SELF-SUPERVISED EVOLUTION OPERATOR LEARNING FOR HIGH-DIMENSIONAL DYNAMICAL SYSTEMS

Anonymous authors
Paper under double-blind review

ABSTRACT

We introduce an end-to-end approach to learn the evolution operators of large-scale non-linear dynamical systems, such as those describing complex natural phenomena. Evolution operators are particularly well-suited for analyzing systems that exhibit spatio-temporal patterns and have become a key analytical tool across various scientific communities. As terabyte-scale weather datasets and simulation tools capable of running millions of molecular dynamics steps per day are becoming commodities, our approach provides an effective tool to make sense of them from a data-driven perspective. The core of it lies in a remarkable connection between self-supervised representation learning methods and the recently established learning theory of evolution operators. We deploy our approach across multiple scientific domains: explaining the folding dynamics of small proteins, the binding process of drug-like molecules in host sites, and autonomously finding patterns in climate data. Our code is available open-source at: `https://anonymous.4open.science/r/encoderops-5F67`.

1 INTRODUCTION

Dynamical systems are fundamental to understanding phenomena across a vast range of scientific disciplines, from physics and biology to climate science and engineering. Traditionally, scientists have modeled these systems by formulating differential equations from first principles. However, as systems grow in scale and complexity, this approach quickly becomes computationally burdensome and difficult to interpret (Anderson, 1972), hindering the study of large-scale phenomena. Simultaneously, advancements in data collection techniques and computational power have led to an explosion of available data from experiments (Hersbach et al., 2020; Chanussot et al., 2021) and high-fidelity simulations (Harvey et al., 2009; Abraham et al., 2015; Eastman et al., 2017; Bauer et al., 2015). This abundance of data makes data-driven approaches increasingly appealing for studying complex dynamics, with machine learning (Shalev-Shwartz & Ben-David, 2014) becoming a dominant paradigm for learning dynamical systems, largely focusing on predictive tasks such as forecasting. The recent revolution in data-driven weather modeling (Pathak et al., 2022; Bi et al., 2022; Lam et al., 2023; Kochkov et al., 2024) stands as a paradigmatic example of ML's power in handling complex spatio-temporal dynamics. Similarly, reinforcement learning (Sutton & Barto, 1998) has reimagined control theory by leveraging data-driven strategies to optimize system behavior. While these data-driven methods excel at prediction and simulation, there remains a significant gap in approaches that offer interpretability. In scientific contexts, merely predicting system behavior is often insufficient; understanding why a system evolves in a certain way is paramount. For instance, comprehending the dynamical shortcuts and bottlenecks happening through atomistic interaction is crucial for understanding why a drug binds to a specific target or fails to do so, a level of insight not typically provided by black-box predictive models.

A modeling paradigm particularly well-suited for interpretability is that of *evolution operators* (Lasota & Mackey, 1994; Applebaum, 2009). Under mild assumptions, dynamical systems and stochastic processes can be represented by a linear operator — a mathematical entity that maps functions to other functions. This operator-based approach offers multiple advantages. First, it linearizes the dynamics, greatly simplifying tasks like forecasting and controller design. Second, these operators possess a spectral decomposition[1] (Reed & Simon, 1972), which expresses the system's complex dynamics as

[1] A generalization of the eigenvalue decomposition of a matrix.

a linear combination of fundamental, coherent spatio-temporal modes (Molgedey & Schuster, 1994). Each mode represents a distinct, intrinsic pattern associated with a unique spatio-temporal structure defined in terms of growth or decay rates and oscillation frequencies. By identifying and analyzing these principal modes, researchers gain deep insights into the underlying mechanisms driving the system's macroscopic behavior, offering a structured, physically meaningful understanding.

Building on the understanding that evolution operators provide a powerful framework for interpretable analysis, significant effort has been directed towards learning these operators directly from data Kovachki et al. (2023). Data-driven approaches for this task emerged already in the early 2000s, including pioneering work utilizing transfer operators for analyzing stochastic processes in computational biophysics (Schütte et al., 2001), as well as the dynamic mode decomposition family of methods (Schmid, 2010b) for deterministic systems via the Koopman operator. In the ensuing years, there has been a significant acceleration in machine learning methods for evolution operator learning, encompassing theoretical advances through kernel methods and powerful end-to-end deep learning approaches.

Contributions. In this work, we build upon these recent foundations, showing how evolution operator learning can be scaled to structured and high-dimensional dynamical systems. We formalize a principled end-to-end protocol that is amenable to GPU training and prove its equivalence to a self-supervised representation learning problem. Leveraging this link, we also show the transferability of our trained models in both molecular dynamics and climate settings. Code, data, and weights are made available open-source.

2 Evolution operators and how to learn them

Evolution operator learning is a data-driven approach to characterizing dynamical systems, either stochastic, $x_{t+1} \sim p(\cdot|x_t)$, or deterministic, $x_{t+1} \sim \delta(\cdot - F(x_t))$. Throughout, we assume the dynamics to be Markovian, so that the evolution of x_t depends on x_t alone and not on the states at times $s < t$. If this assumption is not satisfied by x_t, a standard trick is to re-define the state as a context $c_t^H = f(x_t, x_{t-1}, \ldots, x_{t-H})$ with history length H, where f can be a simple concatenation, or a learned sequence model (e.g., a recurrent neural network or transformer).

Evolution operators are defined as follows: for every function f of the state of the system, $(\mathsf{E}f)(x_t)$ is the expected value of f one step ahead in the future, given that at time t the system was found in x_t

$$(\mathsf{E}f)(x_t) = \int p(dy|x_t)f(y) = \mathbb{E}_{y\sim X_{t+1}|X_t}[f(y)|x_t]. \tag{1}$$

Notice that E is an operator because it maps any function f to another function, $x_t \mapsto (\mathsf{E}f)(x_t)$, and is *linear* because $\mathsf{E}(f + \alpha g) = \mathsf{E}f + \alpha\mathsf{E}g$. When the dynamics is deterministic, E is known as the *Koopman operator* (Koopman, 1931), while in the stochastic case it is known as the *transfer operator* (Applebaum, 2009).

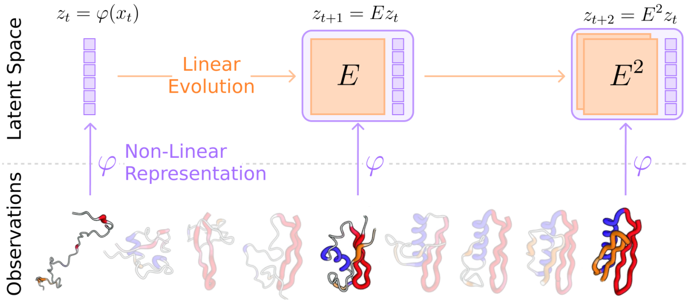
Evolution operators fully characterize the dynamical system because knowing E allows us to reconstruct the dynamical law $p(\cdot|x_t)$. Indeed, for any subset of the state space $B \subseteq \mathcal{X}$, applying E to the indicator function of B, we have

$$(\mathsf{E}1_B)(x_t) = \int_B p(dy|x_t) = \mathbb{P}\left[X_{t+1} \in B|x_t\right].$$

An advantage of the operator approach over dealing directly with the conditional probability $p(\cdot|x_t)$ is that E acts linearly on the objects to which it is applied. This means that operators unlock an arsenal of tools from linear algebra and functional analysis, which would be unavailable otherwise. Arguably the most important of them is the spectral decomposition, allowing us to decompose E, and hence the dynamics, into a linear superposition of dynamical modes. These ideas lie at the core of the celebrated Time-lagged Independent Component Analysis (Molgedey & Schuster, 1994; Pérez-Hernández et al., 2013), and Dynamical Mode Decomposition (Schmid, 2010a; Kutz et al., 2016).

2.1 Learning E and its spectral decomposition from data

We now review the main approaches to learn the evolution operator and its spectral decomposition from a finite dataset of observations, with an emphasis on the least squares approach, which is essential to understand every other method as well.



A core idea of operator learning is that operators are defined by how they act *on a suitable linear space of functions*, similarly to how matrices are defined by their action on a basis of vectors. Of course, not every function f is interesting, and this nicely parallels with the matrix example, where the most "interesting" directions are those that recover most of the variance in the data. Learning E, therefore, is usually cast as the following problem:

> Letting $\varphi(x) \in \mathbb{R}^d$ be a — learned or fixed — encoder of the state, find the best approximation of E *restricted* to the d-dimensional linear space of functions generated by φ, given the data.

In practice, the data is usually a collection of transitions $\mathcal{D} = (x_i, y_i)_{i=1}^{N}$, where it is intended that $x_i \sim \mathbb{P}[X_t]$ are sampled from a distribution of initial states, while $y_i \sim p(\cdot|x_i)$.

Least squares. In this approach the encoder φ is a frozen, that is non-learnable, dictionary of functions, and we are interested in approximating the action of E on functions of the form $f(x) = \langle w, \varphi(x)\rangle$ for every $w \in \mathbb{R}^d$. To this end, one minimizes the empirical error between the true conditional expectation $\mathbb{E}_{y\sim X_{t+1}|X_t}[\langle w, \varphi(y)\rangle\, |x]$, and a linear model $\langle Ew, \varphi(x)\rangle$, where the matrix $E \in \mathbb{R}^{d\times d}$ identifies the restriction of the evolution operator to the linear span of the dictionary:

$$\frac{1}{N}\sum_{i=1}^{N}(\langle w, \varphi(y_i)\rangle - \langle Ew, \varphi(x_i)\rangle)^2 \leq \frac{1}{N}\sum_{i=1}^{N}\|\varphi(y_i) - E^\top\varphi(x_i)\|^2 + \lambda\|E\|^2. \tag{2}$$

On the right-hand side, we assumed $\|w\| \leq 1$, used the Cauchy–Schwarz inequality, and added a ridge penalty. The minimizer of (2) can be computed in closed form (Korda & Mezić, 2018; Kostic et al., 2022, and references therein) as

$$E_\varphi = (C_X + \lambda \mathsf{Id})^{-1} C_{XY}, \quad \text{with } C_{XY} = \frac{1}{N}\sum_{i=1}^{N}\varphi(x_i)\varphi(y_i)^\top \text{ and } C_X = C_{XX}. \tag{3}$$

In the limit of infinite data, $N \to \infty$, and infinitely dimensional encoders, $d \to \infty$, the least squares estimator converges (Korda & Mezić, 2018) in the strong operator topology to the evolution operator E, and similar (but weaker) asymptotic convergence results are proved for its spectrum.

Mode decomposition. The spectral decomposition of E is approximated by expressing the least-squares estimator in its eigenvectors' basis $E_\varphi = Q\Lambda Q^{-1}$, where the columns of $Q = [q_1, \cdots, q_d]$ are the eigenvectors of E_φ, and Λ is a diagonal matrix of eigenvalues. In this basis, the expected value in the future for a function $f(x) = \langle w, \varphi(x)\rangle$ is expressed as

$$\mathbb{E}_{y\sim X_{t+1}|X_t}[f(y)|x] \approx \langle E_\varphi w, \varphi(x)\rangle = \left\langle Q\Lambda Q^{-1}w, \varphi(x)\right\rangle = \sum_{i=1}^{d}\lambda_i \langle q_i, \varphi(x)\rangle\, (Q^{-1}w)_i. \tag{4}$$

The spectral decomposition expresses the transition $x_t \to x_{t+1}$ as a sum of *modes* of the form $\lambda_i \langle q_i, \varphi(x)\rangle\, (Q^{-1}w)_i$, each of which can be broken down into three components:

1. The eigenvalues λ_i determine the time scales of the transition. Indeed, applying the evolution operator s times to analyze the transition $x_t \to x_{t+s}$ leaves (4) unchanged, except that each λ_i becomes λ_i^s. Writing $\lambda_i^s = \rho_i^s e^{is\omega_i}$ in polar coordinates, reveals that the modes decay exponentially over time with rate ρ_i, while oscillating at frequency ω_i.
2. The initial state x influences the decomposition through the factor $\Psi_i(x) = \langle q_i, \varphi(x)\rangle$. This coefficient captures how strongly the state x aligns with the i-th mode. When q_i corresponds to an eigenvalue with slow decay, i.e., $|\lambda_i| \approx 1$, the term $\Psi_i(x)$ serves as a natural quantity for clustering states into *coherent* or *metastable* sets.

3. The coefficient $(Q^{-1}w)_i$, in turn, indicates how the function represented by the vector w relates to the i-th mode. This connection makes it possible to link the dynamical patterns to specific functions — or *observables* – thereby deepening our understanding of the system.

Kernel methods. Leveraging the kernel trick, one can learn evolution operators by deriving a closed-form solution of (2) in terms of kernel matrices whose elements are of the form $k(x_i, x_j) = \langle\varphi(x_i), \varphi(x_j)\rangle$, with $k(\cdot,\cdot)$ a suitable kernel function. Thanks to the theory of reproducing kernel Hilbert spaces, this class of methods is backed up by statistical learning guarantees, such as the ones derived in (Kostic et al., 2022; 2023a; Nüske et al., 2023). Similarly to the least-squares approach, one also approximates the spectral decomposition of E via kernel methods, and this task captured quite a lot of attention from researchers in this area, see (Williams et al., 2014; Kawahara, 2016; Klus et al., 2019; Das & Giannakis, 2020; Alexander & Giannakis, 2020; Meanti et al., 2023).

Deep learning. In contrast to the previous approaches, where the encoder φ is prescribed, a number of methods proposed to approximate E from data with end-to-end schemes including φ as a learnable neural network. Since learning E ultimately entails learning its action on the linear space spanned by φ, it is appealing to choose an encoder capturing the most salient features of the dynamics. To this end, one can train φ via an *encoder-decoder* scheme as proposed in (Takeishi et al., 2017; Lusch et al., 2018; Otto & Rowley, 2019; Azencot et al., 2020; Wehmeyer & Noé, 2018; Frion et al., 2024) or with *encoder-only* approaches as in (Li et al., 2017; Mardt et al., 2018; Yeung et al., 2019; Kostic et al., 2023b; Federici et al., 2023; Jeong et al., 2025).

In encoder-decoder schemes, φ is trained alongside a decoder network, minimizing a combination of prediction and reconstruction errors[2]. Yet, while minimizing a reconstruction loss biases the model towards accurate forecasts of the near future, recent work on model-based reinforcement learning, where one is instead interested in long-term behaviours, suggests that the presence of a decoder is detrimental (Lyu et al., 2023; Schwarzer et al., 2020; Hansen et al., 2023) to control tasks. Similarly, (Balestriero & LeCun, 2024) showed how features learned by reconstruction are both uninformative for perception and hardly transferable.

On the other hand, the competitive advantage of evolution operators over techniques such as (Pathak et al., 2022; Lam et al., 2023; Pfaff et al., 2020; Sanchez-Gonzalez et al., 2020; Li et al., 2020) lies in their spectral decomposition, useful for interpretability, reduced order modeling, and control tasks. Encoder-only approaches follow this intuition and prioritize approximating the spectral decomposition of E over the raw forecasting performances. Concretely, this is accomplished via loss functions that are minimized when φ spans the leading singular space of E. Clearly, once an encoder has been trained, it can be transferred to similar dynamical systems, as we demonstrate in 4.2 and 4.3.

In this work, we propose an encoder-only method based on a loss function originally designed for self-supervised representation learning. Our approach is numerically stable, scales efficiently, enables transfer across related systems, and can incorporate structural priors, e.g., graph-based encoders, beyond the reach of classical DMD approaches. Though our approach is broadly applicable, we mainly focus on applications involving interpretability and model reduction of scientific dynamical systems, highlighting how ML evolution operators can help in advancing fundamental science. Recent works in RL (Lyu et al., 2023; Schwarzer et al., 2020; Rozwood et al., 2024; Novelli et al., 2024) suggest that our approach can be relevant for control tasks, but we leave this for future work.

3 Learning evolution operators via self-supervision

As discussed above, we are interested in the evolution operator

$$(\mathsf{E}f)(x_t) = \mathbb{E}_{y\sim X_{t+1}|X_t}\left[f(y)|x_t\right] = \mathbb{E}_{y\sim X_{t+1}}\left[\frac{p(y|x_t)}{p(y)}f(y)\right], \tag{5}$$

where in the last equality we expressed the expectation in the form of an importance sampling estimator with respect to the probability of the future state $\mathbb{P}(X_{t+1})$. In so doing, we link the evolution operator E to the density ratio

$$r(x_t, x_{t+1}) = \frac{p(x_{t+1}|x_t)}{p(x_{t+1})}. \tag{6}$$

[2] Notice that trying to minimize the prediction error (2) alone immediately leads to a *representation collapse* with φ mapping every input to 0 to obtain a prediction error of 0.

The core of our operator-learning scheme, Alg. 1, is to optimize a model for the density ratio (6) parametrized as the bilinear form $\langle \varphi(x_t), P\varphi(x_{t+1})\rangle$, similar to Oord et al. (2018). Here, φ is a d-dimensional encoder, while P is a linear *predictor* layer which, as discussed below, equals the action of E on the linear subspace of functions spanned by φ, up to a known linear transformation.

We minimize the L^2 error between the density ratio and our bilinear model $\langle \varphi(x_t), P\varphi(x_{t+1})\rangle$:

$$\begin{aligned}\varepsilon(\varphi, P) =&\mathbb{E}_{(x,y)\sim X_t\otimes X_{t+1}}\left[(r(x,y) - \langle \varphi(x), P\varphi(y)\rangle)^2\right] \\ =&\mathbb{E}_{(x,y)\sim X_t\otimes X_{t+1}}\left[\langle \varphi(x), P\varphi(y)\rangle^2\right] - 2\mathbb{E}_{(x,y)\sim(X_t,X_{t+1})}\left[\langle \varphi(x), P\varphi(y)\rangle\right] + \text{cst.},\end{aligned} \tag{7}$$

where $\mathbb{E}_{X_t\otimes X_{t+1}}$ is the expected value between the product of the marginals X_t and X_{t+1}[3].Estimating the squared term in (7) via U-statistics (Hoeffding, 1992) and foregoing the constant term, we finally get to the empirical loss

$$\hat{\varepsilon}(\varphi, P) = \frac{1}{N(N-1)}\sum_{i\neq j}\langle \varphi(x_i), P\varphi(y_j)\rangle^2 - \frac{2}{N}\sum_{i=1}^{N}\langle \varphi(x_i), P\varphi(y_i)\rangle . \tag{8}$$

The loss function (8) was originally proposed for self-supervised contrastive learning in HaoChen et al. (2021; 2022). Indeed, noticing that $\langle \varphi(x), P\varphi(y)\rangle$ can be interpreted as a measure of similarity between x and y, the first term of (8) minimizes the similarity between randomly chosen $i \neq j$ (i.e., *negative*) pairs, while the second term maximizes the similarity of consecutive (i.e., *positive*) pairs. The loss (8) has also been applied in reinforcement learning (Ren et al., 2022), causal estimation (Sun et al., 2024), and recently Lu et al. (2024) showed that it belongs to a wide class of contrastive learning losses defined by Csiszár f-divergences. Concurrently,Wang et al. (2022); Ryu et al. (2024); Kostic et al. (2024); Jeong et al. (2025) applied it to approximate the SVD of linear operators.

3.1 Theoretical properties of our approach.

Our model $\langle \varphi(x_t), P\varphi(x_{t+1})\rangle$ is characterized by the presence of a linear predictor P, and by a shared encoder between x_t and x_{t+1}, in contrast to the more agnostic choice $\langle \varphi(x_t), \psi(x_{t+1})\rangle$ adopted by HaoChen et al. (2021); Wang et al. (2022); Ryu et al. (2024); Kostic et al. (2024); Jeong et al. (2025). Our choice is deliberate, and we now prove a number of theoretical results highlighting how our parametrization is particularly apt for evolution operator learning, with the predictor layer P having a key role. Every lemma in this section is proved in Appendix A. The first observation was already noticed in Wang et al. (2022), and provides a direct link between (7) and the evolution operator regression formalism developed in Kostic et al. (2022).

Lemma 1. *Let $\varphi : \mathcal{X} \to \mathbb{R}^d$ be an encoder whose components are square-integrable with respect to both μ and ν, and let E be a Hilbert-Schmidt evolution operator. Then, the loss function* (7) *is equivalent to the following operator learning loss:*

$$\varepsilon(\varphi, P) = \|\mathsf{E} - \sum_{i,j}\varphi_i \otimes P_{ij}\varphi_j\|^2_{\mathsf{HS}}.$$

We highlight that when the operator E is not Hilbert-Schmidt, the loss function (8) can still be linked to operator learning. In this more general scenario, the loss promotes encoders φ displaying both a strong dynamical response $\mathsf{E}\varphi$, and a good approximation of the true dynamics, see Appendix A.1.

Plugging our model back into (5), the evolution operator gets parametrized as $(\mathsf{E}f)(x_t) \approx \mathbb{E}_{y\sim X_{t+1}}\left[\langle \varphi(x_t), P\varphi(y)\rangle f(y)\right]$, and if the function f is in the linear span generated by the encoder $f(y) = \langle \varphi(y), w\rangle$, we can simplify the expression above as

$$(\mathsf{E}f)(x_t) = \left\langle \varphi(x_t), P\left(\mathbb{E}_{y\sim X_{t+1}}\left[\varphi(y)\varphi(y)^\top\right]\right)w\right\rangle = \langle \varphi(x), PC_Y w\rangle ,$$

where we introduced the covariance of the futures $C_Y = \mathbb{E}_{y\sim X_{t+1}}[\varphi(y)\varphi(y)^\top]$. Thus, the linear predictor P parametrizes the approximation of the evolution operator E over the finite-dimensional space generated by the state representation φ. We remark that to be sure that a prescribed function f lies in the span of the encoder, one can add it as a non-trainable component of the architecture

[3] That is, the product measure $\mathbb{P}[X_t] \otimes \mathbb{P}[X_{t+1}]$

$\varphi(x) = [\mathrm{NN}(x), f(x)]$, as done in Appendix B.1. Alternatively, one can compute its least-squares approximation $\hat{f}_\varphi$ on the features spanned by φ, and use that in place of f. The following Lemma shows that when the predictor P is optimal, one recovers the least squares estimator (3).

Lemma 2. *For any fixed φ, the predictor P minimizing* (7) *can be computed in closed form $P_* = C_X^{-1} C_{XY} C_Y^{-1}$, and the model for the evolution operator is given by*

$$E_\varphi = P_* C_Y = C_X^{-1} C_{XY} = \textit{Eq. } (3) \textit{ with } \lambda \to 0, \tag{9}$$

coinciding with the least-squares estimator (3).

The final interesting fact about (7) is its relation to the VAMP score (Wu & Noé, 2020), originally introduced for representation learning of molecular kinetics. In particular, the VAMP-2 score can be defined in terms of covariances as

$$\mathrm{VAMP}_2(\varphi) = \| C_X^{-1/2} C_{XY} C_Y^{-1/2} \|_{\mathsf{HS}}^2. \tag{10}$$

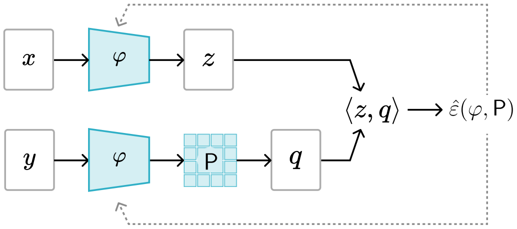
Lemma 3. *For any fixed φ, let P_* the optimal predictor of $\varepsilon(\varphi, P)$, as in Lemma 2. Then, the following holds true:*

$$\varepsilon(\varphi, P_*) = -\| C_X^{-1/2} C_{XY} C_Y^{-1/2} \|_{\mathsf{HS}}^2 = -\mathit{VAMP}_2(\varphi).$$

Our loss function, therefore, matches the negative VAMP-2 score when P is optimal. Compared to methods that directly maximize the VAMP score, such as (Mardt et al., 2018), however, our approach does not require matrix inversions in the computation of the loss (Wu & Noé, 2020), an operation which is unwieldy and prone to instabilities[4] in large-scale applications. Instead, the loss function (8) is written in terms of simple matrix multiplications, making it perfect for GPU-based training.

3.2 Practical implementation

```
for $k = 1$ to $num\_steps$ do
    $\mathcal{B} \leftarrow \{(x_i, y_i) \sim \mathcal{D}\}_{i=1}^{B}$ forall $i$ do
        $z_i \leftarrow \varphi(x_i)$ and $q_i \leftarrow P\varphi(y_i)$
    end
    $r_{ij} \leftarrow \langle z_i, q_j \rangle$
    $d\varphi, dP \leftarrow \nabla \left[ \frac{1}{B(B-1)} \sum_{i \neq j} r_{ij}^2 - \frac{2}{B} \sum_i r_{ii} \right]$
    $\varphi, P \leftarrow \mathrm{opt}(\varphi, P, d\varphi, dP)$
end
```


x
φ
z
y
φ
P
q
$\langle z, q \rangle \longrightarrow \hat{\varepsilon}(\varphi, \mathsf{P})$

Algorithm 1: A pair of consecutive observations (x, y) from a dynamical system are mapped to representations z and q via an embedding function φ. The representation q is also processed by a predictor P. The algorithm iteratively optimizes φ and P using the contrastive objective (8) based on the similarity $\langle z, q \rangle$.

The implementation of our method, summarized in Alg. 1, follows standard self-supervised learning procedures (Chen et al., 2020; Grill et al., 2020; Zbontar et al., 2021; Chen & He, 2021). There, a *positive pair* of data-points, in our case a pair of consecutive observations of the dynamical system, are processed through an encoder network φ, and optionally a predictor network, which in our case is a simple linear layer P. We apply simplicial normalization (Lavoie et al., 2022) to the outputs of the embedding φ. To keep our implementation as close to the theoretical insights as possible, we didn't concatenate additional projection heads to the encoder φ, as suggested in (Chen et al., 2020; Grill et al., 2020; Zbontar et al., 2021; Chen & He, 2021). Furthermore, because of the identity (9) we kept P linear, but it is worth mentioning that tiny MLPs might be employed as predictors instead.

Once a representation φ is learned, we model the evolution operator E via the least-squares estimator E_φ from (3). To compute it, one can make use of the closed form expression (3) by computing the covariances at the end of the training of φ, as done in Kostic et al. (2023b); Jeong et al. (2025). This two-step procedure, however, requires a full forward pass over the full training dataset, which is impractical for large problems. Another option is to use (9), but this again requires the evaluation

[4] Backpropagation through inversions may lead to gradient explosion (Golub & Pereyra, 1973).

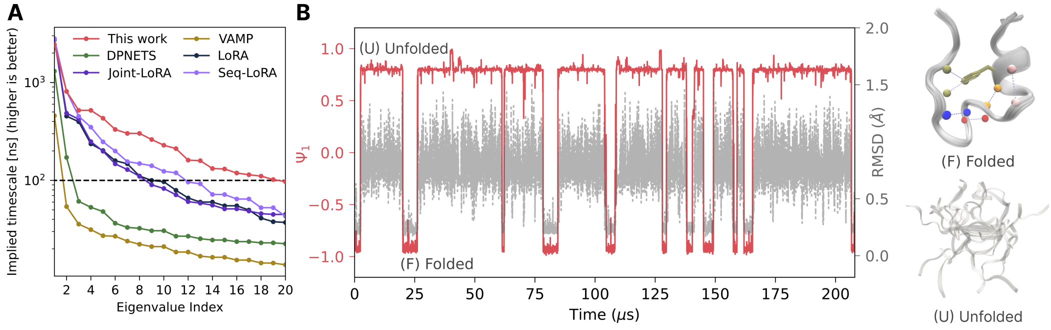


Figure 1: *Trp-Cage folding*. **A**: Implied timescales derived from different baseline methods. Higher implied timescales are associated with more accurate approximations of the slow-modes. **B**: Time series of the leading eigenfunction Ψ_1 (red, left axis) alongside RMSD (gray, right axis), capturing transitions between folded and unfolded states. Representative snapshots of each state are shown. In the folded structure, key hydrogen bonds identified as relevant by the LASSO model are highlighted.

of the covariance, and might be suboptimal whenever P isn't yet converged to the true minimizer P_*. In our implementation, instead, we kept two buffers for C_X and C_{XY}, which are updated online during the training loop via an exponentially moving average of the batch covariances. At the end of the training, we use buffers to compute E_φ as in (3). In Appendix B.5, we show that covariances updated online during training converge to an accurate approximation of the true covariances, and yield identical (or slightly better) results compared to re-evaluating the covariances from scratch.

4 Experiments

We now put to the test our method on high-dimensional dynamical systems from both molecular dynamics and climate domains. Our focus is on assessing the capability of the method to decompose complex dynamics and to evaluate the generalizability of the learned representations. In Appendix B.1 we also report an additional experiment on the Lorenz '63 system (Lorenz, 1963), illustrating how the method can also be used for small forecasting tasks.

4.1 High-resolution dynamical modeling of protein folding

The Trp-Cage miniprotein is a widely studied benchmark for protein folding due to its small size and fast dynamics (Lindorff-Larsen et al., 2011). Previous works, including SRV-based Markov State Models (Sidky et al., 2019) and GraphVAMPNet (Ghorbani et al., 2022), have modeled Trp-Cage dynamics using coarse-grained representations, where the state of the system is defined by the small subset of 20 C$_\alpha$ atoms in the backbone of the protein. Our approach allows us to scale to a more expressive molecular representation based on all 144 heavy atoms, employing the SchNet (Schütt et al., 2017) graph neural network architecture as the encoder φ. After training, we calculate the eigenvalue decomposition of the evolution operator as described in Sec. 2. As shown in Fig. 1B, the leading eigenfunction $\Psi_1(x) = \langle q_1, \varphi(x) \rangle$ correlates strongly with the system's root-mean-square deviation (RMSD) from the folded structure, confirming that Ψ_1 encodes the folding-to-unfolding transition. Clustering the molecular configurations according to the values of Ψ_1 reveals a clear separation between folded and unfolded ensembles (see snapshots in Fig. 1B).

To interpret the nature of this slow mode, we regress Ψ_1 against a library of physically meaningful descriptors—specifically, hydrogen bond interactions across residue pairs—using a sparse LASSO model (Brunton et al., 2016; Zhang et al., 2024; Novelli et al., 2022). This analysis reveals a network of hydrogen bonds stabilizing the folded state, including contributions from side-chain interactions that would be invisible to coarse-grained dynamical models such as (Ghorbani et al., 2022). Finally, we note that the implied timescales[5] τ_i derived from the leading eigenvalues of the learned operators are influenced by both the choice of representation and learning loss. The LoRA baselines from Jeong et al. (2025) use a similar loss function, but do not share the encoder φ between x_t and x_{t+1}. VAMP,

[5] The implied timescale can be computed from the eigenvalues as $\tau_i = -\Delta t / \log(\lambda_i)$, where Δt is the time lag between two consecutive observations.

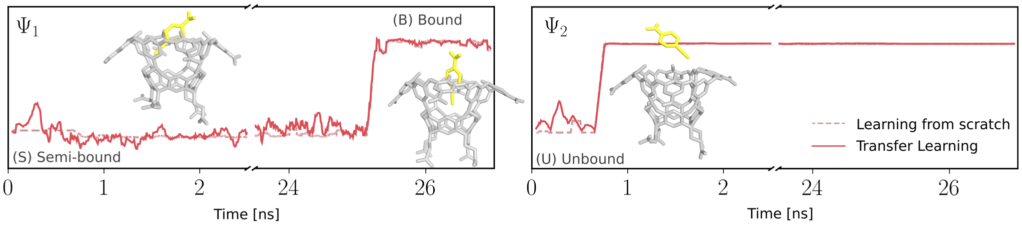


Figure 2: *Calixarene binding*. Eigenfunctions Ψ_1 (left) and Ψ_2 (right) capture ligand transitions from unbound (U) to semi-bound (S) and bound (B) states. The model using a representation transferred from other ligands (solid line) closely matches one trained from scratch (dashed).

from Mardt et al. (2018) and DPNETS from Kostic et al. (2023b), on the other hand, minimize different loss functions. According to the variational principle for Markov processes (Wu & Noé, 2020; Noé & Nuske, 2013), higher implied timescales indicate a better approximation of the system's true slow dynamics (see Fig. 1A).

4.2 Learning transferable representations for the binding of small molecules

Our second case study focuses on the binding of small molecules to a calixarene-based system (Yin et al., 2017), which is often used as a simplified model to study the dynamical processes relevant, for instance, in drug design. Our baseline is obtained by using Alg. 1 to train an encoder φ on molecular dynamics data describing the binding dynamics of a single molecule (G2) to the host system. As in the previous example, we employ a SchNet architecture for φ. As shown in Fig. 2, the slowest dynamical mode, captured by the dominant eigenfunction Ψ_1, is associated with a transition between a semi-bound configuration and the fully bound state. Structural inspection reveals that this intermediate state corresponds to a misaligned pose of the guest, caused by the presence of a water molecule occupying the binding pocket. The second eigenfunction Ψ_2 instead resolves the unbound-to-bound transition. Our findings align with previous works (Rizzi et al., 2021), where water occupancy was identified as a key kinetic bottleneck in host–guest interactions.

We now turn to a key question: can a representation trained on one set of molecular systems generalize to others? This capability is essential for scalable modeling in applications like drug discovery, where retraining a model for every new compound is prohibitive. To test the transferability of the representations φ trained with our method, we trained the encoder on molecular dynamics simulations for two molecules (G1 and G3), and used it to analyze the binding dynamics of a *different ligand* (G2). Using the frozen encoder, we compute the evolution operator of (G2) via (3), and examine its dominant eigenfunctions. Remarkably, the transferred representation successfully recovers the key dynamical modes of the binding process of (G2) without having seen it during the representation learning phase. In particular, it recovers both the entry of the guest molecule into the host cavity and its final locked configuration (Fig. 2). This result illustrates that our self-supervised model learns features that are not only informative but also transferable across molecular systems.

4.3 Patterns in Global Climate

Finally, we test our method on climate data. Specifically, we aim to retrieve El Niño–Southern Oscillation (ENSO), one of the most influential sources of interannual climate variability (Diaz & Markgraf, 2000; Callahan & Mankin, 2023), arising from coupled ocean–atmosphere dynamics in the tropical Pacific (Bjerknes, 1969; Philander, 1983). Characterizing ENSO remains a central goal in climate science, particularly in the context of its potential changes under global warming (McPhaden et al., 2006; Cai et al., 2021).ENSO is conventionally characterized by monthly-averaged sea surface temperature (SST) anomalies, denoted as SST*, computed following the procedure described in (NCP Center). The SST fields are obtained from the ORAS5 reanalysis (Zuo et al., 2019) and provided through the ChaosBench dataset (Nathaniel et al., 2024). However, the dataset comprises only 540 snapshots, which may hinder effective model training. To overcome this, analogous to the drug design experiment described in Sec. 4.2, we adopt a transfer learning strategy using a longer synthetic trajectory generated by the Community Earth System Model (CESM) (Hurrell et al., 2013), consisting of 12,598 samples. The objectives of this experiment are twofold: (i) to determine whether

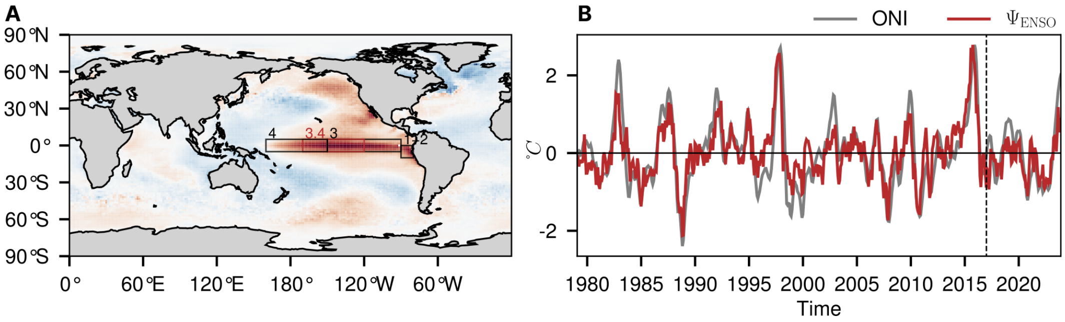


Figure 3: ENSO mode retrieved with our method. **A**: Mode associated with the second leading eigenfunction, highlighting dominant activation in the tropical Pacific. Boxes indicate standard ENSO monitoring zones. **B**: Right eigenfunction corresponding to the second leading eigenvalue, compared to the ONI index. The vertical line marks the split between training and validation sets.

our method can retrieve ENSO dynamics, and (ii) to assess whether representations φ learned from simulated data can be effectively transferred to real-world climate observations.

A convolutional neural network-based encoder is trained using the simulated SST* fields, after which the learned representation φ is applied to real data. The transfer operator is then estimated following (3), using the period 1979–2016 for training and 2017–2023 for validation, and subsequently subjected to spectral decomposition to extract the dominant modes. As expected, this procedure recovers modes corresponding to known climate periodicities, such as annual oscillations (see Tab. 2). Remarkably, one of the leading nontrivial modes (second in magnitude) clearly reflects ENSO dynamics. The associated right eigenfunction exhibits a strong Pearson correlation ($r = 0.82, p < .001$) with the Oceanic Niño Index (ONI) (Fig. 3B), a widely used metric for ENSO monitoring (Glantz & Ramirez, 2020), while the associated spatial mode shows dominant activation over the tropical Pacific (Fig. 3A). Importantly, our method generalizes effectively to unseen data, successfully detecting the 2023 El Niño event within the validation set. It is worth noting that training the same model directly on observational data also recovers the ENSO mode; however, the correlation between the associated right eigenfunction and ONI is weaker ($r = 0.71, p < .001$). Additionally, we compared our method against VAMPNets (Mardt et al., 2018) and DPNets (Kostic et al., 2023b), with results indicating that our approach achieves stronger correlation in capturing the ENSO mode (see Appendix B.4).

This experiment underscores the model's ability to autonomously identify complex climate phenomena in an unsupervised manner without prior localization (unlike previous approaches (Froyland et al., 2021; Lapo et al., 2025)). Importantly, the transfer learning approach enables the model to leverage knowledge from large, high-quality simulations to mitigate for the scarcity of observational data, thereby enabling a more robust extraction of complex patterns such as ENSO. These findings highlight the ability of our approach to learn a robust and generalizable representation, effectively transferring knowledge from synthetic simulations to real-world observations

5 Conclusion

In this work, we proposed an end-to-end framework for learning evolution operators and their spectral decomposition. Our method scales effectively to large and complex systems, making it a practical tool for uncovering physically meaningful patterns in their dynamics. By leveraging a connection between contrastive learning objectives and the spectral properties of evolution operators, we break new ground on the transfer of dynamical representations. Our experiments on atomistic and climate systems demonstrate the versatility of our approach and its generalization capabilities. Looking ahead, this connection opens the door to more expressive learning architectures, robust training strategies, and broader applications in scientific discovery and control.

Limitations. Due to the nature of our experiments, evaluation was more qualitative than typical in ML; benchmarks specifically targeting the accuracy of the spectral decomposition are, to the best of our knowledge, not yet available.

Reproducibility statement

Theory. Our theoretical claims are supported by complete proofs provided in Appendix A.
Code. All code used in this study is available at `https://anonymous.4open.science/r/encoderops-5F67`. Detailed experimental procedures and implementation are described in Appendix B.
Datasets. Instructions for generating, downloading, or requesting the datasets are included in Appendix B.

References

Mark James Abraham, Teemu Murtola, Roland Schulz, Szilárd Páll, Jeremy C. Smith, Berk Hess, and Erik Lindahl. Gromacs: High performance molecular simulations through multi-level parallelism from laptops to supercomputers. *SoftwareX*, 1–2:19–25, September 2015.

Romeo Alexander and Dimitrios Giannakis. Operator-theoretic framework for forecasting nonlinear time series with kernel analog techniques. *Physica D: Nonlinear Phenomena*, 409:132520, 2020.

Philip W Anderson. More is different: broken symmetry and the nature of the hierarchical structure of science. *Science*, 177(4047):393–396, 1972.

David Applebaum. *Lévy Processes and Stochastic Calculus*. Cambridge University Press, April 2009.

Omri Azencot, N Benjamin Erichson, Vanessa Lin, and Michael Mahoney. Forecasting sequential data using consistent koopman autoencoders. In *International Conference on Machine Learning*, pp. 475–485. PMLR, 2020.

Randall Balestriero and Yann LeCun. How learning by reconstruction produces uninformative features for perception. In *Proceedings of the 41st International Conference on Machine Learning*, ICML'24. JMLR.org, 2024.

Caitlin C Bannan, Kalistyn H Burley, Michael Chiu, Michael R Shirts, Michael K Gilson, and David L Mobley. Blind prediction of cyclohexane–water distribution coefficients from the sampl5 challenge. *Journal of Computer-Aided Molecular Design*, 30:927–944, 2016.

Peter Bauer, Alan Thorpe, and Gilbert Brunet. The quiet revolution of numerical weather prediction. *Nature*, 525(7567):47–55, 2015.

Christopher I Bayly, Piotr Cieplak, Wendy Cornell, and Peter A Kollman. A well-behaved electrostatic potential based method using charge restraints for deriving atomic charges: the resp model. *The Journal of Physical Chemistry*, 97(40):10269–10280, 1993.

Soumendranath Bhakat and Pär Söderhjelm. Resolving the problem of trapped water in binding cavities: prediction of host–guest binding free energies in the sampl5 challenge by funnel metadynamics. *Journal of Computer-Aided Molecular Design*, 31:119–132, 2017.

Kaifeng Bi, Lingxi Xie, Hengheng Zhang, Xin Chen, Xiaotao Gu, and Qi Tian. Pangu-weather: A 3d high-resolution model for fast and accurate global weather forecast. *arXiv preprint arXiv:2211.02556*, 2022.

Jakob Bjerknes. Atmospheric teleconnections from the equatorial pacific. *Monthly Weather Review*, 97(3):163–172, 1969.

Luigi Bonati, Enrico Trizio, Andrea Rizzi, and Michele Parrinello. A unified framework for machine learning collective variables for enhanced sampling simulations: mlcolvar. *The Journal of Chemical Physics*, 159(1), 2023.

Steven L Brunton, Joshua L Proctor, and J Nathan Kutz. Discovering governing equations from data by sparse identification of nonlinear dynamical systems. *Proceedings of the national academy of sciences*, 113(15):3932–3937, 2016.

Giovanni Bussi, Davide Donadio, and Michele Parrinello. Canonical sampling through velocity rescaling. *The Journal of Chemical Physics*, 126(1), 2007.

Wenju Cai, Agus Santoso, Matthew Collins, Boris Dewitte, Christina Karamperidou, Jong-Seong Kug, Matthieu Lengaigne, Michael J McPhaden, Malte F Stuecker, Andréa S Taschetto, et al. Changing el niño–southern oscillation in a warming climate. *Nature Reviews Earth & Environment*, 2(9):628–644, 2021.

Christopher W Callahan and Justin S Mankin. Persistent effect of el niño on global economic growth. *Science*, 380(6649):1064–1069, 2023.

Lowik Chanussot, Abhishek Das, Siddharth Goyal, Thibaut Lavril, Muhammed Shuaibi, Morgane Riviere, Kevin Tran, Javier Heras-Domingo, Caleb Ho, Weihua Hu, et al. Open catalyst 2020 (oc20) dataset and community challenges. *Acs Catalysis*, 11(10):6059–6072, 2021.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *International Conference on Machine Learning*, pp. 1597–1607. PmLR, 2020.

Xinlei Chen and Kaiming He. Exploring simple siamese representation learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 15750–15758, 2021.

Matthew J Colbrook, Lorna J Ayton, and Máté Szőke. Residual dynamic mode decomposition: robust and verified koopmanism. *Journal of Fluid Mechanics*, 955:A21, 2023.

Suddhasattwa Das and Dimitrios Giannakis. Koopman spectra in reproducing kernel Hilbert spaces. *Applied and Computational Harmonic Analysis*, 49(2):573–607, 2020.

Henry F Diaz and Vera Markgraf. *El Niño and the Southern Oscillation: multiscale variability and global and regional impacts*. Cambridge University Press, 2000.

Peter Eastman, Jason Swails, John D. Chodera, Robert T. McGibbon, Yutong Zhao, Kyle A. Beauchamp, Lee-Ping Wang, Andrew C. Simmonett, Matthew P. Harrigan, Chaya D. Stern, Rafal P. Wiewiora, Bernard R. Brooks, and Vijay S. Pande. Openmm 7: Rapid development of high performance algorithms for molecular dynamics. *PLOS Computational Biology*, 13(7):e1005659, 2017.

Marco Federici, Patrick Forré, Ryota Tomioka, and Bastiaan S Veeling. Latent representation and simulation of markov processes via time-lagged information bottleneck. *arXiv preprint arXiv:2309.07200*, 2023.

Anthony Frion, Lucas Drumetz, Mauro Dalla Mura, Guillaume Tochon, and Abdeldjalil Aissa El Bey. Neural Koopman prior for data assimilation. *IEEE Transactions on Signal Processing*, 2024.

Gary Froyland, Dimitrios Giannakis, Benjamin R Lintner, Maxwell Pike, and Joanna Slawinska. Spectral analysis of climate dynamics with operator-theoretic approaches. *Nature Communications*, 12(1):6570, 2021.

Damien Garreau, Wittawat Jitkrittum, and Motonobu Kanagawa. Large sample analysis of the median heuristic. *arXiv preprint arXiv:1707.07269*, 2017.

Mahdi Ghorbani, Samarjeet Prasad, Jeffery B Klauda, and Bernard R Brooks. Graphvampnet, using graph neural networks and variational approach to markov processes for dynamical modeling of biomolecules. *The Journal of Chemical Physics*, 156(18), 2022.

Michael H Glantz and Ivan J Ramirez. Reviewing the oceanic niño index (oni) to enhance societal readiness for el niño's impacts. *International Journal of Disaster Risk Science*, 11:394–403, 2020.

G. H. Golub and V. Pereyra. The differentiation of pseudo-inverses and nonlinear least squares problems whose variables separate. *SIAM Journal on Numerical Analysis*, 10(2):413–432, April 1973.

Jean-Bastien Grill, Florian Strub, Florent Altché, Corentin Tallec, Pierre Richemond, Elena Buchatskaya, Carl Doersch, Bernardo Avila Pires, Zhaohan Guo, Mohammad Gheshlaghi Azar, et al. Bootstrap your own latent-a new approach to self-supervised learning. *Advances in Neural Information Processing Systems*, 33:21271–21284, 2020.

Nicklas Hansen, Hao Su, and Xiaolong Wang. Td-mpc2: Scalable, robust world models for continuous control. *arXiv preprint arXiv:2310.16828*, 2023.

Jeff Z HaoChen, Colin Wei, Adrien Gaidon, and Tengyu Ma. Provable guarantees for self-supervised deep learning with spectral contrastive loss. *Advances in neural information processing systems*, 34:5000–5011, 2021.

Jeff Z HaoChen, Colin Wei, Ananya Kumar, and Tengyu Ma. Beyond separability: Analyzing the linear transferability of contrastive representations to related subpopulations. *Advances in Neural Information Processing Systems*, 35:26889–26902, 2022.

Matt J Harvey, Giovanni Giupponi, and G De Fabritiis. Acemd: accelerating biomolecular dynamics in the microsecond time scale. *Journal of Chemical Theory and Computation*, 5(6):1632–1639, 2009.

Hans Hersbach, Bill Bell, Paul Berrisford, Shoji Hirahara, András Horányi, Joaquín Muñoz-Sabater, Julien Nicolas, Carole Peubey, Raluca Radu, Dinand Schepers, Adrian Simmons, Cornel Soci, Saleh Abdalla, Xavier Abellan, Gianpaolo Balsamo, Peter Bechtold, Gionata Biavati, Jean Bidlot, Massimo Bonavita, Giovanna De Chiara, Per Dahlgren, Dick Dee, Michail Diamantakis, Rossana Dragani, Johannes Flemming, Richard Forbes, Manuel Fuentes, Alan Geer, Leo Haimberger, Sean Healy, Robin J. Hogan, Elías Hólm, Marta Janisková, Sarah Keeley, Patrick Laloyaux, Philippe Lopez, Cristina Lupu, Gabor Radnoti, Patricia de Rosnay, Iryna Rozum, Freja Vamborg, Sebastien Villaume, and Jean-Noël Thépaut. The era5 global reanalysis. *Quarterly Journal of the Royal Meteorological Society*, 146(730):1999–2049, June 2020. ISSN 1477-870X. doi: 10.1002/qj.3803. URL `http://dx.doi.org/10.1002/qj.3803`.

Wassily Hoeffding. A class of statistics with asymptotically normal distribution. *Breakthroughs in Statistics: Foundations and Basic Theory*, pp. 308–334, 1992.

James W Hurrell, Marika M Holland, Peter R Gent, Steven Ghan, Jennifer E Kay, Paul J Kushner, J-F Lamarque, William G Large, D Lawrence, Keith Lindsay, et al. The community earth system model: a framework for collaborative research. *Bulletin of the American Meteorological Society*, 94(9):1339–1360, 2013.

Minchan Jeong, J Jon Ryu, Se-Young Yun, and Gregory W Wornell. Efficient parametric svd of koopman operator for stochastic dynamical systems. *arXiv preprint arXiv:2507.07222*, 2025.

William L Jorgensen, Jayaraman Chandrasekhar, Jeffry D Madura, Roger W Impey, and Michael L Klein. Comparison of simple potential functions for simulating liquid water. *The Journal of Chemical Physics*, 79(2):926–935, 1983.

Yoshinobu Kawahara. Dynamic Mode Decomposition with Reproducing Kernels for Koopman Spectral Analysis. In *Advances in Neural Information Processing Systems*, volume 29, 2016.

Stefan Klus, Ingmar Schuster, and Krikamol Muandet. Eigendecompositions of transfer operators in reproducing kernel Hilbert spaces. *Journal of Nonlinear Science*, 30(1):283–315, 2019.

Dmitrii Kochkov, Janni Yuval, Ian Langmore, Peter Norgaard, Jamie Smith, Griffin Mooers, Milan Klöwer, James Lottes, Stephan Rasp, Peter Düben, et al. Neural general circulation models for weather and climate. *Nature*, 632(8027):1060–1066, 2024.

B. O. Koopman. Hamiltonian systems and transformation in Hilbert space. *Proceedings of the National Academy of Sciences*, 17(5):315–318, May 1931. ISSN 1091-6490. doi: 10.1073/pnas.17.5.315. URL `http://dx.doi.org/10.1073/pnas.17.5.315`.

Milan Korda and Igor Mezić. On convergence of extended dynamic mode decomposition to the Koopman operator. *Journal of Nonlinear Science*, 28:687–710, 2018.

Vladimir Kostic, Pietro Novelli, Andreas Maurer, Carlo Ciliberto, Lorenzo Rosasco, and Massimiliano Pontil. Learning dynamical systems via Koopman operator regression in reproducing kernel Hilbert spaces. *Advances in Neural Information Processing Systems*, 35:4017–4031, 2022.

Vladimir Kostic, Karim Lounici, Pietro Novelli, and Massimiliano Pontil. Sharp spectral rates for Koopman operator learning. *Advances in Neural Information Processing Systems*, 36:32328–32339, 2023a.

Vladimir Kostic, Grégoire Pacreau, Giacomo Turri, Pietro Novelli, Karim Lounici, and Massimiliano Pontil. Neural conditional probability for uncertainty quantification. *Advances in Neural Information Processing Systems*, 37:60999–61039, 2024.

Vladimir R Kostic, Pietro Novelli, Riccardo Grazzi, Karim Lounici, and Massimiliano Pontil. Learning invariant representations of time-homogeneous stochastic dynamical systems. *arXiv preprint arXiv:2307.09912*, 2023b.

Nikola Kovachki, Zongyi Li, Burigede Liu, Kamyar Azizzadenesheli, Kaushik Bhattacharya, Andrew Stuart, and Anima Anandkumar. Neural operator: Learning maps between function spaces with applications to pdes. *Journal of Machine Learning Research*, 24(89):1–97, 2023.

J. Nathan Kutz, Steven L. Brunton, Bingni W. Brunton, and Joshua L. Proctor. *Dynamic Mode Decomposition*. Society for Industrial and Applied Mathematics, 2016.

Remi Lam, Alvaro Sanchez-Gonzalez, Matthew Willson, Peter Wirnsberger, Meire Fortunato, Ferran Alet, Suman Ravuri, Timo Ewalds, Zach Eaton-Rosen, Weihua Hu, et al. Learning skillful medium-range global weather forecasting. *Science*, 382(6677):1416–1421, 2023.

Karl Lapo, Sara M Ichinaga, and J Nathan Kutz. A method for unsupervised learning of coherent spatiotemporal patterns in multiscale data. *Proceedings of the National Academy of Sciences*, 122 (7):e2415786122, 2025.

Andrzej Lasota and Michael C. Mackey. *Chaos, Fractals, and Noise*, volume 97 of *Applied Mathematical Sciences*. Springer New York, 1994.

Samuel Lavoie, Christos Tsirigotis, Max Schwarzer, Ankit Vani, Michael Noukhovitch, Kenji Kawaguchi, and Aaron Courville. Simplicial embeddings in self-supervised learning and downstream classification. *arXiv preprint arXiv:2204.00616*, 2022.

Qianxiao Li, Felix Dietrich, Erik M Bollt, and Ioannis G Kevrekidis. Extended dynamic mode decomposition with dictionary learning: A data-driven adaptive spectral decomposition of the koopman operator. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 27(10), 2017.

Zongyi Li, Nikola Kovachki, Kamyar Azizzadenesheli, Burigede Liu, Kaushik Bhattacharya, Andrew Stuart, and Anima Anandkumar. Fourier neural operator for parametric partial differential equations. *arXiv preprint arXiv:2010.08895*, 2020.

Vittorio Limongelli, Massimiliano Bonomi, and Michele Parrinello. Funnel metadynamics as accurate binding free-energy method. *Proceedings of the National Academy of Sciences*, 110(16):6358–6363, 2013.

Kresten Lindorff-Larsen, Stefano Piana, Ron O Dror, and David E Shaw. How fast-folding proteins fold. *Science*, 334(6055):517–520, 2011.

Edward Lorenz. Deterministic nonperiodic flow. *Journal of Atmospheric Sciences*, 20(2):130–148, 1963.

Yiwei Lu, Guojun Zhang, Sun Sun, Hongyu Guo, and Yaoliang Yu. f-micl: Understanding and generalizing infonce-based contrastive learning. *arXiv preprint arXiv:2402.10150*, 2024.

Bethany Lusch, J Nathan Kutz, and Steven L Brunton. Deep learning for universal linear embeddings of nonlinear dynamics. *Nature Communications*, 9(1):4950, 2018.

Xubo Lyu, Hanyang Hu, Seth Siriya, Ye Pu, and Mo Chen. Task-oriented Koopman-based control with contrastive encoder. In *7th Annual Conference on Robot Learning*, 2023. URL https://openreview.net/forum?id=q0VAoefCI2.

Andreas Mardt, Luca Pasquali, Hao Wu, and Frank Noé. Vampnets for deep learning of molecular kinetics. *Nature communications*, 9(1):5, 2018.

Michael J McPhaden, Stephen E Zebiak, and Michael H Glantz. Enso as an integrating concept in earth science. *Science*, 314(5806):1740–1745, 2006.

Giacomo Meanti, Antoine Chatalic, Vladimir Kostic, Pietro Novelli, Massimiliano Pontil, and Lorenzo Rosasco. Estimating koopman operators with sketching to provably learn large scale dynamical systems. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine (eds.), *Advances in Neural Information Processing Systems*, volume 36, pp. 77242–77276. Curran Associates, Inc., 2023.

Igor Mezić. Spectral properties of dynamical systems, model reduction and decompositions. *Nonlinear Dynamics*, 41(1):309–325, 2005.

Thomas P Minka. Old and new matrix algebra useful for statistics. *See www. stat. cmu. edu/minka/papers/matrix. html*, 4, 2000.

Takeru Miyato, Toshiki Kataoka, Masanori Koyama, and Yuichi Yoshida. Spectral normalization for generative adversarial networks. *arXiv preprint arXiv:1802.05957*, 2018.

L. Molgedey and H. G. Schuster. Separation of a mixture of independent signals using time delayed correlations. *Physical Review Letters*, 72(23):3634–3637, June 1994. ISSN 0031-9007. doi: 10.1103/physrevlett.72.3634. URL `http://dx.doi.org/10.1103/PhysRevLett.72.3634`.

Juan Nathaniel, Yongquan Qu, Tung Nguyen, Sungduk Yu, Julius Busecke, Aditya Grover, and Pierre Gentine. Chaosbench: A multi-channel, physics-based benchmark for subseasonal-to-seasonal climate prediction. *arXiv preprint arXiv:2402.00712*, 2024.

NOAA's climate prediction center NCP Center. Climate Prediction Center - ONI — origin.cpc.ncep.noaa.gov. `https://origin.cpc.ncep.noaa.gov/products/analysis_monitoring/ensostuff/ONI_v5.php`. [Accessed 09-05-2025].

Frank Noé and Feliks Nuske. A variational approach to modeling slow processes in stochastic dynamical systems. *Multiscale Modeling & Simulation*, 11(2):635–655, 2013.

Pietro Novelli, Luigi Bonati, Massimiliano Pontil, and Michele Parrinello. Characterizing metastable states with the help of machine learning. *Journal of Chemical Theory and Computation*, 18(9): 5195–5202, 2022.

Pietro Novelli, Marco Pratticò, Massimiliano Pontil, and Carlo Ciliberto. Operator world models for reinforcement learning. *arXiv preprint arXiv:2406.19861*, 2024.

Feliks Nüske, Sebastian Peitz, Friedrich Philipp, Manuel Schaller, and Karl Worthmann. Finite-data error bounds for Koopman-based prediction and control. *Journal of Nonlinear Science*, 33(1):14, 2023.

Aaron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*, 2018.

Samuel E Otto and Clarence W Rowley. Linearly recurrent autoencoder networks for learning dynamics. *SIAM Journal on Applied Dynamical Systems*, 18(1):558–593, 2019.

Bette L Otto-Bliesner, Esther C Brady, John Fasullo, Alexandra Jahn, Laura Landrum, Samantha Stevenson, Nan Rosenbloom, Andrew Mai, and Gary Strand. Climate variability and change since 850 ce: An ensemble approach with the community earth system model. *Bulletin of the American Meteorological Society*, 97(5):735–754, 2016.

Jaideep Pathak, Shashank Subramanian, Peter Harrington, Sanjeev Raja, Ashesh Chattopadhyay, Morteza Mardani, Thorsten Kurth, David Hall, Zongyi Li, Kamyar Azizzadenesheli, et al. Fourcastnet: A global data-driven high-resolution weather model using adaptive fourier neural operators. *arXiv preprint arXiv:2202.11214*, 2022.

Guillermo Pérez-Hernández, Fabian Paul, Toni Giorgino, Gianni De Fabritiis, and Frank Noé. Identification of slow molecular order parameters for markov model construction. *The Journal of Chemical Physics*, 139(1), 2013.

Tobias Pfaff, Meire Fortunato, Alvaro Sanchez-Gonzalez, and Peter Battaglia. Learning mesh-based simulation with graph networks. In *International Conference on Learning Representations*, 2020.

S George H Philander. El nino southern oscillation phenomena. *Nature*, 302(5906):295–301, 1983.

Michael Reed and Barry Simon. *Methods of Modern Mathematical Physics*. Academic Press, 1972. ISBN 9780125850018.

Tongzheng Ren, Tianjun Zhang, Lisa Lee, Joseph E Gonzalez, Dale Schuurmans, and Bo Dai. Spectral decomposition representation for reinforcement learning. *arXiv preprint arXiv:2208.09515*, 2022.

Valerio Rizzi, Luigi Bonati, Narjes Ansari, and Michele Parrinello. The role of water in host-guest interaction. *Nature Communications*, 12(1):93, 2021.

Preston Rozwood, Edward Mehrez, Ludger Paehler, Wen Sun, and Steven L Brunton. Koopman-assisted reinforcement learning. *arXiv preprint arXiv:2403.02290*, 2024.

J. Jon Ryu, Xiangxiang Xu, H. S. Melihcan Erol, Yuheng Bu, Lizhong Zheng, and Gregory W. Wornell. Operator svd with neural networks via nested low-rank approximation. In *Proceedings of the 41st International Conference on Machine Learning*, ICML'24. JMLR.org, 2024.

Alvaro Sanchez-Gonzalez, Jonathan Godwin, Tobias Pfaff, Rex Ying, Jure Leskovec, and Peter Battaglia. Learning to simulate complex physics with graph networks. In *International Conference on Machine Learning*, pp. 8459–8468. PMLR, 2020.

Peter J. Schmid. Dynamic mode decomposition of numerical and experimental data. *Journal of Fluid Mechanics*, 656:5–28, July 2010a.

Peter J Schmid. Dynamic mode decomposition of numerical and experimental data. *Journal of Fluid Mechanics*, 656:5–28, 2010b.

Kristof Schütt, Pieter-Jan Kindermans, Huziel Enoc Sauceda Felix, Stefan Chmiela, Alexandre Tkatchenko, and Klaus-Robert Müller. Schnet: A continuous-filter convolutional neural network for modeling quantum interactions. *Advances in Neural Information Processing Systems*, 30, 2017.

Ch Schütte, Wilhelm Huisinga, and Peter Deuflhard. *Transfer operator approach to conformational dynamics in biomolecular systems*. Springer, 2001.

Max Schwarzer, Ankesh Anand, Rishab Goel, R Devon Hjelm, Aaron Courville, and Philip Bachman. Data-efficient reinforcement learning with self-predictive representations. *arXiv preprint arXiv:2007.05929*, 2020.

Shai Shalev-Shwartz and Shai Ben-David. *Understanding Machine Learning: From Theory to Algorithms*. Cambridge University Press, 2014.

Hythem Sidky, Wei Chen, and Andrew L Ferguson. High-resolution markov state models for the dynamics of trp-cage miniprotein constructed over slow folding modes identified by state-free reversible vampnets. *The Journal of Physical Chemistry B*, 123(38):7999–8009, 2019.

Haotian Sun, Antoine Moulin, Tongzheng Ren, Arthur Gretton, and Bo Dai. Spectral representation for causal estimation with hidden confounders. *arXiv preprint arXiv:2407.10448*, 2024.

Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*, volume 1. MIT Press, 1998.

Naoya Takeishi, Yoshinobu Kawahara, and Takehisa Yairi. Learning Koopman invariant subspaces for dynamic mode decomposition. *Advances in Neural Information Processing Systems*, 30, 2017.

Gareth A Tribello, Massimiliano Bonomi, Davide Branduardi, Carlo Camilloni, and Giovanni Bussi. Plumed 2: New feathers for an old bird. *Computer Physics Communications*, 185(2):604–613, 2014.

Warwick Tucker. The lorenz attractor exists. *Comptes Rendus de l'Académie des Sciences-Series I-Mathematics*, 328(12):1197–1202, 1999.

Junmei Wang, Romain M Wolf, James W Caldwell, Peter A Kollman, and David A Case. Development and testing of a general amber force field. *Journal of Computational Chemistry*, 25(9): 1157–1174, 2004.

Ziyu Wang, Yucen Luo, Yueru Li, Jun Zhu, and Bernhard Schölkopf. Spectral representation learning for conditional moment models. *arXiv preprint arXiv:2210.16525*, 2022.

Christoph Wehmeyer and Frank Noé. Time-lagged autoencoders: Deep learning of slow collective variables for molecular kinetics. *The Journal of Chemical Physics*, 148(24), 2018.

Matthew O Williams, Clarence W Rowley, and Ioannis G Kevrekidis. A kernel-based approach to data-driven Koopman spectral analysis. *arXiv preprint arXiv:1411.2260*, 2014.

Hao Wu and Frank Noé. Variational approach for learning markov processes from time series data. *Journal of Nonlinear Science*, 30(1):23–66, 2020.

Enoch Yeung, Soumya Kundu, and Nathan Hodas. Learning deep neural network representations for Koopman operators of nonlinear dynamical systems. In *2019 American Control Conference (ACC)*, pp. 4832–4839. IEEE, 2019.

Jian Yin, Niel M Henriksen, David R Slochower, Michael R Shirts, Michael W Chiu, David L Mobley, and Michael K Gilson. Overview of the sampl5 host–guest challenge: Are we doing better? *Journal of Computer-Aided Molecular Design*, 31:1–19, 2017.

Jure Zbontar, Li Jing, Ishan Misra, Yann LeCun, and Stéphane Deny. Barlow twins: Self-supervised learning via redundancy reduction. In *International Conference on Machine Learning*, pp. 12310–12320. PMLR, 2021.

Jintu Zhang, Luigi Bonati, Enrico Trizio, Odin Zhang, Yu Kang, TingJun Hou, and Michele Parrinello. Descriptor-free collective variables from geometric graph neural networks. *Journal of Chemical Theory and Computation*, 20(24):10787–10797, 2024.

Jiawei Zhuang, raphael dussin, David Huard, Pascal Bourgault, Anderson Banihirwe, Stephane Raynaud, Brewster Malevich, Martin Schupfner, Filipe, Charles Gauthier, Sam Levang, André Jüling, Mattia Almansi, RichardScottOZ, RondeauG, Stephan Rasp, Trevor James Smith, Ben Mares, Jemma Stachelek, Matthew Plough, Pierre, Ray Bell, Romain Caneill, and Xianxiang Li. pangeo-data/xesmf: v0.8.10, April 2025. URL `https://doi.org/10.5281/zenodo.15304267`.

Hao Zuo, Magdalena Alonso Balmaseda, Steffen Tietsche, Kristian Mogensen, and Michael Mayer. The ecmwf operational ensemble reanalysis–analysis system for ocean and sea ice: a description of the system and assessment. *Ocean Science*, 15(3):779–808, 2019.

Concurrent work. During the preparation of this submission (September 2025), we were made aware of a concurrent preprint (Jeong et al., 2025)[6] proposing a similar methodology to learn the evolution operators of stochastic dynamical systems. In particular, Jeong et al. (2025) proposes a variation of the loss (7) with the more agnostic choice $\langle \varphi(x_t), \psi(x_{t+1}) \rangle$ with both φ, ψ trainable. Our work, however, differs from Jeong et al. (2025) on two key aspects. First, as proved in Sec. 3.1, our parametrization $\langle \varphi(x_t), P\varphi(x_{t+1}) \rangle$ recovers the least-squares evolution operator learning framework (Lemma 1), for which both approximation (Korda & Mezić, 2018) and statistical learning Kostic et al. (2022); Nüske et al. (2023) results have been proved. Furthermore, as discussed in Experiment Sec. 4.1, our parametrization results in higher-implied timescales which, according to the variational principle (Noé & Nuske, 2013; Wu & Noé, 2020), are associated with an improved accuracy in the estimation of the evolution operator. Our parametrization is also directly linked to the VAMP score through Lemma 3.

As a second point of departure from Jeong et al. (2025), our experiments in Sec. 4 focus on high-dimensional dynamical systems, and to the best of our knowledge demonstrate evolution operator learning at scales never reached so far. Further, drawing on the connection of (7) to self-supervised learning – which is not acknowledged in Jeong et al. (2025) despite the prior work Wang et al. (2022) – our experiments Sec. 4.2 and Sec. 4.3, demonstrate that representations learned via (7) can be successfully transferred to new and unseen dynamical systems.

A Proofs of the theoretical claims.

Define $\nu = \mathbb{P}[X_t]$, the distribution of the initial states in our dataset, and $\mu = \mathbb{E}_{x\sim\nu}[p(\cdot|x)] = \mathbb{P}[X_{t+1}]$ the distribution of the evolved states. In practice, ν can be the following:

- If a simulator is available, ν can be *any* distribution of initial states, and μ is obtained by a single step of the simulator on data from ν.
- If one samples trajectories of length T from an initial distribution $\mathbb{P}[X_1]$, then $\nu = \frac{1}{T}\sum_{i=1}^{T-1}\mathbb{P}[X_i]$.
- If — as in molecular dynamics, or the Lorenz 63 example below — one samples from an *invariant* distribution π such that $\mathbb{P}[X_t] = \pi \implies \mathbb{P}[X_{t+1}] = \pi$, one has $\nu = \mu = \pi$.

The evolution operator E maps functions from $L^2(\mu)$ into $L^2(\nu)$, that is $\mathsf{E} : L^2(\mu) \to L^2(\nu)$. Notice that since we allow for general initial and evolved distributions, respectively ν and μ, our method does *not* require E to be associated to a stationary, nor ergodic dynamical system, as often the case in the theoretical literature, see e.g. (Mezić, 2005, Section 2.3) or Kostic et al. (2022). We will now show this simple equivalence:

Lemma 1. *Let $\varphi : \mathcal{X} \to \mathbb{R}^d$ be an encoder whose components are square-integrable with respect to both μ and ν, and let E be a Hilbert-Schmidt evolution operator. Then, the loss function* (7) *is equivalent to the following operator learning loss:*

$$\varepsilon(\varphi, P) = \|\mathsf{E} - \sum_{i,j}\varphi_i \otimes P_{ij}\varphi_j\|^2_{\mathsf{HS}}.$$

Proof. The Lemma was already proved in Wang et al. (2022, Lemma 4.1), or Kostic et al. (2024, Theorem 1). Here we provide a self-contained proof. Since the encoder is square-integrable in both μ and ν — $\varphi_i \in L^2(\mu)$ and $L^2(\nu)$ — we can define the linear operators

$$\Phi_\mu : L^2(\mu) \to \mathbb{R}^d \qquad f \mapsto f = (\langle f, \varphi_i \rangle_{L^2(\mu)})_{i=1}^d.$$

$$\Phi_\nu^* : \mathbb{R}^d \to L^2(\nu) \qquad z \mapsto \sum_{i=1}^d \varphi_i(\cdot) z_i.$$

By direct substitution of the definition above, it follows that

$$\|\mathsf{E} - \Phi_\nu^* P \Phi_\mu\|^2_{\mathsf{HS}} = \|\mathsf{E} - \sum_{i,j}\varphi_i \otimes P_{ij}\varphi_j\|^2_{\mathsf{HS}}.$$

[6] Version 1 of Jul 9, 2025.

Now, let's notice that by direct calculation one obtains

$$\Phi_\nu \Phi_\nu^* = \mathbb{E}_\nu \left[\varphi(x)\varphi(x)^\top\right] \qquad \Phi_\nu \mathsf{E} \Phi_\mu^* = \mathbb{E}_\rho \left[\varphi(x)\varphi(y)^\top\right],$$

where $\rho(dx, dy) = p(dy|x)\nu(dx)$ is the joint distribution of (X_t, X_{t+1}).

By the definition of the Hilbert-Schmidt norm, we have

$$\begin{aligned}
\|\mathsf{E} - \Phi^* P \Phi\|_{\mathsf{HS}}^2 &= \|\mathsf{E}\|_{\mathsf{HS}}^2 - 2\mathrm{Tr}\left[\mathsf{E}^*\Phi_\nu^* P \Phi_\mu\right] + \mathrm{Tr}\left[\Phi_\mu^* P^\top \Phi_\nu \Phi_\nu^* P \Phi_\mu\right] \\
&= \|\mathsf{E}\|_{\mathsf{HS}}^2 - 2\mathrm{Tr}\left[\Phi_\mu \mathsf{E}^* \Phi_\nu^* P\right] + \mathrm{Tr}\left[\Phi_\mu \Phi_\mu^* P^\top \Phi_\nu \Phi_\nu^* P\right] \\
&= \|\mathsf{E}\|_{\mathsf{HS}}^2 - 2\mathbb{E}_{(x,y)\sim\rho}\left[\mathrm{Tr}\left[\varphi(y)\varphi(x)^\top P\right]\right] + \mathbb{E}_{(x,y)\sim\mu\otimes\nu}\left[\mathrm{Tr}\left[\varphi(y)\varphi(y)^\top P^\top \varphi(x)\varphi(x)^\top P\right]\right] \\
&= \|\mathsf{E}\|_{\mathsf{HS}}^2 - 2\mathbb{E}_{(x,y)\sim\rho}\left[\langle\varphi(x), P\varphi(y)\rangle\right] + \mathbb{E}_{(x,y)\sim\mu\otimes\nu}\left[\langle\varphi(x), P\varphi(y)\rangle^2\right],
\end{aligned}$$

where we repeatedly used the cyclic property of the trace. □

The following Lemma shows that when P is optimal with respect to (7), then it recovers the least squares estimator (3).

Lemma 2. *For any fixed φ, the predictor P minimizing (7) can be computed in closed form $P_* = C_X^{-1} C_{XY} C_Y^{-1}$, and the model for the evolution operator is given by*

$$E_\varphi = P_* C_Y = C_X^{-1} C_{XY} = \textit{Eq. } (3) \textit{ with } \lambda \to 0, \tag{9}$$

coinciding with the least-squares estimator (3).

Proof. The proof follows by noticing that $\varepsilon(\varphi, P)$ is convex in P. Taking the gradient (see, for example (Minka, 2000)) one has:

$$\begin{aligned}
\nabla_P \varepsilon(\varphi, P) &= -2\mathbb{E}_{(x,y)\sim\rho}\left[\varphi(y)\varphi(x)^\top\right] + 2\mathbb{E}_{(x,y)\sim\mu\otimes\nu}\left[\varphi(y)\varphi(y)^\top P^\top \varphi(x)\varphi(x)^\top\right] \\
&= -2C_{YX} + 2C_Y P^\top C_X
\end{aligned}$$

As the problem is convex, the global minimum P_* is attained when $\nabla_P \varepsilon(\varphi, P_*) = 0$. This condition is equivalent to solve the equation

$$-2C_{YX} + 2C_Y P_*^\top C_X = 0.$$

By multiplying the expression above by C_X^{-1} on the right and C_Y^{-1} on the left, re-arranging it, and taking the transpose of everything, we finally get

$$P_* = C_X^{-1} C_{XY} C_Y^{-1}.$$

□

The following Lemma shows the equivalence of (7) and the VAMP-2 loss of Wu & Noé (2020); Mardt et al. (2018)

Lemma 3. *For any fixed φ, let P_* the optimal predictor of $\varepsilon(\varphi, P)$, as in Lemma 2. Then, the following holds true:*

$$\varepsilon(\varphi, P_*) = -\|C_X^{-1/2} C_{XY} C_Y^{-1/2}\|_{\mathsf{HS}}^2 = -\mathit{VAMP}_2(\varphi).$$

Proof. By noticing that the loss function (7) can be equivalently rewritten as

$$\varepsilon(\varphi, P) = \mathrm{Tr}[P^\top C_X P C_Y - 2PC_{YX}],$$

and substituting the optimal predictor $P_* = C_X^{-1} C_{XY} C_Y^{-1}$ from Lemma 2 inside this expression, we immediately obtain the identity. □

A.1 On the Hilbert-Schmidt assumption, and beyond

In the main text, we assumed the evolution operator E to be Hilbert-Schmidt, which immediately guarantees the well-posedness of the proposed loss function (7). Lemma 1, indeed, implies that for any Hilbert–Schmidt E it holds

$$\varepsilon(\varphi, P) = \|\mathsf{E} - \sum_{i,j} \varphi_i \otimes P_{ij}\varphi_j\|_{\mathsf{HS}}^2 < \infty.$$

The Hilbert-Schmidt assumption is valid in a broad class of stochastic systems, particularly when the transition kernel exhibits smoothing properties. In atomistic simulations, for instance, the presence of a finite temperature results in a Gaussian smoothing that makes E Hilbert-Schmidt. As an illustrative example, consider the overdamped Langevin dynamics

$$X_{t+1} = X_t - \nabla V(X_t)\,\Delta t + \mathcal{N}(0,\sigma)\sqrt{\Delta t},$$

where V is a potential function, and $\mathcal{N}(0,\sigma)$ denotes an isotropic Gaussian with mean 0 and variance σ proportional to the system's temperature. Assuming that the data is sampled from the equilibrium distribution $\pi(x)dx \propto e^{-\beta V(x)}dx$, we compute

$$\begin{aligned}\|\mathsf{E}\|_{\mathsf{HS}}^2 &= \sum_i \|\mathsf{E}e_i\|_2^2 = \sum_i \int \left| \int p(y \mid x)\, e_i(y)\, \pi(dy)\right|^2 \pi(dx) \\ &= \int p(y \mid x)^2\, \pi(y)\pi(x)\, dy\, dx \propto \int \left| \exp\Big(- \frac{\|y - \nabla V(x)\|}{2\sigma^2}\Big)\right|^2 \pi(y)\pi(x)\, dy\, dx < \infty\end{aligned}$$

where e_i are elements of an orthonormal basis of $L^2(\pi)$, and the third equality follows from Parseval's identity.

The Hilbert-Schmidt assumption, however, is violated in important deterministic dynamical systems, such as those governed by fluid dynamics equations Mezić (2005). Remarkably, the empirical loss (8) still admits a precise operator-theoretic interpretation when E is merely a bounded operator. Indeed, let $\mathsf{P}_\varphi : L^2(\mu) \to L^2(\mu)$ denote the orthogonal projector onto the subspace spanned by the encoder φ. By definition, $\mathsf{P}_\varphi = \Phi_\mu^\dagger \Phi_\mu$, where Φ_μ is as in Lemma 1. With a slight abuse of the notation in (7), we define the abstract loss

$$\varepsilon(\varphi, P) = -2\,\mathrm{Tr}[\mathsf{E}^*\Phi_\nu^* P \Phi_\mu] + \|\sum_{i,j} \varphi_i \otimes P_{ij}\varphi_j\|_{\mathsf{HS}}^2,$$

whose empirical estimator exactly coincides with (8), the loss which we *actually* optimized in our experiments. Now, since φ spans a finite-dimensional subspace, both P_φ and $\mathsf{E}\mathsf{P}_\varphi$ are finite rank, hence Hilbert–Schmidt. In particular, $\mathsf{E}\mathsf{P}_\varphi$ is the restriction of E to the subspace generated by φ. An immediate calculation shows that

$$\varepsilon(\varphi, P) = \|\mathsf{E}\mathsf{P}_\varphi - \sum_{i,j} \varphi_i \otimes P_{ij}\varphi_j\|_{\mathsf{HS}}^2 - \|\mathsf{E}\mathsf{P}_\varphi\|_{\mathsf{HS}}^2, \tag{11}$$

where besides basic algebraic manipulations the result is obtained using the ciclicity of the trace, and the relation $\Phi_\mu \mathsf{P}_\varphi = \Phi_\mu(\Phi_\mu^\dagger \Phi_\mu) = \Phi_\mu$.

The first term on the right-hand side of (11) is familiar, and represents the error incurred by the model $\sum_{i,j} \varphi_i \otimes P_{ij}\varphi_j$ in approximating the restriction of E to the subspace spanned by φ. This error can be linked to the least-squares approach discussed in Sec. 2, see, for example Korda & Mezić (2018, Theorem 1). Minimizing it with respect to φ leads to representation collapse, since $\varphi(x) = 0$ for all x trivially minimizes it. Our learning objective (11), instead, avoids collapse through the second term, $-\|\mathsf{E}\mathsf{P}_\varphi\|_{\mathsf{HS}}^2$, which can be interpreted as follows. Without loss of generality, write $\mathsf{P}_\varphi = \sum_{i=1}^d e_i \otimes e_i$, where e_i form an orthonormal basis of $\mathrm{span}(\varphi_1, \ldots, \varphi_d)$. By definition of Hilbert-Schmidt norm one has

$$\|\mathsf{E}\mathsf{P}_\varphi\|_{\mathsf{HS}}^2 = \sum_{i=1}^d \|\mathsf{E}e_i\|_2^2 = \sum_{i=1}^d \mathbb{E}_{x\sim\nu}\big[(\mathsf{E}e_i)(x)^2\big].$$

Now notice that interpreting e_i as a probe we have at our disposal to observe the system[7], the term $(\mathsf{E}e_i)(x)$ quantifies the "dynamical response" read by our probe, given that the system was prepared

[7] Functions of the state of the systems are commonly referred to as *observables*, too.

Table 1: Forecasting errors and training times for the Lorenz '63 example (20 independent runs). Note that for LinLS and KRR is reported the total fitting time while for the other methods the epoch time is reported. Best results are highlighted in bold.

	Ours	LinLS	KRR	VAMPNets	DPNets	DAE	CAE
RMSE ($\times 10^{-2}$)	**0.49±0.24**	1.29±0.00	2.10±0.00	0.78±0.12	0.58±0.11	0.77±0.12	2.58±0.19
MAE ($\times 10^{-2}$)	**0.32±0.24**	0.84±0.00	1.27±0.00	0.46±0.08	0.36±0.08	0.55±0.08	1.95±0.14
Time (ms)	181.1±40.1	**.4±.1**	$(25.3\pm0.2)10^3$	165.5±10.8	190.7±41.5	166.8±9.10	408.5±41.9

to be in state x. The quantity $\|\mathsf{E}\mathsf{P}_\varphi\|^2_{\mathsf{HS}}$, therefore, measures the average strength of such responses, implying that the second term in the loss promotes encoders φ whose span captures observables with the highest possible dynamical variability. To close the discussion, we highlight that for Hilbert-Schmidt operators, the observables with the highest dynamical response are precisely the leading singular functions, and the loss function (7) is indeed minimized when φ spans the leading singular space of the evolution operator E, see (Kostic et al., 2024, Theorem 1).

B Experimental Details

The experiments have been performed on the following hardware:

- 1 Node with 32 cores Ice Lake at 2.60 GHz, 4 × NVIDIA Ampere A100 GPUs, 64 GB and 512 GB RAM.
- 1 Node with 20 cores Xeon Silver 4210 at 2.20 GHz, 4 × NVIDIA Tesla V100 GPUs, 16 GB and 384 GB RAM.
- A workstation equipped with a i7-5930K CPU at 3.50 GHz, 2 × NVIDIA GeForce GTX TITAN X GPUs, 12 GB and 32 GB of RAM.

B.1 Additional Experiment: Lorenz '63

We evaluated our method on the Lorenz '63 system (Lorenz, 1963), a classical example of a chaotic dynamical system governed by three coupled ordinary differential equations. To validate the performance of our approach, we tested it on a one-step-ahead forecasting task, and we analyzed the learned dynamical modes. Because of the low-dimensionality of the state x_t, we appended it as a non-learnable feature of the encoder $\varphi(x_t) = [\mathrm{MLP}(x_t), x_t]$ to ensure that the forecasting target–the state itself–lies in the linear space of functions spanned by φ by design. The learnable part of the encoder consisted of a small multi-layer perceptron (MLP).

In Tab. 1, we compare the performance of the estimator E_φ from (3), with an encoder φ trained according to Alg. 1, against several baseline models. These include Linear Least Squares (LinLS), Kernel Ridge Regression (Kostic et al., 2022) (KRR) with a Gaussian kernel, VAMPNets (Mardt et al., 2018), DPNets (Kostic et al., 2023b), Dynamic Autoencoder (Lusch et al., 2018) (DAE), and Consistent Autoencoder (Azencot et al., 2020) (CAE). To ensure a fair comparison, we matched the encoder architecture for VAMPNets, DPNets, DAE and CAE, while decoders of DAE and CAE were defined as MLPs symmetric to their respective encoders. For KRR, the rank was set equal to the latent dimensionality used in the deep learning models.

The results on the forecasting task demonstrate that, although our model is not specifically designed for prediction, it achieves the best performance among all considered methods. Finally, we verified that the leading eigenfunctions obtained by our approach correctly identify coherent sets on the stable attractor (see Fig. 4).

Training details. We generated a single long trajectory of 15,000 time steps using the `kooplearn` 1.1.3 implementation of Lorenz '63 dynamical system, with default parameters. To ensure convergence to a system's attractor (Tucker, 1999), we discarded the first 1,000 time steps. Also, to obtain approximately time-independent segments for training, validation and testing, we further discarded 1,000 time steps between each split. In total, 10,000 time steps were used for training, and 1,000 time steps each for validation and testing.

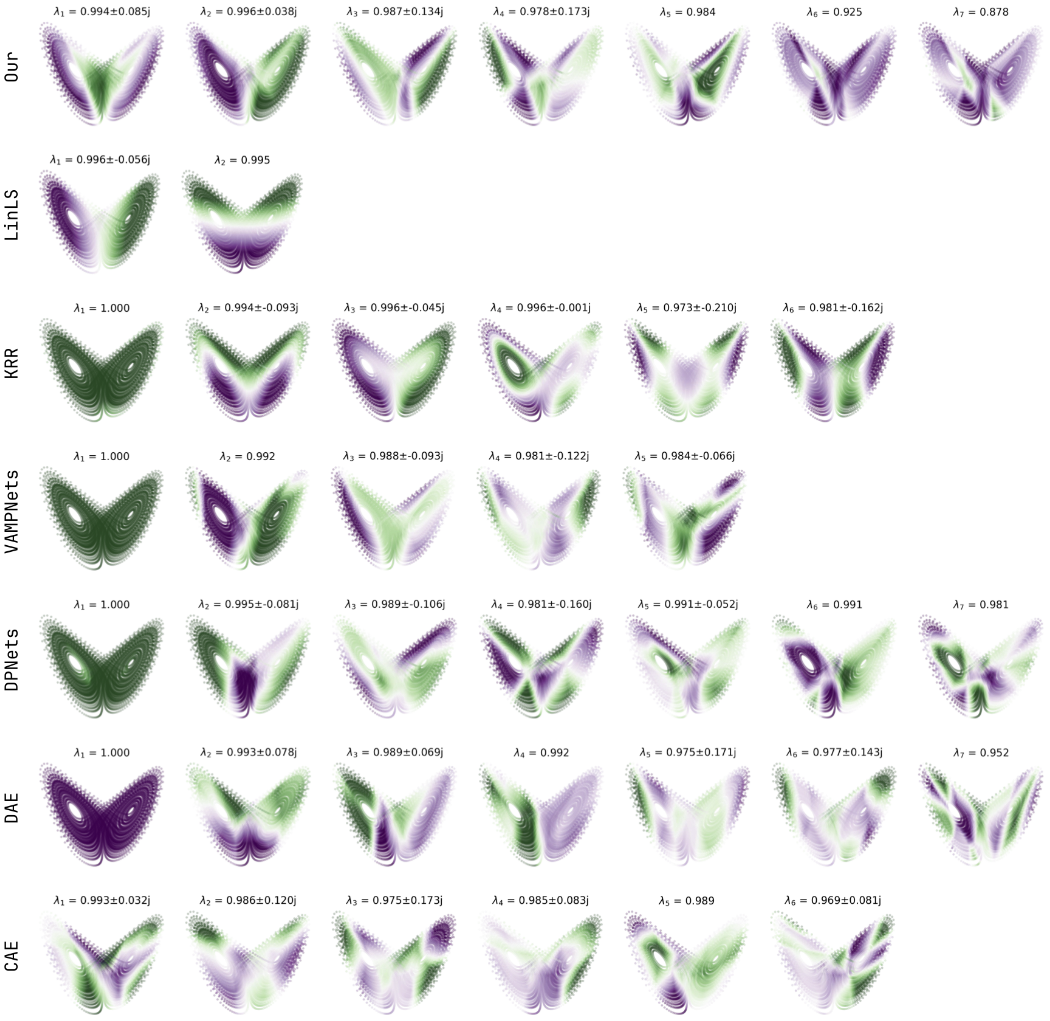


Figure 4: Leading eigenfunctions computed by our and baseline approaches. Each row corresponds to a different method, and each column shows an eigenfunction ordered by decreasing eigenvalue magnitude.

Our encoder consisted of an MLP with an input layer of size 3, two hidden layers with 16 units each, and an 8-dimensional latent space, using ReLU activation functions. The model was trained for 100 epochs using the AdamW optimizer, with an initial learning rate of 10^{-3} decayed to 10^{-4} via a cosine schedule, a batch size of 512, and a lag time of 10 time steps.

Baseline methods. We compared our approach against the following baseline models:

- **Linear Least Squares (LinLS).** A linear regression model trained directly on the raw input features without any nonlinear transformation or latent representation.
- **Kernel Ridge Regression (KRR) (Kostic et al., 2022).** We trained a KRR model with a Gaussian kernel, using the bandwidth estimated via the median heuristic (Garreau et al., 2017). The model was trained with a rank of 8, a Tikhonov regularization parameter of 10^{-6}, and using Arnoldi iterations.
- **VAMPNets (Mardt et al., 2018).** Trained using the same MLP encoder as ours, with the VAMP-2 loss loss and centered covariances.
- **DPNets (Kostic et al., 2023b).** Trained using the same MLP encoder as ours, with the relaxed DP loss and centered covariances.

- **Dynamic Autoencoder (DAE) (Lusch et al., 2018).** Trained with the same MLP encoder architecture as in our approach; the decoder was defined symmetrically. The loss components for reconstruction, prediction, and linear evolution were equally weighted (all set to 1).
- **Consistent Autoencoder (Azencot et al., 2020).** Trained with the same MLP encoder architecture as in our approach; the decoder was defined symmetrically. The CAE loss weights for reconstruction, prediction, backward prediction, linear evolution, and consistency were all set to 1.

For all deep learning-based baselines (VAMPNets, DPNets, DAE, and CAE), models were trained for 100 epochs using a batch size of 512, and a lag time of 10 time steps. VAMPNets and DPNets used the AdamW optimizer with a learning rate of 10^{-4} and 10^{-2}, respectively; DAE and CAE used the Adam optimizer with a learning rate of 10^{-3}. All baselines were implemented using `kooplearn` 1.1.3.

Additional analysis. In Fig. 4, we show the leading eigenfunctions of the transfer operators computed using our method and the baseline approaches. These visualizations highlight qualitative differences in the learned spectral structures, offering insight into the dynamics captured by each method. The leading eigenfunction of KRR, VAMPNets, DPNets, and DAE is constant and associated with the stable attractor. Our method, LinLS, and KRR, find an eigenfunction with eigenvalue $\approx .996$ which clearly separates the two lobes of the attractor.

B.2 Protein folding

Training details. We used data from (Lindorff-Larsen et al., 2011), which can be requested directly to De Shaw Research and are available without charge for academic usage. Our encoder consisted of a SchNet (Schütt et al., 2017) graph neural network with 3 interaction blocks, 16 RBF functions and an hidden dimension of 64. The model was trained with an AdamW optimizer with starting learning rate of 10^{-2} decaying to 10^{-4} with a cosine schedule, using the `mlcolvar` (Bonati et al., 2023) library.

Additional analysis. To understand to what mode is associated the leading eigenfunction Ψ_1, in Fig. 5 we correlated it with two physical quantities associated with the folding, which are the Root-Mean-Square-Deviation (RMSD) and the Radius of Gyration, see Fig. 5. Furthermore, to obtain a finer understanding, we used sparse linear models to approximate the CVs via LASSO regression. This yields a surrogate model which is a linear combination of a few physical descriptors, hence interpretable. To choose the regularization strength, we computed the Mean Square Error of the surrogate model versus the number of features, see Fig. 6.

We performed LASSO regression on a set of contact functions determining the presence of hydrogen bonds. The features selected by this procedure, as well as a snapshot of the protein where these features are highlighted, are reported in Fig. 7. Interestingly, some of the selected features pertain to side-chain interactions, a piece of information that would have been impossible to get using only C_α atoms to train the encoder.

B.3 Ligand binding

Simulations details. We selected a subset of host-guest systems for the SAMPL5 challenge (Bannan et al., 2016; Yin et al., 2017) to evaluate our method's performance, including three ligands (G1, G2, G3) and the octa-acid calixarane host (OAMe). Simulations were run in GROMACS 2024.5 (Abraham et al., 2015) patched with PLUMED 2.9.3 (Tribello et al., 2014). Systems were built using the GAFF (Wang et al., 2004) force field with RESP (Bayly et al., 1993) charges, solvated in a cubic TIP3P (Jorgensen et al., 1983) water box $40.27\,\text{Å}$ length, containing 2100 water molecules. System charge balanced with Na^+ ions. Our timestep is $2\,\text{fs}$ and the temperature is set to $300\,\text{K}$ via a velocity rescale thermostat (Bussi et al., 2007) with a coupling time of $0.1\,\text{ps}$. All simulations aligned the host's vertical axis h with the box axis and centered coordinates on virtual atom V1. All production simulations were initiated from the dissociated state of each ligand. Trajectories were terminated when the ligand fully rebounded into the binding pocket (defined as host-guest distance $h < 6\,\text{Å}$). For each ligand, we performed 10 independent production trajectories, with coordinates saved every 500 steps.

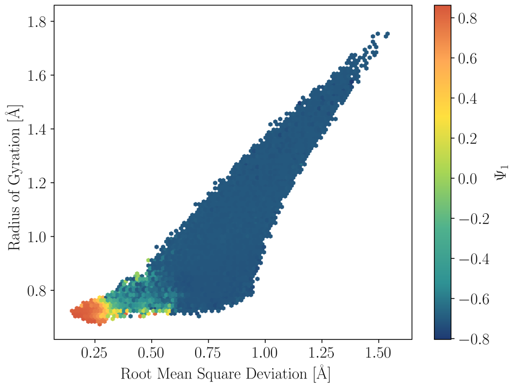


Figure 5: The value of the leading eigenfunction Ψ_1 of the evolution operator is highly correlated with the RMSD and Radius of Gyration of the Trp-cage protein.

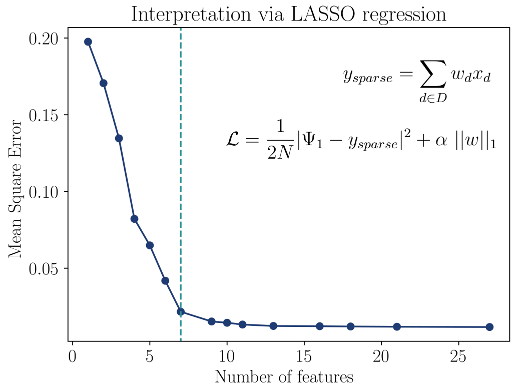


Figure 6: MSE of approximating Ψ_1 by LASSO regression on meaningful physical descriptors. For Trp-cage we constructed a library of hydrogen-bond contact functions. The selected descriptors are reported in Fig. 7

Physical descriptors (H-bonds)	Normalized Coefficient
GLY10-O – SER13-N	0.307
GLY11-O – ARG16-N	0.294
TRP6-O – GLY11-N	0.170
TRP6-NE1s (sidechain) – ARG16-O	0.109
GLN5-O – ASP9-N	0.073
TRP6-NE1s (sidechain) – PRO17-O	0.044
TRP6-NE1s (sidechain) – PRO18-N	0.002

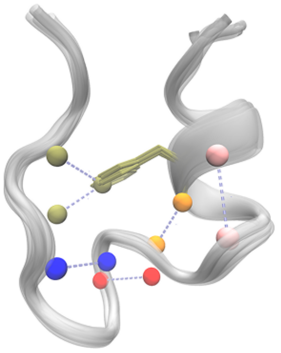


Figure 7: Normalized hydrogen-bond coefficients selected by the LASSO model (left) and representative structural snapshot (right) with the features highlighted.

In our simulations, we applied a funnel restraint (Limongelli et al., 2013) to limit the conformational space explored by the ligand in the unbound state, in turn accelerating the binding process. The parameters are identical to those used in previous studies (Rizzi et al., 2021). We define h as the projection of each ligand along the binding axis, treated as its radial component. For $h \geq 10\,\text{Å}$, the funnel surface is a cylinder with radius $R_{\text{cyl}} = 2\,\text{Å}$ along the vertical axis. For $h < 10\,\text{Å}$, the funnel opens into a conical shape with a 45° angle, defined by $r = 12 - h$. The force acting on a displacement x from the funnel surface is harmonic:

$$F_{\text{funnel}} = -k_F x \quad \text{with} \quad k_F = 20\,\text{kJ mol}^{-1}\,\text{Å}^{-2}$$

An additional harmonic restraint prevents the ligand from escaping too far from the host, enforcing an upper boundary:

$$F_{\text{upper}} = -k_U(h - 18) \quad \text{for} \quad h > 18\,\text{Å}, \quad \text{with} \quad k_U = 40\,\text{kJ mol}^{-1}\,\text{Å}^{-2}$$

The data will be released to ensure the reproducibility of the experiment.

Training details. Our encoder consisted of a SchNet (Schütt et al., 2017) graph neural network with 3 interaction blocks, 16 RBF functions, and a hidden dimension of 64 with an AdamW optimizer with starting learning rate of 10^{-2} decaying to 10^{-4} with a cosine schedule.

Additional analysis. In Fig. 8 we inspect the two leading eigenfunctions of the evolution operator by correlating them with two physical descriptors connected to the binding: the distance along the

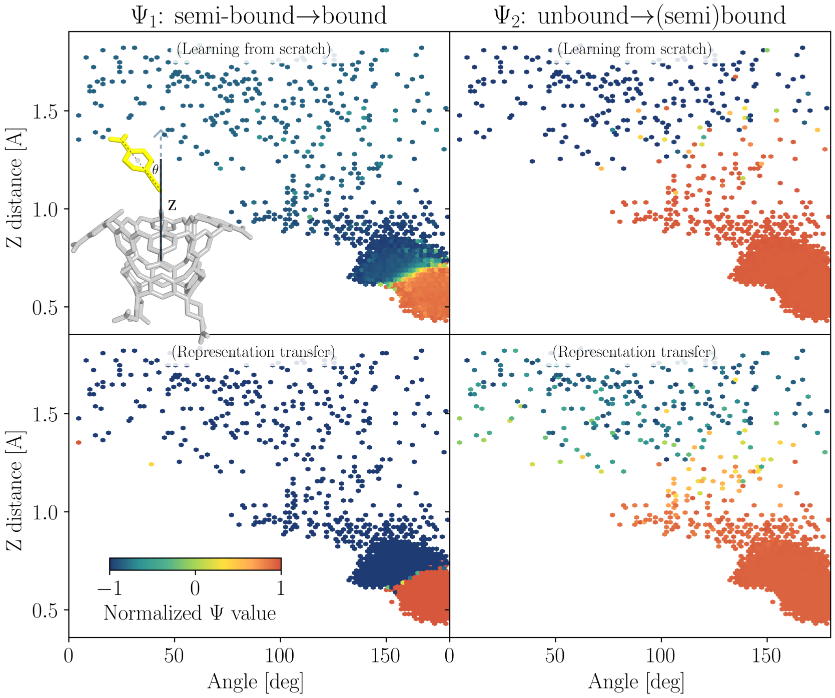


Figure 8: Analysis of the leading eigenfunctions in the space of the host-guest distance z and the ligand orientation θ for Ψ_1 (left) and Ψ_2 (right). The first row contains the results obtained from training from scratch the representation, while the second row contains the case in which it is transferred from other systems.

z direction between the center of mass of the host and the guest and the angle of the ligand with respect to the z axis (see figure in the inset). These results allow us to correlate the Ψ_1 eigenfunction to the transition between the semi-bound pose to the native one, which is due to the presence of trapped water molecules inside the pocket (Rizzi et al., 2021; Bhakat & Söderhjelm, 2017). The second eigenfunction Ψ_2 is instead associated with the binding process. Furthermore, we compared the eigenfunctions obtained by training the representation from scratch on the G2 ligand with the case in which this is transferred from other ligands (G1 and G3), obtaining a remarkable agreement. The ligands G1, G2, and G3 are represented in Fig. 9

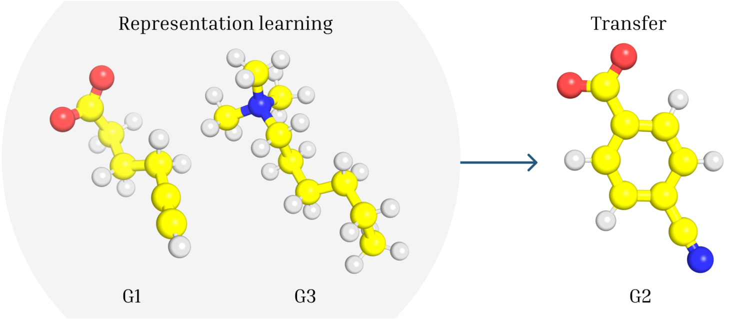


Figure 9: The three different molecules studied in the ligand-binding experiment.

B.4 Climate modeling

Datasets. Following the methodology outlined in (NCP Center), we compute SST* from sea surface temperature (SST) data provided by the ORAS5 reanalysis (Zuo et al., 2019), as made available through the ChaosBench dataset (Nathaniel et al., 2024). The dataset spans a 45-year period

(1979–2023) at a spatial resolution of 1.5°, resulting in a time series of 540 monthly snapshots, each with dimensions 121×240. Data from 1979 to 2016 was used for training, while the 2017–2023 period was reserved for validation.

For the transfer learning task, we employed simulations from the CESM Last Millennium Ensemble project (Otto-Bliesner et al., 2016), spanning the years 850–2006 (files `b.e11.BLMTRC5CN.f19_g16.001.pop.h.SST.085001-089912.nc` to `b.e11.BLMTRC5CN.f19_g16.001.pop.h.SST.185001-200512.nc` available here: `https://gdex.ucar.edu/datasets/d651058/#`). To ensure spatial compatibility between synthetic and observational data, the CESM SST fields were regridded onto the same 1.5° × 1.5° regular latitude–longitude grid of ORAS5 using the `xESMF` (Zhuang et al., 2025) python package.

Training details. Both models trained with CESM and ORAS5 data use a lightweight CNN encoder with four convolutional layers, batch normalization, and max pooling. A masked global average pooling layer, leveraging a binary land–ocean mask, ensures only ocean data contribute to the output representation. The pooled features are mapped through a final linear embedding layer.

For the CESM model, the linear layer P maps to a 128-dimensional latent space. Training included simplicial normalization (Lavoie et al., 2022) (dimension 2), spectral normalization (Miyato et al., 2018) on the linear layer P, gradient clipping (max norm 0.2), a lag time of one month, 100 epochs, AdamW optimizer, and a cosine-decayed learning rate from 10^{-3} to 10^{-5} with a batch size of 64. Leading eigenvalues of the transfer operator are reported in Tab. 2.

Table 2: Leading eigenvalues of the transfer operator learned on ORAS5 data with φ trained on CESM data. Each eigenvalue is expressed in terms of its real (Re), imaginary (Im), and absolute (Abs) components. The associated decorrelation times and oscillation frequencies (in years) are also reported. Eigenvalues are listed in descending order with respect to their absolute value, and those with a decorrelation time shorter than 1/12 years, i.e., the sampling frequency, were discarded.

Idx	Re	Im	Abs	Decor (yr)	Freq (yr)
6	0.92	0.00	0.92	1.01	0.00
4	0.88	0.09	0.89	0.70	5.23
5	0.88	-0.09	0.89	0.70	-5.23
2	0.76	0.40	0.86	0.54	1.09
3	0.76	-0.40	0.86	0.54	-1.09
7	0.85	0.00	0.85	0.50	0.00
8	0.79	0.00	0.79	0.35	0.00
0	0.41	0.67	0.78	0.34	0.52
1	0.41	-0.67	0.78	0.34	-0.52
9	0.74	0.12	0.75	0.29	3.26
10	0.74	-0.12	0.75	0.29	-3.26
11	0.71	0.00	0.71	0.24	0.00
12	0.64	0.00	0.64	0.19	0.00
14	0.60	-0.11	0.61	0.17	-3.01
13	0.60	0.11	0.61	0.17	3.01
15	0.58	0.13	0.59	0.16	2.37
16	0.58	-0.13	0.59	0.16	-2.37
17	0.58	0.03	0.58	0.15	9.78
18	0.58	-0.03	0.58	0.15	-9.78
19	0.52	0.12	0.53	0.13	2.34
20	0.52	-0.12	0.53	0.13	-2.34
21	0.47	0.15	0.49	0.12	1.68
22	0.47	-0.15	0.49	0.12	-1.68
23	0.47	0.03	0.47	0.11	9.27
24	0.47	-0.03	0.47	0.11	-9.27
31	0.39	0.00	0.39	0.09	0.00
27	0.35	0.16	0.38	0.09	1.19
28	0.35	-0.16	0.38	0.09	-1.19

For the ORAS5 model, the linear layer P maps to a 256-dimensional latent space. Training details were otherwise identical, except a 12-month input history was used.

The hyperparameters reported above were selected via grid search; Tab. 3 summarizes the ranges explored.

Comparisons. We further compared our method to VAMPNets (Mardt et al., 2018), DPNets (Kostic et al., 2023b), Linear Least Squares (LinLS), and Kernel Ridge Regression (KRR) with a Gaussian kernel. For the deep-learning methods, we used identical training parameters across models. For the classical approaches applied to the raw inputs, we selected the best model via a grid search over regularization strengths $\alpha \in [10^{-7}, 10^{-3}]$ and, for KRR, kernel coefficients $\gamma \in [10^{-5}, 10^{-2}]$. We also varied the estimator rank in the set $\{5, 8, 10, 16, 32, 50, 64, 128\}$ to assess if low-rank approximations in the raw space could recover the dynamics. As shown in Tab. 6 and Fig. 13, our method outperforms both baselines on the evaluated tasks.

Table 3: Hyperparameter ranges explored during grid search for the climate modeling task.

Hyperparameter	Search Range
Latent dimensions	`[32, 64, 128, ..., 1024]`
Max gradient clipping norm	`[None, 0.1, 0.2, 0.5]`
Normalization of linear layer	`[False, True]`
Regularization	`[0, 1e-5, ..., 1e-2]`
Simplicial normalization dimensions	`[0, 2, ..., 16]`
History length	`[0, 1, 2, 3, 6, 12]`

B.5 Ablations

In our first set of ablations, we investigated the dependence of our self-supervised scheme on the encoder's architecture. Specifically, we studied the scaling of the loss function with respect to (i) the latent dimension and (ii) the overall parameter count of the encoder.

Scaling laws: Graph-NN encoder. We retrained the SchNet architecture Schütt et al. (2017) on the data from the protein folding experiment Sec. 4.1 for three different sizes of the encoder, summarized in Tab. 4, and values of the latent dimension from 4 to 256. The results of this comprehensive ablation study are reported in Fig. 10. We observed monotonically improving losses with respect to both an increasing number of training dimensions (panel A) and an increasing model size (panel B). This result provides robust confirmation of the good scalability properties of the loss function (8) studied in this work. As a test-time metric, we evaluated the eigenvalue residuals, as defined in (Colbrook et al., 2023, Algorithm 1), see panel C of Fig. 10. This metric assesses the extent to which the eigenvalues obtained from our model satisfy the eigenvalue equation $\mathsf{E}g = \lambda g$. The leading eigenvalue λ_1 is the one enjoying the overall best approximation. Larger architectures are associated with smaller residuals across all the leading eigenvalues.

Table 4: Architectural configuration of the three SchNet model sizes used in the ablation study.

Model	Layers	Filters	Hidden Channels	Params
SchNet-S	2	16	32	6,480
SchNet-M	3	32	64	33,088
SchNet-L	3	64	128	125,536

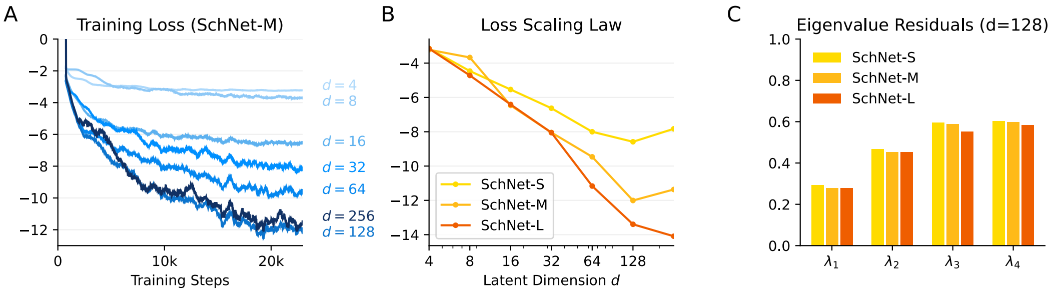


Figure 10: Scaling laws for the protein folding experiment in 4.1. **A** Training loss dynamics as a function of the number of latent dimensions d. **B** Final training loss for three different model sizes, as a function of the number of latent dimensions. **C** Eigenvalue residuals (lower is better), defined in (Colbrook et al., 2023, Algorithm 1) for three different model sizes.

Scaling laws: CNN encoder. The same set of ablations for the climate experiment Sec. 4.3 with a convolutional NN encoder, are reported in Fig. 11. The overall qualitative behavior exactly matches what was already observed for the Graph NN encoder: increasing latent dimensions and/or model size (see Tab. 5) is associated with higher performance. To rule out the possibility that these improvements are linked to overfitting, in Fig. 11, we report the validation loss, instead of the training loss of Fig. 10.

Obtaining the same qualitative results across such distinct physical domains provides strong empirical evidence for the generality of the self-supervised method we propose.

Table 5: Architectural configuration of the three CNN model sizes used in the ablation study.

Model	Layers	Hidden Channels	Params
CNN-S	4	[8, 16, 24, 32]	12,888
CNN-M	4	[16, 32, 64, 128]	101,760
CNN-L	4	[32, 64, 128, 256]	397,024

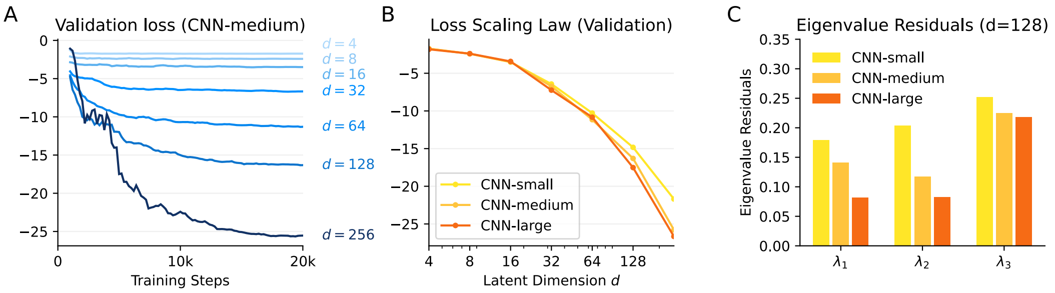


Figure 11: Scaling laws for the climate experiment in 4.3. **A** Validation loss dynamics as a function of the number of latent dimensions d. **B** Final validation loss for three different model sizes, as a function of the number of latent dimensions. **C** Eigenvalue residuals (lower is better), defined in (Colbrook et al., 2023, Algorithm 1) for three different model sizes.

Online versus offline covariances. We conducted an ablation study to assess the effect of using covariances C_X, C_{XY} computed either online during training via EMA or offline from the full training set when estimating the evolution operator E_φ.

In the Lorenz '63 experiment, we trained the models as in the main Lorenz-63 experiment (see Appendix B.1) except for lag time set to 1 to enable a direct comparison between covariance estimation methods. The results show that online covariances yielded better performance, with RMSE and MAE of 0.51 ± 0.11 and 0.30 ± 0.06, respectively, compared to 0.63 ± 0.21 and 0.45 ± 0.19 for offline covariances.

In the climate experiment, the ENSO mode is easily recovered with both approaches. Specifically, for the model trained on ORAS5, the Pearson correlation between the right eigenfunction of E_φ and the ONI was 0.72 with online covariances and 0.71 with offline covariances, indicating comparable performance. The associated eigenvalues were also very similar: $\lambda_{\text{ENSO}} = 0.9531 \pm 0.1206i$ (online) and $\lambda_{\text{ENSO}} = 0.9527 \pm 0.1277i$ (offline).

Stability of EMA covariance. To assess how EMA-based covariances converge toward their offline counterparts, computed via a full-pass over the entire training set, we measured their discrepancy in terms of Frobenius norm, i.e., $\|C_{\text{EMA}} - C_{\text{full-pass}}\|_F$, during training on the Lorenz '63 data. As shown in Fig. 12, this difference peaks in correspondence with the step-like drop in the validation loss, which we interpret as the encoder discovering new representational directions. For a sufficiently large number of epochs, as the network converges and settles into a stable representation, the discrepancy steadily decreases and approaches zero. These observations demonstrate how EMA offers a robust and practical online approximation of the offline, full-pass covariance, offering a clear advantage when dealing with large-scale datasets where computing full-pass covariances may be computationally infeasible.

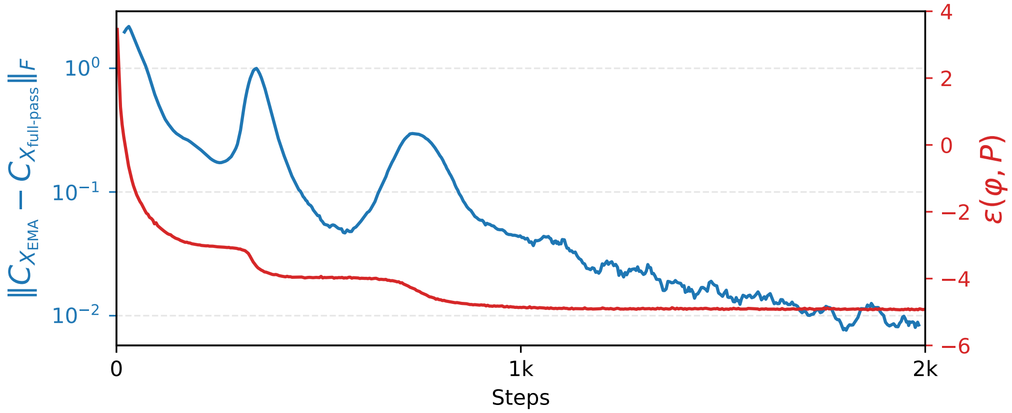


Figure 12: Stability of EMA-based covariance during training on Lorenz '63. The blue curve (left y-axis) shows the discrepancy between EMA and full-pass covariances, while the red curve (right y-axis) shows the validation loss.

Table 6: Performance comparison in terms of Pearson correlation between the right eigenfunction associated with the ENSO mode and ONI, alongside the time per training epoch. Best results are highlighted in bold.

Transfer learning task (model trained on CESM, evaluated on ORAS5).					
	Ours	VAMPNets	DPNets	LinLS	KRR
Pearson correlation (r)	**0.81**	0.56	0.77	N/A	N/A
Time per epoch (s)	**25.27 ± 0.74**	28.44 ± 0.79	29.17 ± 0.79	N/A	N/A
Model trained and evaluated on ORAS5 data.					
	Ours	VAMPNets	DPNets	LinLS	KRR
Pearson correlation (r)	**0.72**	0.56	0.62	0.60	0.63
Time per epoch (s)	**1.91 ± 0.17**	2.03 ± 0.15	2.05 ± 0.15	N/A	N/A

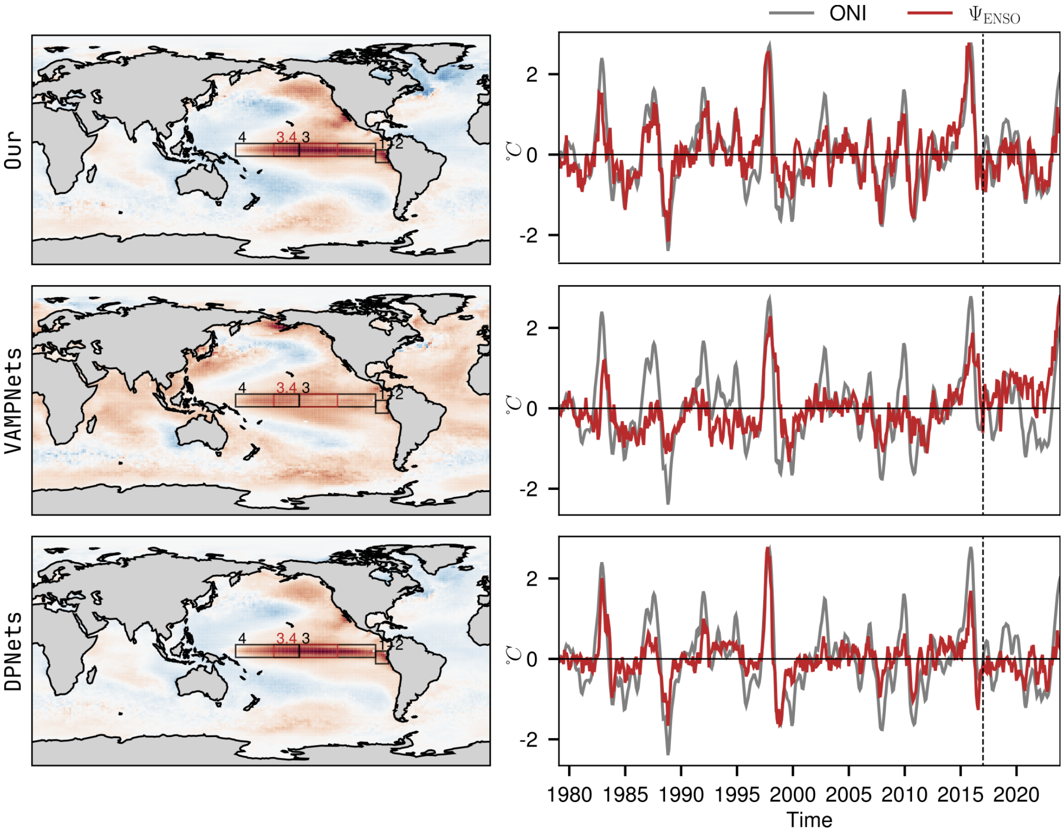


Figure 13: Comparison of ENSO modes retrieved using transfer learning by our method, VAMPNets, and DPNets.