Learning Deep Kernels for Exponential Family Densities: Supplementary material

A. DKEFs can be normalized

Proposition 2. *Consider the kernel $k(\boldsymbol{x},\boldsymbol{y}) = \kappa(\boldsymbol{\phi}(\boldsymbol{x}),\boldsymbol{\phi}(\boldsymbol{y}))$, where κ is a kernel such that $\kappa(\boldsymbol{a},\boldsymbol{a}) \le L_\kappa \|\boldsymbol{a}\|^2 + C_\kappa$ and $\boldsymbol{\phi}$ a function such that $\|\boldsymbol{\phi}(\boldsymbol{x})\| \le L_{\boldsymbol{\phi}}\|\boldsymbol{x}\| + C_{\boldsymbol{\phi}}$. Let $q_0(\boldsymbol{x}) = Q\, r_0(\boldsymbol{V}^{-1}(\boldsymbol{x}-\boldsymbol{\mu}))$, where $Q > 0$ is any scalar and r_0 is a product of independent generalized Gaussian densities, with each $\beta_d > 1$:*

$$r_0(\boldsymbol{z}) = \prod_{d=1}^{D} \frac{\beta_d}{2\,\Gamma\left(\frac{1}{\beta_d}\right)} \exp\left(-|z_d|^{\beta_d}\right).$$

(For example, $\mathcal{N}(\boldsymbol{\mu},\boldsymbol{\Sigma})$ for strictly positive definite $\boldsymbol{\Sigma}$ could be achieved with $\beta_d = 2$ and $\boldsymbol{V}$ the Cholesky factorization of $\boldsymbol{\Sigma}$.) Then, for any function f in the RKHS $\mathcal{H}$ corresponding to k,

$$\int \exp(f(\boldsymbol{x}))\, q_0(\boldsymbol{x})\,\mathrm{d}\boldsymbol{x} < \infty.$$

Proof. First, we have that $f(\boldsymbol{x}) = \langle f, k(\boldsymbol{x},\cdot)\rangle_{\mathcal{H}} \le \|f\|_{\mathcal{H}}\sqrt{k(\boldsymbol{x},\boldsymbol{x})}$, and

$$k(\boldsymbol{x},\boldsymbol{x}) = \kappa(\boldsymbol{\phi}(\boldsymbol{x}),\boldsymbol{\phi}(\boldsymbol{x})) \le L_\kappa\|\boldsymbol{\phi}(\boldsymbol{x})\|^2 + C_\kappa \le L_\kappa(L_{\boldsymbol{\phi}}\|\boldsymbol{x}\|^2 + C_{\boldsymbol{\phi}}) + C_\kappa.$$

Combining these two yields

$$f(\boldsymbol{x}) \le \|f\|_{\mathcal{H}}\sqrt{L_\kappa L_{\boldsymbol{\phi}}\|\boldsymbol{x}\|^2 + L_\kappa C_{\boldsymbol{\phi}} + C_\kappa} \le \|f\|_{\mathcal{H}}\sqrt{L_\kappa L_{\boldsymbol{\phi}}}\|\boldsymbol{x}\| + \|f\|_{\mathcal{H}}\sqrt{L_\kappa C_{\boldsymbol{\phi}} + C_\kappa} \le C_0 + C_1\|\boldsymbol{x}\|,$$

defining $C_1 := \|f\|_{\mathcal{H}}\sqrt{L_\kappa L_{\boldsymbol{\phi}}}$, $C_0 := \|f\|_{\mathcal{H}}\sqrt{L_\kappa C_{\boldsymbol{\phi}} + C_\kappa}$.

Let $\boldsymbol{z} = \boldsymbol{V}^{-1}(\boldsymbol{x}-\boldsymbol{\mu})$, and let C_r be the normalizing constant of r_0, $C_q := \prod_{d=1}^{D}\frac{\beta_d}{2\alpha_d\,\Gamma\left(\frac{1}{\beta_d}\right)}$. Then

$$\begin{aligned}
\int \exp(f(\boldsymbol{x}))\, q_0(\boldsymbol{x})\,\mathrm{d}\boldsymbol{x} &\le \int \exp\left(C_0 + C_1\|\boldsymbol{x}\|\right) q_0(\boldsymbol{x})\mathrm{d}\boldsymbol{x} \\
&= Q\exp(C_0)\,\mathbb{E}_{\boldsymbol{z}\sim r_0}\left[\exp\left(C_1\|\boldsymbol{V}\boldsymbol{z}+\boldsymbol{\mu}\|\right)\right] \\
&\le Q\exp(C_0 + C_1\|\boldsymbol{\mu}\|)\,\mathbb{E}_{\boldsymbol{z}\sim r_0}\left[\exp\left(C_1\|\boldsymbol{V}\|\|\boldsymbol{z}\|\right)\right] \\
&\le Q\exp(C_0 + C_1\|\boldsymbol{\mu}\|)\,\mathbb{E}_{\boldsymbol{z}\sim r_0}\left[\exp\left(C_1\|\boldsymbol{V}\|\sum_{d=1}^{D}|z_d|\right)\right] \\
&= Q\exp(C_0 + C_1\|\boldsymbol{\mu}\|)\prod_{d=1}^{D}\mathbb{E}_{\boldsymbol{z}\sim r_0}\left[\exp\left(C_1\|\boldsymbol{V}\||z_d|\right)\right].
\end{aligned}$$

We can now show that each of these expectations is finite: letting $C = C_1\|\boldsymbol{V}\|$,

$$\begin{aligned}
\mathbb{E}_{\boldsymbol{z}\sim r_0}\left[\exp\left(C|z_d|\right)\right] &= \int_{-\infty}^{\infty}\exp\left(C|z|\right)\cdot\frac{\beta}{2\Gamma(1/\beta)}\exp\left(-|z|^\beta\right)\,\mathrm{d}z \\
&= 2\frac{\beta}{2\Gamma(1/\beta)}\int_0^\infty \exp\left(Cz - z^\beta\right)\,\mathrm{d}z \\
&= 2\frac{\beta}{2\Gamma(1/\beta)}\left(\int_0^s \exp\left(Cz - z^\beta\right)\,\mathrm{d}z + \int_s^\infty \exp\left(Cz - z^\beta\right)\,\mathrm{d}z\right)
\end{aligned}$$

for any $s \in (0,\infty)$. The first integral is clearly finite. Picking $s = (2|C|)^{\frac{1}{\beta-1}}$, so that $|Cz| < \frac{1}{2}z^\beta$ for $z > s$, gives that

$$\int_s^\infty \exp\left(Cz - z^\beta\right)\,\mathrm{d}z \le \int_s^\infty \exp\left(-\tfrac{1}{2}z^\beta\right)\,\mathrm{d}z < \frac{1}{\beta}2^{\frac{1}{\beta}}\Gamma\left(\frac{1}{\beta}\right) < \infty,$$

so that $\int\exp(f(\boldsymbol{x}))q_0(\boldsymbol{x})\mathrm{d}\boldsymbol{x} < \infty$ as desired. $\square$

The condition on $\boldsymbol{\phi}$ holds for any $\boldsymbol{\phi}$ given by a deep network with Lipschitz activation functions, such as the softplus function we use in this work. The condition on κ also holds for a linear kernel (where $L_\kappa = 1$, $C_\kappa = 0$), any translation-invariant kernel ($L_\kappa = 0$, $C_\kappa = \kappa(0, 0)$), or mixtures thereof. If κ is bounded, the integral is finite for any function $\boldsymbol{\phi}$.

The given proof would not hold for a quadratic κ, which has been used previously in the literature; indeed, it is clearly possible for such an f to be unnormalizable.

B. Finding the optimal $\boldsymbol{\alpha}$

We will show a slightly more general result than we need, also allowing for an $\|f\|^2_{\mathcal{H}}$ penalty. This result is related to Lemma 4 of Sutherland et al. (2018), but is more elementary and specialized to our particular needs while also allowing for more types of regularizers.

Proposition 3. *Consider the loss*

$$\hat{J}(f^k_{\boldsymbol{\alpha},\boldsymbol{z}}, \boldsymbol{\lambda}, \mathcal{D}) = \hat{J}(p^k_{\boldsymbol{\alpha},\boldsymbol{z}}, \mathcal{D}) + \frac{1}{2}\left[\lambda_\alpha \|\boldsymbol{\alpha}\|^2 + \lambda_{\mathcal{H}} \|f^k_{\boldsymbol{\alpha},\boldsymbol{z}}\|^2_{\mathcal{H}} + \lambda_C \frac{1}{N}\sum_{n=1}^N \sum_{d=1}^D \left[\partial_d^2 \log \tilde{p}^k_{\boldsymbol{\alpha},\boldsymbol{z}}(\boldsymbol{x}_n)\right]^2\right]$$

where

$$\hat{J}(p^k_{\boldsymbol{\alpha},\boldsymbol{z}}, \mathcal{D}) = \frac{1}{N}\sum_{n=1}^N \sum_{d=1}^D \left[\partial_d^2 \log \tilde{p}^k_{\boldsymbol{\alpha},\boldsymbol{z}}(\boldsymbol{x}_n) + \frac{1}{2}\left(\partial_d \log \tilde{p}^k_{\boldsymbol{\alpha},\boldsymbol{z}}(\boldsymbol{x}_n)\right)^2\right].$$

For fixed k, $\boldsymbol{z}$, and $\boldsymbol{\lambda}$, as long as $\lambda_{\boldsymbol{\alpha}} > 0$ then the optimal $\boldsymbol{\alpha}$ is

$$\begin{aligned}
\boldsymbol{\alpha}(\boldsymbol{\lambda}, k, \boldsymbol{z}, \mathcal{D}) &= \operatorname*{arg\,min}_{\boldsymbol{\alpha}} \hat{J}(f^k_{\boldsymbol{\alpha},\boldsymbol{z}}, \boldsymbol{\lambda}, \mathcal{D}) = -\left(\boldsymbol{G} + \lambda_\alpha \boldsymbol{I} + \lambda_{\mathcal{H}} \boldsymbol{K} + \lambda_C \boldsymbol{U}\right)^{-1} \boldsymbol{b} \\
G_{m,m'} &= \frac{1}{N}\sum_{n=1}^N \sum_{d=1}^D \partial_d k(\boldsymbol{x}_n, \boldsymbol{z}_m)\, \partial_d k(\boldsymbol{x}_n, \boldsymbol{z}_{m'}) \\
U_{m,m'} &= \frac{1}{N}\sum_{n=1}^N \sum_{d=1}^D \partial_d^2 k(\boldsymbol{x}_n, \boldsymbol{z}_m)\, \partial_d^2 k(\boldsymbol{x}_n, \boldsymbol{z}_{m'}) \\
K_{m,m'} &= k(\boldsymbol{z}_m, \boldsymbol{z}_{m'}) \\
b_m &= \frac{1}{N}\sum_{n=1}^N \sum_{d=1}^D \partial_d^2 k(\boldsymbol{x}_n, \boldsymbol{z}_m) + \partial_d \log q_0(\boldsymbol{x}_n)\, \partial_d k(\boldsymbol{x}_n, \boldsymbol{z}_m) + \lambda_C \partial_d^2 \log q_0(\boldsymbol{x}_n)\, \partial_d^2 k(\boldsymbol{x}_n, \boldsymbol{z}_m).
\end{aligned}$$

Proof. We will show that the loss is quadratic in $\boldsymbol{\alpha}$. Note that

$$\begin{aligned}
\frac{1}{N}\sum_{n=1}^N \sum_{d=1}^D \partial_d^2 \log \tilde{p}^k_{\boldsymbol{\alpha},\boldsymbol{z}}(\boldsymbol{x}_n) &= \frac{1}{N}\sum_{n=1}^N \sum_{d=1}^D \left[\sum_{m=1}^M \alpha_m \partial_d^2 k(\boldsymbol{x}_n, \boldsymbol{z}_m) + \partial_d^2 \log q_0(\boldsymbol{x}_n)\right] \\
&= \boldsymbol{\alpha}^\mathsf{T} \left[\frac{1}{N}\sum_{n=1}^N \sum_{d=1}^D \partial_d^2 k(\boldsymbol{x}_n, \boldsymbol{z}_m)\right]_m + \text{const} \\
\frac{1}{N}\sum_{n=1}^N \sum_{d=1}^D \frac{1}{2}\left(\partial_d \log \tilde{p}^k_{\boldsymbol{\alpha},\boldsymbol{z}}(\boldsymbol{x}_n)\right)^2 &= \frac{1}{N}\sum_{n=1}^N \sum_{d=1}^D \frac{1}{2}\Bigg(\sum_{m,m'=1}^M \alpha_m \alpha_{m'} \partial_d k(\boldsymbol{x}_n, \boldsymbol{z}_m) \partial_d k(\boldsymbol{x}_n, \boldsymbol{z}_{m'}) \\
&\qquad + 2\sum_{m=1}^M \alpha_m \partial_d \log q_0(\boldsymbol{x}_n) \partial_d k(\boldsymbol{x}_n, \boldsymbol{z}_m) + \left(\partial_d \log q_0(\boldsymbol{x}_n)\right)^2\Bigg) \\
&= \frac{1}{2}\boldsymbol{\alpha}^\mathsf{T} \boldsymbol{G} \boldsymbol{\alpha} + \boldsymbol{\alpha}^\mathsf{T}\left[\frac{1}{N}\sum_{n=1}^N \sum_{d=1}^D \partial_d \log q_0(\boldsymbol{x}_n) \partial_d k(\boldsymbol{x}_n, \boldsymbol{z}_m)\right] + \text{const.}
\end{aligned}$$

The λ_C term is of the same form, but with second derivatives:

$$\frac{1}{2N}\sum_{n=1}^{N}\sum_{d=1}^{D}\left(\partial_d^2 \log \tilde{p}^k_{\boldsymbol{\alpha},\boldsymbol{z}}(\boldsymbol{x}_n)\right)^2 = \frac{1}{2}\boldsymbol{\alpha}^\mathsf{T}\boldsymbol{U}\boldsymbol{\alpha} + \boldsymbol{\alpha}^\mathsf{T}\left[\frac{1}{N}\sum_{n=1}^{N}\sum_{d=1}^{D}\partial_d^2 \log q_0(\boldsymbol{x}_n)\partial_d^2 k(\boldsymbol{x}_n, \boldsymbol{z}_m)\right] + \text{const.}$$

We also have as usual

$$\frac{1}{2}\|f^k_{\boldsymbol{\alpha},\boldsymbol{z}}\|^2_{\mathcal{H}} = \frac{1}{2}\sum_{m=1}^{M}\sum_{m'=1}^{M}\alpha_m \langle k(\boldsymbol{z}_m,\cdot), k(\boldsymbol{z}_{m'},\cdot)\rangle_{\mathcal{H}}\, \alpha_{m'} = \frac{1}{2}\boldsymbol{\alpha}^\mathsf{T}\boldsymbol{K}\boldsymbol{\alpha}.$$

Thus the overall optimization problem is

$$\begin{aligned}\boldsymbol{\alpha}(\boldsymbol{\lambda}, k, \boldsymbol{z}, \mathcal{D}) &= \operatorname*{arg\,min}_{\boldsymbol{\alpha}} \hat{J}(f^k_{\boldsymbol{\alpha},\boldsymbol{z}}, \boldsymbol{\lambda}, \mathcal{D}) \\ &= \operatorname*{arg\,min}_{\boldsymbol{\alpha}} \frac{1}{2}\boldsymbol{\alpha}^\mathsf{T}\left(\boldsymbol{G} + \lambda_{\boldsymbol{\alpha}}\boldsymbol{I} + \lambda_{\mathcal{H}}\boldsymbol{K} + \lambda_C \boldsymbol{U}\right)\boldsymbol{\alpha} + \boldsymbol{\alpha}^\mathsf{T}\boldsymbol{b}.\end{aligned}$$

Because $\lambda_\alpha > 0$ and $\boldsymbol{G}$, $\boldsymbol{K}$, $\boldsymbol{U}$ are all positive semidefinite, the matrix in parentheses is strictly positive definite, and the claimed result follows directly from standard vector calculus. □

C. Behavior on mixtures

Proposition 4. *Let $\mathcal{D} = \bigcup_{i=1}^{I}\mathcal{D}_i$, where $\mathcal{D}_i \subset \mathcal{X}_i$, $|\mathcal{D}_i| = \pi_i N$, $\sum_{i=1}^{I}\pi_i = 1$. Also suppose that the inducing points are partitioned as $\boldsymbol{Z} = [\boldsymbol{Z}_1; \ldots; \boldsymbol{Z}_I]$, with $\boldsymbol{Z}_i \subset \mathcal{X}_i$. Further let the kernel k be such that $k(\boldsymbol{x}_1, \boldsymbol{x}_2) = 0$ when $\boldsymbol{x}_1 \in \mathcal{X}_i$, $\boldsymbol{x}_2 \in \mathcal{X}_j$ for $i \neq j$, with its first and second derivatives also zero. Then the kernel exponential family solution of Proposition 3 is*

$$\boldsymbol{\alpha}(\boldsymbol{\lambda}, k, \boldsymbol{z}, \mathcal{D}) = \begin{bmatrix}\boldsymbol{\alpha}\left(\left(\frac{\lambda_\alpha}{\pi_1}, \frac{\lambda_{\mathcal{H}}}{\pi_1}, \lambda_C\right), k, \boldsymbol{Z}_1, \mathcal{D}_1\right) \\ \vdots \\ \boldsymbol{\alpha}\left(\left(\frac{\lambda_\alpha}{\pi_I}, \frac{\lambda_{\mathcal{H}}}{\pi_I}, \lambda_C\right), k, \boldsymbol{Z}_I, \mathcal{D}_I\right)\end{bmatrix}.$$

Proof. Let $\boldsymbol{G}_i$, $\boldsymbol{b}_i$ be the $\boldsymbol{G}$, $\boldsymbol{b}$ of Proposition 3 when using only $\boldsymbol{Z}_i$ and $\mathcal{D}_i$. Then, because the kernel values and derivatives are zero across components, if m and m' are from separate components then

$$G_{m,m'} = \frac{1}{N}\sum_{n=1}^{N}\sum_{d=1}^{D}\partial_d k(\boldsymbol{x}_n, \boldsymbol{z}_m)\partial_d k(\boldsymbol{x}_n, \boldsymbol{z}_{m'}) = 0,$$

as at least one of the kernel derivatives will be zero for each term of the sum. When m and m' are from the same component, the total will be the same except that N is bigger, giving

$$\boldsymbol{G} = \begin{bmatrix}\pi_1\boldsymbol{G}_1 & \boldsymbol{0} & \cdots & \boldsymbol{0} \\ \boldsymbol{0} & \pi_2\boldsymbol{G}_2 & \cdots & \boldsymbol{0} \\ \vdots & \vdots & \ddots & \vdots \\ \boldsymbol{0} & \boldsymbol{0} & \cdots & \pi_I\boldsymbol{G}_I\end{bmatrix}.$$

$\boldsymbol{U}$ is of the same form and factorizes in the same way. $\boldsymbol{K}$ does not scale:

$$\boldsymbol{K} = \begin{bmatrix}\boldsymbol{K}_1 & \cdots & \boldsymbol{0} \\ \vdots & \ddots & \vdots \\ \boldsymbol{0} & \cdots & \boldsymbol{K}_I\end{bmatrix}.$$

Recall that $\boldsymbol{b}$ is given as

$$b_m = \frac{1}{N}\sum_{n=1}^{N}\sum_{d=1}^{D}\partial_d^2 k(\boldsymbol{x}_n, \boldsymbol{z}_m) + \partial_d \log q_0(\boldsymbol{x}_n)\, \partial_d k(\boldsymbol{x}_n, \boldsymbol{z}_m) + \lambda_C \partial_d^2 \log q_0(\boldsymbol{x}_n)\, \partial_d^2 k(\boldsymbol{x}_n, \boldsymbol{z}_m).$$

Each term in the sum for which $\boldsymbol{x}_n$ is in a different component than $\boldsymbol{z}_m$ will be zero, giving $\boldsymbol{b} = (\pi_1 \boldsymbol{b}_1, \cdots, \pi_I \boldsymbol{b}_I)$. Thus $\boldsymbol{\alpha}(\boldsymbol{\lambda}, k, \boldsymbol{z}, \mathcal{D})$ becomes

$$\begin{aligned}
\boldsymbol{\alpha} &= -\left(\boldsymbol{G} + \lambda_\alpha \boldsymbol{I} + \lambda_{\mathcal{H}} \boldsymbol{K} + \lambda_C \boldsymbol{U}\right)^{-1} \boldsymbol{b} \\
&= -\begin{bmatrix} \pi_1 \boldsymbol{G}_1 + \lambda_\alpha \boldsymbol{I} + \lambda_{\mathcal{H}} \boldsymbol{K}_1 + \lambda_C \pi_1 \boldsymbol{U}_1 & \cdots & \mathbf{0} \\ \vdots & \ddots & \vdots \\ \mathbf{0} & \cdots & \pi_I \boldsymbol{G}_I + \lambda_\alpha \boldsymbol{I} + \lambda_{\mathcal{H}} \boldsymbol{K}_I + \lambda_C \pi_I \boldsymbol{U}_I \end{bmatrix}^{-1} \begin{bmatrix} \pi_1 \boldsymbol{b}_1 \\ \vdots \\ \pi_I \boldsymbol{b}_I \end{bmatrix} \\
&= \begin{bmatrix} -(\boldsymbol{G}_1 + \frac{\lambda_\alpha}{\pi_1} \boldsymbol{I} + \frac{\lambda_{\mathcal{H}}}{\pi_1} \boldsymbol{K}_1 + \lambda_C \boldsymbol{U}_1)^{-1} \boldsymbol{b}_1 \\ \vdots \\ -(\boldsymbol{G}_I + \frac{\lambda_\alpha}{\pi_I} \boldsymbol{I} + \frac{\lambda_{\mathcal{H}}}{\pi_I} \boldsymbol{K}_I + \lambda_C \boldsymbol{U}_I)^{-1} \boldsymbol{b}_2 \end{bmatrix}. \qquad \square
\end{aligned}$$

Thus the fits for the components are essentially added together, except that each component uses a different λ_α and $\lambda_{\mathcal{H}}$; smaller components are regularized more. λ_C, interestingly, is unscaled.

It is difficult in general to tell how two components will be weighted relative to one another; the problem is essentially equivalent to computing the overall normalizing constant of a fit. However, we can gain some insight by analyzing a greatly simplified case, in Appendix C.1.

C.1. Small Gaussian components with a large Gaussian kernel

Consider, for the sake of our study of mixture fits, one of the simplest possible situations for a kernel exponential family: $p_0 = \mathcal{N}(\mathbf{0}, \boldsymbol{I})$, with a kernel $k(\boldsymbol{x}, \boldsymbol{y}) = \exp\left(-\frac{1}{\sigma^2}\|\boldsymbol{x} - \boldsymbol{y}\|^2\right)$ for $\sigma \gg \sqrt{D}$, so that $k(\boldsymbol{x}, \boldsymbol{y}) \approx 1$ for all $\boldsymbol{x}, \boldsymbol{y}$ sampled from p_0. Let q_0 be approximately uniform, $q_0 = \mathcal{N}(\mathbf{0}, q\boldsymbol{I})$ for $q \gg \sigma^2$, so that $\nabla \log q_0(\boldsymbol{x}) = \frac{-1}{q^2}\boldsymbol{x} \approx 0$. Also assume that $N \to \infty$, but M is fixed. Assume that $\lambda_{\mathcal{H}} = \lambda_C = 0$, and refer to λ_α as simply λ. Then we have

$$\begin{aligned}
G_{m,m'} &= \frac{1}{N} \sum_{n=1}^{N} \sum_{d=1}^{D} \partial_d k(\boldsymbol{x}_n, \boldsymbol{z}_m) \partial_d k(\boldsymbol{x}_n, \boldsymbol{z}_{m'}) \\
&= \frac{1}{N} \sum_{n=1}^{N} \sum_{d=1}^{D} \left(\frac{z_{m,d} - \mathrm{x}_d}{\sigma^2} k(\boldsymbol{x}_n, \boldsymbol{z}_m) \right) \left(\frac{z_{m',d} - \mathrm{x}_d}{\sigma^2} k(\boldsymbol{x}_n, \boldsymbol{z}_{m'}) \right) \\
&\approx \sigma^{-4} \, \mathbb{E}_{\mathbf{x} \sim p_0} \left[(\boldsymbol{z}_m - \mathbf{x})^\mathsf{T} (\boldsymbol{z}_{m'} - \mathbf{x}) \right] \\
&= \sigma^{-4} \left(\boldsymbol{z}_m^\mathsf{T} \boldsymbol{z}_{m'} + D \right) \\
\boldsymbol{G} &\approx \frac{1}{\sigma^4} \boldsymbol{Z}\boldsymbol{Z}^\mathsf{T} + \frac{D}{\sigma^4} \boldsymbol{I} \\
b_m &= \frac{1}{N} \sum_{n=1}^{N} \sum_{d=1}^{D} \partial_d^2 k(\boldsymbol{x}_n, \boldsymbol{z}_m) + \partial_d \log q_0(\boldsymbol{x}_n) \, \partial_d k(\boldsymbol{x}_n, \boldsymbol{z}_m) \\
&\approx \mathbb{E}_{\mathbf{x} \sim p_0} \left[\sum_{d=1}^{D} \partial_d^2 k(\boldsymbol{x}_n, \boldsymbol{z}_m) \right] \\
&= \mathbb{E}_{\mathbf{x} \sim p_0} \left[\sum_{d=1}^{D} \left(\frac{(\mathrm{x}_d - z_{m,d})^2}{\sigma^4} - \frac{1}{\sigma^2} \right) k(\mathbf{x}, \boldsymbol{z}_m) \right] \\
&\approx \mathbb{E}_{\mathbf{x} \sim p_0} \left[\frac{\|\mathrm{x}_d - \boldsymbol{z}_m\|^2}{\sigma^4} - \frac{D}{\sigma^2} \right] \\
&= \frac{\|\boldsymbol{z}_m\|^2 + D}{\sigma^4} - \frac{D}{\sigma^2} \\
\boldsymbol{b} &\approx \frac{1}{\sigma^4} \operatorname{diag}(\boldsymbol{Z}\boldsymbol{Z}^\mathsf{T}) - \frac{D(\sigma^2 - 1)}{\sigma^4} \mathbf{1}.
\end{aligned}$$

Because $k(\boldsymbol{z}, \boldsymbol{x}) \approx k(\boldsymbol{z}', \boldsymbol{x})$ for any $\boldsymbol{z}, \boldsymbol{z}'$ near the data in this setup, it's sufficient to just consider a single $\boldsymbol{z} = \mathbf{0}$. In that case,

$$\begin{aligned}\boldsymbol{\alpha} &= -(\boldsymbol{G}+\lambda\boldsymbol{I})^{-1}\boldsymbol{b} \\ &\approx -\left(\frac{1}{\sigma^4}\boldsymbol{Z}\boldsymbol{Z}^\mathsf{T} + \left(\frac{D}{\sigma^4}+\lambda\right)\boldsymbol{I}\right)^{-1}\left(\frac{1}{\sigma^4}\operatorname{diag}(\boldsymbol{Z}\boldsymbol{Z}^\mathsf{T}) - \frac{D(\sigma^2-1)}{\sigma^4}\mathbf{1}\right) \\ &= -\left(\left(\frac{D}{\sigma^4}+\lambda\right)\boldsymbol{I}\right)^{-1}\left(-\frac{D(\sigma^2-1)}{\sigma^4}\mathbf{1}\right) \\ &= \frac{1}{D\sigma^{-4}+\lambda}\frac{D(\sigma^2-1)}{\sigma^4}\mathbf{1} \\ &= \frac{\sigma^2-1}{1+\lambda\sigma^4/D}\mathbf{1}\end{aligned}$$

and so

$$f_{\boldsymbol{\alpha}}(\mathbf{0}) \approx \frac{\sigma^2-1}{1+\lambda\sigma^4/D}.$$

Thus, if we attempt to fit the mixture $\pi\,\mathcal{N}(\mathbf{0},\boldsymbol{I}) + (1-\pi)\,\mathcal{N}(\boldsymbol{r},\boldsymbol{I})$ with $q^2 \gg \|\boldsymbol{r}\|^2 \gg \sigma^2 \gg D$, we are approximately in the regime of Proposition 4 and so the ratio between the two components in the fit is

$$\begin{aligned}\exp\left(f(\mathbf{0}) - f(\boldsymbol{r})\right) &\approx \exp\left(\frac{\sigma^2-1}{1+\frac{\lambda\sigma^4}{\pi D}} - \frac{\sigma^2-1}{1+\frac{\lambda\sigma^4}{(1-\pi)D}}\right) \\ &= \exp\left(\lambda\sigma^4(\sigma^2-1)\left(\frac{\frac{1}{(1-\pi)D} - \frac{1}{\pi D}}{1+\frac{\lambda\sigma^4}{\pi D}+\frac{\lambda\sigma^4}{(1-\pi)D}+\frac{\lambda^2\sigma^8}{\pi(1-\pi)D^2}}\right)\right) \\ &= \exp\left(\frac{1}{2}\lambda\sigma^4(\sigma^2-1)\left(\frac{\pi-\frac{1}{2}}{D\pi(1-\pi)+\lambda\sigma^4+\frac{\lambda^2\sigma^8}{D}}\right)\right).\end{aligned}$$

If $\pi = \frac{1}{2}$, the density ratio is correctly 1. If we further assume that $\lambda \gg D/\sigma^4$, so that the denominator is dominated by the last term, then the ratio becomes approximately

$$\exp(f(\mathbf{0}) - f(\boldsymbol{r})) \approx \exp\left(\frac{D}{2\sigma^2\lambda}\left(\pi - \frac{1}{2}\right)\right).$$

Thus, depending on the size of $D/(2\sigma^2\lambda) \ll \sigma^2/2$, the ratio will usually either remain too close to $\frac{1}{2}$ or become too extreme as π changes; only in a very narrow parameter range is it approximately correct.

D. Upper bound on normalizer bias

Recall the importance sampling setup of Section 3.2:

$$\hat{Z}_{\boldsymbol{\theta}} = \frac{1}{U}\sum_{u=1}^{U}\mathrm{r}_u \quad \text{where } \boldsymbol{y}_u \sim q_0, \mathrm{r}_u := \frac{\tilde{p}_{\boldsymbol{\theta}}(\boldsymbol{y}_u)}{q_0(\boldsymbol{y}_u)} \quad \text{so} \quad \mathbb{E}\,\hat{Z}_{\boldsymbol{\theta}} = \int \frac{\tilde{p}_{\boldsymbol{\theta}}(\boldsymbol{y}_u)}{q_0(\boldsymbol{y}_u)} q_0(\boldsymbol{y}_u) = Z_{\boldsymbol{\theta}}.$$

Proposition 1. *Suppose that* $a, s \in \mathbb{R}$ *are such that* $\Pr(\mathrm{r}_u \geq a) = 1$ *and* $\Pr(\mathrm{r}_u \leq s) \leq \rho < \frac{1}{2}$. *Define* $t := (s+a)/2$, $\psi(q, Z_{\boldsymbol{\theta}}) := \log\frac{Z}{q} + \frac{q}{Z} - 1$, *and let* $P := \max\left(\psi(a, Z_{\boldsymbol{\theta}}), \psi(t, Z_{\boldsymbol{\theta}})\right)$. *Then*

$$\log Z_{\boldsymbol{\theta}} - \mathbb{E}\log\hat{Z}_{\boldsymbol{\theta}} \leq \frac{\psi\left(t, Z_{\boldsymbol{\theta}}\right)}{\left(Z_{\boldsymbol{\theta}} - t\right)^2}\frac{\operatorname{Var}[\mathrm{r}_u]}{U} + P\left(4\rho(1-\rho)\right)^{\frac{U}{2}}.$$

Proof. Inspired by the technique of Liao & Berg (2018), we will decompose the bias as follows. (We will suppress the subscript $\boldsymbol{\theta}$ for brevity.)

First note that the following form of a Taylor expansion holds identically:

$$\begin{aligned}\varphi(x) &= \varphi(Z) + \varphi'(Z)(x - Z) + h(x, Z)(x - Z)^2 \\ h(x, Z) &:= \frac{\varphi(x) - \varphi(Z) - \varphi'(Z)(x - Z)}{(x - Z)^2}.\end{aligned}$$

We can thus write the bias as the following, where $\varphi(x) = -\log(x)$, P is the distribution of $\hat{Z}$, and $t \geq a$:

$$\begin{aligned}\mathbb{E}[\varphi(\hat{Z})] - \varphi(\mathbb{E}\,\hat{Z}) &= \int_a^\infty (\varphi(x) - \varphi(Z))\,\mathrm{d}P(x) \\ &= \int_a^\infty \left(\varphi'(Z)(x - Z) + h(x, Z)(x - Z)^2\right)\mathrm{d}P(x) \\ &= \varphi'(Z)\left(\int_a^\infty x\,\mathrm{d}P(x) - Z\right) + \int_a^\infty h(x, Z)(x - Z)^2 \mathrm{d}P(x) \\ &= \int_a^t h(x, Z)(x - Z)^2 \mathrm{d}P(x) + \int_t^\infty h(x, Z)(x - Z)^2 \mathrm{d}P(x) \\ &\leq \left[\sup_{a \leq x \leq t} h(x, Z)(x - Z)^2\right] \int_a^t \mathrm{d}P(x) + \left[\sup_{x \geq t} h(x, Z)\right] \int_t^\infty (x - Z)^2 \mathrm{d}P(x) \\ &\leq \left[\sup_{a \leq x \leq t} h(x, Z)(x - Z)^2\right] \Pr(\hat{Z} \leq t) + \left[\sup_{x \geq t} h(x, Z)\right] \mathrm{Var}[\hat{Z}].\end{aligned}$$

Now,

$$h(x, Z)(x - Z)^2 = \log\frac{Z}{x} + \frac{x}{Z} - 1$$

is convex in x and thus its supremum is $\max\left(\log\frac{Z}{a} + \frac{a}{Z} - 1, \log\frac{Z}{t} + \frac{t}{Z} - 1\right)$, with the term at a being necessarily larger as long as $t < Z$.

Picking $t = (s + a)/2$ gives the desired bound on $\Pr(\hat{Z} \leq t)$ via Lemma 5.

Lemma 1 of Liao & Berg (2018) shows that since $\varphi'(x) = -1/x$ is concave, $h(x, Z)$ is decreasing in x. Thus $\sup_{x \geq t} h(x, Z) = h(t, Z)$, giving the claim. □

Lemma 5. *Let a and s be such that $\Pr(\mathrm{r}_u \geq a) = 1$ and $\Pr(\mathrm{r}_u \leq s) \leq \rho < \frac{1}{2}$, with $a < s$. Then*

$$\Pr\left(\frac{1}{U}\sum_{i=1}^U \mathrm{r}_u \leq \frac{s + a}{2}\right) \leq (4\rho(1 - \rho))^{\frac{U}{2}}.$$

Proof. Let K denote the number of samples of r_u which are smaller than s, so that $U - K$ samples are at least s. Then we have

$$\begin{aligned}\Pr\left(\frac{1}{U}\sum_{u=1}^U \mathrm{r}_u \leq \frac{s + a}{2}\right) &\leq \Pr\left(\frac{K}{U}a + \frac{U - K}{U}s \leq \frac{s + a}{2}\right) \\ &= \Pr\left(K(a - s) \leq U\frac{a - s}{2}\right) \\ &= \Pr\left(K \geq \frac{U}{2}\right).\end{aligned}$$

K is distributed binomially with probability of success at most $\rho < \frac{1}{2}$, so applying Theorem 1 of Arratia & Gordon (1989) yields

$$\Pr\left(\frac{1}{U}\sum_{i=1}^{U} \mathrm{r}_u \le \frac{s+a}{2}\right) \le \exp\left(-U\left[\frac{1}{2}\log\frac{1}{2\rho} + \frac{1}{2}\log\frac{1}{2(1-\rho)}\right]\right)$$
$$= \exp\left(\frac{U}{2}\log\left(4\rho(1-\rho)\right)\right)$$
$$= \left(4\rho(1-\rho)\right)^{\frac{U}{2}} . \qquad \square$$

Proposition 6. *The function* $\chi_t(x) := \left(\log\frac{x}{t} + \frac{t}{x} - 1\right)/(x-t)^2$ *is strictly convex for all* $x > 0$. *Thus we have that* $\mathbb{E}\,\chi_t(\hat{Z}_{\boldsymbol\theta}) \ge \chi_t(\mathbb{E}\,\hat{Z}_{\boldsymbol\theta}) = \chi_t(Z_{\boldsymbol\theta})$, *with equality only if* $\Pr(\hat{Z}_{\boldsymbol\theta} = Z_{\boldsymbol\theta}) = 1$.

Proof. We can compute that

$$\chi_t''(x) = \frac{2\frac{t^3}{x^3} - 9\frac{t^2}{x^2} + 18\frac{t}{x} - 11 - 6\log\frac{t}{x}}{(x-t)^4}.$$

Let $r := t/x$, so $x \in [t, \infty)$ corresponds to $r \in (0, 1]$, and $x \in (0, t]$ corresponds to $r \in [1, \infty)$. Then

$$\chi_t''\left(\frac{t}{r}\right) = \frac{2r^3 - 9r^2 + 18r - 11 - 6\log r}{t^4\left(\frac{1}{r} - 1\right)^4}.$$

We can evaluate $\lim_{r\to 1}\chi''(t/r) = \frac{3}{2t^4} > 0$. For $r \ne 1$, $\chi_t'' > 0$ if and only if $f(r) > 0$, where

$$f(r) := 2r^3 - 9r^2 + 18r - 11 - 6\log r.$$

Clearly $\lim_{r\to 0} f(r) = \infty$ and $f(1) = 0$. But notice that

$$f'(r) = 6r^2 - 18r + 18 - \frac{6}{r} = \frac{6(r-1)^3}{r},$$

so that $f(r)$ is strictly decreasing on $(0, 1)$, and strictly increasing on $(1, \infty)$. Thus $f(r) > 0$ for all $r \in (0, 1) \cup (1, \infty)$, and $\chi_t''(x) > 0$ for all $x > 0$. The claim follows by Jensen's inequality. $\square$

D.1. Estimator of bias bound

For a kernel such as (7) bounded in $[0, 1]$, $a := \exp\left(\sum_{m=1}^{M}\min(\alpha_m, 0)\right)$ is a uniform lower bound on r_u.

For large U, essentially any $\rho < \frac{1}{2}$ will make the second term practically zero, so we select s as slightly less than the 40th percentile of an initial sample of r_u, and confirm a high-probability (0.999) Hoeffding upper bound ρ on $\Pr(\mathrm{r}_u \le s)$ with another sample. (We use s as $\exp(-0.001) \approx 0.999$ times the estimate of the 40th percentile, to avoid ties.) We use 10^7 samples for each of these.

We estimate $\mathrm{Var}[\mathrm{r}_u]$ on a separate sample with the usual unbiased estimator, using 10^9 samples for most cases but 10^{10} for MiniBoone.

To finally estimate the bound, we estimate $Z_{\boldsymbol\theta}$ on yet another independent sample, again usually of size 10^9 but 10^{10} for MiniBoone.

Crucially, the function $\psi(t, x)/(x-t)^2$ is convex (Proposition 6); because the variance is unbiased, our estimate of the bias bound is itself biased upwards. As Proposition 1's bound is also not tight, our estimate thus likely overstates the actual amount of bias.

E. Additional experimental details

E.1. Synthetic datasets

For each synthetic distribution, we sample 10 000 random points from the distribution, 1 000 of which are used for testing; of the rest, 90% (8 100) are used for training, and 10% (900) are used for validation. Training was early stopped when

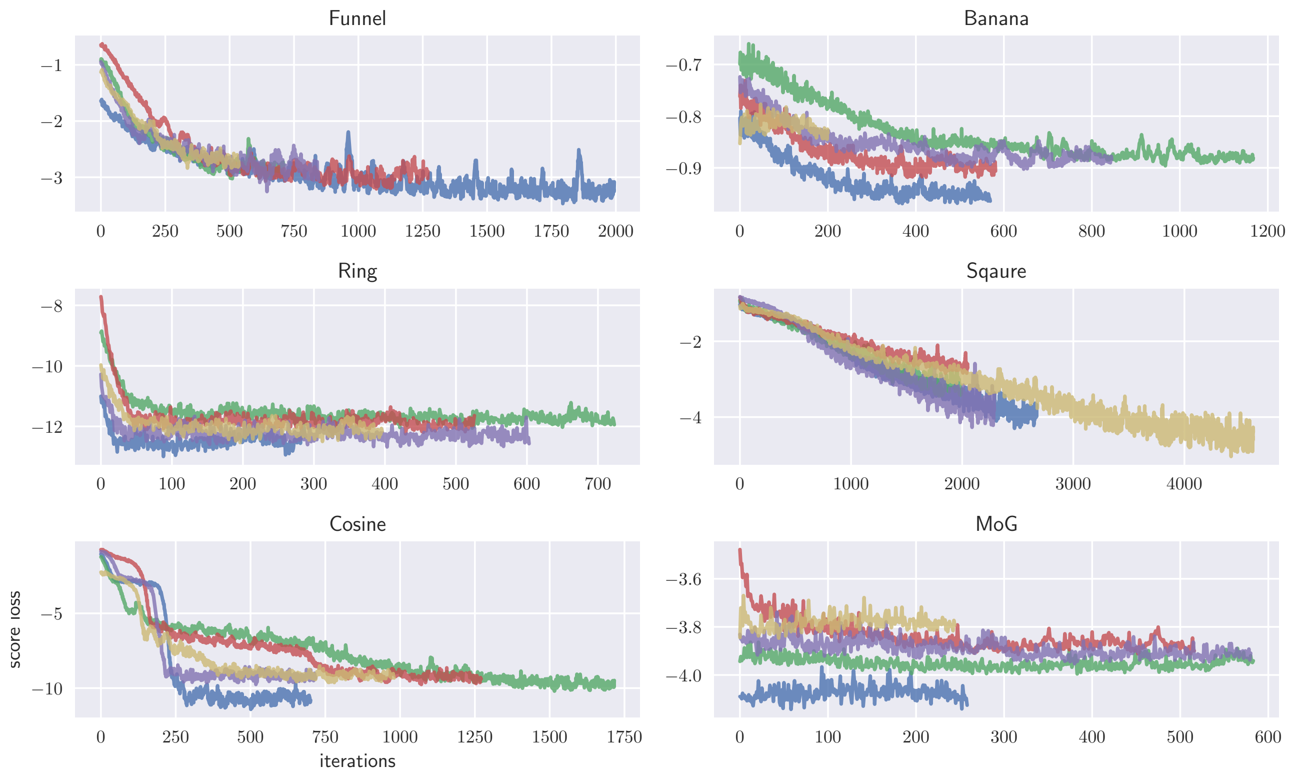

Figure 4: Validation score loss on 6 synthetic datasets for 5 runs.

validation cost does not improve for 200 minibatches. The current implementation of KCEF does not include a Nyström approximation, and trains via full-batch L-BFGS-B, so we down-sampled the training data to 1000 points. We used the Adam optimizer (Kingma & Ba, 2015) for all other models. For MADE, RealNVP, and MAF, we used minibatches of size 200 and the learning rate was 10^{-3} For KEF-G and DKEF, we used 200 inducing points, used $|\mathcal{D}_t| = |\mathcal{D}_v| = 100$, and learning rate 10^{-3}. The same parameters are used for each component for mixture models trained on MoR.

To show that learning is stable, we ran the experiments on 5 random draws of training, validation and test sets from the synthetic distributions, trained KDEF initialized using 5 random seeds and calculated validation score at each iteration until convergence in the first phase of training (before optimizing for λ's). The traces are shown in Figure 4.

The same data for benchmark datasets are shown in Figure 5. There is no overfitting except for the small Redwine dataset. Runs on Hepmass and Miniboone do not seem to fully converge, despite having met the early stopping criterion.

E.2. Benchmark datasets

Pre-processing RedWine and WhiteWine are quantized, and thus problematic for modeling with continuous densities; we added to each dimension uniform noise with support equal to the median distances between two adjacent values. For HepMass and MiniBoone, we removed ill-conditioned dimensions as did Papamakarios et al. (2017). For all datasets except HepMass, 10% of the entire data was used as testing, and 10% of the remaining was used for validation with an upper limit of $1\,000$ due to time cost of validation at each iteration. For HepMass, we used the same splitting as done in Papamakarios et al. (2017) and with the same upper limit on validation set. The data is then whitened before fitting and the whitening matrix was computed on at most $10\,000$ data points.

Likelihood-based models We set MADE, MADE-MOG, each autoregressive layer of MAF and each scaling and shifting layers of real NVP to have two hidden layers of 100 neurons. For real NVP, MAF and MAF-MOG, five autoregressive layers were used; MAF-MOG and MADE-MOG has a mixture of 10 Gaussians for each conditional distribution. Learning rate was 10^{-3} The size of a minibatch is 200.

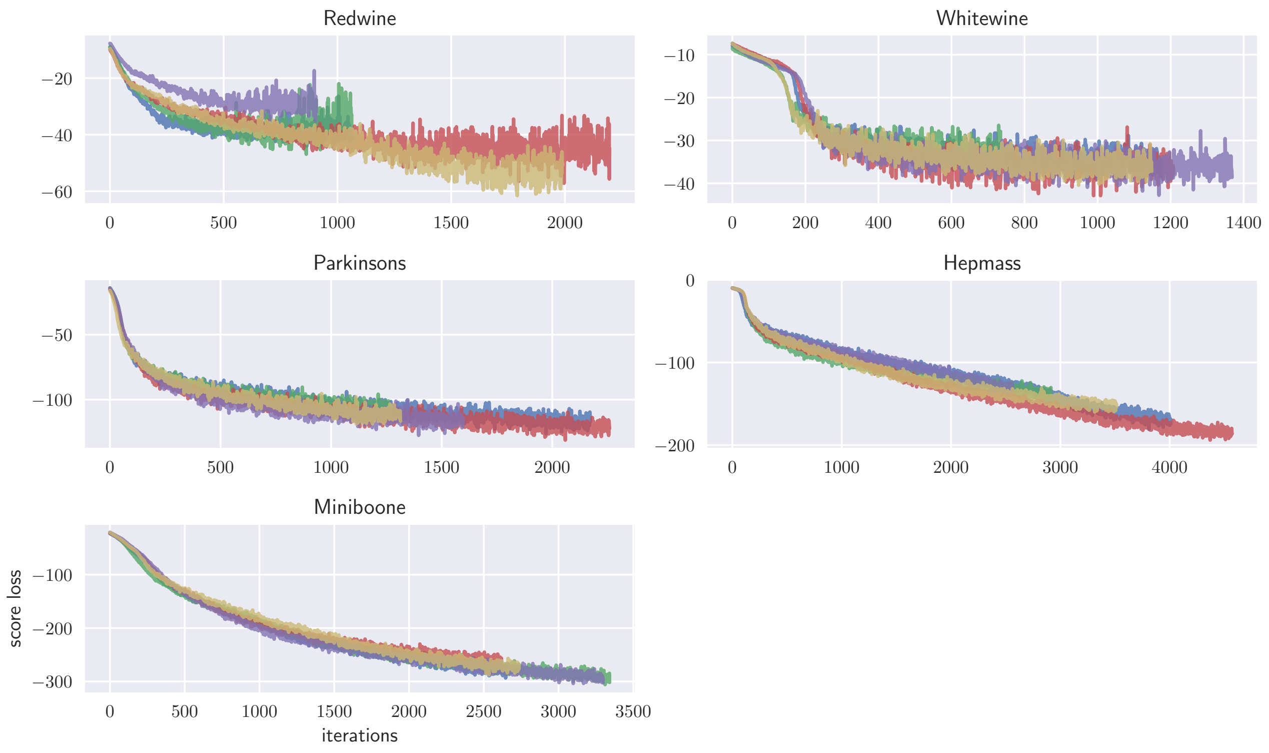

Figure 5: Validation score loss on 5 benchmark datasets for 5 runs.

Deep kernel exponential family We set the DKEF model to have three kernels ($R = 3$), each a Gaussian on features of a 3-layer network with 30 neurons in each layer. There was also a skip-layer connection from data directly to the last layer which accelerated learning. Length scales σ_r were initialized to 1.0, 3.3 and 10.0. Each λ was initialized to 0.001. The weights of the network were initialized from a Gaussian distribution with standard deviation equal to $1/\sqrt{30}$. We also optimized the inducing points $\boldsymbol{z}_m$ which were initialized with random draws from training data. The number of inducing points $M = 300$, and $|\mathcal{D}_t| = |\mathcal{D}_v| = 100$. The learning rate was 10^{-2}. We found that our initialization on the weight std and σ_r's are importance for fast and stable learning; other parameters did not significantly change the results under similar computational budget (time and memory).

FSSD tests were conducted using 100 points $\boldsymbol{v}_b$ selected at random from the test set, with added normal noise of standard deviation 0.2, using code provided by the authors.

We estimated $\log Z_{\boldsymbol{\theta}}$ with 10^{10} samples proposed from q_0, as in Section 3.2, and estimated the bias as in Appendix D.1.

We added independent $\mathcal{N}(0, 0.05^2)$ noise to the data in training. This is similar to the regularization applied by (Kingma & LeCun, 2010; Saremi et al., 2018), except that the noise is added directly to the data instead of the model.

For all models, we stopped training when the objective ((4) or log likelihood) did not improve for 200 minibatches. We also set a time budget of 3 hours on each model; this was fully spent by MAF, MOG-MAF and Real NVP on HepMass. We found that MOG-MADE had unstable runs on some datasets; out of 15 runs on each dataset, 7 on WhiteWine, 4 on Parkinsons and 9 on MiniBoone produced invalid log likelihoods. These results were discarded in Figure 3 log likelihood panels.

The DKEF in our main results (Figure 3) has an adaptive q_0 which is a generalized normal distribution. We also trained DKEF with q_0 being an isotropic multivariate normal of standard deviation 2.0. These results Figure 6 are similar to Figure 3 but exhibit much smaller bias estimates in the log normalizer for RedWine and Parkinsons.

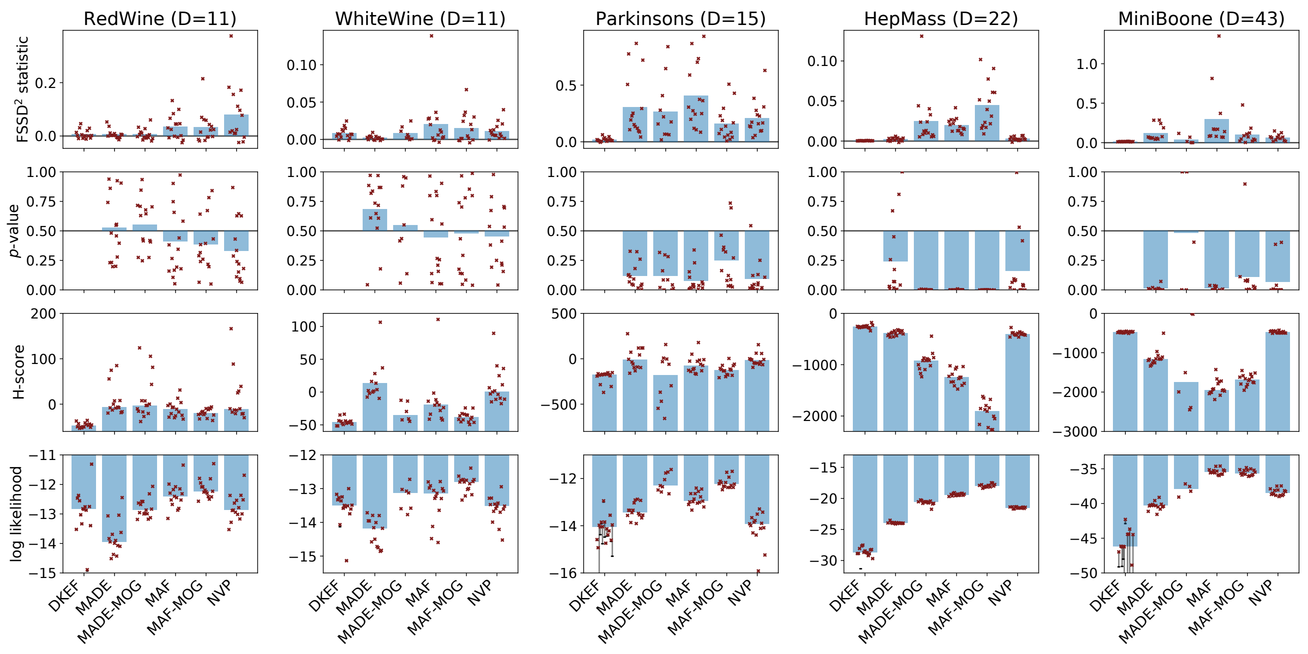

Figure 6: Results on benchmark datasets as in Figure 3 with the q_0 in DKEF being isotropic multivariate normal of std 2.0.