# Supplementary Material : Sum-of-Squares Polynomial Flow

## A. Simulated Experiments

Here, we explore the effect of relative ordering for the conditioner network for SOS flows as well as mixture of Gaussians. We again generated two sets of 2D densities given by $p(x_1, x_2) = \mathcal{N}(x_2\ ; 0, 4)\mathcal{N}(x_1\ ; 0.25x_2^2, 1)$ and $p(x_1, x_2) = \mathcal{N}(x_2\ ; 2, 2)\mathcal{N}(x_1\ ; 1/3x_2^3, 1.5)$. However, we trained both SOS flows and GMMs with the reverse order i.e. $x_1, x_2$. For SOS flows we again tested using both deep and wide flows whereas for MoGs we tested with varying number of components for each conditional. We present the plots in Figure 4. The best performance here is by a deep SOS flow. Furthermore, while a flat SOS flow is able to achieve almost the same geometrical shape as the target density, its learned density still differs from the true density. For mixture of Gaussians, a large number of components for each conditional improved the performance of the resulting model.

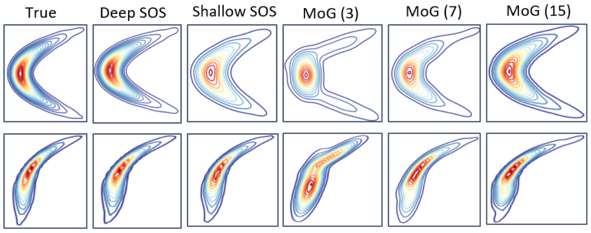

*Figure 4.* **Top:** Left plot shows the target density given by $p(x_1, x_2) = \mathcal{N}(x_2\ ; 0, 4)\mathcal{N}(x_1\ ; 0.25x_2^2, 1)$. The second plot shows the density learnt by SOS flows with 3 blocks and a sum of 2 polynomials with degree 3 with ordering $(x_1, x_2)$. Third plot shows the density learnt by SOS flows with 1 block and a sum of 2 polynomials with degree 4 and ordering $(x_1, x_2)$. The last three plots estimate this density using a Mixture of Gaussian conditionals with varying components given in parenthesis and ordering $(x_1, x_2)$. **Bottom:** Same as Top but with target density given by $p(x_1, x_2) = \mathcal{N}(x_2\ ; 2, 2)\mathcal{N}(x_1\ ; 0.33x_1^3, 1.5)$.

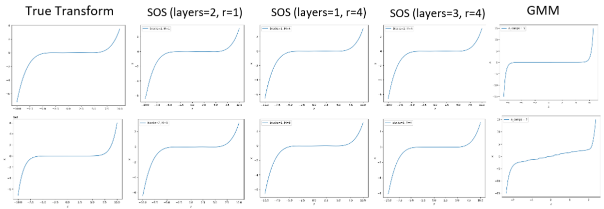

*Figure 5.* **Top Row:** Transformation defined by a deep SOS flow with $M = r = 1$ and blocks =4. The next three plots show SOS flows learning this transformation with different configurations (deep, wide and, wide-deep). The last plot shows the transformation learned when a Gaussian mixture model learns the density (or transformation). **Bottom Row:** Same as Top Row but the true transformation was derived by a wide and shallow SOS flow with $M = r = 4$ and blocks=1.

We also test the representational power of deep and wide SOS flows and the results are given in Figure 5. In the first row, the true transformation was simulated by stacking multiple blocks of SOS transformation. Subsequently, we generated the target density using this transformation and estimated it using a deep flow, wide flow, wide-deep flow and mixture of Gaussians. In the second row, we simulated the true transformation using a single block SOS transformation and performed the same experiment as before. In both simulations, we tried to break our model by adding random noise to the coefficients of simulated transformation. As the figure shows, however, both deep and wide variants performed equally well in terms of representation. As expected however, the training time for wider flows was significantly longer than that for deeper flows.

Finally, we tested SOS flows on a suite of 2D simulated datasets – Funnel, Banana, Square, Mixture of Gaussians and Mixture of Rings. These datasets cover a broad range of geometries and have been considered before by Wenliang et al. (2018). For these experiments, we constructed our model by stacking 3 blocks with each block being a sum of *two* polynomials each of degree *four*. We plot the log density function learned by SOS flow and the true model in Figure 6. The model is able to capture the true log density of datasets like Funnel and Banana. The true densities of Funnel and Banana are a simple linear transformation of Gaussians. Hence, flow based models that learn a continuous and smooth transformation are expected to perform well on these datasets. However, SOS demonstrates certain artifacts at the sharp corners of the Square although it is able to capture the overall density nicely. These three datasets – Funnel, Banana, and Square – were part of the unimodal simulated datasets.

The multimodal datasets included Mixture of Gaussians (MoG) and Mixture of Rings (MoR). As discussed earlier in Remark 1, when the target distribution has regions of near zero mass, the learned transformation admits sharp jumps to capture such regions. Flow based models by virtue of being invertible and smooth are often unable to learn such sharp jumps. SOS flows performs reasonably well for mixture of Gaussians although there are certain artifacts in the model

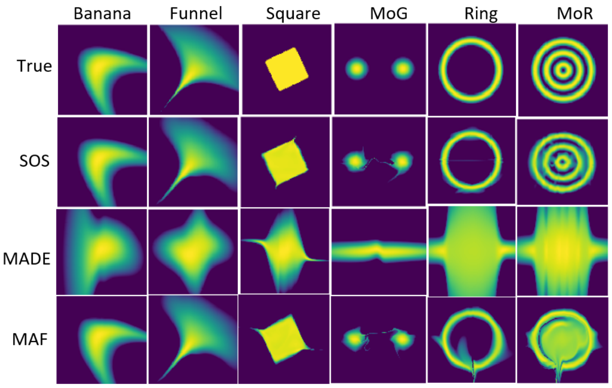

*Figure 6.* Log-densities for various toy-datasets. The top row shows the true log-densities. The next three rows are the log-densities for SOS flows, MADE and, MAF respectively.

that try to connect the two components. Similarly, there are some artifacts connecting the rings for the Mixture of Rings datasets. However, this issue of separated components can be dealt with relative ease in practice using clustering.

## B. Transformation for Mixture of Gaussians

The slope $T'(z)$ of $T$ at any point $z$ is given by

$$\begin{aligned} T'(z) &= \frac{p(z)}{q\Big(T(z)\Big)}, \quad \text{where} \quad x = T(z) \\ &= \frac{p\Big(F^{-1}(t)\Big)}{q\Big(G^{-1}(t)\Big)}, \quad \text{where} \quad t = F(z) \end{aligned}$$

i.e. the slope $T'(z)$ is the ratio of probability density quantiles (pdQs) of the source random variable and the target random variable.

We now analyze the transformation required to transform a standard normal distribution to mixture of normal distributions. Figure 7 shows three columns of plots: In the leftmost column, the top plot is the source distribution (Z) which is standard normal. The bottom plot is the target distribution for the random variable X which is a Gaussian mixture model with two components. The means are $-10$ and 10, the variance is 1 and weights are 0.5 for each component respectively. The middle plot shows the transformation $T$ required to push forward a standard normal distribution to the target. In the second column of plots, we now transform a standard normal distribution to a mixture distribution but with means as -20 and 20, i.e. the components are more separated. Finally, in the plots given in the rightmost column, we transform a standard normal distribution to a mixture of three Gaussians with means -20, -5, and 15. The variances are 1 and weights are $\frac{1}{3}$ respectively.

We make the following observations here: In all three plots for the transformation, we notice that the transformation admits jumps (close to being vertical) i.e. the slope at these points is large and close to infinity. This is expected since the regions where the target has almost zero mass but the source has finite mass would lead to a slope with such behavior. In the plots, this is the region in between the components where the mass of the target density approaches zero. Furthermore, the larger this area, the longer is the height of this jump (see plots on column one and column two). With densities that have two such areas, the transformation as expected has two jumps (plots on column three). The slope of $T$ on the extremes is a constant and is equal to the standard deviation of the component on that extreme. This is because:

$$\lim_{z\to\pm\infty} T'(z) = \lim_{z\to\pm\infty} \frac{p(z)}{q\Big(T(z)\Big)}$$

As $z \to \infty$, $q$ is approximately equal to the component on the positive extreme of the x-axis. This easily gives that $\lim_{z\to\infty} T'(z) = \sigma_+$ where $\sigma_+$ is the standard deviation of the component on the positive extreme of the x-axis; similarly, we get $\lim_{z\to-\infty} T'(z) = \sigma_-$ i.e. $T'(z)$ is a constant in almost all the region of zero mass on the left of the component on the negative extreme and on the right of the positive extreme (verified in Figure 7). Finally, the only regions where $T'(z)$ is finite is whenever $q(x) > q(\tilde{x})$ where $\tilde{x} \leq \mu_i \pm 2\sigma_i$ where the index $i$ stands for the $i^{th}$ component. Therefore, any $T$ that transforms a standard normal distribution to a mixture of Gaussians will be approximately piece-wise linear with jumps. The number of linear pieces in this transformation will be equal to the number of components in the mixture. The slopes of these linear pieces will be a function of the standard deviations of the mixture components. Additionally, the height of the jump will be a function of the mixing weights and standard deviation of the mixture components.

## C. Proofs

**Lemma 1** (Mulansky & Neamtu 1998)**.** *Let $S$ be a dense subspace of $X$ and let $C \subseteq X$ be a convex set such that* $\mathrm{int}(C) \neq \emptyset$. *Then $C \cap S$ is dense in $C$.*

*Proof.* Since the interior $\mathrm{int}(C)$ is open and nonempty, and $S$ is dense, we know $\mathrm{int}(C) \cap S$ is dense in $\mathrm{int}(C)$. (Every open set of $\mathrm{int}(C)$ is also an open set of $X$, hence intersects the dense set $S$.) Moreover, since $C$ is convex and $\mathrm{int}(C) \neq \emptyset$, we know $\mathrm{cl}(\mathrm{int}(C)) = \mathrm{cl}(C)$, hence $\mathrm{cl}(\mathrm{int}(C) \cap S) = \mathrm{cl}(C)$, i.e., $\mathrm{int}(C) \cap S$, whence also the "larger" set $C \cap S$, is dense in $C$. □

**Theorem 3.** *Let* C *be the space of real univariate continuous functions, equipped with the topology of compact convergence. Then, the set of increasing polynomials is dense*

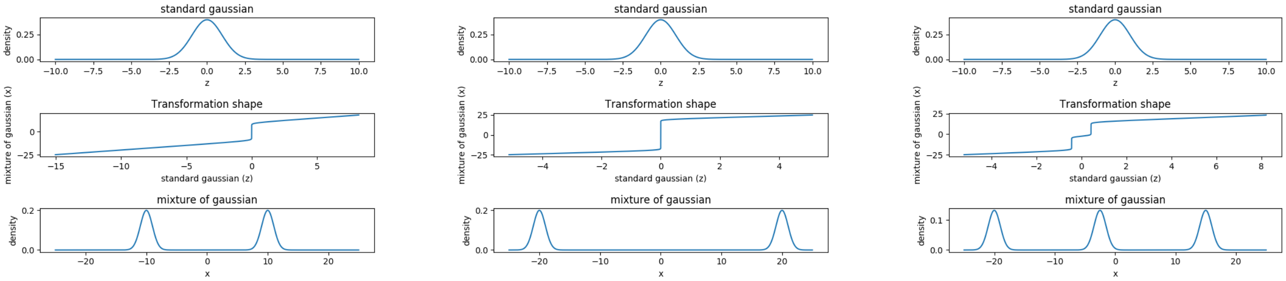

*Figure 7.* Transformation curves from standard Gaussian to mixture of Gaussians.

*in the cone of increasing continuous functions.*[9]

*Proof.* Let us define $\mathsf{P}$ to be the space of polynomials, and $\mathsf{I}$ the space of increasing functions. We need only prove on any compact set $K$, the set of polynomials of the form (8), i.e. $\mathsf{I} \cap \mathsf{P}$ thanks to Theorem 2, is dense in $\mathsf{C}(K) \cap \mathsf{I}$. By Weierstrass' theorem we know $\mathsf{P}$ is dense in $\mathsf{C}(K)$. Moreover, the convex subset $\mathsf{I} \cap \mathsf{C}(K)$ has nonempty interior (take say a linear function with positive slope). Applying Lemma 4 of (Mulansky & Neamtu, 1998) completes the proof[10]. □

[9] This result does not seem to be surprising, given that it is well-known polynomials are dense in $\mathsf{C}$. Note however that it does *not* follow immediately. For instance, it took significant effort for Huang et al. (2018) to prove a similar result, which is arguably one of their main contributions.

[10] We thank Csaba Szepesvári for bringing (Mulansky & Neamtu, 1998) to our attention, which allowed us to reduce a lengthy proof to the current slim one.