Sampling from Arbitrary Functions via PSD Models

Ulysse Marteau-Ferey, Francis Bach, Alessandro Rudi

INRIA, École normale supérieure, CNRS, PSL Research University

An ideal model for probabilities

Non-negative	The estimator should be a function $p_\theta : \mathcal{X} \to \mathbb{R}_+$ that admits only non-negative values.
Sum & Product Rules	The model should allow for efficient computations of key operations between probabilities (e.g. Sum & Product rules).
Concise Approximation	The number of parameters θ required to approximate a density p depends on its regularity properties and does not explode in the number of dimensions of the ambient space.
Optimal Learning	The model can learn a probability p from a finite number of i.i.d. samples from it achieving minimax learning rates.
Efficient Sampling	It is possible to sample from p_θ (or an approximation of it) in polynomial time.

State of the art

	Non-negative	Sum Rule	Product Rule	Concise Approximation	Optimal Learning	Efficient Sampling
Linear Models	✗	✓	✓	✓	✓	✗
Mean Embeddings	✗	✓	✓	✓	✓	✗
Mixture Models	✓	✓	✓	✗	✗	✓
Exponential Models	✓	✗	✓	✓	✓	✗

> Can we have the best of all worlds?

Positive semidefinite (PSD) models

Generalisation of mixture models

Matrix of centers

$x_1, \ldots, x_m$

$$f(x; A, X, \eta) = \sum_{i,j=1}^{m} A_{ij} k_\eta(x_i, x) k_\eta(x_j, x)$$

PSD matrix

$A \succeq 0$

Gaussian function

$e^{-\eta \|x_i - x\|^2}$

Marteau-Ferey, Bach, Rudi *"Non-parametric models for nonnegative functions".* NeurIPS 2020

Rudi, Ciliberto *"PSD Representations for effective probability models".* NeurIPS 2021

PSD models to the rescue?

	Non-negative	Sum Rule	Product Rule	Concise Approximation	Optimal Learning	Efficient Sampling
Linear Models	✗	✓	✓	✓	✓	✗
Mean Embeddings	✗	✓	✓	✓	✓	✗
Mixture Models	✓	✓	✓	✗	✗	✓
Exponential Models	✓	✗	✓	✓	✓	✗
PSD Models	✓	?	?	?	?	?

Marteau-Ferey, Bach, Rudi
"Non-parametric models for nonnegative functions". NeurIPS 2020

PSD models to the rescue?

	Non-negative	Sum Rule	Product Rule	Concise Approximation	Optimal Learning	Efficient Sampling
Linear Models	✗	✓	✓	✓	✓	✗
Mean Embeddings	✗	✓	✓	✓	✓	✗
Mixture Models	✓	✓	✓	✗	✗	✓
Exponential Models	✓	✗	✓	✓	✓	✗
PSD Models	✓	✓	✓	✓	✓	?

Marteau-Ferey, Bach, Rudi
"Non-parametric models for nonnegative functions". NeurIPS 2020

Rudi, Ciliberto
"PSD Representations for effective probability models". NeurIPS 2021

PSD models to the rescue?

	Non-negative	Sum Rule	Product Rule	Concise Approximation	Optimal Learning	Efficient Sampling
Linear Models	✗	✓	✓	✓	✓	✗
Mean Embeddings	✗	✓	✓	✓	✓	✗
Mixture Models	✓	✓	✓	✗	✗	✓
Exponential Models	✓	✗	✓	✓	✓	✗
PSD Models	✓	✓	✓	✓	✓	✓

This work!

Marteau-Ferey, Bach, Rudi
"Non-parametric models for nonnegative functions". NeurIPS 2020

Rudi, Ciliberto
"PSD Representations for effective probability models". NeurIPS 2021

Sampling from PSD model

Key ingredient: Exact integral on hyper-rectangles

- Given a hyper-rectangle $Q = [a_1, b_1] \times [a_2, b_2] \times \cdots \times [a_d, b_d] \subset \mathbb{R}^d$
- Exact integral in closed form

$$I(Q) = \int_Q f(x\ ; A, X, \eta) dx = \sum_{ij=1}^{m} A_{ij} G_{ij}$$

- G_{ij} in closed form, in terms of the error function $\mathrm{erf}(z) = \frac{2}{\sqrt{\pi}} \int_0^z e^{-t^2} dt$
- Total cost: $O(dm^2)$ arithmetic and erf operations

Implemented in any scientific computing library as numpy, R, Matlab, pytorch, …

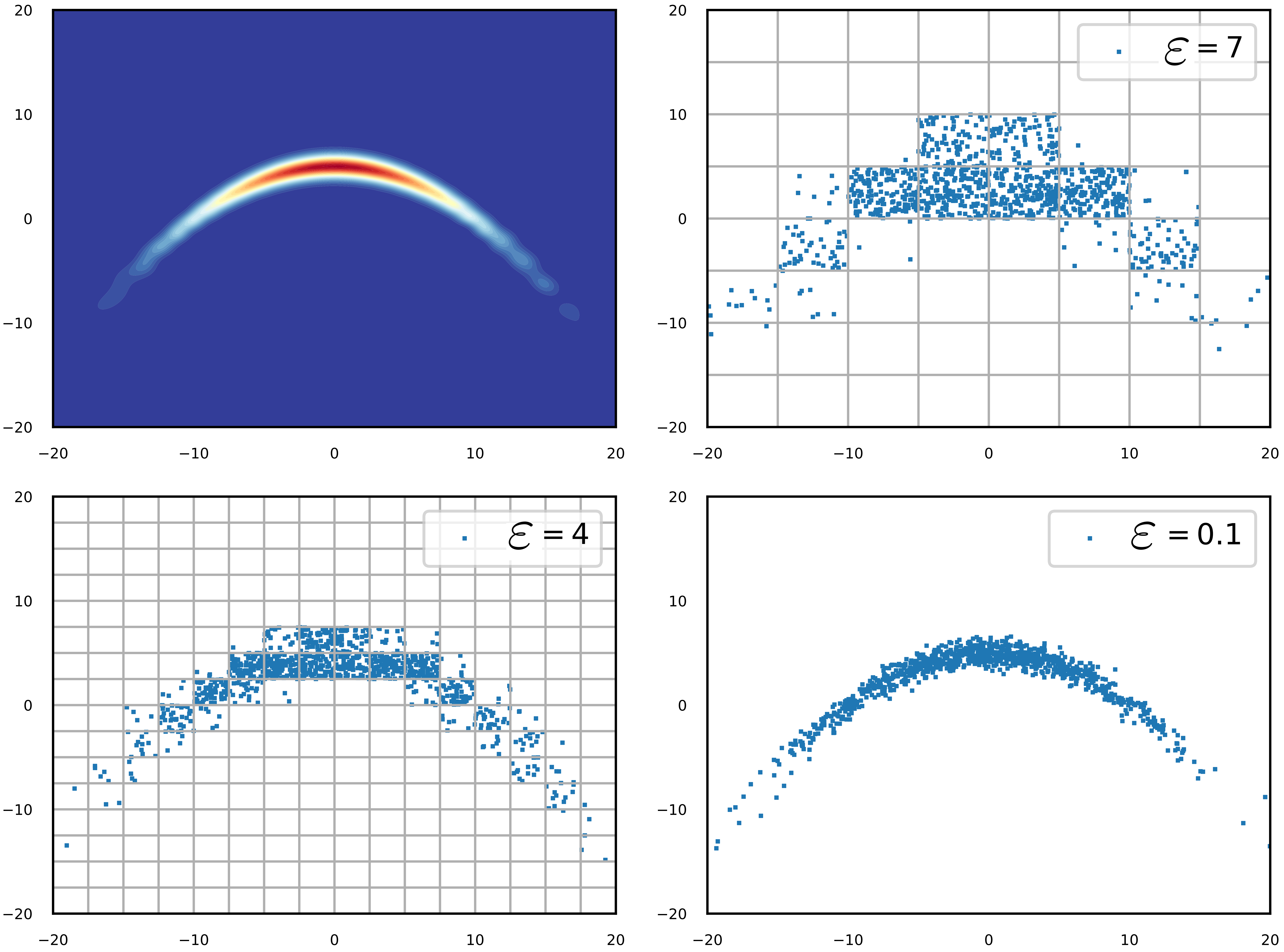
Sampling from PSD model

The algorithm

Divide and conquer:

- Start with a hyper-rectangle Q
    - Divide Q in half along the longest side, obtaining Q_1 and Q_2
    - Set $Q := Q_1$ with probability $\frac{I(Q_1)}{I(Q)}$ (otherwise set $Q := Q_2$)
- Repeat until each side of Q is shorter than ε
- Return x sampled uniformly from Q

Total cost $O(dm^2 \log \frac{|Q|}{\varepsilon})$ **per sample**



$\mathcal{E} = 7$
$\mathcal{E} = 4$
$\mathcal{E} = 0.1$

Sampling from PSD model

Theoretical properties

- The samples produced by the algorithm are i. i. d.
- Moreover

$$\mathbb{W}_1(\tilde{p}_\varepsilon, p) \;\leq\; \sqrt{d}\,\varepsilon$$

$\mathbb{W}_1$ is the Wasserstein 1 distance (similar result holds for TV or Hellinger)

$\tilde{p}_\varepsilon$ probability of the samples produced by the algorithm

$p = \frac{1}{Z} f(\cdot\,; A, X, \eta)$ where Z is the normalization constant

Sampling from arbitrary densities

Using PSD models

- Step 1: Evaluate the density p in n points, then fit a PSD model
- Step 2: Sampling from the resulting PSD model $\hat{p}$

- Fitting algorithm: kind of Kernel Regression $\longrightarrow$ fast methods (FALKON: $O(n\sqrt{n})$)

Rudi, Carratino, Rosasco. "Falkon: An optimal large scale kernel method." *NIPS 2017.*

- Theoretical guarantees : Assuming $p(x) = e^{V(x)}, V \in C^m(\mathbb{R}^d)$

$$n = O\big(\varepsilon^{-\frac{d}{2m}}\big) \quad \Rightarrow \quad \mathbb{W}_1(\tilde{p}_\varepsilon, p) \leq 2\sqrt{d}\varepsilon$$