Scalable Operator Spectral Decomposition with Neural Networks

Jongha (Jon) Ryu

MIT EECS / RLE

MIT JTL Urban Mobility Lab

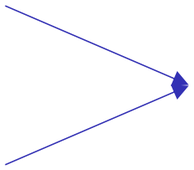
September 8, 2025

From Correlation to Linear Operators

- We wish to understand correlation between two (high-dim.) variables **x** and **y**
- **Classification**: Given an image, what's the label?
- **Regression**: Given a covariate, what's the (average) response?
- **Multimodal setting**: Can we capture relation between image and text?
- **Time series**: Given today's stock price, what'll be tomorrow's price?
- **Dynamical system**: Given a current state, what'll be the next state?

$p(\mathbf{x}, \mathbf{y})$

Physical operator



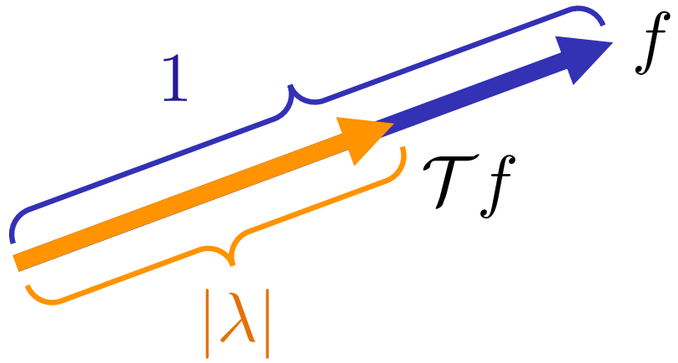
$\mathcal{T}$ (linear)

Eigenvalue Problems & Spectral Decomposition

$$(\mathcal{T} f)(\mathbf{x}) = \lambda f(\mathbf{x})$$

f: eigenfunction

λ: eigenvalue



For "symmetric" or normal $\mathcal{T}$:

Top-k eigenvalue decomposition (EVD)

$$\mathcal{T} \approx \sum_{n=1}^{k} \lambda_n \psi_n \otimes \psi_n$$

$$\langle \psi_i, \psi_j \rangle = \delta_{ij}$$

For compact $\mathcal{T}$:

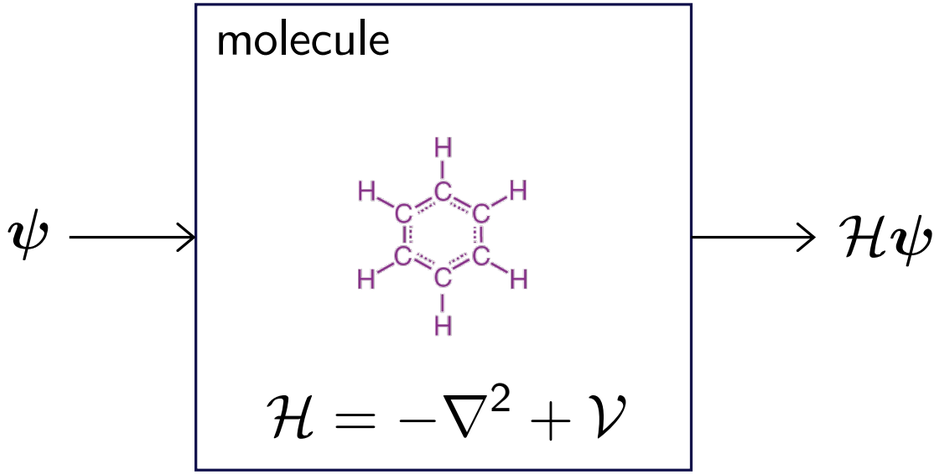
Top-k singular value decomposition (SVD)

$$\mathcal{T} \approx \sum_{n=1}^{k} \sigma_n \phi_n \otimes \psi_n$$

$$\langle \phi_i, \phi_j \rangle = \delta_{ij} \quad \langle \psi_i, \psi_j \rangle = \delta_{ij}$$

Motivating Example 1. **Molecule Simulation**

Q. Can we predict the property of a molecule from simulation?



- The most microscopic scale is governed by the principle of quantum mechanics

Schrödinger's equation $\mathcal{H}\psi = \lambda\psi$

  - With N electrons, $\mathcal{X} = \mathbb{R}^{3N}$, $\psi \in \mathbb{R}^{\mathcal{X}}$
  - Hermitian → eigenvalues / functions are real
  - Physical meaning: $\lambda_1 \leq \lambda_2 \leq \ldots$

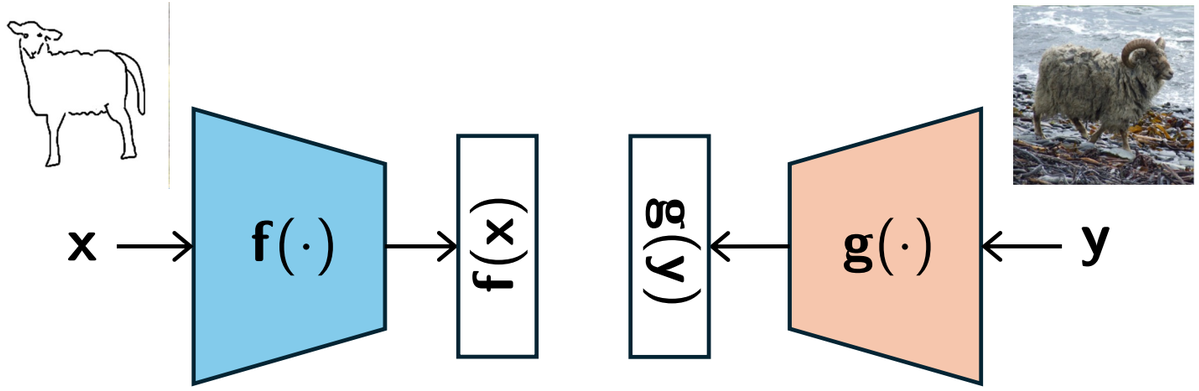
    ψ_i is the i-th stable configuration of electrons

If we can better perform **EVD** of Hamiltonians, we can **simulate molecules more efficiently**

Motivating Example 2. **Representation Learning**

- Success of AI/ML relies on **learning good representation** of complex data
  - However, often learned end-to-end,

    lacking interpretation & structure
- SVD can provide a **systematic** view
- Consider two modalities $(\mathbf{x}, \mathbf{y}) \sim p(\mathbf{x}, \mathbf{y})$



$$k(\mathbf{x}, \mathbf{y}) \triangleq \frac{p(\mathbf{x}, \mathbf{y})}{p(\mathbf{x})p(\mathbf{y})} \longleftrightarrow (\mathcal{K}g)(\mathbf{x}) \triangleq \mathbb{E}[g(\mathbf{y})|\mathbf{x}]$$

kernel — operator

$$= \sum_{i=1}^{\infty} \sigma_i \phi_i(\mathbf{x}) \psi_i(\mathbf{y}), \quad \sigma_1 \geq \sigma_2 \geq \ldots \geq 0$$

$$\mathbf{x} \mapsto \begin{bmatrix} \sqrt{\sigma_1}\phi_1(\mathbf{x}) \\ \vdots \\ \sqrt{\sigma_L}\phi_L(\mathbf{x}) \end{bmatrix} \quad \text{and} \quad \mathbf{y} \mapsto \begin{bmatrix} \sqrt{\sigma_1}\psi_1(\mathbf{y}) \\ \vdots \\ \sqrt{\sigma_L}\psi_L(\mathbf{y}) \end{bmatrix}$$

optimal & **structured** L-dim. representation!

If we can efficiently perform **SVD**,
we can learn
a structured representation!

Spectral Decomposition is a Fundamental Building Block

Problem	Operator	Application
Quantum chemistry	Hamiltonian	Quantum many-body system
Density functional theory	Kohn-Sham equation	Electronic structure
Nonlinear dynamical systems	Koopman operator	Control, molecular dynamics
Principal component analysis	Covariance matrix / operator	Representation learning, dependence analysis, downstream tasks (classification, regression), ...
Canonical component analysis	Conditional expectation operator	
Graphical data analysis	Graph Laplacian operator	
Representation learning	Conditional expectation operator	

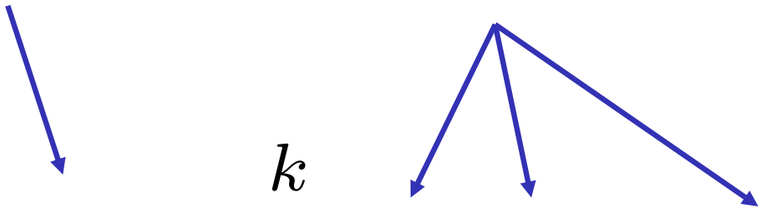
Physics and Engineering Simulation

Statistics and Machine Learning

Efficient top-k EVD/SVD can tackle various problems!

Problem: Top-k Operator Spectral Decomposition

Given how can we find (compute / estimate)



$$\mathcal{T} \approx \sum_{n=1}^{k} \sigma_n \phi_n \otimes \psi_n$$

$$\sigma_1 \geq \sigma_2 \geq \ldots \geq 0$$

$$\langle \phi_i, \phi_j \rangle = \delta_{ij}$$

$$\langle \psi_i, \psi_j \rangle = \delta_{ij}$$

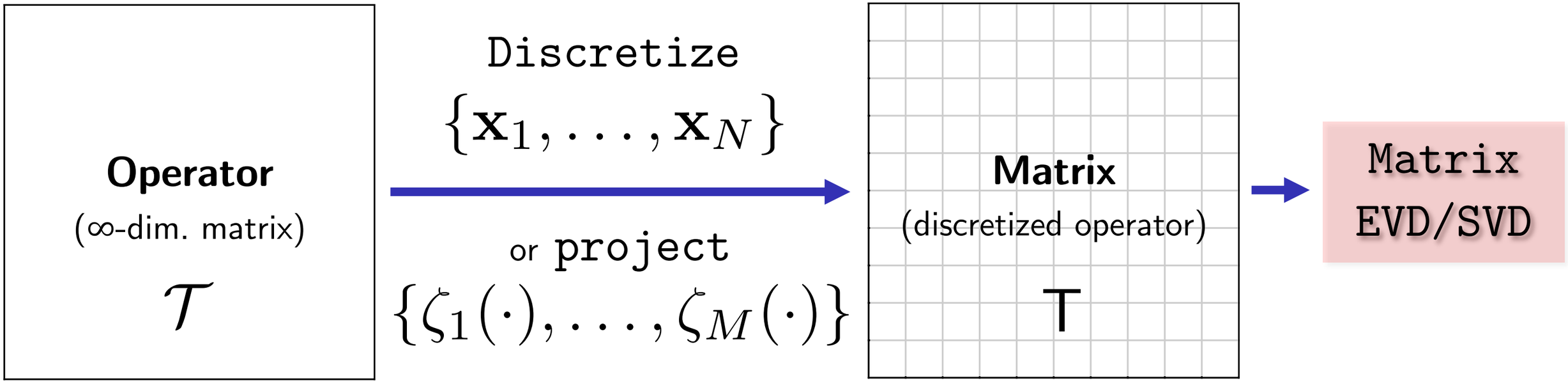
under this constraint?

Can be viewed as **structured** operator learning

Classical Approaches

Discretization-based methods
(e.g., finite element method)

PCA / CCA



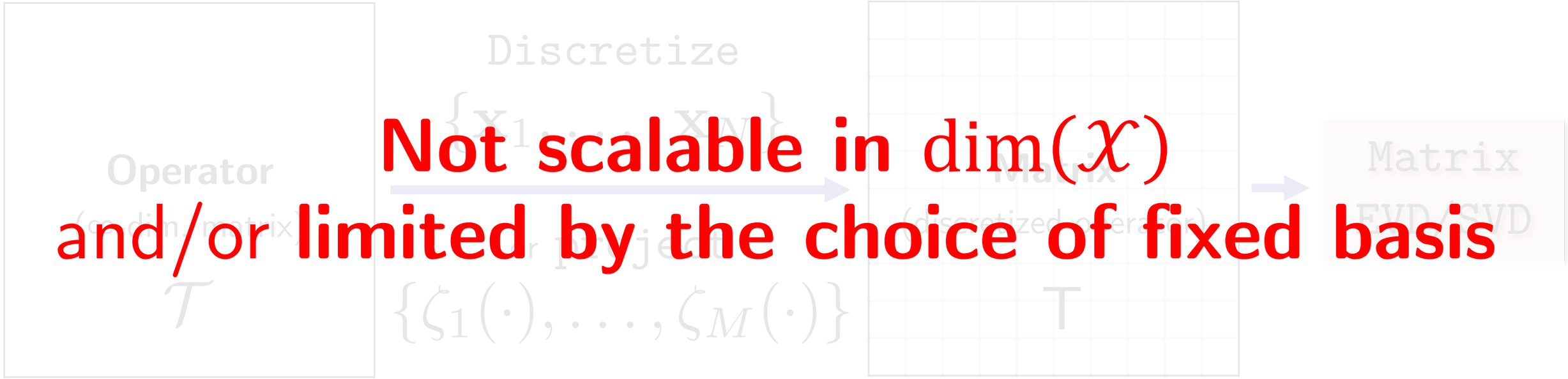
Galerkin-type projection methods
(e.g., extended DMD)

Kernel PCA / CCA

Classical Approaches

Discretization-based methods
(e.g., finite element method)

PCA / CCA

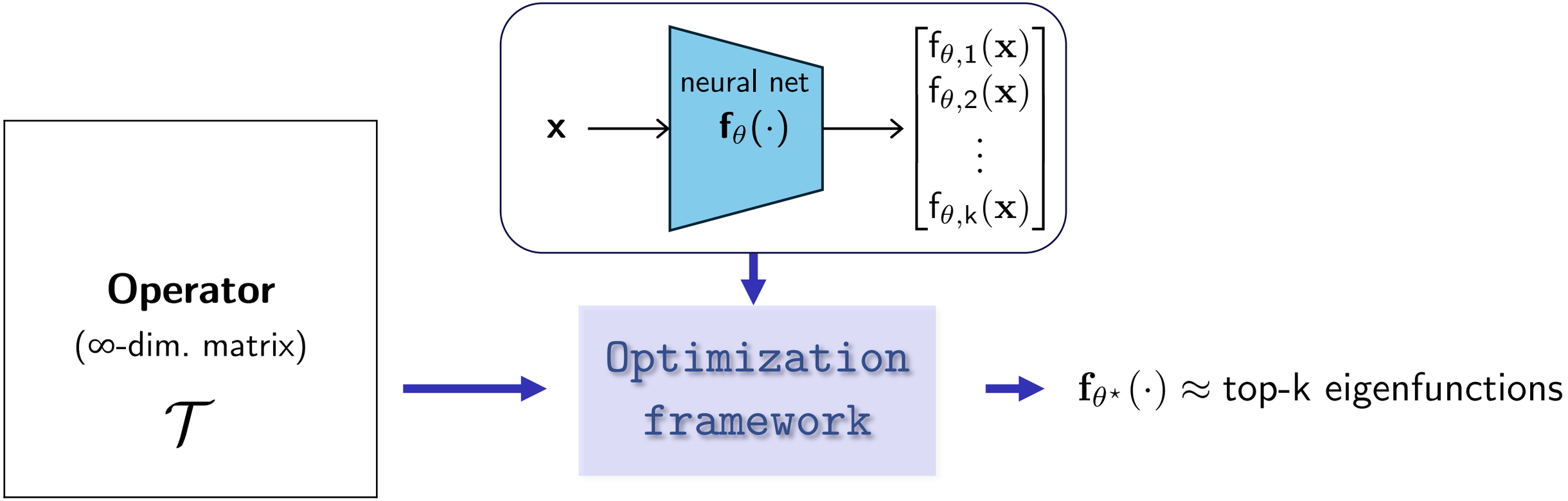


Not scalable in $\dim(\mathcal{X})$
and/or limited by the choice of fixed basis

Galerkin-type projection methods
(e.g., extended DMD)

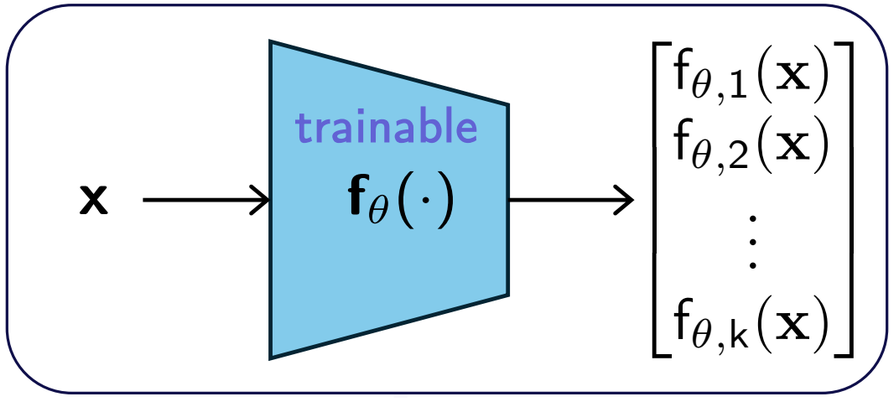
Kernel PCA / CCA

Modern Approach



Given (1) a "sufficiently expressive" neural network and (2) a "good" optimization framework, this approach can be much more scalable

A Unified View on {Classical, Modern} Approaches



The discriminative feature:
the parametric form of the trial function!

Discretization-based approach	Projection-based approach	Fully parametric approach
given N data points $\{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$	given fixed basis functions $\{\zeta_1, \ldots, \zeta_M\}$	
$f_{\theta,k}(\mathbf{x}) = \sum_{i=1}^{N} v_{k,i}\delta(\mathbf{x} - \mathbf{x}_i)$	$f_{\theta,k}(\mathbf{x}) = \mathbf{a}_k^\intercal \zeta_{1:M}(\mathbf{x})$	$f_{\theta,k}(\mathbf{x}) = (\text{neural network})$

Which Optimization Framework Should We Use?

- Researchers in different domains have proposed various (often similar) proposals under different names...

Computational Statistics
(Streaming PCA/CCA)
Oja's algorithm (1982)
Krasulina's algorithm (1969)
Sanger's algorithm (1989)
Noisy power method (2013)
EigenGame (2020)

Machine Learning / Deep Learning
SpIN (2019)
NeuralEF (2022)
NeuralEigenmaps (2022)

Reinforcement Learning
Spectral graph drawing (2003)
Graph drawing objective (2019)
Generalized GDO (2021)
ALLO (2024)

Graph Learning
SpectralNet (2018)

Computational Physics/Chemistry
Orbital minimization methods (1993)
Variational Quantum Monte Carlo (2001)
Natural excited states (2024)

Nonlinear Dynamical Systems
VAMPnet (2018)
DPNet (2024)

Numerical Linear Algebra
Trace-penalty minimization method (2015)
Symmetric low-rank product method (2015)
Triangularized orthogonalization-free method (2020)

Which Optimization Framework Should We Use?

- Researchers in different domains have proposed various (often similar) proposals under different names...

Can be categorized based on:
(0) Problem of interest/dimensionality/parametric form;
(1) **Variational optimization framework**;
   a. How to handle **orthogonality;**
   b. Whether to find **ordered outputs**

Dominant Variational Framework

- Most existing frameworks rely on:

$$\max_{\mathbf{f}} \frac{\mathbf{f}^\intercal \mathsf{T} \mathbf{f}}{\mathbf{f}^\intercal \mathbf{f}}$$

$$\max_{\mathbf{f}_1,\ldots,\mathbf{f}_k} \sum_{i=1}^{k} \mathbf{f}_i^\intercal \mathsf{T} \mathbf{f}_i \quad \text{s.t.} \quad \mathbf{f}_i^\intercal \mathbf{f}_j = \delta_{ij}$$

$$\max_{\{(\mathbf{u}_i,\mathbf{v}_i)\}_{i=1}^k} \sum_{i=1}^{k} (\mathbf{u}_i^\intercal \mathsf{T} \mathbf{v}_i)^r \quad \text{s.t.} \quad \mathbf{u}_i^\intercal \mathbf{u}_j = \delta_{ij}, \; \mathbf{v}_i^\intercal \mathbf{v}_j = \delta_{ij}$$

  - Rayleigh quotient maximization
  - Optimal solution = Top-k eigen/singular subspaces (up to rotation)

- Constrained optimization:
  require regularization, → require hyperparameter tuning
  biased gradient, → bad for large-scale optimization
  numerically unstable operations → unstable training

- Learning with order is often done in a complicated manner

Our Proposal: Nested Low-Rank Approximation

- **Variational principle:** Low-rank approximation (LoRA)

$$\min_{\{(\mathbf{f}_i,\mathbf{g}_i)\}_{i=1}^k} \left\| \mathsf{T} - \sum_{i=1}^{k} \mathbf{f}_i \mathbf{g}_i^\intercal \right\|_{\mathrm{F}}^2$$

  - Optimal solution = Top-k singular subspaces (up to rotation)
  - Unconstrained optimization, **no regularization**, **unbiased gradient estimate**
- Nesting: Can learn **ordered** singular-functions **w/o explicit orthogonalization**
  - **Idea**: Learn rank-i approximation for all $i = 1, \ldots, k$ simultaneously
- NeuralSVD = **LoRA** + **nesting** + neural nets
- This is **a versatile framework** in computational simulation as well as ML/AI

(1) Learning Subspace via LoRA

$$\min_{\{(\mathbf{f}_i,\mathbf{g}_i)\}_{i=1}^k} \left\| \mathsf{T} - \sum_{i=1}^k \mathbf{f}_i \mathbf{g}_i^\intercal \right\|_{\mathrm{F}}^2$$

Characterizes the top-k singular subspaces (Eckart-Young-Mirsky)

$$\sum_{i=1}^k \mathbf{f}_i^\star (\mathbf{g}_i^\star)^\intercal = (\text{best rank-}k \text{ approx. of } \mathsf{T}) = \sum_{i=1}^k \sigma_i \boldsymbol{\phi}_i \boldsymbol{\psi}_i^\intercal$$

$$\min_{\{\mathbf{f}_i,\mathbf{g}_i\}_{i=1}^k} -2 \sum_{i=1}^k \mathbf{g}_i^\intercal \mathsf{T} \mathbf{f}_i + \sum_{i=1}^K \sum_{j=1}^k \mathbf{f}_i^\intercal \mathbf{f}_j \mathbf{g}_i^\intercal \mathbf{g}_j$$

Can be estimated in an unbiased manner from samples

$$\mathbf{g}^\intercal \mathsf{T} \mathbf{f} = \int g(\mathbf{y})(\mathcal{T} f)(\mathbf{y}) \nu(d\mathbf{y}) \qquad \mathbf{f}_i^\intercal \mathbf{f}_j = \int f_i(\mathbf{x}) f_j(\mathbf{x}) \mu(d\mathbf{x})$$

$$\approx \frac{1}{B} \sum_{n=1}^B g(\mathbf{y}_n)(\mathcal{T} f)(\mathbf{y}_n) \qquad \approx \frac{1}{B} \sum_{n=1}^B f_i(\mathbf{x}_n) f_j(\mathbf{x}_n)$$

(2) Learning Order via Nesting

- **High-level idea**: perform rank ℓ-approximation for all $\ell = 1, \ldots, k$ simultaneously
- Let

$$\mathcal{L}_{\text{LoRA}}(\mathbf{f}_{1:\ell}, \mathbf{g}_{1:\ell}) \triangleq -2 \sum_{i=1}^{\ell} \mathbf{g}_i^\mathsf{T} \mathsf{T} \mathbf{f}_i + \sum_{i=1}^{\ell} \sum_{j=1}^{\ell} \mathbf{f}_i^\mathsf{T} \mathbf{f}_j \mathbf{g}_i^\mathsf{T} \mathbf{g}_j$$

- **Nesting** aims to achieve:

$$\begin{aligned}
(\mathbf{f}_1, \mathbf{g}_1) &\in \arg\min_{\mathbf{f}_1, \mathbf{g}_1} \mathcal{L}_{\text{LoRA}}(\mathbf{f}_1, \mathbf{g}_1) \\
(\mathbf{f}_{1:2}, \mathbf{g}_{1:2}) &\in \arg\min_{\mathbf{f}_{1:2}, \mathbf{g}_{1:2}} \mathcal{L}_{\text{LoRA}}(\mathbf{f}_{1:2}, \mathbf{g}_{1:2}) \\
&\vdots \\
(\mathbf{f}_{1:k}, \mathbf{g}_{1:k}) &\in \arg\min_{\mathbf{f}_{1:k}, \mathbf{g}_{1:k}} \mathcal{L}_{\text{LoRA}}(\mathbf{f}_{1:k}, \mathbf{g}_{1:k})
\end{aligned}$$

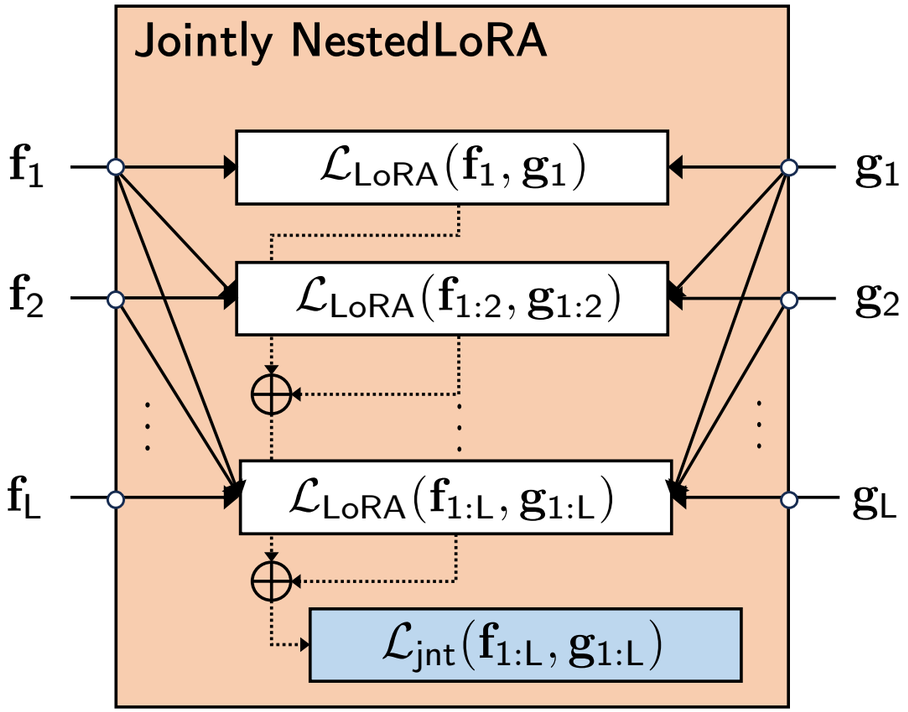
$$\begin{aligned}
\mathbf{f}_1 \mathbf{g}_1^\mathsf{T} &= \sigma_1 \boldsymbol{\phi}_1 \boldsymbol{\psi}_1^\mathsf{T} \\
\mathbf{f}_1 \mathbf{g}_1^\mathsf{T} + \mathbf{f}_2 \mathbf{g}_2^\mathsf{T} &= \sigma_1 \boldsymbol{\phi}_1 \boldsymbol{\psi}_1^\mathsf{T} + \sigma_2 \boldsymbol{\phi}_2 \boldsymbol{\psi}_2^\mathsf{T} \\
&\vdots \\
\mathbf{f}_1 \mathbf{g}_1^\mathsf{T} + \ldots + \mathbf{f}_k \mathbf{g}_k^\mathsf{T} &= \sigma_1 \boldsymbol{\phi}_1 \boldsymbol{\psi}_1^\mathsf{T} + \ldots + \sigma_k \boldsymbol{\phi}_k \boldsymbol{\psi}_k^\mathsf{T}
\end{aligned}$$

$$\Rightarrow \mathbf{f}_i \mathbf{g}_i^\mathsf{T} = \sigma_i \boldsymbol{\phi}_i \boldsymbol{\psi}_i^\mathsf{T} \quad \forall i \in [k]$$

- Orthogonality only implicitly enforced

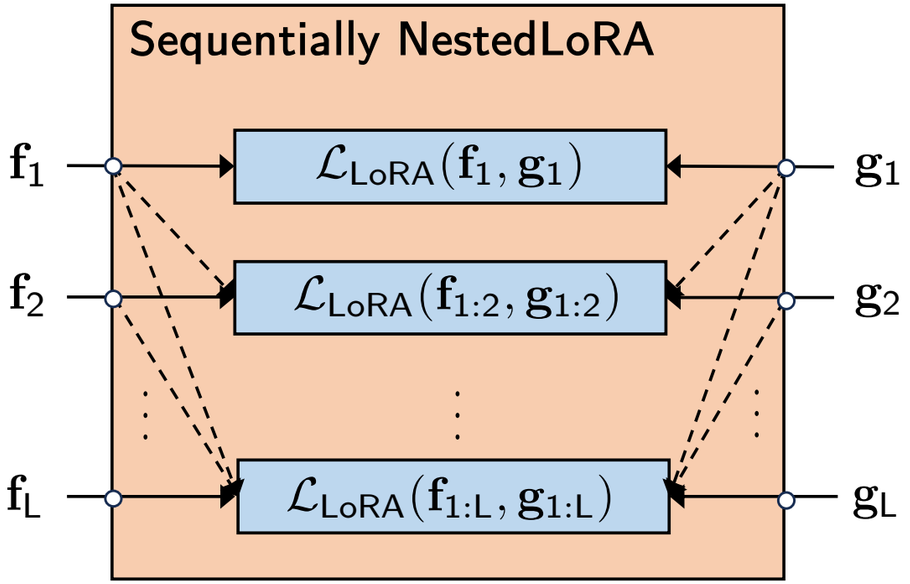
(2) Learning Order via Nesting: Implementation

- Joint nesting



- when using a single neural network

- Sequential nesting

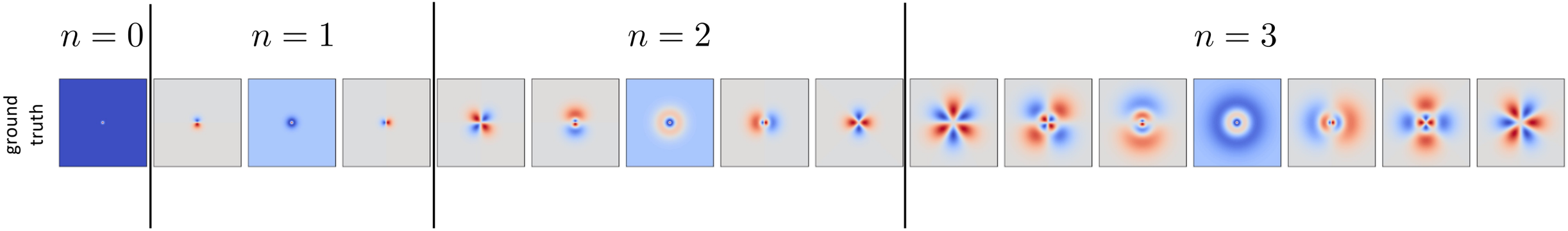


- when using multiple neural networks

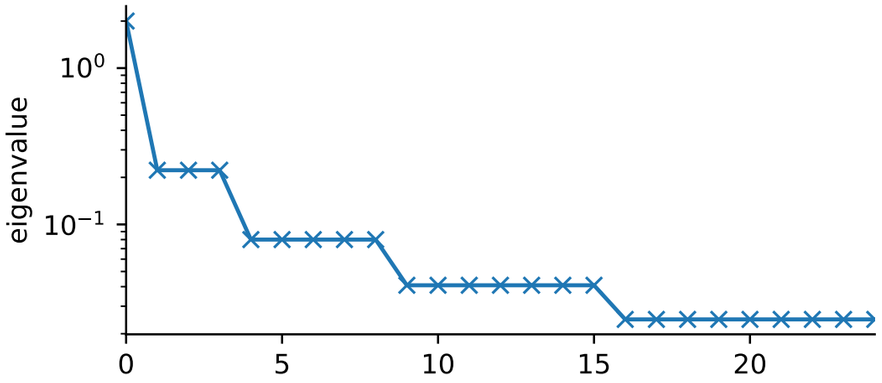
Application 1. Solving **Linear PDEs**

- Time-independent **Schrödinger equation** $\mathcal{H}\psi(\mathbf{x}) = \lambda\psi(\mathbf{x})$
- 2D-confined hydrogen $\mathcal{V}(\mathbf{x}) = -\dfrac{1}{\|\mathbf{x}\|_2},\ \mathbf{x} \in \mathbb{R}^2 \times \{0\}$

$$\mathcal{H} = \underbrace{-\nabla^2}_{\text{kinetic}} + \underbrace{\mathcal{V}(\mathbf{x})}_{\text{potential}}$$



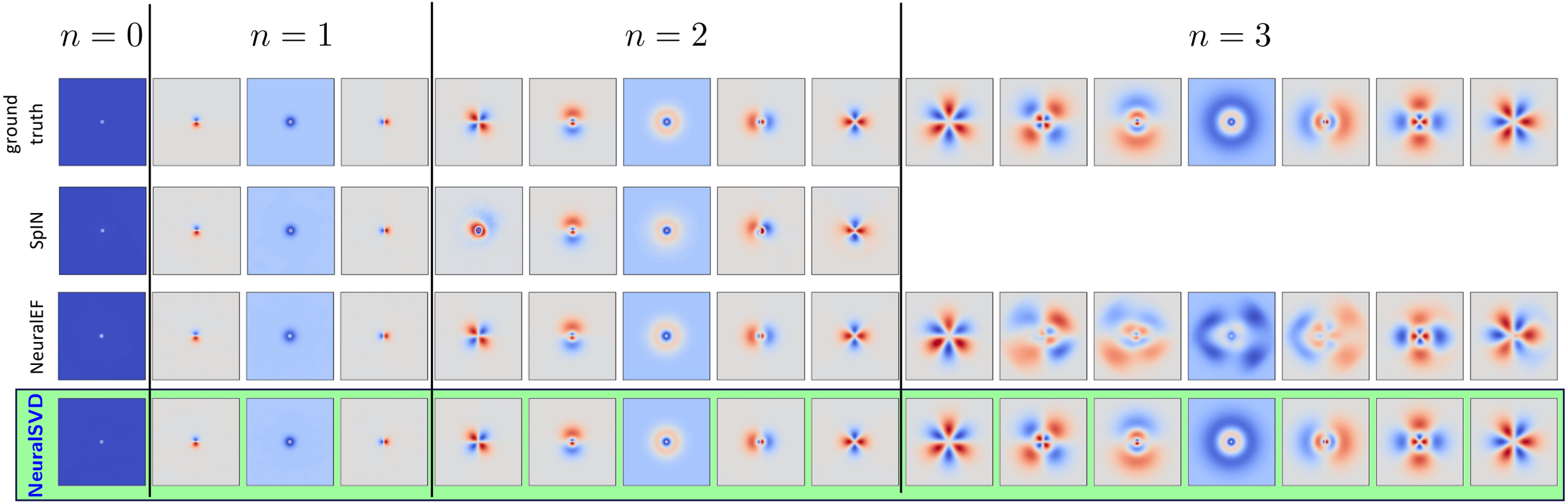
Eigenfunction: stationary configuration
Eigenvalue: energy



Application 1. Solving **Linear PDEs**

- Time-independent **Schrödinger equation** $\mathcal{H}\psi(\mathbf{x}) = \lambda\psi(\mathbf{x})$
- 2D-confined hydrogen $\mathcal{V}(\mathbf{x}) = -\dfrac{1}{\|\mathbf{x}\|_2},\ \mathbf{x} \in \mathbb{R}^2 \times \{0\}$

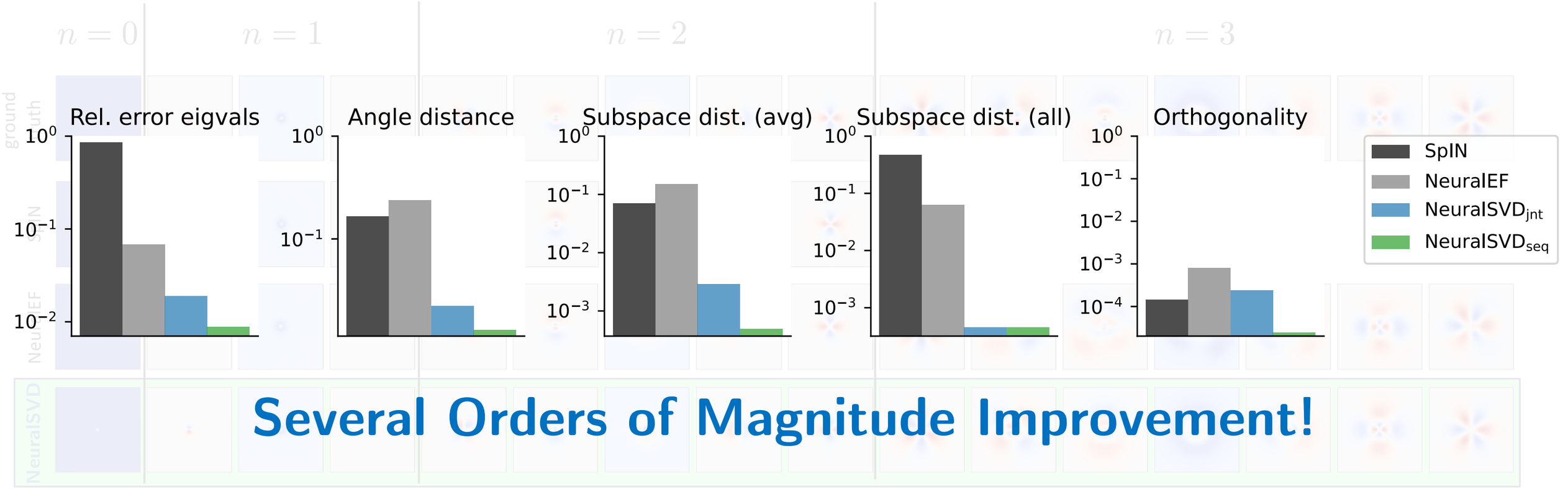
$$\mathcal{H} = \underbrace{-\nabla^2}_{\text{kinetic}} + \underbrace{\mathcal{V}(\mathbf{x})}_{\text{potential}}$$



Application 1. Solving **Linear PDEs**

- Time-independent **Schrödinger equation** $\mathcal{H}\psi(\mathbf{x}) = \lambda\psi(\mathbf{x})$
- 2D-confined hydrogen $\mathcal{V}(\mathbf{x}) = -\dfrac{1}{\|\mathbf{x}\|_2},\ \mathbf{x} \in \mathbb{R}^2 \times \{0\}$

$$\mathcal{H} = \underbrace{-\nabla^2}_{\text{kinetic}} + \underbrace{\mathcal{V}(\mathbf{x})}_{\text{potential}}$$



Several Orders of Magnitude Improvement!

Application 1. Solving **Linear PDEs**

Scalability of Parametric Spectral Decomposition

Even in this simple 2D example,
the parametric approach
demonstrates
favorable scalability!

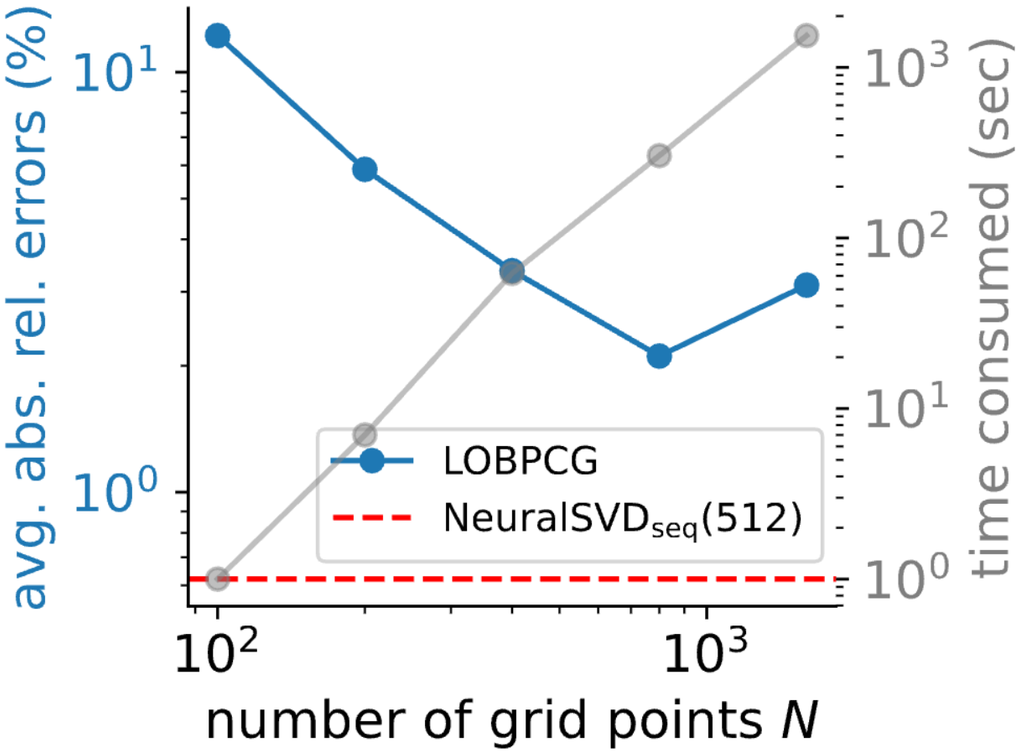


Figure 6: Performance of LOBPCG 2D hydrogen experiment.

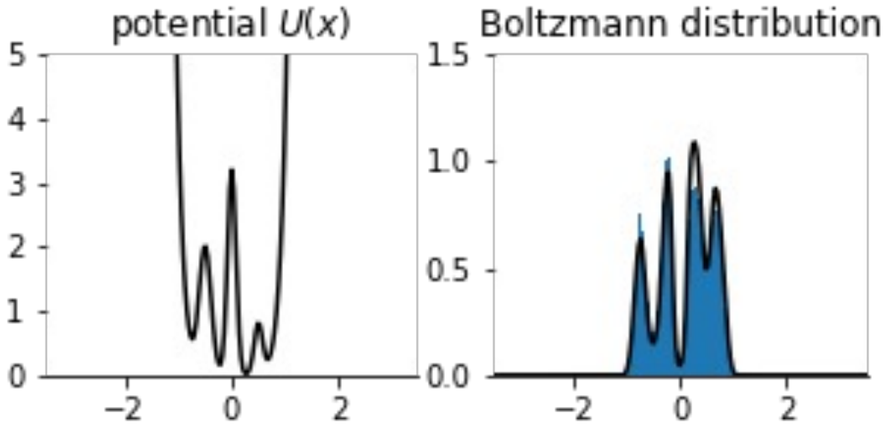
Application 2. Analyzing **Dynamical Systems**

(Overdamped) Langevin dynamics

$$dX_t = -\frac{U'(x_t)}{\gamma}dt + \sqrt{\frac{2}{\beta\gamma}}dW_t$$

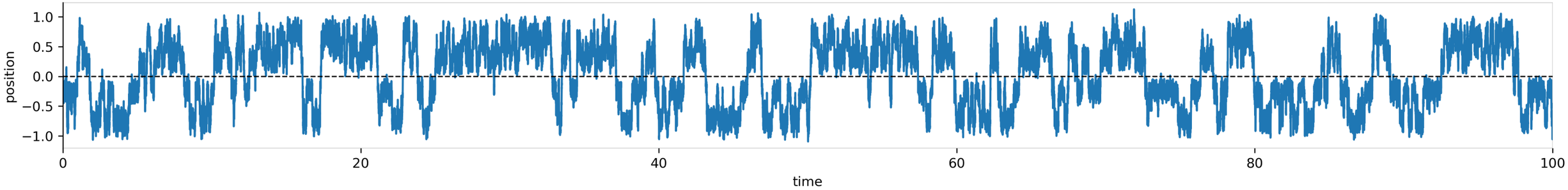
Infinitesimal generator

$$(\mathcal{L}f)(x) = -\frac{U'(x_t)}{\gamma}f'(x) + \frac{1}{\beta\gamma}f''(x)$$



$\mathcal{L}$ is symmetric (time-reversal process)

Eigenfunction: dynamics-invariant distribution / **Eigenvalue**: decay rate



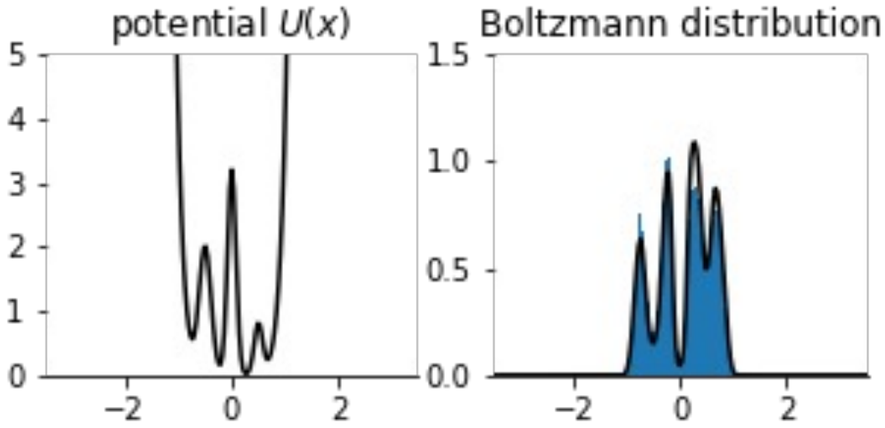
Application 2. Analyzing **Dynamical Systems**

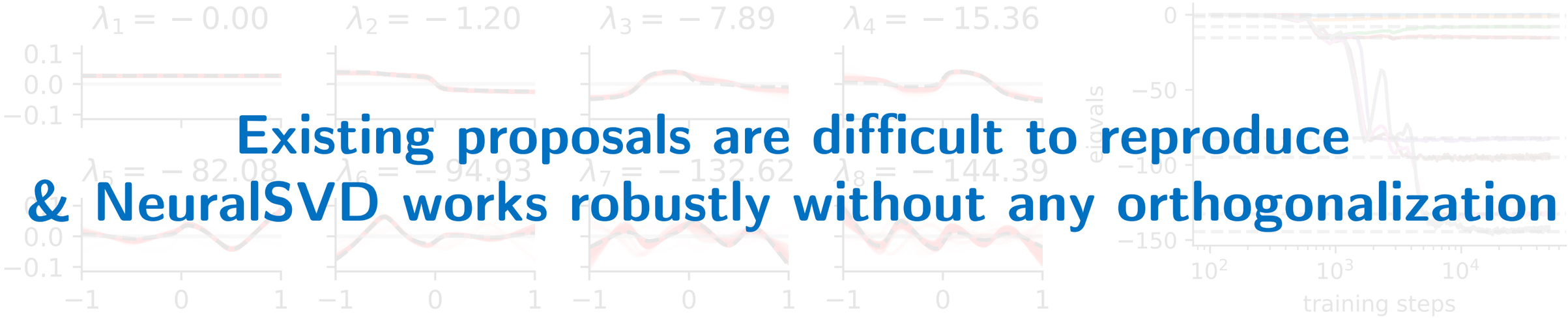
(Overdamped) Langevin dynamics

$$dX_t = -\frac{U'(x_t)}{\gamma}dt + \sqrt{\frac{2}{\beta\gamma}}dW_t$$

Infinitesimal generator

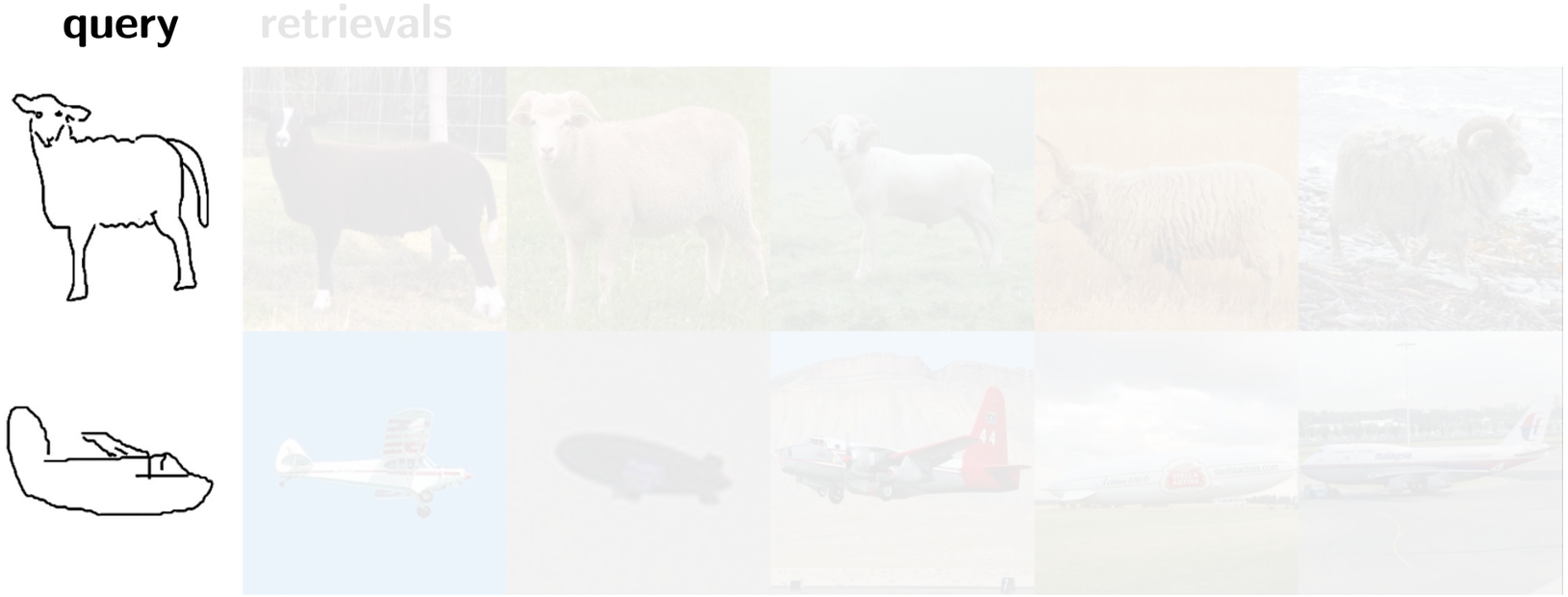
$$(\mathcal{L}f)(x) = -\frac{U'(x_t)}{\gamma}f'(x) + \frac{1}{\beta\gamma}f''(x)$$





Existing proposals are difficult to reproduce
& NeuralSVD works robustly without any orthogonalization

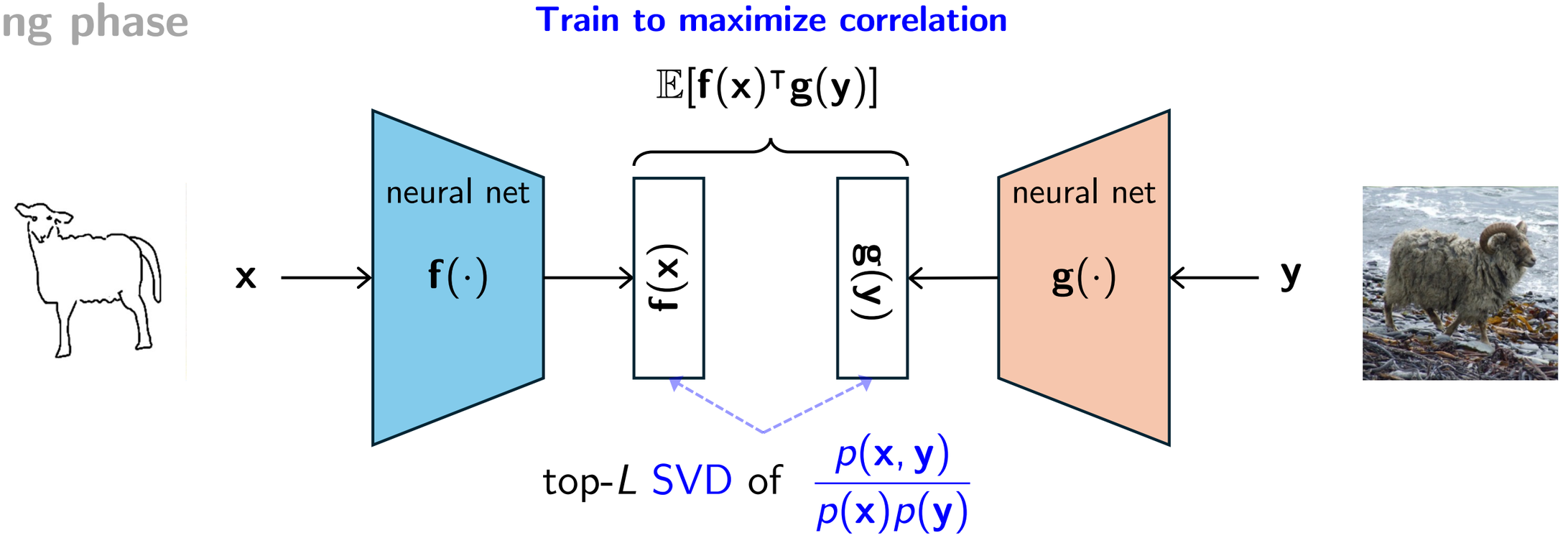
Application 3. **Sketch-Based Image Retrieval**



Application 3. **Sketch-Based Image Retrieval**

Training phase

Train to maximize correlation

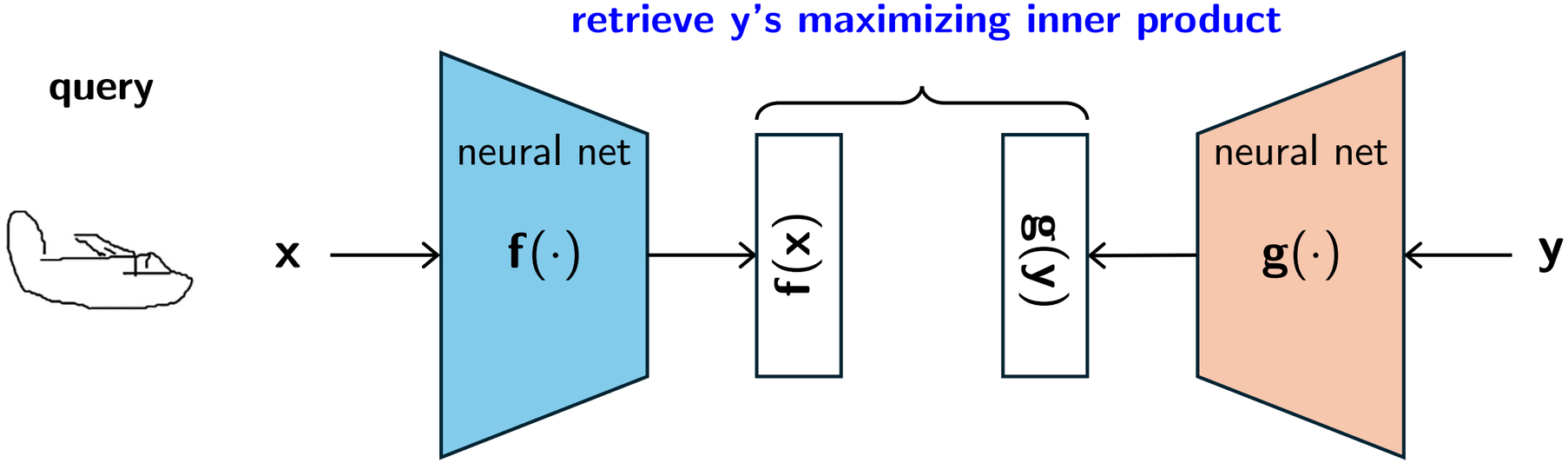


Equivalent to SVD of conditional expectation operator

$$(\mathcal{K}g)(\mathbf{x}) = \mathbb{E}_{p(\mathbf{y}|\mathbf{x})}[g(\mathbf{y})]$$

Application 3. **Sketch-Based Image Retrieval**

Test phase



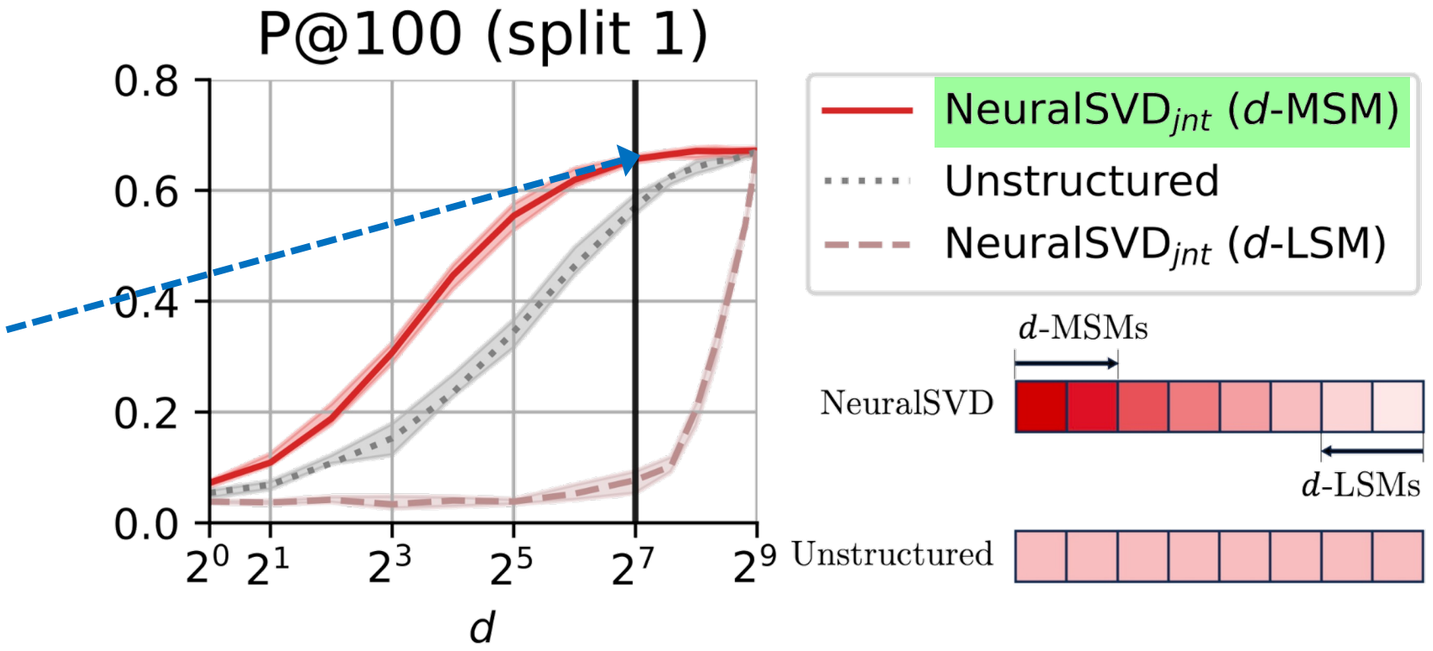
Return $\arg\max_{\mathbf{y}} \mathbf{f}(\mathbf{x})^\intercal \mathbf{g}(\mathbf{y})$

Application 3. **Sketch-Based Image Retrieval**

Table: Evaluation of the ZS-SBIR task with the Sketchy Extended dataset [9].

Model	Gen. model	Ext. knowledge	P@100	mAP
LCALE [10]	*	Word embed.	0.583	0.476
IIAE [11]	*		0.659	0.573
NeuralSVD			**0.670** ±0.010	**0.581** ±0.008

Structured representation: full performance w/ only ¼ dim.



Takeaways and Future Directions

- **NeuralSVD** provides **principled, clean & scalable solutions**

- **Ongoing / future directions**
  - **Large-scale scientific simulation**
    (quantum chemistry, density functional theory, molecular dynamics, ...)
  - **New variational principles**
  - **Beyond two modalities** via structured tensor decomposition?
  - **Theoretical understanding** on architectures / scaling law

Thank You!

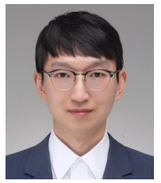
References

J. Jon Ryu, Xiangxiang Xu, Melichan Erol, Yuheng Bu, Lizhong Zheng, Gregory W. Wornell, "Operator SVD with neural networks via nested low-rank approximation," ICML 2024





Minchan Jeong*, **J. Jon Ryu***, Se-Young Yun, Gregory W. Wornell, "Efficient Parametric SVD of Koopman Operator for Stochastic Dynamical Systems," arXiv:2507.07222, Submitted


J. Jon Ryu, Samuel Zhou, Gregory W. Wornell, "Revisiting Orbital Minimization for Neural Operator Decomposition," Submitted