking advantage of the elapsed time, choosing $t = 25$ eigenvalues is the better choice among the three levels for t studied here. This combined with a higher sparseness level (to speed up the computation of the matrix $\widetilde{L}^{-1}$) appears to be a winning combination. The results yielded by PAM are not as good. In conclusion, the NNGP approach provides a very good and efficient alternative to the use of the complete dense matrix L.

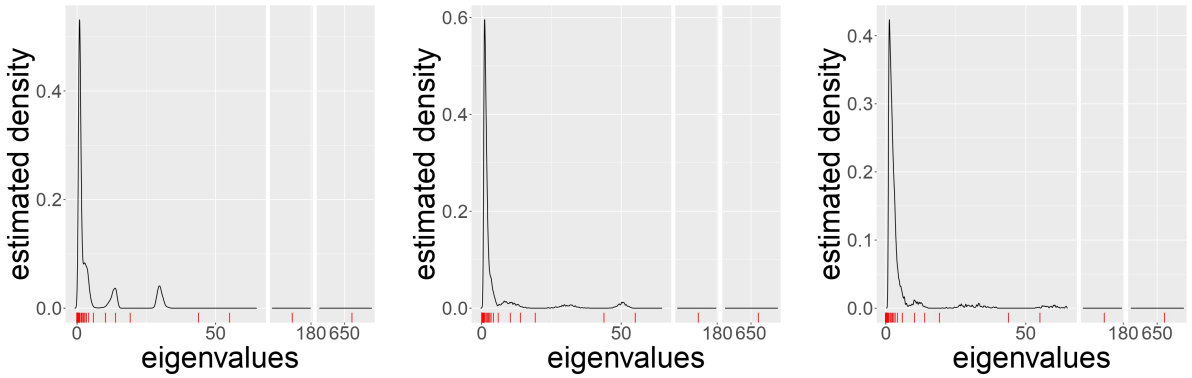
Results with approach based on small submatrices from L. Following the notation presented in Section 3.3.2, we studied the effect of three parameters on the clustering results. These are the proportion γ of points chosen (or, equivalently, the size r of any submatrix $L^{(i_k)},\ k = 1, 2, \ldots, N$), the sparsennes level of any submatrix $L^{(i_k)}$ (along with the number k of nearest neighbors needed to achieve such level), and the number $t \in \{10, 25, 50\}$ of the largest eigenvalues to be extracted. Table 3.5 shows the choices for γ and sparseness levels with the corresponding values of r and k. Note that each value of k ensures the same sparseness for all nine data scenarios.

		k			
γ	r	20%	40%	60%	80%
0.05	50	34	25	17	8
0.1	100	68	50	34	16
0.2	200	136	100	68	32

Table 3.5 – Number of nearest neighbors k associated with choices for proportion of data γ and sparseness levels 20%, 40%, 60% and 80%.

As we did with the NNGP approach, we set the number of partitions to obtain a consensus clustering to 200. The whole procedure was repeated ten times. The analysis of the results mimics the one made in the previous section with the NNGP approach. Figure 3.2 shows density estimates of the eigenvalue distribution associated with three combinations of (γ, sparseness, eigenvalues) $\in \{(0.05, 20\%, 10),\ (0.1, 40\%, 25),\ (0.2, 60\%, 50)\}$. The plots are associated with a given dataset; it depicts typical patterns oberved in all datasets. The tick-marks in the horizontal axis locate all eigenvalues of the original kernel matrix L. We

can see that the density estimations concentrate well around the smaller eigenvalues of L, but fail to capture the largest eigenvalues.



(a) $\gamma = 0.05$; Sparseness=20% ; t=10 eigenvalues

(b) $\gamma = 0.1$; Sparseness=40% ; t=25 eigenvalues

(c) $\gamma = 0.2$; Sparseness=60% ; t=50 eigenvalues

Figure 3.2 – Density estimators of three sets of eigenvalues extracted from the random small matrices approach associated with three scenarios of Table 3.5 on a given dataset. The tick-marks are placed on the eigenvalues of L.

Table 3.6 shows the Kullback-Leibler symmetrized divergences between the distribution of the eigenvalues extracted from the random small matrices and the distribution of the eigenvalues of L, for the (γ, sparseness) combinations displayed in Table 3.5. The eigenvalues distributions were estimated with kernel density estimators. We can see that, globally, increasing the proportion γ of points sampled reduces the KL divergence when combined with moderate to low levels of sparseness and a higher number of eigenvalues. Overall, the KL divergences are larger than those obtained with the NNGP approach. But, again, we stress that the objective of the approaches is not to estimate the eigenvalues of L, but to offer efficient DPP sample alternatives to $\text{DPP}_{\mathcal{S}}(L)$.

γ	Number of eigenvalues	Sparseness level			
		20%	40%	60%	80%
0.05	10	2.09 (1.28)	1.28 (0.37)	1.00 (0.08)	1.01 (0.03)
	25	0.02 (0.01)	0.58 (1.25)	1.64 (0.46)	1.01 (0.04)
	50	0.02 (0.01)	0.03 (0.23)	1.71 (0.88)	1.10 (0.17)
0.1	10	1.95 (1.39)	1.34 (0.43)	1.02 (0.10)	1.00 (0.03)
	25	0.01 (0.00)	0.12 (0.58)	1.87 (0.89)	1.04 (0.10)
	50	0.02 (0.00)	0.01 (0.00)	0.15 (0.68)	1.32 (0.26)
0.2	10	2.07 (1.43)	1.36 (0.44)	1.03 (0.12)	1.00 (0.03)
	25	0.01 (0.00)	0.16 (0.69)	1.76 (1.08)	1.12 (0.23)
	50	0.01 (0.00)	0.01 (0.00)	0.01 (0.00)	1.64 (0.51)

Table 3.6 – Means and standard deviations (within parentheses) of Kullback-Leibler divergences associated with the random small matrices approach.

Recalling the notation of Section 3.3.2, Table 3.7 shows the means and standard deviations of the elapsed times in seconds for eigenvalues computation, using the sampled submatrix $L^{(i_k)}$ or its sparse approximation $\widehat{L}^{(i_k)}$. The results were obtained while running a Julia script on a PC with an Intel Core i5-4460 CPU running at 3.20GHz with 16GB of RAM. Note that contrary to NNGP, the level of sparseness does not affect the elapsed times for computing the eigenvalues with the Lanczos method. This fact is probably due to the small size of the matrices $\widehat{L}^{(i_k)}$.

γ	Submatrix	Number of eigenvalues		
		10	25	50
0.05	$L^{(i_k)}$	0.001 (0.000)	0.001 (0.000)	0.001 (0.000)
	$\widehat{L}^{(i_k)}$	0.001 (0.000)	0.001 (0.000)	0.001 (0.000)
0.1	$L^{(i_k)}$	0.004 (0.003)	0.004 (0.003)	0.004 (0.003)
	$\widehat{L}^{(i_k)}$	0.003 (0.001)	0.003 (0.001)	0.003 (0.001)
0.2	$L^{(i_k)}$	0.01 (0.004)	0.01 (0.004)	0.01 (0.004)
	$\widehat{L}^{(i_k)}$	0.006 (0.002)	0.006 (0.002)	0.006 (0.002)

Table 3.7 – Mean and standard deviation (within parentheses) of elapsed times in seconds to eigenvalues calculation, for $L^{(i_k)}$ and its sparse approximation $\widehat{L}^{(i_k)}$.

Table 3.8 displays the ARI means and standard deviations over all (γ, sparseness) combinations in Table 3.5. For each combination, these statistics were computed from a sample of ninety ARI scores given by the ten replica datasets from the nine data scenarios. Table 3.9 displays the same statistics obtained by keeping the dense version of the submatrices $L^{(i_k)}$,

and extracting all its eigenvalues, instead of using their sparse approximations. This comparison is appropriate to study the effect of making these small matrices sparse. These results should be compared to those of (i) consensus DPP applied to the original dense kernel matrix L, from which all eigenvalues are extracted, and (ii) the consensus clustering methodology applied to partitions generated by PAM. These latter results were already mentioned above when showing the results of the NNGP approach. They are, respectively, (i) a mean ARI and standard deviation of 0.91 and 0.08, and (ii) a mean ARI and standard deviation of 0.86 and 0.14.

γ	Sparseness level	$t = 10$	$t = 25$	$t = 50$
0.05	20%	0.92 (0.06)	0.94 (0.05)	0.96 (0.06)
	40%	0.93 (0.06)	0.95 (0.05)	0.95 (0.05)
	60%	0.94 (0.05)	0.95 (0.05)	0.95 (0.06)
	80%	0.93 (0.06)	0.95 (0.06)	0.95 (0.06)
0.1	20%	0.93 (0.05)	0.93 (0.06)	0.94 (0.07)
	40%	0.95 (0.06)	0.95 (0.05)	0.95 (0.06)
	60%	0.95 (0.05)	0.95 (0.05)	0.95 (0.06)
	80%	0.94 (0.06)	0.94 (0.07)	0.95 (0.06)
0.2	20%	0.93 (0.05)	0.94 (0.06)	0.93 (0.07)
	40%	0.94 (0.05)	0.95 (0.05)	0.94 (0.06)
	60%	0.94 (0.05)	0.94 (0.05)	0.92 (0.06)
	80%	0.94 (0.06)	0.95 (0.06)	0.91 (0.08)

Table 3.8 – Global ARI means and standard deviations (within parentheses) associated with consensus DPP on the sparse random small submatrices, for each (γ, sparseness) combinations in Table 3.5, and over the corresponding ninety datasets for each combination.

$\gamma = 0.05$	$\gamma = 0.1$	$\gamma = 0.2$
0.94 (0.05)	0.95 (0.05)	0.95 (0.06)

Table 3.9 – Global ARI means and standard deviations (within parentheses) associated with consensus DPP on the dense version of the random small submatrices considering all datasets.

The quality of the clustering results is much better in terms of ARI, if we use either the dense $L^{(i_k)}$ or sparse $\widehat{L}^{(i_k)}$ random small submatrices rather than the whole dense kernel matrix L. The sparse approximations $\widehat{L}^{(i_k)}$ require more computation since the nearest neighbors must be computed. This extra cost does not seem worth when comparing the results associated with the dense approximations $L^{(i_k)}$. However, there is a slight improvement in the results when $\gamma = 0.05$. In this case, combining any sparseness level with a

higher number of eigenvalues is generally a good combination, since computational times for eigenvalue extraction do not differ significantly. For other levels of γ, the gain is not worth the complication of making the submatrices sparse.

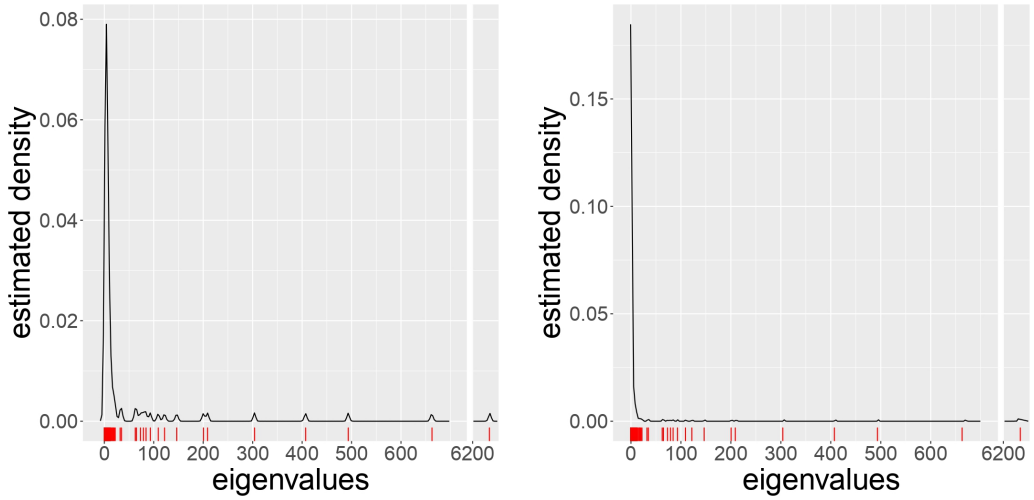
Again, as with the NNGP approach, the random small submatrices approach outperforms PAM and shows less variability in the quality of the results. Furthermore, this approach achieves comparable quality results to those of the NNGP approach. From a computational point of view, the NNGP approach is more expensive because it requires dealing with larger matrices. Therefore, the random small submatrices are a good and very efficient alternative to the use of the complete dense matrix L, and to the use of the NNGP approach.

3.4.2. Large dataset with $n = 10000$ observations

Data generation. For the second experiment, due to hardware limitations, we simulated only two datasets with $n = 10000$ observations. As in the previous case, these were generated with the algorithm of Melnykov et al. (2012). The first one has $p = 15$ variables and $K = 10$ components, while the second one has $p = 10$ variables and $K = 5$ components. Throughout this section, these data will be referred to as dataset I and dataset II, respectively. Both datasets have a maximum pairwise overlap of 0.01. We ensured that no cluster with size less than $\sqrt{n}$ is present among the simulated datasets, as it will be inevitably merged with a larger cluster, according to the procedure described in Section 3.2.1. We applied the two approaches presented in Section 3.3 to the simulated datasets.

Results with approach based on NNGP. We studied the same four levels of sparseness for the matrix $\widetilde{L}^{-1}$ $(20\%, 40\%, 60\%, 80\%)$, setting the appropriate values for the maximum number m of nonzero elements in each row of the matrix A (Section 3.3.1). The first largest $t \in \{100, 250, 500\}$ eigenvalues and corresponding eigenvectors of $\widetilde{L}^{-1}$ are extracted with the Lanczos algorithm. The determinantal consensus clustering of Section 3.2 was then applied so as to obtain 200 partitions of the data. The whole procedure was repeated five times. This time, the experiment consists of two plots of five repeated measures each, from a 4×3 factorial design given by the combination of four levels of sparseness and three levels for the number of extracted eigenvalues. All 60 observations of each plot are dependent, since the same datasets were used for all scenarios. The analysis of the results follows the same steps as the NNGP approach applied to the smaller datasets.

Figure 3.3 shows density estimates of the eigenvalue distribution of $\widetilde{L}$ for dataset I, for two combinations of sparseness and number of eigenvalues: $(20\%, t = 100)$ and $(60\%, t = 500)$, respectively. We can see that the density estimations concentrate correctly around the true eigenvalues of the kernel matrix L.



(a) Sparseness: 20% ; t=100 eigenvalues **(b)** Sparseness: 60% ; t=500 eigenvalues

Figure 3.3 – Kernel density estimates of the eigenvalue distribution associated with $\widetilde{L}$ for dataset I, for two sparseness-eigenvalue conditions among the twelve experimental conditions. The tick-marks indicate the eigenvalues of L.

For comparison purposes between the two sparse approximations $\widetilde{L}$ and $\check{L}$, we computed the Frobenius distance between these matrices and the kernel matrix L. Table 3.10 displays the results for both datasets. As seen in the experiment with the smaller datasets, the distance of L to $\widetilde{L}$ is always much smaller than the distance of L to $\check{L}$. For both datasets, the distances always increase with sparseness. The approximation based on NNGP is a better choice for extracting eigenvalues. This also explains the correct concentration of the density estimation of the eigenvalue distribution of $\widetilde{L}$ around the eigenvalues of L, as seen in Figure 3.3.

Approximation matrix	Dataset	Sparseness level			
		20%	40%	60%	80%
$\widetilde{L}$	I	2.75	8.82	15.65	96.71
	II	0.33	4.47	14.07	191.45
$\check{L}$	I	2122.11	3269.61	4265.59	5224.48
	II	1955.09	3079.10	4128.15	5228.46

Table 3.10 – Frobenius distances for both datasets (I and II).

Table 3.11 reports the symmetrized Kullback-Leibler divergence of both datasets between the density estimates of the two eigenvalue distributions from $\widetilde{L}$ and L. The densities were estimated with kernel density estimators. Again, as observed with the smaller datasets, the

very small divergence values indicate a good resemblance between the eigenvalue distributions.

Number of eigenvalues	Dataset	Sparseness level			
		20%	40%	60%	80%
100	I	0.0000338	0.0000337	0.0000337	0.0000336
	II	0.0000329	0.0000132	0.0000015	0.0000329
250	I	0.0000365	0.0000359	0.0000359	0.0000358
	II	0.0000133	0.0000015	0.0000329	0.0000139
500	I	0.0000090	0.0000090	0.0000090	0.0000090
	II	0.0000016	0.0000322	0.0000166	0.0000017

Table 3.11 – Kullback-Leibler divergences for both datasets (I and II).

Table 3.12 shows the comparison of the elapsed times in seconds for calculating each set of eigenvalues for dataset I. Extracting all eigenvalues from the original matrix L yields an elapsed time 170.08 seconds and 156.74 seconds for datasets I and II, respectively. The results were obtained on Julia v1.1.1 running on a PC with an Intel-Core i5-4460 CPU running at 3.20GHz with 16GB of RAM. The elapsed times for dataset I are well explained by a linear regression on sparseness and number of eigenvalues extracted, presenting a coefficient of determination of 0.998. Similar results were obtained for dataset II.

Number of eigenvalues	Dataset	Sparseness level			
		20%	40%	60%	80%
100	I	37.34	34.46	31.79	27.39
250	I	68.03	65.86	60.62	55.37
500	I	125.48	120.27	115.55	112.85

Table 3.12 – Elapsed times in seconds for eigenvalues calculation of dataset I.

Sparseness level	$t = 100$		$t = 250$		$t = 500$	
	I	II	I	II	I	II
20%	0.97 (0.04)	0.79 (0.04)	0.93 (0.02)	0.87 (0.05)	0.70 (0.04)	0.88 (0.07)
40%	0.98 (0.00)	0.87 (0.03)	0.94 (0.02)	0.87 (0.08)	0.77 (0.03)	0.89 (0.04)
60%	0.97 (0.04)	0.86 (0.09)	0.94 (0.01)	0.87 (0.07)	0.83 (0.02)	0.82 (0.07)
80%	0.98 (0.02)	0.86 (0.09)	0.97 (0.02)	0.86 (0.04)	0.71 (0.10)	0.87 (0.03)

Table 3.13 – ARI means and standard deviations (within parentheses) obtained by consensus DPP on the sparse kernel matrices over the twelve experimental conditions and both datasets (I and II).

Table 3.13 displays the ARI means and standard deviations over all twelve experimental conditions for both datasets (I and II). For comparison purposes, we computed the same statistics for (i) consensus DPP applied to the original dense kernel matrix L, from which all eigenvalues were extracted, and for (ii) the consensus clustering methodology applied to partitions generated by PAM. For dataset I, the corresponding ARI means and standard deviations are 0.57 and 0.02 for consensus DPP, and 0.93 and 0.01 for PAM. For dataset II, we obtain ARI means and standard deviations of 0.82 and 0.06 for consensus DPP, and of 0.83 and 0.03 for PAM.

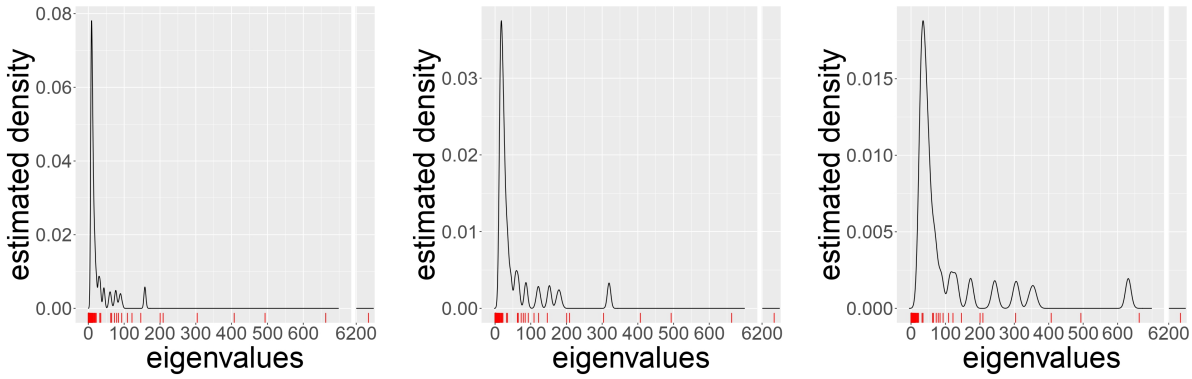
Observe that the quality of the clustering results is better if we consider the sparse approximation $\widetilde{L}^{-1}$ and a lower number of eigenvalues, $t \in \{100, 250\}$. Considering a higher number of eigenvalues is not such a good option. The quality of the results decreases with the number of eigenvalues extracted for each sparseness level. The use of the original dense matrix L with all eigenvalues is not a good alternative either. The most probably reason for the observed results is the large size of the kernel matrix. Most eigenvalues cannot be computed reliably, numerically speaking; hence, using all numerically extracted eigenvalues might introduce noise, leading the algorithm to perform poorly. Also, if most of the eigenvalues are small, they will be estimated with a lot of error, specially if the difference between the largest eigenvalues and the smallest ones is orders of magnitude. This is known as ill-conditioning of the kernel matrix. We would like to note that for all three cases, NNGP, the original dense matrix and PAM, we used the $\sqrt{n}$ criterion to merge small clusters during consensus (see Section 3.2.1). However, as pointed out in Vicente and Murua (2020), a criterion close to $n^{2/3}$ might have been more appropriate given the small number of clusters K, and the large number of observations n. In fact, using this larger size of small cluster in the merging step of the consensus give slightly better results for all methods, specially for dataset II.

To summarize, determinantal consensus clustering with the NNGP approach outperforms PAM for all cases. In addition, if we would like to favor high quality clustering results with low computational cost, combining $t = 100$ eigenvalues with a high level of sparseness appears to be the best option.

Results with approach based on small submatrices from L. For this approach, due to hardware limitations, we decided to reduce the number of evaluated scenarios in both datasets. Thus, maintaining the proportion $\gamma \in \{0.05, 0.1, 0.2\}$ of points selected from the dense matrix L, the sampled submatrices $L^{(i_k)}$, $k = 1, \ldots, N$, are of size $r \in \{500, 1000, 2000\}$, respectively. These sizes are considerably larger than those used with the smaller datasets. Hence, sparse approximations of $L^{(i_k)}$ are used with a unique high level of sparseness equal to 80%. This sparseness is achieved with 80, 160 and 320 nearest neighbors, respectively. Recall that with the NNGP approach applied to the smaller datasets, a low mean elapsed

time for eigenvalue calculation was achieved when combining a low number of eigenvalues with a high level of sparseness (see Table 3.3). Therefore, to speed up the computation time, we extract only the largest 100 eigenvalues for all the cases. The following analysis of the results is organized as with the above simulations.

Figure 3.4 shows the kernel density estimates for the aforementioned situations. The tick-marks in the horizontal axis locate all eigenvalues of the original kernel matrix L. As seen in the results with the smaller datasets, the density estimates of this approach do not capture well the largest eigenvalues when the sparseness is too high (for a low value of γ).



(a) $\gamma = 0.05$ **(b)** $\gamma = 0.1$ **(c)** $\gamma = 0.2$

Figure 3.4 – Kernel density estimates of the set of eigenvalues extracted from sparse $L^{(i_k)}$. The tick-marks are placed on all eigenvalues of L .

Table 3.14 reports the symmetrized Kullback-Leibler divergences between the distribution of the eigenvalues extracted from the sparse submatrices $L^{(i_k)}$ and the distribution of the eigenvalues of L, for $\gamma \in \{0.05, 0.1, 0.2\}$. The densities were estimated with kernel density estimators. The KL divergences decrease with γ and are very small, indicating a good resemblance between the two distributions.

Dataset	$\gamma = 0.05$	$\gamma = 0.1$	$\gamma = 0.2$
I	0.000875	0.000301	0.000095
II	0.000579	0.000173	0.000061

Table 3.14 – Kullback-Leibler divergences for both datasets (I and II).

Table 3.15 shows the comparison of the elapsed time in seconds for eigenvalue computation, using the sampled submatrix $L^{(i_k)}$ or its sparse approximation $\widehat{L}^{(i_k)}$. The results were obtained with Julia Language, version 1.1.1, on a Desktop PC with Intel Core i5-4460 CPU @ 3.20GHz Processor and 16 GB DDR3 RAM. In this case, and due to the relatively large size of the matrices $L^{(i_k)}$, it does make a difference to use the sparse matrices $\widehat{L}^{(i_k)}$ instead of the dense ones.

Dataset	Submatrix	$\gamma = 0.05$	$\gamma = 0.1$	$\gamma = 0.2$
I	$L^{(i_k)}$	0.14	0.22	1.02
	$\widehat{L}^{(i_k)}$	0.03	0.08	0.30
II	$L^{(i_k)}$	0.17	0.38	1.66
	$\widehat{L}^{(i_k)}$	0.09	0.35	1.33

Table 3.15 – Elapsed times in seconds for eigenvalues calculation of datasets I and II.

Table 3.16 displays the ARI means and their standard deviations for $\gamma \in \{0.05, 0.1, 0.2\}$ and both datasets (I and II). It also reports the results obtained with consensus DPP applied to the original dense kernel matrix L, from which all eigenvalues were extracted, and PAM. These latter results were already reported in Section 3.4.2.

Dataset	$\gamma = 0.05$	$\gamma = 0.1$	$\gamma = 0.2$	Original L	PAM
I	0.95 (0.00)	0.96 (0.01)	0.96 (0.01)	0.57 (0.02)	0.93 (0.01)
II	0.88 (0.02)	0.87 (0.04)	0.87 (0.03)	0.82 (0.06)	0.83 (0.03)

Table 3.16 – ARI means and standard deviations (within parentheses) obtained from consensus DPP with approach based on small submatrices from L for datasets I and II. The results for the original dense matrix and PAM are also displayed.

As we did with the smaller datsets, we show in Table 3.17, the same statistics obtained by keeping the dense version of the submatrices $L^{(i_k)}$, and extracting all its eigenvalues, instead of using their sparse approximations.

Dataset	$\gamma = 0.05$	$\gamma = 0.1$	$\gamma = 0.2$
I	0.94 (0.05)	0.95 (0.05)	0.95 (0.06)
II	0.88 (0.04)	0.81 (0.06)	0.83 (0.05)

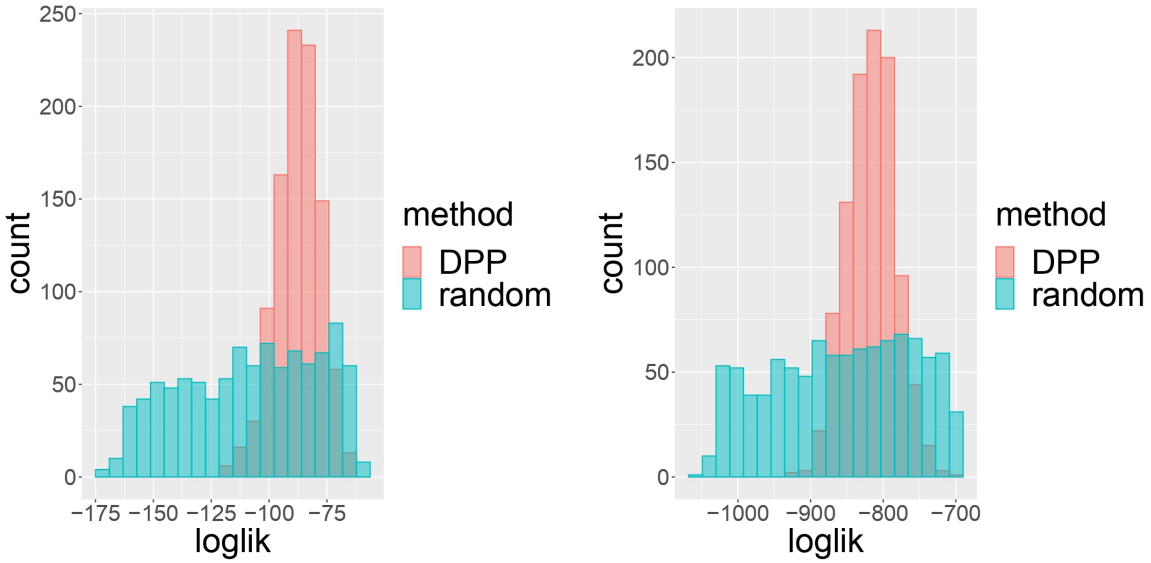
Table 3.17 – Global ARI means and standard deviations (within parentheses) of consensus DPP on the dense version of the random small submatrices for datasets I and II.

The random small matrices approach yields excellent results for any proportion γ. If we consider the results with the dense version of the submatrices, we can see that the results obtained are similar or better. As already observed with the small samples, the use of a sparse approximation $\widehat{L}^{(i_k)}$ does not hurt the quality of the results. The main advantage of using the sparse approximations is the reduced amount of time needed to compute the eigenvalues. Another advantage is the resulting stability of the clustering quality results (low dispersion of ARI values). In summary, if one would like low computational time and low dispersion, using the sparse approximation of submatrices with a low proportion of sampled points (5%) is a good option.

To end this section, we observe that the approach based on random sampling of submatrices outperforms PAM, for all γ values in both datasets. It also reaches comparable quality clustering levels to the approach based on NNGP with $t = 100$ and a sparseness of 80%. Sampling submatrices of lower dimension is then a good alternative to the use of the NNGP approach.

3.4.3. DPP as a measure of diversity

The differences between DPP and PAM can also be highlighted using the logarithm of the probability mass function of the DPP, given by (21). Figure 3.5 displays histograms of these probability logarithms. The histograms are based on 1000 random subsets of datapoints drawn (i) using the DPP sampling algorithm of Ben Hough et al. (2006) and Kulesza and Taskar (2012), and (ii) using the simple random sampling of PAM. The subsets were drawn from two simulated large datasets: one of size $n = 1000$, and another of size $n = 10000$ observations.



(a) Dataset with $n = 1000$ observations

(b) Dataset with $n = 10000$ observations

Figure 3.5 – Histograms of the logarithm of the probability mass function (loglik), using DPP and simple random sampling, for two simulated large datasets.

The histograms clearly show that DPP selects random subsets with higher and less dispersed probability mass values (likelihood) than simple random sampling. This explains the low dispersion of the ARI observed in most results of this section, when sampling is performed with DPP. The higher likelihood achieved by DPP also implies a higher diversity of the sampled subsets. On its turn, subsets sampled as in PAM yield a highly dispersed likelihood, resulting in highly or poorly diverse subsets. DPP tends to be more consistent

and stable since it ensures a high level of diversity among the selected elements at each sampling.

3.5. Application to real data

In this section we evaluate the performance of consensus DPP versus PAM on three large real datasets:

1.- A dataset about human activity recognition and postural transitions using smartphones, collected from 30 subjects who performed six basic postures (downstairs, upstairs, walking, laying, sitting and standing), and six transitional postures between static postures (stand-to-sit, sit-to-stand, sit-to-lie, lie-to-sit, stand-to-lie and lie-to-stand). The experiment was realized in the same environment and conditions, while carrying a waist-mounted smartphone with embedded inertial sensors. The dataset consists of 10929 observations, with 561 time and frequency extracted features, which are commonly used in the field of human activity recognition. The dataset is available on the *UCI Machine Learning Repository* (Dua and Graff, 2017), a well known database in the machine learning community for clustering and classification problems. The six transitional postures between static postures comprises a relatively small subset of observations. Therefore, we apply our clustering algorithm only to the six basic postures. Using the notation of the simulated datasets of Section 3.4, the final dataset has $n = 10411$ observations, $p = 561$ variables and $K = 6$ components.

2.- The Modified National Institute of Standards and Technology (MNIST) dataset (LeCun et al., 2010), one of the most common datasets used for image classification. This dataset contains 60000 training images and 10000 testing images of handwritten digits, obtained from American Census Bureau employees and American high school students. Each 784-dimensional observation represents a 28×28 pixel gray-scale image depicting a handwritten version of one of the ten possible digits (0 to 9). As it is a common practice with the MNIST dataset, due to its intrinsic characteristics, we transformed the data to its multiplicative inverse as hinted by the Box-Cox transformation. We only use the testing set of 10000 images, so that the final dataset has $n = 10000$ observations, $p = 784$ variables and $K = 10$ components.

3.- The Fashion-MNIST dataset (Xiao et al., 2017), also one of the most common datasets used for image classification. This dataset contains 60000 training images and 10000 testing images of Zalando's articles [1]. Each observation represents a 28×28 pixel gray-scale images of clothes associated with a label from 10 classes. For the same reasons as MNIST, we transformed the data with an appropriate Box-Cox transformation, and only worked with

1. Zalando is a European e-commerce company specializing in fashion. They provided image data in repositories like Github.

the testing portion of the data. The dataset has $n = 10000$ observations, $p = 784$ variables and $K = 10$ components.

We applied both approaches of Section 3.3 to each dataset, and followed the same analysis procedure of Section 3.4. For the NNGP approach, we set the maximum number m of nonzero elements in each row of the matrix A to obtain a sparse approximation of the kernel matrix L with 80% of sparseness. The Lanczos algorithm was applied to extract the $t = 100$ largest eigenvalues from the sparse approximated matrix. For the approach based on random small submatrices from L, we sampled a proportion $\gamma = 0.05$ of points from the kernel matrix L, and obtained sparse approximations of the sampled submatrices with 80% of sparseness. The Lanczos algorithm was applied to extract the $t = 100$ largest eigenvalues from the sparse approximated submatrices. Determinantal consensus clustering was applied to 200 partitions. The whole procedure was repeated five times.

For comparison purposes, we also include the results from a couple of popular algorithms: consensus clusterings with PAM, and k-means. The PAM algorithm was already mentioned in Section 3.4. The k-means algorithm was introduced by Stuart Lloyd in 1957, with a supporting publication in Lloyd (1982). Given an initial set of k means, representing k clusters, it assigns each observation to the cluster with the nearest mean, and proceeds to recompute means from observations in the same cluster. The procedure is repeated until no changes are observed in the assignments. Among the many algorithms to initialize the k centers of k-means (Forgy, 1965; Pena et al., 1999; Gonzalez, 1985; Katsavounidis et al., 1994), we chose the k-means++ algorithm of Arthur and Vassilvitskii (2007). This method has become largely popular (Capó et al., 2017; Fränti and Sieranoja, 2019). It is a probability-based technique that avoids the usually poor results found by standard k-means initialization methods. We performed consensus clusterings with PAM and k-means using 200 partitions, also repeating the whole procedure five times. We report the mean and the standard deviation of the ARI achieved by the different methods in Table 3.18.

Dataset	NNGP	Small submatrices	PAM	k-means
Smartphones	0.59 (0.01)	0.58 (0.01)	0.43 (0.12)	0.49 (0.02)
MNIST	0.58 (0.05)	0.36 (0.02)	0.24 (0.01)	0.54 (0.02)
Fashion-MNIST	0.43 (0.01)	0.40 (0.01)	0.40 (0.01)	0.36 (0.03)

Table 3.18 – ARI means and standard deviations (within parentheses) associated with NNGP, random small submatrices, PAM and k-means.

Both DPP approaches, based on NNGP and small submatrices, attain good results. The NNGP approach outperformed all other three methods. Overall, the gains are very important

when compared with PAM and k-means. The method based on random small submatrices also ourperformed the other methods, except for the MNIST dataset.

3.6. Conclusions

Within the context of determinantal consensus clustering, we proposed two different approaches to overcome the computational burden induced by the eigendecomposition of the kernel matrix associated with large datasets. The first one is based on NNGP, and the second one, is based on random sampling of small submatrices from the kernel matrix.

The NNGP approach finds a sparse approximation of the kernel matrix, based on nearest-neighbors. The sparse matrix substitutes the original dense kernel matrix to extract the largest eigenvalues, so as to be able to perform determinantal consensus clustering on the large dataset. Simulations showed that using a high sparseness level and extracting a few largest eigenvalues are enough to ensure good clustering results. Extracting 1% to 3% of the eigenvalues seems to be a reasonable strategy. The amount of time needed to achieve the final clustering configuration is reduced considerably.

The approach based on random sampling of small submatrices from the kernel matrix is a good alternative to the NNGP approach. Its performance is comparable to that of the NNGP approach, even though the distribution of eigenvalues extracted with this method is not as close to the original eigenvalue distribution of the kernel matrix. Rather than keeping the original kernel matrix, this approach shows that considering the extraction of eigenvalues from several rather small submatrices of the kernel matrix is enough to obtain good results with consensus DPP. Our simulations hint at sampling a number of small matrices in the range of $0.05n$ to $0.1n$. If the submatrices are still too large, finding their sparse approximation with the k-nearest neighbors graph method is a good option, even with a small k. In fact, for very large datasets, our simulations showed that it is strongly recommended to find a sparse approximation of the small submatrices. In order to speed up computational time, choosing a high sparseness level is the best option.

The two approaches are able to reach better quality results than consensus clustering applied to PAM. Applications on large real datasets confirm the results found with the simulations. Determinantal consesus clustering also proved to be superior to k-means for most of the datasets. The presence of diversity in the sampled centroids helps to improve the quality of clustering and the stability of the results.

An issue we found with the NNGP approach, on some real datasets, is the possibility of ill-conditioning of some of the intermediate calculation submatrices $W_{[1:i-1]}$ (see equation (22)). This might occur because of strong similarity between some observations. As with the case of regression, the original kernel matrix should be evaluated for numerical conditioning before attempting to use the NNGP approach. A possible solution to avoid the problem is to

eliminate observations that are too similar, so as to only consider representative observations instead of all of them. Another possibility is to regularize the submatrices by adding a small positive constant to the main diagonal, as it is done in ridge regression. Alternatively, the approach based on random small matrices might be used instead, if the problem arises.

Bibliography

Arthur, D., Vassilvitskii, S., 2007. K-means++: The advantages of careful seeding, in: Proceedings of the Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms, Society for Industrial and Applied Mathematics, USA. pp. 1027–1035.

Aurenhammer, F., 1991. Voronoi diagrams—a survey of a fundamental geometric data structure. ACM Comput. Surv. 23, 345–405. doi:10.1145/116873.116880.

Ben Hough, J., Krishnapur, M., Peres, Y., Virág, B., 2006. Determinantal processes and independence. Probability Surveys [electronic only] 3, 206–229.

Bezanson, J., Edelman, A., Karpinski, S., Shah, V.B., 2017. Julia: A fresh approach to numerical computing. SIAM review 59, 65–98. URL: https://doi.org/10.1137/141000671.

Blatt, M., Wiseman, S., Domany, E., 1996. Superparamagnetic clustering of data. Phys. Rev. Lett. 76, 3251–3254.

Blatt, M., Wiseman, S., Domany, E., 1997. Data clustering using a model granular magnet. Neural Comput. 9, 1805–1842.

Bonner, S., Kureshi, I., Brennan, J., Theodoropoulos, G., 2017. Exploring the evolution of big data technologies, in: Software Architecture for Big Data and the Cloud. Elsevier, pp. 253–283.

Borodin, A., Olshanski, G., 2000. Distributions on Partitions, Point Processes, and the Hypergeometric Kernel. Communications in Mathematical Physics 211, 335–358. doi:10.1007/s002200050815, arXiv:math/9904010.

Borodin, A., Soshnikov, A., 2003. Janossy densities determinantal ensembles. Journal of Statistical Physics 113, 595–610. doi:10.1023/A:1026025003309.

Calvetti, D., Reichel, L., D C Sorensen, A., 1994. An implicitly restarted lanczos method for large symmetric eigenvalue problems. Electronic Trans. Numer. Anal. 2, 1–21.

Capó, M., Pérez, A., Lozano, J.A., 2017. An efficient approximation to the k-means clustering for massive data. Knowledge-Based Systems 117, 56–69.

Clark, L.T., Watkins, L., Piña, I.L., Elmer, M., Akinboboye, O., Gorham, M., Jamerson, B., McCullough, C., Pierre, C., Polis, A.B., et al., 2019. Increasing diversity in clinical trials: overcoming critical barriers. Current Problems in Cardiology 44, 148–172.

Cullum, J., 1978. The simultaneous computation of a few of the algebraically largest and smallest eigenvalues of a large, sparse, symmetric matrix. BIT Numerical Mathematics 18, 265–275. URL: https://doi.org/10.1007/BF01930896, doi:10.1007/BF01930896.

Daley, D., Vere Jones, D., 2003. An introduction to the theory of point processes. Vol. I: Elementary theory and methods. 2nd ed. volume Vol. 1. Springer-Verlag.

Datta, A., Banerjee, S., Finley, A.O., Gelfand, A.E., 2016. On nearest-neighbor gaussian process models for massive spatial data. Wiley Interdisciplinary Reviews: Computational Statistics 8, 162–171. URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/wics.1383, doi:10.1002/wics.1383.

Dua, D., Graff, C., 2017. UCI machine learning repository. URL: http://archive.ics.uci.edu/ml.

Fan, Z., Jiang, X., Xu, B., Jiang, Z., 2010. An automatic index validity for clustering, in: Tan, Y., Shi, Y., Tan, K.C. (Eds.), Advances in Swarm Intelligence, Springer Berlin Heidelberg, Berlin, Heidelberg. pp. 359–366.

Finley, A.O., Datta, A., Cook, B.D., Morton, D.C., Andersen, H.E., Banerjee, S., 2019. Efficient algorithms for bayesian nearest neighbor gaussian processes. Journal of Computational and Graphical Statistics 28, 401–414. URL: https://doi.org/10.1080/10618600.2018.1537924, doi:10.1080/10618600.2018.1537924, arXiv:https://doi.org/10.1080/10618600.2018.1537924.

Forgy, E.W., 1965. Cluster analysis of multivariate data: efficiency versus interpretability of classifications. biometrics 21, 768–769.

Fränti, P., Sieranoja, S., 2019. How much can k-means be improved by using better initialization and repeats? Pattern Recognition 93, 95–112.

Gartrell, M., Dohmatob, E., Alberdi, J., 2018. Deep determinantal point processes. arXiv preprint arXiv:1811.07245 .

Gillenwater, J., Kulesza, A., Taskar, B., 2012. Discovering diverse and salient threads in document collections, in: Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning, pp. 710–720.

Girolami, M., 2002. Mercer kernel-based clustering in feature space. IEEE Transactions on Neural Networks 13, 780–784. doi:10.1109/TNN.2002.1000150.

Gonzalez, T.F., 1985. Clustering to minimize the maximum intercluster distance. Theoretical computer science 38, 293–306.

Grimes, R., Lewis, J., Simon, H., 1994. A shifted block lanczos algorithm for solving sparse symmetric generalized eigenproblems. SIAM Journal on Matrix Analysis and Applications 15, 228–272. URL: https://doi.org/10.1137/S0895479888151111, doi:10.1137/S0895479888151111, arXiv:https://doi.org/10.1137/S0895479888151111.

Hafiz Affandi, R., Fox, E.B., Adams, R.P., Taskar, B., 2014. Learning the Parameters of Determinantal Point Process Kernels. ArXiv e-prints arXiv:1402.4862.

Hafiz Affandi, R., Fox, E.B., Taskar, B., 2013. Approximate Inference in Continuous Determinantal Point Processes. ArXiv e-prints arXiv:1311.2971.

Horn, R.A., Johnson, C.R., 2012. Matrix Analysis. 2nd ed., Cambridge University Press, USA.

Howley, T., Madden, M.G., 2006. An evolutionary approach to automatic kernel construction, in: Kollias, S., Stafylopatis, A., Duch, W., Oja, E. (Eds.), Artificial Neural Networks – ICANN 2006, Springer Berlin Heidelberg, Berlin, Heidelberg. pp. 417–426.

Hubert, L., Arabie, P., 1985. Comparing partitions. Journal of Classification 2, 193–218. URL: https://doi.org/10.1007/BF01908075, doi:10.1007/BF01908075.

Kang, B., 2013. Fast Determinantal Point Process Sampling with Application to Clustering, in: Burges, C., Bottou, L., Welling, M., Ghahramani, Z., Weinberger, K. (Eds.), Advances in Neural Information Processing Systems 26. Curran Associates, Inc., pp. 2319–2327.

Katsavounidis, I., Kuo, C.C.J., Zhang, Z., 1994. A new initialization technique for generalized lloyd iteration. IEEE Signal processing letters 1, 144–146.

Kaufmann, L., Rousseeuw, P., 1987. Clustering by means of medoids. Data Analysis based on the L1-Norm and Related Methods , 405–416.

Kulesza, A., Taskar, B., 2012. Determinantal point processes for machine learning. ArXiv e-prints arXiv:1207.6083.

Kullback, S., Leibler, R.A., 1951. On information and sufficiency. Ann. Math. Statist. 22, 79–86. URL: https://doi.org/10.1214/aoms/1177729694, doi:10.1214/aoms/1177729694.

Lanckriet, G.R.G., Cristianini, N., Bartlett, P., Ghaoui, L.E., Jordan, M.I., 2004. Learning the kernel matrix with semidefinite programming. J. Mach. Learn. Res. 5, 27–72.

Lanczos, C., 1950. An iteration method for the solution of the eigenvalue problem of linear differential and integral operators. United States Governm. Press Office Los Angeles, CA.

LeCun, Y., Cortes, C., Burges, C., 2010. Mnist handwritten digit database. ATT Labs [Online]. Available: http://yann.lecun.com/exdb/mnist 2.

Lehoucq, R.B., Sorensen, D.C., Yang, C., 1998. ARPACK users' guide: solution of large-scale eigenvalue problems with implicitly restarted Arnoldi methods. SIAM.

Lloyd, S.P., 1982. Least squares quantization in pcm. IEEE Trans. Inf. Theory 28, 129–136.

Macchi, O., 1975. The Coincidence Approach to Stochastic Point Processes. Advances in Applied Probability 7, 83–122.

Mariet, Z.E., Ovadia, Y., Snoek, J., 2019. Dppnet: Approximating determinantal point processes with deep networks, in: Advances in Neural Information Processing Systems, pp. 3223–3234.

Mashey, J.R., 1999. Big data and the next wave of infrastress problems, solutions, opportunities, in: USENIX Annual Technical Conference, Monterey, California, June, pp. 6–11.

Melnykov, V., Chen, W.C., Maitra, R., 2012. Mixsim: An r package for simulating data to study performance of clustering algorithms. Journal of Statistical Software, Articles 51, 1–25. URL: https://www.jstatsoft.org/v051/i12, doi:10.18637/jss.v051.i12.

Monti, S., Tamayo, P., Mesirov, J., Golub, T., 2003. Consensus clustering: A resampling-based method for class discovery and visualization of gene expression microarray data. Machine Learning 52, 91–118. doi:10.1023/A:1023949509487.

Murua, A., Wicker, N., 2014. The Conditional-Potts Clustering Model. Journal of Computational and Graphical Statistics 23, 717–739. doi:10.1080/10618600.2013.837828.

Parlett, B.N., Nour-Omid, B., 1989. Towards a black box lanczos program. Computer Physics Communications 53, 169 – 179. URL: http://www.sciencedirect.com/science/article/pii/0010465589901586, doi:https://doi.org/10.1016/0010-4655(89)90158-6.

Pena, J.M., Lozano, J.A., Larranaga, P., 1999. An empirical comparison of four initialization methods for the k-means algorithm. Pattern recognition letters 20, 1027–1040.

Rand, W.M., 1971. Objective criteria for the evaluation of clustering methods. Journal of the American Statistical Association 66, 846–850. URL: http://www.jstor.org/stable/2284239.

Shah, A., Ghahramani, Z., 2013. Determinantal clustering processes-a nonparametric bayesian approach to kernel based semi-supervised clustering. arXiv preprint arXiv:1309.6862 .

Silverman, B.W., 1986. Density estimation for statistics and data analysis. volume 26 of *Monographs on Statistics & Applied Probability*. Chapman and Hall. URL: https://www.crcpress.com/Density-Estimation-for-Statistics-and-Data-Analysis/Silverman/9780412246203.

Snijders, C., Matzat, U., Reips, U.D., 2012. " big data": big gaps of knowledge in the field of internet science. International journal of internet science 7, 1–5.

Strehl, A., Ghosh, J., 2002. Cluster ensembles — a knowledge reuse framework for combining multiple partitions. J. Mach. Learn. Res. 3, 583–617. doi:10.1162/153244303321897735.

Szelei, N., Tinoca, L., Pinho, A.S., 2019. Professional development for cultural diversity: the challenges of teacher learning in context. Professional Development in Education , 1–17.

Vega-Pons, S., Ruiz-Shulcloper, J., 2011. A survey of clustering ensemble algorithms. International Journal of Pattern Recognition and Artificial Intelligence 25, 337–372. doi:10.1142/S0218001411008683.

Vicente, S., Murua, A., 2020. Determinantal consensus clustering. arXiv e-prints .

Wagstaff, I.R., LaPorte, G., 2018. The importance of diversity and inclusion in the forensic sciences. National Institute of Justice Journal 279, 81–91.

Xiao, H., Rasul, K., Vollgraf, R., 2017. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. arXiv e-prints 1708.07747 .

Chapitre 4

High-dimensional determinantal variable selection

par

Sergio Vicente[1] et Alejandro Murua[2]

([1]) Département de Mathématiques et de Statistique, Université de Montréal.
E-mail : vicentes@dms.umontreal.ca

([2]) Département de Mathématiques et de Statistique, Université de Montréal.
E-mail : murua@dms.umontreal.ca

Cet article sera soumis dans Journal of Computational and Graphical Statistics.

Résumé. La sélection de variables est un aspect-clé de l'analyse de régression, particulièrement dans un contexte de grande dimension. L'évaluation de tous les modèles possibles est une tâche impraticable dans ce contexte. Par une approche bayésienne, les méthodes de Monte Carlo par chaînes de Markov constituent un puissant outil pour explorer l'espace des modèles. On introduit un processus ponctuel déterminantal graphique comme distribution *a priori* appropriée pour l'exploration des modèles. Sa principale propriété est d'introduire la corrélation négative entre variables explicatives qui sont similaires d'un point de vue de régression. La principale conséquence de cette propriété est l'attribution d'une faible probabilité à des modèles qui incluent des variables explicatives redondantes. Ceci induit une procédure efficace pour l'exploration de l'espace des modèles par les chaînes de Markov. Au moyen de simulations et d'applications à des données réelles, on montre que le fait de prendre le modèle graphique déterminantal comme distribution *a priori* de l'espace des modèles permet de détecter les variables explicatives informatives et d'éliminer la plupart des variables non informatives. On effectue également une étude comparative avec des distributions *a priori* uniformes et des approches classiques comme le lasso et la régression pas à pas. Cette étude montre la supériorité du modèle proposé.
Mots clés : Approximation de Laplace ; choix de Barker ; distribution *a priori* informative ; Metropolis-Hastings ; modèles hiérarchiques ; prédiction *a posteriori* ; régression linéaire ; régression logistique

Abstract. Variable selection is essential in regression analysis, specially in the context of high-dimensional data. Unfortunately, within this context, evaluating each possible model is infeasible. Following a Bayesian approach, Markov Chain Monte Carlo is a powerful method to explore the space of models. We introduce a graphical determinantal point process as a suitable prior for model exploration. Its main property is the negative correlation it introduces between explanatory variables that are similar from the point of view of regression. The main consequence of this property is the low probability assigned to models that include redundant explanatory variables. This produces a very efficient procedure for the exploration of the model space through Markov chain Monte Carlo. Through simulations and applications to real datasets, we show that the determinantal graphical model space prior is able to detect informative explanatory variables and eliminate most of the non-informative variables. A comparative study with uniform priors and classical approaches such as lasso and stepwise show the superiority of the proposed model.
Keywords: Barker's choice; hierarchical models; informative prior; Laplace's approximation; linear regression; logistic regression; Metropolis-Hastings; posterior prediction.

4.1. Introduction

Model or variable selection is important when there is uncertainty as to which of the variables are important to explain the variability of a response variable modeled as a generalized linear model (O'Hagan and Forster, 2004). When the number of explanatory variables is small, a popular approach is the best subset regression (Furnival and Wilson, 1974), which selects the best model among all possible combinations of the explanatory variables according to some statistical criteria. Among the most used criteria, we find the Akaike Information Criterion (Akaike, 1974), the Bayesian Information Criterion (Schwarz et al., 1978), and the Mallows's C_p criterion (Mallows, 1973). When the number of potential explanatory variables is large, the best subset regression technique becomes intractable. If we have p potential explanatory variables, then there are 2^p possible models, that is, an exponential growth with p. Several variable selection algorithms have been developed to evaluate different models in a high dimensional context. Among the most popular algorithms, we cite backward elimination (Marill and Green, 1963), forward selection (Pope and Webster, 1972), and LASSO (Tibshirani, 1996). A good review of variable selection algorithms can be found in Heinze et al. (2018). We also find variable selection methods under a Bayesian approach. Among the most popular methods, we cite the method of Kuo and Mallick (1998), the Gibbs variable selection (Dellaportas et al., 2002), the stochastic search variable selection (George and McCulloch, 1993) and model space approaches such as the reversible jump MCMC (Green, 1995) and the composite model space (Godsill, 1998). We can find a good review of these methods in O'Hara et al. (2009).

In this work, we propose a Bayesian approach for the problem of variable selection in a high-dimensional context. Bayesian models are more flexible. They can handle more complex models by allowing for the introduction of prior information through hierarchical modeling (O'Hagan and Forster, 2004). These characteristics best suit our purposes. According to Bayesian approach, a common criterion used for variable selection consists of computing the posterior model probability, that is, the posterior probability of the set of selected variables. The selection of which explanatory variables explain best the response variable is done by comparing the posterior model probabilities or posterior model odds. The computation of the posterior model probability requires the calculation of the marginal likelihood and the prior probability of each possible model. This becomes cumbersome and even intractable when the number of models is too large. Markov Chain Monte Carlo methods provide an efficient way to draw a sample of models from the posterior distribution of the model space through the construction of a Markov chain. Several known algorithms have been proposed to construct such a chain, such as the Metropolis-Hastings algorithm (Metropolis et al., 1953; Hastings, 1970), and the Gibbs sampler (Geman and Geman, 1984). The construction of a Markov chain is based on the definition of a transition kernel, which specifies the conditional

probability of choosing a new model, given a current model. A naive approach consists in choosing a uniform distribution as a transition kernel. Given a current state model, this kernel assigns a constant probability to all potential new models (Raftery et al., 1997). However, as pointed out by Zanella (2019) and Gagnon (2019), in general, the uniform transition kernel produces chains with low mixing properties. This is probably due to its failure to incorporate any information about the new models to be chosen. A transition kernel that accounts for information contained in these news models is more desirable, because the choice of a new model will be skewed towards high probability models (Gagnon, 2019), given rise to Markov chains with better mixing properties.

We introduce the determinantal point process (DPP) as a probabilistic model that defines a prior probability distribution on the collection of all possible models. A DPP is a random point process well-suited for modeling repulsion between the elements in the subsets it generates, thus engendering diversity (Kulesza and Taskar, 2012; Hafiz Affandi et al., 2013). The tendency for repulsion is conveyed in the determinant of a kernel similarity matrix between variables. Recalling the kernel-trick (Theodoridis and Koutroumbas, 2009), each variable may be mapped to an object in a higher dimensional space. The determinant is directly related to the volume spanned by the selected objects, and hence, to the similarity between the variables. A subset with very similar variables corresponds to a subset of nearly linearly dependent objects, and hence it is less likely to be selected. The possibility of expressing a DPP as a graphical model (Sadeghi and Rinaldo, 2019) over the space of all possible models makes the determinantal point process a suitable candidate to define prior probabilities over that space. The DPP model prior may also be used to define an informative transition kernel of a Markov chain generated by a Metropolis-Hastings procedure. Since with a DPP prior subsets containing dissimilar variables have higher probability, models with dissimilar explanatory variables are preferred. We show that using such an informative prior within the linear and logistic regression models results in Markov chains with higher mixing properties.

The paper is organized as follows. In Section 4.2, we recall the linear and logistic regression models under a Bayesian framework. In Section 4.3, we define the model space prior and introduce the determinantal point process as a suitable choice for this prior. A graphical model is used to express properly the determinantal point process as a prior probability model. In Section 4.4, we detail the methodology to perform Bayesian variable selection for the cases of a few explanatory variables, and high-dimensional data. In Section 4.5, we illustrate through simulations the variable selection methodologies for the linear and logistic regression models. In Section 4.6, we apply the same methodologies to some real datasets. We conclude with a few words in Section 4.7.

4.2. Bayesian regression

In this section, we recall the linear and logistic regression models within a Bayesian framework. In order to perform an efficient model/variable selection, we recur to the marginal likelihood of both regression models so as to obtain explicit expressions of the posterior probabilities. The computation is exact only for the linear regression model given conjugate priors for the coefficients and variance parameters. For the logistic regression model, we used the Laplace approximation.

4.2.1. The linear regression model

Consider the usual linear model with normal errors

$$y = X\beta + \varepsilon, \tag{24}$$

where $y = (y_1, \ldots, y_n)^T$ is the $n \times 1$ vector of the observed dependent variable, β is the $p \times 1$ vector of the regression coefficients, ε is the $n \times 1$ random error vector, and $X = (x_1|x_2|\cdots|x_n)^T$ is the $n \times p$ design matrix of the p explanatory variables $X_1, \ldots, X_p$ (here and throughout the paper, the superscript ^{T} stands for transposition). We assume that the elements of ε are independent and normally distributed, that is, $\varepsilon \sim N_n(0, \sigma^2 I_n)$, where $N_n(\cdot)$ is the n-variate Normal distribution, and I_n is the $n \times n$ identity matrix. We assume that data have been standardized, so that y and the observed vectors of each explanatory variable have zero mean and unit variance. The likelihood, or within a Bayesian framework, the conditional distribution of y, is given by $y|\beta, \sigma^2 \sim N_n(X\beta, \sigma^2 I_n)$.

In many applications of the linear model, the core problem is to select a subset of variables among p potential explanatory variables that may explain the dependent variable. If any possible combination of the p explanatory variables is an admissible linear model, we have 2^p possible models. Let $\mathcal{M}$ represent the sets of all possible models. Let $Z = (Z_1, \cdots, Z_p) \in \{0,1\}^p$ be a vector of binary variables indicating whether or not the explanatory variables $\{X_1, \cdots, X_p\}$ are included in a particular model. $Z_j = 1$ indicates that variables X_j is present in the model, whilst $Z_j = 0$, indicates that the variable is not part of the model (that is , β_j is set to 0). Any model M_k is given then by a particular configuration of values of Z. For example $Z = (0,0,\ldots,0)$ corresponds to the null model, and $Z = (1,1,\ldots,1)$, to the complete or full model. In the Bayesian framework, the model selection problem consists in finding the model $M_k \in \mathcal{M}$ that generated y with the highest probability. The linear model M_k is defined as in (24) by

$$y = X_k\beta_k + \varepsilon,$$

where X_k is the submatrix of the design matrix X, with the columns indexed by the elements of M_k, and β_k is the corresponding subvector of coefficients. The probability that M_k is the model to be selected, conditionally on having observed y, is computed with the posterior

model probability

$$P(M_k|y) = \frac{P(y|M_k)P(M_k)}{\sum_{M_k \in \mathcal{M}} P(y|M_k)P(M_k)}, \tag{25}$$

where

$$P(y|M_k) = \int\int P(y|\beta_k,\sigma_k^2,M_k)P(\beta_k,\sigma_k^2|M_k)\, d\beta_k\, d\sigma_k^2 \tag{26}$$

is the marginal likelihood of y under model M_k and $P(M_k)$ is the prior probability of model M_k. It is obtained from the conditional probability of y given the parameters of model M_k, $P(y|\beta_k,\sigma_k^2,M_k)$, where $y|\beta_k,\sigma_k^2,M_k \sim N_n\left(X_k\beta_k,\sigma_k^2 I_n\right)$.

As usually done with the regression model, we adopt a conjugate prior for the parameters of the regression. This also will help us to develop an efficient sampling of the models. Therefore, we assume that

$$\begin{aligned} P\left(\beta_k,\sigma_k^2|M_k\right) &= P\left(\beta_k|\sigma_k^2,M_k\right) P\left(\sigma_k^2|M_k\right) \\ &= P\left(\beta_k|\sigma_k^2,M_k\right) P\left(\sigma_k^2\right), \end{aligned}$$

where

$$\beta_k|\sigma_k^2,M_k \sim N_{p_k}\left(0,\sigma_k^2 I_{p_k}\right) \quad \text{and} \quad \sigma_k^2 \sim IG\left(\tfrac{\delta+p_k}{2},\tfrac{1}{2}\right),$$

where p_k is the number of explanatory variables present in M_k, and $IG(a,b)$ represents the Inverse-Gamma distribution, with parameters a and b. We fixed $\delta = 3$, so that the prior of σ_k^2 has finite mean and variance moments for all $k \geq 0$. Recall that in linear regression, the more explanatory variables are in the model, the better the estimation of the regression variance. Therefore, our prior for σ_k^2 is more diffuse, the less explanatory variables are present in the model.

Under these prior assumptions, the marginal likelihood defined in (26) can be computed in a closed-form, which can be found in Dobra et al. (2004), with computational details found in O'Hagan and Forster (2004):

$$P(y|M_k) = \frac{\Gamma\left(\frac{n+\delta+p_k}{2}\right)}{\pi^{n/2}\det(V_k)^{1/2}\left(1+q_k\right)^{\frac{n+\delta+p_k}{2}}\Gamma\left(\frac{\delta+p_k}{2}\right)}, \tag{27}$$

where $V_k = I_k + X_k^T X_k$ and $q_k = y^T y - y^T X_k V_k^{-1} X_k^T y$. The computation of (27) does not consider the special case of the null model, when there are no explanatory variables. Formally, for the null model, we take $p_k = 0$, $\det(V_k) = 1$ and $q_k = y^T y$. The computation of the posterior model probability given by (25) is then feasible using the closed-form of the conditional probability given in (27).

4.2.2. Logistic regression

Keeping the notation introduced previously, now we consider $y = (y_1, \ldots, y_n)^T$ to be an observed vector of independently distributed Bernoulli variables. The likelihood is given by the expression $P(y) = \prod_{i=1}^{n} \pi_i^{y_i}(1-\pi_i)^{1-y_i}$, where $\pi_i = P(y_i = 1|x_i)$, where x_i is the ith observation. We model the probabilities p_i with a logistic link function, that is,

$$\text{logit}(\pi_i) = \log\Big(\tfrac{\pi_i}{1-\pi_i}\Big) = (1, x_i)^T\beta \quad i = 1, \ldots, n,$$

where β is the $(p+1)$-dimensional vector of logistic regression coefficients. As before, we assume that all p explanatory variables have been standardized. Note that, contrary to the linear model, the vector β includes an intercept term. This inclusion is necessary to account for the baseline response probability.

As in the linear case, we assume a normal prior for the coefficients β_k associated with model $M_k \in \mathcal{M}$, that is $\beta_k|M_k \sim N_k(0, \sigma^2 I_{p_k})$, where $p_k = \text{card}(M_k)$, and σ^2 is a positive hyper-parameter. In our experiments, we have set $\sigma^2 = 1$.

Unfortunately, and contrary to the linear case, the closed form computation of the marginal likelihood $P(y|M_k)$ is not possible. Following Chipman et al. (2001) and Hans et al. (2007), we can approximate it using the Laplace integral approximation (Tierney and Kadane, 1986):

$$\widehat{P}(y|M_k) = (2\pi)^{\frac{p_k+1}{2}} \det\Big(-H^{-1}(\widehat{\beta_k})\Big)^{\frac{1}{2}} P(y|\widehat{\beta}_k, M_k) P(\widehat{\beta}_k|M_k), \tag{28}$$

where $\widehat{\beta}_k$ is the maximum a posteriori (MAP) estimator of $P(y|\beta_k, M_k)$ $P(\beta_k|M_k)$, and $H^{-1}(\widehat{\beta}_k)$ is the inverse Hessian of $\log P(y|\beta_k, M_k) + \log P(\beta_k|M_k)$, evaluated at $\widehat{\beta}_k$.

4.3. Model space prior

We still need a proper model prior distribution for the model space $\mathcal{M}$. According to Chipman et al. (2001), a popular choice is the uniform prior which attributes the same probability to each model. This is a non-informative choice because each model is treated equally. The authors warn that the choice of a non-informative prior can be misleading, since distinct model characteristics are not taken into account under the uniform assumption. A more subtle issue, also pointed out by the authors, occurs in setups where many models are very similar, and only a few are significantly different. It is precisely this last problem that motivated us to use the determinantal point process (DPP) as a model space prior. Indeed, in some domains of research, where the diversity of elements is a major concern, uniform random sampling can miss the coverage of all its facets. In contrast to simple random sampling, the determinantal point process (Borodin and Olshanski, 2000) introduces negative correlation between similar variables, so that models that include more variable diversity have higher probabilities. In other words, models that include similar variables have small probabilities.

The key to stress diversity in DPPs is the so-called kernel or Gram matrix. The strength of the negative correlations is incorporated in it (Kulesza and Taskar, 2012). Its entries represent a measure of similarity between each pair of variables, so that similar variables have less probability to co-occur. Because of this property, DPPs are already being used in machine learning as models for subset selection (Hafiz Affandi et al., 2013). However, to our knowledge, our work is the first to introduce DPPs into variable selection for regression.

4.3.1. Determinantal point process

Consider a discrete set of p elements, $V = \{1, \ldots, p\}$. A point process on V is a probability measure on 2^V, the set of all subsets of V. It is called a DPP if the probability mass function of any given random subset $A \in 2^V$ is equal to

$$P(A) = \frac{\det(L_A)}{\det(L + I_p)}, \tag{29}$$

where L is a $p \times p$ real, symmetric and positive semidefinite matrix measuring pairwise similarity between elements of V; L_A is the submatrix of L with rows and columns indexed by A, i.e., $L_A = [L_{ij}]_{i,j \in A}$ and I_p is the $p \times p$ identity matrix. We say that $A \sim DPP_V(L)$.

Determinants can be interpreted geometrically. As L is positive semidefinite, it admits a decomposition $L = B^T B$, where B is a $m \times p$ matrix, with $m \leq p$. Denoting the columns of B by B_j, for $j = 1, \ldots, p$, we have

$$P(A) \propto \det(L_A) = \mathrm{Vol}^2\left(\{B_j\}_{j \in A}\right),$$

where $\mathrm{Vol}^2\left(\{B_j\}_{j \in A}\right)$ represents the volume of the parallelepiped spanned by the elements of B_j, indexed by the elements of A. The columns of B can be interpreted as feature vectors describing the elements of V. Then, L measures similarity using dot products between feature vectors, and the probability assigned by a DPP to a subset A is related to the volume spanned by its associated feature vectors. Therefore, sets A that are more diverse in the sense of L, have larger probabilities, because their feature vectors span larger volumes.

4.3.2. The DPP graphical model prior

When working with multivariate data, it is common practice to analyze the relationships between the variables. Graphical models (Whittaker, 1990; Lauritzen, 1996) are a widespread tool to encompass multivariate independence and conditional independence between variables. An undirected graph $\mathcal{G}$ is a pair $\mathcal{G} = (V, E)$, where the set of vertices $V = \{1, \ldots, p\}$ represents observed variables, and E is the set of undirected edges $E \subset \{(i, j) : i, j \in V\}$. According to Mitra and Müller (2015), one of the advantages of graphical models is that any distribution on the graph space V can be specified. The most common graphical model is the

Gaussian graphical model. It models patterns of dependence of a p-dimensional Normal distribution through its precision matrix Σ^{-1}. A zero entry in Σ^{-1} is equivalent to the absence of the corresponding edge in E. The absence of an edge between two variables corresponds to the conditional independence of these two variables given the rest of the variables (Wermuth, 1976). Interestingly, and importantly for our work, Sadeghi and Rinaldo (2019) demonstrate that a DPP on V defines a undirected graph with essentially all the independence and conditional independence properties of a Gaussian graphical model. The similarity or kernel matrix of the DPP plays the role of the covariance matrix Σ of a Gaussian graphical model: the edge $(i,j) \notin E$ if and only if the (i,j) entry of the inverse of the kernel matrix is zero.

A DPP prior on $\mathcal{M}$ may be constructed from a DPP on V. Let L be an $p \times p$ kernel matrix whose entries are seen as pairwise similarities between the variables $\{X_1, \ldots, X_p\}$. A realization of $A \sim \text{DPP}_V(L)$ may be seen as a draw of the vector of indicators $Z = (Z_1, \ldots, Z_p)$ introduced in the previous section, as follows: $j \in A$ if and only if $Z_j = 1$, and $j \notin A$ if and only if $Z_j = 0$, $j = 1, \ldots, p$. Thanks to the results of Sadeghi and Rinaldo (2019), we can also see the DPP prior on V as a DPP graphical model on Z. Let $Z_k = (Z_{1k}, \ldots, Z_{pk})$ be the vector of variable indicators associated with M_k. With a slight abuse of notation, we will write $Z_k \sim \text{DPP}_V(L)$, and equivalently, also write $M_k \sim \text{DPP}_V(L)$. The prior distribution for M_k is given by

$$P(M_k) = P(Z_k) = \det(L_{Z_k}) / \det(L + I_p), \tag{30}$$

where L_{Z_k} stands for the submatrix of L composed of only the rows and columns j for which $Z_{jk} = 1$, $j = 1, \ldots, p$.

For the linear regression model, combining (27) and (30), the posterior model probability in (25) is given by

$$P(M_k|y) \propto \frac{\Gamma\left(\frac{n+\delta+p_k}{2}\right)}{\pi^{n/2} \det(V_k)^{1/2} (1+q_k)^{\frac{n+\delta+p_k}{2}} \Gamma\left(\frac{\delta+p_k}{2}\right)} \times \frac{\det(L_{Z_k})}{\det(L + I_p)}. \tag{31}$$

For the logistic regression model, combining (28) and (30), the posterior model probability in (25) is approximated by

$$P(M_k|y) \propto (2\pi)^{\frac{p_k+1}{2}} \det\Big(-H^{-1}(\widehat{\beta_k})\Big)^{\frac{1}{2}} P(y|\widehat{\beta}_k, M_k) P(\widehat{\beta}_k|M_k) \times \frac{\det(L_{Z_k})}{\det(L + I_p)}. \tag{32}$$

The computation of (30) requires the definition of the similarity matrix L. A natural choice for a similarity measure between variables is the correlation coefficient. However, the pairwise similarity between X_i and X_j should also reflect their degree of association within the context of regression. The partial correlation coefficient (Whittaker, 1990; Kim, 2015)

is a simple measure that takes into account the effect of the other variables when assessing correlation. Therefore, we adopted the partial correlation rather than the correlation coefficient, as a measure of pairwise similarity in the context of all explanatory variables.

Let $\mathcal{X} = \{X_1, \ldots, X_p\}$ be the collection of available variables. Denote by $\mathcal{X}_{-(i,j)}$ the set of all variables except X_i and X_j, i,$j = 1, \ldots$,p. The partial correlation coefficient between X_i and X_j is given by

$$r_{ij|\mathcal{X}_{-(i,j)}} = d_{ij}/\sqrt{d_{ii}d_{jj}},$$

where d_{ij} denotes the (i,j)-th entry of the inverse of the sample covariance matrix. Therefore, the similarity or kernel matrix L associated with our DPP model prior is given by

$$L = \left[r_{ij|\mathcal{X}_{-(i,j)}}\right]_{i,j=1}^{p}.$$

This kernel matrix can also be computed as $L = \text{diag}\left(1/\sqrt{d_{ii}}\right) W^{-1} \text{diag}\left(1/\sqrt{d_{jj}}\right)$, where W is the sample covariance matrix (Bilodeau, 2014). Using this latter expression, one can see directly that L is symmetric, and positive definite.

4.4. Model selection

When the number p of explanatory variables is small, we can compute for the linear model, or approximate for the logistic model, all the model probabilities given by (25). We can make all $\binom{2^p}{2}$ pairwise comparisons using the posterior odds, which for any two models M_k,$M_\ell \in \mathcal{M}$ is given by

$$\frac{P(M_k|y)}{P(M_\ell|y)} = \frac{P(y|M_k)}{P(y|M_\ell)} \frac{P(M_k)}{P(M_\ell)}. \tag{33}$$

The model posterior probabilities are the main tools for variable selection. They summarize the relevant information contained in y, and the model uncertainty in a post-data scenario. The model selected is the one showing the largest posterior odds among all models in $\mathcal{M}$.

When the number p of explanatory variables is large, calculation of (25) is not feasible, because the number of elements in $\mathcal{M}$ grows exponentially with p. Computing posterior odds involves $\binom{2^p}{2}$ evaluations. Hence, the calculation of all posterior odds becomes intractable. O'Hagan and Forster (2004) suggest that, when $p > 25$, Markov Chain Monte Carlo (MCMC) computation may be employed in an attempt to search for models with high posterior probability. Starting from an arbitrary model $M^{(1)} \in \mathcal{M}$, MCMC methods generate a Markov chain of models, $M^{(1)}, M^{(2)}, \ldots$ that converges to the target distribution, that is, to the model posterior distribution $P(M_i|y)$. The construction of the chain is based on the transition kernel

$$P\left(M^{(t+1)}|M^{(t)}\right), \quad t = 1, 2, \ldots, \tag{34}$$

which specifies the conditional distribution of model $M^{(t)}$, given the model $M^{(t-1)}$; here t indicates the time direction of the chain.

The most used MCMC method is the Metropolis-Hastings (MH) algorithm (Metropolis et al., 1953; Hastings, 1970). It works as follows. Let $M^{(1)} \in \mathcal{M}$ be an arbitrarily chosen model to initialize the MH algorithm. For $t = 1, 2, \ldots,$

(1) Sample a candidate M' from the transition kernel $P\left(M'|M^{(t)}\right)$;

(2) Compute the acceptance probability

$$p_{M',M^{(t)}} = \min\left\{1, \frac{P\left(M'|y\right)}{P\left(M^{(t)}|y\right)} \cdot \frac{P\left(M^{(t)}|M'\right)}{P\left(M'|M^{(t)}\right)}\right\},$$

where $P(M|y)$, is given by expression (31).

(3) Accept the *move* M' with probability $p_{M',M^{(t)}}$.

(4) If M' is rejected, stay at state $M^{(t)}$ and set $M^{(t+1)} = M^{(t)}$. If M' is accepted, move to state M' and set $M^{(t+1)} = M'$.

To initiate the chain, we draw a model $M^{(1)} \sim \text{DPP}_V(L)$ using (29). In practice, a draw from $\text{DPP}_V(L)$ is done using the algorithm of Ben Hough et al. (2006) and Kulesza and Taskar (2012). As the implementation of the MH algorithm relies on the specification of the transition kernel in (34), it is crucial to establish a conditional probability that guarantees the convergence of the chain to the posterior model distribution given by (25). Our transition kernels rely on the neighboring model approaches of Raftery et al. (1997) and Zanella (2019). Let $\delta\left(M^{(t)}\right)$ be the neighborhood of a current model $M^{(t)}$. Following the notation of Hans et al. (2007), we set $\delta\left(M^{(t)}\right) = \left\{M_+^{(t)}, M_-^{(t)}\right\}$, where $M_+^{(t)}$ is the set of all models with one more explanatory variable than $M^{(t)}$, and $M_-^{(t)}$ is the set of all models with one less explanatory variable than $M^{(t)}$. So, if the number of variables in $M^{(t)}$ is $\text{card}(M^{(t)}) = k$, we have $\text{card}(M_+^{(t)}) = k + 1$ and $\text{card}(M_-^{(t)}) = k - 1$. A typical way to proceed with the chain is to propose a model M' when the current state of the chain is $M^{(t)}$ uniformly among the models in $\delta\left(M^{(t)}\right)$. This results in an uniform transition kernel, that is,

$$P\left(M'|M^{(t)}\right) = \frac{1}{k + (p - k)} = \frac{1}{p}, \quad t = 1, 2, \ldots. \tag{35}$$

In fact, suppose again that $\text{card}(M^{(t)}) = k$. Then, there are k possibilities of explanatory variables to drop from $M^{(t)}$ and $p - k$ possibilities of variables to add to $M^{(t)}$, so that $\text{card}\Big(\delta\left(M^{(t)}\right)\Big) = p$. This approach was followed by Raftery et al. (1997). However, as pointed out by Gagnon (2019), poor models may often be proposed with uniform samplers such as the uniform transition kernel. When probabilities vary within neighborhoods, Markov chains with better mixing properties than uniform samplers should be used. This motivated us to propose two non-uniform transition kernels. The first one, uses the DPP presented in

(29). The transition probabilities are given by

$$P\left(M'|M^{(t)}\right) = \frac{\frac{\det(L_{M'})}{\det(L+I_p)}\mathbb{1}_{M'\in\delta\left(M^{(t)}\right)}}{\sum_{M\in\delta\left(M^{(t)}\right)}\frac{\det(L_M)}{\det(L+I_p)}} = \frac{\det(L_{M'})\mathbb{1}_{M'\in\delta\left(M^{(t)}\right)}}{\sum_{M\in\delta\left(M^{(t)}\right)}\det(L_M)}. \tag{36}$$

Since the DPP is the model space prior, our first transition kernel uses the prior information contained in the neighborhood of the current model $M^{(t)}$. This introduces a diversity effect, where models with more distinct variables are preferred as proposed candidates.

For the second transition kernel, we applied the procedure of Zanella (2019), who show that defining a transition kernel with the aid of a function h of the posterior model probability produces Markov chains with optimal mixing properties. The transition kernel probabilities are given by

$$P\left(M'|M^{(t)}\right) = \frac{h\left(\frac{P(M'|y)}{P\left(M^{(t)}|y\right)}\right)\mathbb{1}_{M'\in\delta\left(M^{(t)}\right)}}{\sum_{M\in\delta\left(M^{(t)}\right)}h\left(\frac{P(M|y)}{P\left(M^{(t)}|y\right)}\right)}, \tag{37}$$

where $h(x) = x/(1+x)$ and $P(M|y)$ is given by (31). This choice for the function h is commonly known as Barker's choice (Barker, 1965). In his paper, Zanella (2019) compares the efficiency of several MH schemes. He proposes transition kernels that incorporate information about the target distribution, and that bias the sampling of the proposals M' towards high-probability states. This information is incorporated through a multiplicative term of the form $g(\pi(M')/\pi(M^{(t)}))$, where $g(\cdot)$ is a continuous positive function on the positive reals, and $\pi(\cdot)$ is the target distribution. Zanella (2019) shows that these type of transition kernels are asymptotically optimal in terms of Peskun ordering (Peskun, 1973) when g satisfies $g(x) = xg(1/x),\ \forall x > 0$. Several g functions were proposed. Barker's choice $g(x) = x/(1+x)$ is the optimal choice that leads to the smallest mixing time.

After completing N iterations of the MH algorithm, we keep the chain $\left(M^{(1)}, M^{(2)}, \ldots, M^{(N)}\right)$ and compute the relative frequency of the models visited by the chain. For the computation of the relative frequency, we consider a *burn-in* period based on the recommendations in Van Ravenzwaaij et al. (2018). For the analysis, we only consider the elements of the chain after removing the elements corresponding to the *burn-in* period. Also, we keep only the ten most frequent models. The optimal model is selected among these ten models as the one associated to the highest value of the logarithm of the pairwise posterior odds given by (33).

In the case of the logistic regression, we only used the first kernel transition probabilities given by expression (36). At each step of the MH algorithm, the acceptance probability ratio requires the computation of (28) for the proposed candidate model. This requires finding the MAP estimates for the proposed model, which induces a higher computational cost than the linear model. Furthermore, the use of the transition kernel given by (37) with the logistic model would require the search for the MAP estimates of all current state

neighboring models. Hence, to avoid aggravating the computational cost incurred by the MH acceptance probability ratio, we do not work with the transition kernel given by (37) with the logistic model.

4.5. Experiments

4.5.1. Bayesian Linear Model

To illustrate our approach on Bayesian variable selection with DPP as model space prior in the case of a small number p of potential explanatory variables, we considered the multiple linear regression model

$$y_i \;=\; \beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \beta_3 x_{i3} + \beta_4 x_{i4} + \beta_5 x_{i5} + \varepsilon_i, \qquad i = 1, \ldots, n.$$

The $p = 5$ explanatory variables $(X_1, X_2, X_3, X_4, X_5)$ were drawn from a multivariate normal distribution with zero mean, unit variance, and pairwise autoregressive correlations given by $\text{cov}(X_i,X_j) = \rho^{|i-j|}$, $i,j = 1,\ldots,5$, with $\rho = 0.5$. For the generation of the dependent variable $y_i|\beta,\sigma^2 \sim N\left(x_i^T\beta,\sigma^2\right)$, the true values of the regression coefficients were set to $(\beta_0, \beta_1, \beta_2, \beta_3, \beta_4, \beta_5) = (1.5, 0, 2, 0, 1.5, 0)$. The variance was set to $\sigma^2 = 1$. A sample of $n = 50$ observations was generated. As in the setup of the linear model in Section 4.2.1, we standardized all the data. The model space $\mathcal{M}$ comprises $2^p = 32$ models. Table 4.1 displays the top five models ordered according to the largest posterior model probabilities given by (25) using (31).

Model	Posterior model probability
(X_2, X_4)	0.5132
(X_2, X_3, X_4)	0.2202
(X_1, X_2, X_4)	0.0934
(X_2, X_4, X_5)	0.0917
(X_1, X_2, X_3, X_4)	0.0335

Table 4.1 – Models with largest top 5 posterior probabilities.

We can see that the true model, with explanatory variables X_2 and X_4, is the model with the highest posterior odds. Its posterior probability is also large, representing almost 2.5 times the posterior probability of the model with the second largest posterior odds. We note that all the subsequent models always include the true model, as they consider X_2 and X_4.

To illustrate our approach on Bayesian variable selection with DPP as model space prior in the case of a large number p of potential explanatory variables, we considered the multiple

linear regression model

$$Y_i \;=\; \beta_0 + \beta_1 X_{i1} + \beta_2 X_{i2} + \cdots + \beta_{100} X_{i100} + \varepsilon_i, \qquad i = 1, 2, \ldots, n.$$

The $p = 100$ explanatory variables were generated with a multivariate normal distribution with zero mean, unit variance, and pairwise autoregressive correlations given by $\text{cov}(X_i,X_j) = \rho^{|i-j|}$, $i,j = 1,\ldots,100$, with autocorrelation parameter $\rho = 0.75$. For the generation of the dependent variable $Y_i|\beta,\sigma^2 \sim N\left(X_i^T\beta,\sigma^2\right)$, the true values of the regression coefficients were chosen with a random procedure. For the intercept, we set $\beta_0 = 1.5$. For the remaining 100 coefficients, we choose at random which of them will be set to 0. Finally, a value from the set $\{-2.5, -2, -1.5, 1.5, 2, 2.5\}$ was assigned randomly and sequentially to the non-zero coefficients. We set $\sigma^2 = 1$, and standardized all the data after obtaining the generated samples.

We studied the effect of two factors on the choice of the optimal model. These are the number of observations per dataset $n \in \{150, 500, 1500\}$, which correspond to low, medium and large values for n; and the number of non-zero regression coefficients of β : low (25), medium (55) and large (85). The model space $\mathcal{M}$ comprises $2^{100} = 1.267651 \times 10^{30}$ models. This implies the evaluation of $\binom{100}{2}$ posterior odds. Since the number of explanatory variables is large, we search for the optimal model with a Metropolis-Hastings algorithm. We considered three cases for the number N of MH iterations, including the *burn-in* period. These are shown in Table 4.2.

Scenario	Number of iterations (N)	*Burn-in* period	Chain length
I	10250	250	10000
II	102500	2500	100000
III	512500	12500	500000

Table 4.2 – Number of iterations, *burn-in* period and chain length for each scenario.

Nine datasets were generated: one for each combination of n and number of non-zero regression coefficients. For the transition kernel of the MH algorithm, we considered: (i) the transition kernel probabilities based on the model space DPP prior defined in (36), which we will refer to as *DPP*; (ii) the transition kernel based on Barker's choice, defined in (37), which we will refer to as *Barker*; and (iii) the uniform transition kernel defined in (35), which will be referred to as *Uniform*. For the first two cases (i) and (ii) a graphical DPP model space prior was used. For the uniform case, a uniform model prior was used. Table 4.3 shows the number of explanatory variables K in the selected optimal model, for the nine datasets, considering each scenario of Table 4.2, and the three aforementioned transition kernels. If the true model coincides with the selected optimal model, the result is highlighted in bold. We

also report, the number of visited models V. In order to measure the quality of the variables selected when the true model is not the one selected, we compared the estimated subset of variables with the true model through the so-called F_1-measure (Chekouo and Murua, 2018). This is defined as the harmonic average between recall and precision, which are two measures of retrieval quality introduced in the text mining literature (Allan et al., 1998). Let A, B be two subsets, and $|A|$ and $|B|$ be the number of elements in A and B, respectively. Recall and precision are given by recall $= |A \cap B|/|B|$, precision $= |A \cap B|/|A|$. So, recall is the proportion of variables in B that are in A, and precision is the proportion of variables in A that are also found in B. F_1 may be interpreted as a measure of the power of the method to detect all informative variables. It can be written as $F_1 = 2(p_o - q_o)/(p_o - q_o + p - q)$, where p_o is the true number of informative variables, q_o is the number of informative variables excluded from the model, and q is the number of non-informative variables excluded from the model.

Sample size	Non-zero coefficients	Transition kernel	Scenario								
			I			II			III		
			K	F_1	V	K	F_1	V	K	F_1	V
$n = 150$	25	DPP	27	0.96	2543	**25**	1.00	24781	**25**	1.00	123784
		Barker	**25**	1.00	9273	**25**	1.00	91029	**25**	1.00	446922
		Uniform	30	0.91	2398	28	0.94	25103	29	0.93	126164
	55	DPP	**55**	1.00	938	**55**	1.00	6584	**55**	1.00	30769
		Barker	**55**	1.00	7325	**55**	1.00	59417	**55**	1.00	240602
		Uniform	59	0.95	2122	59	0.95	20685	59	0.95	102859
	85	DPP	50	0.61	1284	56	0.68	8882	54	0.65	42045
		Barker	44	0.67	9107	46	0.64	91224	46	0.64	454243
		Uniform	90	0.88	799	86	0.87	4898	86	0.87	17846
$n = 500$	25	DPP	29	0.93	1990	**25**	1.00	20347	**25**	1.00	102038
		Barker	29	0.93	9304	**25**	1.00	92894	**25**	1.00	459587
		Uniform	31	0.89	1761	29	0.93	18668	28	0.94	93989
	55	DPP	**55**	1.00	934	**55**	1.00	8278	**55**	1.00	36739
		Barker	**55**	1.00	7336	**55**	1.00	59241	**55**	1.00	237037
		Uniform	60	0.96	1323	57	0.98	12993	**55**	1.00	64129
	85	DPP	**85**	1.00	158	**85**	1.00	328	**85**	1.00	761
		Barker	83	0.99	534	83	0.99	1547	83	0.99	2858
		Uniform	**85**	1.00	350	**85**	1.00	1780	**85**	1.00	4266
$n = 1500$	25	DPP	27	0.96	1178	**25**	1.00	11360	**25**	1.00	54901
		Barker	26	0.82	2987	26	0.82	22281	26	0.82	83287
		Uniform	26	0.98	1098	**25**	1.00	10916	**25**	1.00	53407
	55	DPP	**55**	1.00	544	**55**	1.00	4676	**55**	1.00	18936
		Barker	48	0.85	998	48	0.85	11378	48	0.85	51270
		Uniform	56	0.99	803	56	0.99	7771	56	0.99	34367
	85	DPP	**85**	1.00	84	**85**	1.00	219	**85**	1.00	468
		Barker	70	0.83	452	68	0.85	4204	67	0.83	20779
		Uniform	**85**	1.00	153	**85**	1.00	711	**85**	1.00	1730

Table 4.3 – Number of explanatory variables K of the optimal model for the nine datasets, with the true model highlighted in bold. Also shown are the number of visited models V, and the F_1 measure of model quality.

Table 4.4 shows the elapsed times in seconds for the two MH algorithms, *DPP* and *Barker*, when 10250 MH iterations were run with a dataset with 55 non-zero regression coefficients. The elapsed times were obtained with Julia Language version 1.1.1 on a PC with an Intel Core i5-4460 CPU running at 3.20GHz with 16GB of RAM.

Sample size	DPP	Barker
150	196.45	2182.20
500	265.35	3179.82
1500	372.60	5904.73

Table 4.4 – Elapsed time in seconds associated with two transitional kernels used to build a chain of length 10250 with a dataset with 55 non-zero coefficients.

Observing the results, we can see that using a DPP-based transition kernel presents clear advantages when compared to Barker and uniform transition kernels: the true model is almost always selected as the optimal model. Even when the true model is not selected, the quality of the model as measured by the F_1 value, remains high, indicating a good proximity between the selected model and the true model. Also, DPP needs to visit less models most of the times to select the optimal model, and requires a lower amount of time to run the MH algorithm when compared to Barker transition kernel. Based on these results, setting $N = 102500$ iterations, which includes a *burn-in* period of 2500 elements, is enough to obtain good results. Observe that the number of visited models by DPP-based algorithm decreases with the number n of observations, and the number of non-zero coefficients, specially for $n = 500$ and $n = 1500$. The uniform transition kernel is a bad option since it almost never selects the true model as optimal model. It only performs better for $n = 150$ and 85 non-zeros coefficients. Even when the uniform approach selects the true model as the optimal one, the DPP-based algorithm does the same with fewer visited models for most of the cases.

4.5.2. Bayesian Logistic Model

To illustrate our approach on Bayesian variable selection with DPP as model space prior in the case of logistic regression with a small number p of potential explanatory variables, we considered $p = 5$ explanatory variables. They were generated from a multivariate normal distribution with zero mean, unit variance, and pairwise autoregressive correlation, with autoregression parameter $\rho = 0.5$. The linear predictor is given by $\eta_i = x_i^T\beta$, for $i = 1, 2, \ldots, n$, where $x_i^T = (1, x_{i1}, x_{i2}, x_{i3}, x_{i4}, x_{i5})$ and $\beta = (\beta_0, \beta_1, \beta_2, \beta_3, \beta_4, \beta_5)^T$. To generate n observations of the dependent variable, we considered $y_i|\beta \sim$ Bernoulli(π_i), with $\pi_i = \exp(\eta_i)/(1 + \exp(\eta_i))$. The true values of the regression coefficients were set to $(1.5, 0, 2, 0, 1.5, 0)$, and we set $n = 50$. We standardized all the generated explanatory variables. The model space $\mathcal{M}$ has $2^p = 32$ models, with $\binom{32}{2} = 496$ posterior odds to compute.

To avoid scale problems with the posterior odds, we computed its logarithm. Table 4.5 shows the top five models with the largest posterior odds. We report the corresponding posterior model probabilities estimated with the Laplace approximation as shown in (32).

Model	Posterior model probability
(X_2, X_4)	0.3939
(X_1, X_2, X_4)	0.1316
(X_2, X_3, X_4)	0.0945
(X_1, X_4)	0.0488
(X_1, X_2, X_3, X_4)	0.0306

Table 4.5 – Models with largest top 5 posterior probabilities.

The true model, with explanatory variables X_2 and X_4, is the model with the highest posterior odds. Its posterior probability is about 3 times as large as the posterior probability of the model with the second largest posterior odds. We note that almost all the subsequent models always include the true model.

To illustrate our approach on Bayesian variable selection with DPP as model space prior in the case of logistic regression with a large number p of potential explanatory variables, we considered $p = 20$. This implies $2^{20} = 1{,}048{,}576$ possible models. The variables were generated from a multivariate normal distribution with zero mean, unit variance, and pairwise autoregressive correlation, with $\rho = 0.75$. The value of the intercept was set to $\beta_0 = 1.5$. The remaining coefficients were chosen from the set $\{-2.5, -2, -1.5, 1.5, 2, 2.5\}$. We created the same nine experimental conditions as in the linear model. We considered three cases for the number of observations per dataset $n \in \{150, 500, 1500\}$, and three cases for the number of non-zero logistic regression coefficients of β : low (5), medium (10) and large (15). We simulated a dataset for each condition. Because the number p of explanatory variables is moderately large, we search for the optimal model with a Metropolis-Hastings algorithm. In fact, for the calculation of the posterior odds, we would need to evaluate $\binom{2^{20}}{2} = 549,755,289,600$ pairs of models. For the number N of iterations of the MH algorithm, we considered the same three cases as we did in the linear model. They are shown in Table 4.2. Recall that, besides the uniform transition kernel, we only used the DPP-based transition kernel given by (36) to build the chain. Table 4.6 displays the number of explanatory variables in the selected model for each situation. If the true model coincides with the selected model, the result is highlighted in bold. We also report, the number of visited models by the chain, and the quality of the model as given by the F_1 measure defined above.

Sample size	Non-zero coefficients	Transition kernel	Scenario								
			I			II			III		
			K	F_1	V	K	F_1	V	K	F_1	V
$n = 150$	5	DPP	**5**	1.00	2757	**5**	1.00	10711	**5**	1.00	20342
		Uniform	**5**	1.00	3748	**5**	1.00	17874	**5**	1.00	31679
	10	DPP	8	0.78	2281	8	0.78	11112	8	0.78	26164
		Uniform	9	0.84	3529	9	0.84	15088	9	0.84	30363
	15	DPP	8	0.33	3473	9	0.42	21557	9	0.50	57360
		Uniform	11	0.57	4715	10	0.50	28202	10	0.50	66848
$n = 500$	5	DPP	**5**	1.00	1223	**5**	1.00	4432	**5**	1.00	8819
		Uniform	**5**	1.00	1586	**5**	1.00	6682	**5**	1.00	13366
	10	DPP	**10**	1.00	487	**10**	1.00	1344	**10**	1.00	2657
		Uniform	**10**	1.00	794	**10**	1.00	1773	**10**	1.00	2877
	15	DPP	**15**	1.00	245	**15**	1.00	728	**15**	1.00	1177
		Uniform	**15**	1.00	372	**15**	1.00	623	**15**	1.00	998
$n = 1500$	5	DPP	**5**	1.00	696	**5**	1.00	2709	**5**	1.00	5391
		Uniform	**5**	1.00	972	**5**	1.00	4013	**5**	1.00	7970
	10	DPP	**10**	1.00	206	**10**	1.00	400	**10**	1.00	620
		Uniform	**10**	1.00	291	**10**	1.00	693	**10**	1.00	890
	15	DPP	**15**	1.00	22	**15**	1.00	29	**15**	1.00	30
		Uniform	**15**	1.00	29	**15**	1.00	32	**15**	1.00	32

Table 4.6 – Number of explanatory variables in the optimal model chosen on each situation, with the true model highlighted in bold. Also shown are the number of visited models V, and the F_1 measure of model quality.

We can conclude that the MH algorithm with DPP-based transition kernel performs well at visiting high posterior probability models. This is specially true when the number of observations is large. The results show that the number of visited models decreases with the number of observations. As with the linear model case, this only corroborates what is generally known about fitting any regression model: the larger the data size, the better the fit. The data speak by themselves, hinting at the true model, or a close-to-optimal model. Based on the results, we can say that using moderate size chains with $N = 10000$ or $N = 100000$ iterations, is enough to obtain good results. Considering the results with a uniform transition kernel, we can see that the results are very similar to those of DPP. So, unlike the linear model, the gains achieve using a DPP-based transition kernel are not as large. However, in general, DPP achieves the true model with less effort than the uniform case. We recall that the uniform transition kernel results do not use our graphical DPP space model prior. So, to study if the prior has an effect on the procedure we decided to study our graphical DPP prior with a uniform transition kernel approach. We refer to this strategy as the *mixed approach*, so as to differentiate from the pure uniform approach (that is, uniform prior and uniform transition kernel), and the pure DPP approach (that

is, graphical DPP prior and DPP-based transition kernel). Table 4.7 displays the number of explanatory variables selected by the optimal model. This number is highlighted in bold when the optimal model coincides with the true model.

Sample size	Non-zero coefficients	Scenario		
		I	II	III
$n = 150$	5	**5** (2322)	**5** (9702)	**5** (19403)
	10	8 (2315)	8 (11444)	8 (26868)
	15	10 (3910)	9 (24075)	9 (62807)
$n = 500$	5	**5** (925)	**5** (3940)	**5** (7788)
	10	**10** (512)	**10** (1446)	**10** (2614)
	15	**15** (339)	**15** (825)	**15** (1279)
$n = 1500$	5	**5** (593)	**5** (2322)	**5** (4655)
	10	**10** (202)	**10** (419)	**10** (593)
	15	**15** (22)	**15** (31)	**15** (32)

Table 4.7 – Number of explanatory variables in the optimal model chosen on each situation with the mixed Graphical DPP prior and uniform transition kernel approach. If the true model is the selected model, then the number if has been highlighted in bold. The number of models visited by the chains are shown within parentheses.

In terms of the number of selected explanatory variables, setting a DPP prior for the model space and a uniform transition kernel achieves the same results as setting a DPP for both (see Table 4.6). In terms of the number of visited models, the mixed approach shows a mixed behavior. Generally, more models are visited by mixed approach than the pure DPP approach when the number of explanatory variables is low. But, the mixed approach visits less models than the pure uniform approach for all cases.

4.6. Application to real data

In this section, we illustrate the application of our DPP-based space prior Bayesian variable selection to some real datasets. The datasets were obtained from two publicly available websites: the *UCI Machine Learning Repository* (Dua and Graff, 2017), and *OpenML* (Vanschoren et al., 2013). We have split the applications in two sections: one for the linear model and another for the logistic model.

4.6.1. Linear Model

Table 4.8 shows the selected real datasets for the linear model, showing some of their features: its source, number n of observations and number p of potential explanatory variables.

All the datasets have been processed so as eliminate any missing data, and redundant (i.e., perfectly correlated) variables.

Dataset	Source	n	p
Tecator	OpenML	215	100
Mtp	OpenML	4450	198
Communities	UCI	1993	99
Scm1d	OpenML	9803	280

Table 4.8 – Selected datasets from the UCI and OpenML repositories

For each dataset, we built a Markov chain with MH algorithm. We set $N = 102500$ iterations, including a *burn-in* period of 2500 elements, as suggested by the results of Section 4.5.1. We decided also to compute the transition kernel with approach (36), as suggested by the same results. Table 4.9 shows the number of selected explanatory variables of the final model, the number of models visited by the Markov chain and the visit frequency of the final model.

Dataset	Number of selected variables	Number of visited models	Visits frequency
Tecator	27	31184	0.0006
Mtp	37	4187	0.0025
Communities	13	8521	0.0040
Scm1d	58	8680	0.0010

Table 4.9 – Number of selected explanatory variables of the final model, number of visited models and visits frequency of the final model, for the Linear Model

To evaluate the predictive power of the models selected with DPP, we used a K-fold cross validation procedure. For each model of Table 4.9, we extracted the portion of the whole dataset corresponding to the selected variables and divided it randomly into K folds. We fitted a Bayesian linear model, as described in Section 4.2, using $K - 1$ folds for training and the remaining K-th fold for testing. The posterior predictive distribution of the Bayesian linear model was estimated from 2000 samples of a MH chain and the estimates of the coefficients were obtained with the posterior mean. Using the estimated coefficients, we computed the mean posterior prediction for the cases considered in the testing fold and obtained the prediction error and the corresponding root mean square error (RMSE). Convergence of the fitted models was measured with Gelman and Rubin's test (Gelman et al., 1992) using four parallel chains. Repeating the process until every individual fold is used for testing, we obtain K values for the RMSE, which we summarize by a mean and a standard

deviation. Here, we chose $K = 10$. For comparison, we considered the models whose variables have been selected by stepwise regression (Efroymson, 1960; Hocking, 1976), and by LASSO (Tibshirani, 1996). For LASSO in particular, the tuning parameter λ was chosen by cross validation. For these two methods, after extracting the portion of the whole dataset corresponding to the selected variables, we used the same process as DPP: a K-fold cross validation procedure with a Bayesian linear model fitted to each training set, with $K = 10$. We obtain K RMSE values for the two methods, summarized by the mean and the standard deviation. Table 4.10 shows the mean and the standard deviation of the RMSE for the three methods, along with the number of selected explanatory variables.

Dataset	Method	Number of selected variables	RMSE
Tecator	DPP	27	0.23 (0.04)
	Stepwise	63	0.22 (0.05)
	LASSO	25	0.25 (0.05)
Mtp	DPP	37	0.70 (0.03)
	Stepwise	132	0.70 (0.03)
	LASSO	195	0.75 (0.13)
Communities	DPP	13	0.58 (0.03)
	Stepwise	54	0.57 (0.05)
	LASSO	76	0.58 (0.03)
Scm1d	DPP	58	0.33 (0.02)
	Stepwise	140	0.33 (0.02)
	LASSO	173	0.33 (0.02)

Table 4.10 – Number of selected explanatory variables, and RMSE means and standard deviations (within parentheses), associated with the three variable selection methods.

We can see that the models selected by the DPP-based procedure always selects fewer explanatory variables than the other two popular methods. As we noted through the simulations, the DPP-based procedure is very efficient, requiring relatively few model visits to obtain a good model. Now, we see that this efficiency is accompanied with parsimony in the chosen model. Moreover, in terms of prediction error, this parsimony seems advantageous since the DPP-based procedure performs similarly or better than stepwise and LASSO.

4.6.2. Logistic Model

For the Logistic Model, we present in Table 4.11, the selected datasets, with their source, number n of observations and number p of potential explanatory variables.

Dataset	Source	n	p
Bodyfat	OpenML	252	14
Halloffame	OpenML	1320	15
Diabetic	UCI	1151	16
Lungcancer	OpenML	226	19
Vehicle	OpenML	846	18

Table 4.11 – Selected datasets from the UCI and OpenML repositories

For each dataset, we constructed a Markov chain with MH algorithm, setting $N = 10250$ iterations, which includes a *burn-in* period of 250 elements. Table 4.12 reports the number of selected explanatory variables of the final model, the number of models visited by the Markov chain and the visit frequency of the final model.

Dataset	Number of selected variables	Number of visited models	Visits frequency
Bodyfat	2	2632	0.0086
Halloffame	3	877	0.0742
Diabetic	7	707	0.0193
Lungcancer	2	3281	0.0162
Vehicle	11	223	0.0738

Table 4.12 – Number of selected explanatory variables of the final model, number of visited models and visits frequency of the final model, for the Logistic Model.

Globally speaking, we can see that the number of visited models for real data follows the tendency of the visited models for simulated data (Table 4.6), if we consider the combination of n with the number of selected variables.

To evaluate the predictive power of each dataset, we proceed as in the Bayesian linear model, and compare the model selected by the DPP-based procedure to the models selected by stepwise regression and LASSO. Table 4.13 shows the mean and the standard deviation of the RMSE for the three methods, along with the number of selected explanatory variables.

Dataset	Method	Number of selected variables	RMSE
Bodyfat	DPP	2	0.10 (0.05)
	Stepwise	1	0.12 (0.04)
	LASSO	8	0.13 (0.05)
Halloffame	DPP	3	0.23 (0.03)
	Stepwise	8	0.20 (0.03)
	LASSO	6	0.20 (0.04)
Diabetic	DPP	7	0.41 (0.02)
	Stepwise	11	0.41 (0.01)
	LASSO	16	0.41 (0.02)
Lungcancer	DPP	2	0.34 (0.04)
	Stepwise	9	0.34 (0.07)
	LASSO	5	0.35 (0.05)
Vehicle	DPP	2	0.16 (0.02)
	Stepwise	8	0.16 (0.06)
	LASSO	16	0.16 (0.03)

Table 4.13 – Number of selected explanatory variables by each method and their corresponding RMSE mean and standard deviation (within parentheses), for the Bayesian logistic model.

Again, as already observed with the Bayesian linear model, the DPP-based procedure tends to choose more parsimonious models, that is, with less explanatory variables, than the other two methods (the exception is the Bodyfat dataset). In terms of RMSE, the DPP-based procedure performs similarly or better than the other two methods, for the majority of the datasets. To summarize, the DPP-based procedure is able to select parsimonious models with good predictive power.

4.7. Conclusions

We proposed the discrete graphical determinantal point process as a prior distribution for the model space in linear and logistic regression models. But this idea is not limited to these models: the same prior may be used for more complex generalized linear models. We also incorporated this prior to define the transition probability kernel of a Metropolis-Hastings algorithm so as to perform informative Bayesian variable selection in the context of high-dimensional data. Our strategy is very efficient since we obviate the sampling of model parameters, and work directly with the marginal likelihoods. In the case of generalized linear models, such as the logistic regression, numerical approximations to the marginal likelihood are needed. In this work, we used the Laplace approximation with good results

for a moderate number of variables. Further studies are needed to investigate the usefulness of more complex quadrature procedures in the case of much bigger data dimensions.

For the linear regression model, DPP proved to be a suitable prior that allow the retrieval of the true model under several simulated experimental conditions. It also proved to be a very efficient approach, requiring a low number of visited models in the model space. To achieve a very good performance, about 100000 samples of the MH algorithm are sufficient if the model space contains of the order of 2^{100} models. Using a non informative uniform prior for the model space does not produce as good results. Even when the best selected model is the same as the one selected by the DPP-based approach, the uniform approach takes a greater effort, requiring the visiting of a larger number of models. For the logistic regression, the results obtained with a DPP-based prior or a uniform prior are similar. However, we still recommend a DPP-based approach to define the model space prior, since the same results are attained with fewer visited models. About 10000 samples of the MH algorithm are sufficient if the model space contains of the order of 2^{20} models.

A variety of interesting questions remain for future research. Three immediate issues are the following. The extension of Bayesian variable selection with DPP to datasets with both continuous and categorical variables; this implies the choice of a convenient measure of similarity between mixed variables. The development a variable selection methodology with DPP that can be applied to generalized linear models with complex likelihoods; as mentioned above, this would entail the study of suitable quadrature procedures in high-dimensions. And the replacement of the partial correlation to other measures of similarity between explanatory variables.

Bibliography

Akaike, H., 1974. A new look at the statistical model identification. IEEE transactions on automatic control 19, 716–723.

Allan, J., Carbonell, J., Doddington, G., Yamron, J., Yang, Y., 1998. Topic detection and tracking pilot study: Final report., in: Proceedings of the DARPA Broadcast News Transcription and Understanding Workshop, pp. 194–218.

Barker, A.A., 1965. Monte carlo calculations of the radial distribution functions for a proton? electron plasma. Australian Journal of Physics 18, 119–134.

Ben Hough, J., Krishnapur, M., Peres, Y., Virág, B., 2006. Determinantal processes and independence. Probability Surveys [electronic only] 3, 206–229.

Bilodeau, M., 2014. Graphical lassos for meta-elliptical distributions. Canadian Journal of Statistics 42, 185–203.

Borodin, A., Olshanski, G., 2000. Distributions on Partitions, Point Processes, and the Hypergeometric Kernel. Communications in Mathematical Physics 211, 335–358. doi:10.1007/s002200050815, arXiv:math/9904010.

Chekouo, T., Murua, A., 2018. High-dimensional variable selection with th eplaid mixture model for clustering. Computational Statistics 33, 1475–1496.

Chipman, H., George, E.I., McCulloch, R.E., 2001. The practical implementation of bayesian model selection. Institute of Mathematical Statistics Lecture Notes - Monograph Series Volume 38, 65–116.

Dellaportas, P., Forster, J.J., Ntzoufras, I., 2002. On bayesian model and variable selection using mcmc. Statistics and Computing 12, 27–36.

Dobra, A., Hans, C., Jones, B., Nevins, J.R., Yao, G., West, M., 2004. Sparse graphical models for exploring gene expression data. Journal of Multivariate Analysis 90, 196 – 212. doi:https://doi.org/10.1016/j.jmva.2004.02.009. special Issue on Multivariate Methods in Genomic Data Analysis.

Dua, D., Graff, C., 2017. UCI machine learning repository. URL: http://archive.ics.uci.edu/ml.

Efroymson, M., 1960. Multiple regression analysis. Mathematical methods for digital computers , 191–203.

Furnival, G.M., Wilson, R.W.J., 1974. Regressions by leaps and bounds. Technometrics 16, 499–511.

Gagnon, P., 2019. A step further towards automatic and efficient reversible jump algorithms. arXiv preprint arXiv:1911.02089 .

Gelman, A., Rubin, D.B., et al., 1992. Inference from iterative simulation using multiple sequences. Statistical science 7, 457–472.

Geman, S., Geman, D., 1984. Stochastic relaxation, gibbs distributions, and the bayesian restoration of images. IEEE Transactions on pattern analysis and machine intelligence 6, 721–741.

George, E.I., McCulloch, R.E., 1993. Variable selection via gibbs sampling. Journal of the American Statistical Association 88, 881–889.

Godsill, S.J., 1998. On the relationship between MCMC model uncertainty methods. Citeseer.

Green, P.J., 1995. Reversible jump markov chain monte carlo computation and bayesian model determination. Biometrika 82, 711–732.

Hafiz Affandi, R., Fox, E.B., Taskar, B., 2013. Approximate Inference in Continuous Determinantal Point Processes. ArXiv e-prints arXiv:1311.2971.

Hans, C., Dobra, A., West, M., 2007. Shotgun Stochastic Search for "Large p" Regression. Journal of the American Statistical Association 102, 507–516. doi:10.1198/016214507000000121, arXiv:https://doi.org/10.1198/016214507000000121.

Hastings, W.K., 1970. Monte carlo sampling methods using markov chains and their applications. Biometrika 57, 97–109.

Heinze, G., Wallisch, C., Dunkler, D., 2018. Variable selection–a review and recommendations for the practicing statistician. Biometrical Journal 60, 431–449.

Hocking, R.R., 1976. A biometrics invited paper. the analysis and selection of variables in linear regression. Biometrics 32, 1–49.

Kim, S., 2015. ppcor: An r package for a fast calculation to semi-partial correlation coefficients. Communications for statistical applications and methods 22, 665.

Kulesza, A., Taskar, B., 2012. Determinantal point processes for machine learning. ArXiv e-prints `arXiv:1207.6083`.

Kuo, L., Mallick, B., 1998. Variable selection for regression models. Sankhyā: The Indian Journal of Statistics, Series B , 65–81.

Lauritzen, S.L., 1996. Graphical models. volume 17. Clarendon Press.

Mallows, C.L., 1973. Some comments on cp. Technometrics15 , 661–675.

Marill, T., Green, D., 1963. On the effectiveness of receptors in recognition systems. IEEE transactions on Information Theory 9, 11–17.

Metropolis, N., Rosenbluth, A.W., Rosenbluth, M.N., Teller, A.H., Teller, E., 1953. Equation of state calculations by fast computing machines. The journal of chemical physics 21, 1087–1092.

Mitra, R., Müller, P., 2015. Nonparametric Bayesian inference in biostatistics. Springer.

O'Hagan, A., Forster, J.J., 2004. Kendall's advanced theory of statistics, volume 2B: Bayesian inference. volume 2. Arnold.

O'Hara, R.B., Sillanpää, M.J., et al., 2009. A review of bayesian variable selection methods: what, how and which. Bayesian analysis 4, 85–117.

Peskun, P.H., 1973. Optimum monte-carlo sampling using markov chains. Biometrika 60, 607–612.

Pope, P., Webster, J., 1972. The use of an f-statistic in stepwise regression procedures. Technometrics 14, 327–340.

Raftery, A.E., Madigan, D., Hoeting, J.A., 1997. Bayesian model averaging for linear regression models. Journal of the American Statistical Association 92, 179–191.

Sadeghi, K., Rinaldo, A., 2019. Markov properties of discrete determinantal point processes, in: The 22nd International Conference on Artificial Intelligence and Statistics, pp. 1313–1321.

Schwarz, G., et al., 1978. Estimating the dimension of a model. The annals of statistics 6, 461–464.

Theodoridis, S., Koutroumbas, K., 2009. Pattern recognition. 2003. Google Scholar Google Scholar Digital Library Digital Library .

Tibshirani, R., 1996. Regression shrinkage and selection via the lasso. Journal of the Royal Statistical Society: Series B (Methodological) 58, 267–288.

Tierney, L., Kadane, J.B., 1986. Accurate approximations for posterior moments and marginal densities. Journal of the american statistical association 81, 82–86.

Van Ravenzwaaij, D., Cassey, P., Brown, S.D., 2018. A simple introduction to markov chain monte–carlo sampling. Psychonomic bulletin & review 25, 143–154.

Vanschoren, J., van Rijn, J.N., Bischl, B., Torgo, L., 2013. Openml: Networked science in machine learning. SIGKDD Explorations 15, 49–60. URL: https://www.openml.org/, doi:10.1145/2641190.2641198.

Wermuth, N., 1976. Analogies between multiplicative models in contingency tables and covariance selection. Biometrics , 95–108.

Whittaker, J., 1990. Graphical models in applied multivariate statistics. John Wiley & Sons.

Zanella, G., 2019. Informed proposals for local mcmc in discrete spaces. Journal of the American Statistical Association , 1–27.

Conclusion

Le DPP a révélé de nombreux avantages en statistique tout au long de cette thèse. En introduisant une dimension de diversité en tant que méthode d'échantillonnage, le DPP a un effet positif sur la qualité des résultats finaux. Tout d'abord, on a montré qu'en choisissant le DPP comme méthode d'initialisation d'un algorithme de partitionnement de données par consensus, les partitions finales obtenues sont meilleures que celles que l'on obtient en prenant la distribution uniforme pour choisir les points initiaux de l'algorithme. Pour obtenir une partition de bonne qualité avec le DPP, il n'est pas nécessaire d'exécuter l'algorithme un grand nombre de fois. En moyenne, 100 à 200 répétitions sont suffisantes. Pour obtenir des résultats similaires avec une sélection uniforme de points, un plus grand nombre de répétitions de l'algorithme est nécessaire. Par la suite, en appliquant le partitionnement de données par consensus aux données avec un grand nombre d'observations, on a montré que l'utilisation du DPP produisait aussi de meilleurs résultats. En utilisant la méthode du NNGP ou la méthode d'échantillonnage de petites sous-matrices sélectionnées à partir de la matrice noyau, on atteint des partitions de données de qualité supérieure à celles qui résultent d'une sélection aléatoire de points. L'approche basée sur le NNGP s'est montrée efficace pour contourner le problème computationnel imposé par la décomposition spectrale de la matrice noyau du DPP. En utilisant une matrice creuse dont l'inverse est une approximation de la matrice noyau du DPP, il suffit d'extraire un sous-ensemble de valeurs propres de la matrice creuse pour pouvoir sélectionner des points par le DPP, plutôt que d'extraire toutes les valeurs propres et vecteurs propres associés de la matrice noyau originale. L'approche basée sur l'échantillonnage de petites matrices à partir de la matrice noyau originale s'est aussi révélé prometteuse. En effet, en sélectionnant des matrices de plus petite dimension et en sélectionnant des points avec le DPP dans ces sous-matrices, la qualité des résultats obtenus est comparable à celle de l'approche basée sur le NNGP. Finalement, la dernière partie de la thèse a abordé l'utilisation du DPP dans un contexte de sélection de variables par une approche bayésienne. Cette partie s'est spécialement penchée sur le cas où on a un grand nombre de variables. En instaurant le DPP comme distribution *a priori* de l'espace de tous les modèles possibles et en construisant des chaînes de Markov avec l'algorithme de MH, on a montré que le DPP sélectionnait de meilleurs

sous-ensembles de variables que la distribution uniforme prise comme distribution *a priori*. Cette conclusion est renforcée par le fait de définir les probabilités de transition entre les états de la chaîne avec le DPP, plutôt qu'avec la distribution uniforme. Les chaînes de Markov basées sur le DPP atteignent très vite des modèles avec une grande probabilité *a posteriori*, privilégiant une parcimonie et efficacité dans ses visites des membres de l'espace des modèles. Ceci a largement été exploré par des simulations en régression linéaire et logistique. Des applications à des données réelles ont également montré les avantages de définir le DPP comme distribution *a priori* de l'espace de tous les modèles. Suivant la tendance des résultats obtenus au moyen des simulations, le DPP mise sur la parcimonie de variables comme choix de modèle final. Les résultats ont également montré que les modèles sélectionnés par le DPP comme distribution *a priori* ont une excellente performance en termes de puissance prédictive, rivalisant avec des méthodes classiques de sélection de variables comme le LASSO et la sélection pas à pas.

Cette thèse ne couvre évidemment pas tous les aspects des sujets présentés. Il reste encore des lacunes et des zones d'ombres à combler. Il y a encore des détails à améliorer et des aspects à approfondir. Plusieurs questions pertinentes peuvent également être posées. Mais plutôt que de considérer tout cela comme une mauvaise nouvelle, c'est plutôt le contraire qui s'impose. Cette thèse a ouvert la voie à de futures recherches dans le domaine du DPP et nous dit que l'histoire n'est pas terminée. Les très bons résultats obtenus avec le DPP servent de motivation et d'encouragement à appliquer ce processus ponctuel de façon plus intense en statistique, comme méthode d'échantillonnage.