PyKoopman: A Python Package for Data-Driven Approximation of the Koopman Operator

Shaowu Pan [1,3], **Eurika Kaiser** [2], **Brian M. de Silva** [1], **J. Nathan Kutz** [1], **and Steven L. Brunton** [2]

1 Department of Applied Mathematics, University of Washington **2** Department of Mechanical Engineering, University of Washington **3** Department of Mechanical, Aerospace, and Nuclear Engineering, Rensselaer Polytechnic Institute

Summary

PyKoopman is a Python package for the data-driven approximation of the Koopman operator associated with a dynamical systems. The Koopman operator is a principled linear embedding of nonlinear dynamics and facilitates the prediction, estimation, and control of strongly nonlinear dynamics using linear systems theory. In particular, PyKoopman provides tools for data-driven system identification for unforced and actuated systems that build on the equation-free dynamic mode decomposition (DMD) (Schmid, 2010) and its variants (Steven L. Brunton et al., 2022; Kutz et al., 2016; Schmid, 2022). In this work, we provide a brief description of the mathematical underpinnings of the Koopman operator, an overview and demonstration of the features implemented in PyKoopman (with code examples), practical advice for users, and a list of potential extensions to PyKoopman. Software is available at https://github.com/dynamicslab/pyKoopman.

Statement of need

Engineers have long relied on linearization to bridge the gap between simplified, descriptions where powerful analytical tools exist, and the intricate complexities of nonlinear dynamics where analytical solutions are elusive (Ljung, 2010; Wright et al., 1999). Local linearization, implemented via first-order Taylor series approximation, has been widely used in system identification (Ljung, 2010), optimization (Wright et al., 1999), and many other fields to make problems tractable. However, many real-world systems are fundamentally nonlinear and require solutions outside of the local neighborhood where linearization is valid. Rapid progress in machine learning and big data methods are driving advances in the data-driven modeling of such nonlinear systems in science and engineering (Steven L. Brunton & Kutz, 2022).

In the diverse landscape of data-driven modeling approaches, Koopman operator theory has received considerable attention in recent years (Steven L. Brunton et al., 2017; Budišić et al., 2012; Klus et al., 2018; Li et al., 2017; Mezić, 2013; Williams, Kevrekidis, et al., 2015). The main idea is illustrated in Fig. 1. This methodology enables the application of closed-form, convergence-guaranteed methods from linear system theory to general nonlinear dynamics. To fully leverage the potential of data-driven Koopman theory across a diverse range of scientific and engineering disciplines, it is critical to have a central toolkit to automate state-of-the-art Koopman operator algorithms.
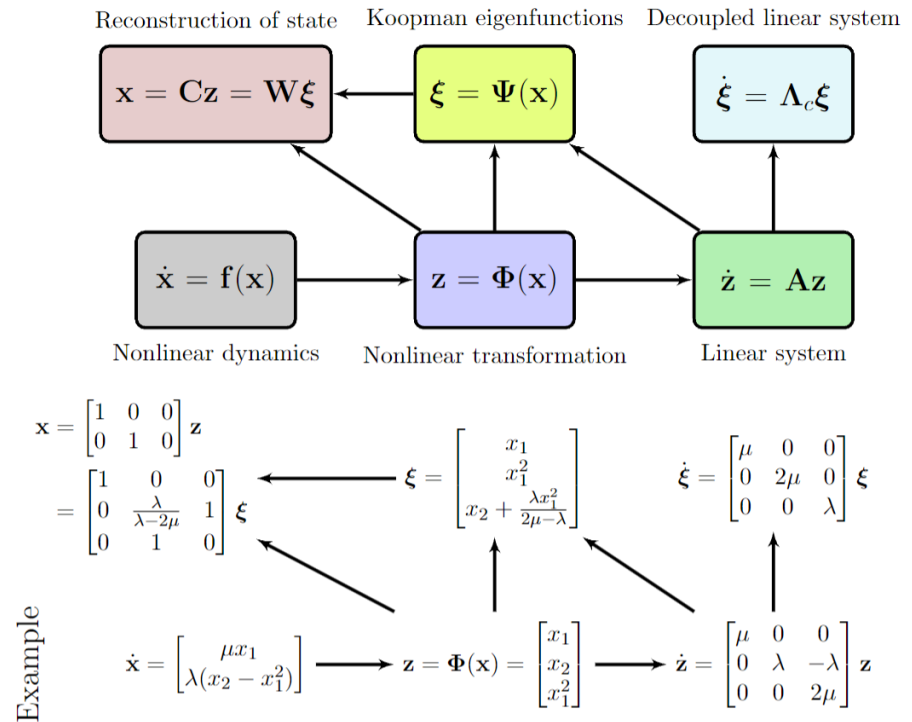
Figure 1: Illustration of Koopman operator for a 2D nonlinear system. This system can be linearized into a 3D linear system with the nonlinear transform $(\mathbf{x})$. In PyKoopman, searching such $(\mathbf{x})$ is facilitated in a data-driven manner.

As a result, PyKoopman is developed as a Python package for approximating the Koopman operator associated with natural and actuated dynamical systems from measurement data. Compared to implementation of DMD (e.g., PyDMD (Demo et al., 2018)) which can be viewed as a linear projection of Koopman operator, PyKoopman offers a comprehensive set of *nonlinear* projection methods. Specifically, PyKoopman offers tools for designing the observables (i.e., functions of the system state) and inferring a finite-dimensional linear operator that governs the dynamic evolution of these observables in time. These steps can either be performed sequentially (Williams, Rowley, et al., 2015; Williams, Kevrekidis, et al., 2015) or combined, as demonstrated in more recent neural network models (Lusch et al., 2018; Mardt et al., 2018; Otto & Rowley, 2019; Takeishi et al., 2017). Besides, we also support data from multiple trajectories. Once a linear embedding is discovered from the data, the linearity of the transformed dynamical system can be leveraged for enhanced interpretability (Pan et al., 2021) or for designing near-optimal observers (Surana & Banaszuk, 2016) or controllers for the original nonlinear system (Kaiser et al., 2021; Korda & Mezić, 2020; Mauroy et al., 2020; Peitz et al., 2020; Peitz & Klus, 2019).
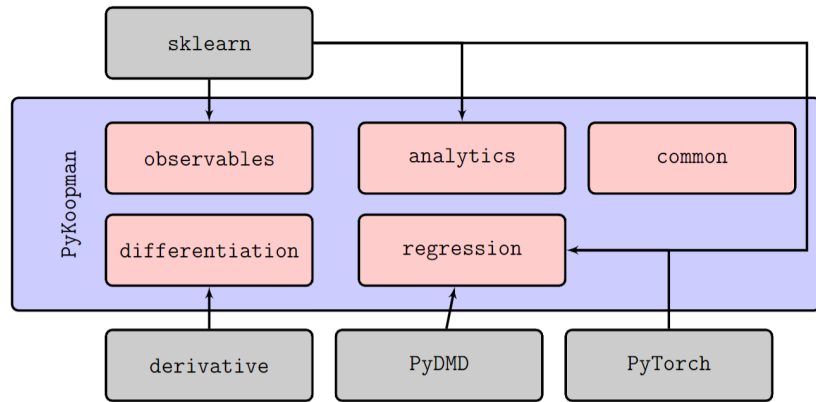
Features

Figure 2: External package dependencies of PyKoopman.

The core component of the PyKoopman package is the Koopman model class. We used several base classes from `sklearn` to build the machine learning pipeline. We used `torch` and `lightning` for implementing deep learning methods for Koopman operator. We also used `pydmd` to incorporate some existing implementation for regression after nonlinear observables are chosen. Finally, we used `derivative` to obtain time derivative to deal with non-uniformly sampled data. To summarize, the external package dependencies are depicted in Fig. 2.

As illustrated in Fig. 3, PyKoopman is designed to lift nonlinear dynamics into a linear system with linear actuation. Specifically, our PyKoopman implementation involves two major steps:

Figure 3: Broad categorization of model types that can be identified with current PyKoopman. While the dotted parts (marked with "·") can be simultaneously discovered within the framework, they are typically ignored for control purposes.

- `observables`: the nonlinear observables used to lift $\mathbf{x}$ to $\mathbf{z}$, and reconstruct $\mathbf{x}$ from $\mathbf{z}$;
- `regression`: the regression used to find the optimal $\mathbf{A}$.

Additionally, we have a `differentiation` module that evaluates the time derivative from a trajectory and the `analytics` module for sparsifying arbitrary approximations of the Koopman operator.

At the time of writing, we have the following features implemented:

- Observable library for lifting the state $\mathbf{x}$ into the observable space
    - Identity (for DMD/DMDc or in case users want to compute observables themselves): Identity

- Multivariate polynomials: `Polynomial` (Williams, Kevrekidis, et al., 2015)
- Time delay coordinates: `TimeDelay` (Steven L. Brunton et al., 2017; Mezić & Banaszuk, 2004)
- Radial basis functions: `RadialBasisFunctions` (Williams, Kevrekidis, et al., 2015)
- Random Fourier features: `RandomFourierFeatures` (DeGennaro & Urban, 2019)
- Custom library (defined by user-supplied functions): `CustomObservables`
- Concatenation of observables: `ConcatObservables`

- System identification method for performing regression

  - Dynamic mode decomposition (Rowley et al., 2009; Schmid, 2010): `PyDMDRegressor`
  - Dynamic mode decomposition with control (Proctor et al., 2016): `DMDc`
  - Extended dynamic mode decomposition (Williams, Kevrekidis, et al., 2015): `EDMD`
  - Extended dynamic mode decomposition with control (Korda & Mezić, 2020): `EDMDc`
  - Kernel dynamic mode decomposition (Williams, Rowley, et al., 2015): `KDMD`
  - Hankel Alternative View of Koopman Analysis (Steven L. Brunton et al., 2016): `HAVOK`
  - Neural Network DMD (Lusch et al., 2018; Otto & Rowley, 2019; Pan & Duraisamy, 2020): `NNDMD`

- Sparse construction of Koopman invariant subspace

  - Multi-task learning based on linearity consistency (Pan et al., 2021): `ModesSelectionPAD21`

- Numerical differentiation for computing $\dot{\mathbf{X}}$ from $\mathbf{X}$

  - Finite difference: `FiniteDifference`
  - 4th order central finite difference: `Derivative(kind="finite_difference")`
  - Savitzky-Golay with cubic polynomials: `Derivative(kind="savitzky-golay")`
  - Spectral derivative: `Derivative(kind="spectral")`
  - Spline derivative: `Derivative(kind="spline")`
  - Regularized total variation derivative: `Derivative(kind="trend_filtered")`

- Common toy dynamics

  - Discrete-time random, stable, linear state-space model: `drss`
  - Van del Pol oscillator: `vdp_osc`
  - Lorenz system: `lorenz`
  - Two-dimensional linear dynamics: `Linear2Ddynamics`
  - Linear dynamics on a torus: `torus_dynamics`
  - Forced Duffing Oscillator: `forced_duffing`
  - Cubic-quintic Ginzburg-Landau equation: `cqgle`
  - Kuramoto-Sivashinsky equation:`ks`
  - Nonlinear Schrodinger equation: `nls`
  - Viscous Burgers equation: `vbe`

- Validation routines for consistency checks

Example

The PyKoopman GitHub repository provides several helpful Jupyter notebook tutorials. Here, we briefly demonstrate a typical workflow using the PyKoopman package to approximate Koopman operator of a 2D nonlinear system.

First, consider the dynamical system

$$\dot{x}_1 = -0.05x_1$$
$$\dot{x}_2 = -x_2 + x_1^2.$$

Given X and Xnext matrices are two one-step away trajectories of the above nonlinear system, we can choose polynomial observables and EDMD regressor to learn the Koopman operator by feeding X and Xnext to the .fit method of an instance of pykoopman.Koopman class.

```python
from pykoopman import Koopman
from pykoopman.observables import Polynomial
from pykoopman.regression import EDMD

model = Koopman(observables=Polynomial(2),regressor=EDMD())
model.fit(X,Xnext)
```

Once the Koopman object model has been fit, we can use the model.simulate method to make predictions over an arbitrary time horizon. Fig. 4 displays the excellent agreement between ground truth and the EDMD prediction from the aforementioned 'Koopman" model on randomly generated unseen test data.
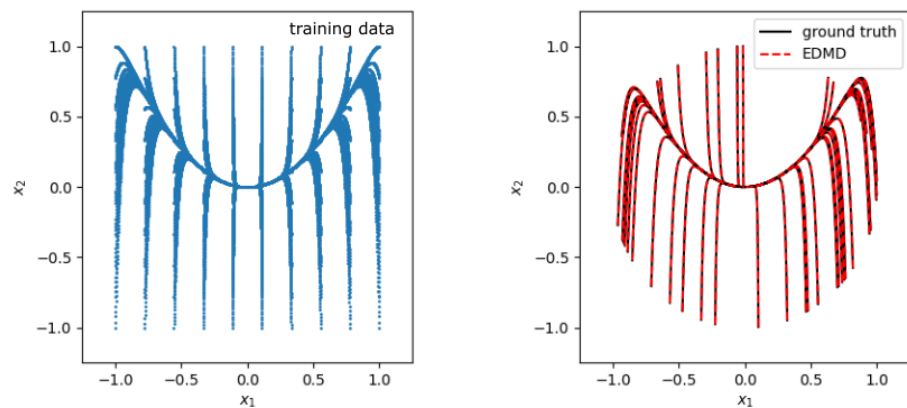
Figure 4: Example of learning Koopman operator for a 2D nonlinear system. Left: distribution of training data consists of multiple trajectories. Right: prediction on unseen test trajectories.

Conclusion

Our goal of the PyKoopman package is to provide a central hub for education, application and research development of learning algorithms for Koopman operator. The PyKoopman package is aimed at researchers and practitioners alike, enabling anyone with access to discover linear embeddings of nonlinear systems from data. Following PySINDy (Silva et al., 2020) and Deeptime (Hoffmann et al., 2021), PyKoopman is designed to be accessible to users with basic knowledge of linear systems, adhering to scikit-learn standards, while also being modular for more advanced users. We hope that researchers and practioners will use PyKoopman as a platform for algorithms developement and applications of linear embedding.

Acknowledgments

References

Brunton, Steven L., Brunton, B. W., Proctor, J. L., Kaiser, E., & Kutz, J. N. (2017). Chaos as an intermittently forced linear system. *Nature Communications*, *8*(1), 1–9. https://doi.org/10.1038/s41467-017-00030-8

Brunton, Steven L., Budišić, M., Kaiser, E., & Kutz, J. N. (2022). Modern Koopman theory for dynamical systems. *SIAM Review*, *64*(2), 229–340. https://doi.org/10.1137/21M1401243

Brunton, Steven L., & Kutz, J. N. (2022). *Data-driven science and engineering: Machine learning, dynamical systems, and control.* Cambridge University Press. https://doi.org/10.1017/9781108380690

Brunton, Steven L., Proctor, J. L., & Kutz, J. N. (2016). Discovering governing equations from data by sparse identification of nonlinear dynamical systems. *Proceedings of the National Academy of Sciences*, *113*(15), 3932–3937. https://doi.org/10.1073/pnas.1517384113

Budišić, M., Mohr, R., & Mezić, I. (2012). Applied Koopmanism. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, *22*(4), 047510. https://doi.org/10.1063/1.4772195

DeGennaro, A. M., & Urban, N. M. (2019). Scalable extended dynamic mode decomposition using random kernel approximation. *SIAM Journal on Scientific Computing*, *41*(3), A1482–A1499. https://doi.org/10.1137/17M115414X

Demo, N., Tezzele, M., & Rozza, G. (2018). PyDMD: Python dynamic mode decomposition. *Journal of Open Source Software*, *3*(22), 530.

Hoffmann, M., Scherer, M., Hempel, T., Mardt, A., Silva, B. de, Husic, B. E., Klus, S., Wu, H., Kutz, N., Brunton, S. L., & others. (2021). Deeptime: A python library for machine learning dynamical models from time series data. *Machine Learning: Science and Technology*, *3*(1), 015009. https://doi.org/10.1088/2632-2153/ac3de0

Kaiser, E., Kutz, J. N., & Brunton, S. L. (2021). Data-driven discovery of Koopman eigenfunctions for control. *Machine Learning: Science and Technology*, *2*(3), 035023. https://doi.org/10.1088/2632-2153/abf0f5

Klus, S., Nüske, F., Koltai, P., Wu, H., Kevrekidis, I., Schütte, C., & Noé, F. (2018). Data-driven model reduction and transfer operator approximation. *Journal of Nonlinear Science*, *28*(3), 985–1010. https://doi.org/10.1007/s00332-017-9437-7

Korda, M., & Mezić, I. (2020). Optimal construction of Koopman eigenfunctions for prediction and control. *IEEE Transactions on Automatic Control*, *65*(12), 5114–5129. https://doi.org/10.1109/TAC.2020.2978039

Kutz, J. N., Brunton, S. L., Brunton, B. W., & Proctor, J. L. (2016). *Dynamic mode decomposition: Data-driven modeling of complex systems.* SIAM. https://doi.org/10.1137/1.9781611974508

Li, Q., Dietrich, F., Bollt, E. M., & Kevrekidis, I. G. (2017). 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), 103–111. https://doi.org/10.1063/1.4993854

Ljung, L. (2010). Perspectives on system identification. *Annual Reviews in Control*, *34*(1), 1–12. https://doi.org/10.1016/j.arcontrol.2009.12.001

Lusch, B., Kutz, J. N., & Brunton, S. L. (2018). Deep learning for universal linear embeddings of nonlinear dynamics. *Nature Communications*, *9*(1), 4950. https://doi.org/10.1038/s41467-018-07210-0

Mardt, A., Pasquali, L., Wu, H., & Noé, F. (2018). VAMPnets for deep learning of molecular kinetics. *Nature Communications*, *9*(1), 1–11. https://doi.org/10.1038/s41467-017-02388-1

182 Mauroy, A., Susuki, Y., & Mezić, I. (2020). *Koopman operator in systems and control.*
183 Springer. https://doi.org/10.1007/978-3-030-35713-9

184 Mezić, I. (2013). Analysis of fluid flows via spectral properties of the Koopman oper-
185 ator. *Annual Review of Fluid Mechanics*, *45*, 357–378. https://doi.org/10.1146/
186 annurev-fluid-011212-140652

187 Mezić, I., & Banaszuk, A. (2004). Comparison of systems with complex behavior. *Physica D:*
188 *Nonlinear Phenomena*, *197*(1-2), 101–133. https://doi.org/10.1016/j.physd.2004.06.015

189 Otto, S. E., & Rowley, C. W. (2019). Linearly recurrent autoencoder networks for learning
190 dynamics. *SIAM Journal on Applied Dynamical Systems*, *18*(1), 558–593. https://doi.org/
191 10.1137/18M1177846

192 Pan, S., Arnold-Medabalimi, N., & Duraisamy, K. (2021). Sparsity-promoting algorithms for
193 the discovery of informative Koopman-invariant subspaces. *Journal of Fluid Mechanics*,
194 *917*, A18. https://doi.org/10.1017/jfm.2021.271

195 Pan, S., & Duraisamy, K. (2020). Physics-informed probabilistic learning of linear embeddings
196 of nonlinear dynamics with guaranteed stability. *SIAM Journal on Applied Dynamical*
197 *Systems*, *19*(1), 480–509. https://doi.org/10.1137/19M1267246

198 Peitz, S., & Klus, S. (2019). Koopman operator-based model reduction for switched-system
199 control of PDEs. *Automatica*, *106*, 184–191. https://doi.org/10.1016/j.automatica.2019.
200 05.016

201 Peitz, S., Otto, S. E., & Rowley, C. W. (2020). Data-driven model predictive control using
202 interpolated koopman generators. *SIAM Journal on Applied Dynamical Systems*, *19*(3),
203 2162–2193. https://doi.org/10.1137/20M1325678

204 Proctor, J. L., Brunton, S. L., & Kutz, J. N. (2016). Dynamic mode decomposition with
205 control. *SIAM Journal on Applied Dynamical Systems*, *15*(1), 142–161. https://doi.org/
206 10.1137/15M1013857

207 Rowley, C. W., Mezić, I., Bagheri, S., Schlatter, P., & Henningson, D. S. (2009). Spectral
208 analysis of nonlinear flows. *Journal of Fluid Mechanics*, *641*, 115–127. https://doi.org/10.
209 1017/S0022112009992059

210 Schmid, P. J. (2010). Dynamic mode decomposition of numerical and experimental data.
211 *Journal of Fluid Mechanics*, *656*, 5–28. https://doi.org/10.1017/S0022112010001217

212 Schmid, P. J. (2022). Dynamic mode decomposition and its variants. *Annual Review of Fluid*
213 *Mechanics*, *54*, 225–254. https://doi.org/10.1146/annurev-fluid-030121-015835

214 Silva, B. M. de, Champion, K., Quade, M., Loiseau, J.-C., Kutz, J. N., & Brunton, S. L.
215 (2020). PySINDy: A python package for the sparse identification of nonlinear dynamical
216 systems from data. *Journal of Open Source Software*, *5*(49), 2104. https://doi.org/10.
217 21105/joss.02104

218 Surana, A., & Banaszuk, A. (2016). Linear observer synthesis for nonlinear systems using
219 Koopman operator framework. *IFAC-PapersOnLine*, *49*(18), 716–723. https://doi.org/10.
220 1016/j.ifacol.2016.10.250

221 Takeishi, N., Kawahara, Y., & Yairi, T. (2017). Learning koopman invariant subspaces
222 for dynamic mode decomposition. *Advances in Neural Information Processing Systems*,
223 1130–1140. https://doi.org/10.48550/arXiv.1710.04340

224 Williams, M. O., Kevrekidis, I. G., & Rowley, C. W. (2015). A data–driven approximation of
225 the Koopman operator: Extending dynamic mode decomposition. *Journal of Nonlinear*
226 *Science*, *25*(6), 1307–1346. https://doi.org/10.1007/s00332-015-9258-5

227 Williams, M. O., Rowley, C. W., & Kevrekidis, I. G. (2015). A kernel approach to data-
228 driven Koopman spectral analysis. *Journal of Computational Dynamics*, *2*, 247–265.
229 https://doi.org/10.3934/jcd.2015005

230 Wright, S., Nocedal, J., & others. (1999). Numerical optimization. *Springer Science*,
231 *35*(67-68), 7. https://doi.org/10.1007/978-0-387-40065-5