ROBUST SYMBOLIC REGRESSION FOR NETWORK TRAJECTORY INFERENCE

Ramzi Dakhmouche
Institute of Mathematics, EPFL, Switzerland
Computational Engineering Lab, Empa, Switzerland
ramzi.dakhmouche@epfl.ch

Ivan Lunati
Computational Engineering Lab, Empa, Switzerland
ivan.lunati@empa.ch

Hossein Gorji
Computational Engineering Lab, Empa, Switzerland
mohammadhossein.gorji@empa.ch

ABSTRACT

Real-world complex systems often miss high-fidelity physical descriptions and are typically subject to partial observability. Learning dynamics of such systems is a challenging and ubiquitous problem, encountered in diverse critical applications which require interpretability and qualitative guarantees. Our paper addresses this problem in the case of probability distribution flows governed by ODEs. Specifically, we devise a *white box* approach -dubbed Symbolic Distribution Flow Learner (SDFL)- combining symbolic search with a Wasserstein-based loss function, resulting in a robust model-recovery scheme which naturally lends itself to cope with partial observability. Additionally, we furnish the proposed framework with theoretical guarantees on the number of required *snapshots* to achieve a certain level of fidelity in the model-discovery. We illustrate the performance of the proposed scheme on the prototypical problem of Kuramoto networks and a standard benchmark of single-cell RNA sequence trajectory data. The numerical experiments demonstrate the competitive performance of SDFL in comparison to the state-of-the-art.

1 INTRODUCTION

Complex systems often emerge as large-scale networks of physical and societal interactions, with examples from epidemics to consensus dynamics, and from power grids to biological organisms. Despite their omnipresence, they often defy high-fidelity and fine-grain mathematical descriptions. The pursuit of accurate symbolic expressions that describe the evolution of such dynamical systems, is thus of paramount importance in many areas of science and engineering. Indeed, such parsimonious model descriptions offer several advantages, including compactness, explicit interpretations and high-fidelity generalization. Unlike *black-box* approaches, they give explicit insights on the underlying processes. Most notably, they allow for safety related qualitative guarantees such as asymptotic stability, which are crucial for critical applications. Hence, devising robust and efficient learning methods to uncover explicit equations directly from data -also known as symbolic regression- is of great interest.

Symbolic regression has been extensively studied over recent years in both algebraic and differential equation discovery contexts. Common approaches include: sparse regression Brunton et al. (2016); Rudy et al. (2017); Chen et al. (2021); Kubalík et al. (2023), sequence-to-sequence deep neural network modelling

Becker et al. (2022); Vastl et al. (2022); Biggio et al. (2021), and symbolic search-based formulations Cornforth & Lipson (2012); Gaucel et al. (2014); Cazenave (2013); Lu et al. (2021); Sun et al. (2023). For a more extensive related works review, see Appendix E. One important setting, which has not been studied much though, is the discovery of dynamical systems governing probability distribution flows using only a few sampled screen shots, which is particularly relevant for many applications ranging from epidemics modelling to cellular evolution prediction Bunne et al. (2022). Consequently, we propose to address this question, in the case of network flows, by designing a suitable equation recovery framework and illustrating its numerical performance on both synthetic and real-world data. This setup of symbolic learning of probability flows in networks entails several challenges, which we address by the following contributions:

- We combine the neural ODE framework Chen et al. (2018) with a symbolic search approach - respecting the permutation invariance- resulting in a *white-box* model for *trajectory inference*.
- We devise a suitable loss function that leverages the robustness properties of the Wasserstein distance Villani (2009), while taking into account the limited observability of the system under study.
- We back our devised algorithm with suitable sample complexity theoretical results.
- We demonstrate the performance of the proposed approach on the prototypical problem of Kuramoto networks and a standard benchmark of single-cell population trajectory data.

The rest of the paper is organized as follows. In section 2, we introduce the notions upon which the proposed algorithm is based and motivate our design choices. In section 3, we outline the main contributions of this work and provide the theoretical results that guided the design. In sections 4, we present numerical evaluations, followed by concluding remarks in section 5.

2 Problem Formulation and Background

2.1 General Setup

Fix $T > 0$ as a positive time horizon, and consider the random variable $\mathbf{x}_0 \in \mathbb{R}^d$ to be the initial condition of the state variable $\mathbf{x}_t$ which evolves according to the ODE system

$$\dot{\boldsymbol{y}} \quad = \quad \boldsymbol{f}(\boldsymbol{y}) \tag{1}$$

in time $t \in [0, T]$. In our setting, $\boldsymbol{f}$ encodes the interactions between different components of $\mathbf{x}_t$ over time, which are distributed according a known and fixed network topology $\mathcal{G}$. For $n \geq 2$ measurement times $\{t_0, t_1, \ldots, t_n\}$, assume $m \geq 1$ samples of each of the distributions $\mu_{t_0}, \mu_{t_1}, \ldots, \mu_{t_n}$ are observed, resulting in *snapshots* represented by the corresponding empirical probability measures $\{\hat{\mu}_{t_1,m}, ..., \hat{\mu}_{t_n,m}\}$. The question addressed in this paper is that of the recovery of the function $\boldsymbol{f}$ defining the ODE that governs the given dynamics. That is, *trajectory inference* Hashimoto et al. (2016); Tong et al. (2020); Bunne et al. (2022); Huguet et al. (2022), through an explicit closed form ODE. For that matter, the solution of the ODE can be seen as the image of $\boldsymbol{f}$ by the operator given by $\boldsymbol{F}(\boldsymbol{f}) : (t, \boldsymbol{x}_0) \mapsto \boldsymbol{F}^t(\boldsymbol{f})(\boldsymbol{x}_0)$, where $t \mapsto \boldsymbol{F}^t(\boldsymbol{f})(\boldsymbol{x}_0)$ is the solution of the Cauchy problem:

$$\begin{cases} \dot{\boldsymbol{y}} = \boldsymbol{f}(\boldsymbol{y}) \\ \boldsymbol{y}(0) = \boldsymbol{x}_0 \end{cases} \tag{2}$$

for a given initial condition $\boldsymbol{x}_0 \in \mathbb{R}^d$. Note that, the problem of recovering $\boldsymbol{f}$ from $(\hat{\mu}_{t_i,m})_{i \in \{1,\ldots,n\}}$ is *ill-posed* in general, and further assumptions are needed. Such assumptions should represent prior information that can act as a form of regularization rendering the learning task at hand more feasible. In our case, the required prior information comes from the fact that the estimator of $\boldsymbol{f}$ is constructed by combining analytic expressions from a fixed pre-selected set. We make the assumption that the observation instants fulfill $\forall i \in \{0, \ldots, n\}$, $t_i \in (iT/n, (i+1)T/n)$. A straight-forward extension, though, can be obtained for

uniformly sampled time instants $(t_i)_{i\in\{0,\dots,n\}}$ based on a Quasi-Monte Carlo scheme convergence argument Niederreiter (1978). To translate this setting to an optimization framework, first, we review some concepts about distances in probability spaces upon which a suitable goodness-of-fit measure is proposed.

2.2 WASSERSTEIN GUIDANCE

Considering probability distribution flow modelling, the 2-Wasserstein distance $\mathcal{W}_2$ represents a natural choice for the design of a robust algorithm (see figure 3). Recall that, given two measure spaces $(\mathcal{X}, \mu)$ and $(\mathcal{Y}, \nu)$ and denoting by $\Pi(\mu, \nu)$ the set of their couplings, the $\mathcal{W}_2$ distance reads

$$\mathcal{W}_2(\mu,\nu) \quad = \quad \min_{\pi\in\Pi(\mu,\nu)} \int_{\mathcal{X}\times\mathcal{Y}} \|\alpha-\beta\|_2^2 \,\mathrm{d}\pi(\alpha,\beta)\ , \tag{3}$$

where $\| \,.\, \|_2$ is the Euclidean norm. If we had access to $(\mu_t)_{t\in[0,T]}$, the goodness-of-fit of a candidate estimator $\hat{\boldsymbol{f}}$ could be suitably defined as:

$$L(\hat{\boldsymbol{f}}) = \int_0^T \mathcal{W}_2(\boldsymbol{F}^t(\hat{\boldsymbol{f}})_{\#}\mu_0,\, \mu_t)\,\mathrm{d}t\ , \tag{4}$$

where $\boldsymbol{F}^t(\hat{\boldsymbol{f}})_{\#}\mu_0$ is the push-forward of the initial probability measure by the partial flow map $\boldsymbol{x}_0 \mapsto F^t(\hat{f})(\boldsymbol{x}_0)$. In other words, $L(\hat{\boldsymbol{f}})$ is the time aggregated Wasserstein distance between the measure resulting from the inferred trajectory and the one resulting from the ground-truth dynamics. However, since $(\mu_t)_t$ is only partially known, we work with an approximation based on the observed *snapshots*, as discussed in the following section.

3 TECHNICAL APPROACH

To minimize the introduced loss function, given by equation 4, we replace the neural net estimator in the neural ODE framework of Chen et al. (2018); Kidger (2022) with a search for an explicit analytic formula for $\boldsymbol{f}$. This is achieved through a symbolic search algorithm instantiated by Monte-Carlo Tree Search (MCTS), see for e.g. Sun et al. (2023). The devised workflow -of inferring the network dynamics from given snapshots- is illustrated in Fig. 1. From a guarantee perspective, in addition to a bound on the approximation error of $L(\hat{f})$ by its empirical estimate $\hat{L}_{m,n}(\hat{f})$ for a given $\hat{f}$, we quatify the sample complexity of MCTS, using regret bound-based results from bandit theory Lattimore & Szepesvári (2020).

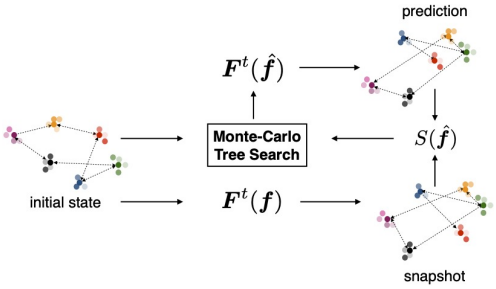

Figure 1: The devised model-discovery setup

3.1 DISCRETE LOSS FUNCTION

The convergence of the discrete version of the loss function, defined for continuously differentiable $\hat{\boldsymbol{f}} : \mathbb{R}^d \to \mathbb{R}^d$ by

$$\hat{L}_{m,n}(\hat{\boldsymbol{f}}) = \frac{1}{n}\sum_{i=1}^{n} \mathcal{W}_2(\boldsymbol{F}^{t_i}(\hat{\boldsymbol{f}})_{\#}\hat{\mu}_{t_0,m}, \hat{\mu}_{t_i,m}),$$

to the continuous one i.e. L, is controlled by the number of snapshots and the size of the snapshot sample set. We build upon results of Fournier & Guillin (2015); Bonnans & Shapiro (2013) about *finite-sample* rates of convergence of the empirical measure in the Wasserstein space, and regularity of the Wasserstein distance to obtain the following theorem, where the proof is postponed to Appendix A.

Theorem 1.
Let $(\mu_t)_{t\geq 0}$ have a compact support and $\hat{\boldsymbol{f}} : \mathbb{R}^d \longrightarrow \mathbb{R}^d$ be a differentiable function. Assume that $t \mapsto \mu_t$ is differentiable. Then, for all $m, n \geq 2$,

$$\mathbb{E}\left|\frac{1}{n}\sum_{i=1}^{n}\mathcal{W}_2(\boldsymbol{F}^{t_i}(\hat{\boldsymbol{f}})_{\#}\hat{\mu}_{t_0,m}, \hat{\mu}_{t_i,m}) \;-\; \frac{1}{T}\int_0^T \mathcal{W}_2(\boldsymbol{F}^{t}(\hat{\boldsymbol{f}})_{\#}\mu_0,\, \mu_t)\,\mathrm{d}t\right| = O\left(\frac{1}{m^{(d/2+1)}} + \frac{1}{n}\right)\,,$$

where $\hat{\mu}_{t_i,m}$, for $i \in \{1, \dots, n\}$, denotes the empirical measure corresponding to the m observed realizations of each of the probability distributions $(\mu_{t_i})_{1\leq i\leq n}$.

3.2 Symbolic flow discovery algorithm

Given the discrete loss $\hat{L}_{m,n}$, the model-recovery algorithm is based on computing predicted snapshots corresponding to a candidate estimate $\hat{\boldsymbol{f}}$ and comparing them to the observed ones. The predictions will be determined by solving the obtained ODE, i.e. equation 1 where $\boldsymbol{f}$ is replaced by $\hat{\boldsymbol{f}}$, through a numerical integration scheme, such as Runge–Kutta solvers. In parallel, the optimization over candidate estimates is realized using a MCTS-based symbolic search. For that matter, the goal of MCTS will be to maximize a score function defined for a differentiable $\hat{\boldsymbol{f}} : \mathbb{R}^d \longrightarrow \mathbb{R}^d$ by $S(\hat{\boldsymbol{f}}) = 1/(1 + \hat{L}_{m,n}(\hat{\boldsymbol{f}}))$. To ensure permutation invariance in the obtained expressions, which is characteristic of network systems, we restrict the search space by applying a permutation invariant aggregation operation (e.g. *sum*) to each expression before each evaluation. We report an ablation study corresponding to the removal of the permutation invariance module in Appendix D.2. The pseudo-code of the algorithm is summarized in *Algorithm*1, with more details in Appendix F.

Algorithm 1 Symbolic Distribution Flow Learner

Inputs: Number of episodes N, number of roll-outs H, screen-shots $(\hat{\mu}_{t_i,m})_{i,m}$ at $(t_i)_{0\leq i\leq n}$
Initialization: Estimate the value of each operation $(+, -, \times, \sin, \dots)$ as a root node ;
for $e = 1, \dots, N$ **do**:
  Randomly select a root node and build an expression tree as follows:
  if Tree is complete **then**
    Evaluate the corresponding estimate $\hat{\boldsymbol{f}}$ by computing $S(\hat{\boldsymbol{f}})$, Back-propagate the obtained value ;
  else
    Run H roll-outs, Store the best estimate, Back-propagate the corresponding value ;
    Select the operation a maximizing $UCT(\boldsymbol{s}, a)$ where $\boldsymbol{s}$ is the current state of the tree ;
  end if
end for
Return: Most accurate $\hat{\boldsymbol{f}}$ over the N episodes ;

3.3 Monte-Carlo Tree Search sample complexity

A crucial question in MCTS-based algorithms is to estimate the number of required episodes, executed by the algorithm, in order to honor a certain error tolerance in the obtained solution of the target optimization

problem. We apply a non-asymptotic error analysis result by Shah et al. (2020) to determine the minimal number of episodes -also known as sample complexity- that ought to be used (see Appendix B for the proof).

Proposition 1.
The average number of score evaluations E required for the MCTS algorithm[1] to find an ε-optimal[2] solution where $\varepsilon > 0$, is at most given by:

$$E = O\left(q \cdot \varepsilon^{-(4+M)} \cdot (\log \frac{1}{\varepsilon})^5\right) ,$$

where M is the maximum allowed expression length and q the size of the chosen elementary function set.

4 Numerical Experiments

First, we illustrate the performance of the proposed algorithm on the Kuramoto network system of ODEs, used across the biological, chemical and electrical domains to model circadian oscillators, pacemaker cells in the heart and electrical power networks among other applications Discacciati & Hesthaven (2021); Dörfler & Bullo (2014). Then, we conduct an evaluation on a real-world dataset of embryoid stem cell trajectories Moon et al. (2019). We provide comparisons of our algorithm with the current trajectory inference state-of-the-art algorithms, namely `TrajectoryNet` Tong et al. (2020) and `JKOnet` Bunne et al. (2022). `TrajectoryNet` relies on a (neural net-based) continuous normalizing flow Grathwohl et al. (2018) formulation, augmented with relevant regularizations such as growth rate and velocity penalization. `JKOnet` builds upon the celebrated JKO scheme Jordan et al. (1998) describing energy gradient flows, where it parameterizes the energy function and the Monge potential using Input Convex Neural Networks Amos et al. (2017). In addition to being based on black-box models -in this instance, neural nets- these approaches contrast with `SDFL` in that they require (extra)-hyper-parameter tuning. We conduct experiments in the small and noisy data regime with varying Training Sample Sizes (TSS) per snapshot, illustrating the suitability of `SDFL` for practical costly data-collection conditions. Similar to previous studies, numerical evaluations are conducted based on the Wasserstein distance between the predicted empirical distributions and the ground-truth distributions (averaged over 3 runs). We report the experimental details in Appendix C and a computational time comparison in Appendix D.1.

4.1 Kuramoto system of ODEs

We investigate the recovery of the Kuramoto system with the state variable $\boldsymbol{x}_t = (\theta_i(t))_{1 \leq i \leq d}$ following

$$\dot{\theta}_i(t) = \omega_i + \frac{1}{d} \sum_{j=1}^{d} K_{ij} \sin\left(\theta_j(t) - \theta_i(t)\right) ,$$

where $(\omega_i)_{1 \leq i \leq d}$ are the corresponding natural frequencies. Gaussian initial condition (with mean 2 and unity variance) is employed. Moreover $\mathcal{G} = (K_{ij})_{1 \leq i,j \leq d}$ is the graph weight matrix. For simplicity, following Discacciati & Hesthaven (2021), we assume a fully connected uniformly weighted graph (i.e. we take $K_{ij} = K$ for all $i, j \in \{1, \ldots, d\}$). We consider $n = 15$ snapshots, in

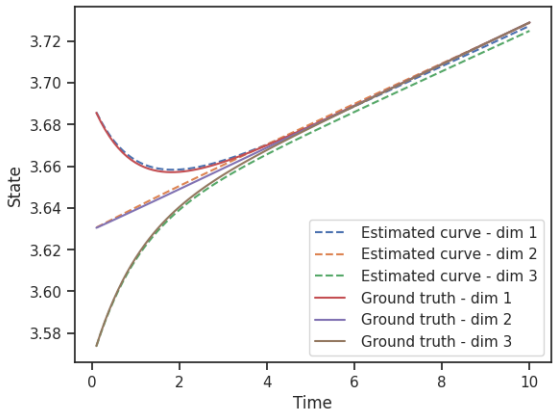

Figure 2: Inferred and ground truth state trajectory for the Kuramoto model

[1]With UCT defined as in Shah et al. (2020).

[2]An ε-optimal solution of an optimzation problem $\min_{x \in E} g(x)$ is a value $x_\varepsilon \in E$ that satisfies $g(x_\varepsilon) \leq \min_{x \in E} g(x) + \varepsilon$.

$d = 3$ dimensions, over the time horizon $T = 30$. Furthermore the natural frequencies are set to 0.01. To highlight the accuracy and robustness of our approach, Figure 2 illustrates the inferred trajectory of state variable (using SDFL). Note that to reduce the SDFL running time, it was run in two steps where the global function structure is fitted first, followed by the estimation of regression coefficients (see Appendix C for details). The results reported in table 1 suggest SDFL is competitive with the state-of-the-art, while the performance for the different methods improves with increasing sample size.

Training sample size	SDFL	JKONet	TrajectoryNet
50	0.57 ± 0.04	0.86 ± 0.22	4.33 ± 0.13
100	0.57 ± 0.06	0.76 ± 0.18	4.83 ± 0.10
150	0.46 ± 0.05	0.76 ± 0.22	3.15 ± 0.11

Table 1: Prediction error comparison in the Wasserstein metric for the Kuramoto model

4.2 Single-Cell Population Dynamics

We consider the problem of learning the evolution of embryonic stem cells based on single-cell RNA sequencing data measurements over a period of 27 days, where the data was collected at 5 different snapshots (see Moon et al. (2019)). Apart from the high cost of obtaining a large dataset, a key difficulty is that a cell is (usually) destroyed during a measurement Bunne et al. (2022). Hence, there is a need for schemes which predict the distribution evolution across time, rather than individual trajectories, from limited observations. The dataset was pre-processed using the recent dimensionality reduction technique PHATE Moon et al. (2019), which is specifically designed to preserve maximum variablity in the low-dimensional space while targeting intuitive visualization, and therefore interpretation. Similar to the previous case of Kuramoto network, SDFL outperforms the benchmarks, confirming the relevance of SDFL as a dynamic regression tool for real-world data subject to noise and limited observability. It should be emphasized that neural-net based approaches (TrajectoryNet and JKONet) require sufficiently large amounts of data to reach their optimal performance. As for JKONet, one potential explanation for why it performs better than TrajectoryNet, is that it enforces an inductive bias through the JKO scheme Bunne et al. (2022).

Training sample size	SDFL	JKONet	TrajectoryNet
100	0.97 ± 0.14	3.18 ± 0.54	6.78 ± 0.26
200	0.87 ± 0.11	3.72 ± 0.51	6.65 ± 0.06
300	0.79 ± 0.16	3.03 ± 0.70	6.25 ± 0.11

Table 2: Prediction error comparison in the Wasserstein metric for embryoid stem cell trajectory

5 Conclusion

We investigated model-discovery of dynamic systems, subject to randomness. In particular, leveraging a symbolic regression approach, we considered a setup in which only a few random observations of the system are provided in time and space. We proposed SDFL, incorporating several innovative ideas, in order to

tackle this multifaceted dynamic inference problem. Specifically, an appropriate measure for goodness-of-fit was introduced by devising a time integrated Wasserstein loss. This design choice turned out to be powerful by offering robustness of the model-discovery in presence of noise/uncertainty in the data. In addition, we derived theoretical results on the model performance, along with guarantees on its sample complexity. Motivated by the performance of `SDFL` in our numerical experiments, we believe these developments contribute to paving the way for progress on robust model-discovery from noisy and limited data. The computational scaling of `SDFL` to the higher dimensions remains an important open problem for future studies.

ACKNOWLEDGMENTS

This work is supported by the Swiss National Science Foundation under grant No. 212876.

REFERENCES

Brandon Amos, Lei Xu, and J Zico Kolter. Input convex neural networks. In *International Conference on Machine Learning*, pp. 146–155. PMLR, 2017.

Douglas Adriano Augusto and Helio JC Barbosa. Symbolic regression via genetic programming. In *Proceedings. Vol. 1. Sixth Brazilian Symposium on Neural Networks*, pp. 173–178. IEEE, 2000.

Sören Becker, Michal Klein, Alexander Neitz, Giambattista Parascandolo, and Niki Kilbertus. Discovering ordinary differential equations that govern time-series. *arXiv preprint arXiv:2211.02830*, 2022.

Luca Biggio, Tommaso Bendinelli, Alexander Neitz, Aurelien Lucchi, and Giambattista Parascandolo. Neural symbolic regression that scales. In *International Conference on Machine Learning*, pp. 936–945. PMLR, 2021.

Josh Bongard and Hod Lipson. Automated reverse engineering of nonlinear dynamical systems. *Proceedings of the National Academy of Sciences*, 104(24):9943–9948, 2007.

Lorenzo Boninsegna, Feliks Nüske, and Cecilia Clementi. Sparse learning of stochastic dynamical equations. *The Journal of chemical physics*, 148(24):241723, 2018.

J Frédéric Bonnans and Alexander Shapiro. *Perturbation analysis of optimization problems*. Springer Science & Business Media, 2013.

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.

Charlotte Bunne, Laetitia Papaxanthos, Andreas Krause, and Marco Cuturi. Proximal optimal transport modeling of population dynamics. In *International Conference on Artificial Intelligence and Statistics*, pp. 6511–6528. PMLR, 2022.

Tristan Cazenave. Monte-carlo expression discovery. *International Journal on Artificial Intelligence Tools*, 22(01):1250035, 2013.

Ricky TQ Chen, Yulia Rubanova, Jesse Bettencourt, and David K Duvenaud. Neural ordinary differential equations. *Advances in neural information processing systems*, 31, 2018.

Zhao Chen, Yang Liu, and Hao Sun. Physics-informed learning of governing equations from scarce data. *Nature communications*, 12(1):6136, 2021.

Earl A Coddington, Norman Levinson, and T Teichmann. Theory of ordinary differential equations, 1956.

Theodore Cornforth and Hod Lipson. Symbolic regression of multiple-time-scale dynamical systems. In *Proceedings of the 14th annual conference on Genetic and evolutionary computation*, pp. 735–742, 2012.

Allan Costa, Rumen Dangovski, Owen Dugan, Samuel Kim, Pawan Goyal, Marin Soljačić, and Joseph Jacobson. Fast neural models for symbolic regression at scale. *arXiv preprint arXiv:2007.10784*, 2020.

Miles Cranmer, Alvaro Sanchez Gonzalez, Peter Battaglia, Rui Xu, Kyle Cranmer, David Spergel, and Shirley Ho. Discovering symbolic models from deep learning with inductive biases. *Advances in Neural Information Processing Systems*, 33:17429–17442, 2020.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

Niccolò Discacciati and Jan S Hesthaven. Modeling synchronization in globally coupled oscillatory systems using model order reduction. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 31(5), 2021.

Florian Dörfler and Francesco Bullo. Synchronization in complex networks of phase oscillators: A survey. *Automatica*, 50(6):1539–1564, 2014.

Nicolas Fournier and Arnaud Guillin. On the rate of convergence in wasserstein distance of the empirical measure. *Probability theory and related fields*, 162(3-4):707–738, 2015.

Sébastien Gaucel, Maarten Keijzer, Evelyne Lutton, and Alberto Tonda. Learning dynamical systems using standard symbolic regression. In *Genetic Programming: 17th European Conference, EuroGP 2014, Granada, Spain, April 23-25, 2014, Revised Selected Papers 17*, pp. 25–36. Springer, 2014.

Will Grathwohl, Ricky TQ Chen, Jesse Bettencourt, Ilya Sutskever, and David Duvenaud. Ffjord: Free-form continuous dynamics for scalable reversible generative models. *arXiv preprint arXiv:1810.01367*, 2018.

Tatsunori Hashimoto, David Gifford, and Tommi Jaakkola. Learning population-level diffusions with generative rnns. In *International Conference on Machine Learning*, pp. 2417–2426. PMLR, 2016.

Yunfei Huang, Youssef Mabrouk, Gerhard Gompper, and Benedikt Sabass. Sparse inference and active learning of stochastic differential equations from data. *Scientific Reports*, 12(1):21691, 2022.

Guillaume Huguet, Daniel Sumner Magruder, Alexander Tong, Oluwadamilola Fasina, Manik Kuchroo, Guy Wolf, and Smita Krishnaswamy. Manifold interpolating optimal-transport flows for trajectory inference. *Advances in Neural Information Processing Systems*, 35:29705–29718, 2022.

Mohiul Islam, Nawwaf N Kharma, and Peter Grogono. Expansion: A novel mutation operator for genetic programming. In *IJCCI*, pp. 55–66, 2018.

Richard Jordan, David Kinderlehrer, and Felix Otto. The variational formulation of the fokker–planck equation. *SIAM journal on mathematical analysis*, 29(1):1–17, 1998.

Pierre-Alexandre Kamienny, Stéphane d'Ascoli, Guillaume Lample, and Francois Charton. End-to-end symbolic regression with transformers. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho (eds.), *Advances in Neural Information Processing Systems*, 2022. URL `https://openreview.net/forum?id=GoOuIrDHG_Y`.

Patrick Kidger. On neural differential equations. *arXiv preprint arXiv:2202.02435*, 2022.

John R Koza. Genetic programming as a means for programming computers by natural selection. *Statistics and computing*, 4:87–112, 1994.

Jiří Kubalík, Erik Derner, and Robert Babuška. Neural networks for symbolic regression. *arXiv preprint arXiv:2302.00773*, 2023.

Matt J Kusner, Brooks Paige, and José Miguel Hernández-Lobato. Grammar variational autoencoder. In *International conference on machine learning*, pp. 1945–1954. PMLR, 2017.

Guillaume Lample and François Charton. Deep learning for symbolic mathematics. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=S1eZYeHFDS.

Tor Lattimore and Csaba Szepesvári. *Bandit algorithms*. Cambridge University Press, 2020.

Li Li, Minjie Fan, Rishabh Singh, and Patrick Riley. Neural-guided symbolic regression with asymptotic constraints. *arXiv preprint arXiv:1901.07714*, 2019.

Jean-Christophe Loiseau and Steven L Brunton. Constrained sparse galerkin regression. *Journal of Fluid Mechanics*, 838:42–67, 2018.

Qiang Lu, Fan Tao, Shuo Zhou, and Zhiguang Wang. Incorporating actor-critic in monte carlo tree search for symbolic regression. *Neural Computing and Applications*, 33:8495–8511, 2021.

Georg Martius and Christoph H Lampert. Extrapolation and learning equations. *arXiv preprint arXiv:1610.02995*, 2016.

Kevin R Moon, David van Dijk, Zheng Wang, Scott Gigante, Daniel B Burkhardt, William S Chen, Kristina Yim, Antonia van den Elzen, Matthew J Hirn, Ronald R Coifman, et al. Visualizing structure and transitions in high-dimensional biological data. *Nature biotechnology*, 37(12):1482–1492, 2019.

Rémi Munos et al. From bandits to monte-carlo tree search: The optimistic principle applied to optimization and planning. *Foundations and Trends® in Machine Learning*, 7(1):1–129, 2014.

Harald Niederreiter. Quasi-monte carlo methods and pseudo-random numbers. *Bulletin of the American mathematical society*, 84(6):957–1041, 1978.

Brenden K Petersen, Mikel Landajuela, T Nathan Mundhenk, Claudio P Santiago, Soo K Kim, and Joanne T Kim. Deep symbolic regression: Recovering mathematical expressions from data via risk-seeking policy gradients. *arXiv preprint arXiv:1912.04871*, 2019.

Markus Quade, Markus Abel, Kamran Shafi, Robert K Niven, and Bernd R Noack. Prediction of dynamical systems by symbolic regression. *Physical Review E*, 94(1):012214, 2016.

Samuel Rudy, Alessandro Alla, Steven L Brunton, and J Nathan Kutz. Data-driven identification of parametric partial differential equations. *SIAM Journal on Applied Dynamical Systems*, 18(2):643–660, 2019.

Samuel H Rudy, Steven L Brunton, Joshua L Proctor, and J Nathan Kutz. Data-driven discovery of partial differential equations. *Science advances*, 3(4):e1602614, 2017.

Subham Sahoo, Christoph Lampert, and Georg Martius. Learning equations for extrapolation and control. In *International Conference on Machine Learning*, pp. 4442–4450. PMLR, 2018.

Hayden Schaeffer. Learning partial differential equations via data discovery and sparse optimization. *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 473(2197):20160446, 2017.

Hayden Schaeffer, Giang Tran, and Rachel Ward. Extracting sparse high-dimensional dynamics from limited data. *SIAM Journal on Applied Mathematics*, 78(6):3279–3295, 2018.

Devavrat Shah, Qiaomin Xie, and Zhi Xu. Non-asymptotic analysis of monte carlo tree search. In *Abstracts of the 2020 SIGMETRICS/Performance Joint International Conference on Measurement and Modeling of Computer Systems*, pp. 31–32, 2020.

Hongzhi Shi, Jingtao Ding, Yufan Cao, Li Liu, Yong Li, et al. Learning symbolic models for graph-structured physical mechanism. In *The Eleventh International Conference on Learning Representations*, 2023.

Guido F Smits and Mark Kotanchek. Pareto-front exploitation in symbolic regression. *Genetic programming theory and practice II*, pp. 283–299, 2005.

Fangzheng Sun, Yang Liu, Jian-Xun Wang, and Hao Sun. Symbolic physics learner: Discovering governing equations via monte carlo tree search. In *The Eleventh International Conference on Learning Representations*, 2023. URL `https://openreview.net/forum?id=ZTK3SefE8_Z`.

Alexander Tong, Jessie Huang, Guy Wolf, David Van Dijk, and Smita Krishnaswamy. Trajectorynet: A dynamic optimal transport network for modeling cellular dynamics. In *International conference on machine learning*, pp. 9526–9536. PMLR, 2020.

Giang Tran and Rachel Ward. Exact recovery of chaotic systems from highly corrupted data. *Multiscale Modeling & Simulation*, 15(3):1108–1129, 2017.

Harsha Vaddireddy, Adil Rasheed, Anne E Staples, and Omer San. Feature engineering and symbolic regression methods for detecting hidden physics from sparse sensor observation data. *Physics of Fluids*, 32(1):015113, 2020.

Martin Vastl, Jonáš Kulhánek, Jirí Kubalík, Erik Derner, and Robert Babuška. Symformer: End-to-end symbolic regression using transformer-based architecture. *arXiv preprint arXiv:2205.15764*, 2022.

Cédric Villani. *Optimal transport: old and new*, volume 338. Springer, 2009.

A PROOF OF THEOREM 1

Proof. Recall that we would like to prove there are positive constants $M_1, M_2 > 0$ such that:

$$\mathbb{E}\left|\frac{1}{n}\sum_{i=1}^{n}\mathcal{W}_2(\boldsymbol{F}^{t_i}(\hat{\boldsymbol{f}})_{\#}\hat{\mu}_{t_0,m}, \hat{\mu}_{t_i,m}) \;-\; \frac{1}{T}\int_0^T \mathcal{W}_2(\boldsymbol{F}^t(\hat{\boldsymbol{f}})_{\#}\mu_0,\, \mu_t)\,\mathrm{d}t\right| \leq \frac{M_1}{m^{(d/2+1)}} + \frac{M_2}{n}$$

provided $m, n \geq 2$. Without loss of generality, we assume $T = 1$. By proposition 5 and 7 in Fournier & Guillin (2015), given that $\boldsymbol{x}_0 \mapsto \boldsymbol{F}^t(\boldsymbol{x}_0)$ is continuous and for all $i \in \{1, \ldots, n\}$, μ_{t_i} has compact support on the Polish space $\mathbb{R}^d$, there exists $C_1, C_2 > 0$ such that

$$\begin{aligned}
\mathbb{E}\left[\frac{1}{n}\sum_{i=1}^{n}\mathcal{W}_2(\boldsymbol{F}^{t_i}(\hat{\boldsymbol{f}})_{\#}\hat{\mu}_{t_0,m}, \boldsymbol{F}^{t_i}(\hat{\boldsymbol{f}})_{\#}\mu_{t_0})\right] &\leq \frac{C_1}{m^{(d/2+1)}} \\
\text{and} \qquad \mathbb{E}\left[\frac{1}{n}\sum_{i=1}^{n}\mathcal{W}_2(\hat{\mu}_{t_i,m}, \mu_{t_i})\right] &\leq \frac{C_2}{m^{(d/2+1)}}\,.
\end{aligned}$$

Therefore, thanks to the triangular inequality satisfied by $\mathcal{W}_2$, we get

$$\begin{aligned}
\mathbb{E}\left|\frac{1}{n}\sum_{i=1}^{n}\mathcal{W}_2(\boldsymbol{F}^{t_i}(\hat{\boldsymbol{f}})_{\#}\hat{\mu}_{t_0,m}, \hat{\mu}_{t_i,m}) - \frac{1}{n}\sum_{i=1}^{n}\mathcal{W}_2(\boldsymbol{F}^{t_i}(\hat{\boldsymbol{f}})_{\#}\mu_{t_0}, \mu_{t_i})\right| &\leq \\
\mathbb{E}\left[\frac{1}{n}\sum_{i=1}^{n}\mathcal{W}_2(\boldsymbol{F}^{t_i}(\hat{\boldsymbol{f}})_{\#}\hat{\mu}_{t_0,m}, \boldsymbol{F}^{t_i}(\hat{\boldsymbol{f}})_{\#}\mu_{t_0})\right] + \mathbb{E}\left[\frac{1}{n}\sum_{i=1}^{n}\mathcal{W}_2(\hat{\mu}_{t_i,m}, \mu_{t_i})\right] &\leq \frac{C_1 + C_2}{m^{(d/2+1)}}\,. \qquad (5)
\end{aligned}$$

On the other hand, denoting by $\mathcal{M}(\mathbb{R}^d \times \mathbb{R}^d)$ the Banach space of finite measures on $\mathbb{R}^d \times \mathbb{R}^d$ equipped with the total variation norm, the mapping $(t, \gamma) \in [0,1] \times \mathcal{M}(\mathbb{R}^d \times \mathbb{R}^d) \mapsto \int_{\mathbb{R}^d \times \mathbb{R}^d} \|\boldsymbol{F}^t(\hat{\boldsymbol{f}})(\boldsymbol{x}) - \boldsymbol{y}\|_2^2 \,\mathrm{d}\gamma(\boldsymbol{x}, \boldsymbol{y})$ is differentiable. Moreover, by compactness of the set of couplings given two fixed marginals and by theorem 5.20 in Villani (2009), conditions (ii) and (iii) of theorem 4.24 in Bonnans & Shapiro (2013) are satisfied leading to the fact that the mapping $t \mapsto \mathcal{W}_2(\boldsymbol{F}^t(\hat{\boldsymbol{f}})_{\#}\mu_0, \mu_t)$ is continuously differentiable on $[0,1]$[3]. Hence, there exists a positive constant $C_3 > 0$ such that

$$\left|\int_{(i-1)/n}^{i/n} \mathcal{W}_2(\boldsymbol{F}^t(\hat{\boldsymbol{f}})_{\#}\mu_0,\, \mu_t)\,\mathrm{d}t - \frac{1}{n}\mathcal{W}_2(\boldsymbol{F}^{t_i}(\hat{\boldsymbol{f}})_{\#}\mu_{t_0}, \mu_{t_i})\right| \leq \frac{C_3}{n^2}$$

for $i \in \{1, ..., n\}$, which yields

$$\left|\int_0^1 \mathcal{W}_2(\boldsymbol{F}^t(\hat{\boldsymbol{f}})_{\#}\mu_0,\, \mu_t)\,\mathrm{d}t - \frac{1}{n}\sum_{i=1}^{n}\mathcal{W}_2(\boldsymbol{F}^{t_i}(\hat{\boldsymbol{f}})_{\#}\mu_{t_0}, \mu_{t_i})\right| \leq \frac{C_3}{n}\,.$$

Consequently, by summation with (5), we get

$$\mathbb{E}\left|\frac{1}{n}\sum_{i=1}^{n}\mathcal{W}_2(\boldsymbol{F}^{t_i}(\hat{\boldsymbol{f}})_{\#}\hat{\mu}_{t_0,m}, \hat{\mu}_{t_i,m}) \;-\; \int_0^1 \mathcal{W}_2(\boldsymbol{F}^t(\hat{\boldsymbol{f}})_{\#}\mu_0,\, \mu_t)\,\mathrm{d}t\right| \leq \frac{C_1 + C_2}{m^{(d/2+1)}} + \frac{C_3}{n}\,.$$

□

[3]Given that the optimal coupling is unique and that the directional derivative is continuous with respect to $t \in [0,1]$ -as a supremum of a differentiable parametric family of convex function.

B Proof of Proposition 1

Proof. First, denote by $(\mu_t)_{t\in[0,T]}$ the ground-truth probability flow and let $\boldsymbol{f}$, $\boldsymbol{g}$ be the continuously differentiable functions defined by two analytic expressions from the selection space of MCTS. Recall that the goal is to to maximize the functional given by

$$S : \boldsymbol{f} \mapsto \frac{1}{1 + L(\boldsymbol{f})} \ .$$

The result is obtained by showing that the conditions of theorem 2 in Shah et al. (2020) are satisfied in our setting. For that matter, it suffices[4] to show that

$$L : \boldsymbol{f} \mapsto \int_0^T \mathcal{W}_2(\boldsymbol{F}^t(\boldsymbol{f})_{\#}\mu_0,\, \mu_t)\, \mathrm{d}t$$

is Lipschitz with respect to the L^1 or $\|\cdot\|_\infty$ norms; since the derivative of $\boldsymbol{h} : \boldsymbol{x} \mapsto \frac{1}{1+\boldsymbol{x}}$ is bounded on $\mathbb{R}^+$ implying that $\boldsymbol{h}$ is Lipschitz. Furthermore, we have

$$\begin{aligned}
|S(\boldsymbol{f}) - S(\boldsymbol{g})| &\leq \int_0^T \left|\mathcal{W}_2(\boldsymbol{F}^t(\boldsymbol{f})_{\#}\mu_0,\, \mu_t) - \mathcal{W}_2(\boldsymbol{F}^t(\boldsymbol{g})_{\#}\mu_0,\, \mu_t)\right|\, \mathrm{d}t \\
&\leq \int_0^T \mathcal{W}_2(\boldsymbol{F}^t(\boldsymbol{f})_{\#}\mu_0, \boldsymbol{F}^t(\boldsymbol{g})_{\#}\mu_0)\, \mathrm{d}t
\end{aligned}$$

because $\mathcal{W}_2$ is a distance. Additionally, since both distributions have compact support, and by regularity of $\boldsymbol{F}^t(\boldsymbol{f}), \boldsymbol{F}^t(\boldsymbol{g})$, there exists a constant $C > 0$ such that for all $t \in [0, T]$,

$$\begin{aligned}
\mathcal{W}_2(\boldsymbol{F}^t(\boldsymbol{f})_{\#}\mu_0, \boldsymbol{F}^t(\boldsymbol{g})_{\#}\mu_0) &\leq C \cdot \mathcal{W}_1(\boldsymbol{F}^t(\boldsymbol{f})_{\#}\mu_0, \boldsymbol{F}^t(\boldsymbol{g})_{\#}\mu_0) \\
&\leq C \cdot \sup_{\mathrm{Lip}(q)\leq 1} \left| \int q\, \mathrm{d}(\boldsymbol{F}^t(\boldsymbol{f})_{\#}\mu_0) - \int q\, \mathrm{d}(\boldsymbol{F}^t(\boldsymbol{g})_{\#}\mu_0) \right| \\
&\leq C \cdot \sup_{\mathrm{Lip}(q)\leq 1} \left| \int q \circ \boldsymbol{F}^t(\boldsymbol{f})\, \mathrm{d}\mu_0 - \int q \circ \boldsymbol{F}^t(\boldsymbol{g})\, \mathrm{d}\mu_0 \right| \\
&\leq C \cdot \sup_{\mathrm{Lip}(q)\leq 1} \int \left| q \circ \boldsymbol{F}^t(\boldsymbol{f}) - q \circ \boldsymbol{F}^t(\boldsymbol{g}) \right|\, \mathrm{d}\mu_0 \\
&\leq C \cdot \int \left| \boldsymbol{F}^t(\boldsymbol{f}) - \boldsymbol{F}^t(\boldsymbol{g}) \right|\, \mathrm{d}\mu_0 \\
&\leq T \cdot C \cdot \|\boldsymbol{f} - \boldsymbol{g}\|_\infty
\end{aligned}$$

where the second inequality is justified by the dual representation of $\mathcal{W}_1$, the third by the change of variable formula[5], and the fifth by the Lipschitz property of q. The last inequality is justified by the fact that the solutions $t \mapsto \boldsymbol{F}^t(\boldsymbol{f})(\boldsymbol{x})$ and $t \mapsto \boldsymbol{F}^t(\boldsymbol{g})(\boldsymbol{x})$ of the differential equations $\dot{\boldsymbol{y}} = \boldsymbol{f}(\boldsymbol{y})$ and $\dot{\boldsymbol{y}} = \boldsymbol{g}(\boldsymbol{y})$ respectively, for a given initial condition $\boldsymbol{x} \in \mathbb{R}^d$, are fixed points of the Picard operator[6], which is Lipschitz.

Consequently, integrating over $t \in [0, T]$ on both sides, we obtain the Lipschitz property of S. Finally, since $L(\boldsymbol{f})$ can be approximated by $\hat{L}_{m,n}(\boldsymbol{f})$ with arbitrary accuracy, we get the sample complexity upper-bound. □

[4]Note that, since the considered state space is finite, we do not need to construct an explicit covering.

[5]Note that $q \circ \boldsymbol{F}^t(\boldsymbol{f})$ denotes the composition of q by $\boldsymbol{F}^t(\boldsymbol{f})$.

[6]The Picard operator is introduced in the proof of the existence of (local) solutions to ODEs (see e.g. Coddington et al. (1956)).

C IMPLEMENTATION DETAILS & REPRODUCIBILITY

For the implementation of SDFL, we set the building operations consisting of $\{+, -, \times, \div, \cos, \sin, \exp\}$ with a maximum of $L = 20$ operations per expression, with a number of episodes of 500 to 1000. For the recovery of the Kuramoto system, we use 15 snapshots with time-stamps $t_i = 2i$ for $1 \leq i \leq 15$, and we set $K = 1/3$. The obtained equation, after fitting two regression parameters for 150 samples per screenshot, is given by:

$$\begin{cases} \dot{\theta_1} = 0.0087 + 0.3293 * (\sin(\theta_2 - \theta_1) + \sin(\theta_3 - \theta_1)) \\ \dot{\theta_2} = 0.0087 + 0.3293 * (\sin(\theta_1 - \theta_2) + \sin(\theta_3 - \theta_2)) \\ \dot{\theta_3} = 0.0087 + 0.3293 * (\sin(\theta_2 - \theta_3) + \sin(\theta_1 - \theta_3)) \end{cases} \tag{6}$$

The fitted parameters correspond to $m = 50$ sample points per screenshot.
For the cellular dynamics data, after a pre-processing step using PHATE Moon et al. (2019), the dimension is reduced to $d = 3$ then standard Gaussian noise samples were added to the training set. The global structure of the obtained ODE system (after 50 episodes) is given by:

$$\begin{cases} \dot{x_1} = \cos(x_2) * x_1 + \cos(x_3) * x_1 \\ \dot{x_2} = \cos(x_1) * x_2 + \cos(x_3) * x_2 \\ \dot{x_3} = \cos(x_2) * x_3 + \cos(x_1) * x_3 \end{cases} \tag{7}$$

Given the inductive bias of searching an explicit ODE model, we note that SDFL gives the same model across sample sizes. After this model is obtained, additive regression coefficients are fitted resulting in the prediction errors reported in table 2. For comparison, we used the publicly available implementations of JKOnet and TrajectoryNet, from Bunne et al. (2022) and Tong et al. (2020) respectively. we retrain the models with the architectures and hyper-parameters proposed by the respective authors Tong et al. (2020); Bunne et al. (2022); however, we employ early-stopping to avoid over-fitting to the smaller data-sets[7]. For JKOnet, we use a small regularization parameter $\varepsilon = 0.001$ to make its target closer to the Wasserstein distance. Additionally, to foster reproducibility, a Python implementation of SDFL will be made public following the review of the paper.

D ADDITIONAL NUMERICAL EXPERIMENT DETAILS

D.1 COMPUTATIONAL TIME

We present in tables 3 and 4 a computational time comparison between SDFL, JKOnet and TrajectoryNet. For a fair comparison, all the reported running times are obtained on an Intel(R) Core(TM) i7-7500U CPU. The reported times correspond to training on a sample of $m = 50$ per screen-shot for the Kuramoto system modelling task and $m = 100$ per screen-shot for the scRNA-seq evolution modelling one. It is worth emphasizing that the reported running times for JKONet and TrajectoryNet do not take into account hyper-parameter tuning. And, since SDFL does not require the latter, we believe it is still competitive (for low dimensions).

Method	SDFL	JKONet	TrajectoryNet
Time (hours)	5.8166	0.7044	5.7709

Table 3: Average running time for the Kuramoto system modelling

[7]Number of iterations used were 1000 for JKOnet and 1500 for TrajectoryNet

Method	SDFL	JKONet	TrajectoryNet
Time (hours)	4.4419	0.4696	3.8472

Table 4: Average running time for the scRNA-seq evolution modelling

D.2 Permutation-invariance ablation study

For the purpose of illustrating the importance of enforcing permutation-invariance, we report below the performance metric when the latter property is not enforced; for the scRNA-seq evolution modelling task. We denote by SDFL-WPI the scheme Without the Permutation Invariance module.

Method	SDFL	SDFL-WPI
$\mathcal{W}_2$ metric	2.0822	2.3247

Table 5: Prediction loss for the scRNA-seq evolution modelling task

D.3 Robustness of the designed loss function

We illustrate in figure 3 below the robustness of the Wasserstein distance on which builds our loss function. More precisely, we represent the variation in the values of the $\mathcal{W}_2$-distance when adding noise, in comparison to the standard Euclidean mean squared error.

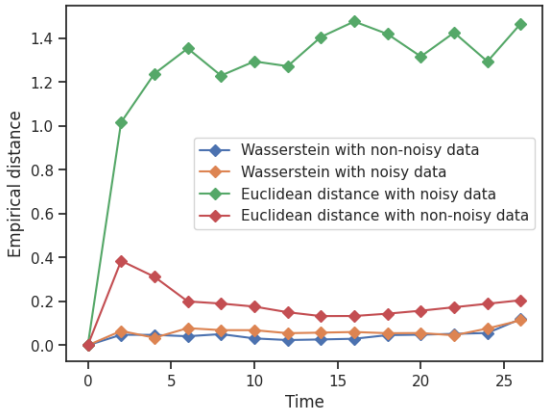

Figure 3: Distance between inferred and reference distributions, by different metrics.

E Related works

Symbolic search. Symbolic regression was initially formulated as a discrete optimization problem Augusto & Barbosa (2000); Smits & Kotanchek (2005); Cornforth & Lipson (2012); Cazenave (2013), where the goal is to find the most accurate mathematical expression based on a predefined set of elementary

operations and functions (e.g. $+, -, \times, \sin, \exp$). The mathematical expressions were represented through a one-to-one correspondence with *pre-order traversal* trees. For the resolution, Genetic Programming (GP) heuristics Koza (1994) were used to recover the equations underlying the training data. This method subsequently inspired applications in population evolution modelling Bongard & Lipson (2007), prediction of solar power production Quade et al. (2016), and Eulerian fluid flow hidden parameterization discovery Vaddireddy et al. (2020). Yet, GP suffers from a number of issues including over-fitting, brittleness to noise and poor-scalability Brunton et al. (2016); Petersen et al. (2019); Lu et al. (2021). A more recent approach Petersen et al. (2019) uses a Deep Reinforcement Learning (DRL) model to solve the optimization problem and generally outperforms GP-based models. However, deep learning models require large amounts of data in addition to their relative lack of full-automation, since architecture hyper-parameters have to be tuned by a human expert. On the other hand, a parallel well performing approach with the potential to achieve full-automation is Monte-Carlo Tree Search (MCTS). Based on the tree representation used in GP formulations, MCTS builds upon exploration-exploitation trade-off insights from sequential learning Munos et al. (2014). It is designed for a stochastic setting where data-points are costly to obtain and therefore naturally handles noisy input and scare-data settings well. Some works have applied MCTS successfully to simple Cazenave (2013); Islam et al. (2018); Lu et al. (2021) and broader Sun et al. (2023) symbolic regression problems. The former used MCTS to uncover non-linear expressions in a supervised setting, whereas the latter also applied it to Ordinary Differential Equation (ODE) discovery when the initial conditions are deterministic. In the case of network structured problems, Shi et al. (2023); Cranmer et al. (2020) are the only works -to the best of our knowledge- which distill explicit equations. That is done by training a Graph Neural Network (GNN) and distilling equations for the message-passing operators. Additionally, only deterministic physical systems for which the whole trajectory is observed are considered. In contrast, we address a setting, where the goal is to identify the governing equations of *stochastic* network dynamics (with randomness in the initial condition) from the observation of a reduced number of screen shots across time.

Sparse regularization. One common property that emerges when modelling different natural phenomena and engineering problems is sparsity. Leveraging this fact, a number of works Brunton et al. (2016); Schaeffer (2017); Rudy et al. (2017); Loiseau & Brunton (2018); Rudy et al. (2019); Chen et al. (2021) formulate (deterministic) dynamic discovery as a ℓ_1-regularized linear regression problem, over a predefined dictionary of basis functions. Subsequently, Boninsegna et al. (2018); Huang et al. (2022) extend this approach to the discovery of Stochastic Differential Equations (SDEs), using the Kramers-Moyal formula. From a theoretical perspective, Tran & Ward (2017) addresses the setting of highly corrupted data and provides conditions under which the underlying (polynomial) ODE can be recovered through ℓ_1-regularized regression. On the other hand, Schaeffer et al. (2018) studies the question of minimal number of screenshots required to recover multivariate quadratic ODEs, in the case of random initial. However, they assume velocities to be known which limits their approach for the small and scattered data setting. Overall, although usefulness of sparsity-promoting approaches has been extensively demonstrated, they still rely on prior knowledge to define one of the main components, namely the function library. Whereas, if a library of massive size is chosen, the algorithm empirically fails to hold the sparsity constraint Sun et al. (2023). In comparison, we apply MCTS for our symbolic search part, which is not bound by such constraints.

Sequence to sequence models. The most recent approach to symbolic regression leverages the success of deep learning in sequential data modeling, such as in natural language processing Devlin et al. (2018). More precisely, Biggio et al. (2021); Kamienny et al. (2022); Vastl et al. (2022) propose a transformer-based architecture, which is trained to output mathematical expressions based on data-sets of feature-prediction pairs. That is, for each input data-set $\{(x_i, y_i)\}_{i=1}^n$, the model outputs an expression e corresponding to a function f_e satisfying $\forall\, i \in [1, n],\ y_i \simeq f_e(x_i)$. One challenge though, is to generate a training data-set $\left\{(\{(x_i^j, f_{e_j}(x_i^j))\}_{i=1}^n, f_{e_j})\right\}_{j=1}^N$ which is rich enough to represent parsimonious equations

that are frequently encountered in practice. To achieve that, equations are generated as binary trees, following the work of Lample & Charton (2020); Kusner et al. (2017) and previously mentioned symbolic search approach representations, e.g. Cazenave (2013). On the other hand, Li et al. (2019) combine a recurrent neural network with MCTS to enforce asymptotic constraints on the learned expression, while some works Martius & Lampert (2016); Sahoo et al. (2018); Costa et al. (2020); Kubalík et al. (2023) propose to use different elementary mathematical operators (e.g. $+, \times, \cos \ldots$ etc) as activation functions for neural net architectures, while imposing sparsity on the parameters to extract interpretable analytic expressions. However, these approaches rely on the generation of large data-sets, which is prohibitively costly for high-dimensional ODE/PDEs.

F DETAILED PSEUDO-CODE

Below, we provide a more detailed pseudo-code of `SDFL` while featuring more clearly the MCTS components.

Symbolic Distribution Flow Learner [extended description]

Inputs: Number of episodes N, number of roll-outs H, maximal expression length M, elementary functions set $(+, -, \times, \sin, \ldots)$, screen-shots $(\hat{\mu}_{t_i,m})_{i,m}$ at $(t_i)_{0 \leq i \leq n}$,

Initialization:
$\rightarrow$ Estimate the value of each operation $(+, -, \times, \sin, \ldots)$ as a root node through H stochastic roll-outs;
$\rightarrow$ Store these values in $V(0, a)$, where $s = 0$ represents the empty tree state and a the chosen root operation
$\rightarrow$ Define $S_{\max} := \max_a V(0, a)$

for $e = 1, \ldots, N$ **do**:

    Randomly select a root node and build an expression tree as follows:
    if Tree is complete **then**
        Evaluate the corresponding estimate $\hat{\boldsymbol{f}}$ by computing $S(\hat{\boldsymbol{f}})$;
        if $S(\hat{\boldsymbol{f}}) > S_{\max}$ **then**
            $\rightarrow S_{\max} := S(\hat{\boldsymbol{f}})$;
            $\rightarrow$ Back-propagate the obtained value $S(\hat{\boldsymbol{f}})$ by updating the values of $(V(\boldsymbol{s}_p, a))_{p \geq 1}$ where $(\boldsymbol{s}_p)_{p \geq 1} = [a_0, a_1, \ldots, a_{p-1}]$ is the finite sequence of encountered tree states before completion;
        end if
    else
        $\rightarrow$ Run H roll-outs by randomly selecting operations to extend and complete the tree;
        $\rightarrow$ Assign a value of 0 to trees resulting in an inconsistent mathematical expression;
        $\rightarrow$ Store the best estimate in $V(\boldsymbol{s}, a_b)$ where a_b is the corresponding best operation;
        $\rightarrow$ Back-propagate the obtained value of $V(\boldsymbol{s}, a_b)$ to the encountered tree states $(\boldsymbol{s}_p)_{p \geq 1} = [a_0, a_1, \ldots, a_{p-1}]$;
        $\rightarrow$ Select the operation a maximizing $UCT(\boldsymbol{s}, a)$ where $\boldsymbol{s}$ is the current state of the tree;
    end if

end for
Return: Most accurate $\hat{\boldsymbol{f}}$ over the N episodes ;

Remarks:

- The back-propagation, such as in line 15, consists in updating the values of the tree states that have been encountered until that step, in that specific episode. Note that the tree states consist of the sequence of operations which have been selected to constitute the chosen expression (until that step).
- The idea is to keep track of the operations (nodes) leading to a good score, and to give them higher chance of being selected in the next rounds.