Exact Sampling of Determinantal Point Processes without Eigendecomposition

Claire Launay, Bruno Galerne, Agnès Desolneux

Applied Probability Trust (18 April 2020)

EXACT SAMPLING OF DETERMINANTAL POINT PROCESSES WITHOUT EIGENDECOMPOSITION

CLAIRE LAUNAY,* *Université de Paris*

BRUNO GALERNE,** *Université d'Orléans*

AGNÈS DESOLNEUX,*** *CNRS and ENS Paris-Saclay*

Abstract

Determinantal point processes (DPPs) enable the modeling of repulsion: they provide diverse sets of points. The repulsion is encoded in a kernel K that can be seen, in a discrete setting, as a matrix storing the similarity between points. The main exact algorithm to sample DPPs uses the spectral decomposition of K, a computation that becomes costly when dealing with a high number of points. Here, we present an alternative exact algorithm to sample in discrete spaces that avoids the eigenvalues and the eigenvectors computation. The method used here is innovative and numerical experiments show competitive results with respect to the initial algorithm.

Keywords: Determinantal point processes; Exact Sampling; Thinning; Cholesky decomposition; General marginal

2010 Mathematics Subject Classification: Primary 68U20

Secondary 60G55

Determinantal point processes (DPPs) are processes that capture negative corre-

* Postal address: Laboratoire MAP5

Université de Paris, CNRS

Paris, 75006, FRANCE

** Postal address: Institut Denis Poisson,

Université d'Orléans, Université de Tours, CNRS

Orléans, 45100, FRANCE

*** Postal address: Centre Borelli, CNRS

ENS Paris Saclay

Gif-sur-Yvette, 91190, FRANCE

lations. The more similar two points are, the less likely they are to be sampled simultaneously. Then DPPs tend to create sets of diverse points. They naturally arise in random matrix theory [22] or in the modelling of a natural repulsive phenomenon like the repartition of trees in a forest [31]. Ever since the work of Kulesza and Taskar [27], these processes have become more and more popular in machine learning, because of their ability to draw subsamples that account for the inner diversity of data sets. This property is useful for many applications, such as summarizing documents [14], improving a stochastic gradient descent by drawing diverse subsamples at each step [45] or extracting a meaningful subset of a large data set to estimate a cost function or some parameters [44, 6, 3]. Several issues are under study, as learning DPPs, for instance through maximum likelihood estimation [28, 10], or sampling these processes. Here we will focus on the sampling question and we will only deal with a discrete and finite determinantal point process Y, defined by its kernel matrix K, a configuration particularly adapted to machine learning data sets.

The main algorithm to sample DPPs is a spectral algorithm [24]: it uses the eigendecomposition of K to sample Y. It is exact and in general quite fast. Yet, the computation of the eigenvalues of K may be very costly when dealing with large-scale data. That is why numerous algorithms have been conceived to bypass this issue. Some authors tried to design a sampling algorithm adapted to specific DPPs. For instance, it is possible to speed up the initial algorithm by assuming that K has a bounded rank [26, 15]. These authors use a dual representation of the kernel so that almost all the computations in the spectral algorithm are reduced. One can also deal with another class of DPPs associated to kernels K that can be decomposed in a sum of tractable matrices [14]. In this case, the sampling is much faster and the authors study the inference on these classes of DPPs. At last, Propp and Wilson [37] use Markov chains and the theory of coupling from the past to sample exactly particular DPPs: uniform spanning trees. Adapting Wilson's algorithm, Avena and Gaudillière [5] provide another algorithm to efficiently sample a parametrized DPP kernel associated to random spanning forests.

Another type of sampling algorithms is the class of approximate methods. Some authors approach the original DPP with a low rank matrix, either by random projections [27, 21] or using the Nystrom approximation [1]. The Monte Carlo Markov Chain

methods offer also nice approximate sampling algorithms for DPPs. It is possible to obtain satisfying convergence guarantees for particular DPPs; for instance, k-DPPs with fixed cardinality [4, 32] or projection DPPs [17]. Li et al. [33] even proposed a polynomial-time sampling algorithm for general DPPs, thus correcting the initial work of Kang [25]. These algorithms are commonly used as they save significant time but the price to pay is the lack of precision of the result.

As one can see, except the initial spectral algorithm, no algorithm allows for the exact sampling of a general DPP. The main contribution of this paper is to introduce such a general and exact algorithm, that does not involve the kernel eigendecomposition, to sample discrete DPPs. The proposed algorithm is a sequential thinning procedure that relies on two new results: (i) the explicit formulation of the marginals of any determinantal point process and (ii) the derivation of an adapted Bernoulli point process containing a given DPP. This algorithm was first presented in [29] and was, to our knowledge, the first exact sampling strategy without spectral decomposition. MATLAB and Python implementations of this algorithm (using the PyTorch library in the Python code) are available online (`https://www.math-info.univ-paris5.fr/~claunay/exact_sampling.html`) and hopefully soon in the repository created by Guillaume Gautier [18] gathering exact and approximate DPP sampling algorithms. Let us mention that three very recent preprints [36, 20, 13] also propose new algorithms to sample general DPPs without spectral decomposition. Poulson [36] presents factorization strategies of Hermitian and non-Hermitian DPP kernels to sample general determinantal point processes. As our algorithm, it heavily relies on Cholesky decomposition. Gillenwater and al. [20] use the dual representation of L-ensembles presented in [27] to construct a binary tree containing enough information on the kernel to sample DPPs in sublinear time. Dereziński et al. [13] apply a preprocessing step that preselects a portion of the points using a regularized DPP. Then, a usual DPP sampling is done on the selection. This is related to our thinning procedure of the initial set by a Bernoulli point process. However note that the authors report that the overall complexity of their sampling scheme is sublinear while ours is cubic due to Cholesky decomposition. Finally, in [8], Blaszczyszyn and Keeler present a similar procedure based on a continuous space: they use discrete determinantal point processes to thin a Poisson point process defined on that continuous space. The point process generated offers theoretical guarantees on

repulsion and is applied to fit network patterns.

The rest of the paper is organized as follows: in the next section, we present the general framework of determinantal point processes and the classic spectral algorithm. In Section 2, we provide an explicit formulation of the general marginals and pointwise conditional probabilities of any determinantal point process, from its kernel K. Using these formulations, we first introduce a "naive", exact but slow, sequential algorithm that relies on the Cholesky decomposition of the kernel K. In Section 3, using the thinning theory, we accelerate the previous algorithm and introduce a new exact sampling algorithm for DPPs that we call the sequential thinning algorithm. Its computational complexity is compared with that of the two previous algorithms. In Section 4, we display the results of some experiments comparing these three sampling algorithms and we describe the conditions under which the sequential thinning algorithm is more efficient than the spectral algorithm. Finally, we discuss and conclude on this algorithm.

1. DPPs and their Usual Sampling Method: the Spectral Algorithm

In the next sections, we will use the following notations. Let us consider a discrete finite set $\mathcal{Y} = \{1, \dots, N\}$. This set represents the space on which the point process is defined. In point process theory, it can be called the carrier space or state space. In this paper, we choose a machine learning term and refer to $\mathcal{Y}$ as the ground set. For $M \in \mathbb{R}^{N\times N}$ a matrix, we will denote by $M_{A\times B}$, $\forall A, B \subset \mathcal{Y}$, the matrix $(M(i,j))_{(i,j)\in A\times B}$ and the short notation $M_A = M_{A\times A}$. Suppose that K is a Hermitian positive semi-definite matrix of size $N\times N$, indexed by the elements of $\mathcal{Y}$, so that any of its eigenvalues is in $[0,1]$. A subset $Y \subset \mathcal{Y}$ is said to follow a DPP distribution of kernel K if,

$$\mathbb{P}\left(A \subset Y\right) = \det(K_A),\ \forall\, A \subset \mathcal{Y}.$$

The spectral algorithm is standard for drawing a determinantal point process. It relies on the eigendecompostition of its kernel K. It was first introduced by Hough et al. [24] and is also presented in a more detailed way by Scardicchio [40], Kulesza and Taskar [27] or Lavancier et al. [31]. It proceeds in 3 steps: the first step is the computation of the eigenvalues λ_j and the eigenvectors v^j of the matrix K. The second step consists in randomly selecting a set of eigenvectors according to N Bernoulli

variables of parameter λ_i, for $i = 1, \dots, N$. The third step is drawing sequentially the associated points using a Gram-Schmidt process.

Algorithm 1 The spectral sampling algorithm

1. Compute the orthonormal eigendecomposition (λ_j, v^j) of the matrix K.
2. Select a random set of eigenvectors: Draw a Bernoulli process $\mathbf{X} \in \{0,1\}^N$ with parameter $(\lambda_j)_j$. Denote by n the number Bernoulli samples equal to one, $\{\mathbf{X} = 1\} = \{j_1, \dots, j_n\}$. Define the matrix $V = \begin{pmatrix} v^{j_1} & v^{j_2} & \cdots & v^{j_n} \end{pmatrix} \in \mathbb{R}^{N \times n}$ and denote by $V_k \in \mathbb{R}^n$ the k-th line of V, for $k \in \mathcal{Y}$.
3. Return the sequence $Y = \{y_1, y_2, \dots, y_n\}$ sequentially drawn as follows:
   For $l = 1$ to n
   - Sample a point $y_l \in \mathcal{Y}$ from the discrete distribution,
   $$p_k^l = \frac{1}{n-l+1} \left(\|V_k\|^2 - \sum_{m=1}^{l-1} |\langle V_k, e_m \rangle|^2 \right), \forall k \in \mathcal{Y}.$$
   - If $l < n$, define $e_l = \frac{w_l}{\|w_l\|} \in \mathbb{R}^n$ where $w_l = V_{y_l} - \sum_{m=1}^{l-1} \langle V_{y_l}, e_m \rangle e_m$.

This algorithm is exact and relatively fast but it becomes slow when the size of the ground set grows. For a ground set of size N and a sample of size n, the third step costs $O(Nn^3)$ because of the Gram-Schmidt orthonormalisation. Tremblay et al. [43] propose to speed it up using optimized computations and they achieve the complexity $O(Nn^2)$ for this third step. Nevertheless, the eigendecomposition of the matrix K is the heaviest part of the algorithm, as it runs in time $O(N^3)$, and we will see in the numerical results that this first step represents in general more than 90% of the running time of the spectral algorithm. As nowadays the amount of data explodes, in practice the matrix K is very large so it seems relevant to try to avoid this costly operation. We compare the time complexities of the different algorithms presented in this paper at the end of Section 3. In the next section, we show that any DPP can be exactly sampled by a sequential algorithm that does not require the eigendecomposition of K.

2. Sequential Sampling Algorithm

Our goal is to build a competitive algorithm to sample DPPs that does not involve the eigendecomposition of the matrix K. To do so, we first develop a "naive" sequential sampling algorithm and subsequently, we will accelerate it using a thinning procedure, presented in Section 3.

2.1. Explicit General Marginal of a DPP

First, we need to specify the marginals and the conditional probabilities of any DPP. When $I - K$ is invertible, a formulation of the explicit marginals already exists [27], it implies to deal with a L-ensemble matrix L instead of the matrix K. However, this hypothesis is reductive: among others, it ignores the useful case of projection DPPs, when the eigenvalues of K are either 0 or 1. We show below that general marginals can easily be formulated from the associated kernel matrix K. For all $A \subset \mathcal{Y}$, we denote I^A the $N \times N$ matrix with 1 on its diagonal coefficients indexed by the elements of A, and 0 anywhere else. We also denote $|A|$ the cardinality of any subset $A \subset \mathcal{Y}$ and $\overline{A} \in \mathcal{Y}$ the complementary set of A in $\mathcal{Y}$.

Proposition 1. (Distribution of a DPP.) *For any $A \subset \mathcal{Y}$, we have*

$$\mathbb{P}(Y = A) = (-1)^{|A|} \det(I^{\overline{A}} - K).$$

Proof. We have that $\mathbb{P}(A \subset Y) = \sum_{B \supset A} \mathbb{P}(Y = B)$. Using the Möbius inversion formula (see Appendix A), for all $A \subset \mathcal{Y}$,

$$\begin{aligned}\mathbb{P}(Y = A) &= \sum_{B \supset A} (-1)^{|B \setminus A|} \mathbb{P}(B \subset Y) = (-1)^{|A|} \sum_{B \supset A} (-1)^{|B|} \det(K_B) \\ &= (-1)^{|A|} \sum_{B \supset A} \det((-K)_B)\end{aligned}$$

Furthermore, Kulesza and Taskar [27] state in Theorem 2.1 that for all $L \in \mathbb{R}^{N \times N}$, for all $A \subset \mathcal{Y}$, $\sum_{A \subset B \subset \mathcal{Y}} \det(L_B) = \det(I^{\overline{A}} + L)$. Then we obtain

$$\mathbb{P}(Y = A) = (-1)^{|A|} \det(I^{\overline{A}} - K).$$

□

We have by definition $\mathbb{P}(A \subset Y) = \det(K_A)$ for all A, and as a consequence $\mathbb{P}(B \cap Y = \emptyset) = \det((I-K)_B)$ for all B. The next proposition gives for any DPP the expression of the general marginal $\mathbb{P}(A \subset Y, B \cap Y = \emptyset)$, for any A, B disjoint subsets of $\mathcal{Y}$, using K. In what follows, H^B denotes the symmetric positive semi-definite matrix

$$H^B = K + K_{\mathcal{Y}\times B}((I-K)_B)^{-1}K_{B\times\mathcal{Y}}.$$

Theorem 1. (General Marginal of a DPP.) *Let $A, B \subset \mathcal{Y}$ be disjoint. If $\mathbb{P}(B \cap Y = \emptyset) = \det((I-K)_B) = 0$, then $\mathbb{P}(A \subset Y, B \cap Y = \emptyset) = 0$. Otherwise, the matrix $(I-K)_B$ is invertible and*

$$\mathbb{P}(A \subset Y, B \cap Y = \emptyset) = \det((I-K)_B)\det(H^B_A).$$

Proof. Let $A, B \subset \mathcal{Y}$ disjoint such that $\mathbb{P}(B \cap Y = \emptyset) \neq 0$. Using the previous proposition,

$$\mathbb{P}(A \subset Y, B \cap Y = \emptyset) = \sum_{A\subset C\subset\overline{B}} \mathbb{P}(Y = C) = \sum_{A\subset C\subset\overline{B}} (-1)^{|C|}\det(I^{\overline{C}} - K).$$

For any C such that $A \subset C \subset \overline{B}$, one has $B \subset \overline{C}$. Hence, by reordering the matrix coefficients, and using the Schur's determinant formula [23],

$$\begin{aligned}\det(I^{\overline{C}} - K) &= \det\begin{pmatrix}(I^{\overline{C}} - K)_B & (I^{\overline{C}} - K)_{B\times\overline{B}}\\ (I^{\overline{C}} - K)_{\overline{B}\times B} & (I^{\overline{C}} - K)_{\overline{B}}\end{pmatrix}\\ &= \det\begin{pmatrix}(I-K)_B & -K_{B\times\overline{B}}\\ -K_{\overline{B}\times B} & (I^{\overline{C}} - K)_{\overline{B}}\end{pmatrix}\\ &= \det((I-K)_B)\det((I^{\overline{C}} - H^B)_{\overline{B}}).\end{aligned}$$

Thus, $\mathbb{P}(A \subset Y, B \cap Y = \emptyset) = \det((I-K)_B)\sum_{A\subset C\subset\overline{B}} (-1)^{|C|}\det((I^{\overline{C}} - H^B)_{\overline{B}})$.
According to Theorem 2.1 in Kulesza and Taskar [27], for all $A \subset \overline{B}$,

$$\sum_{A\subset C\subset\overline{B}} \det(-H^B_C) = \det((I^{\overline{A}} - H^B)_{\overline{B}}).$$

Then, Möbius inversion formula ensures that, $\forall A \subset \overline{B}$,

$$\sum_{A\subset C\subset\overline{B}} (-1)^{|C\setminus A|}\det((I^{\overline{C}} - H^B)_{\overline{B}}) = \det(-H^B_A) = (-1)^{|A|}\det(H^B_A).$$

Hence, $\mathbb{P}(A \subset Y, B \cap Y = \emptyset) = \det((I-K)_B)\det(H^B_A)$. □

With this formula, we can explicitly formulate the pointwise conditional probabilities of any DPP.

Corollary 1. (Pointwise conditional probabilities of a DPP.) *Let $A, B \subset \mathcal{Y}$ be two disjoint sets such that $\mathbb{P}(A \subset Y,\ B \cap Y = \emptyset) \neq 0$, and let $k \notin A \cup B$. Then,*

$$\begin{aligned}\mathbb{P}(\{k\} \subset Y | A \subset Y,\ B \cap Y = \emptyset) &= \frac{\det(H^B_{A\cup\{k\}})}{\det(H^B_A)} \\ &= H^B(k,k) - H^B_{\{k\}\times A}(H^B_A)^{-1}H^B_{A\times\{k\}}.\end{aligned} \tag{1}$$

This is a straightforward application of the previous expression and the Schur determinant formula [23]. Note that these pointwise conditional probabilities are related to the Palm distribution of a point process [12] which characterizes the distribution of the point process under the condition that there is a point at some location $x \in \mathcal{Y}$. Shirai and Takahashi proved in [41] that DPPs on general spaces are closed under Palm distributions, in the sense that there exists a DPP kernel K^x such that the Palm measure associated to DPP(K) and x is a DPP defined on $\mathcal{Y}$ with kernel K^x. Borodin and Rains [9] also provide similar results on discrete spaces, using L-ensembles, that Kulesza and Taskar adapt in [27]. They condition the DPP not only on a subset included in the point process but also, similarly as Corollary 1, on a subset not included in the point process. As Shirai and Takahashi, they derive a formulation of the generated marginal kernel L.

Now, we have all the necessary expressions for the sequential sampling of a DPP.

2.2. Sequential Sampling Algorithm of a DPP

This sequential sampling algorithm simply consists in using Formula (1) and updating at each step the pointwise conditional probability, knowing the previous selected points. It is presented in Algorithm 2. We recall that this sequential algorithm is the first step toward developing a competitive sampling algorithm for DPPs: with this method, one doesn't need eigendecomposition anymore. The second step (Section 3) will be to reduce its computational cost.

Algorithm 2 Sequential sampling of a DPP with kernel K

- Initialization: $A \leftarrow \emptyset$, $B \leftarrow \emptyset$.
- For $k = 1$ to N:

1. Compute $H^B_{A\cup\{k\}} = K_{A\cup\{k\}} + K_{A\cup\{k\}\times B}((I-K)_B)^{-1}K_{B\times A\cup\{k\}}$.
2. Compute the probability p_k given by

$$p_k = \mathbb{P}\left(\{k\} \subset Y | A \subset Y,\ B \cap Y = \emptyset\right) = H^B(k,k) - H^B_{\{k\}\times A}(H^B_A)^{-1}H^B_{A\times\{k\}}.$$

3. With probability p_k, k is included, $A \leftarrow A \cup \{k\}$, otherwise $B \leftarrow B \cup \{k\}$.

- Return A.

The main operations of Algorithm 2 involve solving linear systems related to $(I - K)_B^{-1}$. Fortunately, here we can use the Cholesky factorization, which alleviates the computational cost. Suppose that T^B is the Cholesky factorization of $(I-K)_B$, that is, T^B is a lower triangular matrix such that $(I-K)_B = T^B(T^B)^*$ (where $(T^B)^*$ is the conjugate transpose of T^B). Then, denoting $J^B = (T^B)^{-1}K_{B\times A\cup\{k\}}$, one simply has $H^B_{A\cup\{k\}} = K_{A\cup\{k\}} + (J^B)^*J^B$.

Furthermore, at each iteration where B grows, the Cholesky decomposition $T^{B\cup\{k\}}$ of $(I-K)_{B\cup\{k\}}$ can be computed from T^B using standard Cholesky update operations, involving the resolution of only one linear system of size $|B|$. See Appendix B for the details of a typical Cholesky decomposition update.

In comparison with the spectral sampling algorithm of Hough et al. [24], one requires computations for each site of $\mathcal{Y}$, and not just one for each sampled point of Y. We will see at the end of Section 3 and in the experiments that it is not competitive.

3. Sequential Thinning Algorithm

In this section, we show that we can significantly decrease the number of steps and the running time of Algorithm 2: we propose to first sample a point process X containing Y, the desired DPP, and then make a sequential selection of the points of X to obtain Y. This procedure can be called a sequential thinning.

3.1. General Framework of Sequential Thinning

We first describe a general sufficient condition for which a target point process Y - it will be a determinantal point process in our case - can be obtained as a sequential thinning of a point process X. This is a discrete adaptation of the thinning procedure on the continuous line of Rolski and Szekli [38]. To do this, we will consider a coupling (X, Z) such that $Z \subset X$ will be a random selection of the points of X and that will have the same distribution as Y. From this point onward, we identify the set X with the vector of size N with 1 in the place of the elements of X and 0 elsewhere, and we use the notations $X_{1:k}$ to denote the vector $(X_1, \dots, X_k)$ and $0_{1:k}$ to denote the null vector of size k. We want to define the random vector $(X_1, Z_1, X_2, Z_2, \dots, X_N, Z_N) \in \mathbb{R}^{2N}$ with the following conditional distributions for X_k and Z_k:

$$\begin{cases} \mathbb{P}(X_k = 1 | Z_{1:k-1} = z_{1:k-1}, X_{1:k-1} = x_{1:k-1}) = \mathbb{P}(X_k = 1 | X_{1:k-1} = x_{1:k-1}) \\ \\ \mathbb{P}(Z_k = 1 | Z_{1:k-1} = z_{1:k-1}, X_{1:k} = x_{1:k}) = \mathbf{1}_{\{x_k = 1\}} \dfrac{\mathbb{P}(Y_k = 1 | Y_{1:k-1} = z_{1:k-1})}{\mathbb{P}(X_k = 1 | X_{1:k-1} = x_{1:k-1})}. \end{cases} \tag{2}$$

Proposition 2. (Sequential thinning.) *Assume that X, Y, Z are discrete point processes on $\mathcal{Y}$ that satisfy for all $k \in \{1, \dots, N\}$, and all z, $x \in \{0,1\}^N$,*

$$\begin{gathered} \mathbb{P}(Z_{1:k-1} = z_{1:k-1}, X_{1:k-1} = x_{1:k-1}) > 0 \\ \textit{implies} \\ \mathbb{P}(Y_k = 1 | Y_{1:k-1} = z_{1:k-1}) \leq \mathbb{P}(X_k = 1 | X_{1:k-1} = x_{1:k-1}). \end{gathered} \tag{3}$$

Then, it is possible to choose (X, Z) in such a way that (2) is satisfied. In that case, we have that Z is a thinning of X, that is $Z \subset X$, and Z has the same distribution as Y.

Proof. Let us first discuss the definition of the coupling (X, Z). With the conditions (3), the ratios defining the conditional probabilities of Equation (2) are ensured to be between 0 and 1 (if the conditional events have non zero probabilities). Hence the conditional probabilities allows us to construct sequentially the distribution of the random vector $(X_1, Z_1, X_2, Z_2, \dots, X_N, Z_N)$ of length $2N$, and thus the coupling is well-defined. Furthermore, as Equation (2) is satisfied, $Z_k = 1$ only if $X_k = 1$, so one has $Z \subset X$.

Let us now show that Z has the same distribution as Y. By complementarity of the events $\{Z_k = 0\}$ and $\{Z_k = 1\}$, it is enough to show that for all $k \in \{1, \dots, N\}$, and $z_1, \dots, z_{k-1}$ such that $\mathbb{P}(Z_{1:k-1} = z_{1:k-1}) > 0$,

$$\mathbb{P}(Z_k = 1 | Z_{1:k-1} = z_{1:k-1}) = \mathbb{P}(Y_k = 1 | Y_{1:k-1} = z_{1:k-1}). \tag{4}$$

Let $k \in \{1, \dots, N\}$, $(z_{1:k-1}, x_{1:k-1}) \in \{0,1\}^{2(k-1)}$, such that $\mathbb{P}(Z_{1:k-1} = z_{1:k-1}, X_{1:k-1} = x_{1:k-1}) > 0$. Since $Z \subset X$, $\{Z_k = 1\} = \{Z_k = 1, X_k = 1\}$. Suppose first that $\mathbb{P}(X_k = 1 | X_1 = x_1, \dots, X_{k-1} = x_{k-1}) \neq 0$. Then

$$\begin{aligned}
&\mathbb{P}(Z_k = 1 | Z_{1:k-1} = z_{1:k-1}, X_{1:k-1} = x_{1:k-1}) \\
&= \mathbb{P}(Z_k = 1, X_k = 1 | Z_{1:k-1} = z_{1:k-1}, X_{1:k-1} = x_{1:k-1}) \\
&= \begin{array}{l} \mathbb{P}(Z_k = 1 | Z_{1:k-1} = z_{1:k-1}, X_{1:k-1} = x_{1:k-1}, X_k = 1) \\ \quad \times \mathbb{P}(X_k = 1 | Z_{1:k-1} = z_{1:k-1}, X_{1:k-1} = x_{1:k-1}) \end{array} \\
&= \mathbb{P}(Y_k = 1 | Y_{1:k-1} = z_{1:k-1}), \text{ by Equations (2).}
\end{aligned}$$

If $\mathbb{P}(X_k = 1 | X_{1:k-1} = x_{1:k-1}) = 0$, then $\mathbb{P}(Z_k = 1 | Z_{1:k-1} = z_{1:k-1}, X_{1:k-1} = x_{1:k-1}) = 0$ and using (3), $\mathbb{P}(Y_k = 1 | Y_{1:k} = z_{1:k}) = 0$. Hence the identity

$$\mathbb{P}(Z_k = 1 | Z_{1:k-1} = z_{1:k-1}, X_{1:k-1} = x_{1:k-1}) = \mathbb{P}(Y_k = 1 | Y_{1:k-1} = z_{1:k-1})$$

is always valid. Since the values $x_1, \dots, x_{k-1}$ do not influence this conditional probability, one can conclude that given $(Z_1, \dots, Z_{k-1})$, Z_k is independent of $X_1, \dots, X_{k-1}$, and thus (4) is true. □

The characterization of the thinning defined here allows both extreme cases: there can be no pre-selection of points by X, meaning that $X = \mathcal{Y}$ and that the DPP Y is sampled by Algorithm 2, or there can be no thinning at all, meaning that the final process Y can be equal to the dominating process X. Regarding sampling acceleration, a good dominating process X must be sampled quickly and with a cardinality as close as possible to $|Y|$.

3.2. Sequential Thinning Algorithm for DPPs

In this section, we use the sequential thinning approach, where Y is a DPP of kernel K on the ground set $\mathcal{Y}$, and X is a Bernoulli point process (BPP). BPPs are the fastest and easiest point processes to sample. X is a Bernoulli process if the components

of the vector $(X_1, \dots, X_N)$ are independent. Its distribution is determined by the probability of occurrence of each point k, that we denote by $q_k = \mathbb{P}(X_k = 1)$. Due to the independence property, the conditions (3) simplifies to

$$\begin{gathered}\mathbb{P}(Z_{1:k-1} = z_{1:k-1}, X_{1:k-1} = x_{1:k-1}) > 0 \\ \text{implies} \\ \mathbb{P}(Y_k = 1 | Y_{1:k-1} = z_{1:k-1}) \leq q_k.\end{gathered}$$

The second inequality does not depend on x, hence it must be valid as soon as there exists a vector x such that $\mathbb{P}(Z_{1:k-1} = z_{1:k-1}, X_{1:k-1} = x_{1:k-1}) > 0$, that is, as soon as $\mathbb{P}(Z_{1:k-1} = z_{1:k-1}) > 0$. Since we want Z to have the same distribution as Y, we finally obtain the conditions

$$\forall y \in \{0,1\}^N,\ \mathbb{P}(Y_{1:k-1} = y_{1:k-1}) > 0 \text{ implies } \mathbb{P}(Y_k = 1 | Y_{1:k-1} = y_{1:k-1}) \leq q_k.$$

Ideally, we want the q_k to be as small as possible to ensure that the cardinality of X is as small as possible. So we look for the optimal values q_k^*, that is,

$$q_k^* = \max_{\substack{(y_{1:k-1}) \in \{0,1\}^{k-1} \text{ s.t.} \\ \mathbb{P}(Y_{1:k-1} = y_{1:k-1}) > 0}} \mathbb{P}(Y_k = 1 | Y_{1:k-1} = y_{1:k-1}).$$

A priori, computing q_k^* would raise combinatorial issues. However, due to the repulsive nature of DPPs, we have the following proposition.

Proposition 3. *Let $A, B \subset \mathcal{Y}$ be two disjoint sets such that $\mathbb{P}(A \subset Y,\ B \cap Y = \emptyset) \neq 0$, and let $k \neq l \in \overline{A \cup B}$. If $\mathbb{P}(A \cup \{l\} \subset Y,\ B \cap Y = \emptyset) > 0$, then*

$$\mathbb{P}(\{k\} \subset Y | A \cup \{l\} \subset Y,\ B \cap Y = \emptyset) \leq \mathbb{P}(\{k\} \subset Y | A \subset Y,\ B \cap Y = \emptyset).$$

If $\mathbb{P}(A \subset Y,\ (B \cup \{l\}) \cap Y = \emptyset) > 0$, then

$$\mathbb{P}(\{k\} \subset Y | A \subset Y,\ (B \cup \{l\}) \cap Y = \emptyset) \geq \mathbb{P}(\{k\} \subset Y | A \subset Y,\ B \cap Y = \emptyset).$$

Consequently, for all $k \in \mathcal{Y}$, if $y_{1:k-1} \leq z_{1:k-1}$ (where $\leq$ stands for the inclusion partial order) are two states for $Y_{1:k-1}$, then

$$\mathbb{P}(Y_k = 1 | Y_{1:k-1} = y_{1:k-1}) \geq \mathbb{P}(Y_k = 1 | Y_{1:k-1} = z_{1:k-1}).$$

In particular, $\forall k \in \{1, \dots, N\}$, *if* $\mathbb{P}(Y_{1:k-1} = 0_{1:k-1}) > 0$ *then*

$$\begin{aligned} q_k^* &= \mathbb{P}(Y_k = 1 | Y_{1:k-1} = 0_{1:k-1}) \\ &= K(k,k) + K_{k \times \{1:k-1\}}((I-K)_{\{1:k-1\}})^{-1} K_{\{1:k-1\} \times k}. \end{aligned}$$

Proof. Recall that by Proposition 1, $P(\{k\} \subset Y | A \subset Y,\ B \cap Y = \emptyset) = H^B(k,k) - H^B_{\{k\} \times A}(H^B_A)^{-1} H^B_{A \times \{k\}}$. Let $l \notin A \cup B \cup \{k\}$. Consider T^B the Cholesky decomposition of the matrix H^B obtained with the following ordering the coefficients: A, l, the remaining coefficients of $\mathcal{Y} \setminus (A \cup \{l\})$. Then, the restriction T^B_A is the Cholesky decomposition (of the reordered) H^B_A and thus

$$\begin{aligned} H^B_{\{k\} \times A}(H^B_A)^{-1} H^B_{A \times \{k\}} &= H^B_{\{k\} \times A}(T^B_A (T^B_A)^*)^{-1} H^B_{A \times \{k\}} \\ &= \|(T^B_A)^{-1} H^B_{A \times \{k\}}\|_2^2. \end{aligned}$$

Similarly,

$$H^B_{\{k\} \times A \cup \{l\}}(H^B_{A \cup \{l\}})^{-1} H^B_{A \cup \{l\} \times \{k\}} = \|(T^B_{A \cup \{l\}})^{-1} H^B_{A \cup \{l\} \times \{k\}}\|_2^2.$$

Now note that solving the triangular system with $b = (T^B_{A \cup \{l\}})^{-1} H^B_{A \cup \{l\} \times \{k\}}$ amounts solving the triangular system with $(T^B_A)^{-1} H^B_{A \times \{k\}}$ and an additional line at the bottom. Hence, one has $\|b\|_2^2 \geq \|(T^B_A)^{-1} H^B_{A \times \{k\}}\|_2^2$.

Consequently, provided that $\mathbb{P}(A \cup \{l\} \subset Y,\ B \cap Y = \emptyset) > 0$,

$$\mathbb{P}(\{k\} \subset Y | A \cup \{l\} \subset Y,\ B \cap Y = \emptyset) \leq \mathbb{P}(\{k\} \subset Y | A \subset Y,\ B \cap Y = \emptyset).$$

The second inequality is obtained by complementarity in applying the above inequality to the DPP $\overline{Y}$ with $B \cup \{l\} \subset \overline{Y}$ and $A \cap \overline{Y} = \emptyset$. □

As a consequence, an admissible choice for the distribution of the Bernoulli process is

$$q_k = \begin{cases} \mathbb{P}(Y_k = 1 | Y_{1:k-1} = 0_{1:k-1}) & \text{if } \mathbb{P}(Y_{1:k-1} = 0_{1:k-1}) > 0, \\ 1 & \text{otherwise.} \end{cases} \tag{5}$$

Note that if for some index k, $\mathbb{P}(Y_{1:k-1} = 0_{1:k-1}) > 0$ is not satisfied, then for all the subsequent indexes $l \geq k$, $q_l = 1$, that is the Bernoulli process becomes degenerate and contains all the points after k. In the remaining of this section, X will denote a Bernoulli process with probabilities (q_k) given by (5).

As discussed in the previous section, in addition to being easily simulated, one would like the cardinality of X to be close to the one of Y, the final sample. The next proposition shows that this is verified if all the eigenvalues of K are strictly less than 1.

Proposition 4. ($|X|$ is proportional to $|Y|$.) *Suppose that* $P(Y = \emptyset) = \det(I-K) > 0$ *and denote by* $\lambda_{\max}(K) \in [0, 1)$ *the maximal eigenvalue of* K*. Then,*

$$\mathbb{E}(|X|) \leq \left(1 + \frac{\lambda_{\max}(K)}{2\left(1 - \lambda_{\max}(K)\right)}\right) \mathbb{E}(|Y|). \tag{6}$$

Proof. We know that $q_k = K(k,k) + K_{\{k\}\times\{1:k-1\}}((I-K)_{\{1:k-1\}})^{-1}K_{\{1:k-1\}\times\{k\}}$, by Proposition 1. Since

$$\|((I-K)_{\{1:k-1\}})^{-1}\|_{\mathcal{M}_{k-1}(\mathbb{C})} = \tfrac{1}{1-\lambda_{\max}(K_{\{1:k-1\}})}$$

and $\lambda_{\max}(K_{\{1:k-1\}}) \leq \lambda_{\max}(K)$, one has

$$K_{\{k\}\times\{1:k-1\}}((I-K)_{\{1:k-1\}})^{-1}K_{\{1:k-1\}\times\{k\}} \leq \tfrac{1}{1-\lambda_{\max}(K)}\|K_{\{1:k-1\}\times\{k\}}\|_2^2.$$

Summing all these inequalities gives

$$\mathbb{E}(|X|) \leq \operatorname{Tr}(K) + \tfrac{1}{1-\lambda_{\max}(K)} \sum_{k=1}^{N} \|K_{\{1:k-1\}\times\{k\}}\|_2^2.$$

The last term is the Frobenius norm of the upper triangular part of K, hence in can be bounded by $\frac{1}{2}\|K\|_F^2 = \frac{1}{2}\sum_{j=1}^N \lambda_j(K)^2$. Since $\lambda_j(K)^2 \leq \lambda_j(K)\lambda_{\max}(K)$, $\sum_{j=1}^N \lambda_j(K)^2 \leq \lambda_{\max}(K)\operatorname{Tr}(K) = \lambda_{\max}(K)\mathbb{E}(|Y|)$. □

We can now introduce the final sampling algorithm that we call sequential thinning algorithm (Algorithm 3). It presents the different steps of our sequential thinning algorithm to sample a DPP of kernel K. The first step is a preprocess that must be done only once for a given matrix K. Step 2 is trivial and fast. The critical point is to sequentially compute the conditional probabilities $p_k = \mathbb{P}(\{k\} \subset Y | A \subset Y,\ B \cap Y = \emptyset)$ for each point of X. Recall that in Algorithm 2 we use a Cholesky decomposition of the matrix $(I-K)_B$ which is updated by adding a line each time a point is added in B. Here, the inverse of the matrix $(I-K)_B$ is only needed when visiting a point $k \in X$, so one updates the Cholesky decomposition by a single block, where the new block corresponds to all indices added to B in one iteration (see

Algorithm 3 Sequential thinning algorithm of a DPP with kernel K

1. Compute sequentially the probabilities $\mathbb{P}(X_k = 1) = q_k$ of the Bernoulli process X:

- Compute the Cholesky decomposition T of the matrix $I - K$.
- For $k = 1$ to N:
  - If $q_{k-1} < 1$ (with the convention $q_0 = 0$),

$$q_k = K(k,k) + \|T^{-1}_{\{1,\dots,k-1\}} K_{\{1,\dots,k-1\}\times\{k\}}\|_2^2.$$

  - Else, $q_k = 1$.

2. Draw the Bernoulli process X. Let $m = |X|$ and $k_1 < k_2 < \cdots < k_m$ be the points of X.

3. Apply the sequential thinning to the points of X:

- Attempt to add sequentially each point of X to Y:
  Initialize $A \leftarrow \emptyset$ and $B \leftarrow \{1, \dots, k_1 - 1\}$.
  For $j = 1$ to m
  - If $j > 1$, $B \leftarrow B \cup \{k_{j-1} + 1, \dots, k_j - 1\}$.
  - Compute the conditional probability $p_{k_j} = \mathbb{P}(\{k_j\} \subset Y | A \subset Y,\ B \cap Y = \emptyset)$ (see Formula (1)):
    - Update T^B the Cholesky decomposition of $(I - K)_B$ (see Appendix B).
    - Compute $J^B = (T^B)^{-1} K_{B\times A\cup\{k_j\}}$.
    - Compute $H^B_{A\cup\{k\}} = K_{A\cup\{k_j\}} + (J^B)^t J^B$.
    - Compute $p_{k_j} = H^B(k_j, k_j) - H^B_{\{k_j\}\times A}(H^B_A)^{-1} H^B_{A\times\{k_j\}}$.
  - Add k_j to A with probability $\frac{p_{k_j}}{q_{k_j}}$ or to B otherwise.
- Return A.

Appendix B). The MATLAB implementation used for the experiments is available online (`https://www.math-info.univ-paris5.fr/~claunay/exact_sampling.html`), together with a Python version of this code, using the PyTorch library. Note that, very

recently, Guillaume Gautier [16] proposed an alternative computation of the Bernoulli probabilities, that generate the dominating point process in the first step of Algorithm 3, so that it only requires the diagonal coefficients of the Cholesky decomposition T of $I - K$.

3.3. Computational Complexity

Recall that the size of the ground set $\mathcal{Y}$ is N and the size of the final sample is $|Y| = n$. Both algorithms introduced in this paper have running complexities of order $O(N^3)$, as the spectral algorithm. Yet, if we get into the details, the most expensive task in the spectral algorithm is the computation of the eigenvalues and the eigenvectors of the kernel K. As this matrix is Hermitian, the common routine to do so is the reduction of K to some tridiagonal matrix to which the QR decomposition is applied, meaning that it is decomposed into the product of an orthogonal matrix and an upper triangular matrix. When N is large, the total number of operations is approximately $\frac{4}{3}N^3$ [42]. In Algorithms 2 and 3, one of the most expensive operations is the Cholesky decomposition of several matrices. We recall that the Cholesky decomposition of a matrix of size $N \times N$ costs approximately $\frac{1}{3}N^3$ computations, when N is large [34]. Concerning the sequential algorithm 2, at each iteration k, the number of operations needed is of order $|B|^2|A| + |B||A|^2 + |A|^3$, where $|A|$ is the number of selected points at step k so it's lower than n, and $|B|$ the number of unselected points, bounded by k. Then, when N tends to infinity, the total number of operations in Algorithm 2 is lower than $\frac{n}{3}N^3 + \frac{n^2}{2}N^2 + n^3N$ or $O(nN^3)$, as in general $n \ll N$. Concerning Algorithm 3, the sequential thinning from X, coming from Algorithm 2, costs $O(n|X|^3)$. Recall that $|X|$ is proportional to $|Y| = n$ when the eigenvalues of K are smaller than 1 (see Equation (6)) so this step costs $O(n^4)$. Then, the Cholesky decomposition of $I - K$ is the most expensive operation in Algorithm 3 as it costs approximately $\frac{1}{3}N^3$. In this case, the overall running complexity of the sequential thinning algorithm is of order $\frac{1}{3}N^3$, which is 4 times less than the spectral algorithm. When some eigenvalues of K are equal to 1, Equation (6) does not hold anymore so, in that case, the running complexity of Algorithm 3 is only bounded by $O(nN^3)$.

We will retrieve this experimentally as, depending on the application or on the kernel K, this Algorithm 3 is able to speed up the sampling of DPPs. Note that in

the previous computations, we have not taken into account the possible parallelization of the sequential thinning algorithm. As a matter of fact, the Cholesky decomposition is parallelizable [19]. Incorporating this parallel computations would probably speed up the sequential thinning algorithm, since the Cholesky decomposition of $I - K$ is the most expensive operation when the expected cardinality $|Y|$ is low. The last part of the algorithm, the thinning procedure, operates sequentially, so it is not parallelizable. These comments on the complexity and running times highly depends on the implementation, on the choice of the programming language and speed up strategies, so they mainly serve as an illustration.

4. Experiments

4.1. DPP models for runtime tests

In the following section, we use the common notation of L-ensembles, with matrix $L = K(I - K)^{-1}$. We present the results using four different kernels:

(a) A random kernel: $K = Q^{-1}DQ$, where D is a diagonal matrix with uniformly distributed random values in $(0, 1)$ and Q an unitary matrix created from the QR decomposition of a random matrix.

(b) A discrete analog to the Ginibre kernel: $K = L(I + L)^{-1}$ with for all $x_1, x_2 \in \mathcal{Y} = \{1, \dots, N\}$,

$$L(x_1, x_2) = \frac{1}{\pi} e^{-\frac{1}{2}(|x_1|^2 + |x_2|^2) + x_1 x_2},$$

(c) A patch-based kernel: Let u be a discrete image and $\mathcal{Y} = \mathcal{P}$ a subset of all its patches, i.e. square sub-images of size $w \times w$ in u. Define $K = L(I + L)^{-1}$ where for all $P_1, P_2 \in \mathcal{P}$,

$$L(P_1, P_2) = \exp\left(-\frac{\|P_1 - P_2\|_2^2}{s^2}\right)$$

where $s > 0$ is called the bandwidth or scale parameter. We will detail the definition and the use of this kernel in Section 4.3.

(d) A projection kernel: $K = Q^{-1}DQ$, where D is a diagonal matrix with the n first coefficients equal to 1, the others, equal to 0, and Q is a random unitary matrix

as for model (a).

It is often essential to control the expected cardinality of the point process. For case (d) the cardinality is fixed to n. For the three other cases, we use a procedure similar to the one developed in [7]. Recall that if $Y \sim \mathrm{DPP}(K)$ and $K = L(I+L)^{-1}$, $\mathbb{E}(|Y|) = \mathrm{tr}(K) = \sum_{i\in\mathcal{Y}} \lambda_i = \sum_{i\in\mathcal{Y}} \frac{\mu_i}{1+\mu_i}$, where $(\lambda_i)_{i\in\mathcal{Y}}$ are the eigenvalues of K and $(\mu_i)_{i\in\mathcal{Y}}$ are the eigenvalues of L [24, 27]. Given an initial matrix $L = K(I-K)^{-1}$ and a desired expected cardinality $\mathbb{E}(|Y|) = n$, we run a binary search algorithm to find $\alpha > 0$ such that $\sum_{i\in\mathcal{Y}} \frac{\alpha\mu_i}{1+\alpha\mu_i} = n$. Then, we use the kernels $L_\alpha = \alpha L$ and $K_\alpha = L_\alpha(I+L_\alpha)^{-1}$.

4.2. Runtimes

For the following experiments, we ran the algorithms on a laptop HP Intel(R) Core(TM) i7-6600U CPU and we use the software MATLAB R2018b. Note that the computational time results depend on the programming language and the use of optimized functions by the software. Thus, the following numerical results are mainly indicative.

First, let us compare the sequential thinning algorithm (Algorithm 3) presented here with the two main sampling algorithms: the classic spectral algorithm (Algorithm 1) and the "naive" sequential algorithm (Algorithm 2). Figure 1 presents the running times of the three algorithms as a function of the total number of points of the ground set. Here, we have chosen a patch-based kernel (c). The expected cardinality $\mathbb{E}(|Y|)$ is constant, equal to 20. As foreseen, the sequential algorithm (Algorithm 2) is far slower than the two others. Whatever the chosen kernel and the expected cardinality of the DPP, this algorithm is not competitive. Note that the sequential thinning algorithm uses this sequential method after sampling the particular Bernoulli process. But we will see that this first dominating step can be very efficient and lead to a relatively fast algorithm.

From now on, we restrict the comparison to the spectral and the sequential thinning algorithms (Algorithms 1 and 3). We present in Figure 2 the running times of these algorithms as a function of the size of $|\mathcal{Y}|$ in various situations. The first row shows the

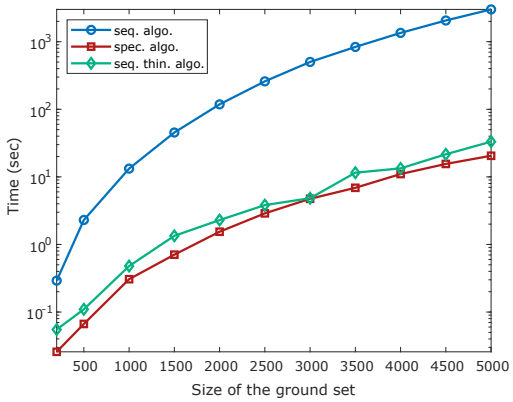


FIGURE 1: Running times of the 3 studied algorithms in function of the size of the ground set, using a patch-based kernel.

running times when the expectation of the number of sampled point $\mathbb{E}(|Y|)$ is equal to 4% of the size of $\mathcal{Y}$: it increases as the total number of points increases. In this case, we can see that whatever the chosen kernel, the spectral algorithm is faster as the complexity of sequential part of Algorithm 3 depends on the size $|X|$ that also grows. On the second row, as $|\mathcal{Y}|$ grows, $\mathbb{E}(|Y|)$ is fixed to 20. Except for the right-hand-side kernel, we are in the configuration where $|X|$ stays proportional to $|Y|$, then the Bernoulli step of Algorithm 3 is very efficient and this sequential thinning algorithm becomes competitive with the spectral algorithm. For these general kernels, we observe that the sequential thinning algorithm can be as fast as the spectral algorithm, and even faster, when the expected cardinality of the sample is small compared to the size of the ground set. The question is: when and up to which expected cardinality is Algorithm 3 faster?

Figure 3 displays the running times of both algorithms in function of the expected cardinality of the sample when the size of the ground set is constant, equal to 5000 points. Notice that, concerning the three left-hand-side general kernels with no eigenvalue equal to one, the sequential thinning algorithm is faster under a certain expected number of points -which depends on the kernel. For instance, when the kernel is randomly defined and the range of desired points to sample is below 25, it is relevant to use this algorithm. To conclude, when the eigenvalues of the kernel are below one, Algorithm 3 seems relevant for large data sets but small samples. This case is quite

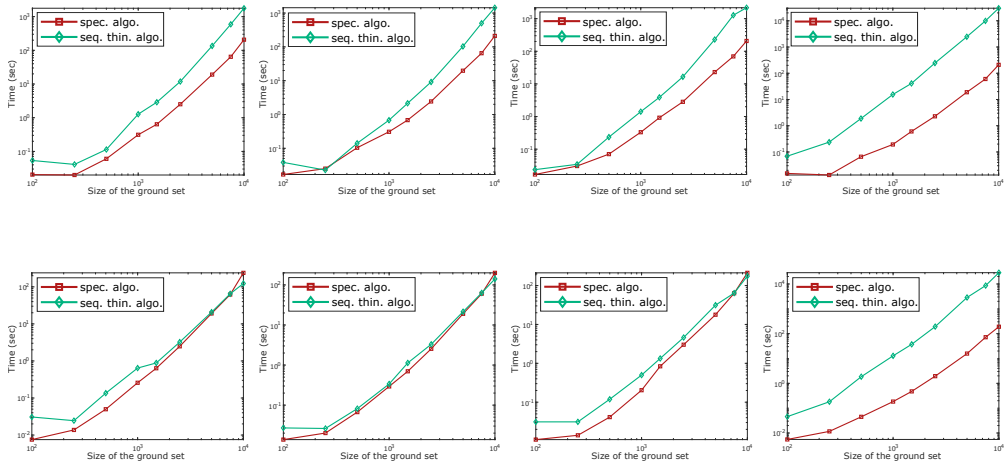


FIGURE 2: Running times in log-scale of the spectral and the sequential thinning algorithms as a function of the size of the ground set $|\mathcal{Y}|$, using "classic" DPP kernels. From left to right: a random kernel, a Ginibre-like kernel, a patch-based kernel and a projection kernel. On the first row, the expectation of the number of sampled points is set to 4% of $|\mathcal{Y}|$ and on the second row, $\mathbb{E}(|Y|)$ is constant, equal to 20.

common, for instance to summarize a text, to work only with representative points in clusters or to denoise an image with a patch-based method.

The projection kernel (when the eigenvalues of K are either 0 or 1) is, as expected, a complicated case. Figure 2 (bottom, right) shows that our algorithm is not competitive when using this kernel. Indeed, the cardinality of the dominating Bernoulli process X can be very large. In this case, the bound in Equation (6) isn't valid (and even tends

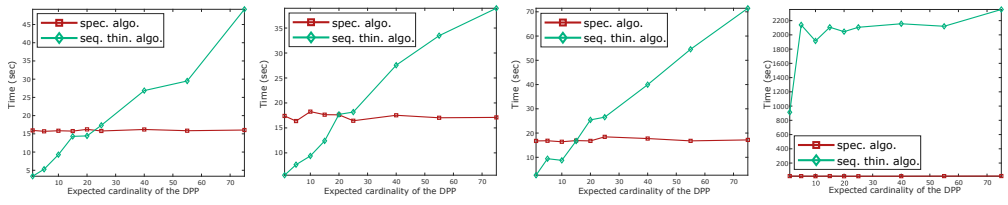
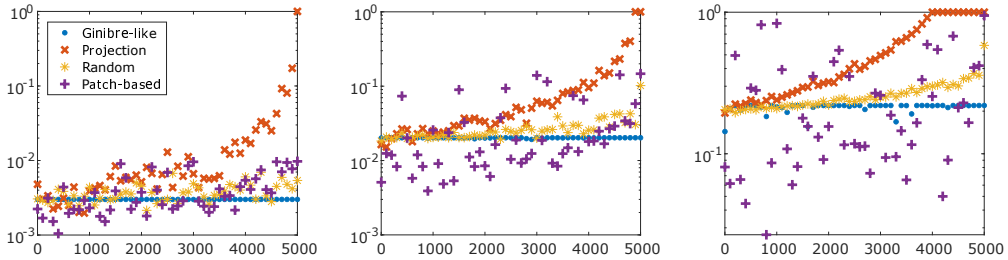


FIGURE 3: Running times of the spectral and sequential thinning algorithms in function of the expected cardinality of the process. From left to right, using a random kernel, a Ginibre-like kernel, the patch-based kernel and a projection kernel. The size of the ground set is fixed to 5000 in all examples.



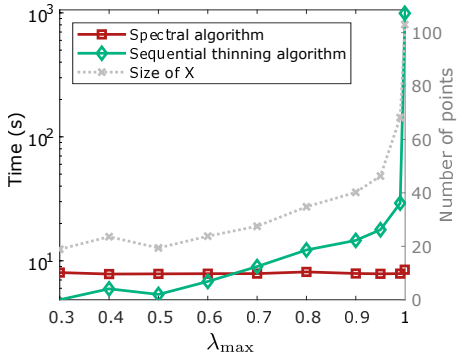
(a) $\mathbb{E}(|Y|) = 15$ (b) $\mathbb{E}(|Y|) = 100$ (c) $\mathbb{E}(|Y|) = 1000$

FIGURE 4: Behavior of the Bernoulli probabilities q_k, $k \in \{1, \dots, N\}$, for the kernels presented in Section 4.1, considering a ground set of $N = 5000$ elements and varying the expected cardinality of the DPP, $\mathbb{E}(|Y|) = 15, 100, 1000$.

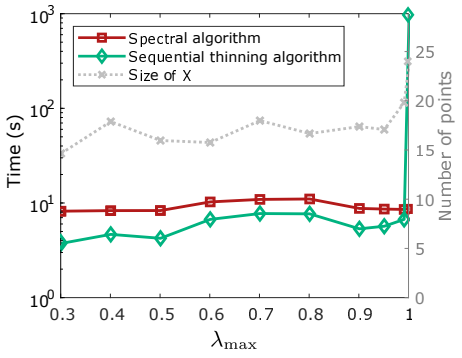
to infinity) as $\lambda_{\max} = 1$, and we necessarily reach the degenerated case when, after some index k, all the Bernoulli probabilities $q_l, l \geq k$, are equal to 1. Then the second part of the sequential thinning algorithm -the sequential sampling part- is done on a larger set which significantly increases the running time of our algorithm. Figure 3 confirms this observation as in that configuration, the sequential thinning algorithm is never the fastest.

Figure 4 illustrates how efficient the first step of Algorithm 3 can be to reduce the size of the initial set $\mathcal{Y}$. It displays Bernoulli probabilities $q_k, k \in \{1, \dots, N\}$ (Equation 5) associated to the previous kernels, for different expected cardinality $\mathbb{E}(|Y|)$. Observe that the probabilities are overall higher for a projection kernel. For such a kernel, we know that they necessarily reach the value 1, at the latest from the item $k = \mathbb{E}(|Y|)$. Indeed projection DPPs have a fixed cardinality (equal to $\mathbb{E}(|Y|)$) and q_k computes the probability to select the item k given that no other item has been selected yet. Notice that in general, considering the other kernels, the degenerated value $q_k = 1$ is rarely reached, even though in our experiments, the Bernoulli probabilities associated to the patch kernel (c) are sometimes close to one, when the expected size of the sample is $\mathbb{E}(|Y|) = 1000$. On the opposite, the Bernoulli probabilities associated to the Ginibre-like kernel remain rather close to a uniform distribution.

In order to understand more precisely to what extent high eigenvalues penalize the efficiency of the sequential thinning algorithm (Algorithm 3), Figure 5 compares its running times with that of the spectral algorithm (Algorithm 1) in function of the



(a) m eigenvalues equal to $\lambda_{\max}$ and $N-m$ zero eigenvalues.



(b) N random eigenvalues between 0 and $\lambda_{\max}$.

FIGURE 5: Running times of the spectral and sequential thinning algorithms (Algorithm 1 and 3) in function of $\lambda_{\max}$. The size of the Bernoulli process X is also displayed in light grey (right axis). Here, $|\mathcal{Y}| = 5000$ and $\mathbb{E}(|Y|) = 15$.

eigenvalues of the kernel K. For these experiments, we consider a ground set of size $|\mathcal{Y}| = 5000$ items and an expected cardinality equal to 15. In the first case (a), the eigenvalues are either equal to 0 or to $\lambda_{\max}$, whith m non-zero eigenvalues so that $m\lambda_{\max} = 15$. It shows that above a certain $\lambda_{\max}$ ($\simeq 0.65$), the sequential thinning algorithm is not the fastest anymore. In particular, when $\lambda_{\max} = 1$, the running time takes off. In the second case (b), the eigenvalues (λ_k) are randomly distributed between 0 and $\lambda_{\max}$ so that $\sum_k \lambda_k = 15$. In practice, $(N-1)$ eigenvalues are exponentially distributed, with expectation $\frac{15-\lambda_{\max}}{N-1}$, and the last eigenvalue is set to $\lambda_{\max}$. In this case, the sequential thinning algorithm remains faster than the spectral algorithm, even with high values of $\lambda_{\max}$, except when $\lambda_{\max} = 1$. This can be explained by the fact that, by construction of this kernel, most of the eigenvalues are very small. The average size of the Bernoulli process generated (light grey, right axes) also illustrates the influence of the eigenvalues.

Table 1 presents the individual weight of the main steps of the three algorithms. Concerning the sequential algorithm, logically, the matrix inversion is the heaviest part taking 74.25% of the global running time. These proportions remain the same when the expected number of points n grows. The main operation of the spectral algorithm is by far the eigendecomposition of the matrix K, counting for 83% of the

Algorithms	Steps	Expected cardinality	
		4% of $\lvert\mathcal{Y}\rvert$	Constant (20)
Sequential	Matrix inversion	74.25%	72.71%
	Cholesky computation	22.96%	17.82%
Spectral	Eigendecomposition	83.34%	94.24%
	Sequential sampling	14.77%	4.95%
Sequential thinning	Preprocess to define q	10.07%	13.43%
	Sequential sampling	89.39%	86.53%

TABLE 1: Detailed running times of the sequential, spectral and sequential thinning algorithms for varying ground sets $\mathcal{Y}$ with $|\mathcal{Y}| \in [100, 5000]$ using a patch-based kernel.

global running time, when the expectation of the number of points to sample evolves with the size of $\mathcal{Y}$. Finally, the sequential sampling is the heaviest step of the sequential thinning algorithm. We have already mentioned that the thinning is very fast and that it produces a point process with a cardinality as close as possible to the final DPP. When the expected cardinality is low, the number of selected points by the thinning process is low too, so the sequential sampling part remains bounded (86.53% when the expected cardinality $\mathbb{E}(|Y|)$ is constant). On the contrary, when $\mathbb{E}(|Y|)$ grows, the number of points selected by the dominated process rises as well so the running time of this step is growing (with a mean of 89.39%). As seen before, the global running time of the sequential thinning algorithm really depends on how good the domination is.

Thus, the main case when this sequential thinning algorithm (Algorithm 3) fails to compete with the spectral algorithm (Algorithm 1) is when the eigenvalues of the kernel are equal or very close to 1. This algorithm improves the sampling running times when the target size of the sample is very low (below 25 in our experiments).

In cases when multiple samples of the same DPP have to be drawn, the eigendecomposition of K can be stored and the spectral algorithm is more efficient than ours. Indeed, in our case the computation of the Bernoulli probabilities can also be saved but the sequential sampling is the heaviest task and needs to be done for each sample.

4.3. Sampling the patches of an image

A random and diverse subselection of the set of patches of an image can be useful for numerous image processing applications. A first obvious one is image compression. Indeed, it is possible to obtain a good reconstruction of the image from a very small portion of its patches. It is sometimes necessary to keep only the most informative patches of the image, if possible a small amount, and reconstruct the image, store it, only using these few patches. Moreover, most of patch-based algorithms could use such a subselection of patches to improve or at least speed up its procedures, e.g. for denoising [11]. To do this, the selected patches must be representative of the patches diversity and this is what DPPs offer. Launay and Leclaire [30] explore this strategy to speed up a texture synthesis algorithm.

Given an image u and a set $\mathcal{P}$ of 10 000 randomly picked patches of u, we compare here the selection strategies using either a DPP or a random uniform selection. Let us recall the patch-based kernel (c) defined as the L-ensemble associated with

$$\forall P_1, P_2 \in \mathcal{P}, \quad L(P_1, P_2) = \exp\left(-\frac{\|P_1 - P_2\|_2^2}{s^2}\right),$$

that is, L is a Gaussian kernel applied to the Euclidean distance between the patches of $\mathcal{P}$. This function is commonly chosen to define a similarity measure between patches. It is relevant since in general the reconstruction error is computed in function of the Euclidean distance between the original image and the reconstructed image. We set the bandwidth or scale parameter s to be proportional to the median of the interdistances between the patches, as advised by Aggarwal [2] and Tremblay et al. [44].

Figure 6 presents several reconstructions of two images, obtained by uniform selection or by the DPP defined above, with various expected sample sizes. Notice that while we can control the exact cardinality of the uniform selections, the number of patches in the DPP selections varies as we can only control the expected cardinality during the sampling process. This figure shows how a selection from a DPP provides better reconstructions than a uniform selection, especially when the number of patches is low. Indeed, as the DPP sampling favors diverse set of patches, it is less likely to miss an essential information of the image. On the contrary, nothing prevents the uniform selection from selecting very similar patches. The Pool image on the bottom of Figure 6, for a cradinality equal to 5, clearly illustrates this. The number of patches

in an image depends on the size of the image and is often higher than 10000 while the selection needs to be small (between 5 and 100): here the use of our sequential thinning algorithm is pertinent.

5. Discussion

In this paper, we proposed a new sampling algorithm (Algorithm 3) adapted to general determinantal point processes, which doesn't use the spectral decomposition of the kernel and which is exact. It proceeds in two phases. The first one samples a Bernoulli process whose distribution is adapted to the targeted DPP. It is a fast and efficient step that reduces the initial number of points of the ground set. We know that if $I - K$ is invertible, the expectation of the cardinality of the Bernoulli process is proportional to the expectation of the cardinality of the DPP. The second phase is a sequential sampling from the points selected in the first step. This phase is made possible by the explicit formulations of the general marginals and the pointwise conditional probabilities of any DPP from its kernel K. The sampling is sped up using updated Cholesky decompositions to compute the conditional probabilities. MATLAB and Python implementations of the sequential thinning algorithm can be found online (`https://www.math-info.univ-paris5.fr/~claunay/exact_sampling.html`).

In terms of running times, we have detailed the cases for which this algorithm is competitive with the spectral algorithm, in particular when the size of the ground set is high and the expected cardinality of the DPP is modest. This framework is common in machine learning applications. Indeed, DPPs are an interesting solution to subsample a data set, initialize a segmentation algorithm or summarize an image, examples where the number of datapoints needs to be significantly reduced.

Appendix A. Möbius Inversion formula

Proposition 5. (Möbius inversion formula.) *Let V be a finite subset and f and g be two functions defined on the power set $\mathcal{P}(V)$ of subsets of V. Then,*

$$\forall A \subset V, \quad f(A) = \sum_{B \subset A} (-1)^{|A \setminus B|} g(B) \quad \Longleftrightarrow \quad \forall A \subset V, \quad g(A) = \sum_{B \subset A} f(B),$$

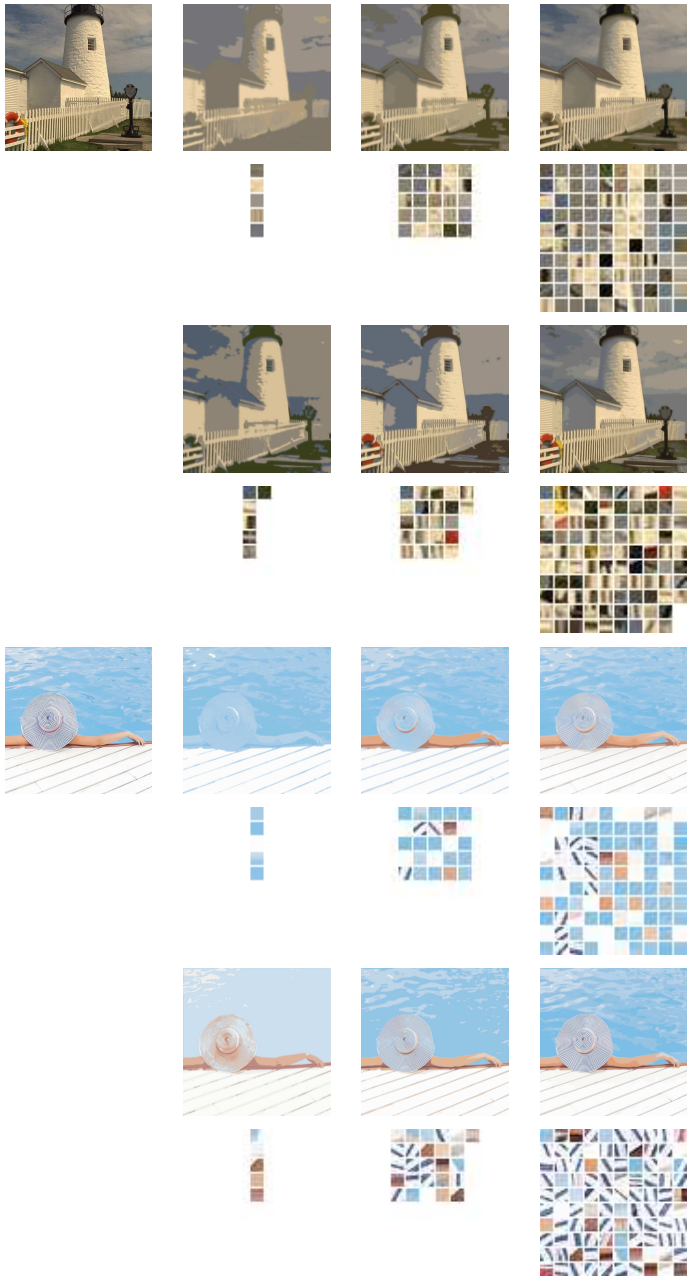


FIGURE 6: Image reconstruction: for each image, first two rows: original and reconstructions with uniformly sampled patches and below, the corresponding selected patches; second two rows: reconstructions with patches sampled according to a DPP and below, the corresponding selected patches.

and

$$\forall A \subset V, \quad f(A) = \sum_{B \supset A} (-1)^{|B \setminus A|} g(B) \quad \Longleftrightarrow \quad \forall A \subset V, \quad g(A) = \sum_{B \supset A} f(B).$$

Proof. The first equivalence is proved e.g. in [35]. The second equivalence corresponds to the first applied to $\widetilde{f}(A) = f(\overline{A})$ and $\widetilde{g}(A) = g(\overline{A})$. You will find more details on this matter in the book of Rota [39]. □

Appendix B. Cholesky Decomposition Update

To be efficient, the sequential algorithm relies on Cholesky decompositions that are updated step by step to save computations. Let M be a symmetric semi-definite matrix of the form $M = \begin{pmatrix} A & B \\ B^t & C \end{pmatrix}$ where A and C are square matrices. We suppose that the Cholesky decomposition T_A of the matrix A has already been computed and we want to compute the Cholesky decomposition T_M of M. Then, set

$$V = T_A^{-1} B \quad \text{and} \quad X = C - V^t V = C - B^t A^{-1} B$$

the Schur complement of the block A of the matrix M. Denote by T_X the Cholesky decomposition of X. Then, the Cholesky decomposition of M is given by

$$T_M = \begin{pmatrix} T_A & 0 \\ V^t & T_X \end{pmatrix}.$$

Indeed,

$$T_M T_M^t = \begin{pmatrix} T_A & 0 \\ V^t & T_X \end{pmatrix} \begin{pmatrix} T_A^t & V \\ 0 & T_X^t \end{pmatrix} = \begin{pmatrix} T_A T_A^t & T_A V \\ V^t T_A^t & V^t V + T_X T_X^t \end{pmatrix} = \begin{pmatrix} A & B \\ B^t & C \end{pmatrix}.$$

Acknowledgement

Accepted for publication by the Applied Probability Trust (http://www.appliedprobability.org) in the Journal of Applied Probability JAP 57.4 (December 2020). This work was supported by grants from Région Ile-de-France. We thank the reviewers for their valuable comments and suggestions that helped us to improve the paper.

References

[1] Affandi, R. H., Kulesza, A., Fox, E. B. and Taskar, B. (2013). Nystrom approximation for large-scale determinantal processes. In *AISTATS*. vol. 31 of *JMLR Workshop and Conference Proceedings*. JMLR.org. pp. 85–98.

[2] Aggarwal, C. C. (2016). *Outlier Analysis* 2nd ed. Springer Publishing Company, Incorporated.

[3] Amblard, P.-O., Barthelme, S. and Tremblay, N. (2018). Subsampling with k determinantal point processes for estimating statistics in large data sets. In *2018 IEEE workshop on Statistical Signal Processing (SSP 2018)*. Freiburg, Germany.

[4] Anari, N., Gharan, S. O. and Rezaei, A. (2016). Monte Carlo Markov chain algorithms for sampling strongly rayleigh distributions and determinantal point processes. In *COLT*. vol. 49 of *JMLR Workshop and Conference Proceedings*. JMLR.org. pp. 103–115.

[5] Avena, L. and Gaudillière, A. (2018). Two Applications of Random Spanning Forests. *Journal of Theoretical Probability* **31,** 1975–2004.

[6] Bardenet, R., Lavancier, F., Mary, X. and Vasseur, A. (2017). On a few statistical applications of determinantal point processes. *ESAIM: Procs* **60,** 180–202.

[7] Barthelmé, S., Amblard, P.-O. and Tremblay, N. (2019). Asymptotic equivalence of fixed-size and varying-size determinantal point processes. *Bernoulli* **25,** 3555–3589.

[8] Błaszczyszyn, B. and Keeler, H. P. (2019). Determinantal thinning of point processes with network learning applications. In *2019 IEEE Wireless Communications and Networking Conference (WCNC)*. pp. 1–8.

[9] Borodin, A. and Rains, E. M. (2005). Eynard–Mehta theorem, Schur process, and their Pfaffian analogs. *Journal of Statistical Physics* **3,** 291–317.

[10] Brunel, V., Moitra, A., Rigollet, P. and Urschel, J. (2017). Rates of estimation for determinantal point processes. In *COLT*. vol. 65 of *Proceedings of Machine Learning Research*. PMLR. pp. 343–345.

[11] Buades, A., Coll, B. and Morel, J. (2005). A non-local algorithm for image denoising. In *Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition - Volume 02*. CVPR '05. IEEE Computer Society. pp. 60–65.

[12] Chiu, S., Stoyan, D., Kendall, W. and Mecke, J. (2013). *Stochastic Geometry and Its Applications*. Wiley Series in Probability and Statistics. Wiley.

[13] Dereziński, M., Calandriello, D. and Valko, M. (2019). Exact sampling of determinantal point processes with sublinear time preprocessing. *arXiv e-prints* arXiv:1905.13476.

[14] Dupuy, C. and Bach, F. (2016). Learning determinantal point processes in sublinear time. Accepted to AISTATS 2018.

[15] Gartrell, M., Paquet, U. and Koenigstein, N. (2017). Low-rank factorization of determinantal point processes. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence*. AAAI'17. AAAI Press. pp. 1912–1918.

[16] Gautier, G. (2020). On sampling determinantal point processes. *PhD thesis*. Ecole Centrale de Lille.

[17] Gautier, G., Bardenet, R. and Valko, M. (2017). Zonotope hit-and-run for efficient sampling from projection DPPs. In *Proceedings of the 34th International Conference on Machine Learning*. ed. D. Precup and Y. W. Teh. vol. 70 of *Proceedings of Machine Learning Research*. PMLR. pp. 1223–1232.

[18] Gautier, G., Bardenet, R. and Valko, M. (2018). DPPy: Sampling determinantal point processes with Python. *CoRR* **abs/1809.07258,**.

[19] George, A., Heath, M. T. and Liu, J. (1986). Parallel Cholesky factorization on a shared-memory multiprocessor. *Linear Algebra and its Applications* **77,** 165–187.

[20] Gillenwater, J., Kulesza, A., Mariet, Z. and Vassilvtiskii, S. (2019). A tree-based method for fast repeated sampling of determinantal point processes. In *Proceedings of the 36th International Conference on Machine Learning*. ed. K. Chaudhuri and R. Salakhutdinov. vol. 97 of *Proceedings of Machine Learning Research*. PMLR, Long Beach, California, USA. pp. 2260–2268.

[21] Gillenwater, J., Kulesza, A. and Taskar, B. (2012). Discovering diverse and salient threads in document collections. In *EMNLP-CoNLL*. ACL. pp. 710–720.

[22] Ginibre, J. (1965). Statistical ensembles of complex: Quaternion, and real matrices. *Journal of Mathematical Physics* **Vol: 6,**.

[23] Horn, R. A. and Johnson, C. R. (1990). *Matrix Analysis*. Cambridge University Press.

[24] Hough, J. B., Krishnapur, M., Peres, Y. and Virág, B. (2006). Determinantal processes and independence. *Probability Surveys* 206–229.

[25] Kang, B. (2013). Fast determinantal point process sampling with application to clustering. In *Advances in Neural Information Processing Systems 26*. ed. C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger. Curran Associates, Inc. pp. 2319–2327.

[26] Kulesza, A. and Taskar, B. (2010). Structured determinantal point processes. In *NIPS*. Curran Associates, Inc. pp. 1171–1179.

[27] Kulesza, A. and Taskar, B. (2012). Determinantal point processes for machine learning. *Foundations and Trends in Machine Learning* **5,** 123–286.

[28] Kulesza, A. and Taskar, B. (2012). Learning determinantal point processes. *CoRR* **abs/1202.3738,**.

[29] Launay, C., Galerne, B. and Desolneux, A. (2018). Exact Sampling of Determinantal Point Processes without Eigendecomposition. *arXiv e-prints* arXiv:1802.08429.

[30] Launay, C. and Leclaire, A. (2019). Determinantal patch processes for texture synthesis. In *GRETSI 2019*. Lille, France.

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

[32] Li, C., Jegelka, S. and Sra, S. (2016). Efficient sampling for k-determinantal point processes. In *Proceedings of the 19th International Conference on Artificial Intelligence and Statistics*. ed. A. Gretton and C. C. Robert. vol. 51 of *Proceedings of Machine Learning Research*. PMLR, Cadiz, Spain. pp. 1328–1337.

[33] Li, C., Sra, S. and Jegelka, S. (2016). Fast mixing Markov chains for strongly Rayleigh measures, dpps, and constrained sampling. In *Advances in Neural Information Processing Systems 29*. ed. D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett. Curran Associates, Inc. pp. 4188–4196.

[34] Mayers, D. and Süli, E. (2003). *An introduction to numerical analysis*. Cambridge Univ. Press, Cambridge.

[35] Mumford, D. and Desolneux, A. (2010). *Pattern Theory: The Stochastic Analysis of Real-World Signals*. Ak Peters Series. Taylor & Francis.

[36] Poulson, J. (2019). High-performance sampling of generic Determinantal Point Processes. *arXiv e-prints* arXiv:1905.00165.

[37] Propp, J. G. and Wilson, D. B. (1998). How to get a perfectly random sample from a generic Markov chain and generate a random spanning tree of a directed graph. *J. Algorithms* **27,** 170–217.

[38] Rolski, T. and Szekli, R. (1991). Stochastic ordering and thinning of point processes. *Stochastic Processes and their Applications* **37,** 299–312.

[39] Rota, G.-C. (1964). On the foundations of combinatorial theory I. Theory of Möbius functions. *Z. Wahrscheinlichkeitstheorie und verw* **2,** 340–368.

[40] Scardicchio, A., Zachary, C. E. and Torquato, S. (2009). Statistical properties of determinantal point processes in high dimensional euclidean spaces. *Phys. Rev. E* **79,**.

[41] Shirai, T. and Takahashi, Y. (2003). Random point fields associated with certain Fredholm determinants. I. Fermion, Poisson and boson point processes. *Journal of Functional Analysis* **205,** 414–463.

[42] Trefethen, L. N. and Bau, D. (1997). *Numerical Linear Algebra*. SIAM: Society for Industrial and Applied Mathematics.

[43] Tremblay, N., Barthelmé, S. and Amblard, P.-O. (2018). Optimized algorithms to sample determinantal point processes. *CoRR* **abs/1802.08471,**.

[44] Tremblay, N., Barthelmé, S. and Amblard, P.-O. (2019). Determinantal Point Processes for Coresets. *Journal of Machine Learning Research*.

[45] Zhang, C., Kjellström, H. and Mandt, S. (2017). Balanced mini-batch sampling for SGD using determinantal point processes. In *Proceedings of the Thirty-Third Conference on Uncertainty in Artificial Intelligence*.