The Journal of Open Source Software

datafold: data-driven models for point clouds and time series on manifolds

Daniel Lehmberg[1, 2], **Felix Dietrich**[2], **Gerta Köster**[1], **and Hans-Joachim Bungartz**[2]

1 Munich University of Applied Sciences, Lothstr. 64, 80333 Munich, Germany **2** Technical University of Munich, Boltzmannstr. 3, 85747 Garching, Germany

DOI: 10.21105/joss.02283

Software

- Review
- Repository
- Archive

Editor: Dan Foreman-Mackey

Reviewers:

- @jsgalan
- @mtezzele

Submitted: 20 May 2020
Published: 14 July 2020

Summary

Ever increasing data availability has changed the way how data is analyzed and interpreted in many scientific fields. While the underlying complex systems remain the same, data measurements increase in both quantity and dimension. The main drivers are larger computer simulation capabilities and increasingly versatile sensors. In contrast to an equation-driven workflow, a scientist can use data-driven models to analyze a wider range of systems, including those with unknown or intractable equations. The models can be applied to a variety of data-driven scenarios, such as enriching the analysis of unknown systems or merely serve as an equation-free surrogate by providing fast, albeit approximate, responses to unseen data.

However, expanding datasets create challenges throughout the analysis workflow from extracting and processing to interpreting the data. This includes the fact that new data does not always provide completely new and uncorrelated information to existing data. One way to extract the essential information is to understand and parametrize the intrinsic data geometry. An intrinsic geometry is what most data-driven models assume implicitly or explicitly in the available data, and successful machine learning algorithms adapt to this underlying structure for tasks like regression or classification (e.g., Bishop, 2006). This geometry is often of much lower dimension than the ambient data space, and finding a suitable set of coordinates can reduce the complexity of the dataset. We refer to this geometric structure encoded in the data as a "manifold". In mathematical terms, a manifold is a topological space that is locally homeomorphic to Euclidean space. Typically, manifold learning attempts to construct a global parametrization (embedding) of this manifold, in a space of much lower dimension than the original ambient space. The well-known manifold hypothesis states that such manifolds underlie many observations and processes, including time-dependent systems.

datafold is a Python package that provides **data**-driven models for point clouds to find an *explicit* mani-**fold** parametrization and to identify non-linear dynamical systems on these manifolds. The explicit data manifold treatment allows prior knowledge of a system and its problem-specific domain to be included. This can be the proximity between points in the dataset (Coifman & Lafon, 2006a) or functions defined on the phase space manifold of a dynamical system, such as (partially) known governing equation terms (Brunton, Proctor, & Kutz, 2016; Williams, Kevrekidis, & Rowley, 2015).

datafold is open-source software with a design that reflects a workflow hierarchy: from low-level data structures and algorithms to high-level meta-models intended to solve complex machine learning tasks. The key benefit of *datafold* is that it accommodates and integrates models on the different workflow levels. Each model has been investigated and tested individually and found to be useful by the scientific community. In *datafold* these models can be used in a single processing pipeline. Our integrated workflow facilitates the application of

data-driven analysis and thus has the potential to boost widespread utilization. The implemented models are integrated into a software architecture with a clear modularization and an API that is templated from the `scikit-learn` project, which can be used as part of its processing pipeline (Pedregosa et al., 2011). The data structures are subclasses from common objects of the Python scientific computing stack, allowing models to generalize for both static point clouds and temporally ordered time series collection data. The software design and modularity in *datafold* reflects two requirements: high flexibility to test model configurations, and openness to new model implementations with clear and well-defined scope. We want to support active research in data-driven analysis with manifold context and thus target students, researchers and experienced practitioners from different fields of dataset analysis.

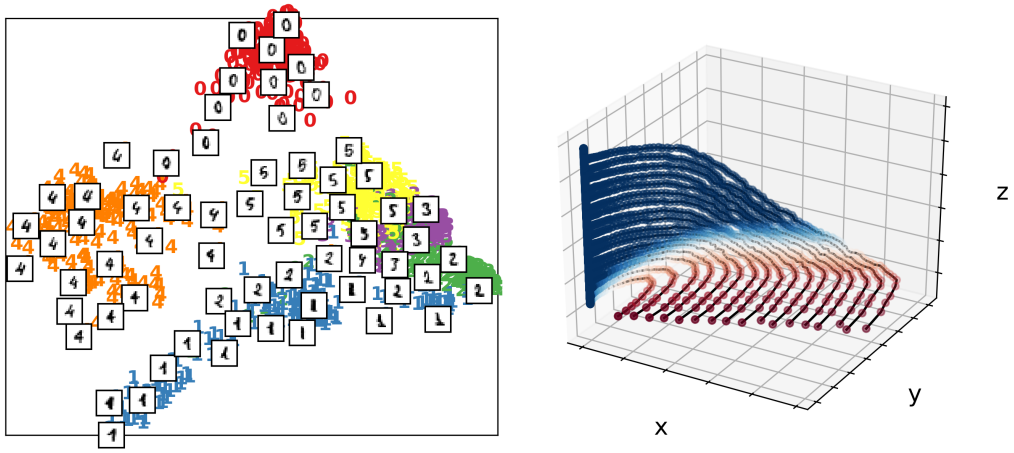

Figure 1: (Left) Point cloud of embedded handwritten digits between 0 and 5 with the "Diffusion Map" model. Each point originally has 64 dimensions where each dimension represents a pixel of an 8 x 8 image. (Right) Conceptual illustration of a three-dimensional time series forming a phase space with geometrical structure. The time series start on the `(x,y)` plane and end on the z-axis.

1. Point cloud data

High-dimensional and unordered point clouds are often directly connected to the "manifold assumption", i.e. that the data lies close to a lower-dimensional manifold. Our software aims to find a low-dimensional parametrization (embedding) of this manifold. In a machine learning context, this is also referred to as "non-linear unsupervised learning" or shorter "manifold learning". Often the models are endowed with a kernel which encodes the proximity between data to preserve local structures. Examples are the general "Kernel Principal Component Analysis" (Bengio et al., 2004), "Local Linear Embedding" (Belkin & Niyogi, 2003), or "Hessian Eigenmaps" (Donoho & Grimes, 2003). A variety of manifold learning models already exist in the `scikit-learn` Python package. In addition to these, *datafold* provides an efficient implementation of the "Diffusion Maps" model (Coifman & Lafon, 2006a). The model includes an optional sparse kernel matrix representation with which the model can scale to larger datasets. In addition to non-linear dimension reduction, "Diffusion Maps" allow the user to approximate mathematically meaningful objects on manifold data, such as the Laplace-Beltrami operator (Coifman & Lafon, 2006a). *datafold* also supplies functionality for follow-up aspects of non-linear manifold learning, such as estimating the kernel scale parameters to describe the locality of points in a dataset and extending the embedding to unseen data. The latter refers to the image and pre-image mapping between the original and latent space (e.g., see analysis in Chiavazzo, Gear, Dsilva, Rabin, & Kevrekidis, 2014). This so-called "out-of-sample" extension interpolates general function values on manifold point clouds and, therefore, has to handle large input data dimensions (Coifman & Lafon, 2006b; Fernández, Rabin, Fishelov, & Dorronsoro, 2020; Rabin & Coifman, 2012). In *datafold*, out-of-sample extensions are imple-

mented efficiently, so that interpolated function values for millions of points can be computed in seconds on a standard desktop computer.

2. Time series data

A special kind of point cloud type targeted by *datafold* are time series and collections thereof. In this case, a data-driven model can fit and generalize the underlying dynamics to perform prediction or regression. Usually, the phase space of the dynamical system, underlying the time series observations, is assumed to be a manifold (see a conceptual illustration in Figure 1). *datafold* focuses on the algorithms "Dynamic Mode Decomposition" (DMD) (Kutz, Brunton, Brunton, & Proctor, 2016; Schmid, 2010; Tu, Rowley, Luchtenburg, Brunton, & Kutz, 2014) and "Extended Dynamic Mode Decomposition" (E-DMD, Williams et al., 2015). DMD linearly decomposes the available time series data into spatio-temporal components, which then define a linear dynamical system. Many DMD based variants address even more general, non-linear underlying dynamical systems. This is usually done by changing the time series coordinates in a step before DMD is applied (Champion, Brunton, & Kutz, 2019; Giannakis, 2019; Le Clainche, Vega, & Soria, 2017; Williams et al., 2015). The justification of this workflow is covered by operator theory and functional analysis, specifically the Koopman operator. In practice, the E-DMD algorithm approximates the Koopman operator with a matrix, based on a finite set of functions evaluated on the available data, the so-called "dictionary". Finding a good choice for the dictionary is comparable to the machine learning task of "model selection" and requires great flexibility in setting up the data processing pipeline. The flexibility of setting an arbitrary dictionary combined with a selection of the provided DMD variants is a core feature of *datafold's* implementation of E-DMD.

Comparison to other software projects

The Python package *statsmodels* (Seabold & Perktold, 2010) includes statistical models for time series analysis, such as the *AutoRegressive Integrated Moving Average* (ARIMA) and variations thereof. These models usually assume an underlying stochastical process and aim to approximate autocorrelations in time series data. Another type of popular data-driven models are deep neural networks; widely adopted packages are the frameworks *tensorflow* (Abadi et al., 2015, 2016) and *PyTorch* (Paszke et al., 2019). Deep learning models generalize well in several problem domains, but common drawbacks are a non-deterministic construction process, the requirement of large datasets, and only limited options to include prior knowledge about the underlying system. For manifold learning, one can use (variational) autoencoders, while recurrent architectures (e.g., LSTM networks Hochreiter & Schmidhuber, 1997) are often used for system identification. Other packages in *datafold*'s scope are the Python packages *PyDMD* (Demo, Tezzele, & Rozza, 2018) and *PySINDy* (Silva et al., 2020). The *PyDMD* project includes numerous variations of DMD, such as the "Higher Order DMD" (Le Clainche et al., 2017). The *PySINDy* software focuses on "Sparse Identification of Non-linear Dynamics" (SINDy) (Brunton et al., 2016). *datafold*'s EDMD implementation allows an arbitrary DMD variant to be used in a processing pipeline to regress the dynamics, which means the models from both *PyDMD* and *PySINDy* could be used in combination with *datafold*.

Acknowledgements

DL is supported by the German Research Foundation (DFG), grant no. KO 5257/3-1 and thanks the research office (FORWIN) of Munich University of Applied Sciences and CeDoSIA of TUM Graduate School at the Technical University of Munich for their support. DL and FD thank Yannis Kevrekidis from the Johns Hopkins University, Baltimore, USA, for his support during a research stay of DL in 2018, which paved the way for the *datafold* software project.

References

Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G. S., et al. (2015). TensorFlow: Large-Scale Machine Learning on Heterogeneous Distributed Systems. Retrieved from https://www.tensorflow.org/

Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J., Devin, M., et al. (2016). TensorFlow: A System for Large-Scale Machine Learning. In *Proceedings of the 12th usenix conference on operating systems design and implementation*, OSDI'16 (pp. 265–283). USA: USENIX Association. ISBN: 9781931971331

Belkin, M., & Niyogi, P. (2003). Laplacian Eigenmaps for Dimensionality Reduction and Data Representation. *Neural Computation*, *15*(6), 1373–1396. doi:10.1162/089976603321780317

Bengio, Y., Delalleau, O., Roux, N. L., Paiement, J.-F., Vincent, P., & Ouimet, M. (2004). Learning Eigenfunctions Links Spectral Embedding and Kernel PCA. *Neural Computation*, *16*(10), 2197–2219. doi:10.1162/0899766041732396

Bishop, C. M. (2006). *Pattern recognition and machine learning*. Information science and statistics. New York: Springer. ISBN: 978-0-387-31073-2

Brunton, S. 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. doi:10.1073/pnas.1517384113

Champion, K. P., Brunton, S. L., & Kutz, J. N. (2019). Discovery of Nonlinear Multiscale Systems: Sampling Strategies and Embeddings. *SIAM Journal on Applied Dynamical Systems*, *18*(1), 312–333. doi:10.1137/18M1188227

Chiavazzo, E., Gear, C., Dsilva, C., Rabin, N., & Kevrekidis, I. (2014). Reduced Models in Chemical Kinetics via Nonlinear Data-Mining. *Processes*, *2*(1), 112–140. doi:10.3390/pr2010112

Coifman, R. R., & Lafon, S. (2006a). Diffusion maps. *Applied and Computational Harmonic Analysis*, *21*(1), 5–30. doi:10.1016/j.acha.2006.04.006

Coifman, R. R., & Lafon, S. (2006b). Geometric harmonics: A novel tool for multiscale out-of-sample extension of empirical functions. *Applied and Computational Harmonic Analysis*, *21*(1), 31–52. doi:10.1016/j.acha.2005.07.005

Demo, N., Tezzele, M., & Rozza, G. (2018). PyDMD: Python Dynamic Mode Decomposition. *The Journal of Open Source Software*, *3*(22), 530. doi:10.21105/joss.00530

Donoho, D. L., & Grimes, C. (2003). Hessian eigenmaps: Locally linear embedding techniques for high-dimensional data. *Proceedings of the National Academy of Sciences*, *100*(10), 5591–5596. doi:10.1073/pnas.1031596100

Fernández, Á., Rabin, N., Fishelov, D., & Dorronsoro, J. R. (2020). Auto-adaptive multi-scale Laplacian Pyramids for modeling non-uniform data. *Engineering Applications of Artificial Intelligence*, *93*, 103682. doi:10.1016/j.engappai.2020.103682

Giannakis, D. (2019). Data-driven spectral decomposition and forecasting of ergodic dynamical systems. *Applied and Computational Harmonic Analysis*, *47*(2), 338–396. doi:10.1016/j.acha.2017.09.001

Hochreiter, S., & Schmidhuber, J. (1997). Long Short-Term Memory. *Neural Computation*, *9*(8), 1735–1780. doi:10.1162/neco.1997.9.8.1735

Kutz, J. N., Brunton, S. L., Brunton, B. W., & Proctor, J. L. (2016). *Dynamic mode decomposition. Data-Driven modelling of complex systems*. Society for Industrial; Applied Mathematics. ISBN: 978-1-61197-450-8

Le Clainche, S., Vega, J. M., & Soria, J. (2017). Higher order dynamic mode decomposition of noisy experimental data: The flow structure of a zero-net-mass-flux jet. *Experimental Thermal and Fluid Science*, *88*, 336–353. doi:10.1016/j.expthermflusci.2017.06.011

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., et al. (2019). PyTorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, & R. Garnett (Eds.), *Advances in neural information processing systems 32* (pp. 8026–8037). Curran Associates, Inc. Retrieved from http://papers.nips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., et al. (2011). Scikit-learn: Machine Learning in Python. *Machine Learning in Python*, 6.

Rabin, N., & Coifman, R. R. (2012). Heterogeneous datasets representation and learning using diffusion maps and Laplacian pyramids. In *Proceedings of the 2012 SIAM International Conference on Data Mining* (pp. 189–199). Society for Industrial; Applied Mathematics. doi:10.1137/1.9781611972825.17

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

Seabold, S., & Perktold, J. (2010). Statsmodels: Econometric and Statistical Modeling with Python. In (pp. 92–96). Austin, Texas. doi:10.25080/Majora-92bf1922-011

Silva, B. de, Champion, K., Quade, M., Loiseau, J.-C., Kutz, J., & Brunton, S. (2020). PySINDy: A Python package for the sparse identification of nonlinear dynamical systems from data. *Journal of Open Source Software*, *5*(49), 2104. doi:10.21105/joss.02104

Tu, J. H., Rowley, C. W., Luchtenburg, D. M., Brunton, S. L., & Kutz, J. N. (2014). On Dynamic Mode Decomposition: Theory and Applications. *Journal of Computational Dynamics*, *1*(2), 391–421. doi:10.3934/jcd.2014.1.391

Williams, M. O., Kevrekidis, I. G., & Rowley, C. W. (2015). A DataDriven Approximation of the Koopman Operator: Extending Dynamic Mode Decomposition. *Journal of Nonlinear Science*, *25*(6), 1307–1346. doi:10.1007/s00332-015-9258-5