

## A Appendix

We provide in this appendix updated results for all experimental settings.

**We would like to draw the fact that most of our runs for image generation finished after the submission time for the main text.**

As a result, the results provided here for image generation paint a **stronger** picture than those shown in main text, with **noticeable improvements** in FID and other metrics when scaling up  $n$  across our 3 image generation tasks. We will update the main text in time with these results.

Given the very large scale of our study, we still have runs that have not completed (notably for the largest  $n$  and smallest  $\varepsilon$  choices) for Imagenet 32 and Imagenet 64, these points will be added to our curves at a later date.

- Gaussian, piecewise affine ground truth OT map: **Figures 10, 11**
- [Korotin et al.](#) benchmark: **Figures 12, 13**
- CIFAR-10: **Figures 14, 15**
- Imagenet-32: **Figures 16, 17**
- Imagenet-64: **Figures 18, 19**

Solver (NFE) $\rightarrow$ Algorithm $\downarrow$	Euler (4)	Euler (8)	Euler (16)	Dopri5 (Adaptive)
IFM	66.4	24.3	12.1	5.55
$n = 2048$	38.2	16.8	10.0	5.89
$n = 65536$	33.1	15.1	9.28	4.88
$n = 262144$	32.0	14.9	<b>9.15</b>	4.92
$n = 524288$	<b>31.5</b>	<b>14.8</b>	9.19	<b>4.85</b>

Table 1: Validation FID on images generated by models trained on **ImageNet-32**.  $n$  denotes the total OT batch size. We use the best checkpoint (w.r.t FID at Dopri5) for each model, restricting results to the setting where the relative epsilon value  $\varepsilon = 0.1$  for ease of presentation (more detailed results can be seen in the plots of Figure 16). The average number of steps for the Dopri5 solver is between 19.0 and 19.5 for all rows above, with each step roughly requiring 6 function evaluations. Compared to IFM, We observe that models trained with larger batch-size attain a better FID faster, that slightly increases after 250k steps. This agrees with our conclusion, in which we state that other choices such as learning-rate and learning-rate scheduler should be reconsidered compared to the config choices used for IFM.

Solver (NFE) $\rightarrow$ Algorithm $\downarrow$	Euler (4)	Euler (8)	Euler (16)	Dopri5 (Adaptive)
IFM	81.4	37.9	19.8	9.39
$n = 4096$	51.0	25.4	16.2	9.38
$n = 32768$	49.7	25.0	15.9	9.11
$n = 131072$	<b>47.7</b>	<b>24.3</b>	<b>15.6</b>	<b>8.99</b>

Table 2: Validation FID on images generated by models trained on **ImageNet-64**. As above, we use the best checkpoint (w.r.t FID at Dopri5) for each model, and fix  $\varepsilon = 0.1$ . The average number of steps for the Dopri5 solver is between 44.3 and 44.5 for all rows above, with each step roughly requiring 6 function evaluations.

### A.1 Using the negative dot-product cost rather than squared-Euclidean in Sinkhorn

As we mention in the main text, entropically regularized optimal transport plan for the squared Euclidean cost can be equivalently recast using exclusively the negative scalar product  $(\mathbf{x}, \mathbf{y}) \mapsto$

539  $-\langle \mathbf{x}, \mathbf{y} \rangle$  between source and target, and not on any absolute measure of scale. To see this, consider  
 540 an affine map  $\bar{\mathbf{x}} = \alpha \mathbf{x} + \beta$  with  $\alpha > 0$ . Then:

$$\langle \bar{\mathbf{x}}, \mathbf{y} \rangle = \alpha \langle \mathbf{x}, \mathbf{y} \rangle + \langle \beta, \mathbf{y} \rangle.$$

541 The second term is a rank-1 term that will be absorbed by the optimal dual potentials (see (3)) and  
 542 the factor  $\alpha$  amounts to a rescaling of the entropic regularization level  $\varepsilon$ . In particular, when there is  
 543 only translation, then  $\alpha = 1$  and the transport plans are identical for the same  $\varepsilon$ .

544 Therefore for input data  $\mathbf{X} \in \mathbb{R}^{n \times d}$ ,  $\mathbf{Y} \in \mathbb{R}^{m \times d}$  the Sinkhorn transport plan depends only on the  
 545 dot-product cost  $-\mathbf{XY}^T$ . We argue that it is always more natural to use the dot-product cost than the  
 546 full squared-Euclidean cost, and we find that in practice using directly the dot-product can improve  
 547 the numerical conditioning of the Sinkhorn algorithm. This is because we drop terms arising from  
 548 the squared norm, which can be very large. This becomes especially important for single-precision  
 549 floating point computations, as is the case for the large scale GPU applications we consider.

550 We illustrate this in Figure 5: we sample  $N = 8192$  points  $\{\mathbf{x}_i\}_{i=1}^N$  in dimension  $d = 128$  from  
 551 the Gaussian example described in Section 4.2 and map them through the piecewise affine Brenier  
 552 map, i.e.  $\mathbf{y}_i = T(\mathbf{x}_i)$ . We then introduce a translation,  $\bar{\mathbf{y}}_i = \mathbf{y}_i + 5$ . We use the Sinkhorn algorithm  
 553 (Algorithm 1) with either the dot-product or squared Euclidean cost to compute the transport plan  
 554 and record the number of iterations taken, and our computations are carried out on GPU with  
 555 single-precision arithmetic. Even though we already use log-domain computation tricks to prevent  
 556 under/overflow, we find that for small  $\varepsilon$ , Sinkhorn with squared Euclidean cost begins to suffer from  
 557 numerical issues and fails to converge within the iteration limit of 50,000.

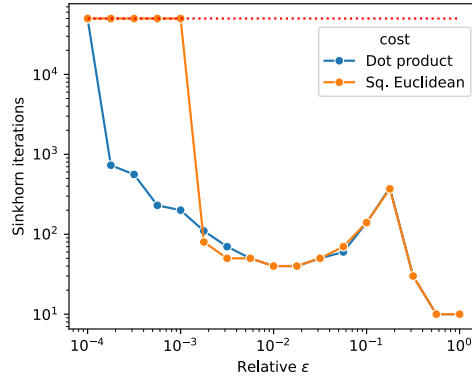


Figure 5: Number of Sinkhorn iterations against relative  $\varepsilon$  for Gaussian piecewise affine OT example.

## 558 A.2 Sinkhorn Convergence

559 We provide uncured figures (with a very large numbers of runs spanning various OT batch-sizes  
 560 and  $\varepsilon$  choices) that track the convergence of the Sinkhorn algorithm on the datasets studied in our  
 561 work in: the piecewise affine synthetic dataset in Figure 6; the Korotin et al. benchmark in Figure 7  
 562 and the CIFAR-10 generation in Figure 8.

563 **Note that we do not plan to publish these figures ultimately.** They are only provided to let  
 564 reviewers appreciate that all our Sinkhorn runs converge efficiently and predictably, even for very  
 565 large batch sizes of millions of points. Note also that all batch sizes reported in the legends of these  
 566 plots correspond to the *per GPU batch size*. Hence the actual OT batch size  $n$  is 8 times those  
 567 reported in these raw WANDB plots.

## 568 A.3 Gaussian Transported with a Piecewise Affine Ground-Truth OT Map

569 We propose in Figure 9 examples of our piecewise affine OT map generation, corresponding to results  
 570 presented more widely in Figures 6, 10, 11.

## 571 A.4 Korotin et al. Benchmark Examples

572 The reader may find examples of the Korotin et al. benchmark in their paper, App. A.1, Figure 6.

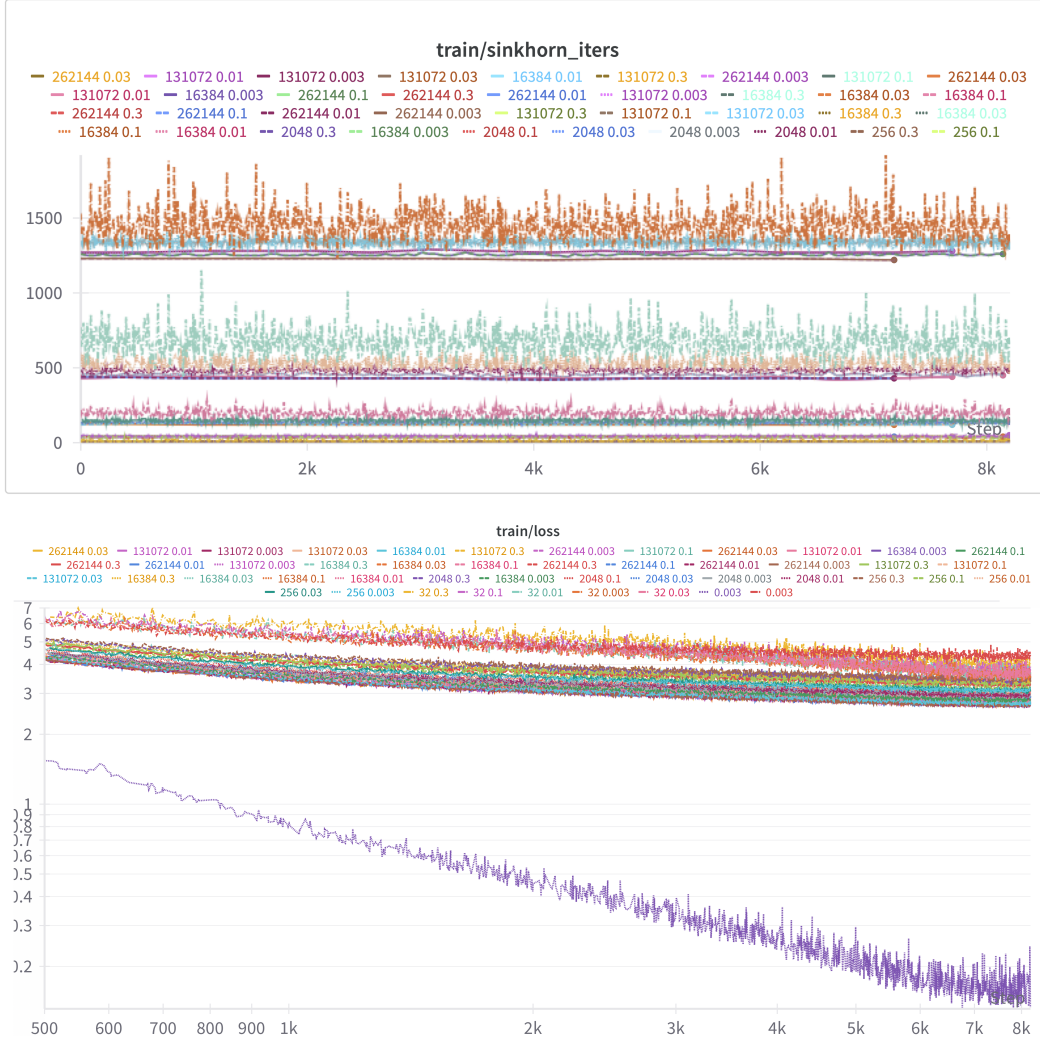


Figure 6: Number of Sinkhorn iterations across training runs, and loss across optimization steps, for the piecewise affine benchmark (to be put in perspective with Figure 1). Legends display the batch size *per device* used to compute optimal transport, followed by the relative  $\varepsilon$  choice. Since these runs are carried out on a single node of 8 GPUs, the corresponding *total* OT batch sizes (what we call  $n$  in our Algorithm 2) should be multiplied by 8 (e.g. 256 in the legend correspond to  $n = 2048$  per our notations). Each step corresponds to a minibatch size of 256 per GPU, a total effective batch size of 2048. Sinkhorn is therefore run every  $n/2048$  steps. The maximal number of iterations is capped at 50k by default, using a tolerance  $\tau = 0.001$  (the default choice in OTT-JAX). As can be seen, almost all runs, even those using low entropy, exit their calls to [Sinkhorn \[1964\]](#) after having converged. The loss curve displayed at the bottom corresponds to that of the network shown ground-truth paired data, denoted as  $\blacktriangle$  in our figures. The IFM curve  $\blacktriangledown$  would be that appearing at the top. Both IFM and ground-truth corresponds to the only labels (displayed last) shown without a batch size first, and labelled with an arbitrary epsilon value of 0.003. Note that these curves are not directly comparable, in the sense that they do not track the same loss function (the loss function is influenced by the coupling or in the extreme case by the fact that ground-truth matched data is provided to  $\blacktriangle$ ).

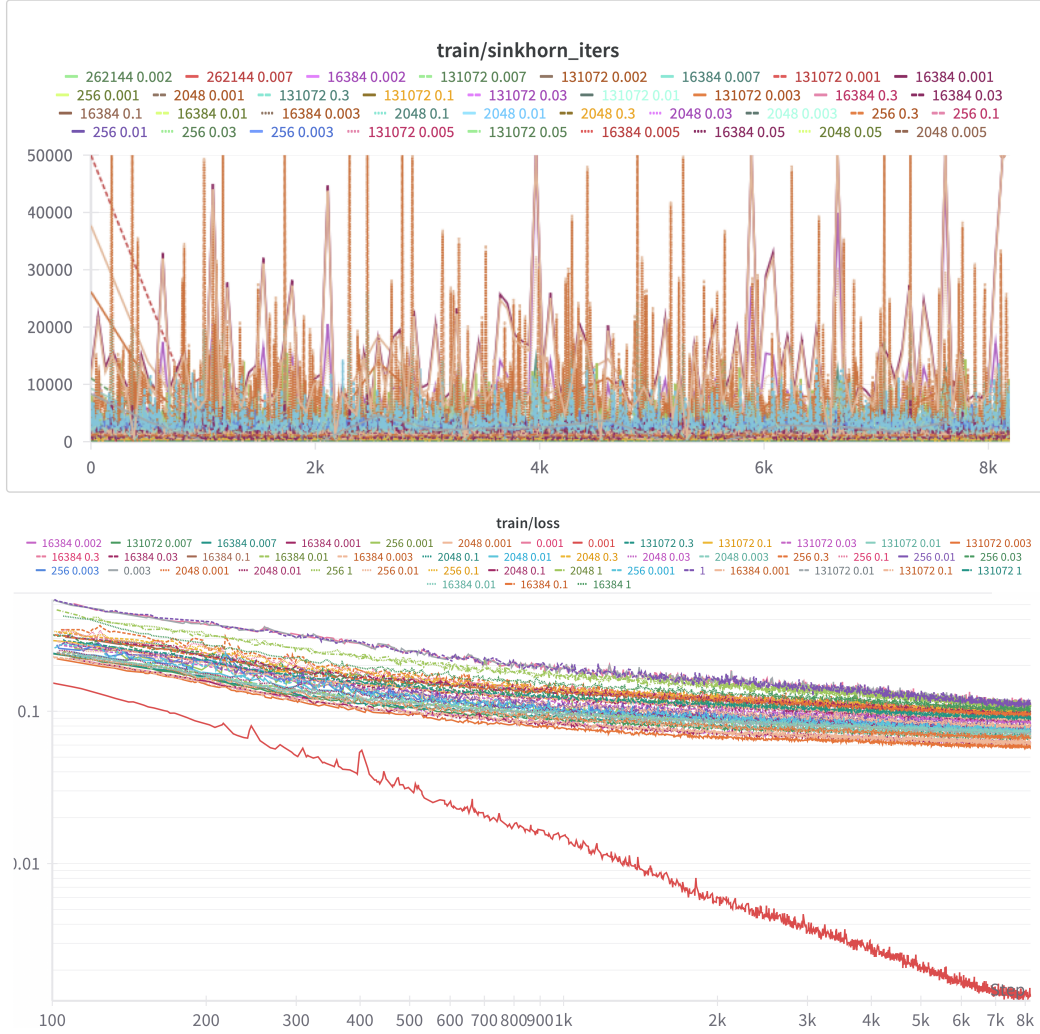


Figure 7: Number of Sinkhorn iterations across training runs, and loss across optimization steps, for the piecewise affine benchmark (to be put in perspective with Figure 2). See text in caption of Fig. 8 for more details on how to interpret these figures. For the train loss plots at the bottom, the curve achieving the lowest loss is that trained with ground-truth pairing / perfect supervision, denoted as  $\blacktriangle$  in our plots. Both IFM  $\blacktriangledown$  and ground-truth  $\blacktriangle$  correspond to the only labels (displayed in the 7th/8th positions in the legend) shown without a batch size first, and labeled with an arbitrary epsilon value of 0.001.





Figure 8: From top to bottom: # of Sinkhorn iterations and times (seconds, log-scale) when recomputing couplings; smoothed loss curves for CIFAR generation (see also Figure 14) for various OT per-device batch size (correspond to  $n/8$  in Algorithm 2) and relative epsilon values, **as shown in the legend in that order**. Compared to Figures 6 and 7 there is no ground-truth pairing known for this task. Runs for I-FM  $\blacktriangledown$  cluster at the top in terms of losses (they correspond to legend entries that do not mention of a GPU-batch size, the epsilon values in these plots simply depicting different initializations).

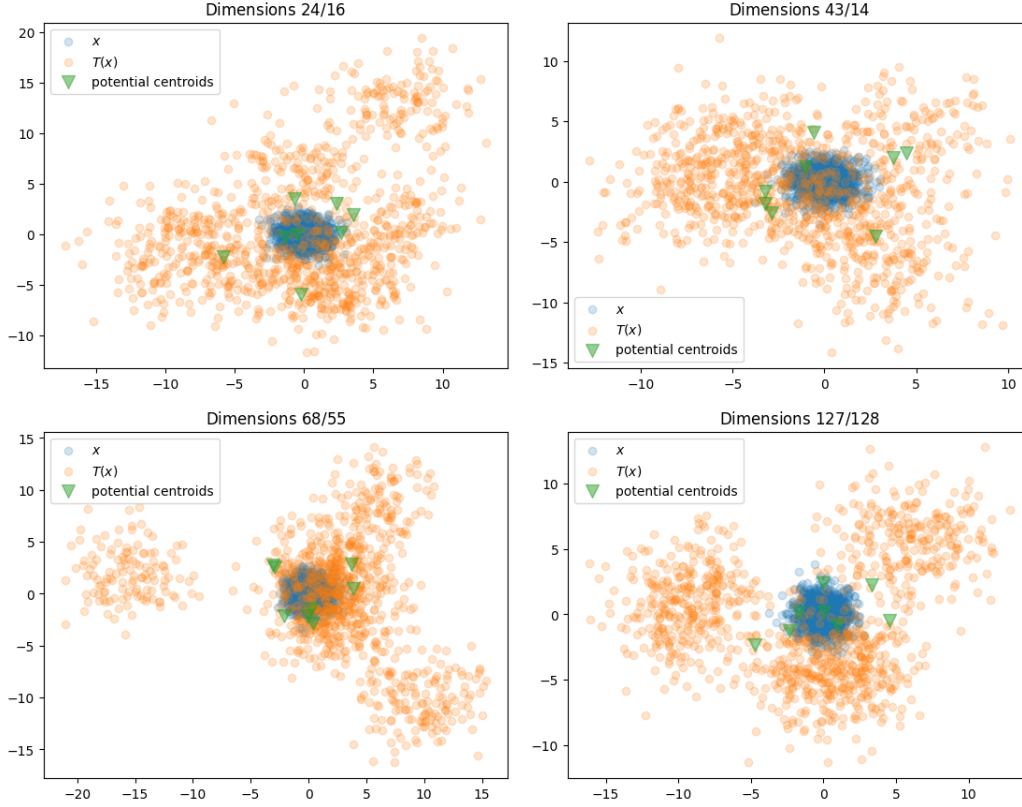


Figure 9: Example of the maps generated in our piecewise affine benchmark task. In these plots  $d = 128$  and there are therefore  $128/16 = 8$  quadratic potentials sampled around 0. These 2D plots illustrate the action of the same 128 dimensional map, pictured using 2D projections over pairs chosen in  $[1, \dots, 128]$ .

### A.5 CIFAR-10 Detailed Results

We present the complete metrics for our CIFAR-10 experiments in Figure 14 and sample generated images in Figure 15. We see general quantitative and qualitative improvements for larger OT batch size and smaller renormalized entropy. However, these improvements are not as significant as our observation for the more complex downsampled ImageNet datasets in Appendices A.6 and A.7.

### A.6 ImageNet-32 Detailed Results

We use the  $32 \times 32$  downsampled variants [Chrabaszcz et al., 2017] of the ImageNet dataset [Deng et al., 2009]. In Figure 16, we plot FID and curvature for different OT batch size and renormalized entropy, and observe significant improvements for larger batch size and smaller entropy. Figure 17 depicts the generated images using IFM and OT-FM with different batch sizes and different ODE solvers. As expected, the greatest improvements in the quality of images occur with smaller number of integration steps, which demonstrates the benefit of OT-FM for reducing inference cost.

### A.7 ImageNet-64 Detailed Results

We also perform experiments on the  $64 \times 64$  downsampled ImageNet dataset, where we observe an even bigger gap between IFM and OT-FM with large batch size both in terms of metrics (Figure 18) and in terms of qualitative results (Figure 19). This observation implies that with a proper choice of entropy and batch size, OT-FM is a promising approach to reduce inference cost and generate higher quality high-resolution images.

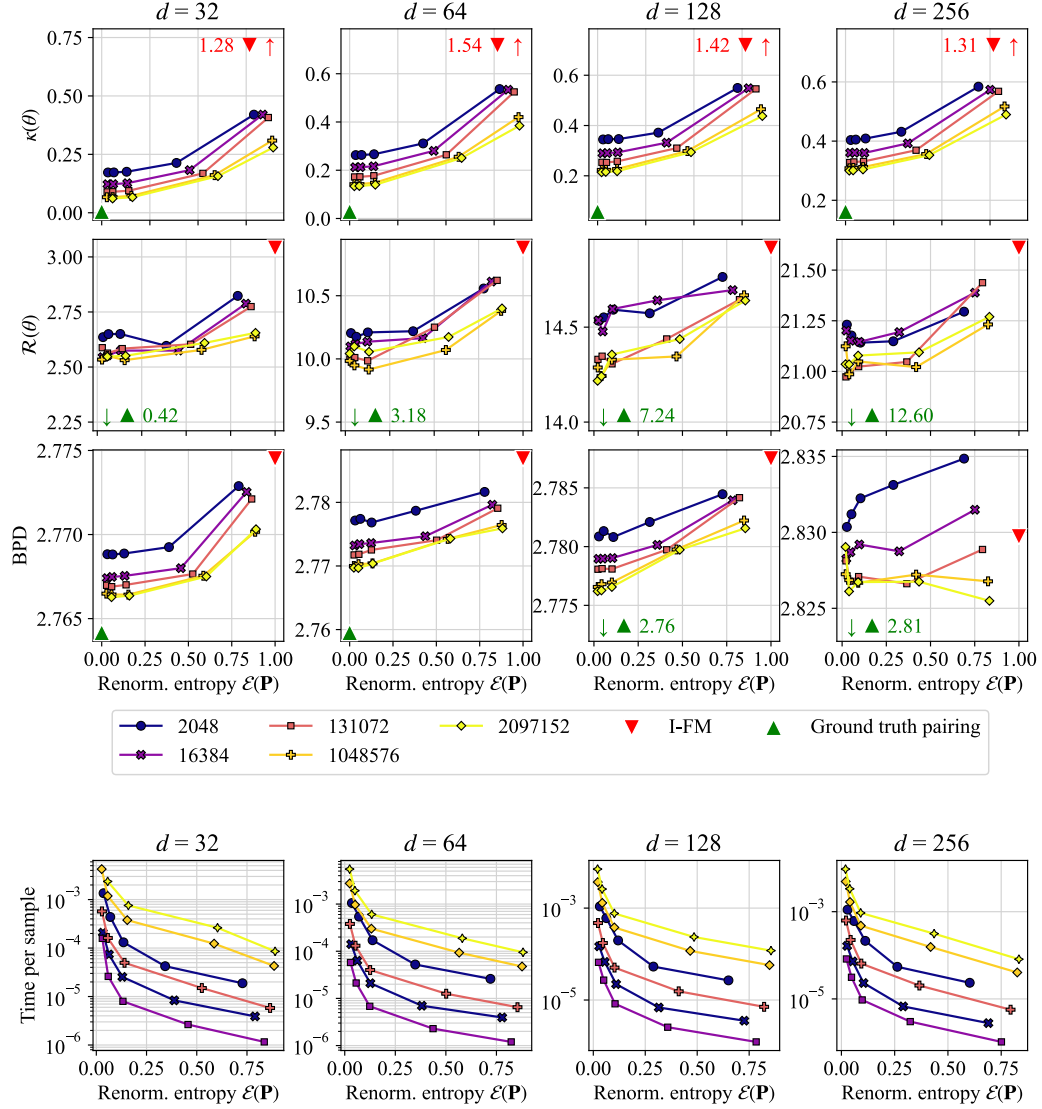


Figure 10: Final plots for the piecewise affine benchmark task, corresponding to those presented in Figure 1 of the main body of the paper, using renormalized entropies

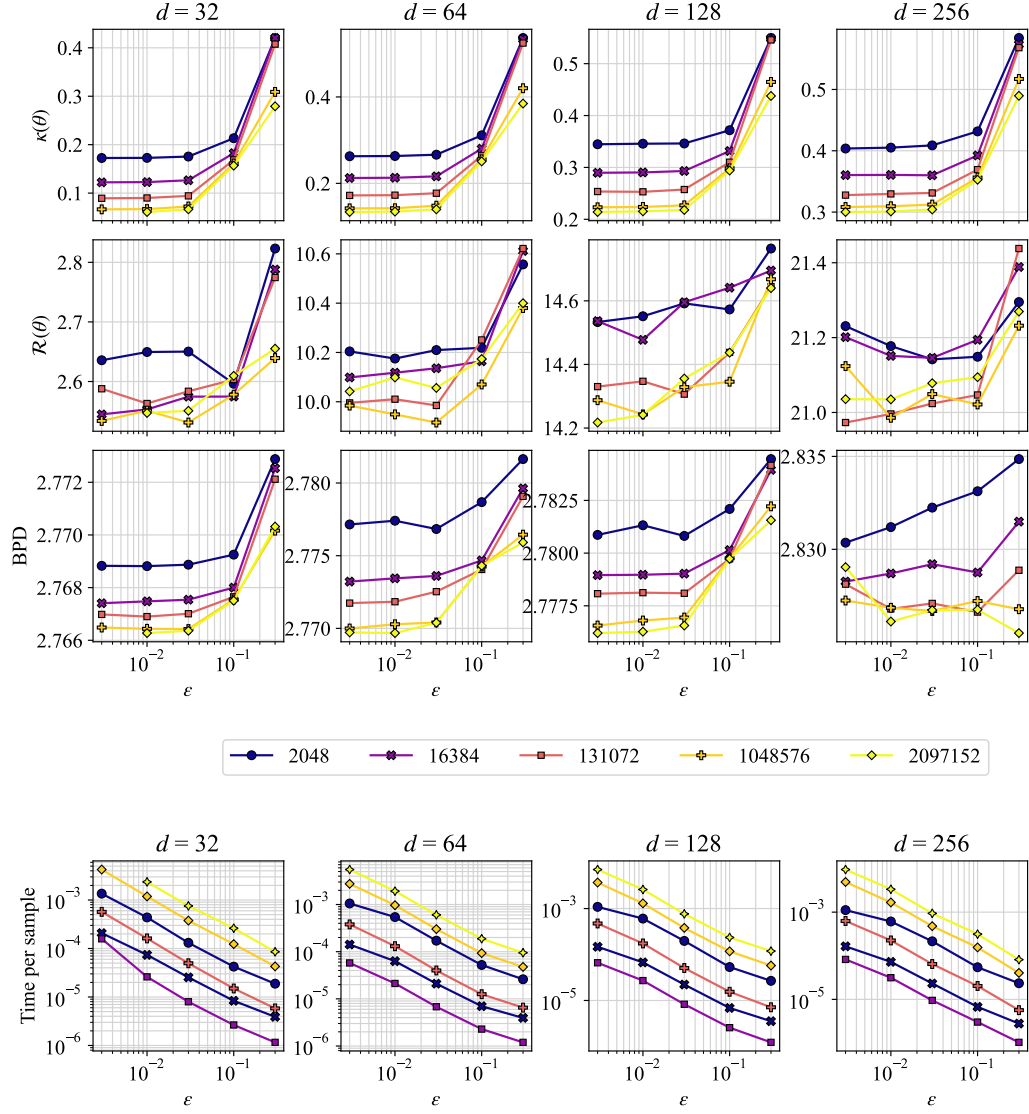


Figure 11: Plots corresponding to Figure 10 shown in previous page, on piecewise affine synthetic benchmark, using directly the relative epsilon parameter as the x-axis (log-scale), instead of renormalized entropy.

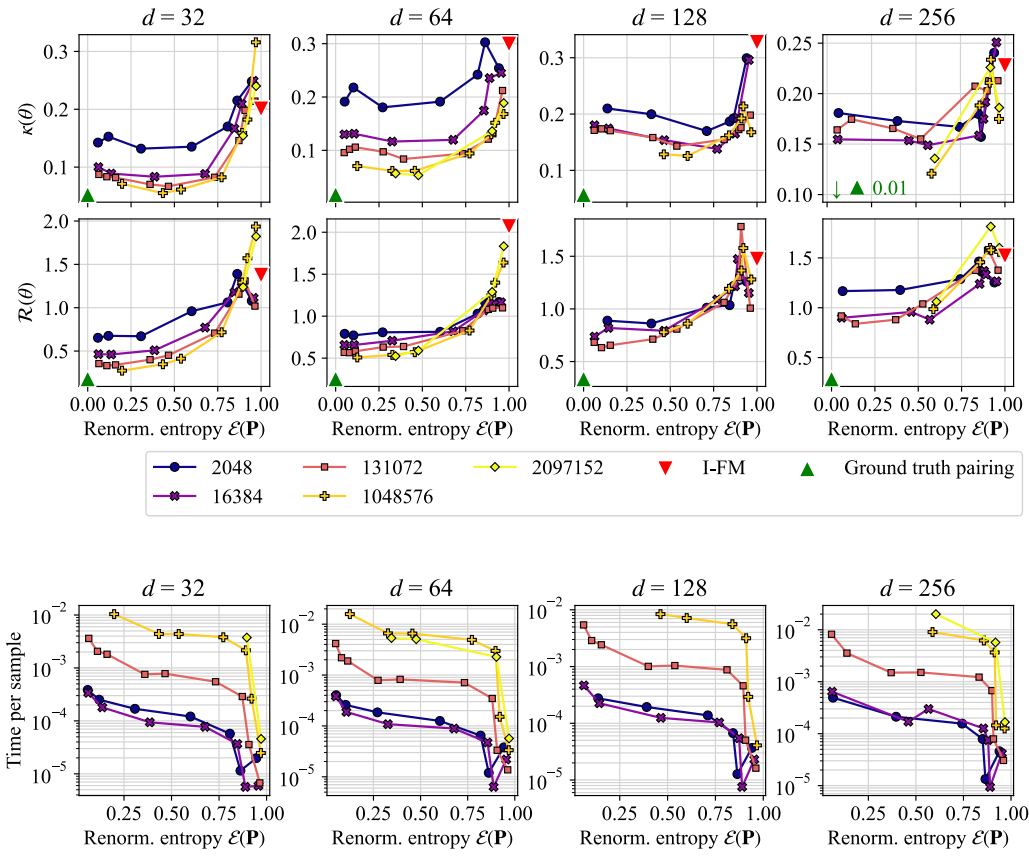


Figure 12: Final plots corresponding to Figure 2 in the main body of the paper (some points have been added following the convergence of some larger scale runs, using renormalized entropies)

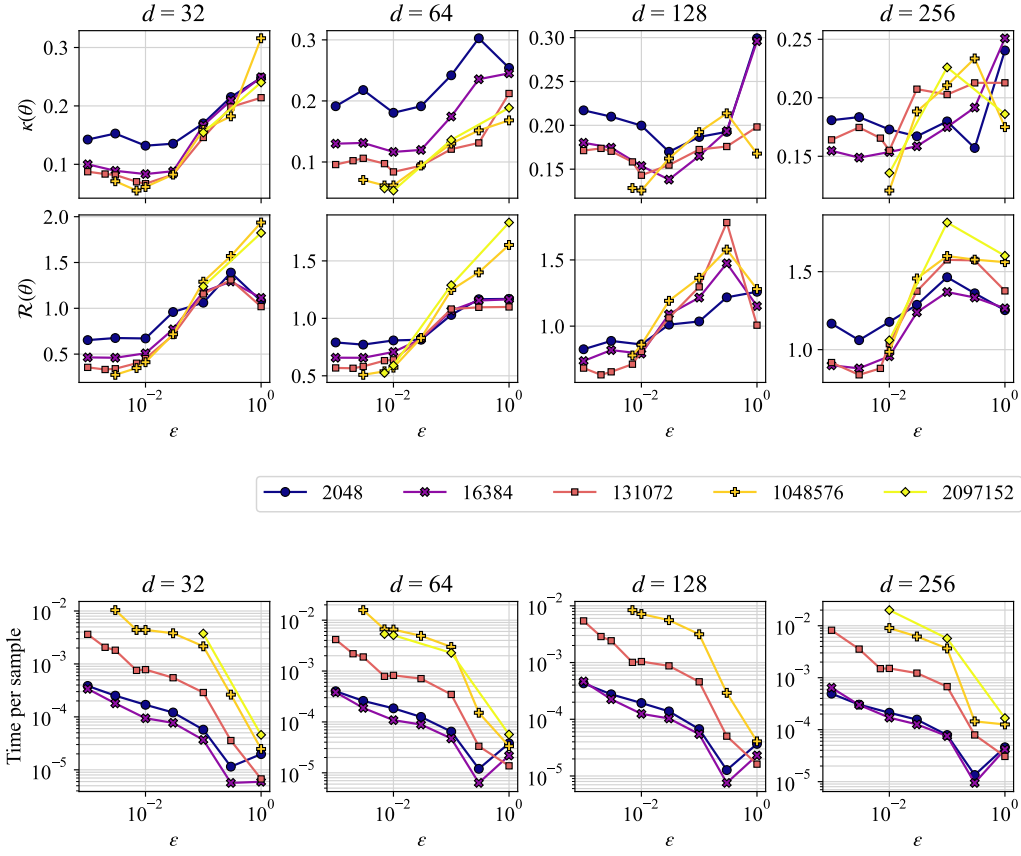


Figure 13: Final plots for the [Korotin et al.](#) benchmark, shown initially in Figure 12, using the relative epsilon  $\varepsilon$  parameter directly in the x-axis. Note that we have some missing runs at low entropy for very large batch sizes. That compute was used to reprioritize image generation tasks.

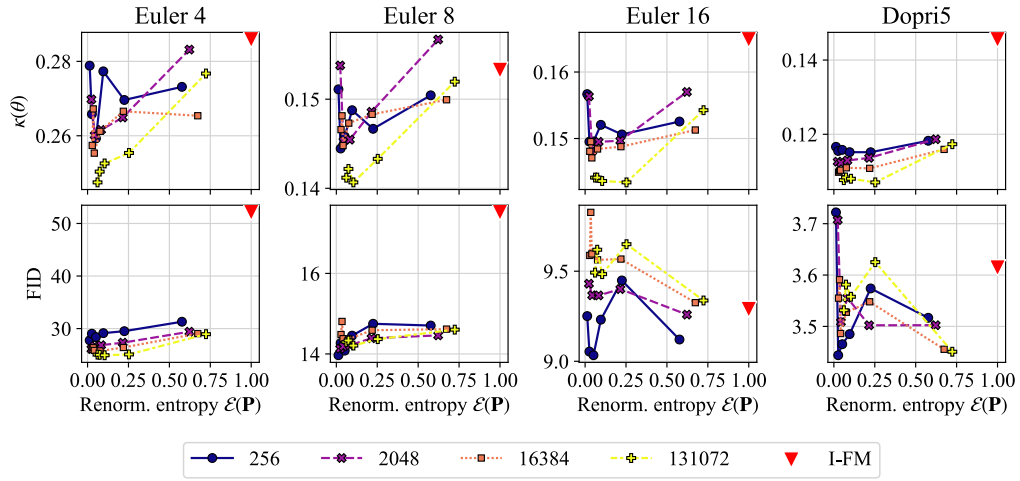


Figure 14: Final experiment metrics on **CIFAR-10** corresponding to those in Figure 3 in the main body of the paper.



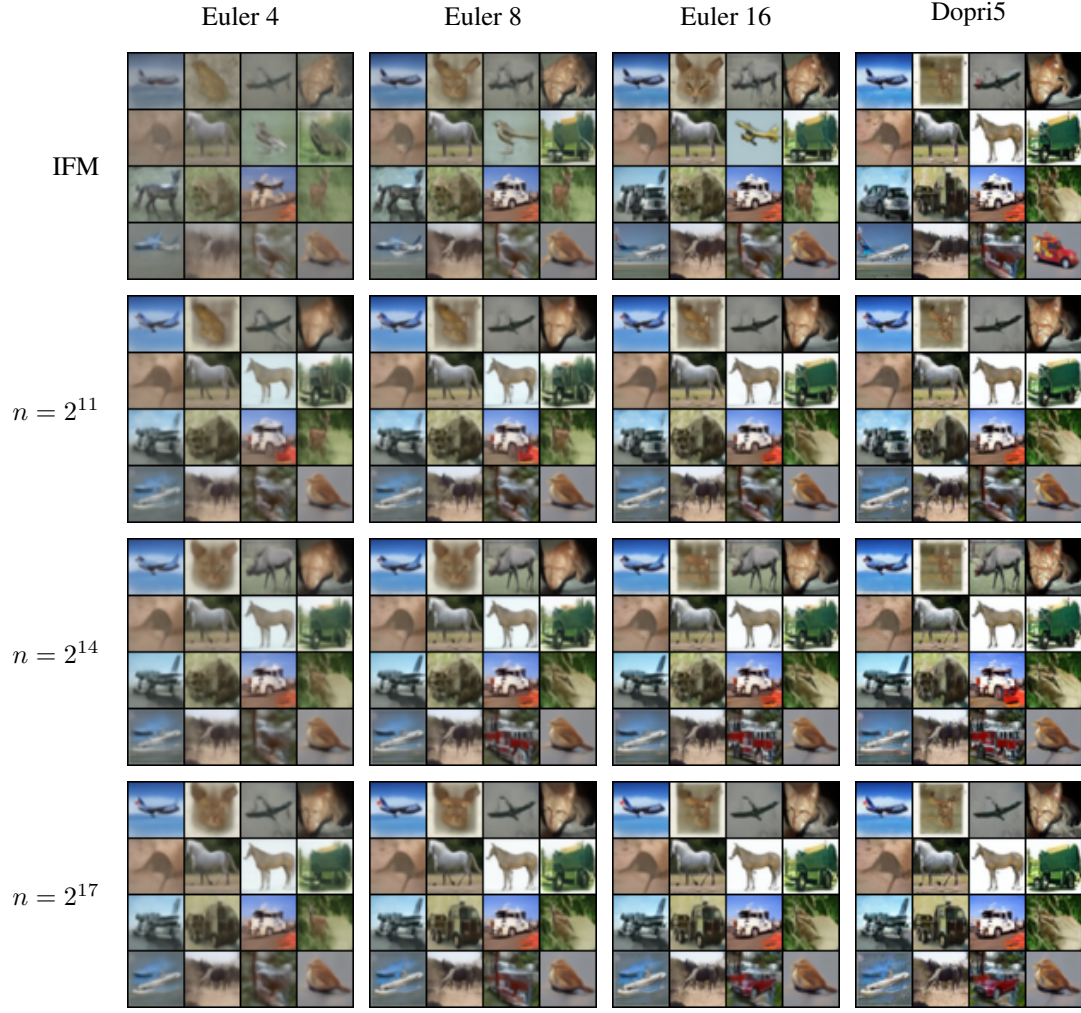


Figure 15: Non-curated images generated from models trained on **CIFAR-10**. The number following Euler denotes NFE, while Dopri5 uses an adaptive number of evaluations.  $n$  denotes the total batch size for the Sinkhorn algorithm. We use OT-FM models trained with  $\varepsilon = 0.01$ .

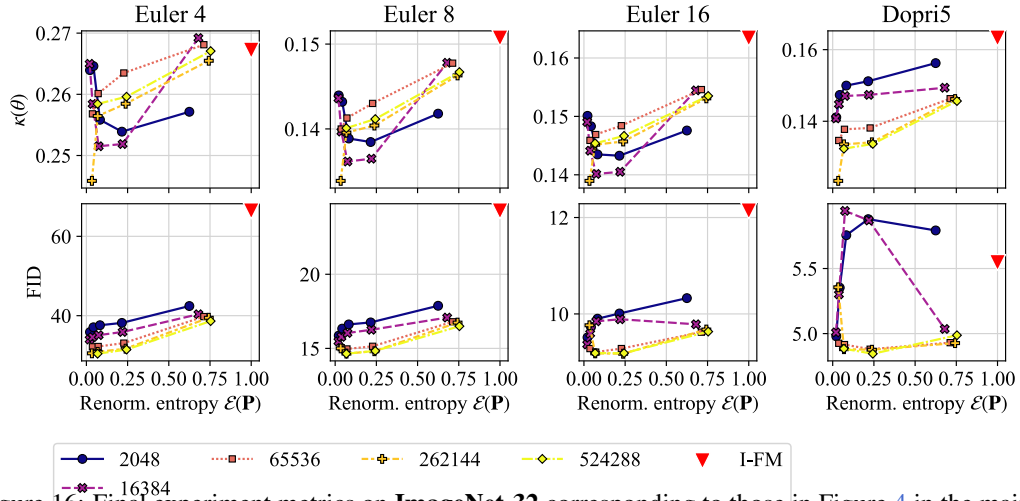


Figure 16: Final experiment metrics on **ImageNet-32** corresponding to those in Figure 4 in the main body of the paper. Our experiments consistently demonstrate that both FID and curvature are smaller when using larger OT batch size, and smaller renormalized entropy tends to result in better metrics.

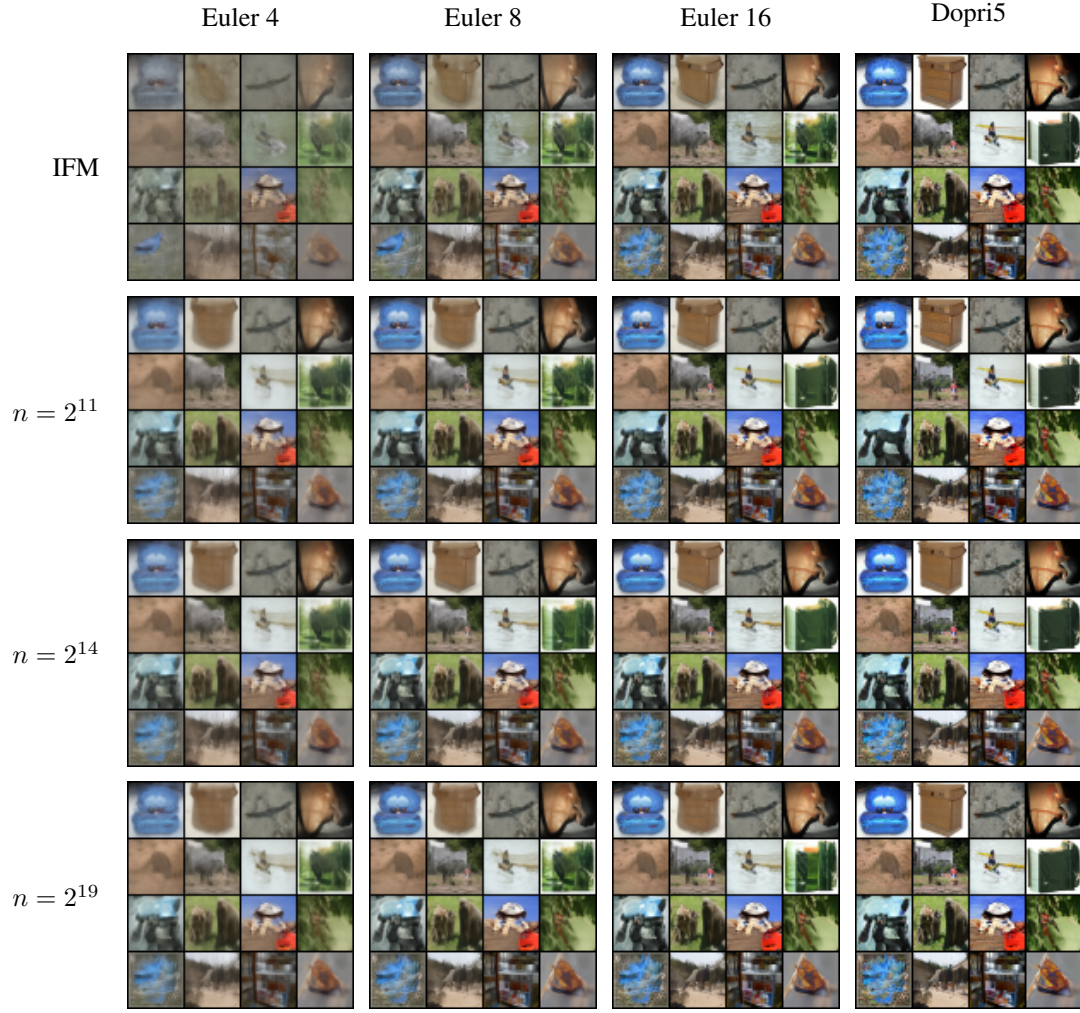


Figure 17: Non-curved images generated from models trained on **ImageNet-32**.  $n$  denotes the total batch size for the Sinkhorn algorithm. We use OT-FM models trained with  $\varepsilon = 0.1$ .

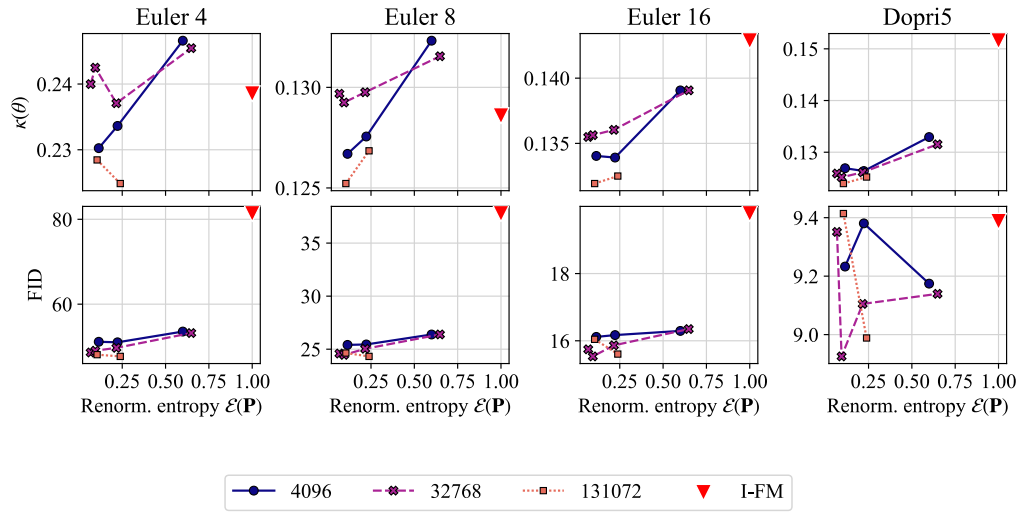
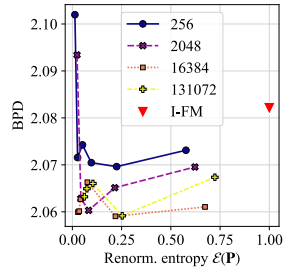


Figure 18: ImageNet64 results: Curvature and FID obtained with Euler integration with varying number of steps, as well as Dopri5 integration.

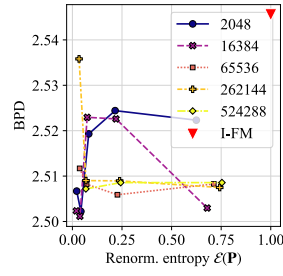




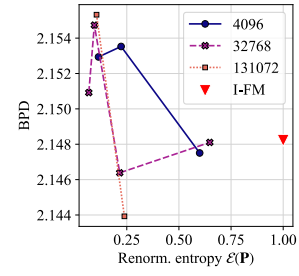
Figure 19: Non-curated images generated from models trained on **ImageNet-64**.  $n$  denotes the total batch size for the Sinkhorn algorithm. We use models trained with a varying trained with  $\varepsilon = 0.1$ .



(a) CIFAR-10



(b) ImageNet-32



(c) ImageNet-64

Figure 20: BPD for all three image generation tasks. The BPDs are computed using Dopri5 integration, evaluated on 50 times steps, and computed using 8 vectors for the Hutchinson trace estimator.