Kernel Thinning

Raaz Dwivedi
raaz.rsk@berkeley.edu
Berkeley | EECS

Lester Mackey
lmackey@microsoft.com
Microsoft Research

python `pip install kernelthinning`
GitHub `rzrsk/kernel_thinning`
arXiv.org `https://arxiv.org/abs/2105.05842`

Motivation: MCMC Thinning

- Markov Chain Monte Carlo (MCMC): Workhorse for approximating intractable expectations with asymptotically exact averages

$$\mathbb{P}^\star f := \int f(x) d\mathbb{P}^\star(x) \approx \frac{1}{n}\sum_{i=1}^{n} f(x_i) =: \mathbb{P}_n f \text{ for } x_i\text{'s from Markov Chain}$$

- Samples thinned to minimize computation for downstream function evaluations—-but the integration error worsens with fewer samples

Standard Thinning: Can not thin too much

$x_1, x_2, \dots, x_n \in \mathbb{R}^d$, $\mathbb{P}_{in} := \frac{1}{n}\sum_{i=1}^{n}\delta_{x_i}$ → **Standard-m Thinning** → **Uniform subsample** of size n/m: $y_1, \dots, y_{n/m}$, $\mathbb{P}_{out} := \frac{1}{n/m}\sum_{i=1}^{n/m}\delta_{y_i}$

Standard thinning guarantee $\quad \sup_{\|f\|\le 1} |\mathbb{P}_{in} f - \mathbb{P}_{out} f| \lesssim \sqrt{\frac{m}{n}}$

Monte Carlo gaurantee:
(Input = n iid or fast mixing MCMC points) $\quad \sup_{\|f\|\le 1} |\mathbb{P}_{in} f - \mathbb{P}^\star f| \lesssim \sqrt{\frac{1}{n}}$

m has to be a constant for not losing $n^{-1/2}$ accuracy after thinning

How can we *provably and practically compress much more* while keeping good accuracy?

Via **Kernel Thinning!**

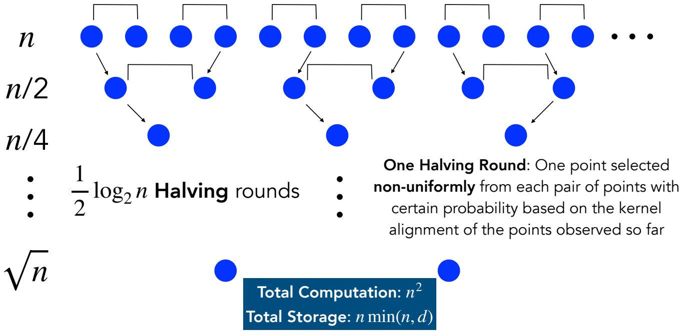
Kernel Thinning: $\sqrt{n}$ points with $n^{-1/2}$ error

$x_1, x_2, \dots, x_n \in \mathbb{R}^d$, suitable kernels, $\mathbb{P}_{in} := \frac{1}{n}\sum_{i=1}^{n}\delta_{x_i}$ → **Kernel Thinning (KT)** → Non-uniform sample of size $\sqrt{n}$: $y_1, \dots, y_{\sqrt{n}}$, $\mathbb{P}_{KT} := \frac{1}{\sqrt{n}}\sum_{i=1}^{\sqrt{n}}\delta_{y_i}$

$$\sup_{\|f\|_{\mathbf{k}}\le 1} |\mathbb{P}_{in} f - \mathbb{P}_{KT} f| \lesssim_d \begin{cases} n^{-1/2}\sqrt{\log n} & \text{(Compactly supported)} \\ n^{-1/2}\sqrt{\log^{d+1} n \log\log n} & \text{(Sub-exponential tails)} \end{cases}$$

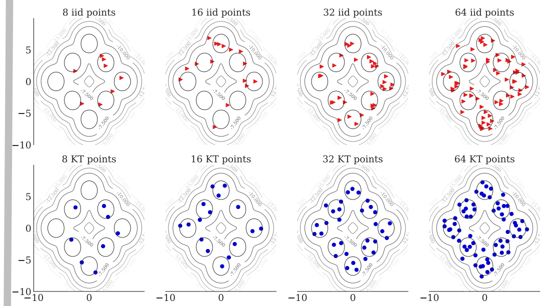
- **Significantly superior to $n^{-1/4}$ rates from Standard-$\sqrt{n}$ Thinning**
- **In fact, nearly minimax integration error in many settings**
- **Quasi Monte Carlo like guarantees, but KT guarantees apply to non-uniform targets with unbounded support**
- **Only kernel evaluations required to implement the algorithm**

Kernel Thinning Algorithm

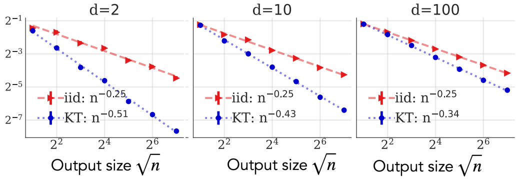


Kernel Thinning in Action

Mixture of Gaussian Data



IID Data



MCMC Data