# Forward Operator Estimation in Generative Models with Kernel Transfer Operators

Zhichun Huang
Carnegie Mellon University

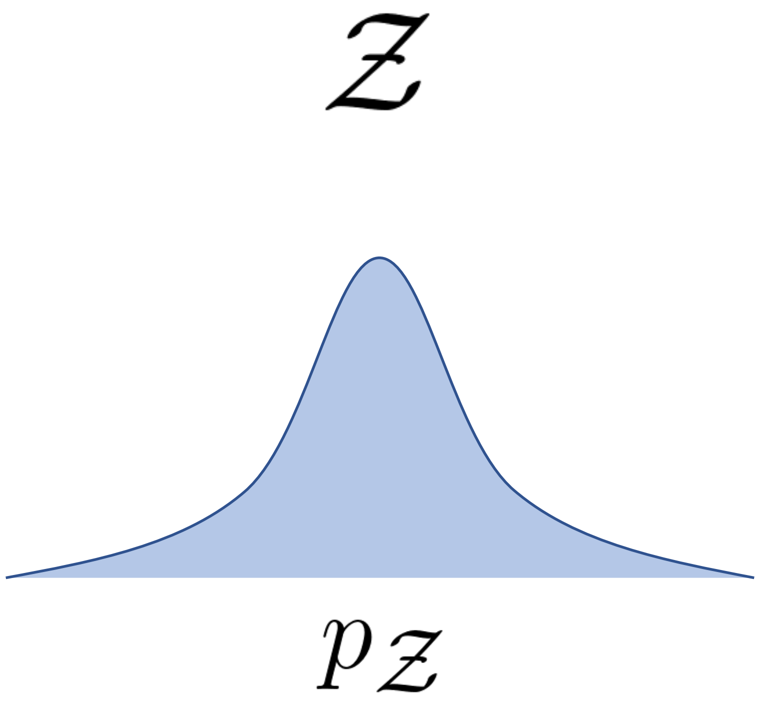
Rudrasis Chakraborty
Butlr

Vikas Singh
University of Wisconsin-Madison

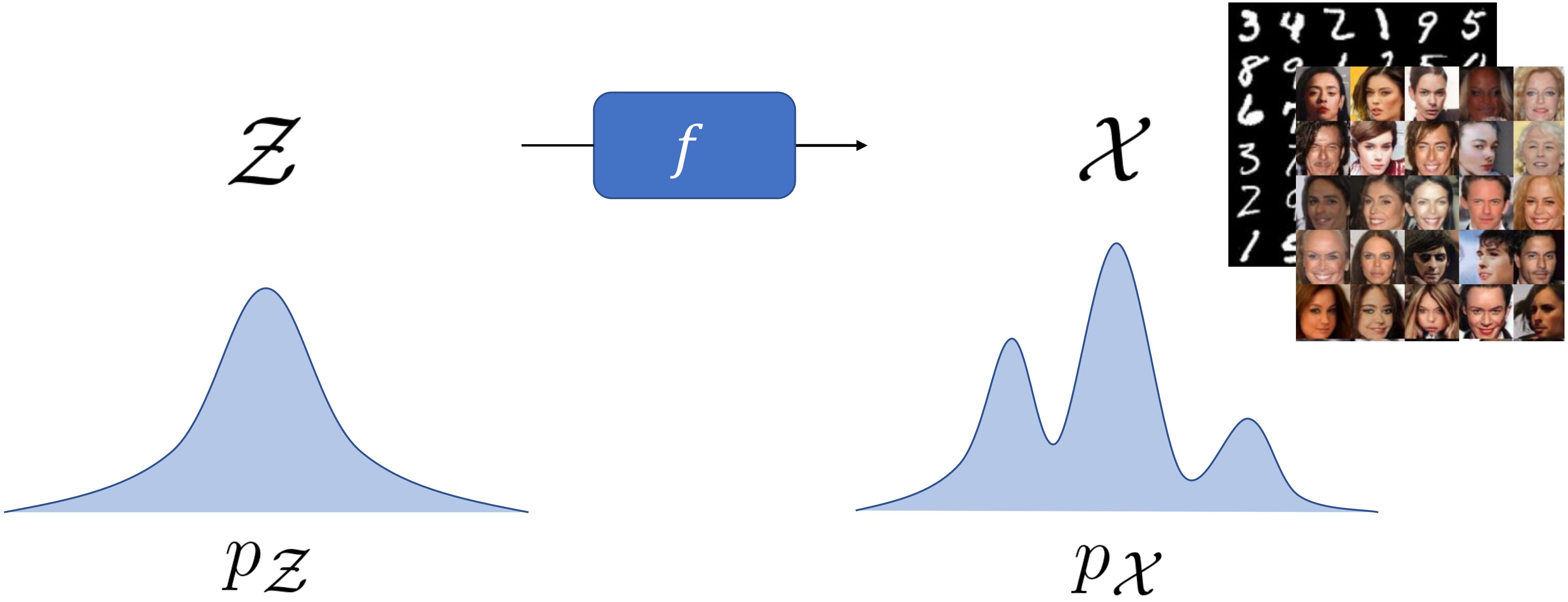
Carnegie Mellon University

WISCONSIN
UNIVERSITY OF WISCONSIN–MADISON

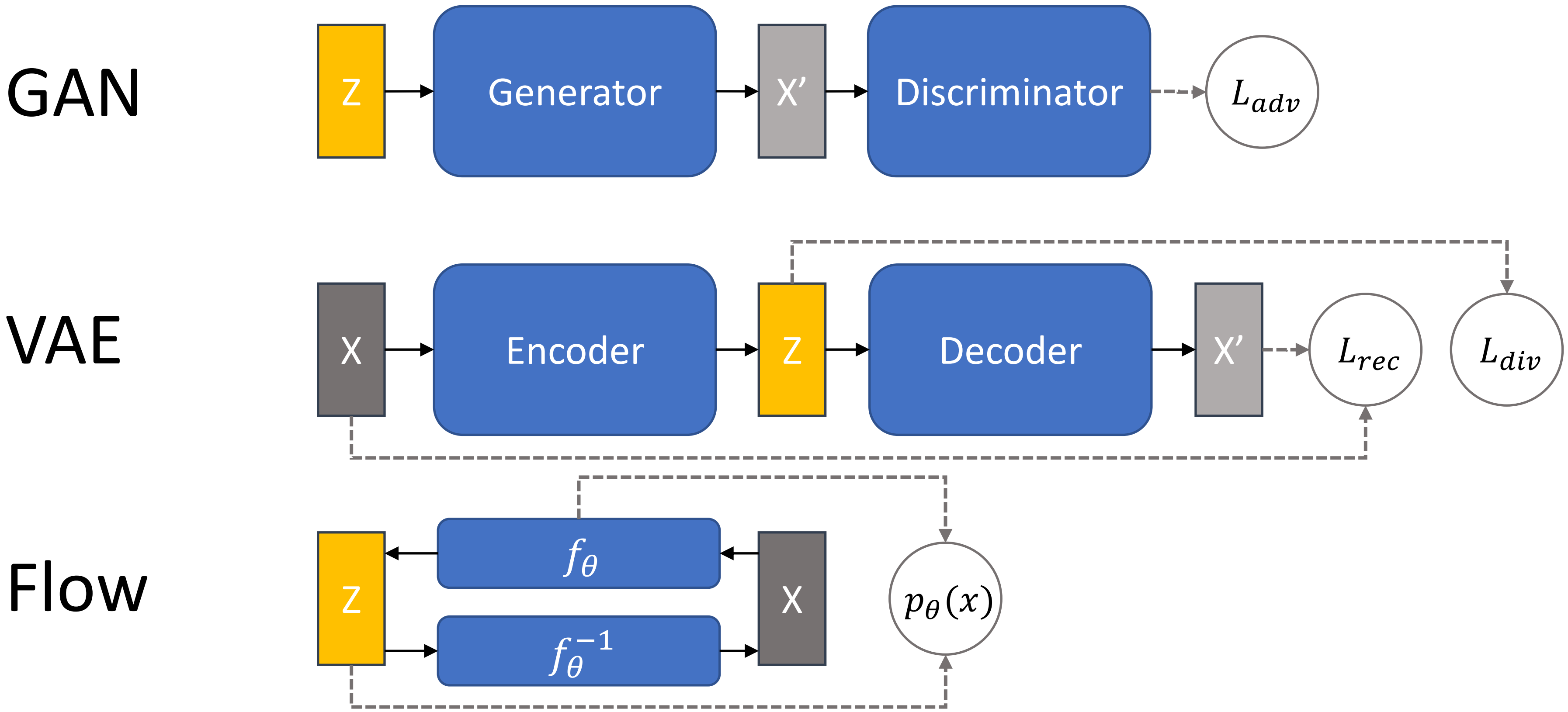
# Forward Operator in generative models



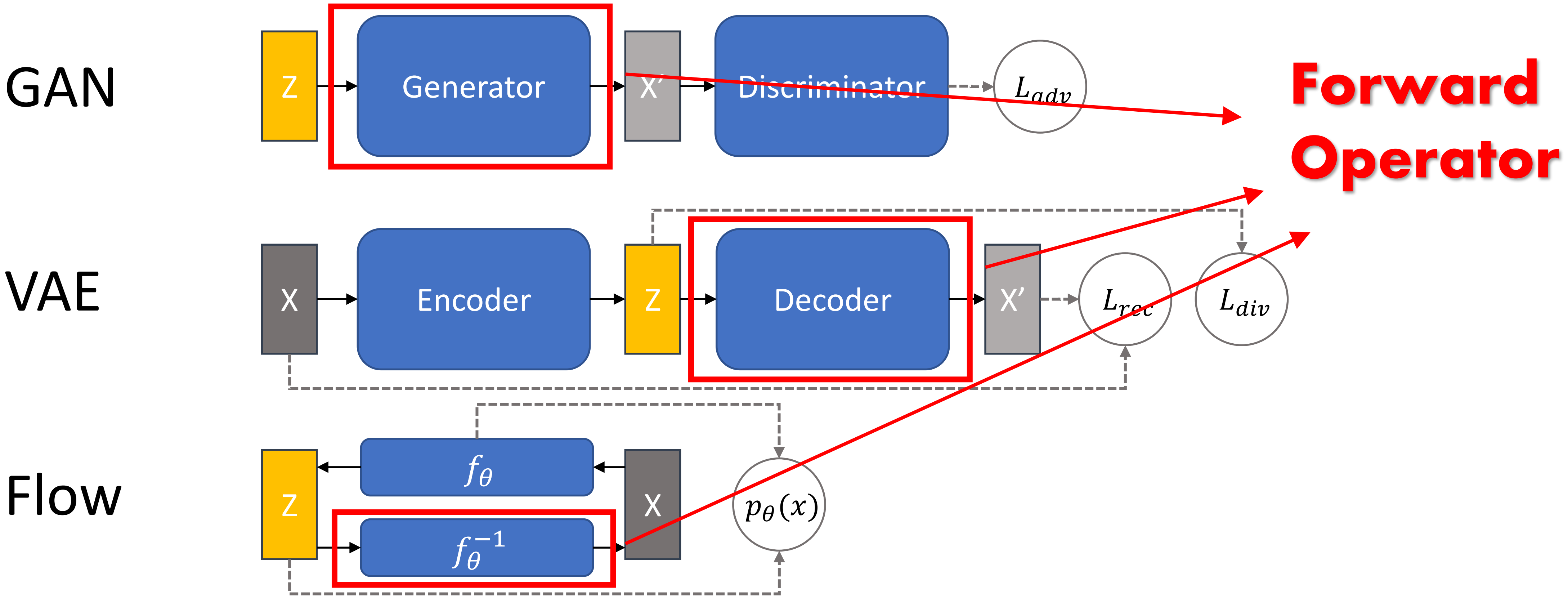
# Forward Operator in generative models



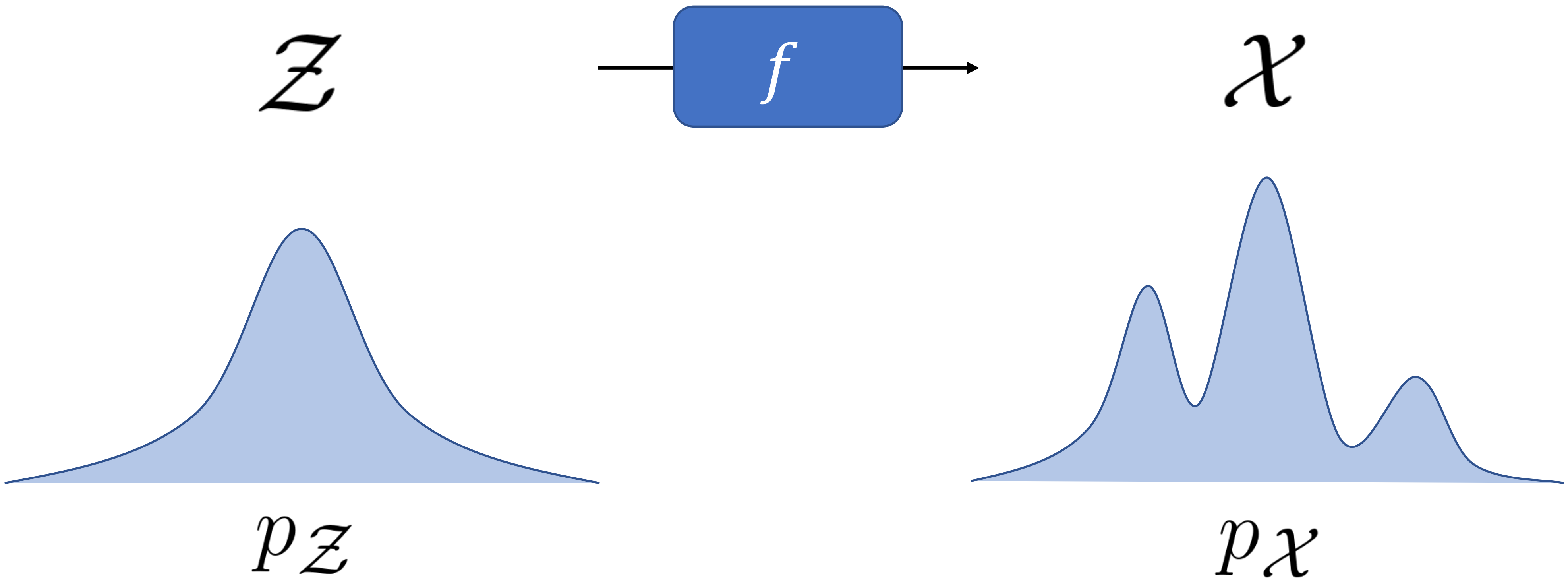
# Forward Operator in generative models



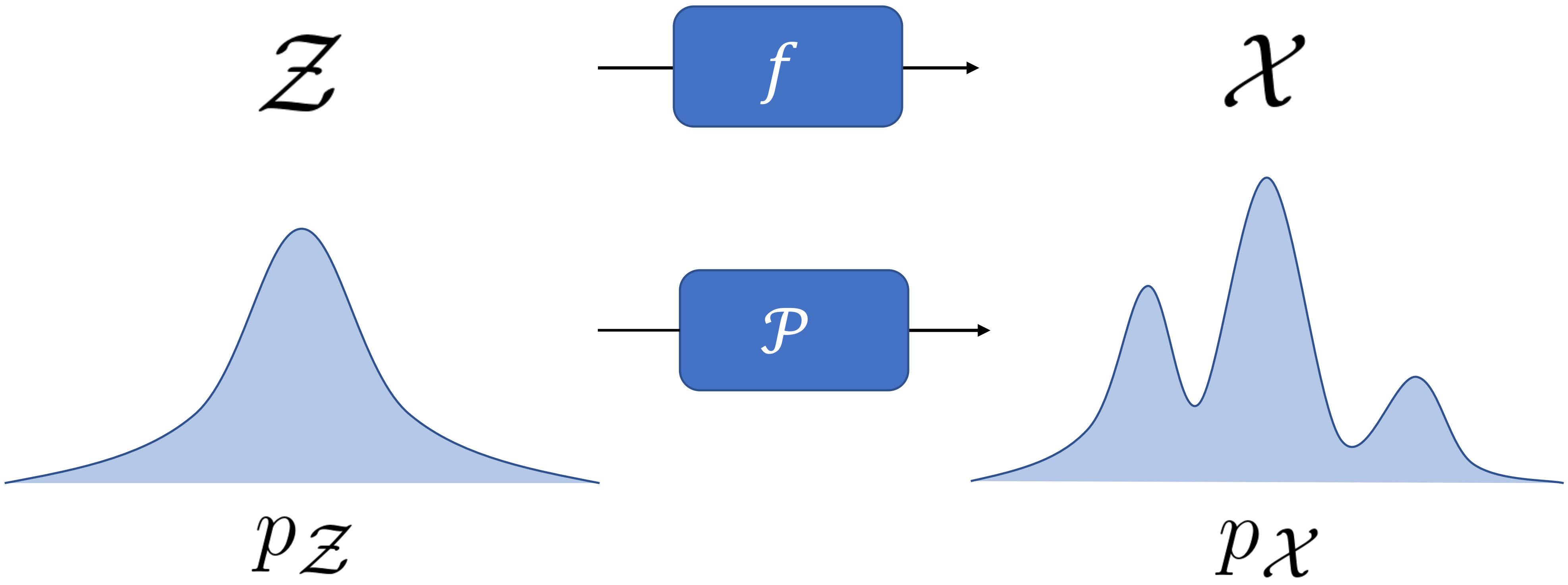
# Forward Operator in generative models



# Forward Operator in generative models



# Forward Operator in generative models



# Transfer operator for distributions

For a *non-singular* deterministic mapping $f$ on a measure space $(\mathbb{X}, \mathfrak{B}, \mu)$ , the transfer operator (or Perron-Frobenius operator) $\mathcal{P} : L^1(\mathbb{X}) \to L^1(\mathbb{X})$ is a *linear* operator defined as

$$\mathcal{P} \in \left\{ \int_\Lambda (\mathcal{P} p_{\mathcal{Z}}) d\mu = \int_{f^{-1}(\Lambda)} p_{\mathcal{Z}} d\mu = \int_\Lambda p_{\mathcal{X}} d\mu, \ \Lambda \in \mathfrak{B} \right\}$$

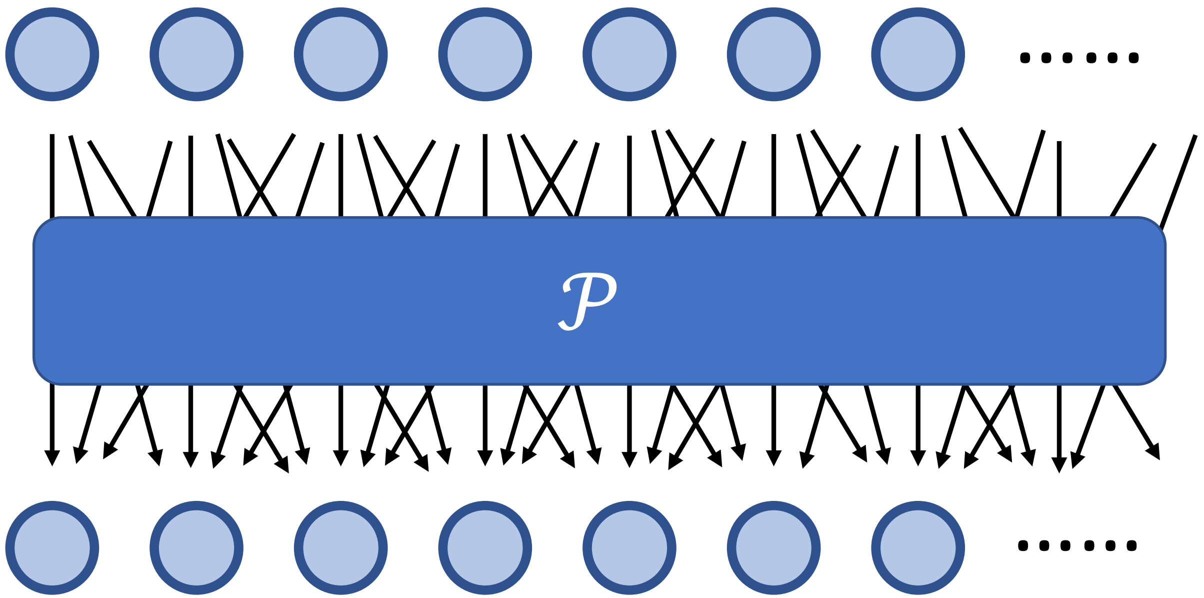
Learning the transfer operator has been studied for a long time in the context of dynamical systems [Preis et al., 2004][Klus et al., 2016]

Klus, S., Koltai, P., & Schütte, C. (2016). On the numerical approximation of the Perron-Frobenius and Koopman operator

Preis, R., Dellnitz, M., Hessel, M., Schütte, C., & Meerbach, E. (2004). Dominant paths between almost invariant sets of dynamical systems.

# Main challenges

To fully capture the dynamics, the transfer operator only exists on a sufficiently large space (i.e. a space supported by a large/infinite set of basis functions)

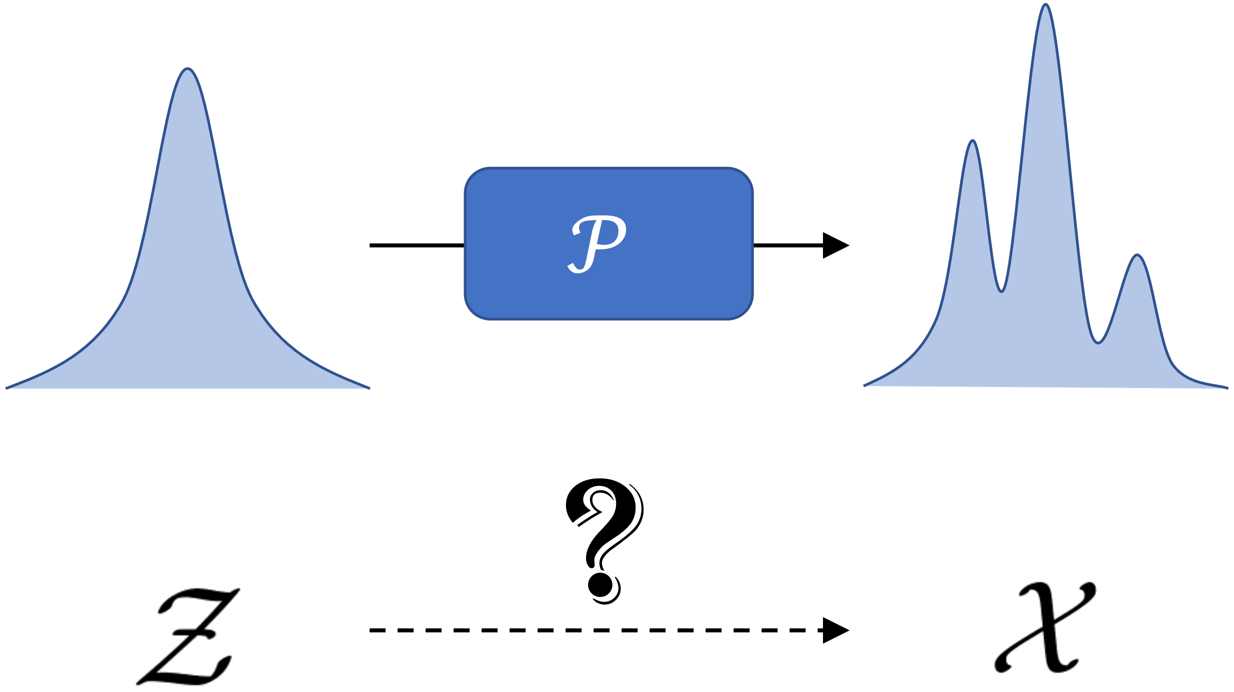


# Main challenges

We do not have direct access to the input density, but only its samples.

# Main challenges

The transfer operator is a ***function of density functions***. It is not immediately clear how to instantiate it on the input space and apply to samples.



# Kernel Perron-Frobenius Operator (KPF)

Use RKHS and kernel embeddings to address **ALL** challenges at once

Let $\phi, \psi$ be the feature mappings of the RKHS $\mathcal{H}, \mathcal{G}$ , respectively

The kernel mean embeddings of the distributions are defined as

$$\mu_{\mathcal{Z}} = \mathcal{E}_H(p_{\mathcal{Z}}) = \mathbb{E}_{\mathcal{Z}}[\phi(\mathcal{Z})]$$
$$\mu_{\mathcal{X}} = \mathcal{E}_G(p_{\mathcal{X}}) = \mathbb{E}_{\mathcal{X}}[\psi(\mathcal{X})]$$

Kernel Mean Embedding Operator

For characteristic kernels, the kernel mean embeddings are **injective**.

# Kernel Perron-Frobenius Operator (KPF)

A remarkable result by [Song et al., 2009] shows that

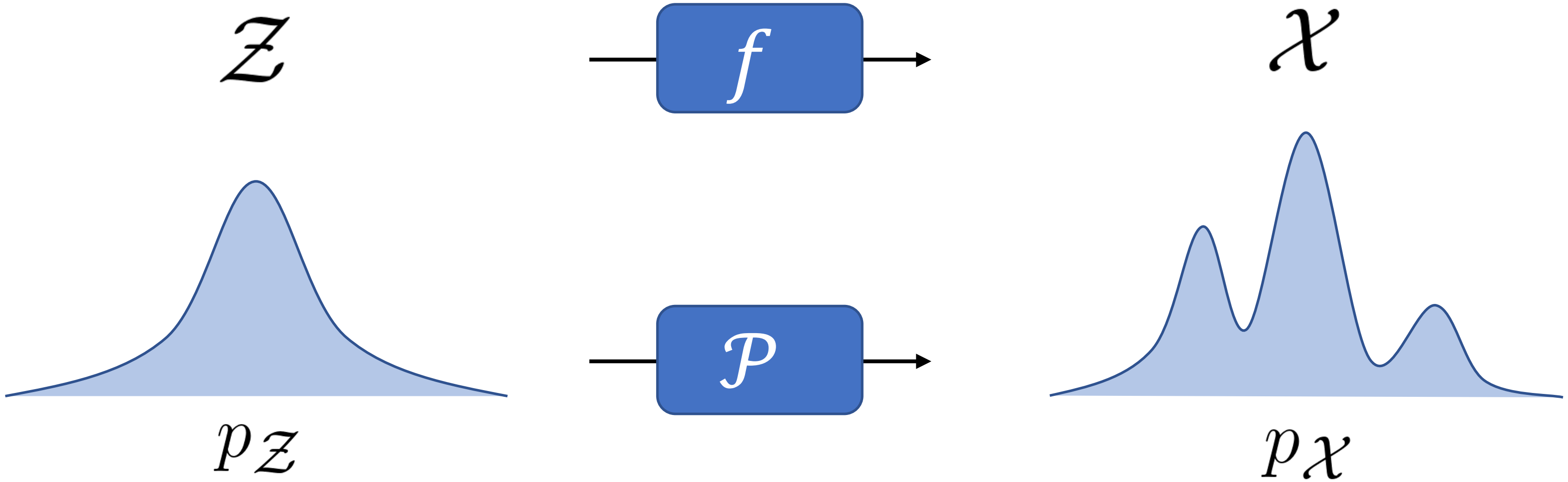
$$\mu_{\mathcal{X}} = \mathcal{U}_{\mathcal{X}|\mathcal{Z}}\mu_{\mathcal{Z}} = \boxed{\mathcal{C}_{\mathcal{X}\mathcal{Z}}\mathcal{C}_{\mathcal{X}\mathcal{X}}^{-1}}\mu_{\mathcal{Z}}$$

[Klus et al., 2017]

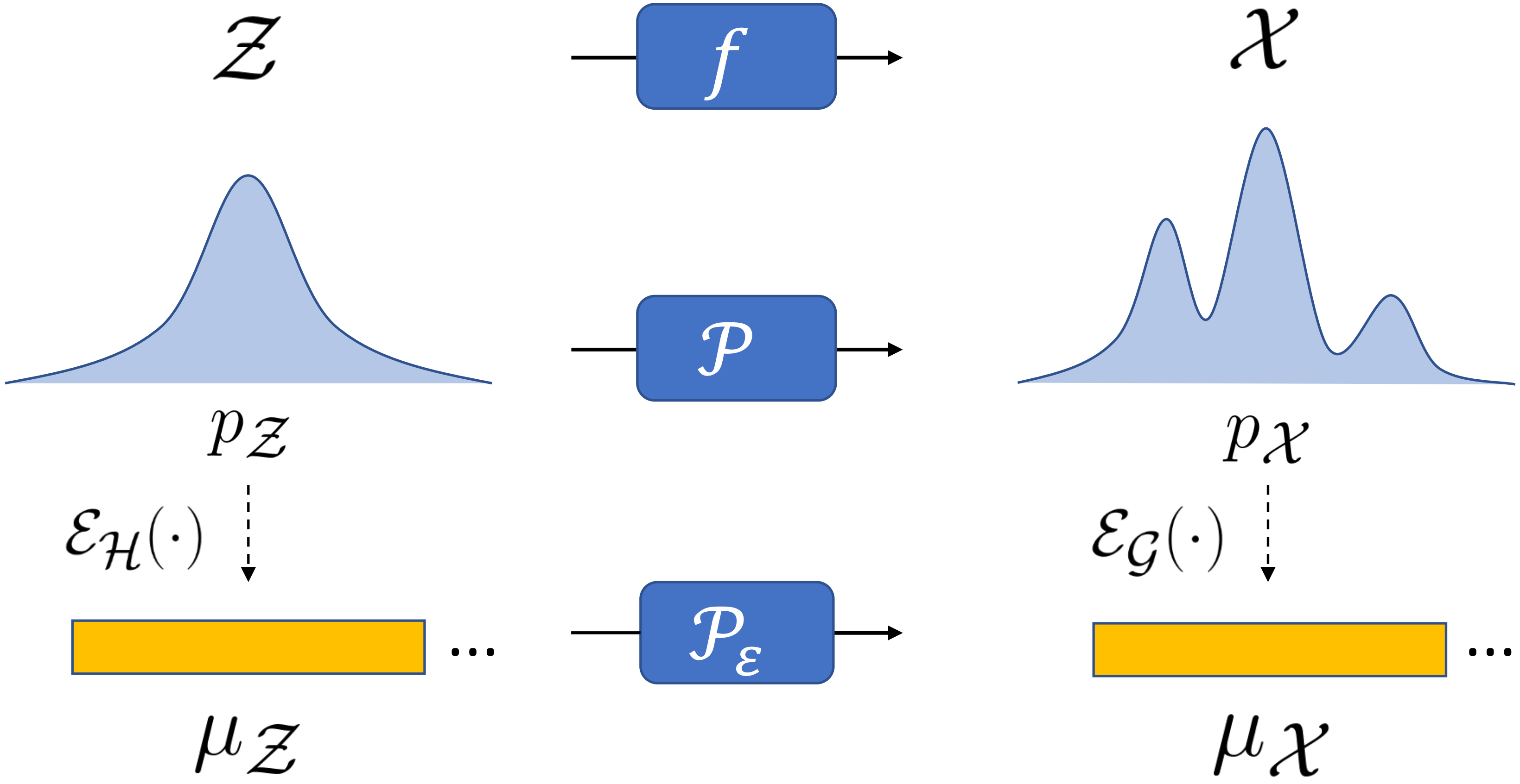
$\mathcal{P}_{\mathcal{E}}$ (KPF)

where $\mathcal{C}_{\mathcal{X}\mathcal{Z}}$ and $\mathcal{C}_{\mathcal{X}\mathcal{X}}$ are the (uncentered) covariance/cross-covariance operators

# Kernel Perron-Frobenius Operator (KPF)

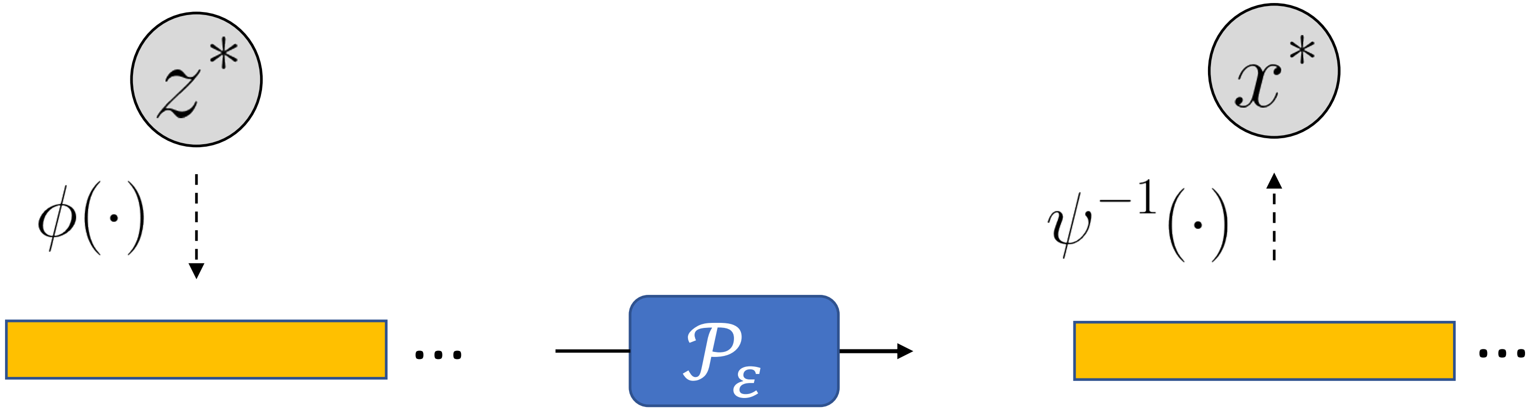


# Kernel Perron-Frobenius Operator (KPF)



# Sampling using KPF

- We can sample $x^* \sim \mathcal{X}^*$ from $\mathcal{X}^* = \psi^{-1}(\mathcal{P}_{\mathcal{E}}\phi(\mathcal{Z}))$




(However, it is often not the case…)

- When the *pre-image map* $\psi^{-1}(\cdot)$ can be computed exactly, we have

$$\mu_{\mathcal{X}^*} = \mathbb{E}_{\mathcal{X}^*}[\psi(\mathcal{X}^*)] = \mathbb{E}_{\mathcal{Z}}[\mathcal{P}_{\mathcal{E}}\phi(\mathcal{Z})] = \mathcal{P}_{\mathcal{E}}\mu_Z = \mu_{\mathcal{X}}$$

# Empirical form of KPF

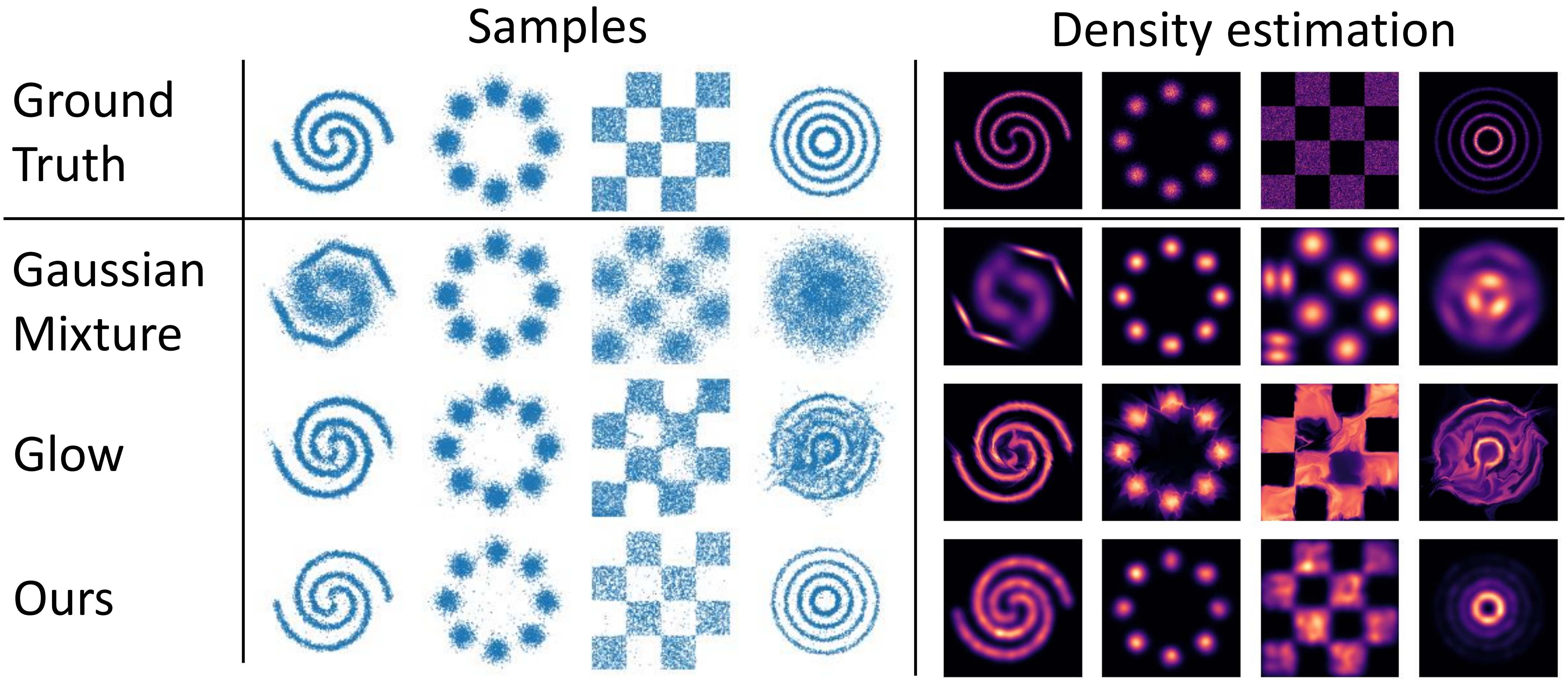
KPF can be estimated empirically through sample estimates of (cross-)covariance operators

Let $\Phi$ and $\Psi$ be the RKHS feature maps of samples of $\mathcal{Z}$ and $\mathcal{X}$

$$\hat{\mathcal{P}}_{\mathcal{E}} = \hat{\mathcal{C}}_{\mathcal{X}\mathcal{Z}}\hat{\mathcal{C}}_{\mathcal{X}\mathcal{Z}}^{-1} = (\frac{1}{n}\Psi\Phi^\top)(\frac{1}{n}\Phi\Phi^\top)^{-1} = \boxed{\Psi(\Phi^\top\Phi)^{-1}\Phi}$$

Notice that the empirical KPF has the exact form of kernel ridge regression!

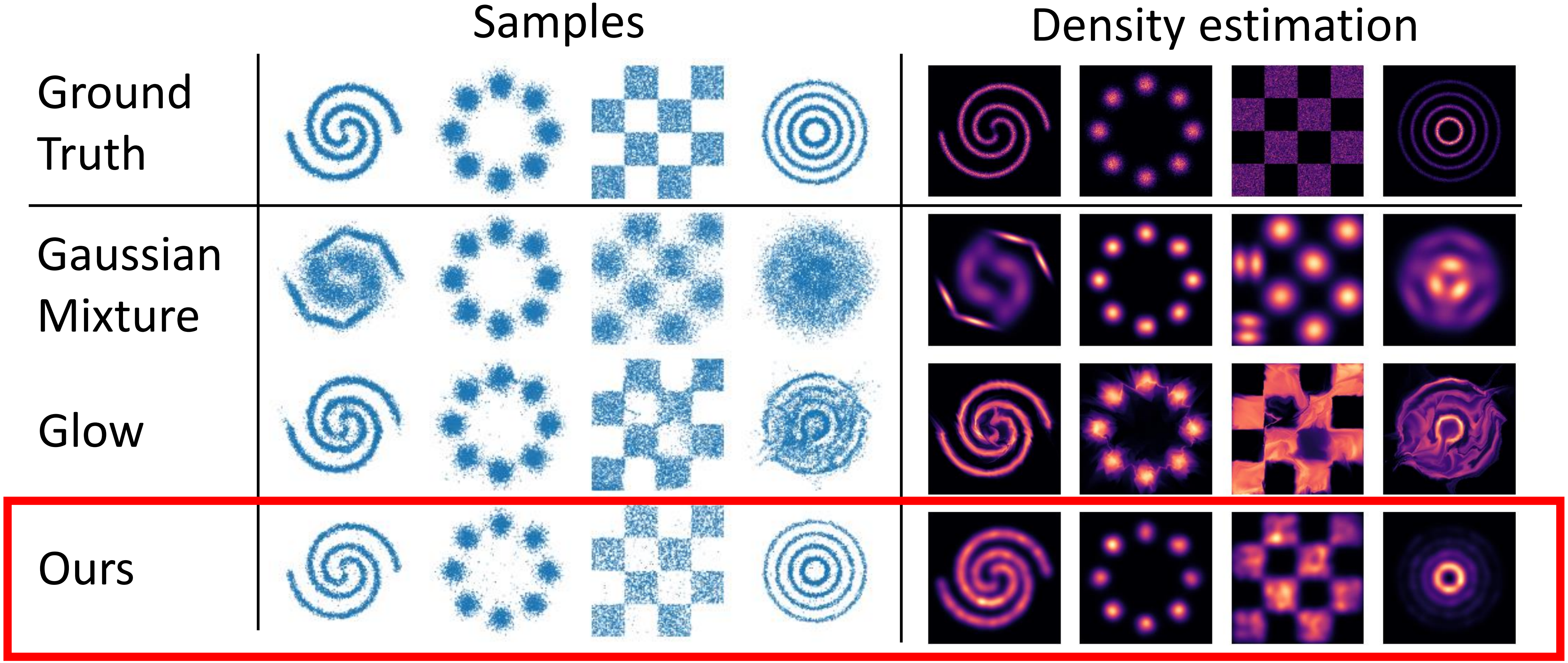
# Distribution learning on toy data




(Schuster et al., 2020)

Schuster, I., Mollenhauer, M., Klus, S. & Muandet, K.. (2020). Kernel Conditional Density Operators.

# Distribution learning on toy data




(Schuster et al., 2020)

Schuster, I., Mollenhauer, M., Klus, S. & Muandet, K.. (2020). Kernel Conditional Density Operators.

# Image generation

- Image data often lives on a low-dimensional manifold in a large ambient space.
- Directly computing the pre-images in the ambient space likely would not produce reasonable, *in-distribution* samples.

$+$

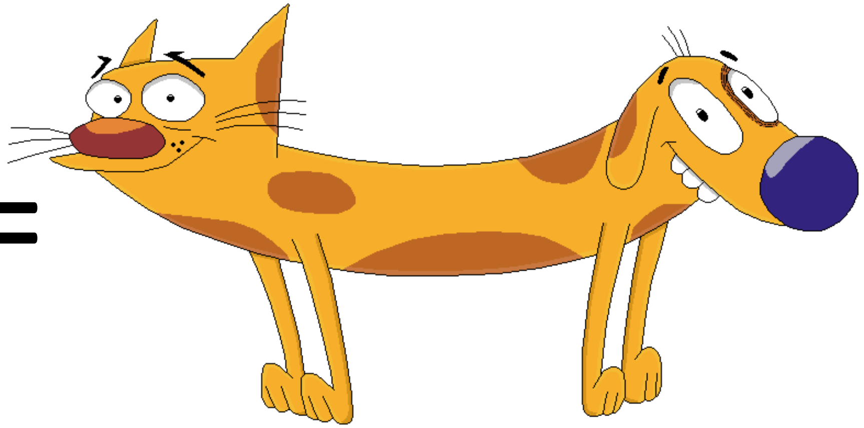
$\stackrel{?}{=}$

# Image generation

- Image data often lives on a low-dimensional manifold in a large ambient space.
- Directly computing the pre-images in the ambient space likely would not produce reasonable, *in-distribution* samples.

+

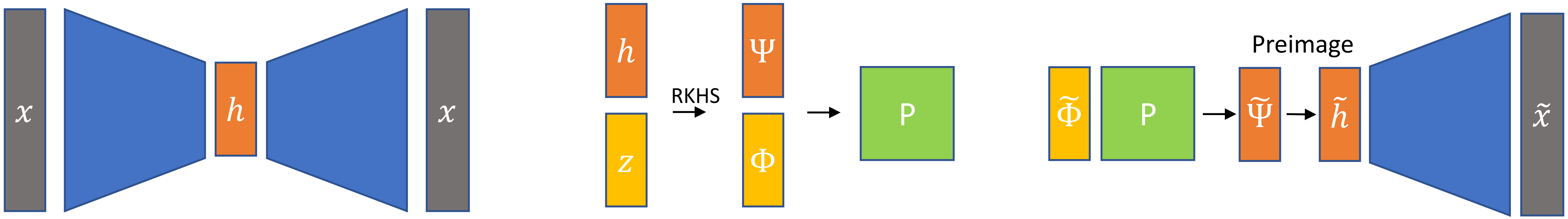
=



# Image generation

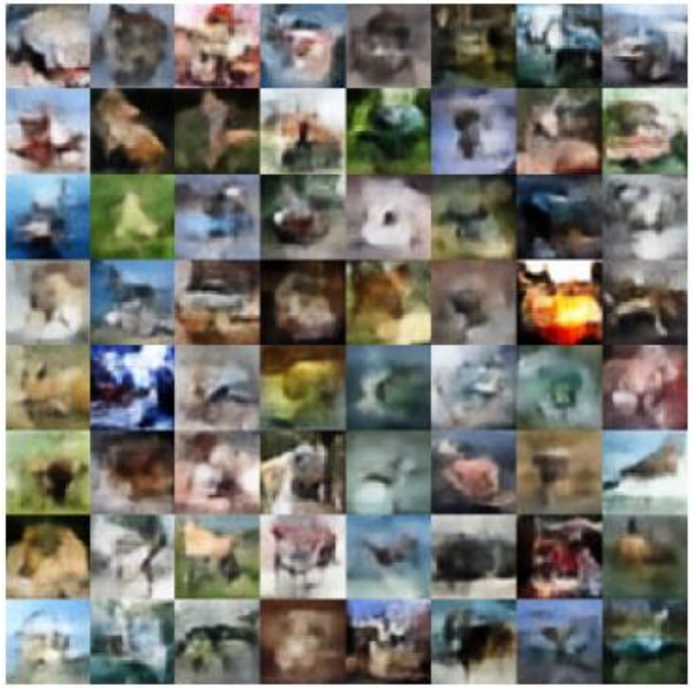
- Similar to [Li et al., 2015], we use a pretrained autoencoder and estimates the density on the latent space

Step 1: Train a (regularized) AE     Step 2: Estimate KPF on latent space     Step 3: Generate new samples




Li, Y., Swersky, K., & Zemel, R. (2015, June). Generative moment matching networks.

# Image generation



| | Glow‡ | CAGlow‡ | Vanilla VAE | WAE† | 2-stage VAE | SRAE$_{Glow}$ | SRAE$_{GMM}$ | SRAE$_{RBF-kPF}$ (ours) | SRAE$_{NTK-kPF}$ (ours) |
|---|---|---|---|---|---|---|---|---|---|
| MNIST | 25.8 | 26.3 | 36.5 | 20.4 | 18.3 | **15.5** | 16.7 | 19.7 | 19.5 |
| CIFAR-10 | - | - | 111.0 | 117.4 | 110.3 | 85.9 | 79.2 | 77.9 | **77.5** |
| CelebA | 103.7 | 104.9 | 52.1 | 53.7 | 44.7 | **35.0** | 42.0 | 41.9 | 41.0 |

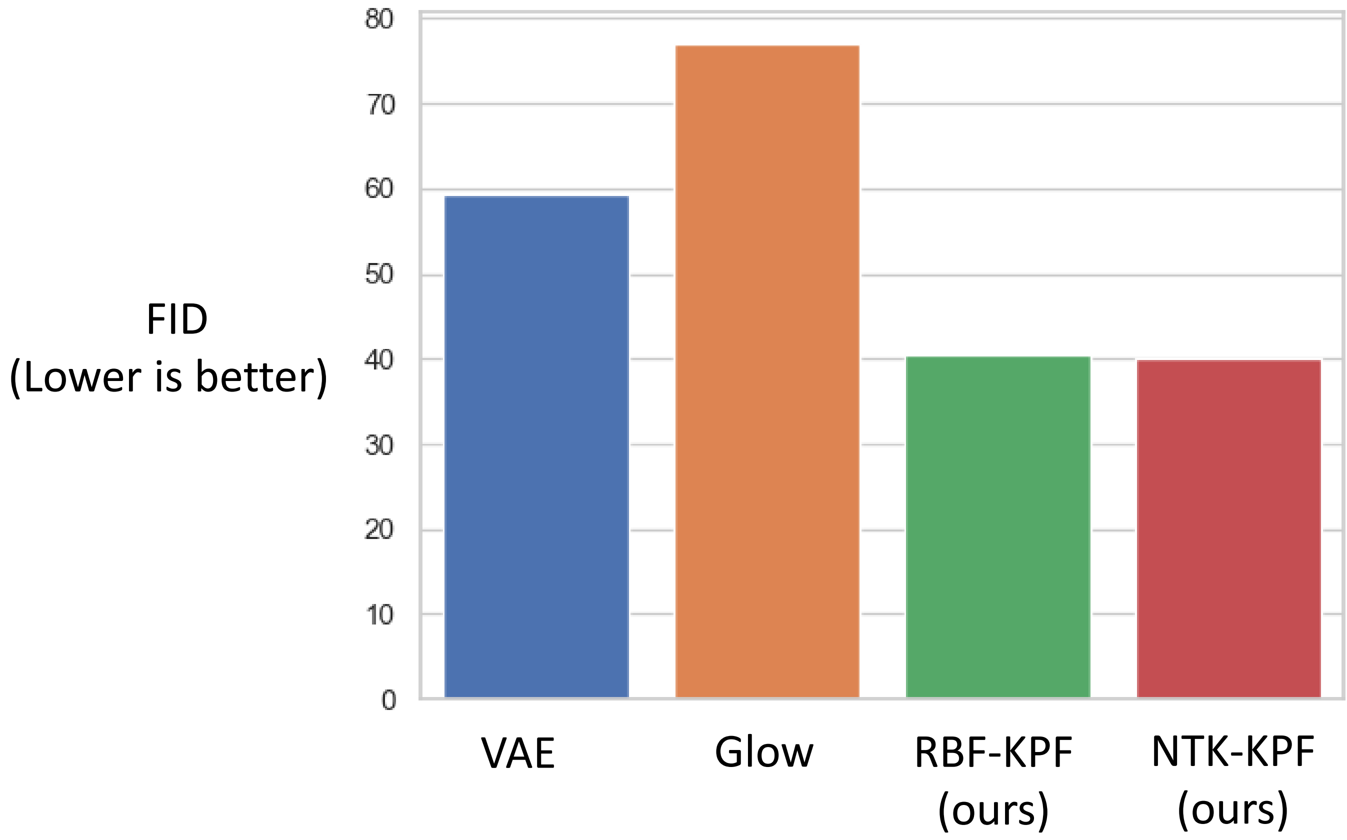
# Image generation

- kPF samples using NVAE [Vahdat & Kautz, 2020] latent space

Vahdat A., Kautz J. (2020). NVAE: A Deep Hierarchical Variational Autoencoder

# KPF in limited data regime

We took 100 samples (<1%) from CelebA and learned the latent distribution



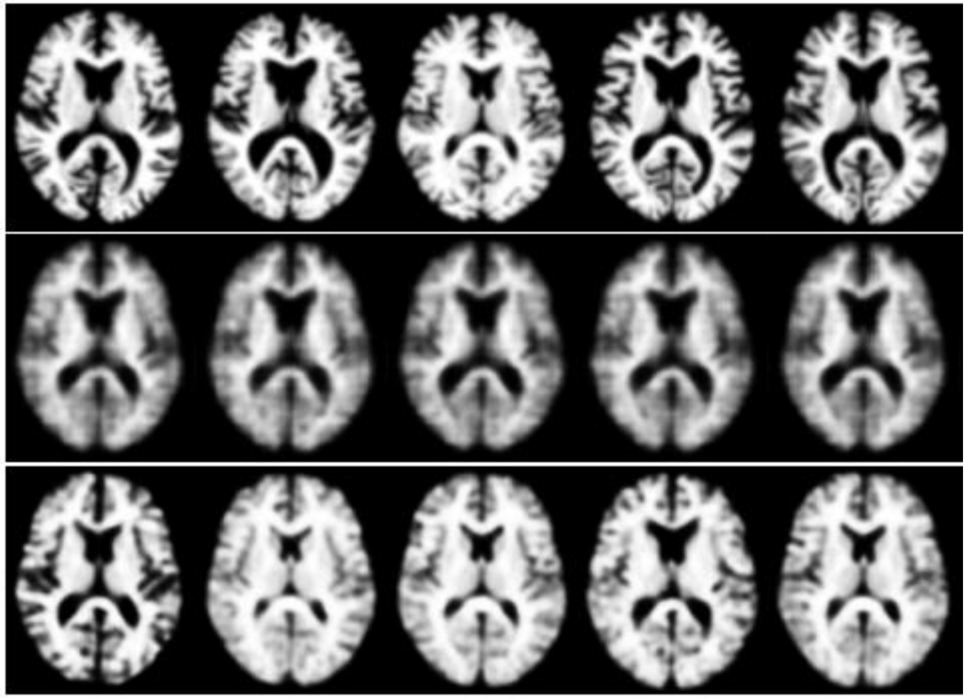
# KPF in limited data regime

We learned KPF from the high-resolution brain MR images of 474 patients

Data

VAE

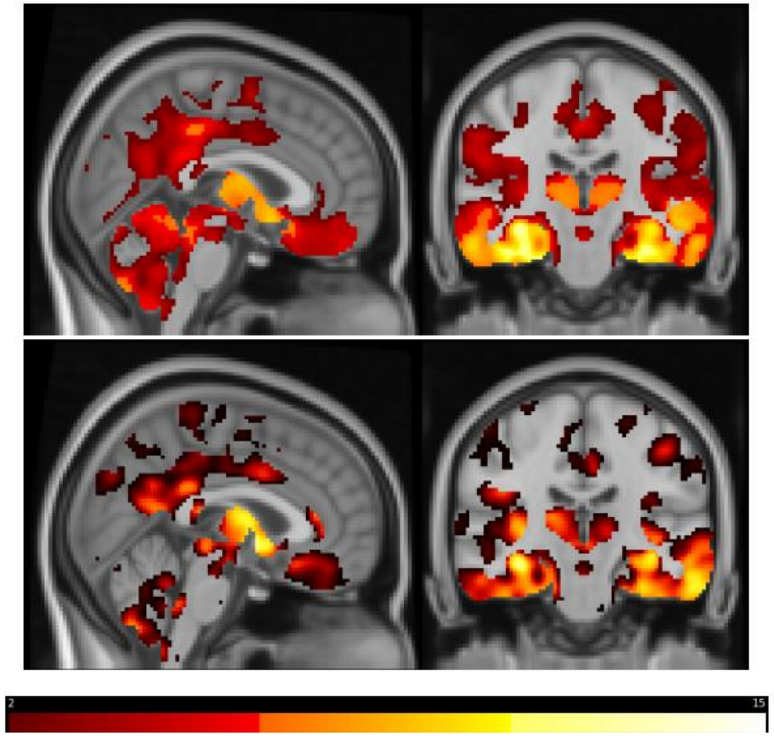
Ours



Statistically significant regions ($p < 0.05$)
control group vs. diseased group

Data

Ours



# Key Takeaways

- kPF is a **closed-form, linear** operator in RKHS that approximates the transfer operator of the forward operators in generative models
- Despite certain limitations (e.g. scalability, requirement of a smooth latent space), kPF compares well with existing decoder-based generative models
- In the low-data regime, kPF outperforms deep generative models in terms of both computational cost and sample quality

# Thank you!