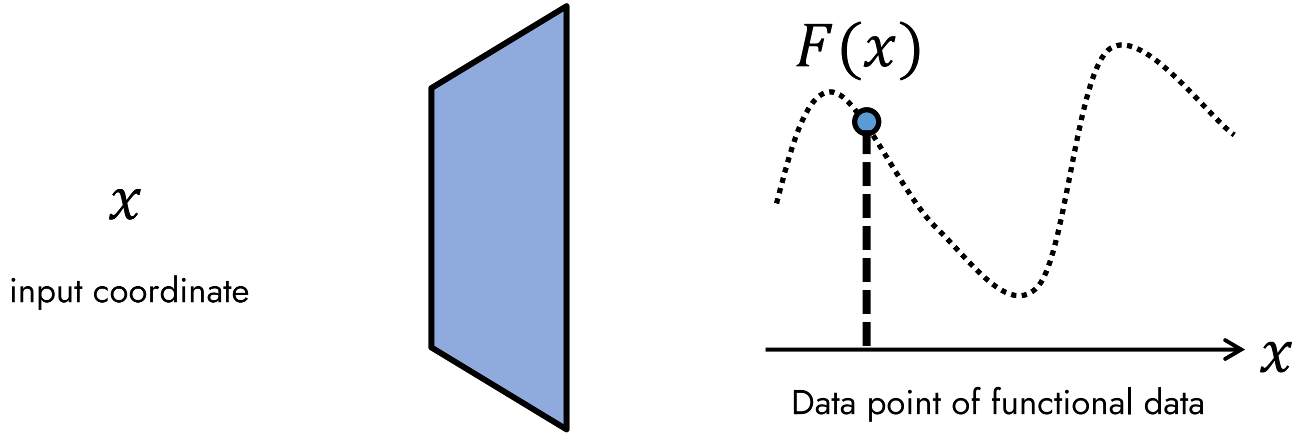
ICLR

Isometric Regularization for Manifolds of Functional Data

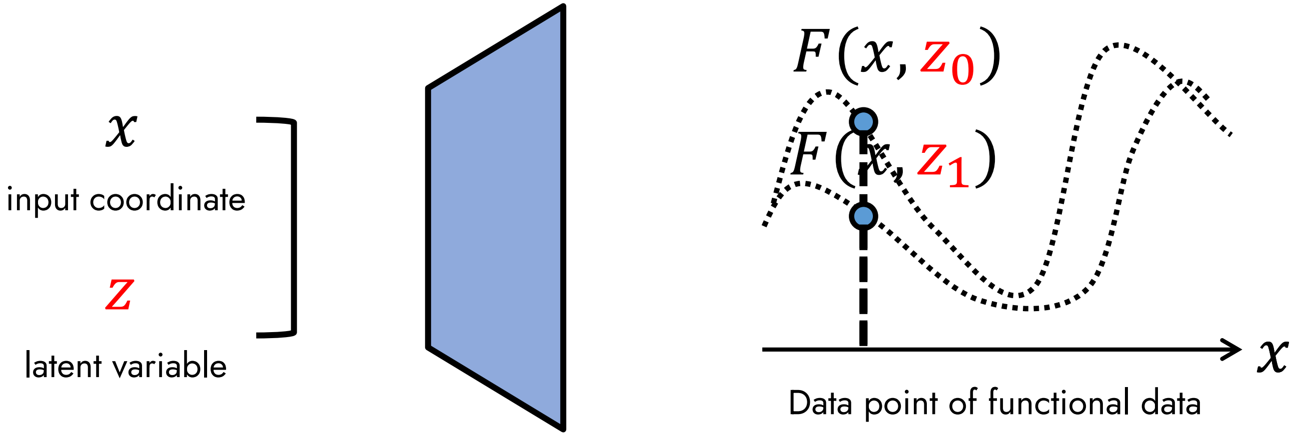
Hyeongjun Heo Seonghun Oh Jae Yong Lee Young Min Kim* Yonghyeon Lee*

*: co-corresponding author

Implicit Neural Representations



Latent Variable Implicit Neural Representations



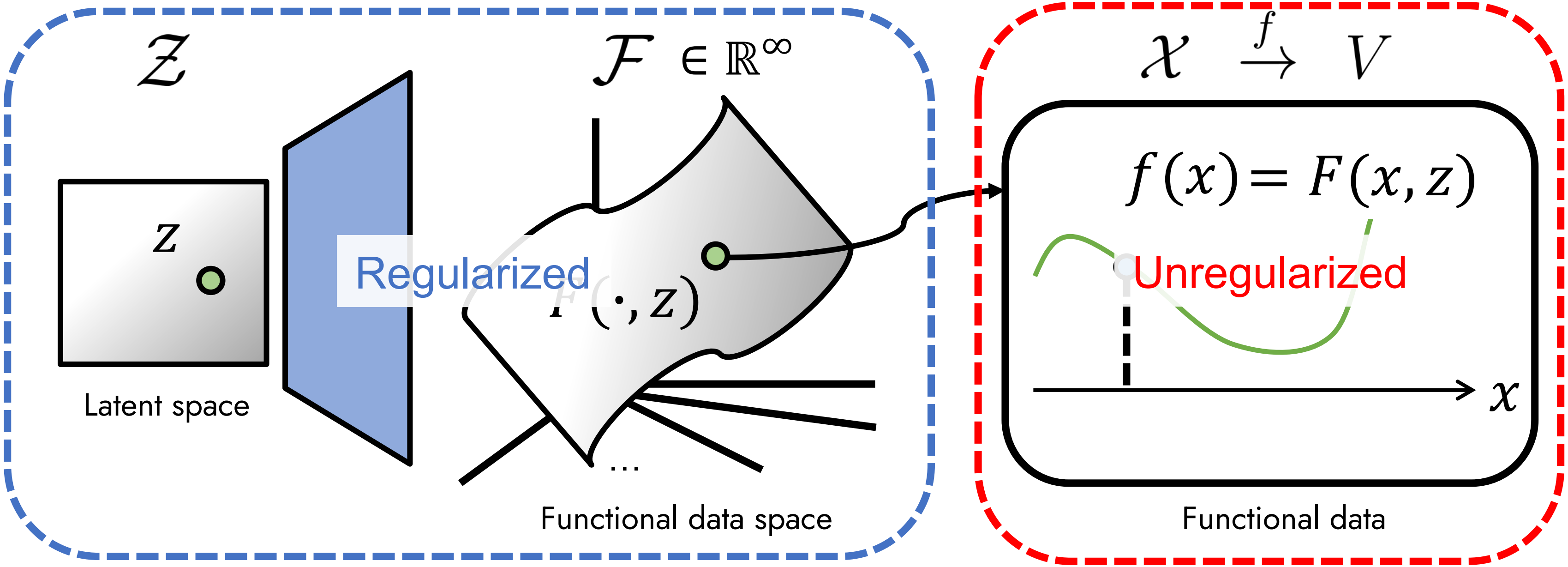
Criteria for Strong Performance

I. The network should accurately fit the data instances resulting in high-fidelity reconstructions.

II. Latent space should be well-behaved, ensuring small changes in the latent space lead to gradual and predictable changes in output.

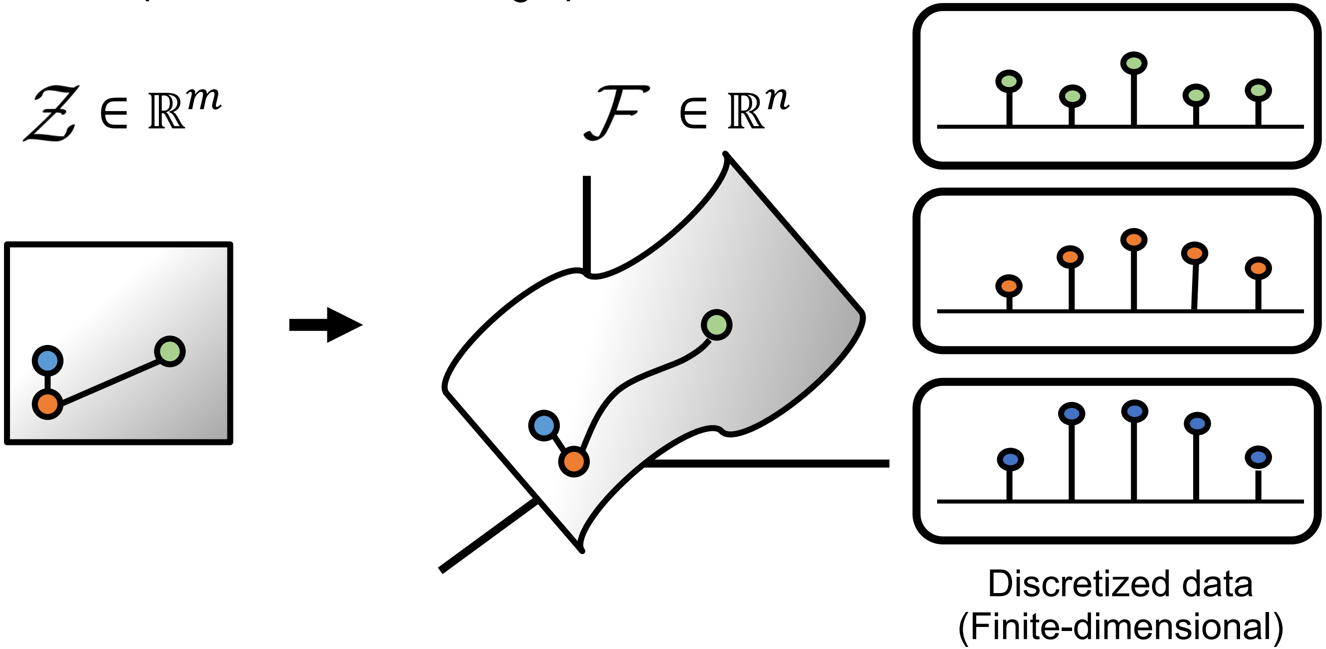
➔ We propose a balanced regularization method that produces well-behaved latent space with accurate data fitting.

Manifold of Functional Data



Isometric Regularization

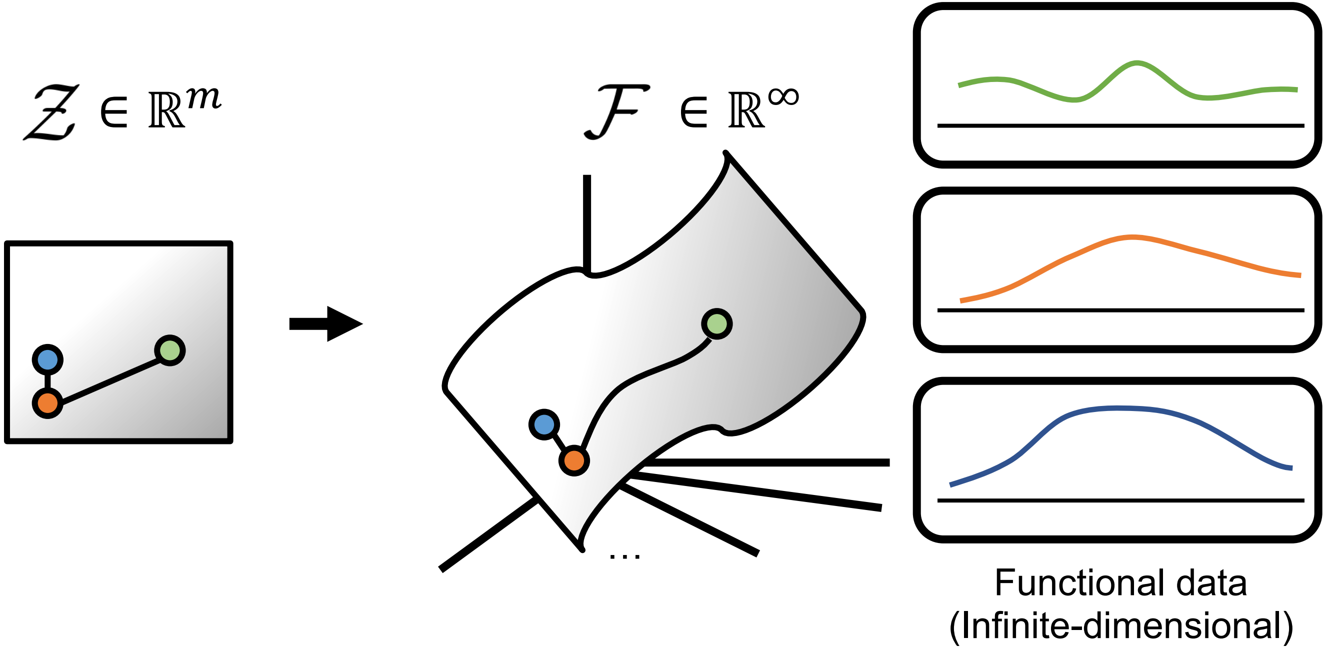
- Scaled Isometry[1]: geometry-preserving mapping from latent space to data space (relative distance, angle)



[1] Lee et al., Regularized Autoencoders for Isometric Representation Learning, ICLR (2022)

Isometric Regularization for Manifolds of Functional Data

- We adopt isometric regularization[1] developed for finite-dimensional data and extend it to "*infinite-dimensional*" *functional data*.



[1] Lee et al., Regularized Autoencoders for Isometric Representation Learning, ICLR (2022)

Isometric Regularization for Manifolds of Functional Data

- Relaxed Distortion Measure (how far is F from being a scaled isometry)

$$\mathcal{G}(F, P_Z) = \frac{\mathbb{E}_{z\sim P_Z}\left[\mathrm{Tr}\left(\left(\mathbb{E}_{x\sim p(x;F_z)}[J^T(x,z)J(x,z)]\right)^2\right)\right]}{\mathbb{E}_{z\sim P_Z}\left[\mathrm{Tr}\left(\mathbb{E}_{x\sim p(x;F_z)}[J^T(x,z)J(x,z)]\right)\right]^2}$$

- Final loss function

➔ Requires too much computation

$$\min \mathcal{L}_{original} + \lambda \mathcal{G}(F, P_Z)$$

J: Jacobian of the mapping
λ: regularization weight

Efficient Approximation for Faster Computation

- Approximate with Hutchinson trace estimator, no need to compute Jacobian

Algorithm 1: Efficient approximation of Eq. (9)

Precondition: input concatenation $(F : \mathbb{R}^{n+m} \rightarrow \mathbb{R}^{l})$
Input: latent codes $\{\mathbf{z}_0, \ldots, \mathbf{z}_N\}$ & input coordinate samples
$\{\{\mathbf{x}_0^{(0)}, \ldots, \mathbf{x}_0^{(K)}\}, \ldots, \{\mathbf{x}_N^{(0)}, \ldots, \mathbf{x}_N^{(K)}\}\}$
Output: Relaxed distortion measure $\mathcal{G}$

```
G_1, G_2 ← 0
Augment z with the modified mix-up data-augmentation
forall z_i in z do
    x_i ← {x_i^(0), ..., x_i^(K)}
    Sample vector v_i ~ N(0, I_{m×m})
    Expand v_i by repeating K times
    Augment vector v_i by concatenating [0⃗_{k×n}, v_i]
    Compute G = J(x_i, z_i) v_i with Jacobian-vector product
    G_1 ← G_1 + E_z[E_x[G^T G]]
    Compute D = G^T ∂F(x_i, z_i)/∂(x, z) with vector-Jacobian product
    Slice the index of D by taking the last m-th components
    G_2 ← G_2 + E_z[E_x[D]^T E_x[D]]
end
G ← G_1 / G_2
return G
```

Experimental Results

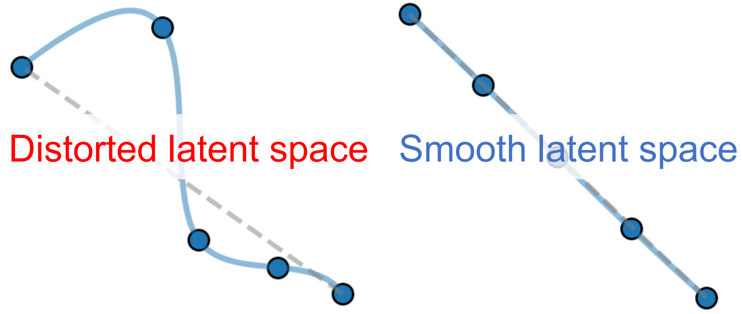
Signed Distance Function (SDFs)

Neural SDFs

- Toy example with five circles (r=0.1,...,0.5)

AD: Auto-decoder

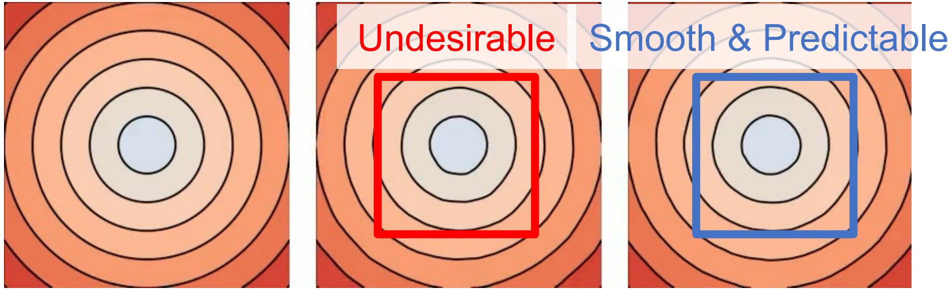
Latent Space



AD

IsoAD (Ours)

Interpolation



GT

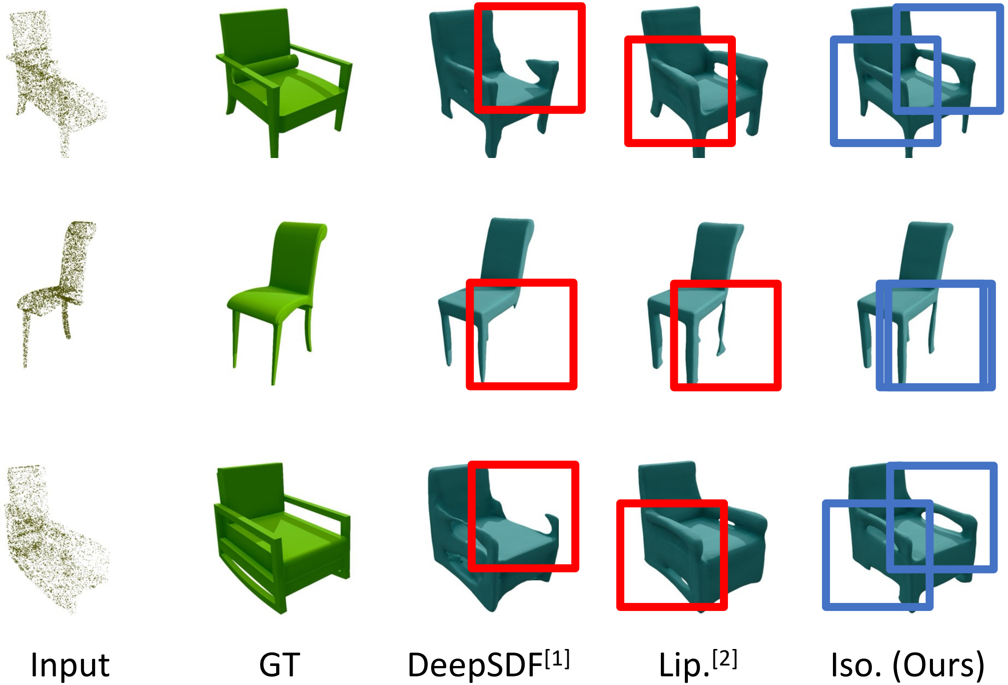
AD

IsoAD (Ours)

Neural SDFs

- ShapeNet dataset

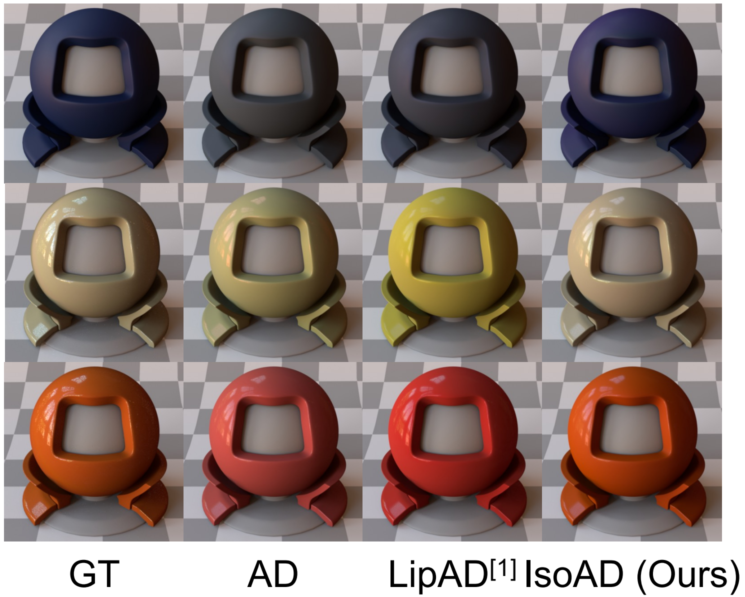
Lip.: Lipschitz Regularization



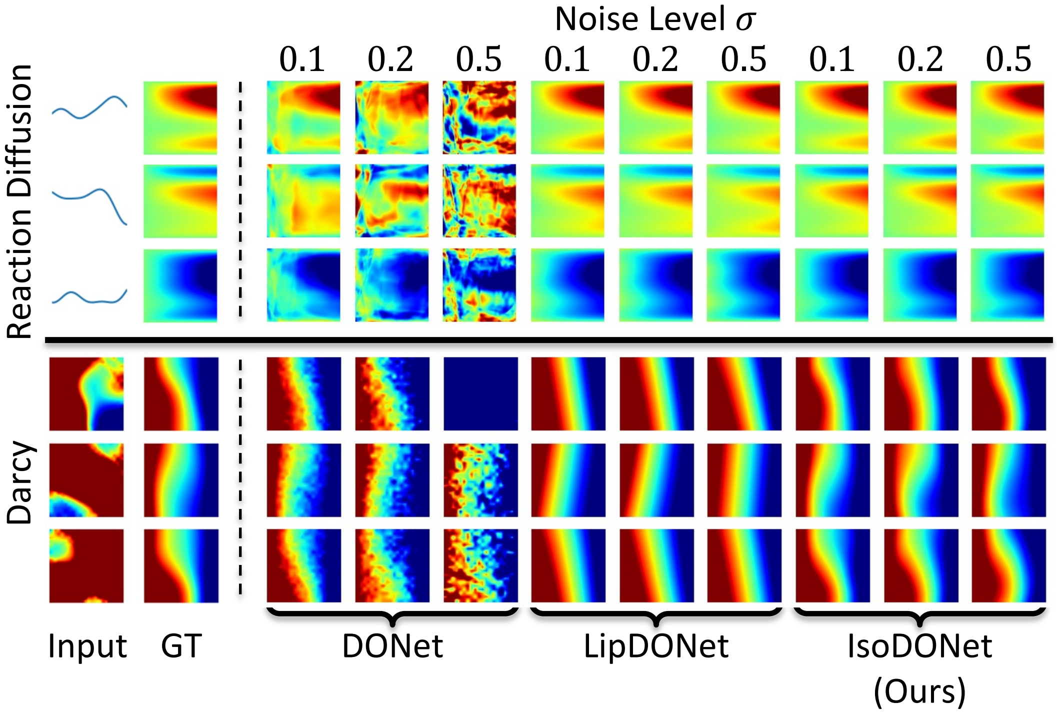
[1] Park et al., DeepSDF: Learning Continuous Signed Distance Functions for Shape Representation, CVPR (2019)
[2] Liu et al., Learning smooth neural functions via lipschitz regularization, ACM SIGGRAPH 2022 Conference Proceedings (2022)

Diverse Application Scenarios

- Neural Bidirectional Radiance Distribution Functions (BRDFs)



- Neural Operators



[1] Liu et al., Learning smooth neural functions via lipschitz regularization, ACM SIGGRAPH 2022 Conference Proceedings (2022)
[2] Lu et al., Learning nonlinear operators via DeepONet based on the universal approximation theorem of operators, Nature machine intelligence (2021)

Thank you!