Probabilistic Surface Optimization and Statistical Inference
Supplementary Material

Dmitry Kopitkov and Vadim Indelman

This document provides supplementary material to the main paper. Therefore, it should not be considered a self-contained document, but instead regarded as its appendix.

D. Kopitkov is with the Technion Autonomous Systems Program (TASP), Technion - Israel Institute of Technology, Haifa 32000, Israel, dimkak@tx.technion.ac.il . V. Indelman is with the Department of Aerospace Engineering, Technion - Israel Institute of Technology, Haifa 32000, Israel, vadim.indelman@technion.ac.il.

Contents

A Appendix: Proof of Theorem 1

Define $\mathbb{S}_{U \cup D}$ to be a support union of two given densities $\mathbb{P}^U$ and $\mathbb{P}^D$; $\mathbb{S}_{U \cap D}$ to be a support intersection of $\mathbb{P}^U$ and $\mathbb{P}^D$; $\mathbb{S}_{U \setminus D}$ to be a support of $\mathbb{P}^U$ where $\mathbb{P}^D(X) = 0$; and $\mathbb{S}_{D \setminus U}$ to be similarly the support of $\mathbb{P}^D$ where $\mathbb{P}^U(X) = 0$.

Further, define *PSO functional*:

$$\begin{aligned} L_{PSO}(f) = -\mathop{\mathbb{E}}_{X \sim \mathbb{P}^U} \widetilde{M}_U\left[X, f(X)\right] + \mathop{\mathbb{E}}_{X \sim \mathbb{P}^D} \widetilde{M}_D\left[X, f(X)\right] = \\ = \int -\mathbb{P}^U(X) \cdot \widetilde{M}_U\left[X, f(X)\right] + \mathbb{P}^D(X) \cdot \widetilde{M}_D\left[X, f(X)\right] dX = \\ = L_{U \cap D}(f) + L_{U \setminus D}(f) + L_{D \setminus U}(f), \end{aligned} \tag{1}$$

$$\begin{aligned} L_{U \cap D}(f) &\triangleq \int_{\mathbb{S}_{U \cap D}} -\mathbb{P}^U(X) \cdot \widetilde{M}_U\left[X, f(X)\right] + \mathbb{P}^D(X) \cdot \widetilde{M}_D\left[X, f(X)\right] dX \\ L_{U \setminus D}(f) &\triangleq -\int_{\mathbb{S}_{U \setminus D}} \mathbb{P}^U(X) \cdot \widetilde{M}_U\left[X, f(X)\right] dX \\ L_{D \setminus U}(f) &\triangleq \int_{\mathbb{S}_{D \setminus U}} \mathbb{P}^D(X) \cdot \widetilde{M}_D\left[X, f(X)\right] dX, \end{aligned}$$

where we define $\widetilde{M}_U\left[X, s\right] \triangleq \int_{-\infty}^{s} M_U(X, t) dt$ and $\widetilde{M}_D\left[X, s\right] \triangleq \int_{-\infty}^{s} M_D(X, t) dt$ to be antiderivatives of $M_U(\cdot)$ and $M_D(\cdot)$ respectively, not necessarily known analytically.

Furthermore, in case $\mathbb{P}^U$ and $\mathbb{P}^D$ have an identical support, then we have $L_{PSO}(f) = L_{U \cap D}(f)$. Such a setting is the one discussed in the paper, and its convergence is proven below. Additionally, afterwards in Sections A.2 and A.3 we continue to discuss a minima of functionals $L_{U \setminus D}(f)$ and $L_{D \setminus U}(f)$ as an extension of Theorem 1.

A.1 Functional $L_{U \cap D}$ over $\mathbb{S}_{U \cap D}$

Consider the following minimization:

$$f^* \in \mathop{\arg\inf}_{f \in C^2(\mathbb{S}_{U \cap D})} L_{U \cap D}(f) \tag{2}$$

where f is a twice continuously differentiable function with a support in $\mathbb{S}_{U \cap D}$. Further, we will characterize f^* in Eq. (2) using calculus of variations and the appropriate Euler-Lagrange equation of the functional $L_{U \cap D}$.

Remark 1. *Importantly, the below variational-based proof is only valid when considering specific function space for which the fundamental lemma of calculus of variations is proven. Therefore, herein we are limiting our proof to the case where $f^* \in C^2(\mathbb{S}_{U \cap D})$. Furthermore, this assumption can be weakened to a class of square integrable functions [4], but the proof becomes more difficult and hence we shall leave it for future work. Likewise, we assume here that all regularity conditions, including functions M_U and M_D being continuous almost* *everywhere, also hold; eliminating these assumptions and providing a more general case is out of this paper scope.*

Next, consider an arbitrary differentiable function $\eta : \mathbb{R}^n \rightarrow \mathbb{R}$ and a small number ϵ, with $\epsilon \cdot \eta(X)$ being a variation of the function $f(X)$, and define the following objective $J(\epsilon)$:

$$J(\epsilon) = L_{U \cap D} \left[f(X) + \epsilon \cdot \eta(X) \right] = \\ = \int_{\mathbb{S}_{U \cap D}} -\mathbb{P}^U(X) \cdot \widetilde{M}_U \left[X, f(X) + \epsilon \cdot \eta(X) \right] + \\ + \mathbb{P}^D(X) \cdot \widetilde{M}_D \left[X, f(X) + \epsilon \cdot \eta(X) \right] dX.$$

From calculus of variations it follows that a specific f is a minima of $L_{U \cap D}(f)$ iff $\epsilon = 0$ is a minima of $J(\epsilon)$ for this f and for any $\eta(X)$ [5]. Hence, we can find the solution of Eq. (2) by finding f that satisfies:

$$0 = \arg\min_{\epsilon} J(\epsilon).$$

To this end, we consider the *second derivative test* for $\epsilon = 0$ to be a minima of $J(\epsilon)$:

$$\frac{dJ(\epsilon)}{d\epsilon}\Big|_{\epsilon=0} = 0, \quad \frac{d^2 J(\epsilon)}{d\epsilon^2}\Big|_{\epsilon=0} > 0 \tag{3}$$

and find f that satisfies these conditions for any differentiable $\eta(X)$. Note that the second condition is sufficient but is not necessary and can be replaced by more general conditions from the *higher-order derivative test*. Yet, in practice it is good enough since it is satisfied by most of PSO instances, as is mentioned below.

The first derivative of the objective $J(\epsilon)$ is:

$$\frac{dJ(\epsilon)}{d\epsilon} = \int_{\mathbb{S}_{U \cap D}} \Big[-\mathbb{P}^U(X) \cdot \frac{\partial \widetilde{M}_U(X, s)}{\partial s}\Big|_{s = f(X) + \epsilon \cdot \eta(X)} + \\ + \mathbb{P}^D(X) \cdot \frac{\partial \widetilde{M}_D(X, s)}{\partial s}\Big|_{s = f(X) + \epsilon \cdot \eta(X)} \Big] \cdot \eta(X) dX = \\ = \int_{\mathbb{S}_{U \cap D}} \Big[-\mathbb{P}^U(X) \cdot M_U \left[X, f(X) + \epsilon \cdot \eta(X) \right] + \\ + \mathbb{P}^D(X) \cdot M_D \left[X, f(X) + \epsilon \cdot \eta(X) \right] \Big] \cdot \eta(X) dX.$$

Further, we apply the first condition from Eq. (3):

$$\frac{dJ(\epsilon)}{d\epsilon}\Big|_{\epsilon=0} = \\ = \int_{\mathbb{S}_{U \cap D}} \Big[-\mathbb{P}^U(X) \cdot M_U \left[X, f(X) \right] + \mathbb{P}^D(X) \cdot M_D \left[X, f(X) \right] \Big] \cdot \eta(X) dX = 0.$$

According to the *fundamental lemma of calculus of variations*, from the above expression it follows that:

$$-\mathbb{P}^U(X)\cdot M_U\left[X, f(X)\right] + \mathbb{P}^D(X)\cdot M_D\left[X, f(X)\right] = 0,$$

which is the Euler-Lagrange equation of the functional $L_{U\cap D}(f)$. Further, we can rewrite it as:

$$\mathbb{P}^U(X)\cdot M_U\left[X, f(X)\right] = \mathbb{P}^D(X)\cdot M_D\left[X, f(X)\right], \tag{4}$$

with this form being referred in the main paper as PSO *balance state*.

The second derivative of the objective $J(\epsilon)$ is:

$$\frac{d^2 J(\epsilon)}{d\epsilon^2} = \int_{\mathbb{S}_{U\cap D}} \big[-\mathbb{P}^U(X)\cdot M'_U\left[X, f(X)+\epsilon\cdot\eta(X)\right] + \\ + \mathbb{P}^D(X)\cdot M'_D\left[X, f(X)+\epsilon\cdot\eta(X)\right]\big]\cdot\eta^2(X)dX,$$

where $M'_U(X,s) \triangleq \frac{\partial M_U(X,s)}{\partial s}$ and $M'_D(X,s) \triangleq \frac{\partial M_D(X,s)}{\partial s}$.

Further, we apply the second condition from Eq. (3):

$$\left.\frac{d^2 J(\epsilon)}{d\epsilon^2}\right|_{\epsilon=0} = \int_{\mathbb{S}_{U\cap D}} \big[-\mathbb{P}^U(X)\cdot M'_U\left[X, f(X)\right] + \mathbb{P}^D(X)\cdot M'_D\left[X, f(X)\right]\big]\cdot\eta^2(X)dX.$$

For the above expression to be positive for any $\eta(X)$, $\left.\frac{d^2 J(\epsilon)}{d\epsilon^2}\right|_{\epsilon=0} > 0$, it is sufficient to satisfy the following condition at any $X \in \mathbb{S}_{U\cap D}$:

$$\big[-\mathbb{P}^U(X)\cdot M'_U\left[X, f(X)\right] + \mathbb{P}^D(X)\cdot M'_D\left[X, f(X)\right]\big] > 0,$$

or

$$\mathbb{P}^U(X)\cdot M'_U\left[X, f(X)\right] < \mathbb{P}^D(X)\cdot M'_D\left[X, f(X)\right]. \tag{5}$$

To conclude, for $f^* \in C^2(\mathbb{S}_{U\cap D})$ to be a minima of $L_{U\cap D}(f)$ it is necessary to satisfy Eq. (4) and it is sufficient to satisfy Eq. (5). These conditions have to be examined w.r.t. a particular pair of *magnitude* functions M_U and M_D to verify that a solution f^* of Eq. (4) is also satisfying Eq. (5). If Eq. (5) holds then f^* is a minima. Further, if the condition in Eq. (5) is not satisfied, the higher derivatives of $J(\epsilon)$ must be tested according to the *higher-order derivative test*. However, in practice it is typically not required since most of the PSO instances satisfy Eq. (5) at f^* (see examples in Tables 1-3 below).

■

A.2 Functional $L_{U\setminus D}$ over $\mathbb{S}_{U\setminus D}$

Herein we extend Theorem 1 from the paper, considering the input space outside of the mutual support $\mathbb{S}_{U\cap D}$. Specifically, consider the following minimization:

$$f^* \in \arg\inf_f L_{U\setminus D}(f) = -\int_{\mathbb{S}_{U\setminus D}} \mathbb{P}^U(X)\cdot \widetilde{M}_U\left[X, f(X)\right] dX$$

where f is an arbitrary function with a support in $\mathbb{S}_{U\setminus D}$, for which the above integral is well-defined. We are going to prove the following Theorem.

Theorem 2. *Depending on properties of a function M_U, for f^* to be minima of $L_{U\setminus D}$ it must satisfy:*

1. *If $\forall X \in \mathbb{S}_{U\setminus D}, t \in \mathbb{R} : M_U(X,t) > 0$, then $f^*(X) = \infty$.*
2. *If $\forall X \in \mathbb{S}_{U\setminus D}, t \in \mathbb{R} : M_U(X,t) < 0$, then $f^*(X) = -\infty$.*
3. *If $\forall X \in \mathbb{S}_{U\setminus D}, t \in \mathbb{R} : M_U(X,t) \equiv 0$, then $f^*(X)$ can be arbitrary.*
4. *Otherwise, additional analysis is required.*

The motivation behind Theorem 2 is that in many PSO instances M_U satisfies one of its conditions. In such case this Theorem can be applied to understand the PSO convergence behavior in the region $\mathbb{S}_{U\setminus D}$.

Proof. Define a new function $f_a(X) \triangleq f(X) + a$ for some scalar a and for some function f. Due to the definition of $\widetilde{M}_U$, for any a and for any f we have:

$$\Delta_a L \triangleq L_{U\setminus D}(f) - L_{U\setminus D}(f_a) = \\ = \int_{\mathbb{S}_{U\setminus D}} \mathbb{P}^U(X) \cdot \left[\int_{f(X)}^{f(X)+a} M_U(X,t)dt \right] dX. \quad (6)$$

Further, from Eq. (6) we can conclude that if M_U is a strictly positive function, $\forall X \in \mathbb{S}_{U\setminus D}, t \in \mathbb{R} : M_U(X,t) > 0$, then any $a > 0$ satisfies $\Delta_a L > 0$. Thus, in such a case $L_{U\setminus D}(f_a)$ is smaller than $L_{U\setminus D}(f)$ and hence $f(X)$ can not be a minima of $L_{U\setminus D}$ unless $\forall X \in \mathbb{S}_{U\setminus D} : f(X) = \infty$.

Due to similar arguments, in case $\forall X \in \mathbb{S}_{U\setminus D}, t \in \mathbb{R} : M_U(X,t) < 0$, any $a < 0$ satisfies $\Delta_a L > 0$. Hence, the minima of $L_{U\setminus D}$ in this case must satisfy $\forall X \in \mathbb{S}_{U\setminus D} : f(X) = -\infty$.

Finally, if $\forall X \in \mathbb{S}_{U\setminus D}, t \in \mathbb{R} : M_U(X,t) \equiv 0$, then for any a we have $\Delta_a L = 0$. Moreover, in such a case we also have $\forall X \in \mathbb{S}_{U\setminus D}, t \in \mathbb{R} : \widetilde{M}_U(X,t) \equiv 0$ and $\forall f : L_{U\setminus D}(f) \equiv 0$. Therefore, $f(X)$ can have an arbitrary outputs at $X \in \mathbb{S}_{U\setminus D}$, since the functional $L_{U\setminus D}$ does not depend on it. In practice this means that the optimization does not enforce any constraint over a learned model within a region $\mathbb{S}_{U\setminus D}$ where *up magnitude* function M_U is zero.

Otherwise, if $M_U(X,t)$ can return positive, negative and zero outputs, a further analysis of this particular *magnitude* function in the context of $L_{U\setminus D}$ needs to be done.

■

A.3 Functional $L_{D\setminus U}$ over $\mathbb{S}_{D\setminus U}$

Consider the following minimization:

$$f^* \in \arg\inf_f L_{D\setminus U}(f) = \int_{\mathbb{S}_{D\setminus U}} \mathbb{P}^D(X) \cdot \widetilde{M}_D\left[X, f(X)\right] dX$$

where f is an arbitrary function with a support in $\mathbb{S}_{D\setminus U}$, for which the above integral is well-defined. The PSO convergence behavior in the region $\mathbb{S}_{D\setminus U}$ can be summarized via the following Theorem.

Theorem 3. *Depending on properties of a function M_D, for f^* to be minima of $L_{D\setminus U}$ it must satisfy:*

1. *If $\forall X \in \mathbb{S}_{D\setminus U}, t \in \mathbb{R} : M_D(X,t) > 0$, then $f^*(X) = -\infty$.*
2. *If $\forall X \in \mathbb{S}_{D\setminus U}, t \in \mathbb{R} : M_D(X,t) < 0$, then $f^*(X) = \infty$.*
3. *If $\forall X \in \mathbb{S}_{D\setminus U}, t \in \mathbb{R} : M_D(X,t) \equiv 0$, then $f^*(X)$ can be arbitrary.*
4. *Otherwise, additional analysis is required.*

The proof of Theorem 3 is symmetric to the proof of Theorem 2 and hence omitted.

B Appendix: Instances of PSO

Many statistical methods exist whose gradient has the form of PSO gradient:

$$\nabla_\theta L_{PSO}(\theta) = - \mathop{\mathbb{E}}_{X\sim\mathbb{P}^U} M_U\left[X, f_\theta(X)\right]\cdot\nabla_\theta f_\theta(X) + \mathop{\mathbb{E}}_{X\sim\mathbb{P}^D} M_D\left[X, f_\theta(X)\right]\cdot\nabla_\theta f_\theta(X),$$

and therefore being instances of the PSO algorithm family. In Tables 1-3 we show multiple PSO instances with both their *empirical* losses (if analytically known) and their *magnitude* functions. Note that $\mathbb{P}^U$ typically represents density of the primary analyzed data. Likewise, $\mathbb{P}^D$ usually denotes an auxiliary distribution or a density of the secondary analyzed data for cases when a density ratio between two datasets is inferred. The depicted *empirical* losses are sample approximations of *PSO functional* L_{PSO} in Eq. (1):

$$L_{PSO}(\theta) \approx -\frac{1}{N_U}\sum_{i=1}^{N_U} \widetilde{M}_U\left[X_U^i, f_\theta(X_U^i)\right] + \frac{1}{N_D}\sum_{i=1}^{N_D} \widetilde{M}_D\left[X_D^i, f_\theta(X_D^i)\right], \qquad (7)$$

where we use batch size $N_U = N_D = 1$ setting to simplify and clarify the composition of the involved mathematical expressions. We also report to what the converged surface $f_\theta(X)$ is equal to. Derivation of this convergence appears below.

B.1 Deriving Convergence For PSO Instance

Given a PSO instance with a particular pair M_U and M_D, NN convergence can be derived by solving PSO *balance state*:

$$\frac{\mathbb{P}^U(X)}{\mathbb{P}^D(X)} = \frac{M_D[X, f_\theta(X)]}{M_U[X, f_\theta(X)]} \qquad (8)$$

for $f_\theta(X)$.

Table 1: PSO Instances, Part 1

Method	***Final*** $f_\theta(X)$ / ***References*** / ***Loss*** / $M_U(\cdot)$ ***and*** $M_D(\cdot)$
PSO-LDE (Log Density Estimator)	F: $\log \mathbb{P}^U(X)$ R: This paper, see main text L: unknown M_U, M_D: $\frac{\mathbb{P}^D(X)}{[[\exp f_\theta(X)]^\alpha + [\mathbb{P}^D(X)]^\alpha]^{\frac{1}{\alpha}}}, \frac{\exp f_\theta(X)}{[[\exp f_\theta(X)]^\alpha + [\mathbb{P}^D(X)]^\alpha]^{\frac{1}{\alpha}}}$
PSO-MAX	F: $\log \mathbb{P}^U(X)$ R: This paper, a limit case of PSO-LDE for $\alpha \to \infty$, see Section K.1 L: unknown M_U, M_D: $\exp\left[-\max\left[f_\theta(X) - \log \mathbb{P}^D(X), 0\right]\right]$, $\exp\left[\min\left[f_\theta(X) - \log \mathbb{P}^D(X), 0\right]\right]$
Importance Sampling	F: $\log \mathbb{P}^U(X)$ R: [1] L: $-f_\theta(X_U) + \frac{\exp[f_\theta(X_D)]}{\mathbb{P}^D(X_D)}$ M_U, M_D: $1, \frac{\exp[f_\theta(X)]}{\mathbb{P}^D(X)}$
PDF Convolution Estimation	F: $\log\left[\mathbb{P}^U(X) * \mathbb{P}^\upsilon(X)\right]$, where $*$ is a convolution operator R: This paper L: $-f_\theta(\bar{X}_U) + \frac{\exp[f_\theta(X_D)]}{\mathbb{P}^D(X_D)}$ M_U, M_D: $1, \frac{\exp[f_\theta(X)]}{\mathbb{P}^D(X)}$ * where *up* samples are produced via $\bar{X}_U = X_U + \upsilon$ with $\upsilon \sim \mathbb{P}^\upsilon(X)$
NCE	F: $\log \mathbb{P}^U(X)$ R: [6, 7, 1, 8, 9, 10] L: $\log \frac{\exp[f_\theta(X_U)] + \mathbb{P}^D(X_U)}{\exp[f_\theta(X_U)]} + \log \frac{\exp[f_\theta(X_D)] + \mathbb{P}^D(X_D)}{\mathbb{P}^D(X_D)}$ M_U, M_D: $\frac{\mathbb{P}^D(X)}{\exp[f_\theta(X)] + \mathbb{P}^D(X)}, \frac{\exp[f_\theta(X)]}{\exp[f_\theta(X)] + \mathbb{P}^D(X)}$
Polynomial	F: $\log \mathbb{P}^U(X)$ R: [1] L: $-\frac{\exp[f_\theta(X_U)]}{\mathbb{P}^D(X_U)} + \frac{1}{2} \cdot \frac{\exp[2 \cdot f_\theta(X_D)]}{[\mathbb{P}^D(X_D)]^2}$ M_U, M_D: $\frac{\exp[f_\theta(X)]}{\mathbb{P}^D(X)}, \frac{\exp[2 \cdot f_\theta(X)]}{[\mathbb{P}^D(X)]^2}$
Inverse Polynomial	F: $\log \mathbb{P}^U(X)$ R: [1] L: $\frac{1}{2} \cdot \frac{[\mathbb{P}^D(X_U)]^2}{\exp[2 \cdot f_\theta(X_U)]} - \frac{\mathbb{P}^D(X_D)}{\exp[f_\theta(X_D)]}$ M_U, M_D: $\frac{[\mathbb{P}^D(X)]^2}{\exp[2 \cdot f_\theta(X)]}, \frac{\mathbb{P}^D(X)}{\exp[f_\theta(X)]}$
Inverse Importance Sampling	F: $\log \mathbb{P}^U(X)$ R: [1] L: $\frac{\mathbb{P}^D(X_U)}{\exp[f_\theta(X_U)]} + f_\theta(X_D)$ M_U, M_D: $\frac{\mathbb{P}^D(X)}{\exp[f_\theta(X)]}, 1$

Table 2: PSO Instances, Part 2

Method	***Final*** $f_\theta(X)$ / ***References*** / ***Loss*** / $M_U(\cdot)$ ***and*** $M_D(\cdot)$
uLSIF	F: $\frac{\mathbb{P}^U(X)}{\mathbb{P}^D(X)}$ R: [11, 12, 13, 14, 15] L: $-f_\theta(X_U) + \frac{1}{2}\Big[f_\theta(X_D)\Big]^2$ M_U, M_D: 1, $f_\theta(X)$
KLIEP	F: $\frac{\mathbb{P}^U(X)}{\mathbb{P}^D(X)}$ R: [16, 17, 15] L: $-\log f_\theta(X_U) + [f_\theta(X_D) - 1]$ M_U, M_D: $\frac{1}{f_\theta(X)}$, 1
Classical GAN Critic	F: $\frac{\mathbb{P}^U(X)}{\mathbb{P}^U(X)+\mathbb{P}^D(X)}$ R: [18] L: $-\log f_\theta(X_U) - \log\Big[1 - f_\theta(X_D)\Big]$ M_U, M_D: $\frac{1}{f_\theta(X)}$, $\frac{1}{1-f_\theta(X)}$ * for $f_\theta(X) = sigmoid(h(X;\theta))$ this loss is identical to Logistic Loss in Table 3
Classical GAN Critic on log-scale	F: $\log\frac{\mathbb{P}^U(X)}{\mathbb{P}^U(X)+\mathbb{P}^D(X)}$ R: This paper L: $\frac{1}{\exp[f_\theta(X_U)]} - \log\frac{\exp[f_\theta(X_D)]}{1-\exp[f_\theta(X_D)]}$ M_U, M_D: $\frac{1}{\exp[f_\theta(X)]}$, $\frac{1}{1-\exp[f_\theta(X)]}$
Noise-Data Mixture Ratio	F: $\frac{\mathbb{P}^U(X)}{\mathbb{P}^U(X)+\mathbb{P}^D(X)}$ R: This paper L: $-f_\theta(X_U) + f_\theta(X_M)^2$, M_U, M_D: 1, $2 \cdot f_\theta(X)$ * where X_M is sampled from mixture $\frac{1}{2}\mathbb{P}^U(X) + \frac{1}{2}\mathbb{P}^D(X)$ instead of density $\mathbb{P}^D(X)$
Noise-Data Mixture Log-Ratio	F: $\log\frac{\mathbb{P}^U(X)}{\mathbb{P}^U(X)+\mathbb{P}^D(X)}$ R: This paper L: $-f_\theta(X_U) + 2 \cdot \exp[f_\theta(X_M)]$, M_U, M_D: 1, $2 \cdot \exp[f_\theta(X)]$ * where X_M is sampled from mixture $\frac{1}{2}\mathbb{P}^U(X) + \frac{1}{2}\mathbb{P}^D(X)$ instead of density $\mathbb{P}^D(X)$
Power Divergence Ratio Estimation	F: $\frac{\mathbb{P}^U(X)}{\mathbb{P}^D(X)}$ R: [17, 19] L: $-\frac{f_\theta(X_U)^\alpha}{\alpha} + \frac{f_\theta(X_D)^{\alpha+1}}{\alpha+1}$ M_U, M_D: $f_\theta(X)^{\alpha-1}$, $f_\theta(X)^\alpha$

Table 3: PSO Instances, Part 3

Method	***Final*** $f_\theta(X)$ / ***References*** / ***Loss*** / $M_U(\cdot)$ ***and*** $M_D(\cdot)$
Reversed KL	F: $\frac{\mathbb{P}^U(X)}{\mathbb{P}^D(X)}$ R: [15] L: $\frac{1}{f_\theta(X_U)} + \log f_\theta(X_D)$ M_U, M_D: $\frac{1}{f^2(X;\theta)}$, $\frac{1}{f_\theta(X)}$
Log-density Ratio	F: $\log \frac{\mathbb{P}^U(X)}{\mathbb{P}^D(X)}$ R: This paper L: $-f_\theta(X_U) + \exp[f_\theta(X_D)]$ M_U, M_D: 1, $\exp[f_\theta(X)]$
Square Loss	F: $\frac{\mathbb{P}^U(X)-\mathbb{P}^D(X)}{\mathbb{P}^U(X)+\mathbb{P}^D(X)}$ R: [19] L: $\frac{1}{2} \cdot [1 - f_\theta(X_U)]^2 + \frac{1}{2} \cdot [1 + f_\theta(X_D)]^2$ M_U, M_D: $1 - f_\theta(X)$, $1 + f_\theta(X)$
Logistic Loss	F: $\log \frac{\mathbb{P}^U(X)}{\mathbb{P}^D(X)}$ R: [19] L: $\log\big[1 + \exp[-f_\theta(X_U)]\big] + \log\big[1 + \exp[f_\theta(X_D)]\big]$ M_U, M_D: $\frac{1}{\exp[f_\theta(X)]+1}$, $\frac{1}{\exp[-f_\theta(X)]+1}$
Exponential Loss	F: $\frac{1}{2} \log \frac{\mathbb{P}^U(X)}{\mathbb{P}^D(X)}$ R: [19] L: $\exp[-f_\theta(X_U)] + \exp[f_\theta(X_D)]$ M_U, M_D: $\exp[-f_\theta(X)]$, $\exp[f_\theta(X)]$
Root Density Estimation	F: $\sqrt[d]{\mathbb{P}^U(X)}$ R: This paper L: $-f_\theta(X_U) \cdot \mathbb{P}^D(X_U) + \frac{1}{d+1} \cdot f_\theta(X_D)^{d+1} sign[f_\theta(X_D)]^{d+1}$ M_U, M_D: $\mathbb{P}^D(X)$, $f_\theta(X)^d \cdot sign[f_\theta(X)]^{d+1}$
LSGAN Critic	F: $\frac{b \cdot \mathbb{P}^U(X) + a \cdot \mathbb{P}^D(X)}{\mathbb{P}^U(X)+\mathbb{P}^D(X)}$ R: [20] L: $\frac{1}{2} \cdot [f_\theta(X_U) - b]^2 + \frac{1}{2} \cdot [f_\theta(X_D) - a]^2$ M_U, M_D: $b - f_\theta(X)$, $f_\theta(X) - a$
Kullback-Leibler Divergence	F: $1 + \log \frac{\mathbb{P}^U(X)}{\mathbb{P}^D(X)}$ R: [3] L: $-f_\theta(X_U) + \exp[f_\theta(X_D) - 1]$ M_U, M_D: 1, $\exp[f_\theta(X) - 1]$
Reverse KL Divergence	F: $-\frac{\mathbb{P}^D(X)}{\mathbb{P}^U(X)}$ R: [3] L: $-f_\theta(X_U) + [-1 - \log[-f_\theta(X_D)]]$ M_U, M_D: 1, $-\frac{1}{f_\theta(X)}$

Example 1: From Table 1 we can see that "Importance Sampling" method has $M_U\left[X, f_\theta(X)\right] = 1$ and $M_D\left[X, f_\theta(X)\right] = \frac{\exp[f_\theta(X)]}{\mathbb{P}^D(X)}$. Given that samples within the loss have densities $X_U \sim \mathbb{P}^U(X)$ and $X_D \sim \mathbb{P}^D(X)$, we can substitute the sample densities and the *magnitude* functions M_U and M_D into Eq. (8) to get:

$$\frac{\mathbb{P}^U(X)}{\mathbb{P}^D(X)} = \frac{\exp[f_\theta(X)]/\mathbb{P}^D(X)}{1} \quad \Rightarrow \quad f_\theta(X) = \log \mathbb{P}^U(X),$$

where we use equality between density ratio of the samples and ratio of *magnitude* functions to derive the final $f_\theta(X)$. Thus, in case of Importance Sampling, the surface will converge to the log density $\log \mathbb{P}^U$.

More generally, we can derive PSO convergence as following. Given a particular pair of M_U and M_D, denote PSO convergence by a transformation $T(X, z) : \mathbb{R}^n \times \mathbb{R} \to \mathbb{R}$ s.t. $f_\theta(X) = T\left[X, \frac{\mathbb{P}^U(X)}{\mathbb{P}^D(X)}\right]$. Further, we define an inverse function $T^{-1}(X, s) : \mathbb{R}^n \times \mathbb{R} \to \mathbb{R}$ s.t. $T^{-1}\left[X, T\left[X, z\right]\right] = z$ and $T\left[X, T^{-1}\left[X, s\right]\right] = s$. Then, $T^{-1}(X, s) \equiv \frac{M_D(X,s)}{M_U(X,s)}$ (see proof below). Therefore, given $T^{-1}(X, s) \triangleq \frac{M_D(X,s)}{M_U(X,s)}$ the transformation $T(X, z)$ can be retrieved by finding an inverse of $T^{-1}(X, s)$ w.r.t. s. After finding $T(X, z)$, we compute the convergence as $f_\theta(X) = T\left[X, \frac{\mathbb{P}^U(X)}{\mathbb{P}^D(X)}\right]$.

Proof. According to the definition T must also satisfy $f_\theta(X) = T\left[X, \frac{M_D(X, f_\theta(X))}{M_U(X, f_\theta(X))}\right]$ due to PSO *balance state*. By replacing $f_\theta(X)$ with a new scalar variable s, we can conclude that given a particular pair of M_U and M_D, the transformation T must satisfy $T\left[X, \frac{M_D(X,s)}{M_U(X,s)}\right] = s$. Likewise, we also have $\frac{M_D(X, f_\theta(X))}{M_U(X, f_\theta(X))} = \frac{\mathbb{P}^U(X)}{\mathbb{P}^D(X)}$. After replacing $f_\theta(X)$ with $T\left[X, \frac{\mathbb{P}^U(X)}{\mathbb{P}^D(X)}\right]$, we get $\frac{M_D(X, T\left[X, \frac{\mathbb{P}^U(X)}{\mathbb{P}^D(X)}\right])}{M_U(X, T\left[X, \frac{\mathbb{P}^U(X)}{\mathbb{P}^D(X)}\right])} = \frac{\mathbb{P}^U(X)}{\mathbb{P}^D(X)}$. Further, replacing $\frac{\mathbb{P}^U(X)}{\mathbb{P}^D(X)}$ with a new scalar variable z, we achieve $\frac{M_D(X, T[X,z])}{M_U(X, T[X,z])} = z$.

Above we see that the function $\frac{M_D(X,s)}{M_U(X,s)}$ satisfies both conditions $T\left[X, \frac{M_D(X,s)}{M_U(X,s)}\right] = s$ and $\frac{M_D(X, T[X,z])}{M_U(X, T[X,z])} = z$. Therefore, it satisfies the inverse conditions, and hence $T^{-1}(X, s) \equiv \frac{M_D(X,s)}{M_U(X,s)}$. ■

Example 2: Consider the "Polynomial" method in Table 1, with $M_U\left[X, f_\theta(X)\right] = \frac{\exp[f_\theta(X)]}{\mathbb{P}^D(X)}$ and $M_D\left[X, f_\theta(X)\right] = \frac{\exp[2 \cdot f_\theta(X)]}{[\mathbb{P}^D(X)]^2}$. Then, according to the above derivation we have $\frac{M_D(X, f_\theta(X))}{M_U(X, f_\theta(X))} = \frac{\exp[f_\theta(X)]}{\mathbb{P}^D(X)}$ and hence $T^{-1}(X, s) = \frac{\exp[s]}{\mathbb{P}^D(X)}$. In this case T^{-1} has a simple form and its invert is simply $T(X, z) = \log \mathbb{P}^D(X) + \log z$. The above T and T^{-1} are inverse of each other w.r.t. the second argument, which can be easily verified. Next, we can calculate PSO convergence as

$f_\theta(X) = T\left[X, \frac{\mathbb{P}^U(X)}{\mathbb{P}^D(X)}\right] = \log \mathbb{P}^D(X) + \log \frac{\mathbb{P}^U(X)}{\mathbb{P}^D(X)} = \log \mathbb{P}^U(X)$. Hence, "Polynomial" method converges to $\log \mathbb{P}^U(X)$.

B.2 Deriving New PSO Instance

In order to apply PSO for learning any function $T\left[X, \frac{\mathbb{P}^U(X)}{\mathbb{P}^D(X)}\right]$, the appropriate PSO instance can be derived as follows. First, we find $T^{-1}(X,s)$ by finding an inverse of $T(X,z)$ w.r.t. z. Then any pair of *magnitudes* satisfying $\frac{M_D(X,s)}{M_U(X,s)} = T^{-1}\left[X,s\right]$ will converge at $f_\theta(X) = T\left[X, \frac{\mathbb{P}^U(X)}{\mathbb{P}^D(X)}\right]$ since:

$$T^{-1}\left[X, f_\theta(X)\right] = \frac{M_D\left[X, f_\theta(X)\right]}{M_U\left[X, f_\theta(X)\right]} = \frac{\mathbb{P}^U(X)}{\mathbb{P}^D(X)} \quad \Rightarrow$$
$$\Rightarrow \quad T\left[X, \frac{\mathbb{P}^U(X)}{\mathbb{P}^D(X)}\right] = T\left[X, T^{-1}\left[X, f_\theta(X)\right]\right] = f_\theta(X).$$

where we used properties of inverse functions.

Hence, once T^{-1} was derived, we can use a simple choice $M_D\left[X,s\right] = T^{-1}\left[X,s\right]$ and $M_U\left[X,s\right] = 1$ in order to converge to the desired target $f_\theta(X) = T\left[X, \frac{\mathbb{P}^U(X)}{\mathbb{P}^D(X)}\right]$. However, typically such first choice of *magnitude* functions will be sub-optimal if for example M_D is an unbounded function. If this is the case, we can derive a new pair of bounded *magnitude* functions by multiplying the old pair by the same factor $q\left[X,s\right]$, see the main paper for more details. Likewise, in a similar manner we can also derive a new PSO instances to infer any function of density $T\left[X, \mathbb{P}^U(X)\right]$.

Example 3: Consider a scenario where we would like to infer $f_\theta(X) = \frac{\mathbb{P}^U(X) - \mathbb{P}^D(X)}{\mathbb{P}^U(X) + \mathbb{P}^D(X)}$, similarly to "Square Loss" method in Table 3. Considering only points $\{X : \mathbb{P}^D(X) > 0\}$, we can rewrite our objective as $f_\theta(X) = \frac{\frac{\mathbb{P}^U(X)}{\mathbb{P}^D(X)} - 1}{\frac{\mathbb{P}^U(X)}{\mathbb{P}^D(X)} + 1}$ and hence the required PSO convergence is given by $T(X,z) = \frac{z-1}{z+1}$. Further, its inverse function is given by $T^{-1}(X,s) = \frac{1+s}{1-s}$. Therefore, *magnitude* functions must satisfy $\frac{M_D(X, f_\theta(X))}{M_U(X, f_\theta(X))} = \frac{1+f_\theta(X)}{1-f_\theta(X)}$. One choice for such *magnitudes* is $M_U\left[X, f_\theta(X)\right] = 1 - f_\theta(X)$ and $M_D\left[X, f_\theta(X)\right] = 1 + f_\theta(X)$, just like in the "Square Loss" method [19], although other variants with the same PSO *balance state* can be easily constructed. For instance, we can use $M_U\left[X, f_\theta(X)\right] = \frac{1-f_\theta(X)}{D(X, f_\theta(X))}$ and $M_D\left[X, f_\theta(X)\right] = \frac{1+f_\theta(X)}{D(X, f_\theta(X))}$ with $D(X, f_\theta(X)) \triangleq |1 - f_\theta(X)| + |1 + f_\theta(X)|$ instead. Such normalization by function $D(\cdot)$ does not change the convergence, yet it produces bounded *magnitude* functions that are typically more stable during the optimization. Furthermore, recall that we considered only points within support of $\mathbb{P}^D(X)$. Outside of this support, any $X : \mathbb{P}^U(X) > 0$ will push the NN surface according to the rules implied by M_U; note also that M_U changes signs at $f_\theta(X) = 1$, with force direction always directed toward it. Therefore, at such points the convergence will be $f_\theta(X) = 1$. Finally at points outside of both supports there is no optimization performed, and hence theoretically nothing moves there. Yet, these areas of $f_\theta(X)$ will be affected by pushes at the training points, according to the elasticity properties of NN expressed via a Neural Tangent Kernel (NTK) (see [21]).

Further, density of samples X_U and X_D within PSO loss can be altered from common choices of two considered density functions $\mathbb{P}^U$ and $\mathbb{P}^D$, to infer other target functions. For example, in "Noise-Data Mixture Ratio" from Table 2 instead of $X_D \sim \mathbb{P}^D(X)$ we sample $X_M \sim \frac{1}{2}\mathbb{P}^U(X) + \frac{1}{2}\mathbb{P}^D(X)$. That is, X_M's density is a mixture of two original densities with equal weights. Then, by substituting sample densities and appropriate *magnitude* functions $M_U\left[X, f_\theta(X)\right] = 1$ and $M_D\left[X, f_\theta(X)\right] = 2 \cdot f_\theta(X)$ into *balance state* equilibrium we will get:

$$\frac{\mathbb{P}^U(X)}{\frac{1}{2}\mathbb{P}^U(X) + \frac{1}{2}\mathbb{P}^D(X)} = \frac{2 \cdot f_\theta(X)}{1} \quad \Rightarrow \quad f_\theta(X) = \frac{\mathbb{P}^U(X)}{\mathbb{P}^U(X) + \mathbb{P}^D(X)}.$$

Hence, the "Noise-Data Mixture Ratio" infers the same target function as the Classical GAN Critic loss in Table 2, and can be used as its alternative. Therefore, the additional freedom is acquired by considering different sampling strategies in PSO framework.

In Tables 1-3, we report previously discovered methods and also introduce several new losses for inference of different stochastic modalities of data, as the demonstration of usage and usefulness of the general PSO formulation.

C Appendix: Proof of Unnormalized Model Family in [1] being a strict subgroup of PSO

The family of estimators in [1] is defined by the objective function:

$$L(\theta, \{X_U^i\}, \{X_D^i\}) = -\frac{1}{N_U}\sum_{i=1}^{N_U} g_1\left[\frac{\exp(f_\theta(X_U^i))}{\mathbb{P}^D(X_U^i)}\right] + \frac{1}{N_D}\sum_{i=1}^{N_D} g_2\left[\frac{\exp(f_\theta(X_D^i))}{\mathbb{P}^D(X_D^i)}\right], \tag{9}$$

where points X_U^i and X_D^i are sampled from approximated data density $\mathbb{P}^U$ and auxiliary distribution $\mathbb{P}^D$ respectively. Further, $f_\theta(X)$ is an estimated model parametrized by weight vector θ. The $g_1[\cdot]$ and $g_2[\cdot]$ are nonlinear functions such that

$$\frac{g_2'[q]}{g_1'[q]} = q. \tag{10}$$

When the above condition holds, the model approximates target log density $f_\theta(X) = \log \mathbb{P}^U(X)$.

Comparing Eq. (9) with the *empirical* PSO loss in Eq. (7), we can conclude that the above algorithm family from [1] is contained inside PSO family, where the *magnitude* functions are defined as

$$M_U(X,s) = \frac{\partial g_1\Big[r(X,s)\Big]}{\partial r(X,s)} \cdot \frac{\exp(s)}{\mathbb{P}^D(X)} = g_1'[r(X,s)] \cdot r(X,s),$$

$$M_D(X,s) = \frac{\partial g_2\Big[r(X,s)\Big]}{\partial r(X,s)} \cdot \frac{\exp(s)}{\mathbb{P}^D(X)} = g_2'[r(X,s)] \cdot r(X,s),$$

$$r(X,s) \triangleq \frac{\exp(s)}{\mathbb{P}^D(X)}. \tag{11}$$

The opposite relation, PSO being contained in algorithm family [1], does not hold since PSO instances are not limited to estimate only $\log \mathbb{P}^U(X)$ (see Appendix B).

Further, by applying PSO principles to the the above *magnitude* functions we can see that PSO *balance state* is achieved when:

$$\frac{\mathbb{P}^U(X)}{\mathbb{P}^D(X)} = \frac{M_D[X, f_\theta(X)]}{M_U[X, f_\theta(X)]} = \frac{g_1'[r(X, f_\theta(X))]}{g_2'[r(X, f_\theta(X))]} = r(X, f_\theta(X)) = \frac{\exp(f_\theta(X))}{\mathbb{P}^D(X)}$$
$$\Rightarrow \quad f_\theta(X) = \log \mathbb{P}^U(X),$$

where we used identities from Eq. (10) and Eq. (11). Thus, the *balance state* of PSO agrees with conclusions made in [1].

■

D Appendix: Proof of Bregman Divergence Framework in [2] being a strict subgroup of PSO

The family of estimators in [2] is defined by the objective functional:

$$L(f) = \int \left[-\Psi(f(X)) + \Psi'(f(X)) \cdot f(X) - \Psi'(f(X)) \cdot h(X)\right] d\mu, \tag{12}$$

which minimizes separable Bregman divergence between the target function $h(X)$ and the approximation model $f(X)$, and where $\Psi : \mathbb{R} \to \mathbb{R}$ is twice-differentiable and strictly convex function:

$$\Psi''(s) > 0. \tag{13}$$

Further, μ is a nondecreasing weighting function.

In order to prove that the above framework is a subgroup of PSO, we will reformulate the functional in Eq. (12) as an instance of the more general *PSO functional* in Eq. (1). To this end, we will find the appropriate *magnitude* functions of the loss in Eq. (12). Assuming for simplicity that μ is a constant weighting function, we can rewrite the above loss as:

$$L(f) = \int \left[-\Psi(f(X)) + \Psi'(f(X)) \cdot f(X) - \Psi'(f(X)) \cdot h(X)\right] dX =$$
$$= \int -\mathbb{P}^U(X) \cdot \frac{\Psi'(f(X)) \cdot h(X)}{\mathbb{P}^U(X)} + \mathbb{P}^D(X) \cdot \frac{-\Psi(f(X)) + \Psi'(f(X)) \cdot f(X)}{\mathbb{P}^D(X)} dX =$$
$$= \int -\mathbb{P}^U(X) \cdot \widetilde{M}_U\left[X, f(X)\right] + \mathbb{P}^D(X) \cdot \widetilde{M}_D\left[X, f(X)\right] dX,$$

where we divided various terms into two groups, belonging to densities $\mathbb{P}^U(X)$ and $\mathbb{P}^D(X)$. Such separation was also done in the original paper (see Eq. (9) in [2]), where $\mathbb{P}^U(X)$ typically represented density of the analyzed data and $\mathbb{P}^D(X)$ - an auxiliary (noise) distribution.

The above loss has a form of *PSO functional* in Eq. (1) for *magnitude* functions:

$$M_U(X, s) = \Psi''(s) \cdot \frac{h(X)}{\mathbb{P}^U(X)}, \tag{14}$$

$$M_D(X, s) = \Psi''(s) \cdot \frac{s}{\mathbb{P}^D(X)}. \tag{15}$$

Further, note that the above $M_U(\cdot)$ in Eq. (14) contains $\mathbb{P}^U(X)$ and $h(x)$ that typically are unknown a priori. Yet, in a usual setting these terms are canceling each other out.

From the above we can see that the estimation family in [2] is sub-family of PSO for *magnitude* functions defined in Eq. (14-15) that depend on an arbitrary strictly positive function $\Psi''(s) > 0$. The opposite relation, PSO being contained in the framework [2], does not hold since not all PSO instances can be expressed via Eq. (14-15). For example, the *magnitudes* from Square Loss in Table 3 can not be expressed as Eq. (14-15) for any $\Psi''(s)$. This is true also for other PSO instances.

Furthermore, additional attention needs to be given to the conditions required by both Bergman and PSO frameworks over feasible pairs of *magnitude* functions. In [2] the main requirement is the strict positiveness in Eq. (13) while PSO requirement is summerized as the second condition of Theorem 1:

$$\mathbb{P}^U(X) \cdot M'_U\left[X, f^*(X)\right] < \mathbb{P}^D(X) \cdot M'_D\left[X, f^*(X)\right]. \tag{16}$$

For M_U and M_D defined as in Eq. (14-15) both requirements are identical. This can be shown by deriving Eq. (13) from Eq. (16) as follows. First, the derivatives of Eq. (14-15) are:

$$M'_U(X, s) = \frac{\partial M_U(X, s)}{\partial s} = \Psi'''(s) \cdot \frac{h(X)}{\mathbb{P}^U(X)},$$

$$M'_D(X, s) = \frac{\partial M_D(X, s)}{\partial s} = \frac{1}{\mathbb{P}^D(X)} \left[\Psi'''(s) \cdot s + \Psi''(s)\right].$$

Further, introducing these identities into Eq. (16) will lead to:

$$\mathbb{P}^U(X) \cdot \Psi''' [f^*(X)] \cdot \frac{h(X)}{\mathbb{P}^U(X)} < \mathbb{P}^D(X) \cdot \frac{1}{\mathbb{P}^D(X)} [\Psi''' [f^*(X)] \cdot f^*(X) + \Psi'' [f^*(X)]] ,$$

$$\Psi''' [f^*(X)] \cdot h(X) < \Psi''' [f^*(X)] \cdot f^*(X) + \Psi'' [f^*(X)] ,$$

$$\Psi''' [f^*(X)] \cdot [h(X) - f^*(X)] < \Psi'' [f^*(X)] .$$

Since $f^*(X) = h(X)$, we will get:

$$\Psi'' [f^*(X)] > 0,$$

which leads to Eq. (13). Likewise, it is possible to derive Eq. (16) from Eq. (13) by following the above deduction in the backward order.

To summarize, the algorithm family in [2] is a strict subset of more general PSO. Furthermore, PSO formulation is more flexible and is much more intuitive.

■

E Appendix: Proof of f-Divergence Framework in [3] being a strict subgroup of PSO

The family of f-divergence estimators in [3] is defined by the loss (see Eq. (9) in the original paper):

$$L(\theta) = - \underset{X \sim \mathbb{P}^U}{\mathbb{E}} f_\theta(X) + \underset{X \sim \mathbb{P}^D}{\mathbb{E}} F^*[f_\theta(X)], \tag{17}$$

where we slightly changed the original syntax to support the format of this paper. Upon convergence, the Eq. (17) will approximate *F-divergence*:

$$D_F(\mathbb{P}^U || \mathbb{P}^D) = \int \mathbb{P}^D(X) \cdot F\left(\frac{\mathbb{P}^U(X)}{\mathbb{P}^D(X)}\right),$$

where $F(\cdot)$ is a *generator function* that defines a particular *F-divergence* variant. Further, $F^*(\cdot)$ from Eq. (17) is the *convex conjugate* of $F(\cdot)$, also known as *Fenchel conjugate*.

We can see that Eq. (17) is a particular instance of *PSO functional* in Eq. (1) for the following *magnitude* functions:

$$M_U(X, s) = 1,$$

$$M_D(X, s) = F^{*\prime}[s].$$

Considering the above, the framework from [3] is contained inside PSO family. The opposite relation, PSO being contained in the framework [3], does not hold since PSO instances are not constrained to the case where $M_U(X, s) = 1$.

■

F Appendix: Learning Probabilistic Occupancy Map via PSO Framework

Below we present a more formal and complete derivation of PSO probabilistic occupancy mapping presented in the main paper.

First, consider the setting described in the paper where we denote by $S \subset \mathbb{R}^d$ the subset of d-dimensional space that we want to map. Here d can have value of 2 or 3, and S can represent the entire space of the considered indoor/outdoor environment (i.e. navigated-through a building). Further, denote by $\Omega \subseteq S$ the space that was observed by any of the acquired lidar scans.

The training dataset $\mathcal{D} = \{X_i, Y_i\}_{i=1}^{N}$ is given, with $X_i \in \Omega$ being the observed location (2D or 3D) and $Y_i \in \{\mathbf{f}, \mathbf{o}\} \triangleq \{free, occupied\}$ being occupancy label. We can consider $\mathcal{D}$ to be implicitly sampled from the joint $\mathbb{P}(X, Y)$. Likewise, $\mathcal{D}$ defines marginal probabilities $\mathbb{P}(X)$ over X and $\mathbb{P}(Y)$ over Y, by considering either only $\mathcal{X} = \{X_i\}_{i=1}^{N}$ or only $\mathcal{Y} = \{Y_i\}_{i=1}^{N}$. Also note that $\mathbb{P}(X)$ can be interpreted as a likelihood of X being observed during gathering of lidar scans, with $\forall X \notin \Omega : \mathbb{P}(X) = 0$. Further, $\mathbb{P}(Y)$ can be inferred from $\mathcal{Y}$ via $\mathbb{P}(Y = \mathbf{f}) = \frac{N_{\mathbf{f}}}{N}$ and $\mathbb{P}(Y = \mathbf{o}) = \frac{N_{\mathbf{o}}}{N}$, with $N_{\mathbf{f}}$ being number of samples $Y_i = \mathbf{f}$ and $N_{\mathbf{o}}$ - number of samples $Y_i = \mathbf{o}$.

For the reasons described in the main paper, here we aim to infer the following modality:

$$J(X) \triangleq \mathbb{P}(X) \cdot \left[\mathbb{P}(Y = \mathbf{o}|X) - \mathbb{P}(Y = \mathbf{f}|X)\right]. \tag{18}$$

For this purpose, we construct two datasets $\mathcal{X}_{\mathbf{o}} = \{X_{\mathbf{o}}^i\}_{i=1}^{N_{\mathbf{o}}}$ and $\mathcal{X}_{\mathbf{f}} = \{X_{\mathbf{f}}^i\}_{i=1}^{N_{\mathbf{f}}}$, with first dataset containing each location $X_i \in \mathcal{D}$ with $Y_i = \mathbf{o}$, and the second - the rest of the locations in $\mathcal{D}$. Note that according to the above considered implicit distributions $X_{\mathbf{o}}^i$ is distributed along $\mathbb{P}(X|Y = \mathbf{o}) = \frac{\mathbb{P}(X, Y=\mathbf{o})}{\mathbb{P}(Y=\mathbf{o})}$, and $X_{\mathbf{f}}^i$ - along $\mathbb{P}(X|Y = \mathbf{f}) = \frac{\mathbb{P}(X, Y=\mathbf{f})}{\mathbb{P}(Y=\mathbf{f})}$, with supports of both distributions being contained within Ω. Finally, we construct one more dataset $\mathcal{X}_{\mathcal{U}} = \{X_{\mathcal{U}}^i\}_{i=1}^{N_{\mathcal{U}}}$ with $N_{\mathcal{U}}$ points sampled from $\mathbb{P}^{\mathcal{U}}(X)$ - a uniform distribution over the entire S.

The proposed PSO-based approach applies the following update to a parameter vector θ:

$$\begin{aligned} d\theta = -\frac{1}{N_{\mathbf{o}}} \sum_{i=1}^{N_{\mathbf{o}}} M_{\mathbf{o}} \left[X_{\mathbf{o}}^i, f_\theta(X_{\mathbf{o}}^i)\right] \cdot \nabla_\theta f_\theta(X_{\mathbf{o}}^i) + \\ + \frac{1}{N_{\mathbf{f}}} \sum_{i=1}^{N_{\mathbf{f}}} M_{\mathbf{f}} \left[X_{\mathbf{f}}^i, f_\theta(X_{\mathbf{f}}^i)\right] \cdot \nabla_\theta f_\theta(X_{\mathbf{f}}^i) + \\ + \frac{1}{N_{\mathcal{U}}} \sum_{i=1}^{N_{\mathcal{U}}} M_{\mathcal{U}} \left[X_{\mathcal{U}}^i, f_\theta(X_{\mathcal{U}}^i)\right] \cdot \nabla_\theta f_\theta(X_{\mathcal{U}}^i), \end{aligned} \tag{19}$$

$$M_{\mathbf{o}}(X, f_\theta(X)) = \frac{N_{\mathbf{o}}}{N_{\mathbf{o}} + N_{\mathbf{f}}} \mathbb{P}^{\mathcal{U}}(X),$$

with

$$M_{\mathbf{f}}(X, f_\theta(X)) = \frac{N_{\mathbf{f}}}{N_{\mathbf{o}} + N_{\mathbf{f}}} \mathbb{P}^{\mathcal{U}}(X),$$
$$M_{\mathcal{U}}(X, f_\theta(X)) = f_\theta(X).$$

Note that $M_{\mathbf{o}}(\cdot)$ and $M_{\mathbf{f}}(\cdot)$ always return positive values. Therefore, Eq. (19) pushes $f_\theta(X)$ at $X_{\mathbf{o}}^i$ *up* and at $X_{\mathbf{f}}^i$ - *down*, similarly to the original PSO equation. Further, $M_{\mathcal{U}}(\cdot)$ is positive when $f_\theta(X) > 0$ and is negative when $f_\theta(X) < 0$. Hence, the applied force at $X_{\mathcal{U}}^i$ is always pushing NN towards $f_\theta(X) = 0$.

More formally, consider $f_\theta^*(X)$ at the convergence. For any $X : f_\theta^*(X) > 0$, the PSO *balance state* at X is satisfied when:

$$\begin{aligned} &M_{\mathbf{o}}(X, f_\theta(X)) \cdot \mathbb{P}(X|Y = \mathbf{o}) = \\ &\qquad = M_{\mathbf{f}}(X, f_\theta(X)) \cdot \mathbb{P}(X|Y = \mathbf{f}) + M_{\mathcal{U}}(X, f_\theta(X)) \cdot \mathbb{P}^{\mathcal{U}}(X). \end{aligned}$$

where LHS represents *up* forces at X, and RHS - *down* forces. After replacing each term with its definition, this equation turns to be:

$$\begin{aligned} &\frac{N_{\mathbf{o}}}{N_{\mathbf{o}} + N_{\mathbf{f}}} \mathbb{P}^{\mathcal{U}}(X) \cdot \frac{\mathbb{P}(X, Y = \mathbf{o})}{\mathbb{P}(Y = \mathbf{o})} = \\ &\qquad = \frac{N_{\mathbf{f}}}{N_{\mathbf{o}} + N_{\mathbf{f}}} \mathbb{P}^{\mathcal{U}}(X) \cdot \frac{\mathbb{P}(X, Y = \mathbf{f})}{\mathbb{P}(Y = \mathbf{f})} + f_\theta(X) \cdot \mathbb{P}^{\mathcal{U}}(X), \end{aligned}$$

$$\frac{N_{\mathbf{o}}}{N_{\mathbf{o}} + N_{\mathbf{f}}} \cdot \frac{\mathbb{P}(X, Y = \mathbf{o})}{\mathbb{P}(Y = \mathbf{o})} = \frac{N_{\mathbf{f}}}{N_{\mathbf{o}} + N_{\mathbf{f}}} \cdot \frac{\mathbb{P}(X, Y = \mathbf{f})}{\mathbb{P}(Y = \mathbf{f})} + f_\theta(X),$$
$$\mathbb{P}(X, Y = \mathbf{o}) = \mathbb{P}(X, Y = \mathbf{f}) + f_\theta(X),$$
$$f_\theta(X) = \mathbb{P}(X, Y = \mathbf{o}) - \mathbb{P}(X, Y = \mathbf{f}) = \mathbb{P}(X) \cdot [\mathbb{P}(Y = \mathbf{o}|X) - \mathbb{P}(Y = \mathbf{f}|X)] = J(X).$$

Likewise, considering any $X : f_\theta^*(X) \leq 0$, the PSO *balance state* at X is satisfied when:

$$\begin{aligned} &M_{\mathbf{o}}(X, f_\theta(X)) \cdot \mathbb{P}(X|Y = \mathbf{o}) + M_{\mathcal{U}}(X, f_\theta(X)) \cdot \mathbb{P}^{\mathcal{U}}(X) = \\ &\qquad = M_{\mathbf{f}}(X, f_\theta(X)) \cdot \mathbb{P}(X|Y = \mathbf{f}), \end{aligned}$$

where term of $X_{\mathcal{U}}^i$ changed sides since now its *magnitude* function is negative and its force is directed *up*. Solving the above equation for f_θ, we will again deduce $f_\theta(X) = J(X)$. Therefore, the described by Eq. (19) approach will converge to Eq. (18).

Furthermore, we can find a "loss" form of the proposed PSO method in Eq. (19) since its *magnitude* functions have analytical anti-derivatives. Specifically, Eq. (19) can be considered as a gradient w.r.t. θ of the following loss:

$$\begin{aligned} L(\theta) = &-\frac{1}{N_{\mathbf{o}}} \sum_{i=1}^{N_{\mathbf{o}}} \frac{N_{\mathbf{o}}}{N_{\mathbf{o}} + N_{\mathbf{f}}} \mathbb{P}^{\mathcal{U}}(X_{\mathbf{o}}^i) \cdot f_\theta(X_{\mathbf{o}}^i) + \\ &\qquad + \frac{1}{N_{\mathbf{f}}} \sum_{i=1}^{N_{\mathbf{f}}} \frac{N_{\mathbf{f}}}{N_{\mathbf{o}} + N_{\mathbf{f}}} \mathbb{P}^{\mathcal{U}}(X_{\mathbf{f}}^i) \cdot f_\theta(X_{\mathbf{f}}^i) + \frac{1}{N_{\mathcal{U}}} \sum_{i=1}^{N_{\mathcal{U}}} \frac{1}{2} \left[f_\theta(X_{\mathcal{U}}^i) \right]^2, \end{aligned}$$

which in its turn is a sample approximation of:

$$
\begin{aligned}
L(\theta) = \int -\mathbb{P}(X|Y=\mathbf{o}) \cdot \frac{N_{\mathbf{o}}}{N_{\mathbf{o}}+N_{\mathbf{f}}} \mathbb{P}^{\mathcal{U}}(X) \cdot f_\theta(X) + \\
+ \mathbb{P}(X|Y=\mathbf{f}) \cdot \frac{N_{\mathbf{f}}}{N_{\mathbf{o}}+N_{\mathbf{f}}} \mathbb{P}^{\mathcal{U}}(X) \cdot f_\theta(X) + \frac{1}{2} \mathbb{P}^{\mathcal{U}}(X) \cdot [f_\theta(X)]^2 \, dX = \\
= \int \mathbb{P}^{\mathcal{U}}(X) \cdot \Big[- \big[\mathbb{P}(X, Y=\mathbf{o}) - \mathbb{P}(X, Y=\mathbf{f})\big] \cdot f_\theta(X) + \frac{1}{2} [f_\theta(X)]^2 \Big] dX = \\
= \frac{1}{2} \int \mathbb{P}^{\mathcal{U}}(X) \cdot \Big[f_\theta(X) - \big[\mathbb{P}(X, Y=\mathbf{o}) - \mathbb{P}(X, Y=\mathbf{f})\big] \Big]^2 dX - \\
- \frac{1}{2} \int \mathbb{P}^{\mathcal{U}}(X) \cdot \big[\mathbb{P}(X, Y=\mathbf{o}) - \mathbb{P}(X, Y=\mathbf{f})\big]^2 dX.
\end{aligned}
$$

Taking into account that the last term is independent of f_θ, this loss can be replaced by:

$$
L(\theta) = \int \mathbb{P}^{\mathcal{U}}(X) \cdot \Big[f_\theta(X) - \big[\mathbb{P}(X, Y=\mathbf{o}) - \mathbb{P}(X, Y=\mathbf{f})\big] \Big]^2 dX. \tag{20}
$$

whose minimizer is obviously written in Eq. (18).

Above we can observe that PSO allowed us to create a new probabilistic loss for statistical occupancy mapping by only considering the various force terms in Eq. (19) and verifying their convergence according to PSO *balance state*. Apparently, it is also possible to derive the very same method by considering the loss $L(\theta)$ in Eq. (20) in the first place. However, from $L(\theta)$'s definition it is not obvious that it can be even computed/approximated, due to unknown $\mathbb{P}(X, Y=\mathbf{o})$ and $\mathbb{P}(X, Y=\mathbf{f})$. In contrast, the thermodynamical paradigm of PSO leads to a very simple and systematic solution.

G Appendix: NN Architecture and Block-diagonal layers

In the main paper we apply two NN architectures to represent a model $f_\theta(X)$. The first is a simple fully-connected (FC) NN with N_L layers of size S each and the second contains N_L layers with first and last layers being FC and the inner $N_L - 2$ layers being block-diagonal (BD). That is, each BD layer is identical to FC layer of appropriate size, where the weight matrix is enforced to have a block-diagonal form, with overall N_B such blocks, and size of each block being $S_B \times S_B$. Such block-structure separates various directions in parameter space $\mathbb{R}^{|\theta|}$, thus producing effect of preconditioning over the optimization problem as was shown in [22]. Further, it enforces gradients at different training points to span more directions inside $\mathbb{R}^{|\theta|}$, which makes the surface $f_\theta(X)$ to be more flexible/*expressive*.

BD-NN has enhanced approximation properties vs. FC-NN, as is shown in additional experiments below. Therefore the main paper mostly considers only experiments where $f_\theta(X)$ is BD-NN . A memory-efficient implementation of BD layers can be found in Section of 6.1 [22].

H Appendix: $LSQR$ Divergence

Here we will prove that $LSQR$ evaluation metric, considered in the main paper, is an actual statistical divergence. First, log-pdf squared error (LSQR) divergence is defined as:

$$LSQR(\mathbb{P}, \mathbb{Q}) = \int \mathbb{P}(X) \cdot \left[\log \mathbb{P}(X) - \log \mathbb{Q}(X)\right]^2 dX,$$

where $\mathbb{P}$ is pdf over compact support $\Omega \in \mathbb{R}^n$; $\mathbb{Q}$ is normalized or unnormalized model whose support is also Ω.

According to definition of the statistical divergence, LSQR must satisfy $\forall \mathbb{P}, \mathbb{Q} : LSQR(\mathbb{P}, \mathbb{Q}) \geq 0$ and $LSQR(\mathbb{P}, \mathbb{Q}) = 0 \Leftrightarrow \mathbb{P} = \mathbb{Q}$. Obviously, this is the case. Hence, $LSQR(\mathbb{P}, \mathbb{Q})$ is statistical divergence.

Also, note again that $LSQR(\mathbb{P}, \mathbb{Q})$ measures discrepancy between pdf $\mathbb{P}$ and model $\mathbb{Q}$ where the latter must not be normalized. Therefore, it can be used to evaluate PSO-based methods since they are only approximately normalized. However, such evaluation is only possible when $\mathbb{P}$ is known analytically.

In the main paper $LSQR$ is measured between the target density, defined as $\mathbb{P}^U$ in the paper, and the pdf estimator $\bar{\mathbb{P}}_\theta$ produced by some method in the following way:

$$LSQR(\mathbb{P}^U, \bar{\mathbb{P}}_\theta) = \frac{1}{N} \sum_{i=1}^{N} \left[\log \mathbb{P}^U(X_U^i) - \log \bar{\mathbb{P}}_\theta(X_U^i)\right]^2,$$

where $\{X_U^i\}_{i=1}^N$ are testing points, sampled from $\mathbb{P}^U$, that were not involved in estimation of $\bar{\mathbb{P}}_\theta$.

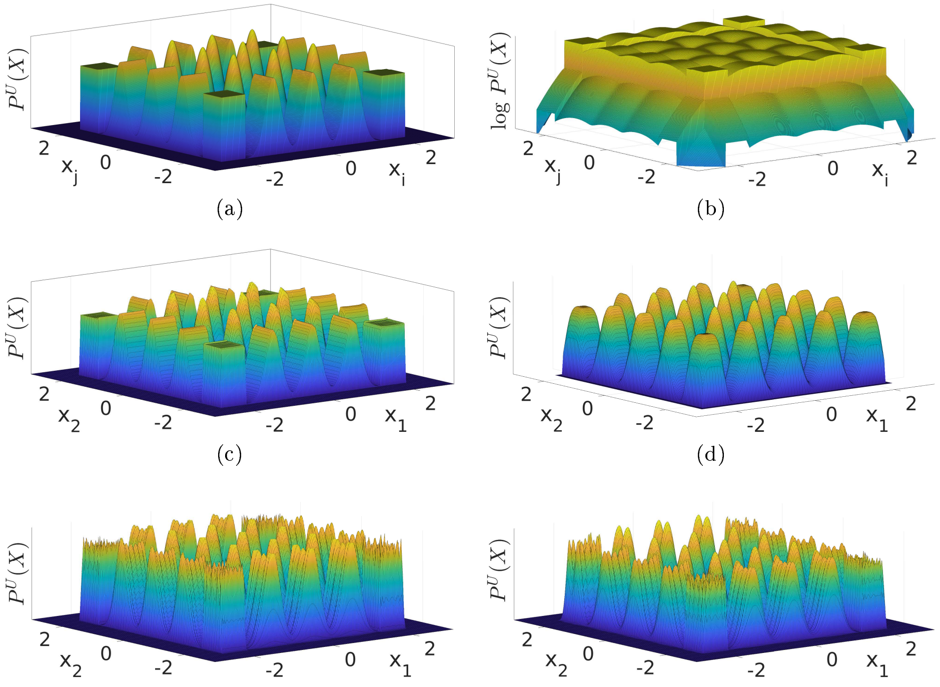
I Appendix: Learned *Columns* Distribution

Here we provide additional information about the *Columns* density, learned in our experiments. It is 20-dimensional distribution defined as:

$$\mathbb{P}^U(x_1, \ldots, x_{20}) = \prod_{i=1}^{20} p(x_i), \tag{21}$$

where $p(\cdot)$ is a mixture distribution with 5 components:

- $Uniform(-2.3, -1.7)$
- $\mathcal{N}(-1.0, std = 0.2)$
- $\mathcal{N}(0.0, std = 0.2)$



(a)

(b)

(c)

(d)

(e)

(f)

Figure 1: (a) Illustration of *Columns* distribution. Every slice of its pdf function $\mathbb{P}(x_i, x_j) = \mathbb{P}^U(0, \dots, 0, x_i, 0, \dots, 0, x_j, 0, \dots, 0)$ in Eq. (21) contains 25 modes of different shape. Overall, this distribution has 5^{20} modes. (b) Logarithm of pdf slice in (a). (c)-(f) Density estimations for methods (c) PSO-LDE (d) ScM (d) MADE and (f) MAF. The depicted slice is $\mathbb{P}(x_1, x_2) = \mathbb{P}^U(x_1, x_2, 0, \dots, 0)$, with x_1 and x_2 forming a grid of points in the first two dimensions.

- $\mathcal{N}(1.0, std = 0.2)$
- $Uniform(1.7, 2.3)$

Each component has weight 0.2. This distribution has overall 5^{20} modes, making the structure of its pdf surface very challenging to learn. For illustration see Figure 1.

Likewise, in Figure 1 are depicted the estimations of the *Columns* density made by different inference techniques, including PSO-LDE and baseline methods.

J Appendix: Technical Setup for Learning *Columns*

All the experiments were done using Adam optimizer [23] since it showed better convergence rate compared to stochastic GD. The used Adam hyper-parameters are $\beta_1 = 0.75$, $\beta_2 = 0.999$ and $\epsilon = 10^{-10}$. Each experiment optimization is performed for 300000 iterations, which typically takes about one hour to run on a GeForce GTX 1080 Ti GPU card. The batch size is $N_U = N_D = 1000$. During each iteration the next batch of *up* points $\{X_U^i\}_{i=1}^{1000}$ is retrieved from the training dataset, and the next batch of *down* points $\{X_D^i\}_{i=1}^{1000}$ is sampled from *down* density $\mathbb{P}^D$. For *Columns* distribution we use a Uniform distribution as $\mathbb{P}^D$, with minimal support that covers all available samples from $\mathbb{P}^U$. Next, the optimizer updates the weights vector θ according to PSO gradient (Eq. (1) in the paper), where the *magnitude* functions are specified according to a specific PSO instance. The applied learning rate is 0.0035. We keep it constant for the first 40000 iterations and then exponentially decay it down to a minimum learning rate of $3 \cdot 10^{-9}$. Further, in all our models we use Leaky-Relu as the non-linearity activation function. Weights are initialized via the popular Xavier initialization [24].

Each model is learned 5 times; we report its mean accuracy and the standard deviation. Further, the reported $LSQR$ error is computed over 10^5 testing samples from $\mathbb{P}^U$ that were not a part of the training dataset. Size of the training dataset is 10^8. Importantly, even for such large datasets, the training and testing datasets do not share any of their points. The reason for this is that probability to sample the same point twice from evaluated continuous distributions is zero.

K Appendix: Additional Experiments for *Columns* Distribution

Here we provide additional experimental evaluation of PSO for density estimation of *Columns* Distribution, which does not appear in the main text due to space limit.

K.1 Evaluation of Additional PSO Instances

Here we perform pdf learning using additional not considered in the main paper PSO instances for an infinite dataset setting (10^8 samples from $\mathbb{P}^U$), and compare their performance. The $f_\theta(X)$ is represented via BD-NN, with $N_L = 6$, $N_B = 50$ and $S_B = 64$.

First we evaluate the Importance Sampling (IS) method from Table 1. Its *magnitude* functions are not bounded, and it may cause instability during optimization. It produced an error $LSQR = 0.46 \pm 0.14$ which indeed is much inferior to PSO-LDE with bounded *magnitudes*, that achieved 0.057 ± 0.004. Additionally, we applied PSO-MAX defined in Table 1, which is a limit case of PSO-LDE with $\alpha \rightarrow \infty$. It received $LSQR = 0.058 \pm 0.002$ which is slightly worse than the average of PSO-LDE. Overall, our experiments, including the evaluation of *Transformed Columns* below, show that PSO instances with bounded *magnitudes* have superior performance in a pdf inference task.

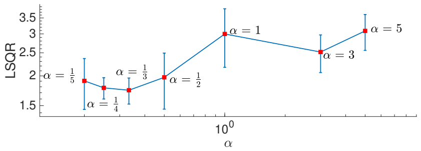


Figure 2: Evaluation of PSO-LDE for density estimation. NN architecture is FC-NN with 4 layers of size 1024. The $LSQR$ error is reported, along with its empirical standard deviation, for different values of the hyper-parameter α.

K.2 NN Architectures Evaluation

Here we compare performance of various NN architectures for the pdf estimation task. Specifically, we apply PSO-LDE with different values of α where the used NN architecture is now FC-NN, with $N_L = 4$ and $S = 1024$. In Figure 2 we show $LSQR$ for different α, where again we can see that $\alpha = \frac{1}{4}$ (and now also $\alpha = \frac{1}{3}$) performs better than other values of α. On average, $LSQR$ error is around 2.5 which is significantly higher than 0.057 for BD architecture. Note also that the BD network used in Section K.1 is twice smaller than FC network, containing only 902401 weights in BD vs 2121729 in FC, yet it produced significantly better accuracy.

Further, we performed learning with a BD architecture, but with increasing number of blocks N_B. The dataset setting is infinite, with 10^8 samples from *Columns* distribution $\mathbb{P}^U$. For N_B taking values $\{100, 125, 150, 175, 200\}$, in Figure 3 we can see that with bigger N_B there is an improvement in approximation accuracy. This can be explained by the fact that bigger N_B produces bigger number of independent transformation channels inside NN; with more such channels the NN becomes highly flexible. Further, in a setting of infinite dataset such high NN flexibility is desirable, and leads to a higher approximation accuracy.

Likewise, we experiment with number of layers N_L to see how the network depth of a BD architecture affects accuracy of pdf inference. In Figure 4 we see that deeper networks allow us to further decrease $LSQR$ error to around 0.03. Also, we can see that at some point increasing N_L causes only a slight error improvement. Thus, increasing N_L beyond that point is not beneficial, since for very small error reduction we will pay with higher computational cost due to increasing size of θ.

Further, we evaluate BD performance for different sizes of blocks S_B, as reported in Figure 5. Here we do not see a monotonic error decrease that was observed for N_B and N_L. The error is big for S_B below 32 or above 160. This probably can be explained as follows. The BD-NN can be viewed as a sum of N_B independent transformation channels. For a small block size S_B each independent channel has too narrow width that is not enough to properly transfer the required signal from NN input to output. Yet, surprisingly it still can achieve a very good approximation, yielding $LSQR$ error of 0.075 for $S_B =$

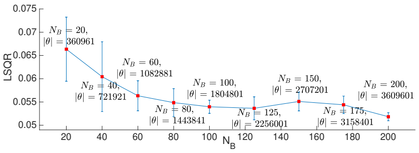
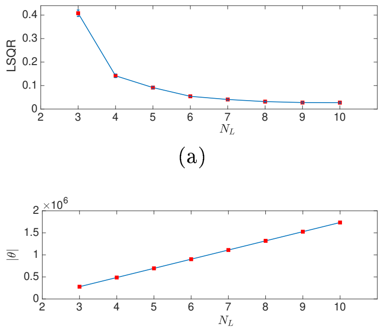


Figure 3: Evaluation of PSO-LDE for estimation of *Columns* distribution, where NN architecture is block-diagonal with 6 layers and block size $S_B = 64$. The number of blocks N_B is changing. The applied loss is PSO-LDE with $\alpha = \frac{1}{4}$. For different values of N_B the $LSQR$ error is reported. Each model was learned only once, due to long computational time required for each optimization.



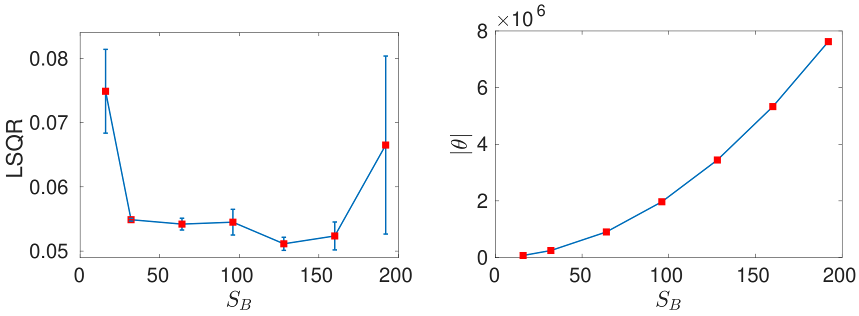
(a)

(b)

Figure 4: Evaluation of PSO-LDE for estimation of *Columns* distribution, where NN architecture is block-diagonal with number of blocks $N_B = 50$ and block size $S_B = 64$. The number of layers N_L is changing from 3 to 10. The applied loss is PSO-LDE with $\alpha = \frac{1}{4}$. (a) For different values of N_L we report $LSQR$ and its empirical standard deviation. (b) Size of θ for each value of N_L is depicted.

16 with only 72001 overall weights, which is very impressive for such small network. Further, for a large block size S_B each independent channel becomes too wide, with information from too many various regions in $\mathbb{R}^n$ passing through it. This in turn causes interference between different regions, similarly to what is going on inside a regular FC network.

Overall, our experiments show that the BD architecture produces superior performance over FC architecture, which can be explained by implicit preconditioning of parameter space. Further, the block size S_B around 64 produces better performance in general. Yet, its small values are very attractive since they yield a network of small size with appropriate computational benefits, with relatively small error increase. Moreover, in an infinite dataset setting we observed that further increase of NN flexibility by increasing N_B or N_L yields even better approximation accuracy.



(a) (b)

Figure 5: Evaluation of PSO-LDE for density estimation. NN architecture is block-diagonal with 6 layers and number of blocks $N_B = 50$. The block size S_B is taking values $\{16, 32, 64, 96, 128, 160, 192\}$. The applied loss is PSO-LDE with $\alpha = \frac{1}{4}$. (a) For different values of S_B we report $LSQR$ and its empirical standard deviation. (b) Size of θ for each value of S_B is depicted.

L Appendix: Experiments for *Transformed Columns* Distribution

Here we will infer a 20D *Transformed Columns* distribution, which is produced from *Columns* by multiplying a random variable $X \sim \mathbb{P}^{Clmns}$ (defined in Eq. (21)) with a dense invertible matrix A which enforces correlation between different dimensions. Its pdf can be written as:

$$\mathbb{P}^{TrClmns}(X) \triangleq \frac{1}{abs\left[\log\det A\right]} \mathbb{P}^{Clmns}(A^{-1} \cdot X),$$

where A was randomly generated under constraint of having determinant 1, to keep the volume of sampled points the same. Its generated entries are:

$$A = \begin{pmatrix}
0.190704135 & -0.103706818 & 0.287080085 & -0.224115607 & -0.0296220322 & -0.200017067 & 0.107420472 & 0.145939799 & -0.21151047 & 0.290260037 & -0.109926743 & -0.138214861 & 0.0739138407 & -0.173910764 & -0.158279581 & -0.138856972 & -0.512741096 & -0.0894244111 & -0.465075432 & 0.138203328 \\
0.254157669 & 0.00972120677 & -0.425381258 & -0.165311223 & -0.0732519109 & 0.316785766 & -0.0651314216 & -0.153534853 & -0.294111694 & -0.29775682 & -0.285308807 & 0.12228138 & -0.11072477 & -0.0955035066 & 0.00942833152 & -0.252106498 & -0.40791915 & 0.14810246 & 0.219848116 & 0.0170452831 \\
0.213837698 & 0.435577026 & -0.0250319656 & 0.297552176 & 0.14030724 & 0.17703815 & 0.179253182 & -0.0710653564 & 0.0507340489 & 0.235684257 & 0.33391508 & -0.40609493 & -0.197930666 & -0.322136609 & -0.146204612 & -0.164195869 & -0.0672063429 & 0.138970011 & 0.121959537 & -0.150499483 \\
0.229971825 & 0.235126465 & 0.158420112 & -0.0223857003 & 0.28726862 & 0.133325519 & -0.352231148 & 0.403451114 & -0.0161522847 & -0.0193998935 & 0.0455239109 & 0.162348247 & -0.108553457 & -0.126312424 & 0.354695129 & -0.18792775 & 0.101384726 & -0.360466001 & 0.0797850581 & 0.338497968 \\
0.168709235 & 0.0770914534 & -0.178165495 & -0.0661545928 & 0.323673824 & -0.216202087 & 0.475022027 & 0.129138946 & -0.0173273685 & -0.305472906 & -0.187205709 & -0.0359566427 & -0.216262061 & -0.0428909211 & -0.354871726 & 0.299579441 & 0.157803981 & -0.190573488 & 0.0280581785 & 0.281767192 \\
0.238265575 & 0.14326338 & 0.323225051 & 0.101182056 & 0.222068216 & -0.42470829 & -0.132128709 & -0.203895612 & -0.38640015 & -0.193494524 & 0.15340899 & 0.0919709633 & -0.150512414 & 0.257608882 & 0.182536036 & 0.223159059 & -0.177186352 & 0.198776911 & 0.130192805 & -0.201342323 \\
0.279579926 & -0.162797928 & -0.0586375005 & -0.211398563 & -0.178520507 & -0.0154862203 & -0.371463145 & 0.187233788 & -0.19506691 & 0.119455231 & -0.202696444 & -0.581504491 & -0.160227753 & 0.135365515 & -0.104123883 & 0.128700751 & 0.341221005 & 0.04776047 & 0.0956523695 & -0.105994479 \\
0.220148935 & -0.2672238 & 0.259200965 & -0.348342982 & 0.155930129 & 0.019456055 & 0.136887538 & -0.264686829 & 0.24360206 & 0.117285157 & -0.002458813 & 0.241121126 & -0.423370778 & 0.01867544 & -0.0604651676 & -0.397344315 & 0.273889032 & 0.122122216 & 0.00146222648 & -0.125711392 \\
0.274834233 & 0.105359473 & 0.135585987 & -0.19681974 & -0.0573374634 & 0.272574082 & 0.0741237415 & 0.0122961299 & 0.331281502 & -0.33605766 & 0.271171768 & -0.205618232 & 0.213806318 & 0.535961439 & -0.103513592 & -0.0402908492 & -0.199964863 & -0.203164108 & -0.0657867077 & -0.0601941311 \\
0.191457811 & -0.234983092 & 0.180777146 & 0.293340161 & -0.0365900702 & 0.214595475 & 0.236285794 & 0.256954426 & 0.152660465 & 0.0502990912 & -0.34518931 & 0.0711801608 & -0.128073093 & -0.0785011303 & 0.222291071 & 0.252921604 & -0.174804068 & -0.217786875 & 0.114727637 & -0.492946359 \\
0.223304218 & -0.0257257131 & -0.413566391 & 0.172073687 & 0.326658323 & -0.251109479 & -0.188027609 & 0.334468985 & 0.161164411 & 0.150742708 & -0.0165671389 & 0.232902107 & 0.21926384 & 0.194760411 & -0.235966926 & -0.210659524 & -0.00475631985 & 0.172782694 & -0.148866241 & -0.300869538 \\
0.243388344 & 0.243683991 & -0.121589184 & -0.187437781 & 0.0578409285 & -0.27767391 & 0.118032949 & -0.346625601 & 0.292226942 & 0.243065097 & -0.348108205 & -0.141543891 & 0.37738745 & -0.0330943339 & 0.390702777 & 0.02656679 & 0.0292203222 & -0.0520266353 & 0.165078206 & 0.0596683021 \\
0.230141811 & 0.0247039796 & -0.160146528 & 0.276454478 & -0.576178718 & -0.329822403 & 0.25548108 & 0.14013065 & -0.0103114951 & -0.162178682 & 0.0992219866 & 0.012231234 & -0.189306474 & 0.0930990418 & 0.282716476 & -0.306277128 & 0.142988577 & -0.00844652753 & -0.179308874 & 0.106622196 \\
0.232700446 & -0.203358735 & -0.0404096623 & 0.0651155788 & -0.0876307509 & -0.191362905 & -0.350703084 & -0.292134625 & 0.189041139 & -0.344463408 & 0.182873652 & 0.00843615229 & 0.103973847 & -0.52808934 & -0.153969081 & 0.0920142117 & 0.0274541623 & -0.30781733 & -0.126023284 & -0.157587751 \\
0.199527977 & 0.204898606 & -0.0450516851 & -0.270830109 & -0.122669508 & 0.203103159 & 0.237732868 & 0.012741777 & -0.462819922 & 0.114896626 & 0.187819585 & 0.289174359 & 0.310962024 & -0.0974490092 & -0.00676534358 & 0.104883625 & 0.361388813 & -0.151993161 & -0.133748076 & -0.307757495 \\
0.175957009 & 0.114449497 & 0.0205901599 & -0.309807805 & -0.235656127 & 0.00557129199 & -0.0424724202 & 0.307742364 & 0.323261613 & -0.00822297356 & 0.171003077 & 0.20675015 & -0.0593725006 & -0.232062167 & 0.0481299972 & 0.415888833 & -0.102561506 & 0.524594092 & 0.0346260903 & 0.106458345 \\
0.223633992 & -0.408554226 & -0.309022911 & 0.131333656 & 0.253238692 & 0.245874156 & 0.0563174345 & -0.203397977 & -0.0592004594 & 0.179237363 & 0.275909961 & -0.0735710289 & -0.0681467794 & 0.0974804318 & 0.340180897 & 0.282145806 & -0.0050587548 & 0.0683927742 & -0.329991983 & 0.246896636 \\
0.215583387 & 0.212554612 & 0.289443614 & 0.366636519 & -0.034606719 & 0.274097732 & -0.127438501 & -0.179674304 & 0.0238811214 & -0.126780371 & -0.419167616 & 0.0954798128 & 0.126317847 & 0.0330201935 & -0.166833728 & 0.039242297 & 0.231007699 & 0.29173484 & -0.381428263 & 0.182896273 \\
0.210225414 & -0.42839288 & 0.239002756 & 0.174593297 & 0.0372987005 & -0.000156767089 & 0.183766314 & 0.132302659 & -0.124351715 & -0.0656208547 & 0.126841957 & -0.0486788409 & 0.471440044 & -0.112599645 & -0.0828678102 & -0.173846575 & 0.0871950404 & 0.210646003 & 0.459392924 & 0.242375288 \\
0.220671036 & 0.0270987729 & -0.0325243953 & 0.176295746 & -0.299408603 & -0.00284534072 & -0.133474574 & -0.186385408 & 0.0205827025 & 0.45209226 & 0.0583508015 & 0.321753008 & -0.120986377 & 0.217766867 & -0.359227919 & 0.180578462 & -0.115873733 & -0.281505687 & 0.296303031 & 0.245563455
\end{pmatrix}.$$

As we will see, the obtained results for this more sophisticated distribution have similar trends to results of *Columns*. The technical setup of a training process is identical to Section J.

First, we evaluate PSO-LDE for different values of α on the density inference task. The applied model is BD with 6 layers, number of blocks $N_B = 50$ and block size $S_B = 64$. The dataset size is 10^8 and $\mathbb{P}^D$ is Gaussian distribution $\sim \mathcal{N}(\mu, \Sigma)$. The mean vector μ is equal to the mean of samples from $\mathbb{P}^U$; the Σ is diagonal matrix whose non-zero values are empirical variances for each

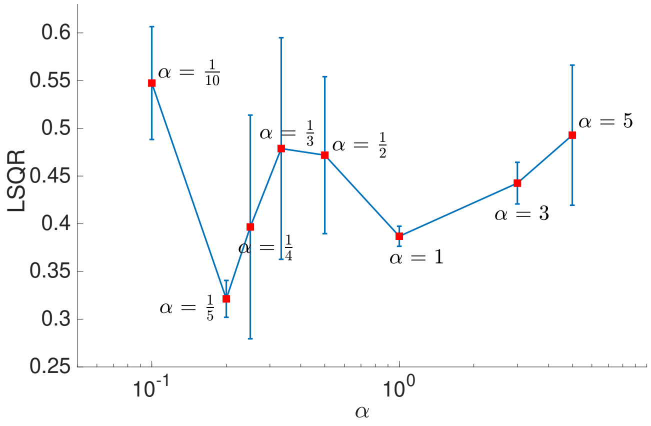


Figure 6: Evaluation of PSO-LDE for estimation of *Transformed Columns* distribution, where NN architecture is block-diagonal with 6 layers, number of blocks $N_B = 50$ and block size $S_B = 64$. The *down* density $\mathbb{P}^D$ is Gaussian. For different values of hyper-parameter α, *LSQR* error is reported, along with its empirical standard deviation. As observed, the target surface is accurately approximated.

dimension of available *up* samples. In Figure 6 we can see that overall achieved accuracy is high, yet it is worse than the results for *Columns* distribution. Such difference can be explained by support mismatch between $\mathbb{P}^U$ and $\mathbb{P}^D$ densities. While *Columns* and Uniform densities in the paper's experiments are relatively aligned and close to each other, the *Transformed Columns* and Gaussian have bounded ratio $\frac{\mathbb{P}^U(X)}{\mathbb{P}^D(X)}$ only around their mean point. In far away regions such ratio becomes too big/small, causing PSO inaccuracies. Hence, we argue that better choice of $\mathbb{P}^D$ would yield better accuracy.

Further, we also can see that in case of *Transformed Columns* PSO-LDE with $\alpha = \frac{1}{5}$ achieves the smallest error, with its *LSQR* being 0.32 ± 0.02. Furthermore, averaged over all values of α, PSO-LDE achieved *LSQR* 0.442 ± 0.095. Additionally, we evaluated the Importance Sampling (IS) method and PSO-MAX from Table 1. PSO-MAX achieved 0.54 ± 0.088, which is slightly worse than PSO-LDE, similarly to what was observed for *Columns*. Moreover, the IS fails entirely, producing very large error 26.63 ± 0.86. Furthermore, in order to stabilize its learning process we were required to reduce the learning rate from 0.0035 to 0.0001. Hence, here we can see again the superiority of bounded *magnitude* functions over not bounded.

Additionally, we evaluated several different NN architectures for *Transformed Columns* distribution. In Figure 7 we report performance for FC networks. As can be observed, the FC architecture has higher error w.r.t. BD architecture in Figure 6. Likewise, PSO-LDE with α around $\frac{1}{4}$ performs better.

Further, in Figure 8 we experiment with BD architecture for different values

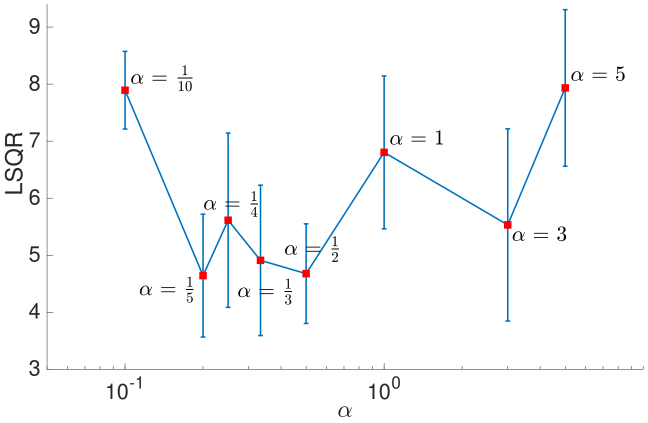


Figure 7: Evaluation of PSO-LDE for estimation of *Transformed Columns* distribution, where NN architecture is fully-connected with 4 layers of size 1024 (see Section G). For different values of hyper-parameter α, *LSQR* error is reported, along with its empirical standard deviation. Again, the small values of α (around $\frac{1}{4}$) have lower error.

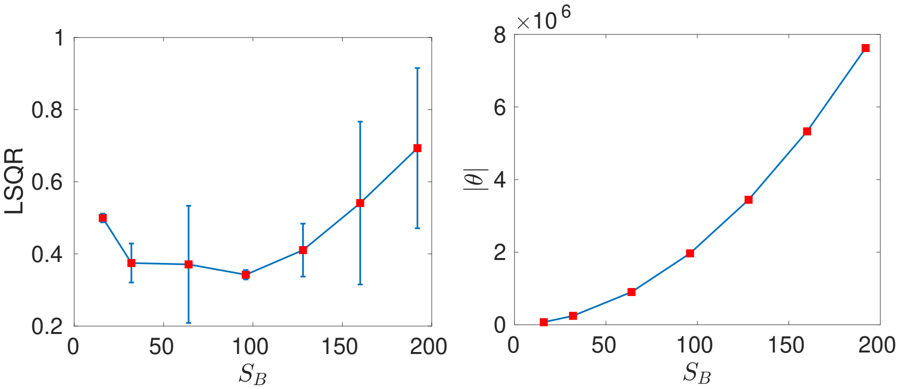
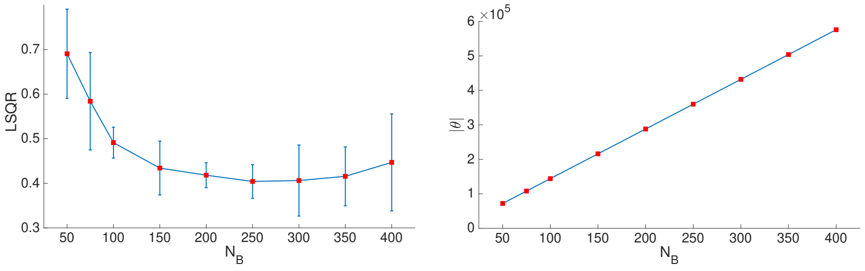


Figure 8: Evaluation of PSO-LDE for estimation of *Transformed Columns* distribution, where NN architecture is block-diagonal with 6 layers and number of blocks $N_B = 50$ (see Section G). The block size S_B is taking values $\{16, 32, 64, 96, 128, 160, 192\}$. The applied loss is PSO-LDE with $\alpha = \frac{1}{5}$. For different values of S_B we report *LSQR* error and its empirical standard deviation. Additionally, in the second column we depict the size of θ for each value of S_B.

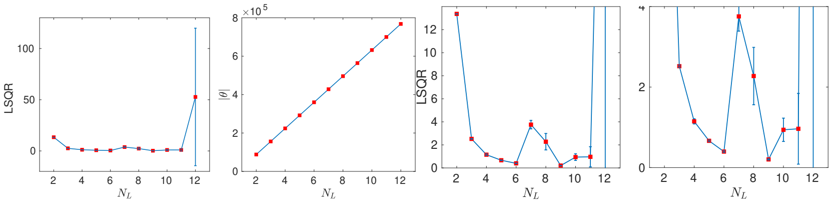
of block size S_B. Similarly to what was observed for *Columns* distribution, also here too small/large block size has worsen accuracy.



(a)

(b)

Figure 9: Evaluation of PSO-LDE for estimation of *Transformed Columns* distribution, where NN architecture is block-diagonal with 6 layers and block size $S_B = 16$ (see Section G). The number of blocks N_B is changing. The applied loss is PSO-LDE with $\alpha = \frac{1}{5}$. For different values of N_B, (a) $LSQR$ error and (b) number of weights $|\theta|$ are reported.



(a)

(b)

Figure 10: Evaluation of PSO-LDE for estimation of *Columns* distribution, where NN architecture is block-diagonal with number of blocks $N_B = 50$ and block size $S_B = 64$ (see Section G). The number of layers N_L is changing from 3 to 10. The applied loss is PSO-LDE with $\alpha = \frac{1}{5}$. (a) For different values of N_L we report $LSQR$ and its empirical standard deviation. Additionally, in the second column we depict size of θ for each value of N_L. (b) Zoom of $LSQR$ in (a).

We also perform additional experiments for BD architecture where this time we use small blocks, with $S_B = 16$. In Figure 9 we see that for such networks the bigger number of blocks per layer N_B, that is bigger number of transformation channels, yields better performance. However, we also see that when number of these channels grows considerably (above 250), the accuracy starts dropping. Furthermore, no matter how big the number of blocks N_B is, the models with small blocks ($S_B = 16$) in Figure 9 produced inferior results w.r.t. model with big blocks ($S_B = 64$, see Figure 6 for $\alpha = \frac{1}{5}$).

Further, we evaluated the same small block network for different number of layers N_L. In Figure 10 we can see that when N_L grows from 1 to 6, the overall performance is getting better. Yet, for larger number of layers the performance trend is inconsistent. When N_L is between 7 and 11, some values of N_L are better than the other, with no evident improvement pattern for large N_L. Moreover, for $N_L = 12$ the error grows significantly. More so, we see that 2 out of 5 runs of this setting didn't succeed to learn at all due to zero gradient. Thus, the most likely conclusion for this setting is that for too deep network the signal from input fails to reach its end, which is known issue in DL domain. Also, from our experiments it follows that the learning still can succeed, depending on the initialization of network weights.

Furthermore, along with the above inconsistency that can occur for too deep networks, for $N_L = 9$ in Figure 10 we still received our best results for *Transformed Columns*, with averaged *LSQR* being 0.204. That is, even when BD network uses blocks of small size ($S_B = 16$), it still can produce superior performance if it is deep enough. Besides, the zero-gradient issue may be tackled by adding shortcut connections between various layers. We shall leave it as direction for future research.

In overall, in our experiments we saw that PSO-LDE with values of α around $\frac{1}{4}$ produced better performance. Further, IS with unbounded *magnitude* functions can not be applied at all for far away densities. BD architecture showed again significantly higher accuracy over FC networks. Block size $S_B = 96$ produced better inference. Finally, for BD networks with small blocks ($S_B = 16$) the bigger number of blocks N_B and the bigger number of layers N_L perform better. However, at some point the increase in both can cause bigger error.

M Appendix: Technical Setup for Learning Joint Distribution over Pose and CNN Features

The technical setup for this task was very similar to the one described in Section J, with few differences. Overall size of training and testing datasets was 100000 and 70000 respectively. BD architecture was used to represent $f_\theta(X)$, with $N_L = 8$, $N_B = 100$ and $S_B = 64$, and with Elu non-linearity activation function. Each experiment optimization is performed for 300000 iterations, that was divided into two stages. During the first stage with 80000 iterations we performed PSO-LDE training where we used GMM-50 as $\mathbb{P}^D$; with a constant learning rate 0.0035; and with batch size $N_U = N_D = 600$. During the second stage with 220000 iterations we performed PSO-LDE training where we used GMM-300 as $\mathbb{P}^D$; with a constant learning rate $1e-5$; and with batch size $N_U = N_D = 100$.

N Appendix: Technical Setup for Learning Probabilistic Occupancy Map

The technical setup for this task was very similar to the one described in Section J, with few differences.

Scenario 1 Overall size of training dataset was $N_{\mathbf{o}} = 7347$ and $N_{\mathbf{f}} = 3.5 \cdot 10^6$. Note that "free" locations are sampled uniformly from scanned areas, and we can construct a dataset of these points as large as we want. FC architecture was used to represent $f_\theta(X)$, with 8 layers of size 256 each, and with Elu non-linearity activation function. Optimization was performed for 300000 iterations, and took around 30 minutes to run on a GeForce GTX 1080 Ti GPU card. Size of mini batches was 300.

Scenario 2 Overall size of training dataset was $N_{\mathbf{o}} = 674563$ and $N_{\mathbf{f}} = 4.3 \cdot 10^6$. FC architecture was used to represent $f_\theta(X)$, with 10 layers of size 256 each, and with Elu non-linearity activation function. Optimization was performed for 300000 iterations, and took around one hour to run on a GeForce GTX 1080 Ti GPU card. Size of mini batches was 1000.

References

[1] M. Pihlaja, M. Gutmann, and A. Hyvarinen, "A family of computationally efficient and simple estimators for unnormalized statistical models," *arXiv preprint arXiv:1203.3506*, 2012.

[2] M. Gutmann and J.-i. Hirayama, "Bregman divergence as general framework to estimate unnormalized statistical models," *arXiv preprint arXiv:1202.3727*, 2012.

[3] S. Nowozin, B. Cseke, and R. Tomioka, "f-gan: Training generative neural samplers using variational divergence minimization," in *Advances in Neural Information Processing Systems*, 2016, pp. 271–279.

[4] J. Jost, J. Jost, and X. Li-Jost, *Calculus of variations.* Cambridge University Press, 1998, vol. 64.

[5] R. Courant and D. Hilbert, "Chapter iv. the calculus of variations," *Methods of Mathematical Physics. New York: Interscience Publishers*, pp. 164–274, 1953.

[6] N. A. Smith and J. Eisner, "Contrastive estimation: Training log-linear models on unlabeled data," in *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics.* Association for Computational Linguistics, 2005, pp. 354–362.

[7] M. Gutmann and A. Hyvärinen, "Noise-contrastive estimation: A new estimation principle for unnormalized statistical models," in *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, 2010, pp. 297–304.

[8] A. Mnih and Y. W. Teh, "A fast and simple algorithm for training neural probabilistic language models," *arXiv preprint arXiv:1206.6426*, 2012.

[9] A. Mnih and K. Kavukcuoglu, "Learning word embeddings efficiently with noise-contrastive estimation," in *Advances in Neural Information Processing Systems (NIPS)*, 2013, pp. 2265–2273.

[10] M. U. Gutmann and A. Hyvärinen, "Noise-contrastive estimation of unnormalized statistical models, with applications to natural image statistics," *J. of Machine Learning Research*, vol. 13, no. Feb, pp. 307–361, 2012.

[11] T. Kanamori, S. Hido, and M. Sugiyama, "A least-squares approach to direct importance estimation," *Journal of Machine Learning Research*, vol. 10, no. Jul, pp. 1391–1445, 2009.

[12] M. Yamada, T. Suzuki, T. Kanamori, H. Hachiya, and M. Sugiyama, "Relative density-ratio estimation for robust distribution comparison," in *Advances in Neural Information Processing Systems (NIPS)*, 2011, pp. 594–602.

[13] M. Sugiyama, T. Suzuki, and T. Kanamori, *Density ratio estimation in machine learning*. Cambridge University Press, 2012.

[14] H. Nam and M. Sugiyama, "Direct density ratio estimation with convolutional neural networks with application in outlier detection," *IEICE TRANSACTIONS on Information and Systems*, vol. 98, no. 5, pp. 1073–1079, 2015.

[15] M. Uehara, I. Sato, M. Suzuki, K. Nakayama, and Y. Matsuo, "b-gan: Unified framework of generative adversarial networks," 2016.

[16] M. Sugiyama, T. Suzuki, S. Nakajima, H. Kashima, P. von Bünau, and M. Kawanabe, "Direct importance estimation for covariate shift adaptation," *Annals of the Institute of Statistical Mathematics*, vol. 60, no. 4, pp. 699–746, 2008.

[17] M. Sugiyama, T. Suzuki, and T. Kanamori, "Density-ratio matching under the bregman divergence: a unified framework of density-ratio estimation," *Annals of the Institute of Statistical Mathematics*, vol. 64, no. 5, pp. 1009–1044, 2012.

[18] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Advances in Neural Information Processing Systems (NIPS)*, 2014, pp. 2672–2680.

[19] A. Menon and C. S. Ong, "Linking losses for density ratio and class-probability estimation," in *Intl. Conf. on Machine Learning (ICML)*, 2016, pp. 304–313.

[20] X. Mao, Q. Li, H. Xie, R. Y. Lau, Z. Wang, and S. P. Smolley, "Least squares generative adversarial networks," in *Intl. Conf. on Computer Vision (ICCV)*. IEEE, 2017, pp. 2813–2821.

[21] A. Jacot, F. Gabriel, and C. Hongler, “Neural tangent kernel: Convergence and generalization in neural networks,” in *Advances in Neural Information Processing Systems (NIPS)*, 2018, pp. 8571–8580.

[22] D. Kopitkov and V. Indelman, “General probabilistic surface optimization and log density estimation,” *arXiv preprint arXiv:1903.10567*, 2019.

[23] D. P. Kingma and J. Ba, “Adam: A method for stochastic optimization,” *arXiv preprint arXiv:1412.6980*, 2014.

[24] X. Glorot and Y. Bengio, “Understanding the difficulty of training deep feedforward neural networks,” in *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, 2010, pp. 249–256.