EAI Diffusion Reading Group

PFGM++: Unlocking the Potential of Physics-Inspired Generative Models

22/04/23

Summary

1. PFGM recap
2. PFGM++ derivation
3. Properties
4. Results

PFGM recap

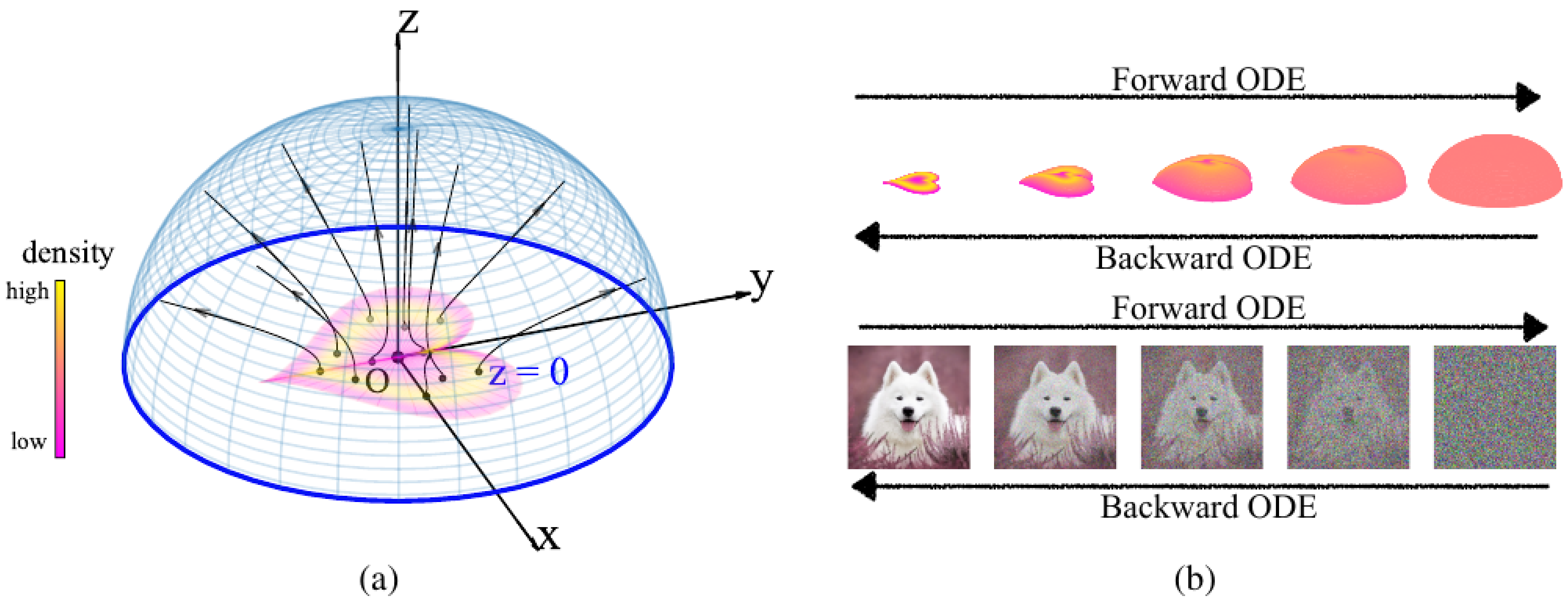


Figure 1: **(a)** 3D Poisson field trajectories for a heart-shaped distribution **(b)** The evolvements of a distribution (**top**) or an (augmented) sample (**bottom**) by the forward/backward ODEs pertained to the Poisson field.

PFGM Recap

- Augment data set with additional dimension: $\tilde{\mathrm{x}} = (\mathrm{x}, z)$
- dataset embedded in $z = 0$ hyperplane
- Learn (normalized) Poisson Field

$$E(\tilde{\mathrm{x}}) = \frac{1}{S_N(1)} \int \frac{\tilde{\mathrm{x}} - \tilde{\mathrm{y}}}{\|\tilde{\mathrm{x}} - \tilde{\mathrm{y}}\|^{N+1}} d\mathrm{y}$$

- Forward Process $d\tilde{\mathrm{x}} = E(\tilde{\mathrm{x}})dt$: maps data distribution p to uniform distribution on $N+1$ dimensional hemisphere

PFGM training

Algorithm 1 Learning the normalized Poisson Field

Input: Training iteration T, Initial model f_θ, dataset $\mathcal{D}$, constant γ, learning rate η.
for $t = 1 \dots T$ **do**
  Sample a large batch $\mathcal{B}_L$ from $\mathcal{D}$ and subsample a batch of datapoints $\mathcal{B} = \{\mathbf{x}_i\}_{i=1}^{|\mathcal{B}|}$ from $\mathcal{B}_L$
  Simulate the ODE: $\{\tilde{\mathbf{y}}_i = \text{perturb}(\mathbf{x}_i)\}_{i=1}^{|\mathcal{B}|}$
  Calculate the normalized field by $\mathcal{B}_L$: $\mathbf{v}_{\mathcal{B}_L}(\tilde{\mathbf{y}}_i) = -\sqrt{N}\hat{\mathbf{E}}_{\mathcal{B}_L}(\tilde{\mathbf{y}}_i)/(\| \hat{\mathbf{E}}_{\mathcal{B}_L}(\tilde{\mathbf{y}}_i) \|_2 + \gamma), \forall i$
  Calculate the loss: $\mathcal{L}(\theta) = \frac{1}{|\mathcal{B}|}\sum_{i=1}^{|\mathcal{B}|} \| f_\theta(\tilde{\mathbf{y}}_i) - \mathbf{v}_{\mathcal{B}_L}(\tilde{\mathbf{y}}_i) \|_2^2$
  Update the model parameter: $\theta = \theta - \eta\nabla\mathcal{L}(\theta)$
end for
return f_θ

Algorithm 2 perturb($\mathbf{x}$)

Sample the power $m \sim \mathcal{U}[0, M]$
Sample the initial noise $(\epsilon_{\mathbf{x}}, \epsilon_z) \sim \mathcal{N}(0, \sigma^2 I_{(N+1)\times(N+1)})$
Uniformly sample the vector from the unit ball $\mathbf{u} \sim \mathcal{U}(S_N(1))$
Construct training point $\mathbf{y} = \mathbf{x} + \| \epsilon_{\mathbf{x}} \| (1+\tau)^m \mathbf{u}$, $z = |\epsilon_z|(1+\tau)^m$
return $\tilde{\mathbf{y}} = (\mathbf{y}, z)$

Problems with PFGM

Training procedure is suboptimal

- Need large batch to approximate Poisson field
- Minimizer is biased estimator of true field
- Not compatible with paired sample training (conditional generation)

PFGM++ overview

1. Generalize by adding D new dimensions
2. Immediately throw them away. D is just a real-valued hyperparam
3. Use perturbation based objective (inspired by denoising score matching)
4. Profit?

Notations

- $\mathrm{x}, \mathrm{y} \in \mathbb{R}^N$: unaugmented datapoints
- $p(\mathrm{y})$ data distribution
- $\tilde{\mathrm{x}} = (\mathrm{x}, z)$: augmented datapoint
- $S_N(1)$ volume of N-dimensional unit hypersphere

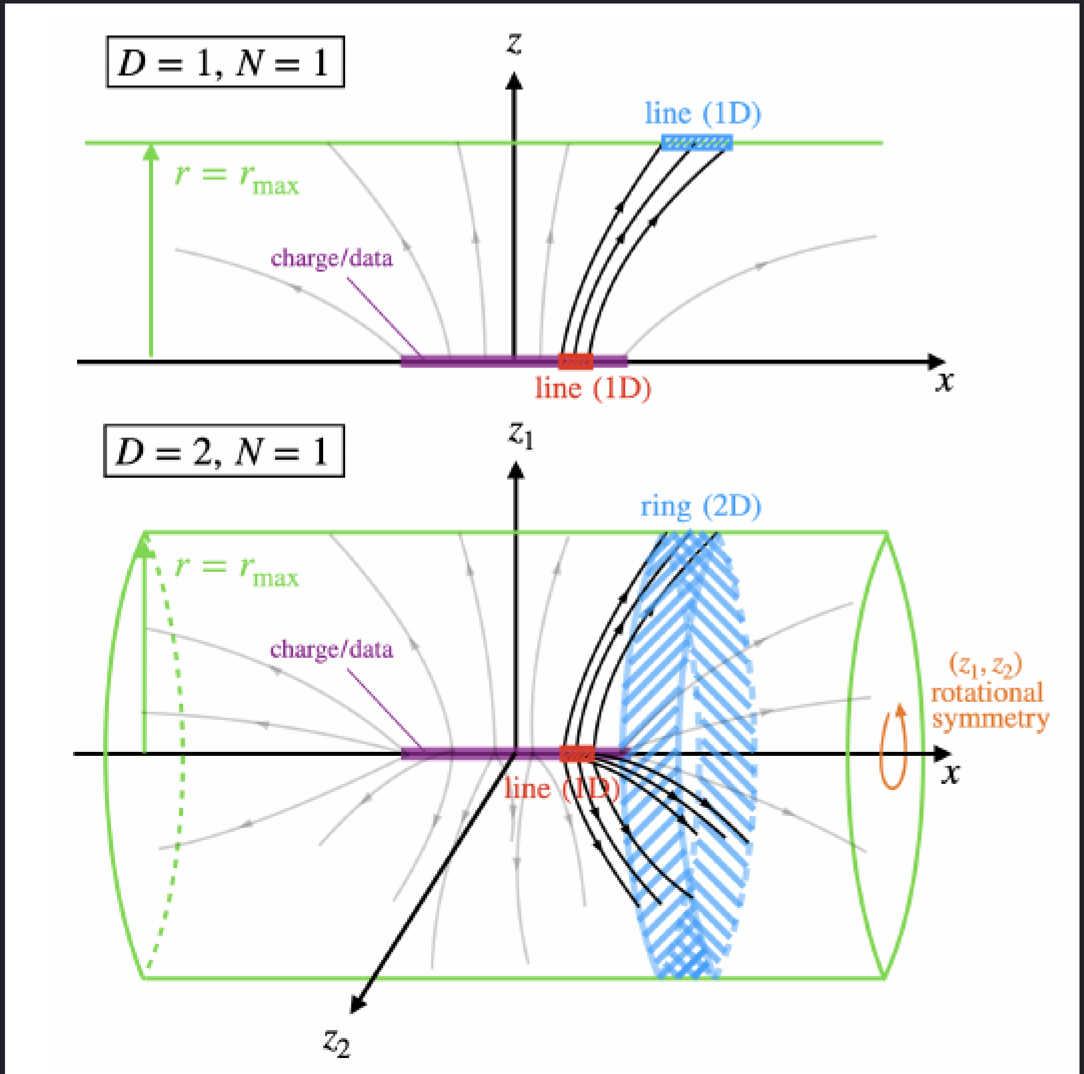
Adding D new dimensions, ...

New $N+D$ dimensional electric field

$$E(\tilde{\mathrm{x}}) = \frac{1}{S_{N+D-1}(1)} \int \frac{\tilde{\mathrm{x}} - \tilde{\mathrm{y}}}{\|\tilde{\mathrm{x}} - \tilde{\mathrm{y}}\|^{N+D}} dy$$

Defines a *surjection* between data distribution on $N+D$-dim hyperplane $\mathrm{z}=0$ and uniform distribution on $N+D$-dim hemisphere.

Or NOT!



Only one effective dimension needed $r(\tilde{\mathrm{x}}) = \|\mathrm{z}\|_2$. The field E is invariant under $SO(D)$ symmetries (rotations)

The actual augmentation

- $\tilde{\mathrm{x}} = (\mathrm{x}, r)$
- $\dfrac{dr}{dt} = \dfrac{1}{S_{N+D-1}(1)} \displaystyle\int \dfrac{r}{\|\tilde{\mathrm{x}} - \tilde{\mathrm{y}}\|^{N+D}} p(\mathrm{y}) d\mathrm{y} \equiv E(\tilde{x})_r$
- Forward process ODE

$$\frac{d\mathrm{x}}{dr} = \frac{E(\tilde{\mathrm{x}})_{\mathrm{x}}}{E(\tilde{x})_r}$$

Perturbation based objective

Old objective:

$$\mathbb{E}_{\tilde{p}_{\text{train}}}\mathbb{E}_{\{\mathrm{y}_i\}_{i=1}^n\sim p^n(\mathrm{y})}\mathbb{E}_{\mathrm{x}\sim p_\sigma(\mathrm{x}|\mathrm{y}_1)}\left[\left\| f_\theta(\tilde{\mathrm{x}}) - \frac{\sum_{i=1}^{n-1}\frac{\tilde{\mathrm{x}}-\tilde{\mathrm{y}}_i}{\|\tilde{\mathrm{x}}-\tilde{\mathrm{y}}_i\|^{N+D}}}{\left\|\sum_{i=1}^{n-1}\frac{\tilde{\mathrm{x}}-\tilde{\mathrm{y}}_i}{\|\tilde{\mathrm{x}}-\tilde{\mathrm{y}}_i\|^{N+D}}\right\|_2+\gamma}\right\|_2^2\right]$$

New objective:

$$\mathbb{E}_{r\sim p(r)}\mathbb{E}_{p(\mathrm{y})}\mathbb{E}_{p_r(\mathrm{x}|\mathrm{y})}\left[\left\| f_\theta(\tilde{\mathrm{x}}) - \frac{\tilde{\mathrm{x}}-\tilde{\mathrm{y}}}{r/\sqrt{D}}\right\|_2^2\right]$$

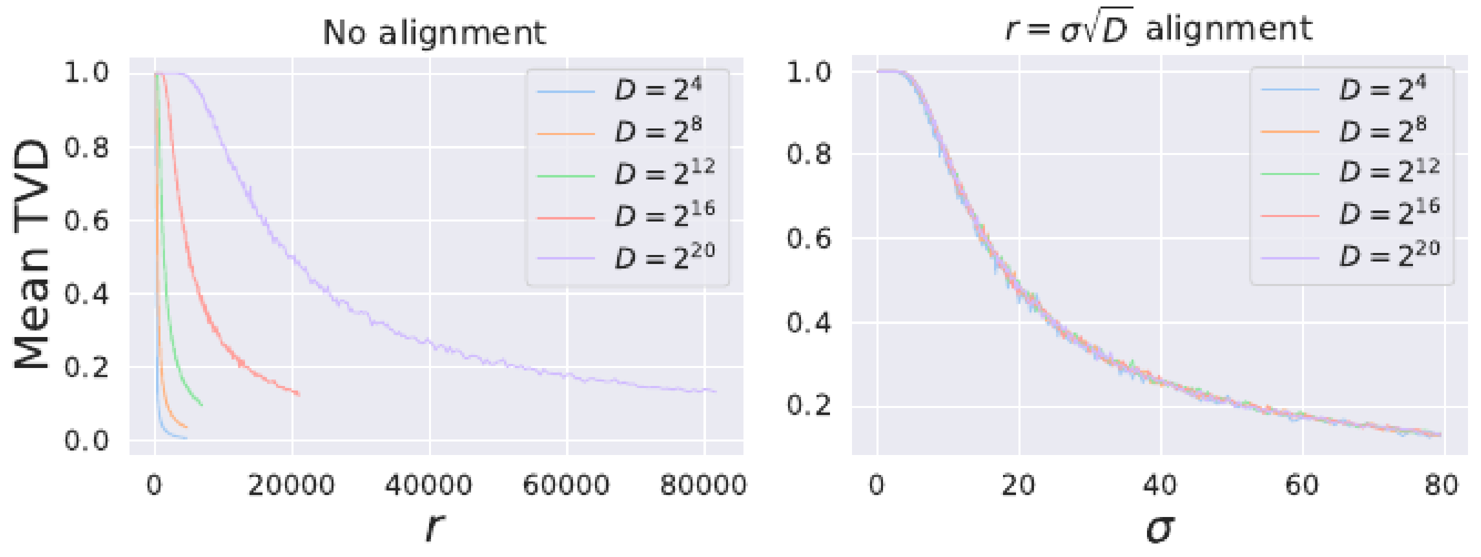
- $\tilde{\mathrm{y}} = (\mathrm{y}, 0)$, $\tilde{\mathrm{x}} = (\mathrm{x}, r)$
- perturbation kernel $p_r(\mathrm{x}|\mathrm{y}) \propto 1/(\|\mathrm{x}-\mathrm{y}\|_2^2 + r^2)^{\frac{N+D}{2}} = p_r(R)\mathcal{U}_\psi(\psi)$
- $p_r(R) \propto \frac{R^{N-1}}{(R^2+r^2)^{\frac{N+D}{2}}}$

$D \to \infty \Leftrightarrow$ Diffusion Models

$$\lim_{\substack{D \to \infty \\ r = \sigma\sqrt{D}}} \left\| \frac{\sqrt{D}}{E(\tilde{\mathrm{x}})_r} E(\tilde{\mathrm{x}})_{\mathrm{x}} - \sigma \nabla_{\mathrm{x}} \log p_{\sigma = r/\sqrt{D}}(\mathrm{x}) \right\|$$

Moreover, the training process are equivalent.

Hyperparameter transfer to finite D



(a) No alignment

(b) $r = \sigma\sqrt{D}$ alignment

Figure 3. Mean TVD between the posterior $p_{0|r}(\cdot|\mathbf{x})$ ($\mathbf{x}$ is perturbed sample) and the uniform prior, w/o **(a)** and w/ **(b)** the phase alignment ($r = \sigma\sqrt{D}$).

Algorithm 1 EDM training

Sample a batch of data $\{\mathbf{y}_i\}_{i=1}^{\mathcal{B}}$ from $p(\mathbf{y})$
Sample standard deviations $\{\sigma_i\}_{i=1}^{\mathcal{B}}$ from $p(\sigma)$
Sample noise vectors $\{\mathbf{n}_i \sim \mathcal{N}(0, \sigma_i^2 \boldsymbol{I})\}_{i=1}^{\mathcal{B}}$
Get perturbed data $\{\hat{\mathbf{y}}_i = \mathbf{y}_i + \mathbf{n}_i\}_{i=1}^{\mathcal{B}}$
Calculate loss $\ell(\theta) = \sum_{i=1}^{\mathcal{B}} \lambda(\sigma_i) \| f_\theta(\hat{\mathbf{y}}_i, \sigma_i) - \mathbf{y}_i \|_2^2$
Update the network parameter θ via Adam optimizer

Algorithm 2 PFGM++ training with hyperparameter transferred from EDM

Sample a batch of data $\{\mathbf{y}_i\}_{i=1}^{\mathcal{B}}$ from $p(\mathbf{y})$
Sample standard deviations $\{\sigma_i\}_{i=1}^{\mathcal{B}}$ from $p(\sigma)$
Sample r from p_r: $\{r_i = \sigma_i \sqrt{D}\}_{i=1}^{\mathcal{B}}$
Sample radiuses $\{R_i \sim p_{r_i}(R)\}_{i=1}^{\mathcal{B}}$
Sample uniform angles $\{\mathbf{v}_i = \frac{\mathbf{u}_i}{\|\mathbf{u}_i\|_2}\}_{i=1}^{\mathcal{B}}$, with $\mathbf{u}_i \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
Get perturbed data $\{\hat{\mathbf{y}}_i = \mathbf{y}_i + R_i \mathbf{v}_i\}_{i=1}^{\mathcal{B}}$
Calculate loss $\ell(\theta) = \sum_{i=1}^{\mathcal{B}} \lambda(\sigma_i) \| f_\theta(\hat{\mathbf{y}}_i, \sigma_i) - \mathbf{y}_i \|_2^2$
Update the network parameter θ via Adam optimizer

Algorithm 3 EDM sampling (Heun's 2nd order method)

$\mathbf{x}_0 \sim \mathcal{N}(\mathbf{0}, \sigma_{\max}^2 \boldsymbol{I})$
for $i = 0, \dots, T-1$ **do**
  $\mathbf{d}_i = (\mathbf{x}_i - f_\theta(\mathbf{x}_i, t_i))/t_i$
  $\mathbf{x}_{i+1} = \mathbf{x}_i + (t_{i+1} - t_i)\mathbf{d}_i$
  if $t_{i+1} > 0$ **then**
    $\mathbf{d}'_i = (\mathbf{x}_{i+1} - f_\theta(\mathbf{x}_{i+1}, t_{i+1}))/t_{i+1}$
    $\mathbf{x}_{i+1} = \mathbf{x}_i + (t_{i+1} - t_i)(\frac{1}{2}\mathbf{d}_i + \frac{1}{2}\mathbf{d}'_i)$
  end if
end for

Algorithm 4 PFGM++ training with hyperparameter transferred from EDM

Set $r_{\max} = \sigma_{\max}\sqrt{D}$
Sample radius $R \sim p_{r_{\max}}(R)$ and uniform angle $\mathbf{v} = \frac{\mathbf{u}}{\|\mathbf{u}\|_2}$, with $\mathbf{u} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
Get initial data $\mathbf{x}_0 = R\mathbf{v}$
for $i = 0, \dots, T-1$ **do**
  $\mathbf{d}_i = (\mathbf{x}_i - f_\theta(\mathbf{x}_i, t_i))/t_i$
  $\mathbf{x}_{i+1} = \mathbf{x}_i + (t_{i+1} - t_i)\mathbf{d}_i$
  if $t_{i+1} > 0$ **then**
    $\mathbf{d}'_i = (\mathbf{x}_{i+1} - f_\theta(\mathbf{x}_{i+1}, t_{i+1}))/t_{i+1}$
    $\mathbf{x}_{i+1} = \mathbf{x}_i + (t_{i+1} - t_i)(\frac{1}{2}\mathbf{d}_i + \frac{1}{2}\mathbf{d}'_i)$
  end if
end for

Results

Table 1. CIFAR-10 sample quality (FID) and number of function evaluations (NFE).

	Min FID ↓	Top-3 Avg FID ↓	NFE ↓
DDPM (Ho et al., 2020)	3.17	-	1000
DDIM (Song et al., 2021a)	4.67	-	50
VE-ODE (Song et al., 2021b)	5.29	-	194
VP-ODE (Song et al., 2021b)	2.86	-	134
PFGM (Xu et al., 2022)	2.48	-	104
PFGM++ (unconditional)			
$D = 64$	1.96	1.98	35
$D = 128$	1.92	1.94	35
$D = 2048$	**1.91**	**1.93**	35
$D = 3072000$	1.99	2.02	35
$D \to \infty$ (Karras et al., 2022)	1.98	2.00	35
PFGM++ (class-conditional)			
$D = 2048$	**1.74**	-	35
$D \to \infty$ (Karras et al., 2022)	1.79	-	35

Table 2. FFHQ sample quality (FID) with 79 NFE in unconditional setting

	Min FID ↓	Top-3 Avg FID ↓
$D = 128$	**2.43**	2.48
$D = 2048$	2.46	**2.47**
$D = 3072000$	2.49	2.52
$D \to \infty$ (Karras et al., 2022)	2.53	2.54

Results summary

- Finite D values outperform Diffusion Models (reaches SOTA)
- Sweet spot: $D = 2048$ (CIFAR-10), $D = 128$ (FFHQ)

Conclusions

- Fixes most problems with PFGM
- Hyperparam D as a continuum between PFGM ($D = 1$) and Diffusion $D = \infty$
- Free Hyperparam transfer from Diffusion Models