DPPy: DPP Sampling with Python

Guillaume Gautier[†*] G.GAUTIER@INRIA.FR
Guillermo Polito[†*] GUILLERMO.POLITO@UNIV-LILLE.FR
Rémi Bardenet[†] REMI.BARDENET@GMAIL.COM
Michal Valko[‡*†] VALKOM@DEEPMIND.COM
[†] *Univ. Lille, CNRS, Centrale Lille, UMR 9189 – CRIStAL, 59651 Villeneuve d'Ascq, France*
[*] *Inria Lille-Nord Europe, 40 avenue Halley 59650 Villeneuve d'Ascq, France*
[‡] *DeepMind Paris, 14 Rue de Londres, 75009 Paris, France*

Editor: Andreas Mueller

Abstract

Determinantal point processes (DPPs) are specific probability distributions over clouds of points that are used as models and computational tools across physics, probability, statistics, and more recently machine learning. Sampling from DPPs is a challenge and therefore we present DPPy, a Python toolbox that gathers known exact and approximate sampling algorithms for both finite and continuous DPPs. The project is hosted on GitHub and equipped with an extensive documentation.

Keywords: determinantal point processes, sampling, MCMC, random matrices, Python

1. Introduction

Determinantal point processes (DPPs) are distributions over configurations of points that encode diversity through a kernel function K. They were introduced by Macchi (1975) as models for beams of fermions, and they have since found applications in fields as diverse as probability (Soshnikov, 2000; König, 2004; Hough et al., 2006), statistical physics (Pathria & Beale, 2011), Monte Carlo methods (Bardenet & Hardy, 2019), spatial statistics (Lavancier et al., 2012), and machine learning (ML, Kulesza & Taskar, 2012).

In ML, DPPs are mainly used as models for diverse sets of items, as in recommendation (Kathuria et al., 2016; Gartrell et al., 2016) or text summarization (Dupuy & Bach, 2018). Consequently, MLers mostly use finite DPPs, which are distributions over subsets of a finite *ground set* of cardinality M, parametrized by an $M \times M$ kernel matrix $\mathbf{K}$. Routine inference tasks such as normalization, marginalization, or sampling have complexity $\mathcal{O}(M^3)$ (Gillenwater, 2014). Like other kernel methods, when M is large, $\mathcal{O}(M^3)$ is a bottleneck.

In terms of software, the R library spatstat of Baddeley & Turner (2005), a general-purpose toolbox on spatial point processes, includes sampling and learning of continuous DPPs with stationary kernels, as described by Lavancier et al. (2012). Complementarily, we propose DPPy, a turnkey Python implementation of known general algorithms to sample *finite* DPPs. We also include algorithms for non-stationary continuous DPPs, e.g., related to random covariance matrices or Monte Carlo methods, which are also of interest for MLers.

The DPPy project, hosted on GitHub, is already being used by the cross-disciplinary DPP community (Burt et al., 2019; Kammoun, 2018; Poulson, 2019; Dereziński et al., 2019; Gautier et al., 2019). We use Travis for continuous integration and Coveralls for test coverage. Through ReadTheDocs we provide an extensive documentation, which covers the essential mathematical background and showcases the key properties of DPPs through DPPy objects and associated methods. DPPy thus also serves as a tutorial on DPPs.

github.com/guilgautier/DPPy
dppy.readthedocs.io
travis-ci.com/guilgautier/DPPy
coveralls.io/github/guilgautier/DPPy

2. Definitions

A point process is a random subset of points $\mathcal{X} = \{X_1, \dots, X_N\} \subset \mathbb{X}$, where the number of points N is itself random. We further add to the definition that N should be almost surely finite and that all points in a sample are distinct. Given a reference measure μ on $\mathbb{X}$, a point process is usually characterized by giving its k-correlation functions ρ_k for all k, where

$$\mathbb{P}\left[\begin{array}{c}\exists \text{ one point of the process in}\\ \text{each ball } B(x_i, \mathrm{d}x_i), \forall i = 1, \dots, k\end{array}\right] = \rho_k\left(x_1, \dots, x_k\right) \prod_{i=1}^{k} \mu(\mathrm{d}x_i),$$

see Møller & Waagepetersen (2004, Section 4). The functions ρ_k describe the interaction among points in $\mathcal{X}$ by quantifying co-occurrence of points at a set of locations.

A point process $\mathcal{X}$ on $(\mathbb{X}, \mu)$ parametrized by a kernel $K : \mathbb{X} \times \mathbb{X} \to \mathbb{C}$ is said to be *determinantal*, denoted as $\mathcal{X} \sim \mathrm{DPP}(K)$, if its k-correlation functions satisfy

$$\rho_k(x_1, \dots, x_k) = \det\left[K(x_i, x_j)\right]_{i,j=1}^{k}, \quad \forall k \geq 1.$$

In ML, most DPPs are in the finite setting where $\mathbb{X} = \{1, \dots, M\}$ and $\mu = \sum_{i=1}^{M} \delta_i$. In this context, the kernel function becomes an $M \times M$ matrix $\mathbf{K}$, and the correlation functions refer to inclusion probabilities. DPPs are thus often defined as $\mathcal{X} \sim \mathrm{DPP}(\mathbf{K})$ if

$$\mathbb{P}[S \subset \mathcal{X}] = \det \mathbf{K}_S, \quad \forall S \subset \mathbb{X}, \tag{1}$$

where $\mathbf{K}_S$ denotes the submatrix of $\mathbf{K}$ formed by the rows and columns indexed by S. The kernel matrix $\mathbf{K}$ is commonly assumed to be real-symmetric, in which case the existence and uniqueness of the DPP in Equation 1 is equivalent to the condition that the eigenvalues of $\mathbf{K}$ lie in $[0, 1]$. The result also holds for general Hermitian kernel functions K with additional assumptions (Soshnikov, 2000, Theorem 3). We note that there are also DPPs with nonsymmetric kernels (Borodin et al., 2010; Gartrell et al., 2019).

Oftentimes, ML practitioners favor a more flexible definition of a DPP in terms of a *likelihood kernel* $\mathbf{L}$, which only requires $\mathbf{L} \succeq 0$,

$$\mathbb{P}[\mathcal{X} = S] = \frac{\det \mathbf{L}_S}{\det\left[I + \mathbf{L}\right]}. \tag{2}$$

This avoids defining a *correlation kernel* $0 \preceq \mathbf{K} \preceq I$. Yet, the $\mathbf{L}$ parametrization makes Equation 1 less interpretable and does not cover important cases such as fixed-size DPPs, which correspond to projection kernels $\mathbf{K}$. Kulesza & Taskar (2012, Section 5) countered that with k-DPPs, which can be understood as DPPs parametrized by a likelihood kernel, further conditioned to have exactly k elements. However, in general, k-DPPs are not DPPs.

The main interest in DPPs in ML is that they model diversity while being tractable. Compared to independent sampling with the same marginals, Equation 1 entails

$$\mathbb{P}[\{i, j\} \subset \mathcal{X}] = \mathbf{K}_{ii}\mathbf{K}_{jj} - \mathbf{K}_{ij}\mathbf{K}_{ji} = \mathbb{P}[\{i\} \subset \mathcal{X}] \times \mathbb{P}[\{j\} \subset \mathcal{X}] - \mathbf{K}_{ij}^2. \tag{3}$$

In particular, the larger $\mathbf{K}_{ij}$, the less likely items i and j co-occur. If $\mathbf{K}_{ij}$ models the similarity between items i and j, DPPs are thus random *diverse* sets of elements.

Most point processes that encode diversity are not tractable, in the sense that efficient algorithms to sample, marginalize, or compute normalization constants are not available. However, DPPs are amenable to these tasks with polynomial complexity (Gillenwater, 2014). Next, we focus on the sampling task, which is the core of DPPy.

3. Sampling determinantal point processes

We henceforth assume that K is real-symmetric and satisfies suitable conditions (Soshnikov, 2000, Theorem 3), so that its spectral decomposition is available

$$K(x,y) = \sum_{i=1}^{\infty} \lambda_i \phi_i(x)\phi_i(y), \quad \text{with} \int_{\mathbb{X}} \phi_i(x)\phi_j(x)\mu(\mathrm{d}x) = \delta_{ij}.$$

Note that, in the finite case, the spectral theorem is enough to eigendecompose $\mathbf{K}$. Hough et al. (2006, Theorem 7) proved that sampling $\operatorname{DPP}(K)$ can be done in two steps:

1. draw $B_i \sim \mathcal{B}\text{er}(\lambda_i)$ independently and denote $\{i_1, \dots, i_N\} = \{i : B_i = 1\}$,
2. sample from the DPP with kernel $\widetilde{K}(x,y) = \sum_{n=1}^{N} \phi_{i_n}(x)\phi_{i_n}(y)$.

In other words, all DPPs are mixtures of *projection* DPPs, i.e., DPPs parametrized by an orthogonal projection kernel. In a nutshell, Step 1 selects a component of the mixture and Step 2 generates a sample of the projection $\operatorname{DPP}(\widetilde{K})$. Hough et al. (2006, Algorithm 18) provide a generic projection DPP sampler that we briefly describe. First, the projection DPP with kernel $\widetilde{K}$ has exactly $N = \operatorname{rank} \widetilde{K}$ points, μ-almost surely. Then, the sequential aspect of the chain rule applied to sample $(X_1, \dots, X_N)$ with probability distribution

$$\frac{\det\left[\widetilde{K}(x_p, x_n)\right]_{p,n=1}^{N}}{N!} \prod_{n=1}^{N} \mu(\mathrm{d}x_n) = \frac{\|\Phi(x_1)\|^2}{N} \mu(\mathrm{d}x_1) \prod_{n=2}^{N} \frac{\operatorname{distance}^2\left(\Phi(x_n), \operatorname{span}\left\{\Phi(x_p)\right\}_{p=1}^{n-1}\right)}{N-(n-1)} \mu(\mathrm{d}x_n), \tag{4}$$

can be discarded to get a valid sample $\{X_1, \dots, X_N\} \sim \operatorname{DPP}(\widetilde{K})$. To each $x \in \mathbb{X}$ we associate a *feature vector* $\Phi(x) \triangleq (\phi_{i_1}(x), \dots, \phi_{i_N}(x))$, so that $\widetilde{K}(x,y) = \Phi(x)^{\mathsf{T}}\Phi(y)$.

A few remarks are in order. First, the LHS of Equation 4 defines an exchangeable probability distribution. Second, the successive ratios that appear in the RHS are the normalized conditional densities (w.r.t. μ) that drive the chain rule. The associated normalizing constants are independent of the previous points. The numerators can be written as the ratio of two determinants and further expanded with Woodbury's formula. They can be identified as the incremental posterior variances in Gaussian process regression with kernel $\widetilde{K}$ (Rasmussen & Williams, 2006, Equation 2.26). Third, the chain rule expressed in Equation 4 has a strong Gram-Schmidt flavor since it actually comes from a recursive application of the base×height formula. In the end, DPPs favor configuration of points whose feature vectors $\Phi(x_1), \dots, \Phi(x_N)$ span a large volume, which is another way of understanding repulsiveness. The previous sampling scheme is exact and generic but, except for projection kernels, it requires the eigendecomposition of the underlying kernel.

In the finite setting, this corresponds to an initial $\mathcal{O}(M^3)$ eigendecomposition cost, followed by an average complexity of order $\mathcal{O}(M\, \mathbb{E}[|\mathcal{X}|]^2)$ (see, e.g., Gillenwater, 2014; Tremblay et al., 2018). Besides, there exist some alternative exact samplers. Poulson (2019) and Launay et al. (2018) use a $\mathcal{O}(M^3)$ Cholesky-based chain rule on sets; each item in turn is decided to be excluded or included in the sample. Dereziński et al. (2019) first sample an intermediate distribution and correct the bias by thinning the intermediate sample (with size smaller than M) using a carefully designed DPP. In certain regimes, this procedure may be more practical with an overall $\mathcal{O}(M \operatorname{poly}(\mathbb{E}[|\mathcal{X}|]) \operatorname{polylog}(M))$ cost. In the continuous case, sampling exactly each conditional that appears in the right-hand side of Equation 4 can be done using rejection sampling with a tailored proposal, see, e.g., Gautier et al. (2019).

In applications where the cost of exact sampling becomes a bottleneck, users rely on approximate sampling. Research has focused mainly on kernel approximation (Affandi et al., 2013) and MCMC samplers (Anari et al., 2016; Li et al., 2016; Gautier et al., 2017).

However, specific DPPs admit efficient exact samplers that do not rely on Equation 4, e.g., uniform spanning trees (UST, Propp & Wilson, 1998, Figure 1(c)) or eigenvalues of random matrices. For instance, a β-ensemble is a set of N points of $\mathbb{R}$ with joint distribution

$$\frac{1}{Z_{N,\beta}} \prod_{p<n} |x_p - x_n|^\beta \prod_{n=1}^{N} \omega(x_n)\, \mathrm{d}x_n, \quad \text{where } \beta > 0.$$

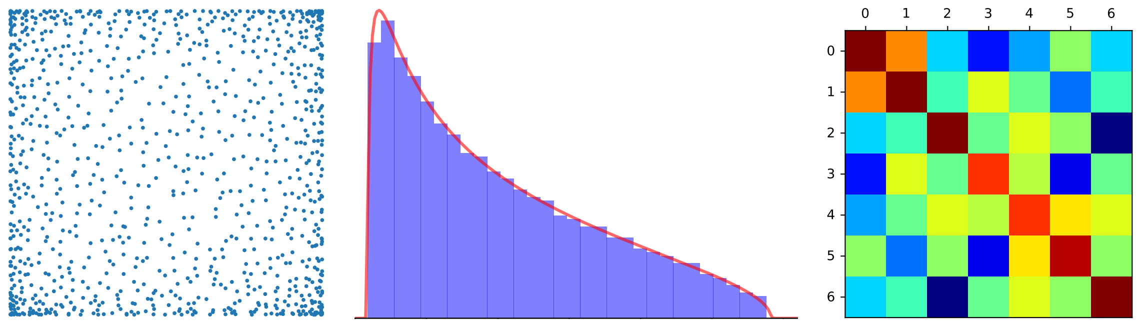
For some choices of the weight function ω, the β-ensemble can be sampled by computing the eigenvalues of simple tridiagonal (Dumitriu & Edelman, 2002) or quindiagonal random matrices (Killip & Nenciu, 2004). In particular, ($\beta = 2$)-ensembles correspond to projection DPPs (König, 2004). They are therefore examples of continuous DPPs that can be sampled exactly in $\mathcal{O}(N^2)$ time, without rejection. Some of these ensembles are of direct interest to MLers. The *Laguerre* ensemble, for instance, has ω be a Gamma pdf, and corresponds to the eigenvalues of the empirical covariance matrix of i.i.d. Gaussian vectors, see Figure 1(b). Finally, we mention that DPPy also features an exact sampler of the multivariate extension of the *Jacobi* ensemble, which has been central in recent results on faster-than-Monte Carlo numerical integration (Bardenet & Hardy, 2019; Gautier et al., 2019; Mazoyer et al., 2019).

4. The DPPy toolbox

DPPy handles Python objects that fit the natural definition of the corresponding DPPs; see also the documentation and the corresponding Jupyter notebooks, which showcase DPPy objects. For example, `FiniteDPP(kernel_type="correlation", **{"K":K})` instantiates a finite DPP($\mathbf{K}$). Its two main methods, `.sample_exact()` and `.sample_mcmc()` implement the different exact samplers and current state-of-the-art MCMC samplers. To sample k-DPPs, the additional `.sample_exact_k_dpp()` and `.sample_mcmc_k_dpp()` methods are available.

A Laguerre β-ensemble is instantiated as `LaguerreEnsemble(beta=2)`. When $\beta \in \{1, 2, 4\}$, it can be sampled either with `.sample_full_model()`, as the eigenvalues of a random covariance matrix, or, more efficiently and for any $\beta > 0$, using a tridiagonal matrix with `.sample_banded_model()`. Samples can be displayed via `.plot()` or `.hist()` to construct the empirical distribution that converges to the Marčenko-Pastur distribution, see Figure 1(b).

DPPy can readily serve as research and teaching support. DPPy is also ready for other contributors to add content and enlarge its scope, e.g., with procedures for learning kernels.



(a) 2D Jacobi ensemble (b) $\beta = 2$-Laguerre ensemble (c) $\mathbf{K}$ kernel of UST

Figure 1: Some displays available in DPPy

Acknowledgments

We acknowledge funding by European CHIST-ERA project DELTA, the French Ministry of Higher Education and Research, the Nord-Pas-de-Calais Regional Council, Inria and Otto-von-Guericke-Universität Magdeburg associated-team north-European project Allocate, and French National Research Agency project BoB (n.ANR-16-CE23-0003).

References

Affandi, R. H., Kulesza, A., Fox, E. B., and Taskar, B. Nyström Approximation for Large-Scale Determinantal Processes. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2013.

Anari, N., Gharan, S. O., and Rezaei, A. Monte Carlo Markov Chain Algorithms for Sampling Strongly Rayleigh Distributions and Determinantal Point Processes. In *Conference on Learning Theory (COLT)*, 2016. arXiv:1602.05242.

Baddeley, A. and Turner, R. spatstat : An R Package for Analyzing Spatial Point Patterns. *Journal of Statistical Software*, 2005.

Bardenet, R. and Hardy, A. Monte Carlo with Determinantal Point Processes. *Annals of Applied Probability, in press*, 2019. arXiv:1605.00361.

Borodin, A., Diaconis, P., and Fulman, J. On adding a list of numbers (and other one-dependent determinantal processes). *Bulletin of the American Mathematical Society*, 2010. arXiv:0904.3740.

Burt, D., Rasmussen, C. E., and Wilk, M. V. D. Rates of Convergence for Sparse Variational Gaussian Process Regression. In *International Conference on Machine Learning (ICML)*, 2019. arXiv:1903.03571.

Dereziński, M., Calandriello, D., and Valko, M. Exact sampling of determinantal point processes with sublinear time preprocessing. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019. arXiv:1905.13476.

Dumitriu, I. and Edelman, A. Matrix Models for Beta Ensembles. *Journal of Mathematical Physics*, 2002. arXiv:math-ph/0206043.

Dupuy, C. and Bach, F. Learning Determinantal Point Processes in Sublinear Time. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2018. arXiv:1610.05925.

Gartrell, M., Paquet, U., and Koenigstein, N. Low-Rank Factorization of Determinantal Point Processes for Recommendation. In *AAAI Conference on Artificial Intelligence*, 2016. arXiv:1602.05436.

Gartrell, M., Brunel, V.-E., Dohmatob, E., and Krichene, S. Learning Nonsymmetric Determinantal Point Processes. *ArXiv e-prints*, 2019. arXiv:1905.12962.

Gautier, G., Bardenet, R., and Valko, M. Zonotope hit-and-run for efficient sampling from projection DPPs. *International Conference on Machine Learning (ICML)*, 2017. arXiv:1705.10498.

Gautier, G., Bardenet, R., and Valko, M. On two ways to use determinantal point processes for Monte Carlo integration. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019.

Gillenwater, J. Approximate inference for determinantal point processes. PhD thesis, University of Pennsylvania, 2014.

Hough, J. B., Krishnapur, M., Peres, Y., and Virág, B. Determinantal Processes and Independence. In *Probability Surveys*, 2006. arXiv:math/0503110.

Kammoun, M. S. Monotonous subsequences and the descent process of invariant random permutations. *Electronic Journal of Probability*, 2018. arXiv:1805.05253.

Kathuria, T., Deshpande, A., and Kohli, P. Batched Gaussian Process Bandit Optimization via Determinantal Point Processes. In *Advances in Neural Information Processing Systems (NIPS)*, 2016. arXiv:1611.04088.

Killip, R. and Nenciu, I. Matrix models for circular ensembles. *International Mathematics Research Notices*, 2004. arXiv:math/0410034.

König, W. Orthogonal polynomial ensembles in probability theory. *Probability Surveys*, 2004. arXiv:math/0403090.

Kulesza, A. and Taskar, B. Determinantal Point Processes for Machine Learning. *Foundations and Trends in Machine Learning*, 2012. arXiv:1207.6083.

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

Lavancier, F., Møller, J., and Rubak, E. Determinantal point process models and statistical inference : Extended version. *Journal of the Royal Statistical Society. Series B: Statistical Methodology*, 2012. arXiv:1205.4818.

Li, C., Jegelka, S., and Sra, S. Fast Mixing Markov Chains for Strongly Rayleigh Measures, DPPs, and Constrained Sampling. In *Advances in Neural Information Processing Systems (NIPS)*, 2016. arXiv:1608.01008.

Macchi, O. The coincidence approach to stochastic point processes. *Advances in Applied Probability*, 1975.

Mazoyer, A., Coeurjolly, J.-F., and Amblard, P.-O. Projections of determinantal point processes. *ArXiv e-prints*, 2019. arXiv:1901.02099.

Møller, J. and Waagepetersen, R. P. Statistical inference and simulation for spatial point processes. Chapman & Hall/CRC. 2004.

Pathria, R. K. and Beale, P. D. Statistical Mechanics. Academic Press. 2011.

Poulson, J. High-performance sampling of generic Determinantal Point Processes. *ArXiv e-prints*, 2019. arXiv:1905.00165.

Propp, J. G. and Wilson, D. B. How to Get a Perfectly Random Sample from a Generic Markov Chain and Generate a Random Spanning Tree of a Directed Graph. *Journal of Algorithms*, 1998.

Rasmussen, C. E. and Williams, C. K. I. Gaussian processes for machine learning. MIT Press. 2006.

Soshnikov, A. Determinantal random point fields. *Russian Mathematical Surveys*, 2000. arXiv:math/0002099.

Tremblay, N., Barthelme, S., and Amblard, P.-O. Optimized Algorithms to Sample Determinantal Point Processes. *ArXiv e-prints*, 2018. arXiv:1802.08471.