Nonlinear Dynamics (2026) 114:1000
https://doi.org/10.1007/s11071-026-12915-9

RESEARCH

Machine-precision prediction of low-dimensional chaotic systems from noise-free data

Christof Schötz[1,2] · Niklas Boers[1,2,3]

Received: 10 March 2026 / Revised: 3 June 2026 / Accepted: 23 July 2026

Abstract
Low-dimensional chaotic systems such as the Lorenz-63 model are commonly used to benchmark system-agnostic methods for learning dynamics from data. This study shows that learning from noise-free observations in such systems can be achieved up to machine precision: using ordinary least squares regression on high-degree polynomial features with 512-bit arithmetic, a system-agnostic method is introduced that matches the accuracy of standard 64-bit numerical ODE solvers using the systems' governing equations. For the Lorenz-63 system, the method achieves valid prediction times of 36 Lyapunov times, and even up to 105 Lyapunov times with favorable precision configurations, dramatically outperforming prior work, which reaches 13 Lyapunov times at most. The results are further validated on Thomas' Cyclically Symmetric Attractor, a non-polynomial chaotic system that is considerably more complex than the Lorenz-63 model, and similar results extend to higher dimensions using the spatiotemporally chaotic Lorenz-96 model. These findings suggest that forecasting low-dimensional chaotic systems from noise-free data is effectively a solved problem.

Keywords Chaotic dynamical systems · Data-driven forecasting · Polynomial regression · High-precision arithmetic · Lorenz attractor

1 Introduction

Chaotic dynamical systems represent a wide range of complex phenomena across many natural systems and scientific fields [22, 35]. While predicting their behavior is crucial for understanding these systems, it remains challenging due to their sensitivity to initial conditions. Recently, data-driven, system-agnostic methods using machine learning (ML) have emerged as promising approaches for forecasting chaotic dynamics, ranging from low-dimensional systems such as the Lorenz-63 model [23, 24] to global weather forecasting [14, 26]. Researchers typically evaluate such methods using synthetic data generated from known chaotic systems. A commonly used metric to assess prediction quality is the valid prediction time (VPT) [24, 28], which measures how long a forecast stays close to the reference trajectory serving as ground truth. Throughout the article, the term *ground truth* denotes a numerical approximation of a system's analytical solution.

The Lorenz-63 (L63) system [16]—a three-dimensional autonomous system of first-order ordinary differential equations (ODEs) exhibiting chaotic behavior—serves as the most widely used benchmark in this field. Prior results on this system are summarized in Table 1, where the state-of-the-art VPT is approximately 13 Lyapunov times (about 14 system time units). Crucially, these results were obtained using noise-free training data and initial conditions.

This study introduces *PolyProp*, a system-agnostic method to learn and predict chaotic dynamics from noise-free data. It relies on linear regression to fit high-order polynomials with high numerical precision (512 bit in the main setting). On low-dimensional systems, it is computationally extremely efficient compared to other state-of-the-art machine learning architectures. Despite its simplicity and system-agnostic nature, *PolyProp* not only vastly outperforms previous data-driven approaches but even matches the precision of numerical ODE solvers that utilize the system's governing equations. On the 3-dimensional L63 model, it

✉ Christof Schötz
christof.schoetz@tum.de

Niklas Boers
n.boers@tum.de

1 Munich Climate Center and Earth System Modelling Group, Department of Aerospace and Geodesy, TUM School of Engineering and Design, Technical University of Munich, Munich, Germany

2 Potsdam Institute for Climate Impact Research, Potsdam, Germany

3 Department of Mathematics, University of Exeter, Exeter, UK

Published online: 04 August 2026

achieves VPT values of 36 Lyapunov times (and up to 105 with favorable precision configurations).

To contextualize this performance, the previous state-of-the-art VPT of 13 Lyapunov times corresponds roughly to a normalized root mean squared error (nRMSE) between 10^{-7} and 10^{-6} during the first Lyapunov time unit of the forecast, with normalization by the system's standard deviation. In the same interval, *PolyProp* achieves an nRMSE of 10^{-16} to 10^{-15}. This is roughly equivalent to 64-bit machine precision, 2^{-52}, and represents an improvement over the state of the art by a factor of 10^9.

To confirm that these findings are not artifacts of the numerical setup, the experiments are successfully replicated on Thomas' Cyclically Symmetric Attractor [36], a 3-dimensional chaotic system with considerably more complex dynamics than the L63 model. Furthermore, machine precision results are achieved also for higher dimensions, as exemplified with the spatiotemporally chaotic Lorenz-96 model [17].

These findings can be generalized to data with measurement noise, viewing noise as a precision limitation. In general, PolyProp trained on reduced-precision data yields a forecast accuracy indistinguishable from that of an ODE solver starting from the same reduced-precision initial condition—regardless of whether the reduction stems from rounding, bit-depth, or noise. This implies that PolyProp approximates the underlying dynamics so well that the forecast error is dominated by the initial condition error (and not by the dynamics error), which affects the numerical ODE solver equally. In this sense, PolyProp yields an optimal forecast (among methods that do not apply data assimilation to denoise the initial conditions).

Contrary to common belief, these findings suggest that chaotic systems can, in principle, be predicted over arbitrarily long timescales—given measurements that are both sufficiently numerous and precise, and a suitable learning algorithm executed with sufficient numerical precision. If the state dimension is moderate, these requirements can be fulfilled in practice to the extent that the system-agnostic, data-driven approach presented here performs at least as well as a numerical integrator with access to the system's governing equations. Thus, the potential disadvantage of not knowing these equations can be overcome with data. In this sense, the problem of learning low-dimensional chaotic systems from noise-free data can be considered solved.

2 Method and Experimental Setup

This section describes the methodology underlying the results of the previous section. After formalizing the prediction problem and evaluation metric, the three chaotic benchmark systems used in the experiments are introduced, followed by a specification of the PolyProp method, a description of the numerical precision choices central to its performance, and a summary of the experimental design.

2.1 Problem setup and evaluation metric

Consider a dynamical system described by a d-dimensional, first order, autonomous ordinary differential equation

$$\dot{u}(t) = f(u(t)), \tag{1}$$

where $u\colon \mathbb{R} \to \mathbb{R}^d$ is a *solution* and $f\colon \mathbb{R}^d \to \mathbb{R}^d$ is the vector field describing the dynamics of the system. Given initial conditions $u(0) = u_0$ for $u_0 \in \mathbb{R}^d$ and assuming f is sufficiently smooth (e.g., globally Lipschitz continuous), the solution u of Equation (1) exists and is unique. In practice, the analytical solution u is typically not available and is approximated numerically as described in Sect. 2.4.

Assume n noise-free observations $y_i = u(t_i)$, $i = 1, \ldots, n$, of the trajectory u are made at time points t_i. The observation times start at a negative value and extend up to $t_n = 0$. These times are assumed to be equally spaced. This means $t_i := (i - n)\Delta t$ for a fixed step size $\Delta t > 0$.

For the prediction task, f and u are assumed to be unknown, butthe observation time series $(y_i)_{i=1,\ldots,n}$ and the step size Δt are available. The goal is to predict the future values $z_j := u(j\Delta t)$ for $j \in \mathbb{N}$ given $z_0 = y_n$. In this case the train and test data are *sequential*. Alternatively, we also consider the *random* test mode, in which $z_j := \tilde{u}(j\Delta t)$ for $j \in \mathbb{N} \cup \{0\}$ for a new solution $\tilde{u}$ of Equation (1), where $\tilde{u}(0) = z_0$ is chosen randomly from the system's attractor. Note that, after training on $(y_i)_{i=1,\ldots,n}$, only a single state z_0 is available to initialize the prediction in the *random* test mode.

To evaluate a given forecast $(\hat{z}_j)_{j=1,\ldots,m}$, we define the Valid Prediction Time (VPT) as the maximum time duration over which the normalized Euclidean distance between the prediction and the ground truth remains below a given threshold $\varepsilon > 0$. Formally,

$$\mathsf{VPT}_\varepsilon((z_j)_{j=1,\ldots,m}, (\hat{z}_j)_{j=1,\ldots,m}) \tag{2}$$

$$:= \Delta t \max\left\{ J \in \{1, \ldots, m\} \,\middle|\, \forall j \in \{1, \ldots, J\} : \frac{\|z_j - \hat{z}_j\|_2}{\sigma} \le \varepsilon \right\}, \tag{3}$$

where σ is the standard deviation of the system, defined as

$$\sigma := \sqrt{\lim_{T\to\infty} \frac{1}{2T} \int_{-T}^{T} \|u(t) - \mu\|_2^2 \, \mathrm{d}t} \tag{4}$$

$$\text{with} \quad \mu := \lim_{T\to\infty} \frac{1}{2T} \int_{-T}^{T} u(t) \, \mathrm{d}t\,. \tag{5}$$

Table 1 Valid prediction times in the literature for the Lorenz-63 system. Forecasting performance on L63 is reported as the valid prediction time (VPT) with threshold parameter 0.5, $\mathsf{VPT}_{0.5}$, given in units of Lyapunov time. For consistency, we estimate $\mathsf{VPT}_{0.5}$ from potentially different error measures in the reference. The *Family* column indicates the algorithmic class to which each prediction method belongs. The earliest publication uses a multi-layer perceptron (MLP). Reservoir computing (RC) methods include the Echo State Network [11], Nonlinear Vector Autoregression [7], and related variants. LSTM refers to the Long Short-Term Memory network [10]. SINDy denotes the Sparse Identification of Nonlinear Dynamical Systems framework [2], which in contrast to the other methods is not fully system-agnostic, as it assumes the system dynamics to be a sparse degree 5 polynomial. For comparison, we show the VPT of a numerical solver (Runge Kutta of order 4, RK4) with access to the system's governing equations. The method we present in this study is PolyProp, short for *Polynomial Propagator*. The number of training data points is given in column n, and the temporal resolution of the data is specified by the time step Δt. Missing entries reflect cases where these details were not clearly reported in the original sources

Reference	Family	n	Δt	VPT
Elsner and Tsonis [5]	MLP	1,000	0.01	1–2
Dubois et al. [4]	LSTM	15,000	0.005	3
Jaurigue [12]	RC	10,000	0.1	3
Griffith et al. [9]	RC	10,000	0.01	4
Köster et al. [13]	RC	1,000	0.1	4
Roberts [29]	LSTM	1,000	0.01	4
Yu et al. [40]	LSTM			4
Viehweg et al. [37]	RC	2,000	0.01	5
Gauthier et al. [7]	RC	400	0.025	6
Pathak et al. [24]	RC	1,000	0.1	6
Wang et al. [39]	LSTM		0.05	6
Pathak et al. [23]	RC	5,000	0.02	7
Santos and Bollt [30]	RC	5,000	0.01	8
Silva et al. [3]	SINDy	5,000	0.002	8
Lu et al. [18]	RC	3,000	0.02	8
Akiyama and Tanaka [1]	RC	5,000	0.02	9
Li et al. [15]	RC	3,000	0.02	9
Schötz et al. [32]	misc	10,000	0.01	>9
Steinegger and Räth [33]	RC	2,100	0.02	12
Brunton et al. [2]	SINDy	100,000	0.001	13
Mandal and Gottwald [19]	RC	50,000	0.01	13
Platt et al. [25]	RC	50,000	0.01	13
RK4 (64 bit)	ODE solver	1	0.001	32
RK4 (512 bit)	ODE solver	1	0.001	35
This Study	PolyProp	16,000	0.03	36
This Study	PolyProp	*33,000	0.002	105

* data is provided at 512-bit precision instead of 64-bit

In practice, σ and μ are approximated empirically from a long ground truth trajectory,

$$\bar{\sigma} := \sqrt{\frac{1}{\ell}\sum_{j=1}^{\ell} \left\| z_j - \bar{\mu} \right\|_2^2}, \quad \text{and} \quad \bar{\mu} := \frac{1}{\ell}\sum_{j=1}^{\ell} z_j \,. \tag{6}$$

Note that the VPT in this definition is translation and scale invariant.

If the system dynamics are recreated well enough, the forecast error roughly increases by a factor of Euler's number each Lyapunov time. This can be used to translate between VPT values for different thresholds: Assuming all VPT values are given in units of Lyapunov time,

$$\mathsf{VPT}_{\varepsilon_2} \approx \mathsf{VPT}_{\varepsilon_1} + \log\left(\frac{\varepsilon_2}{\varepsilon_1}\right), \tag{7}$$

where log is the natural logarithm and $\varepsilon_1, \varepsilon_2 \in (0, 1]$. Empirically, in the simulation study presented here, this formula is a rough but valid approximation (showing at least one significant digit to be correctly approximated for well performing systems; see also the slope in Fig. 5).

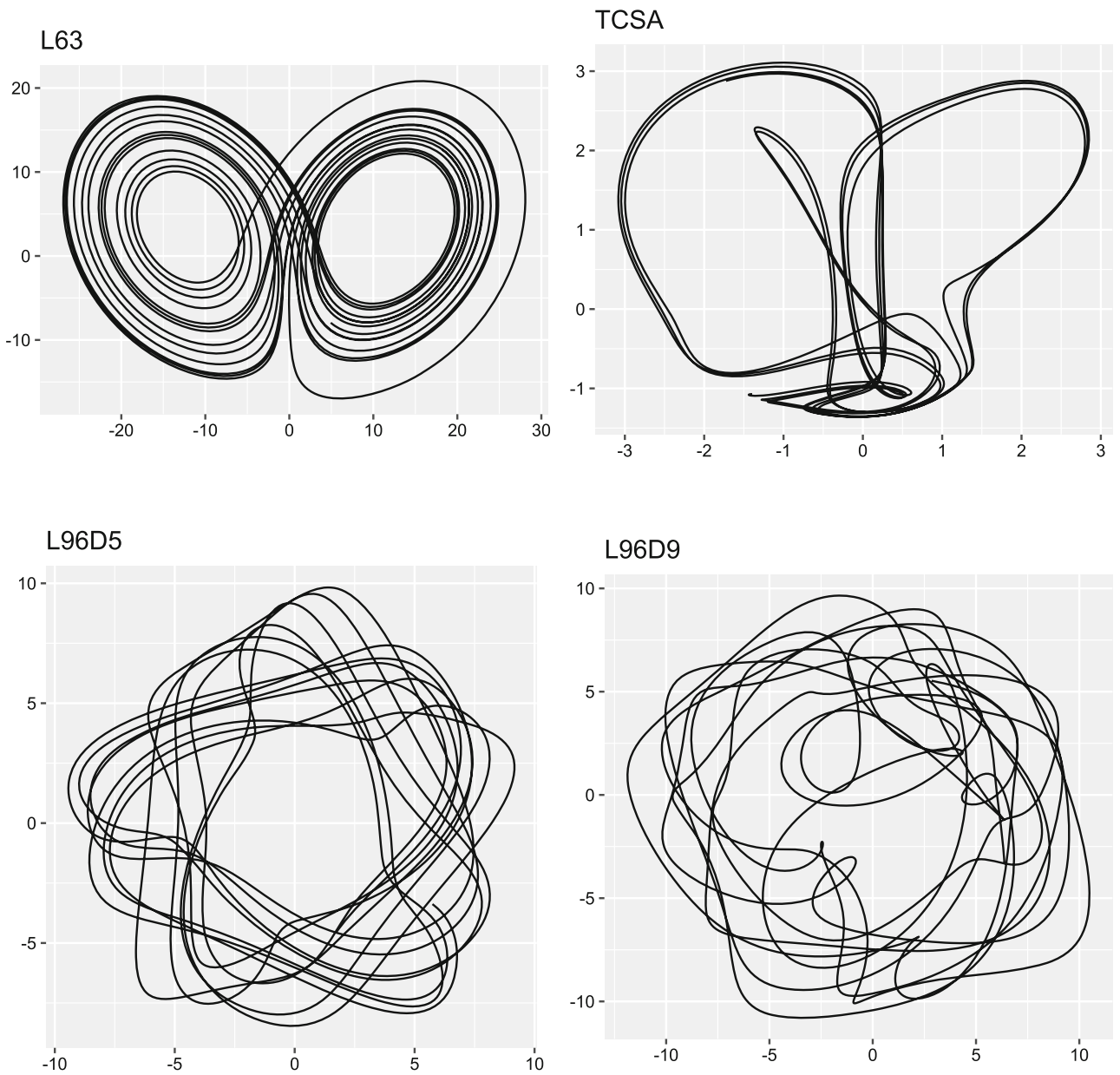

Fig. 1 State space view of the Lorenz-63 (L63), Thomas' cyclically symmetric attractor (TCSA), as well as 5-dimensional and 9-dimensional versions of the Lorenz-96 system (L96D5 and L96D9). L63, L96D5, and L96D9 are integrated for 20 and TCSA for 200 system time units, and the state space is projected to 2 dimensions using principal component analysis

2.2 Chaotic benchmark systems

We focus on three chaotic dynamical systems: the Lorenz-63 system (L63) [16], Thomas' cyclically symmetric attractor (TCSA) [36], and the Lorenz-96 system (L96) [17], see Fig. 1. Further systems are shown in Supplementary Section 8. For L63, the standard parameters are used, so that the dynamics are described by the vector field

$$f\left(\begin{bmatrix} u_1 \\ u_2 \\ u_3 \end{bmatrix}\right) = \begin{bmatrix} 10(u_2 - u_1) \\ u_1(28 - u_3) - u_2 \\ u_1 u_2 - \frac{8}{3} u_3 \end{bmatrix}. \tag{8}$$

For TCSA,

$$f\left(\begin{bmatrix} u_1 \\ u_2 \\ u_3 \end{bmatrix}\right) = \begin{bmatrix} \sin(u_2) - b u_1 \\ \sin(u_3) - b u_2 \\ \sin(u_1) - b u_3 \end{bmatrix}, \tag{9}$$

with $b = 0.208$. For L96, the standard forcing value $F = 8$ is used, yielding

$$f\left(\begin{bmatrix} u_1 \\ \vdots \\ u_i \\ \vdots \\ u_d \end{bmatrix}\right) = \begin{bmatrix} (u_2 - u_{d-1})u_d - u_1 + 8 \\ \vdots \\ (u_{i+1} - u_{i-2})u_{i-1} - u_i + 8 \\ \vdots \\ (u_1 - u_{d-2})u_{d-1} - u_d + 8 \end{bmatrix}. \tag{10}$$

The name L96Dd, with d replaced by 5, 6, 7, 8 or 9, indicates the d-dimensional version of L96.

L63 and TCSA are $(d = 3)$-dimensional; for L96, d can be chosen freely, with chaotic behavior obtained for $d \geq 5$. The vector fields for L63 and L96 are (sparse) polynomials of degree 2; for TCSA the vector field is non-polynomial but smooth. The largest Lyapunov exponent of L63 is about 0.906 [38]; for TCSA we estimate it to be 0.0153. For L96 with $d = 5, 6, 7, 8, 9$, we estimate the largest Lyapunov exponents to be 0.467, 0.946, 1.265, 1.592, 1.216, respectively. See Supplementary Section 9 for details. Time values in units of Lyapunov time are calculated as system time multiplied by the largest Lyapunov exponent.

2.3 PolyProp

To create a forecast $(\hat{z}_j)_{j=1,\dots,m}$ from the observations $(y_i)_{i=1,\dots,n}$, the propagator $\Phi_{\Delta t} : \mathbb{R}^d \to \mathbb{R}^d$, $u(t) \mapsto u(t + \Delta t)$ is first estimated. The learned propagator is then applied repeatedly starting from a given state z_0, which is either the last observed state $z_0 = y_n$ (sequential test mode) or a new randomly chosen state on the attractor of the system (random test mode).

The propagator is estimated using ordinary least squares (OLS) linear regression with multivariate monomials as features. That is, for a given state vector $s = (s_1, \dots, s_d) \in \mathbb{R}^d$ and a fixed maximal degree $p \in \mathbb{N}$, a feature vector $x \in \mathbb{R}^D$ is created using the feature function $\xi_p : \mathbb{R}^d \to \mathbb{R}^D$ such that

$$x = \xi_p(s) := \left(\prod_{k=1}^{d} s_k^{\alpha_k} \,\middle|\, \alpha_1, \dots, \alpha_d \in \{0, \dots, p\}, \sum_{k=1}^{d} \alpha_k \leq p \right). \tag{11}$$

For example,

$$\xi_2(s_1, s_2, s_3) = \left(1, s_1, s_2, s_3, s_1 s_2, s_2 s_3, s_1 s_3, s_1^2, s_2^2, s_3^2\right). \tag{12}$$

The number of features is

$$D = \binom{d+p}{d} = \frac{(d+p)!}{d!\, p!}. \tag{13}$$

For example, in $d = 3$ dimensions, $D = 165$ features are obtained using maximal degree $p = 8$, and $D = 969$ using $p = 16$. The feature function is applied to the observations $y_1, \dots, y_{n-1} \in \mathbb{R}^d$ to obtain $x_1, \dots, x_{n-1} \in \mathbb{R}^D$. Then OLS linear regression is applied to the data $(x_i, y_{i+1})_{i=1,\dots,n-1}$ to obtain the regression coefficients $\hat{\beta} \in \mathbb{R}^{D \times d}$. By stacking the feature vectors and targets row-wise into matrices $X \in \mathbb{R}^{(n-1) \times D}$ and $Y \in \mathbb{R}^{(n-1) \times d}$, respectively, the coefficients can be expressed as

$$\hat{\beta} = (X^\top X)^{-1} X^\top Y \tag{14}$$

assuming the matrix $X^\top X \in \mathbb{R}^{D \times D}$ is invertible. The estimated propagator is

$$\hat{\Phi}_{\Delta t} : \mathbb{R}^d \to \mathbb{R}^d,\ s \mapsto \hat{\beta}^\top \xi_p(s). \tag{15}$$

The forecast $(\hat{z}_j)_{j=1,\dots,m}$ is defined by

$$\hat{z}_j := \hat{\Phi}_{\Delta t}(\hat{z}_{j-1}) \tag{16}$$

with a known initial state $\hat{z}_0 := z_0$.

While PolyProp is conceptually simple, executing it for high polynomial degrees presents a numerical challenge, as systems of linear equations given by potentially ill-conditioned matrices must be solved. We address this issue through appropriate data normalization and high precision arithmetic, as described in Sect. 2.4 below.

Polynomial regression for learning dynamical systems has recently been employed [7] with a focus on time delay embedding with relatively low-degree polynomials and Ridge regression (i.e., Tikhonov regularization). In contrast, PolyProp emphasizes high-degree polynomials without time delay embedding and achieves its results without regularization, i.e., using OLS. Note that regularization is typically used to reduce variance of parameters estimated from noisy data at the cost of an induced bias; here the data is noise-free, and we have found that using regularization makes the results worse due to the introduced bias. A predecessor to this method appears in a recent simulation study [32] (called `LinPo6` therein), but it is limited to degree 6 polynomials and standard 64-bit precision.

The backbone of PolyProp, OLS with polynomial features, can be traced back to the 19th century [8, 34]. Machine learning for L63 was first explored in the early 90 s [5]. Given that 64-bit computing architectures became widely available in the mid-2000s, it is remarkable that—even though the standard-precision version of this method (VPT of 21 Lyapunov times on L63) was both conceptually and technologically available—it has not been discovered and presented in the literature for nearly two decades.

2.4 Numerical precision and implementation

Executing the procedures described above numerically is necessarily an approximation to the mathematically exact calculations. Its precision is influenced by implementation choices, such as data normalization, the algorithm used for solving linear equations, and the amount of information used for storing a single number.

On a standard machine with standard operations, a number is represented with 64 bit, which roughly translates to 15 significant digits. By using the C libraries MPLAPACK [20] and MPFR [6], numbers are represented with 512 bit of information, which allows for more than 150 significant digits. The specific value of 512 bit is not a tight requirement; it is simply the power of 2 closest to a tenfold increase in information over 64 bit, chosen to ensure operation well above the precision threshold at which ill-conditioning of the linear systems involved in polynomial fitting dominates. Any sufficiently high precision would yield qualitatively equivalent results. We run different experiments in which different processing steps are executed with either single precision (32 bit), double precision (64 bit) or multi precision (512 bit).

First, a Runge Kutta ODE solver of order 4 (RK4) is used with a solver time step of $\Delta t_0 = 2^{-10} \approx 0.001$ for L63 and L96 to create a long trajectory of the system. For TCSA, $\Delta t_0 = 2^{-6} \approx 0.016$ is used, as it has a larger characteristic timescale. A starting point is then randomly chosen on the long trajectory, and temporal sub-sampling (taking every k-th element of the trajectory) is performed to create the training and test data with the desired step size Δt, number of observations n, and a sufficiently large number m of states in the test set. The term *ground truth* denotes a numerical approximation of this type to the system's analytical solution.

When using single or double precision for the polynomial forecaster, the results depend on whether and how the data is normalized. The training data y_i may first be transformed to $\tilde{y}_i$ by normalizing it to mean 0 and covariance matrix equal to the identity matrix, i.e.,

$$\tilde{y}_i = \hat{\Sigma}^{-\frac{1}{2}}(y_i - \hat{\mu}),\ \hat{\Sigma} = \frac{1}{n-1}\sum_{i=1}^{n}(y_i - \hat{\mu})(y_i - \hat{\mu})^\top,$$
$$\hat{\mu} = \frac{1}{n}\sum_{i=1}^{n} y_i, \tag{17}$$

where $\hat{\Sigma}^{\frac{1}{2}}$ is the matrix square root of the symmetric positive definite matrix $\hat{\Sigma}$ and $\hat{\Sigma}^{-\frac{1}{2}}$ is its inverse. As this *full* normalization is not common in the literature, we also compare it with the more common *diagonal* normalization, where each dimension is scaled by the inverse of its standard deviation individually without changing correlations between variables, i.e.,

$$\tilde{y}_i = \hat{S}^{-\frac{1}{2}}(y_i - \hat{\mu}),\ \hat{S} = \begin{pmatrix} \hat{\sigma}_1^2 & & \\ & \ddots & \\ & & \hat{\sigma}_d^2 \end{pmatrix},$$
$$\hat{\sigma}^2 = \frac{1}{n-1}\sum_{i=1}^{n}\left(y_i - \hat{\mu}\right)^2. \tag{18}$$

When normalizing the training data, the output of the estimated propagator has to be scaled back to the original scale before comparing it with the test data. When using multi-precision arithmetic (512 bit), we do not normalize the data, as numerical instability does not arise in this case.

We implement the simulation study in the R programming language [27]. To evaluate Equation (14), systems of linear equations need to be solved. If the matrix $X^\top X$ is numerically close to being singular (not invertible), numerical instabilities can cause inaccurate results. Thus, this is a crucial part of the implementation. While the standard approach in R relies on the LAPACK routine `DGESV`, we found improved numerical performance by using the Armadillo library [31] for our single and double precision implementations. Specifically, we employ Armadillo's `solve` function with the options `refine`, `equilibrate`, `allow_ugly`, and `likely_sympd`, which typically yields more accurate results in this context. For the multi-precision implementation (512 bit), we use the MPLAPACK [20] routine `RGESV`.

2.5 Experimental design

If not mentioned otherwise, we run experiments with training set size $n = 2^3, 2^4, \ldots, 2^{15}$ and polynomial degrees $p = 1, 2, \ldots, 16$. For L63, we use the time step sizes $\Delta t = 2^{-10}, 2^{-9}, \ldots, 2^{-3}$; for L96, we use $\Delta t = 2^{-9}, \ldots, 2^{-5}$ and restrict to $p = 1, 2, \ldots, 8$; for TCSA, we use $\Delta t = 2^{-6}, 2^{-5}, \ldots, 2^{1}$. Additionally, L96 is run with $n = 2^{16}, 2^{17}$, and TCSA with $\Delta t = 2^{-2}$, $p = 23, 24, 25$, and $n = 2^{16}, 2^{17}$. Each experiment is repeated 100 times with a different random trajectory sample. In each case, $\mathsf{VPT}_{0.5}$ is calculated. In the results section, we report the mean over the resulting 100 VPT values.

For reference, we estimate the accuracy of the RK4 solver in terms of $\mathsf{VPT}_{0.5}$, starting from differently rounded initial conditions and using calculations of different precision internally. Averaging over 10^4 repetitions with randomly chosen initial conditions on the systems' attractor yields the values given in Supplementary Section 10.

Computational resources used in this study are described in Supplementary Section 11. Details of all experimental results are given in Supplementary Section 12.

3 Forecasting results

This section evaluates PolyProp on the three chaotic benchmark systems introduced in Sect. 2. PolyProp consistently matches the accuracy of the ODE solvers used to generate the data, thereby surpassing previous limits on prediction accuracy in low-dimensional chaotic systems.

3.1 Lorenz-63 with Default Setup

For the L63 model, the dynamics of the three state dimensions $x(t)$, $y(t)$, and $z(t)$ depending on time t are given by the ordinary differential equation (ODE)

$$\dot{x} = 10(y-x)\,, \quad \dot{y} = x(28-z)-y\,, \quad \dot{z} = xy - \frac{8}{3}z\,, \tag{19}$$

where the dot above a variable denotes its temporal derivative.

Since the L63 system models a physical process, it is appropriate to use the most accurate available approximation to the exact solution for data generation. Here, this corresponds to the output of a high-precision (512-bit) fourth-order Runge–Kutta (RK4) ODE solver. To reflect typical computational practice, the data is assumed to be stored in double precision (64 bit). The dataset is then split into train and test sets. The train set is used to fit the PolyProp forecasting model. Given that PolyProp is not computationally intensive, there is no need to artificially limit its computational precision. Therefore, a 512-bit implementation of PolyProp is used for prediction and evaluated on the test set. This default configuration is illustrated in Fig. 2.

An example trajectory is shown in Fig. 3, while results for the best-performing polynomial degree $p \in \{1, \ldots, 16\}$ across different numbers of observations n and time steps Δt are presented in Fig. 4. Note that this section shows the results for a sequential test mode, i.e., the forecast is started from the last training observation. The results for a random test mode, in which the test set is a new randomly chosen time series on the attractor, are almost identical, see Supplementary Section 1.

The optimal polynomial degree is observed to increase with the number of observations n and step size Δt. This is expected, as with larger n more complex models can be fitted, and with larger Δt the true propagator function increases in complexity.

There seems to be a trade-off regarding data efficiency in the step size, with the medium value $\Delta t = 2^{-5} \approx 0.03$ showing the best performance for most values of n. We interpret this as follows: for smaller step sizes, the target function of the regression—the propagator map—becomes simpler, allowing a precise fit with a small amount of data. In general, this is counteracted to some extent by the need for more prediction steps to achieve the same forecast time. But potentially more importantly, the state space is not sufficiently explored for a fixed n if the step size is too small.

Recall that the ground truth is generated by a 512-bit RK4 solver. Due to the sensitive dependence on initial conditions, a 512-bit RK4 prediction initialized with data rounded to 64-bit precision inevitably diverges from this ground truth. A value of $\mathsf{VPT}_{0.5} = 34.7$ Lyapunov times is measured for this effect. Given $n \geq 2^{12}$ noise-free observations of the ground truth trajectory rounded to 64-bit with a suitable time step, PolyProp matches and even slightly exceeds this value. Only the largest time steps do not reach this precision, likely because higher polynomial degrees would be required to fit the higher complexity of the propagator that comes with taking larger steps. In the best case, PolyProp reaches $\mathsf{VPT}_{0.5} = 35.7$ on average for $n = 2^{15} \approx 33{,}000$ observations, a time step of $\Delta t \leq 2^{-5} \approx 0.03$, and polynomial degree $p = 15$. This suggests that PolyProp successfully learns the underlying dynamics from the rounded data and compensates for the initialization error more effectively than the RK4 ODE solver.

An example of the evolution of the error between prediction and ground truth is shown in Fig. 5. In contrast to the defining property of chaos, a constant error is observed for about 8 Lyapunov times when rounded to 64 bit. This, of course, does not break the chaotic behavior. It rather shows that, internally, the polynomial forecaster has a high-precision representation of the system state that is more accurate than the 64 bit at which the train and test data are stored. After the initial period of seemingly constant error, the Euclidean distance grows by a factor of Euler's number each Lyapunov time—the same as it would when using the exact L63 system (by definition of Lyapunov time and Lyapunov exponent).

3.2 Variations on numerical precision

We explore the effect of single (s, 32-bit), double (d, 64-bit), and multi (m, 512-bit) precision in the system- (the RK4 ODE solver), data- (storage of train and test data), and method- (PolyProp) parts of the processing pipeline for L63. The results are shown in Fig. 6. If the source data lacks precision, the prediction accuracy is also limited. To achieve the best results for given data, the method precision needs to be larger than the effective data precision (the minimum of system and data precision). This is because solving systems of linear equations to fit polynomial propagators—especially for higher-degree polynomials—can involve ill-conditioned matrices, where numerical errors amplify and lead to a loss of accuracy beyond the nominal precision of the arithmetic used.

Figure 6 shows that PolyProp achieves predictive fidelity comparable to, or better than, numerical integration with

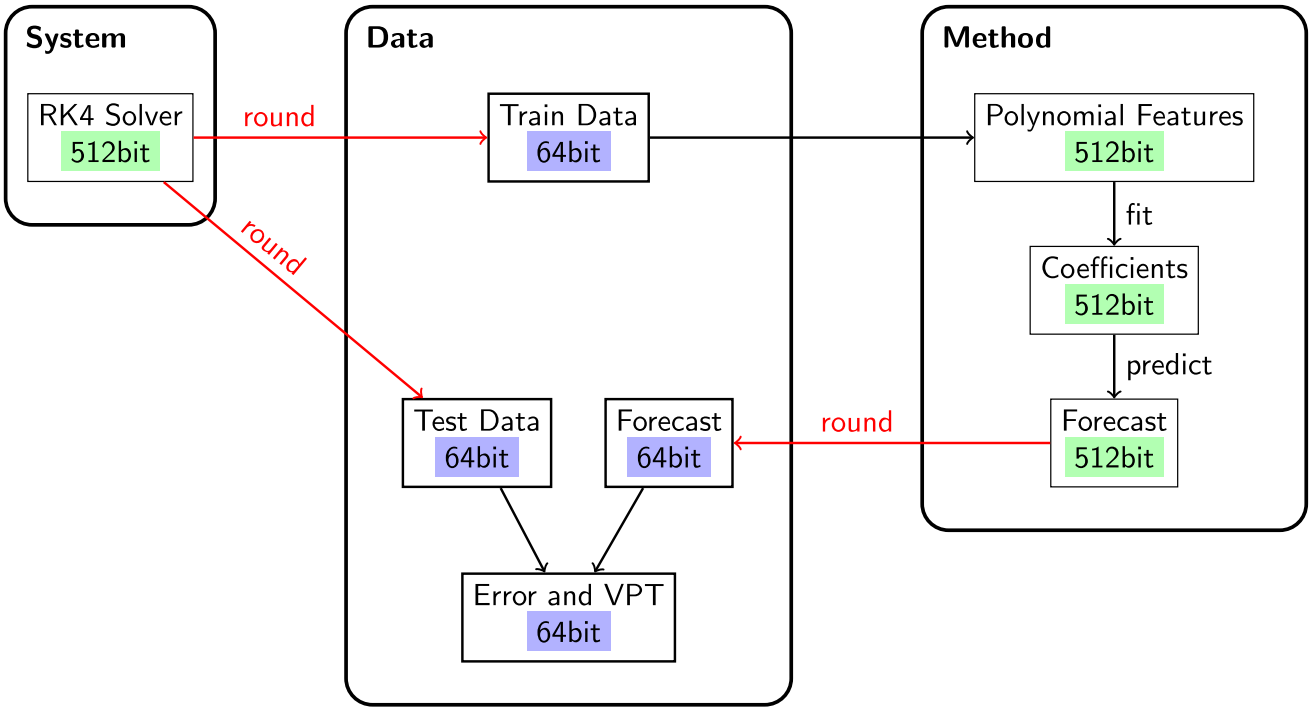

Fig. 2 Data processing diagram for the default setup. The ODE system is solved with a multi precision (512-bit) RK4 solver. The train and test data are then stored at double precision (64 bit). The polynomial propagator method internally calculates with multi precision. The evaluation and calculation of the error metric is executed at standard double precision

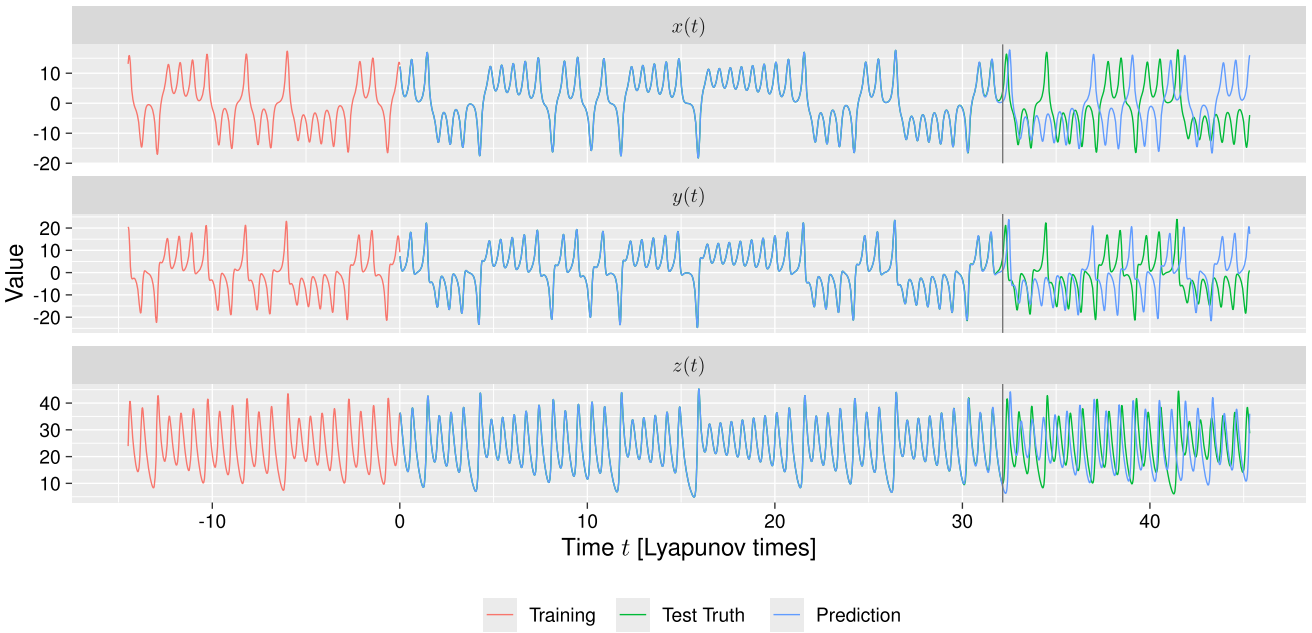

Fig. 3 An example illustrating the forecast abilities of the polynomial propagator introduced in this study. The L63 system is solved with an RK4 ODE solver calculating at 512-bit precision with time step $2^{-10} \approx 0.001$ (system time). The result is stored only at 64-bit precision. It is sub-sampled to obtain a time series with time step $\Delta t = 2^{-6} \approx 0.016$. The first $n = 2^{10} = 1024$ samples (red) are used as training data (16.0 system time units or 14.5 Lyapunov times) to fit a polynomial of degree 9 to the propagator using 512-bit arithmetic. The resulting prediction (blue) is compared with the ground truth of the ODE solver (green). The prediction is valid for about 32 Lyapunov times (dark gray vertical line), more than twice the duration of the training data. Further examples, including one with explicit regression coefficients, can be found in Supplementary Section 2

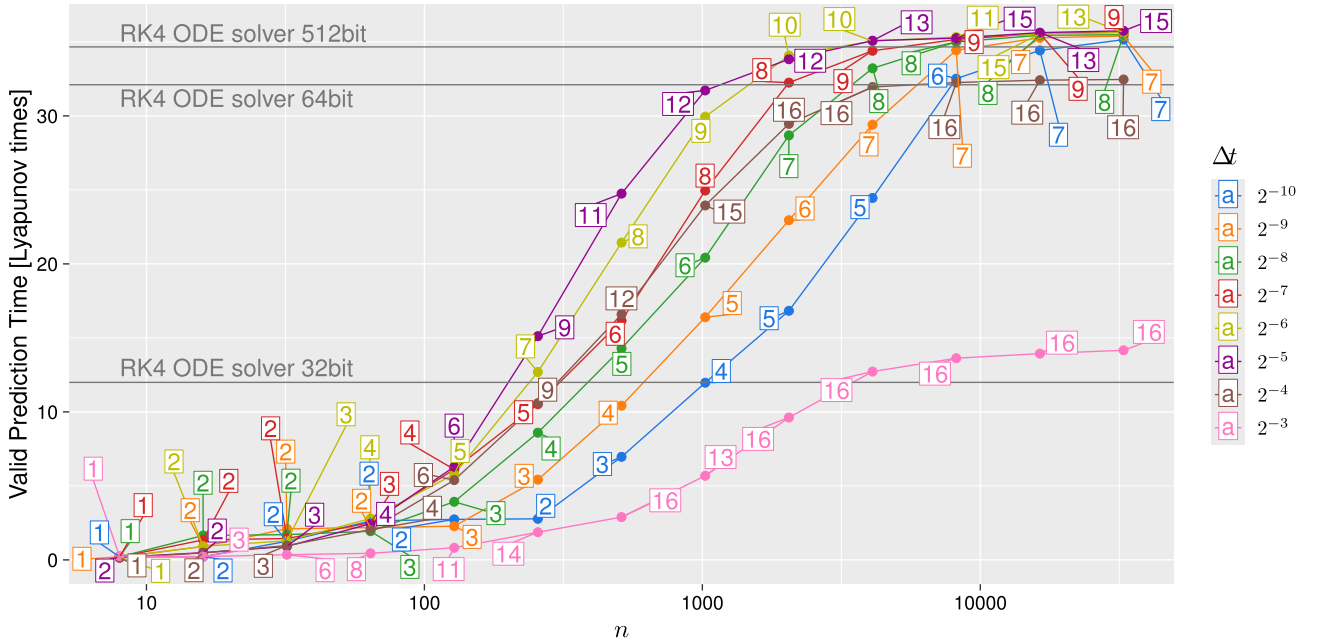

Fig. 4 Valid prediction times (VPT) for the L63 system using PolyProp with optimal degree under the default setup. The setup uses a multi-precision solver (512-bit), double-precision data storage (64-bit), and a multi-precision implementation of PolyProp. The data time step Δt is indicated by color. Label boxes within the plot denote the degree (between 1 and 16) of the optimal polynomial for each case. Gray horizontal lines show the performance of ODE solvers at various precisions, using the system's governing equations and (rounded) 64-bit initial conditions. The VPT (with threshold 0.5) values on the vertical axis are given in multiples of the Lyapunov time

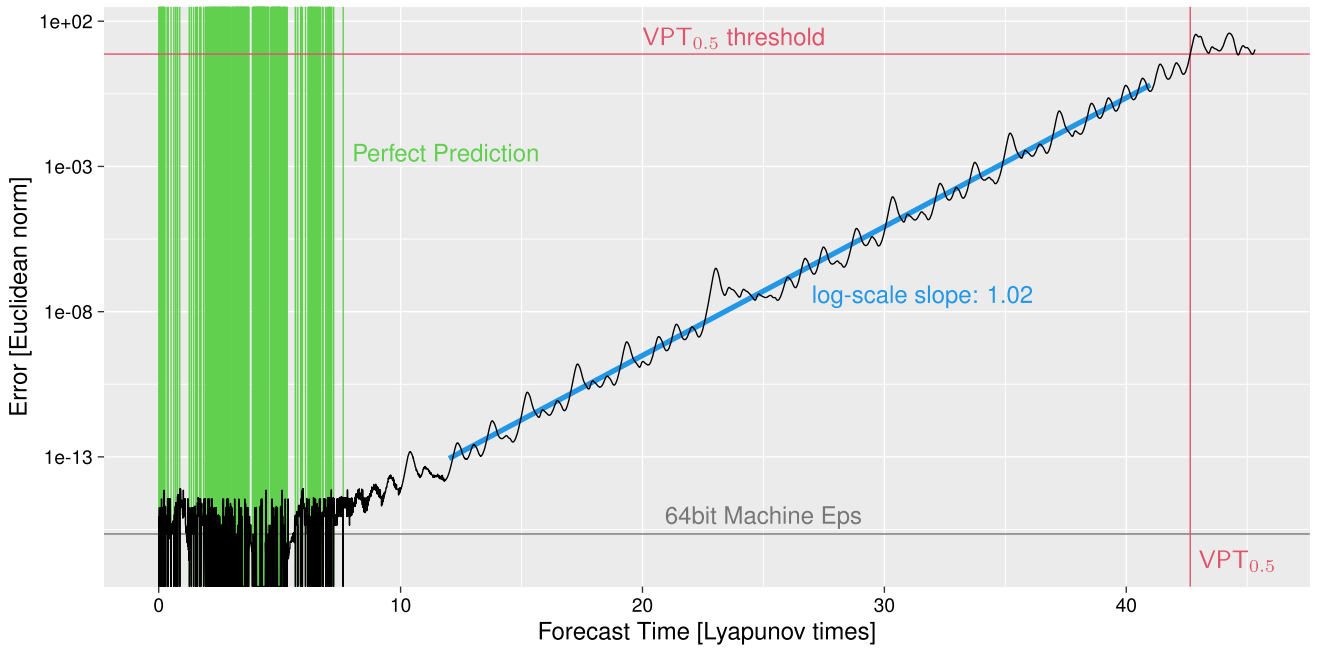

Fig. 5 Euclidean distance between forecast and ground truth of a single L63 run. This is the best-performing repetition of the default setup (L63, 512-bit system, 64-bit data, 512-bit method, sequential test mode, $n = 2^{15}$, $\Delta t = 2^{-7}$, $p = 9$). The horizontal locations of the vertical green lines mark time points of the forecast with numerically perfect prediction ($\hat{u}(t) = u(t)$ in 64-bit arithmetic). The blue line is the linear fit between forecast time in Lyapunov times and the natural logarithm of the Euclidean distance; it has almost the theoretically predicted slope 1 of a perfectly emulated L63 system. The red lines show the VPT calculation; here, $\mathsf{VPT}_{0.5} = 42.6$ is obtained, which is a positive outlier compared to 90% of runs in this setting yielding $\mathsf{VPT}_{0.5}$ between 33.0 and 39.0

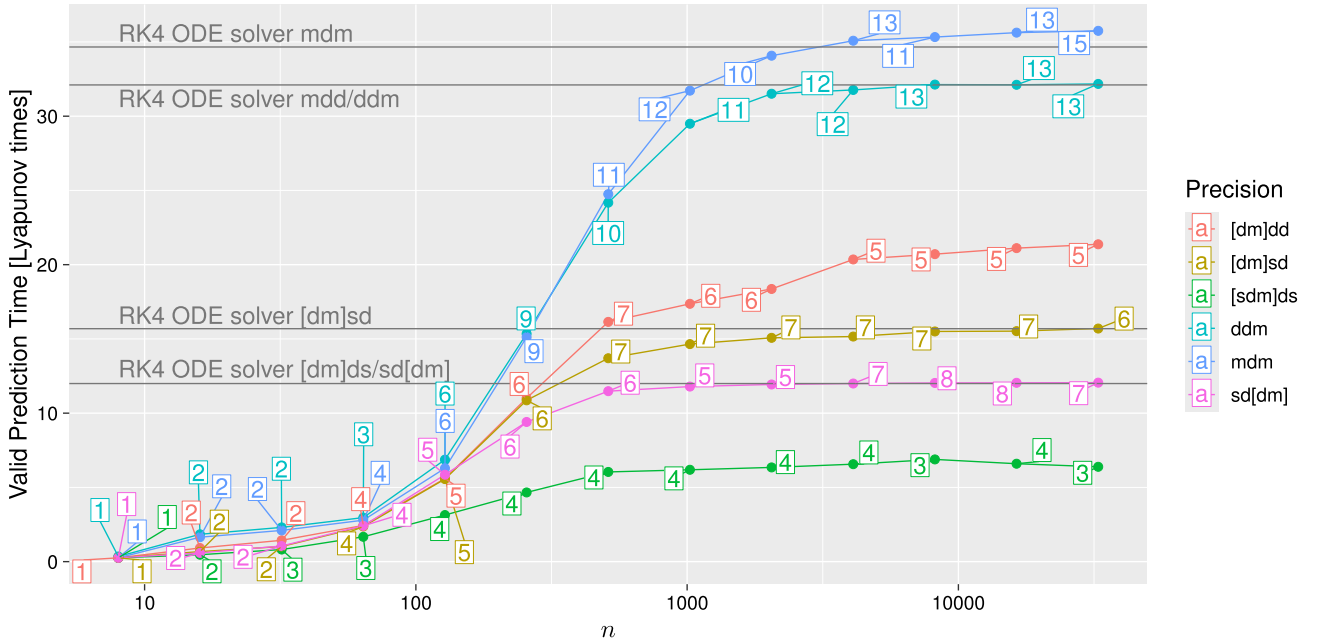

Fig. 6 Valid prediction time (VPT) for L63 using PolyProp with best polynomial degree and best time step in different precision setting. The precision is encoded by three letters. The first letter represents the system (ODE solver), the second the data storage, and the third the forecasting method. For each position, s, d, and m indicate single (32-bit), double (64-bit), and multi (512-bit) precision, respectively. Multiple letters in brackets indicate that the choice of any of these precisions for the given part of the processing pipeline yields almost the same results. For each precision setting and value of n, the $\mathsf{VPT}_{0.5}$ (in Lyapunov times) averaged over 100 repetitions is taken and maximized over the polynomial degree p (given in label boxes) and the data time step $\Delta t \geq 2\Delta t_0$ (not shown). The gray horizontal lines show the accuracy of RK4 ODE solvers in different precision setups for reference

RK4. Based on 95% confidence intervals for PolyProp's VPT value from 100 repetitions, we highlight the following specific comparisons:

(a) The prediction of double-precision PolyProp trained on data ($n \geq 2^{10}$) produced by a single-precision RK4 ODE solver diverges at about the same time as the trajectories produced by a single- and a double-precision RK4 ODE solver. (Precision setting sdd or sdm, $\mathsf{VPT}_{0.5} = 12.0$ Lyapunov times).
(b) Consider ground truth data produced by a double- or multi-precision RK4 ODE solver. The prediction of double-precision PolyProp trained on ground truth data ($n \geq 2^{11}$) rounded to single precision is about as accurate as that of an RK4 ODE solver starting from single-precision initial conditions. (Precision setting dsd or msd, $\mathsf{VPT}_{0.5} = 15.7$ Lyapunov times).
(c) The prediction of multi-precision PolyProp trained on data ($n \geq 2^{13}$) produced by a double-precision RK4 ODE solver diverges at about same time as the trajectories produced by a double- and a multi-precision RK4 ODE solver. (Precision setting ddm, $\mathsf{VPT}_{0.5} = 32.1$ Lyapunov times).
(d) Consider ground truth data produced by a multi-precision RK4 ODE solver. The prediction of multi-precision PolyProp trained on ground truth data ($n \geq 2^{12}$) rounded to double precision is at least as accurate as that of an RK4 ODE solver starting from double-precision initial conditions. (Precision setting mdm, $\mathsf{VPT}_{0.5} = 34.7$ Lyapunov times for the ODE solver and $\mathsf{VPT}_{0.5} = 35.7$ Lyapunov times for PolyProp).

The impact of numerical accuracy in solving linear systems is also evident when comparing different data normalization schemes for single- and double-precision methods (see Supplementary Section 3). In general, normalization enhances results, with more sophisticated schemes outperforming default approaches.

Without artificially limiting the precision of the training data, much higher VPT values can be achieved, see Fig. 7. The limit of about 105 Lyapunov times shown in the plot is likely due to the constraint $p \leq 16$, with even higher VPT values possible for higher degrees. Note that with the data time step equal to the solver time step, i.e., $\Delta t = 2^{-10} = \Delta t_0$, extremely high VPT values are achieved with degree 8 polynomials (more than 322 Lyapunov times for $n = 2^{15}$; not shown in the plot).

This can be explained as follows: applying k steps of the RK4 ODE solver to the L63 model yields a polynomial propagator of degree F_{4k+2}, where F_ℓ is the ℓ-th Fibonacci number.

This is shown in Supplementary Section 4. Thus, one RK4 solver step of the L63 system amounts to a polynomial propagator of degree $F_6 = 8$. Hence, the fits can become near perfect in this case if $p \geq 8$. This is not a concern for larger time steps, as two RK4 solver steps for L63 already require degree $F_{10} = 55$ to be fully represented as a polynomial.

Further experiments using high-precision ODE solvers are detailed in Supplementary Section 5. Notably, when using a Taylor Integrator with extreme precision (absolute tolerance 10^{-60}) as the ground truth, PolyProp—trained and initialized with rounded 64-bit data—achieves a VPT of 35.6 Lyapunov times. This performance is statistically indistinguishable from that of the Taylor Integrator itself when initialized with a rounded 64-bit state (35.8 Lyapunov times).

Finally, Supplementary Section 6 investigates PolyProp's performance on noisy data, viewing observational noise as a stochastic limitation on precision analogous to deterministic rounding. Consistent with the findings in Fig. 6, PolyProp's forecast accuracy matches that of the ODE solver when both are initialized with the same noisy state. This demonstrates that PolyProp achieves the optimal predictive fidelity possible when no data assimilation is applied to denoise the initial state.

3.3 Thomas' cyclically symmetric attractor

One may still wonder whether the success of PolyProp is due to the polynomial nature of the L63 vector field and the corresponding RK4 solution (even if the degree in that case is extremely high). To remove such doubts, the machine precision results shown for L63 in the standard setting (512-bit system, 64-bit data, 512-bit method) are replicated for Thomas' Cyclically Symmetric Attractor (TCSA), a chaotic dynamical system, governed by non-polynomial dynamics given by

$$\dot{x} = \sin(y) - bx\,, \quad \dot{y} = \sin(z) - by\,, \quad \dot{z} = \sin(x) - bz\,, \tag{20}$$

where $b = 0.208$.

The results are presented in Fig. 8. Using the standard setup with up to $n = 2^{15}$ observations and a polynomial degree of up to $p = 16$, $\mathsf{VPT}_{0.5} = 25.9$ Lyapunov times is achieved. This is lower than the reference value of 37.4 obtained from a 512-bit ODE solver initialized with data rounded to 64 bit. By extending the setup to include polynomial degrees $p = 23, 24, 25$ and observation counts $n = 2^{16}, 2^{17}$ for the time step $\Delta t = 2^{-2}$, the result improves to $\mathsf{VPT}_{0.5} = 38.0$, with a 95% confidence interval of [36.8, 39.3], thereby matching the performance of the multi-precision ODE solver and outperforming the double-precision solver ($\mathsf{VPT}_{0.5} = 32.2$) when starting on initial conditions rounded to double precision.

3.4 Lorenz-96 system

To investigate how PolyProp performs in higher dimensions, consider the Lorenz-96 (L96) system [17],

$$\dot{x}_i = (x_{i+1} - x_{i-2})x_{i-1} - x_i + 8 \tag{21}$$

for $i = 1, \ldots, d$ with $x_{-1} = x_{d-1}$, $x_0 = x_d$, $x_{d+1} = x_1$. Instead of using 512-bit arithmetic to achieve double-precision (64-bit) accuracy as in the default setting, the goal here is single-precision (32-bit) machine accuracy on the Lorenz-96 system, using standard double-precision computations for PolyProp.

Single-precision machine accuracy is demonstrated in two ways: 1) a double-precision ODE solver is used to generate the ground truth, and the data is rounded to single precision before training; and 2) the data is generated directly using a single-precision solver. In both cases, the training data is single-precision, but the underlying dynamics are computed at either single or double precision.

To evaluate prediction quality, the VPT of PolyProp (executed in double precision) is compared with that of a double-precision ODE solver initialized with the single-precision data. In all tested settings, the best results from PolyProp are statistically indistinguishable from the solver's, as shown in Fig. 9 and Supplementary Section 7. This demonstrates that PolyProp reaches single-precision machine accuracy.

This holds across all tested dimensions $d = 5, \ldots, 9$. Higher dimensions require more training data, but no consistent trend in the required polynomial degree is observed. Notably, for fixed polynomial degree, higher dimensions naturally yield more input features.

3.5 Additional results

We evaluate PolyProp also on seven additional chaotic dynamical systems to illustrate the generality of our results, see Supplementary Section 8. For each setting we achieve machine precision in the same precision setups as previously used for the Lorenz-96 system.

4 Enabling factors, comparisons, and limitations

The results in the previous section raise three natural questions. Why do existing data-driven methods not reach machine precision? What allows PolyProp to achieve it? And how far does the approach scale? These are addressed in turn.

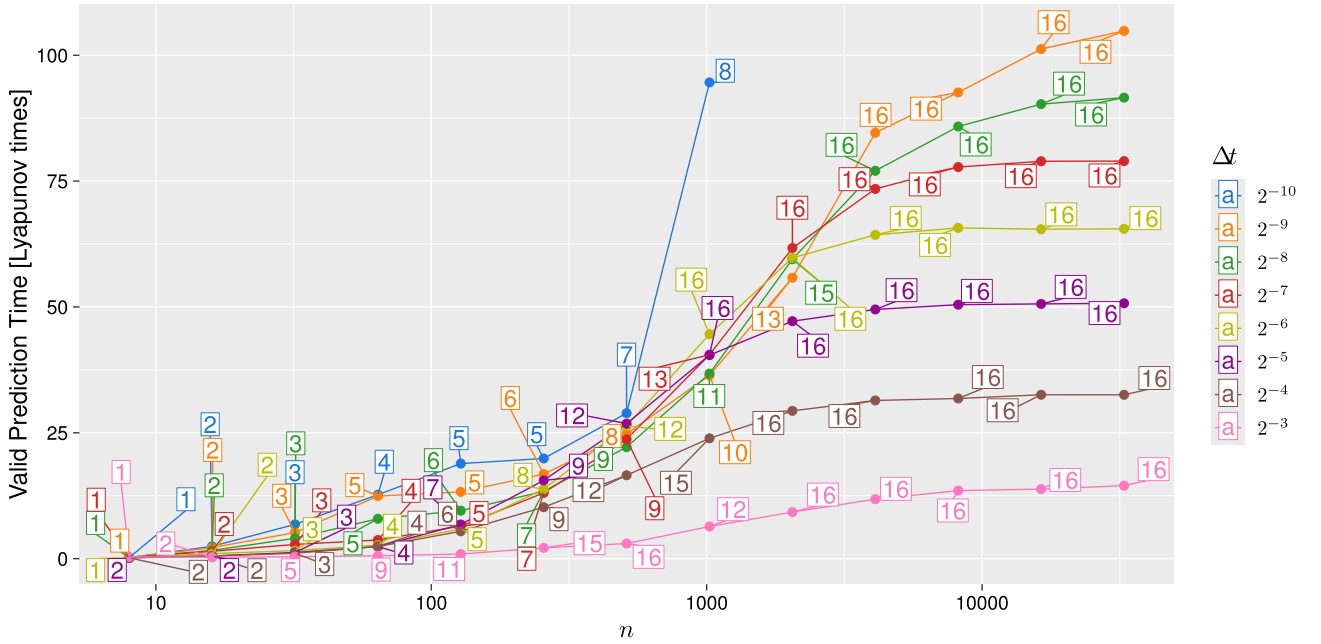

Fig. 7 Valid Prediction Time (VPT) for L63 using PolyProp with best polynomial degree for maximum precision throughout the processing pipeline. The setup uses multi-precision (512-bit) for the system, data, and method. For each value of Δt (indicated by color) and n, the $\mathrm{VPT}_{0.5}$ (in Lyapunov times) averaged over 100 repetitions is taken and maximized over the polynomial degree p (given in the label boxes). For the setting $\Delta t = \Delta t_0$ (blue), only results for $n \leq 2^{10}$ are shown. Note that the polynomial degree p is limited to $p \leq 16$

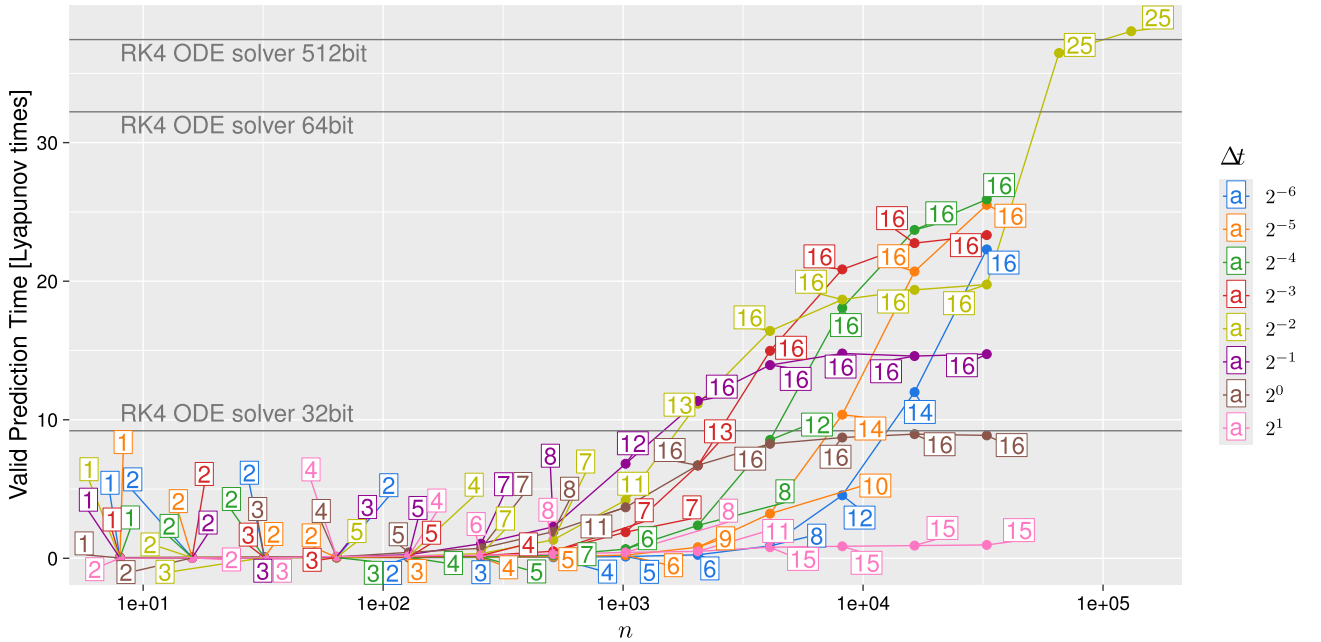

Fig. 8 Valid prediction times (VPTs) for the TCSA system using PolyProp with optimal degree under the default setup. The setup uses a multi-precision solver (512-bit), double-precision data storage (64-bit), and a multi-precision (512-bit) implementation of PolyProp. The data time step Δt is indicated by color. Label boxes within the plot denote the degree of the optimal polynomial for each case. Gray horizontal lines show the performance of ODE solvers at various precisions, using the system's governing equations and (rounded) 64-bit initial conditions. The $\mathrm{VPT}_{0.5}$ values on the vertical axis are given in Lyapunov times. In addition to the default settings ($n \leq 2^{15}$, $p \leq 16$), $p = 23, 24, 25$ and $n = 2^{16}, 2^{17}$ are added for $\Delta t = 2^{-2}$

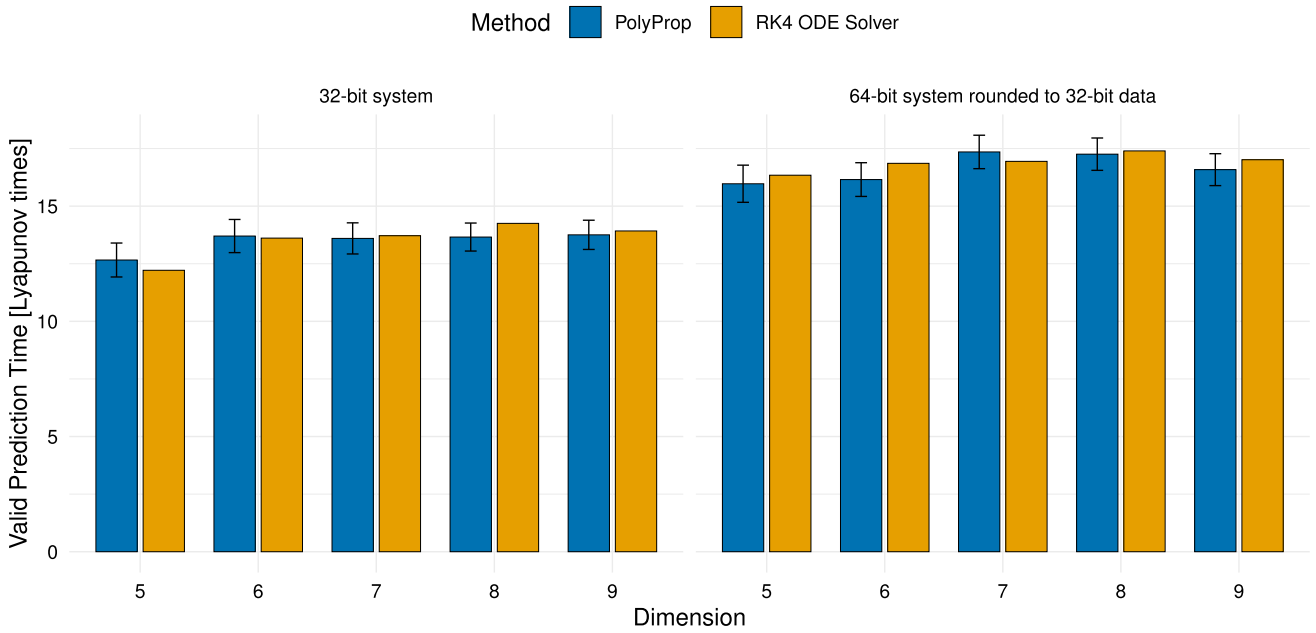

Fig. 9 Valid prediction time (VPT) for the Lorenz-96 (L96) system using PolyProp and RK4 ODE solvers. For the two precision settings and dimensions $d = 5, \ldots, 9$ for L96, the VPT (averaged over 100 repetitions) of PolyProp trained on $n = 2^{17}$ observations and time step $\Delta t = 2^{-7}$ is reported. The mean and 95% confidence interval for the optimal polynomial degree $p \in \{1, \ldots, 8\}$ are shown. These are compared to the mean VPT from 10^4 trajectories computed with a 64-bit RK4 solver initialized from 32-bit initial conditions randomly sampled from the system's attractor. The RK4 mean VPT values have 95% confidence intervals all smaller than 0.3 Lyapunov times and are not shown. All VPT values are expressed in units of the system's Lyapunov time

4.1 Limitations of existing methods

To place PolyProp in the context of existing work, three classes of approaches for forecasting low-dimensional chaotic dynamical systems that have been widely used in the literature are identified: Sparse Identification of Nonlinear Dynamical Systems (SINDy) [2], Long Short-Term Memory networks (LSTMs) [10], and Reservoir Computers [11].

To predict a system $\dot{u} = f(u)$, the SINDy algorithm uses estimates of the derivative $\dot{u}(t)$ to fit the model function f and then uses an ODE solver to create a forecast. The model function f is assumed to be a sparse linear combination of a predefined set of features. For example, in the standard version of SINDy [2], this set of features are the monomials up to degree 5. Thus, the algorithm is not fully system-agnostic, as some information about the target system must be encoded in the feature set. (In contrast, PolyProp fits the typically non-polynomial propagator $\Phi_{\Delta t}$, rather than fitting the model function f, and is effective for any sufficiently smooth ODE.) SINDy achieves a VPT of up to 13 Lyapunov times for L63 (Table 1), but would not be able to accurately recover systems such as TCSA, where f includes non-polynomial components like sine functions. Furthermore, the necessity of estimating derivatives from data introduces error (increasing with larger time steps Δt). Even if the feature set is sufficiently expressive, these derivative estimation errors prevent SINDy from achieving forecasts at or near machine precision.

LSTMs, a class of recurrent neural networks, are trained using gradient-based optimization to minimize prediction error over sequences of data. While they can, in principle, approximate complex temporal dynamics, matching the precision of ODE solvers would require their parameters to be tuned with near-exact accuracy—a level of precision that is unrealistic for gradient descent methods, which are inherently approximate and prone to local minima, saddle points, and vanishing gradients. Accordingly, the best VPT that has been achieved with LSTMs for the L63 system is 6 Lyapunov times (Table 1).

Unlike LSTMs, most Reservoir Computers, such as the Echo State Network (ESN) [11], are typically trained by solving a linear system of equations, which allows for more precise weight estimation. Similar to LSTMs, Reservoir Computers predict the next state $u(t + \Delta t)$ based on the current state $u(t)$ and an internal memory state $r(t)$, known as the reservoir. Up to 13 Lyapunov times VPT have been achieved with Reservoir Computers (Table 1). However, in the setup considered here, where the system evolves deterministically as $u(t + \Delta t) = \Phi_{\Delta t}(u(t))$, the use of an auxiliary memory is unnecessary and complicates the learning task. Moreover, RCs commonly use regularization during training to reduce variance and improve numerical stability, although some

can be trained without regularization in well-conditioned regimes [30]. While beneficial with noisy data, regularization introduces bias that limits forecasting accuracy, preventing performance at machine precision. Finally, some RCs such as the ESN rely on randomly initialized neural networks to construct features. Compared to structured polynomial bases, this random approach is less efficient for precise function approximation and, without regularization, would lead to numerical instability when the resulting features are nearly linearly dependent.

4.2 Factors enabling machine precision forecasting

The propagator $\Phi_{\Delta t}$ of a smooth ODE is itself a smooth function, and smooth functions can be approximated by polynomials—locally via Taylor's theorem, and uniformly on compact sets via the Weierstrass approximation theorem. The best-fit polynomial of a given degree can be found by least squares. Hence, with sufficient data and sufficient polynomial degree, the propagator of any smooth ODE can be approximated to arbitrary accuracy. Two key ingredients are needed to realize this in practice: high-degree polynomials and multi-precision arithmetic.

High-degree polynomials are traditionally viewed with skepticism in machine learning, often associated with overfitting. However, in the absence of noise, this concern does not apply: since the training data is noise-free, there is no noise to overfit to. In the simulation study presented here, polynomials of degree up to 25 are used effectively.

Fitting such high-degree polynomials, in turn, requires multi-precision arithmetic. To exploit the full range of significant digits provided by noise-free data, the linear systems involved in polynomial fitting must be solved in a numerically stable way. For high-degree polynomials, the design matrix $X^\top X$ in (14) becomes severely ill-conditioned [30], so that standard double-precision solvers fail to preserve these digits. Using 512-bit arithmetic provides enough margin that, after the conditioning-induced loss, a useful number of digits remains. To a smaller but non-negligible extent, 512-bit arithmetic also reduces the per-step rounding error in iterated state propagation, which contributes to the near-perfect short-term predictions visible in Fig. 5. The specific value 512 is not a tight requirement—it is simply the power of 2 closest to a tenfold increase in precision over 64 bits. Any sufficiently high precision would yield qualitatively equivalent results.

Note that modern machine learning frameworks, such as PyTorch and TensorFlow, do not support numerical precision beyond double precision (64 bit) on either CPU or GPU. Moreover, they are typically optimized for single precision (32 bit) on GPU, rendering them unsuitable for the multi-precision (512-bit) experiments conducted in this study.

4.3 Scalability and robustness

PolyProp faces challenges related to the curse of dimensionality. For a system with dimension d, the number of parameters D in a multivariate polynomial of degree p scales as $D = \mathcal{O}(d^p)$. Since fitting PolyProp requires solving a linear system with a computational complexity of $\mathcal{O}(D^3)$, the method becomes computationally prohibitive for high-dimensional settings, even with moderate polynomial degrees. Dimension reduction techniques may help to alleviate this scalability issue. Alternatively, in the context of Partial Differential Equations, one may apply localization strategies—using only local spatial patches as input—to reduce the effective input dimension [19].

While this study demonstrates how extremely accurate predictions of chaotic systems can be achieved, the main simulation setup is clearly artificial. In practical scenarios, noise-free observations with the 15-digit precision characteristic of 64-bit floating-point arithmetic are rare, found only in specialized domains such as atomic clock measurements (up to 10^{-18} relative precision) Nicholson et al. [21]. Far more commonly, measurements are corrupted by noise, rendering even leading digits unreliable.

In such settings, PolyProp yields an optimal result among non-denoising methods, matching the forecast accuracy of numerical ODE solvers initialized with noisy data, as shown in Supplementary Section 6. Yet, despite this relative optimality, forecast accuracy deteriorates rapidly in the presence of noise—a limitation shared by many other learning techniques for dynamical systems [32].

5 Conclusion

This study has shown that system-agnostic polynomial regression, trained with suitable numerical precision on sufficiently accurate data, can match the accuracy of ODE solvers when predicting chaotic dynamics from the same initial state. This result was explicitly demonstrated for the 3-dimensional polynomial Lorenz-63 system, the 3-dimensional non-polynomial Thomas Cyclically Symmetric Attractor, and the Lorenz-96 system in dimensions up to $d = 9$. These findings suggest that PolyProp generalizes to a broad class of dynamical systems governed by autonomous first-order ODEs. From this perspective, the problem of forecasting chaotic systems from noise-free data may be considered effectively solved, and standard benchmarks such as the noise-free Lorenz-63 model are no longer adequate for distinguishing methods—they have become trivial under these conditions.

The cornerstones of PolyProp—multi-precision arithmetic and high-degree polynomials—are well suited to the noise-free setting but ill-suited to overcome noise, which

remains the dominant obstacle in practical applications. Matching forecast accuracy under noise requires denoising of the initial conditions rather than higher precision or higher polynomial degree. Given that noise is ubiquitous in real-world data, future research on learning dynamical systems should focus on methods that inherently integrate denoising capabilities.

Supplementary Information The online version contains supplementary material available at https://doi.org/10.1007/s11071-026-12915-9.

Acknowledgements The authors gratefully acknowledge the Ministry of Research, Science and Culture (MWFK) of Land Brandenburg for supporting this project by providing resources on the high performance computer system at the Potsdam Institute for Climate Impact Research.

Author Contributions C.S. developed the methodology, designed and carried out the simulation study, analyzed the simulation results, and wrote the original draft of the manuscript. N.B. contributed to the design of the simulation study, to structuring the presentation of the method and simulation results, and revised the manuscript. Both authors discussed the results.

Funding Open Access funding enabled and organized by Projekt DEAL. N.B. acknowledges funding by the Volkswagen Foundation. This is ClimTip contribution #184; the ClimTip project has received funding from the European Union's Horizon Europe research and innovation programme under grant agreement No. 101137601.

Data Availability The simulation results and source code supporting the findings of this study are archived at Zenodo https://doi.org/10.5281/zenodo.18184265. The source code is also available via GitHub at https://github.com/chroetz/PolyProp.

Declarations

Conflict of interest The authors declare no conflict of interest.

References

1. Akiyama, T., Tanaka, G.: Computational efficiency of multi-step learning echo state networks for nonlinear time series prediction. IEEE Access **10**, 28535–28544 (2022). https://doi.org/10.1109/access.2022.3158755
2. Brunton, S.L., Proctor, J.L., Kutz, J.N.: Discovering governing equations from data by sparse identification of nonlinear dynamical systems. Proc. Natl. Acad. Sci. **113**(15), 3932–3937 (2016). https://doi.org/10.1073/pnas.1517384113
3. de Silva, B.M., Champion, K., Quade, M., Loiseau, J.-C., Kutz, J.N., Brunton, S.L.: Pysindy: a python package for the sparse identification of nonlinear dynamical systems from data. J. Open Source Softw. **5**(49), 2104 (2020). https://doi.org/10.21105/joss.02104
4. Dubois, P., Gomez, T., Planckaert, L., Perret, L.: Data-driven predictions of the Lorenz system. Physica D **408**, 132495 (2020). https://doi.org/10.1016/j.physd.2020.132495. (**ISSN 0167-2789**)
5. Elsner, J.B., Tsonis, A.A.: Nonlinear prediction, chaos, and noise. Bull. Am. Meteor. Soc. **73**(1), 49–60 (1992)
6. Fousse, L., Hanrot, G., Lefèvre, V., Pélissier, P., Zimmermann, P.: MPFR: a multiple-precision binary floating-point library with correct rounding. ACM Trans. Math. Softw. **33**(2), 13-es (2007). https://doi.org/10.1145/1236463.1236468
7. Gauthier, D.J., Bollt, E., Griffith, A., Barbosa, W.A.S.: Next generation reservoir computing. Nat. Commun. **12**(1), 5564 (2021). https://doi.org/10.1038/s41467-021-25801-2. (**ISSN 2041-1723**)
8. Gergonne, A.: Application de la méthode des moindres quarrés à l'interpolation des suites. Ann. Math. Pures Appl., **6**, 242–252, (1815-1816). ISSN 1764-7843. https://www.numdam.org/item/AMPA_1815-1816__6__242_0/
9. Griffith, A., Pomerance, A., Gauthier, D.J.: Forecasting chaotic systems with very low connectivity reservoir computers. Chaos: Interdisc. J. Nonlinear Sci. **29**(12), 123108 (2019). https://doi.org/10.1063/1.5120710. (**ISSN 1054-1500**)
10. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural Comput. **9**(8), 1735–1780 (1997). https://doi.org/10.1162/neco.1997.9.8.1735. (**ISSN 0899-7667**)
11. Jaeger, H.: The "echo state" approach to analysing and training recurrent neural networks-with an erratum note. Bonn, Germany: German National Research Center for Information Technology GMD Technical Report **148**(34), 13 (2001)
12. Jaurigue, L.: Chaotic attractor reconstruction using small reservoirs—the influence of topology. Mach. Learn.: Sci. Tech. **5**(3), 035058 (2024). https://doi.org/10.1088/2632-2153/ad6ee8. (**ISSN 2632-2153**)
13. Köster, F., Patel, D., Wikner, A., Jaurigue, L., Lüdge, K.: Data-informed reservoir computing for efficient time-series prediction. Chaos: Interdisc. J. Nonlinear Sci. **33**(7), 073109 (2023). https://doi.org/10.1063/5.0152311. (**ISSN 1054-1500**)
14. Lam, R., Sanchez-Gonzalez, A., Willson, M., Wirnsberger, P., Fortunato, M., Alet, F., Ravuri, S., Ewalds, T., Eaton-Rosen, Z., Hu, W., Merose, A., Hoyer, S., Holland, G., Vinyals, O., Stott, J., Pritzel, A., Mohamed, S., Battaglia, P.: Learning skillful medium-range global weather forecasting. Science **382**(6677), 1416–1421 (2023). https://doi.org/10.1126/science.adi2336. (**ISSN 1095-9203**)
15. Li, X., Zhu, Q., Zhao, C., Duan, X., Zhao, B., Zhang, X., Ma, H., Sun, J., Lin, W.: Higher-order granger reservoir computing: simultaneously achieving scalable complex structures inference and accurate dynamics prediction. Nat. Commun. **15**(1), 3506 (2024). https://doi.org/10.1038/s41467-024-46852-1. (**ISSN 2041-1723**)
16. Lorenz, E.N.: Deterministic nonperiodic flow. J. Atmos. Sci. **20**(2), 130–141 (1963)
17. Lorenz, E.N.: Predictability: a problem partly solved. PhD thesis, Shinfield Park, Reading (1995)
18. Lu, Z., Hunt, B.R., Ott, E.: Attractor reconstruction by machine learning. Chaos: Interdisc. J. Nonlinear Sci. **28**(6), 061104 (2018). https://doi.org/10.1063/1.5039508. (**ISSN 1054-1500**)
19. Mandal, P., Gottwald, G.A.: Learning dynamical systems with hit-and-run random feature maps. Nat. Commun. **16**(1), 5961 (2025). https://doi.org/10.1038/s41467-025-61195-1. (**ISSN 2041-1723**)
20. Nakata, M.: Mplapack version 2.0.1 user manual (2022). arXiv:2109.13406
21. Nicholson, T., Campbell, S., Hutson, R., Marti, G., Bloom, B., McNally, R., Zhang, W., Barrett, M., Safronova, M., Strouse,

G., Tew, W., Ye, J.: Systematic evaluation of an atomic clock at 2×10^{-18} total uncertainty. Nature Commun. **6**(1), 6896 (2015). https://doi.org/10.1038/ncomms7896
22. Ott, E.: Chaos in dynamical systems, second edition Cambridge University Press, Cambridge (2002). https://doi.org/10.1017/CBO9780511803260. (**ISBN 0-521-01084-5**)
23. Pathak, J., Lu, Z., Hunt, B.R., Girvan, M., Ott, E.: Using machine learning to replicate chaotic attractors and calculate Lyapunov exponents from data. Chaos: Interdisc. J. Nonlinear Sci. **27**(12), 121102 (2017). https://doi.org/10.1063/1.5010300. (**ISSN 1054-1500**)
24. Pathak, J., Wikner, A., Fussell, R., Chandra, S., Hunt, B.R., Girvan, M., Ott, E.: Hybrid forecasting of chaotic processes: Using machine learning in conjunction with a knowledge-based model. Chaos: Interdisc. J. Nonlinear Sci. **28**(4), 041101 (2018). https://doi.org/10.1063/1.5028373. (**ISSN 1054-1500**)
25. Platt, J.A., Penny, S.G., Smith, T.A., Chen, T.-C., Abarbanel, H.D.: A systematic exploration of reservoir computing for forecasting complex spatiotemporal dynamics. Neural Netw. **153**, 530–552 (2022). https://doi.org/10.1016/j.neunet.2022.06.025. (**ISSN 0893-6080**)
26. Price, I., Sanchez-Gonzalez, A., Alet, F., Andersson, T.R., El-Kadi, A., Masters, D., Ewalds, T., Stott, J., Mohamed, S., Battaglia, P., Lam, R., Willson, M.: Probabilistic weather forecasting with machine learning. Nature **637**(8044), 84–90 (2024). https://doi.org/10.1038/s41586-024-08252-9. (**ISSN 1476-4687**)
27. R Core Team. R: a language and environment for statistical computing. R Foundation for Statistical Computing, Vienna (2024). https://www.R-project.org/
28. Ren, H., Chou, J., Huang, J., Zhang, P.: Theoretical basis and application of an analogue-dynamical model in the Lorenz system. Adv. Atmos. Sci. **26**(1), 67–77 (2009)
29. Roberts, D.: Neural networks for Lorenz map prediction: a trip through time (2020). arXiv:1903.07768
30. dos Santos, E.R., Bollt, E.: On the emergence of numerical instabilities in next generation reservoir computing. Chaos: Interdisc J. Nonlinear Sci. **35**(12) (2025). https://doi.org/10.1063/5.0278709
31. Sanderson, C., Curtin, R.: Armadillo: an efficient framework for numerical linear algebra. In: 2025 17th International Conference on Computer and Automation Engineering (ICCAE), pp. 303–307 (2025). https://doi.org/10.1109/ICCAE64891.2025.10980539
32. Schötz, C., White, A., Gelbrecht, M., Boers, N.: Machine learning for predicting chaotic systems (2025). arXiv:2407.20158
33. Steinegger, J., Räth, C.: Predicting three-dimensional chaotic systems with four qubit quantum systems. Sci. Rep. **15**(1), 6201 (2025). https://doi.org/10.1038/s41598-025-87768-0. (**ISSN 2045-2322**)
34. Stigler, S.M.: Gergonne's 1815 paper on the design and analysis of polynomial regression experiments. Historia Math. **1**, 431–447 (1974). https://doi.org/10.1016/0315-0860(74)90033-0. (**ISSN 0315-0860**)
35. Strogatz, S.: Nonlinear Dynamics and Chaos, third edition edition Chapman and Hall/CRC, New York (2024). https://doi.org/10.1201/9780429398490. (**ISBN 978-0-367-02650-9**)
36. Thomas, R.: Deterministic chaos seen in terms of feedback circuits: analysis, synthesis, "labyrinth chaos". Int. J. Bifurcation Chaos **09**(10), 1889–1905 (1999). https://doi.org/10.1142/S0218127499001383
37. Viehweg, J., Worthmann, K., Mäder, P.: Parameterizing echo state networks for multi-step time series prediction. Neurocomputing **522**, 214–228 (2023). https://doi.org/10.1016/j.neucom.2022.11.044. (**ISSN 0925-2312**)
38. Viswanath, D.: The fractal property of the Lorenz attractor. Phys. D **190**(1–2), 115–128 (2004). https://doi.org/10.1016/j.physd.2003.10.006. (**ISSN 0167-2789**)
39. Wang, R., Kalnay, E., Balachandran, B.: Neural machine-based forecasting of chaotic dynamics. Nonlinear Dyn. **98**(4), 2903–2917 (2019). https://doi.org/10.1007/s11071-019-05127-x. (**ISSN 1573-269X**)
40. Yu, R., Zheng, S., Anandkumar, A., Yue, Y.: Long-term forecasting using higher order tensor rnns, (2019). arXiv:1711.00073