https://doi.org/10.1038/s41559-022-01787-y

Supplementary information

Chaos is not rare in natural ecosystems

Supplementary Information
"Chaos is not rare in natural ecosystems"

Tanya L. Rogers, Bethany J. Johnson, Stephan B. Munch

Supplementary Notes

1 Time-delay embedding and methods for selecting E and τ

Takens' theorem[42] provides a foundation for the analysis of time series from nonlinear systems. It demonstrates that under fairly generic conditions it is possible to reconstruct a system's dynamics using lagged observations from one (or a subset) of the state variables. Specifically, Takens[42] shows that for an *M*-dimensional dynamical system that converges to an attractor *A* of dimension $d < M$ that $\{x_t, x_{t-\tau}, x_{t-2\tau}, \dots, x_{t-E\tau}\}$ is an embedding of *A* provided that $E > 2d$. What this means in practice is that for some variable *x* we can write $x_t = f(x_{t-\tau}, x_{t-2\tau}, \dots, x_{t-E\tau})$, use data on *x* to estimate f with some non-parametric regression or machine learning method (e.g.[83,86]), and the function f will retain all of the properties of the original system of which *x* is part (including chaos). Here, the embedding dimension *E* is the number of lags needed in *f*, which is one less than the traditional definition in the dynamics literature. We define *E* this way to provide a more intuitive comparison with the number of inputs in parametric models. This reconstruction of the state space from lags of a single time series is called time-delay embedding. The models for f most commonly used for ecological time series are piecewise constant (simplex)[29], locally linear (s-map)[43], Gaussian processes[86], and neural networks[31]. For computational speed and relative ease of implementation, we focused on s-map, modified as described below.

Of course, we usually do not know d or M, and some means of empirically determining the time delay τ and embedding dimension *E* is required. The direct LE method (DLE), Jacobian LE method (JLE), and recurrence quantification analysis (RQA) all employ time-delay embedding to reconstruct the state space, and so rely on a choice of embedding parameters E and τ. To allow for a fair comparison, we used the same E and τ values in all 3 methods.

There are several standard and widely-used approaches for selecting optimal embedding parameters, including mutual information[87] or autocorrelation for selecting τ, and false nearest neighbors[88] or simplex projection[29] for selecting *E*. Since poor choices of τ and *E* can lead to incorrect conclusions about dynamics[89], we began by testing the effectiveness of several different procedures. We generated an *embedding dataset* consisting of 20 replicate time series (length *T* = 100) from each of 9 different chaotic models (Supplementary Table 3) with prescribed values of E and τ (each ranging from 1-3) and tested the ability of these procedures to correctly identify them. A description of the procedures tested and their performance is given below. Although mutual information is widely used in nonlinear time series analysis, it was not considered here because the data requirements, on the order of 10^4 observations, are far too high for ecological data[90]. The modified s-map regression method (Supplementary Note 1.3) was the most accurate and so was used for the remainder of the analyses.

We note here that although Takens' theorem was originally derived for deterministic systems, it may provide a reasonable approximation for stochastic systems, although Van Kampen[91] showed that infinite memory is the general case for incompletely observed stochastic systems. Stark et al.[92,93], extended Takens' theorem to forced and stochastic systems, albeit with either severe constraints on admissible forcing (e.g. periodic drivers) or additional data requirements. Ragwitz and Kantz[94,95] showed that delay embedding for stochastic systems can sometimes be made explicitly Markovian, and Munch et al.[28] showed that delay embedding accurately reconstructed the conditional expectation for a class of stochastic population models, though neither of these results are generic. Ragwitz and Kantz[94,95] point out that although not exact, error from the finite-memory assumption in delay embedding for a stochastic nonlinear system is often small relative to the other sources of error.

1.1 Autocorrelation/partial autocorrelation

One method to select τ is to use the first zero crossing of the autocorrelation function [96,97] because this makes the coordinates $x_t, x_{t-\tau}$ linearly uncorrelated. In our analysis, we selected the first time lag at which either the autocorrelation function (ACF) or the partial autocorrelation function (PACF) switched from positive to negative, taking the smaller lag when they differed. When tested on the embedding dataset, this method accurately identified τ when its true value was 1 approximately 95% of the time but tended to underestimate τ when its true value was greater than 1. This is likely because the ACF/PACF method is intended for continuous time series and does not extend easily to discrete time series.

1.2 False nearest neighbor algorithm

The method of false nearest neighbors is commonly used to select the embedding dimension[88]. The basic idea is that if two points are true neighbors in a space of dimension E, then they will still be close in $E + 1$ dimensions. For each point in the time series, we found the closest neighbor in E dimensions and calculated the ratio of distances between the points in E and $E + 1$ dimensions. Neighbors were classified as false if the ratio exceeded R_{tol}(= 15). Starting with $E = 1$, E is then increased until the proportion of false nearest neighbors is sufficiently close to zero or we reach some maximum E. In our simulations, this method consistently overestimated E even given the correct τ. This could be due to the method's sensitivity to the choice of R_{tol}, and it is difficult to choose a value that works well for many different systems.

1.3 Simplex and s-map regression

Another method for selecting embedding parameters is to fit models using multiple combinations of E and τ and to select those values that maximize the out-of-sample model fit[98]. We used leave-one-out prediction R^2 as our measure of model fit. Sugihara[43] recommends using simplex (i.e. a piecewise constant model) to select E and τ because it has no additional parameters. Unfortunately, multiple combinations of E and τ can give nearly identical model fits. For example, a periodic time series with a period of 4 can be perfectly described with E/τ combinations of 2/1, 4/1, 2/2, 1/4, 1/8, etc. Any apparent differences in model fit are due to numerical rounding error or observation noise, and algorithms which selected an apparent maximum gave inconsistent results across replicates of the same model.

To alleviate this problem, we made two modifications of the algorithm which substantially improved performance both for embedding parameter selection and chaos classification. The first was to evaluate model fit using local linear regression (s-map), rather than simplex, over a constrained grid of E, τ, and the local weighting parameter θ. Since the identifiable E scales as the square root of time series length[99] we considered E and τ ranging from 1 to 6, requiring $E^2 \leq T$ and $E\tau/T \leq 0.2$, and for each E and τ combination, considered 12 values of θ: 0, 0.1, 0.3, 0.5, 0.75, 1, 1.5, 2, 3, 4, 6, 8. The second modification was to ignore differences in prediction $R^2 < 0.01$ and to select from among equivalent "best" models the one that has the lowest τ, then θ, then E. This prioritizes simpler, more linear models, generally avoids models with no dynamics (e.g. $E = 1$ and τ equal to period length), and selects an optimal value for θ along with E and τ. These modifications together resulted in 100% correct identification of E and τ in the embedding dataset. It also led to more consistent identification of E and τ values across replicates in the test dataset, as well as more accurate classification of dynamics, particularly for periodic time series.

Recently, Cenci et al.[100] introduced the "regularized s-map" and showed that this method improved forecasting and Jacobian estimation. However a major conclusion of [100] was that the best weighting kernel and regularization depended on both the simulation model and whether one is trying to forecast or infer Jacobians. Unfortunately, conditions that optimize forecasting introduce bias in Jacobian estimation. In light of this, we used s-map without explicit regularization. However, we note that the selection criteria we used to choose among models with statistically negligible differences in forecast performance (i.e. favoring low E, low θ models) serves a very similar purpose and performed well in extensive simulations.

2 Chaos detection methods

2.1 Direct LE estimation (DLE)

The dominant Lyapunov exponent (LE), typically denoted as λ, is the most commonly used indicator of chaos[36]. The LE generalizes the dominant eigenvalue for linear systems and measures the rate at which nearby trajectories diverge or converge, averaged over the attractor. Specifically, an initial infinitesimal perturbation, $\Delta x(0)$, will grow or shrink approximately exponentially as $||\Delta x(t)|| \sim e^{\lambda t}||\Delta x(0)||$ where ‖ ‖ is the Euclidean distance between points. Positive LEs are indicative of chaos (sensitive dependence on initial conditions) while negative LEs are indicative of stable (non-chaotic) dynamics.

This formal definition only holds in the limit as $\Delta x(0) \longrightarrow 0$. In multidimensional systems, the initial growth rate will be less than λ unless the initial perturbation happens to be in the direction of greatest growth. Moreover, since the attractor is bounded, two trajectories cannot get infinitely far apart and so the period of exponential growth will be finite. Nevertheless, it is possible, using this definition, to estimate the LE directly from data by measuring the divergence rate of nearest neighbors over a finite time horizon (the "direct" LE method)[37,101]. In ecology, this approach has primarily been used to characterize results for experimental systems (e.g.[12,48,50]).

For each point in the time series, we found its nearest neighbor in delay embedding space (with embedding dimension E and lag τ) and followed all pairs forward in time, computing the distance (d) between them at each step. We regressed the log of the mean distance *t*-steps ahead, $\ln[d(t)]$, on the number of timesteps into future, *t*:

$$\ln[d(t)] = \ln[d(0)] + \lambda t + error$$

and use the slope, λ, as our estimate of the Lyapunov exponent. Since the attractor is bounded, the distance between points will ultimately saturate, so the maximum *t* must be set relatively low to avoid underestimating λ. In our implementation, we used *t* up to 4 steps into the future. Although this method does not depend on the details of the underlying dynamics and is fairly robust to choices of E and τ, it is known to be sensitive to noise[38]. Therefore, to ensure a conservative estimate of the frequency of chaos, only time series with p(λ>0.01)>0.975 were classified as chaotic, i.e. λ−1.96×SE(λ) > 0.01.

These analyses were performed in R version 3.6.3[102].

2.2 Jacobian LE estimation (JLE)

The LE may also be estimated by fitting a delay embedding model (with embedding dimension E and lag τ) of the form

$$x_t = f(x_{t-\tau}, x_{t-2\tau}, \ldots, x_{t-E\tau})$$

to the available time series and computing the LE from the model's Jacobian matrices[30,31]. A variety of methods may be used to estimate *f* (e.g. polynomials, splines, GAMs, neural networks, Gaussian processes). Following Ushio et al.[103], we used local linear regression (s-map)[43] to estimate *f*, obtaining embedding parameters E and τ and local weighting parameter θ as described in Supplementary Note 1.3.

In order to find the best description of the dynamics, we fit 3 different forms of the delay embedding model (first difference as a function of abundance, population growth rate as a function of abundance, and population growth rate as a function of log abundance; Supplementary Table 4) and selected the one with the best leave-one-out prediction R^2 for abundance. Previous meta-analyses using delay embedding (e.g.[30]) did not perform this model selection step, which may, in part explain the difference in our results.

For simulation models that did not generate strictly positive values, only the first model was fit; all 3 were considered for the remaining simulation models and all of the empirical data. We had also considered models of abundance as a function of abundance and log abundance as a function of log abundance, but these produced identical abundance predictions and LE estimates as models using first difference as a function of abundance, and population growth rate as a function of log abundance, respectively. We opted for the forms we used because they allowed us to also compute the leave-one-out prediction R^2 for growth rate or first difference. Although this quantity was not used to select the best model form, it offered another measure of predictability.

Although not strictly necessary, Jacobian matrices for all models were formulated in terms of abundance (as opposed to log abundance or growth rate). The Jacobians are constructed from the local regression coefficients (partial derivatives) from the model and (depending on the model formulation) the predicted growth rate and/or past observations of abundance.

The LE, λ, is computed by multiplying sequential Jacobian matrices and taking the log absolute value of the dominant eigenvalue (Λ_1) of this product. Formally, the LE is defined (and will converge) in the limit as $T \to \infty$, but for finite time series, it is calculated over the available data.

$$\lambda = \frac{1}{T} \ln \left| \Lambda_1 \left(\prod_{t=1}^{T} J(x_t) \right) \right|$$

In cases where $\tau > 1$, Jacobians every τ steps were multiplied, the LEs were divided by τ, and then the τ different LEs were averaged. The multiplication and eigen decomposition were done using the QR procedure for numerical stability. For a 1-d system (E = 1), the LE is simply the arithmetic mean of the log absolute value of the derivatives.

$$\lambda = \frac{1}{T}\sum_{t=1}^{T} \ln\left|\frac{\partial f}{\partial x_t}\right|$$

Estimation of the LE comes with some uncertainty, and for chaos classification, we seek to evaluate whether the LE, given this uncertainty, is significantly greater than 0. Unfortunately, the best method for computing confidence intervals for LEs, particularly at small sample sizes, remains an unresolved question. A variety of methods can be found in the literature[12,104–109], but there has not be a systematic evaluation of performance, hypothesis testing, and coverage probabilities for these methods. Therefore, we evaluated the ability of 4 methods for JLE confidence interval generation to accurately classify the simulated chaotic and non-chaotic time series in the test dataset (see Supplementary Note 3). All classify a series as chaotic if p(LE>0)>0.95. For comparison, we also included classification based on a point estimate (no confidence interval, method 1).

1. Point estimate computed from the full time series (no confidence interval).
2. The residual bootstrap method[12]. We did not refit the embedding parameters for each bootstrap as this would have created impractically long computation times. We used 1000 replicates.
3. The asymptotic LE standard error method described by Shintani and Linton[105].
4. A moving window method in which LEs were computed over several long sub-segments of Jacobians from the best model. The mean and standard deviation of the resulting LEs were used to compute an interval.
5. An importance sampling method in which local regression coefficients are re-drawn from the time point-specific multivariate normal distributions defined by the local least-squares estimates and covariance matrices. This is similar to the method presented in [104], but adapted for use with our local linear model. We used 1000 replicates.

We found that the most accurate classifications were obtained by using method 4 (Supplementary Table 2). This method was based on the fact that local LEs computed from segments of a time series converge to the global LE as segment length increases. For sufficiently long segments, local LEs are approximately normally distributed around the global LE with variance inversely proportional to segment length[110]. Specifically, we took the full Jacobian sequence of length $n = T - E\tau$, and computed LEs for all $i + 1$ possible sequences of length $n - i$ for $i = 3,4,5,6$. For each value of i, we then computed means, standard errors, and one-tailed 95% confidence intervals. As a further buffer against false positives, we took the minimum lower bound of these i intervals as our lower bound on the LE, and if this value was >0.01, the time series was classified as chaotic. Since we were primarily using this interval as a classifier (as opposed to a statistical statement about plausible values for the "true" LE), and this method performed much better than the other interval generation methods in our simulations (Supplementary Table 2), particularly for periodic series, we used this method in subsequent analyses.

For empirical time series that contained missing values (35% of GPDD series), we fit the model using all data (skipping over delay vectors with missing elements), but computed the LE only over the longest string of consecutive, non-missing values, which encompassed 73% of non-missing data points on average (range: 35-98%), and was 35 timepoints on average (range: 12-108). Since we are using variability in LEs over subsegments to establish approximate confidence intervals, classification for shorter series is likely to be conservative. Consistent with this, all time series with a longest string <30 (8% of GPDD series) were classified as not chaotic.

For the empirical time series, we also tested sensitivity to the selection of embedding parameters by refitting each model but with E or τ fixed to be either 1 greater or less than the value in the best model, provided those values were within the ranges used in model selection. The local weighting parameter θ was refit. These changes resulted in a somewhat lower overall percent chaotic (23-29%), and also greatly decreased mean and median R^2 values (Supplementary Table 5). Proportionally, most of the changes in classification occurred in chaotic series with E=1 or 2.

Readers may reasonably question the validity of LEs computed from time series with as few as 30 data points. While it is true that one needs 10^4 or more data points to get a *precise* estimate of LE, our primary ecological interest was in determining (conservatively) whether or not the LE was greater than 0, a task which may require considerably less data. The purpose of our simulation work was to establish whether or not the *sign* of the LE could be accurately determined, given limited data and noise.

These analyses were performed in R version 3.6.3[102]. The s-map models were fit using the package 'rEDM' version 0.7.4[111].

2.3 Recurrence quantification analysis (RQA)

Recurrence quantification analysis (RQA) is based on the notion that most deterministic systems tend to revisit regions of state space[112]. RQA has been used to test for chaos in physiological and financial data[32,113–117], but has only rarely been used in ecology[118–120].

RQA begins with the construction of the recurrence matrix, R, from the observed time series $\overrightarrow{x_t}$ for $t = 1, \dots, T$. An entry of the $T \times T$ matrix, R, is 1 whenever two time points are within a threshold distance in state space and 0 otherwise. That is, $R_{t,s} = 1$ when $\|\overrightarrow{x_t} - \overrightarrow{x_s}\| \leq r$ and 0 otherwise, where || || indicates Euclidean distance. We set r equal to 0.2 times the standard deviation of the data and used time-delay embedding was used to reconstruct the state space with the same embedding parameters E and τ as used for direct and Jacobian LE estimation (Supplementary Note 1.3).

Several metrics can be derived from R [121]. After trying several, we selected 3 metrics that consistently partitioned dynamical regimes in our simulated data. Specifically, we used the RQA metrics "determinism" (*DET*), "entropy" (*ENTR*), and "average length" (L), which are based on the distribution $P(l)$ of diagonal segments of length l contained in R. Determinism[32] measures the percentage of recurrence points which form diagonal lines greater than a threshold length, l_{min}, for a time series of length T i.e.

$$DET = \frac{\sum_{l=l_{min}}^{T} lP(l)}{\sum_{l=1}^{T} lP(l)}$$

DET helps distinguish between determinism and noise. The RQA metric "entropy" is the Shannon entropy of $P(l)$, i.e.

$$ENTR = -\sum_{l=l_{min}}^{T} P(l) \ln P(l)$$

and measures the complexity of the deterministic structure. Chaotic time series tend to have higher *ENTR* than periodic time series. “Average length” is simply the mean length of diagonal line segments in R [122], i.e.

$$L = \frac{\sum_{l=l_{min}}^{T} lP(l)}{\sum_{l=l_{min}}^{T} P(l)}$$

L helps distinguish between chaotic and periodic dynamics because the reciprocal of L is related to the largest positive Lyapunov exponent[81].

Unfortunately, the ranges of *DET*, *ENTR*, and L that are best for classifying dynamics as periodic, chaotic, or stochastic are case-specific. We used the first 20 replicates of the test dataset to establish useful thresholds. Based on this analysis, a time series was classified as chaotic if $0.45 < DET < 0.99$ and $0.39 < ENTR < 2.3$ and $1.9 < L < 5.3$. These thresholds were used to classify the remaining simulations and the GPDD time series.

To test the sensitivity of our empirical results to the choice of thresholds, we performed a sensitivity analysis in which we increased or decreased the upper and/or lower thresholds for *DET*, *ENTR*, and L by 10%. Using the original thresholds, 42% of series in the GPDD were classified as chaotic. Changing the thresholds resulted in 39-46% classified as chaotic. For the empirical time series, we also tested sensitivity to the selection of the embedding parameters by using E or τ values either 1 greater or less than the value in the best model, provided those values were within the ranges used in model selection. These changes resulted in 32-49% classified as chaotic (Supplementary Table 5).

All RQA analyses were performed in MATLAB R2019a with the Cross Recurrence Plot (CRP) Toolbox version 5.22 (R32.4)[123]. We did not make any modifications to the toolbox and handled missing values according to the procedures built into the code. The CRP toolbox removes any missing values from the time series prior to analysis and performs the standard procedure on the remaining points.

2.4 Permutation entropy (PE)

Bandt and Pompe[33] introduced permutation entropy (PE) as a measure of time series complexity that can be used to quantify predictability and distinguish between periodic, chaotic, and random time series[124]. Pennekamp et al.[125] recently used PE to quantify predictability of ecological time series.

The PE of a time series x_t for $t = 1, \dots, T$ is computed by creating embedding vectors of order E given by $\overrightarrow{x_t}(E) = \{x_{t-1}, \dots, x_{t-E}\}$. There are $E!$ possible permutations corresponding to the rank order of the values. Each embedding vector in the time series maps to one of the $E!$ possible permutations. The permutation entropy of the data is determined from the observed frequency distribution of the permutations, P_i, $i = 1, \dots, E!$ as

$$PE = -\frac{\sum_{i=1}^{E!} P_i \ln P_i}{E - 1}.$$

PE tends to be high for stochastic time series, low for periodic time series, and intermediate for chaotic time series.

To use the PE as a classification tool in our analysis, we had to make choices regarding the order of embedding vectors and thresholds of classification. The rule of thumb is that E should be the largest value such that $5E! \leq N$ [126] and PE is badly negatively biased when $E!$ is large relative to time series length. To reduce this bias, given the time series lengths that we had, we fixed $E = 3$ for all simulated and empirical time series. Other studies in ecology have also used $E = 3$ on empirical time series (e.g. [125]), and it performed tolerably well in our simulation study. As with RQA, PE thresholds are operationally determined, and we used the first 20 simulations in the test set to determine thresholds for classifying time series as chaotic. Based on this, dynamics were classified as chaotic if the permutation entropy was between 1.06 and 1.23. We used this criterion to classify the remaining simulation data and empirical time series.

To test the sensitivity of our empirical results to the choice of thresholds, we performed a sensitivity analysis in which we increased or decreased the upper and/or lower thresholds by 10%. Using the original thresholds, 51% of series in the GPDD were classified as chaotic. Changing the thresholds resulted in 45-62% classified as chaotic.

The permutation entropy computations were performed in MATLAB R2019a with the Toolboxes for Complex Systems (TOCSY) *petropy* function[127,128]. This function retains missing values when creating the embedding vectors. It ranks missing values higher than the numeric values in the vectors, and if two missing values occur adjacently, the one that occurs first is given the lower rank.

2.5 Horizontal visibility graphs (HVG)

The visibility algorithm[129] constructs a mapping between a time series and a network, the horizontal visibility graph (HVG), which allows one to use network theory to characterize time series. While there have been several applications of the HVG in physiology[130], physics[131–133], and economics[134] there is, to our knowledge, no ecological application of this method.

To transform a time series into a network by means of the HVG, the following procedure is performed. Each point in the time series is a node on the visibility graph. Two nodes are connected based on whether they are “visible” to one another, i.e. if it is possible to draw a horizontal line between the two points in the time series without intersecting any points in the middle. Mathematically, this occurs when $x_t, x_s > x_n$ for all $t < n < s$. This criterion is checked for every pair of points in the time series to construct the HVG.

The visibility algorithm considers the “degree of connectivity” (k) of each node in the HVG (i.e. the number of nodes each node is connected to). The degree distribution for white noise follows $P(k) \sim \exp(-zk)$ with $z = z_{un} = \ln(3/2)$ [34]. Correlated noise and deterministic dynamics deviate from this power law distribution with $z < z_{un}$ for chaos and $z > z_{un}$ for correlated noise[135]. The Shannon entropy of the HVG degree distribution

$$h = -\sum_{k=2}^{\infty} P(k) \ln P(k)$$

behaves like the LE with $h > \ln 4$ indicating chaotic dynamics[82]. Therefore, by using a combination of the deviation of the degree distribution from the power law (mean squared error, MSE) and the entropy of the degree distribution, we can differentiate chaotic dynamics from periodic and stochastic dynamics. As with RQA and PE, we used the first 20 replicates of the test dataset to tune the thresholds of h and MSE. Based on this analysis, time series were classified as chaotic if the entropy of the HVG was between 0.32 and 0.46 and the MSE was between 0.21 and 0.48. These thresholds were used to classify the remaining simulation data.

All visibility algorithm analyses were performed in MATLAB R2019a with the Fast Horizontal Visibility Graph (HVG) for MATLAB file exchange[136].

2.6 Chaos Decision Tree (CDT)

The chaos decision tree (CDT), proposed by Toker et al.[35], is a method that combines several tools into one algorithm to detect the presence or absence of chaos in time series. It has not been applied to ecological data. The CDT has the following procedure.

1. The CDT tests for stochasticity in the data by comparing the permutation entropy of the original time series to the permutation entropy of 1000 random Amplitude Adjusted Fourier Transform surrogates[137] and 1000 Cyclic Phase Permutation surrogates[138]. If the permutation entropy of the original time series falls within either of the surrogate distributions, the data are classified as stochastic. Otherwise, the algorithm proceeds.
2. The CDT de-noises the time series with Schreiber's noise-reduction algorithm[139].
3. The algorithm checks if the data are over-sampled. If so, the time series is downsampled by a factor of 2 and the process is iterated until it is not over-sampled.
4. Finally, the CDT performs a 0-1 test for chaos[140] modified to account for observation noise[141].

In our analysis, we used all of the default settings in the publicly available MATLAB code[142] and focused on the classification output of the algorithm. The classification output of the CDT is either "stochastic," "chaotic," "periodic," or "nonstationary". We aggregated the "stochastic," "periodic," and "nonstationary" time series as "not chaotic" in order to easily compare the output to the other chaos detection methods.

Analyses were performed in MATLAB R2019a with the Chaos Decision Tree algorithm[142].

3 Simulation testing

Several of the chaos detection algorithms we used originated in the nonlinear dynamics / physics literature where they were benchmarked using relatively noise-free datasets with thousands to millions of observations. Therefore, before analyzing the GPDD time series (which had from 30 to 197 observations), we tested the accuracy of each of the 6 methods (Supplementary Note 2) on simulated datasets with limited time series lengths and relevant levels of noise. We did this with two sets of simulations which we refer to as the test and validation datasets. All simulated data were generated in MATLAB R2019a.

As the *test dataset*, we simulated data from 21 different models with periodic dynamics (Supplementary Table 6), chaotic dynamics (Supplementary Table 7), and stochastic dynamics (Supplementary Table 8). For generality, we included both ecological models and more generic models[35] for which the dynamical regimes had been previously determined. For each model, we generated time series of 5 different lengths (T = 25, 50, 75, 100, and 250) crossed with 4 different levels of white observation noise (1, 10, 20, and 30% of the standard deviation of the data). This range of observation noise was chosen to cover the range that has been empirically estimated in several natural systems, which is about 1-17% in large ungulates[143] and 10-15% in fish and plankton[144,145] on average. We simulated 100 replicate time series for each combination of model, time series length, and observation error. Each replicate was started from random initial conditions, and for the chaotic and periodic models, the first 500 timepoints were discarded to avoid transients. For models with more than one state variable, we only used data from the first variable. The observation noise was lognormal for models that produced strictly positive values, and Gaussian otherwise. For the colored noise stochastic models, we followed Toker et al.[35] and did not add observation noise to them in order to avoid interfering with the power spectra used in the CDT.

We used each of the 6 methods to classify each simulated time series as "chaotic" or "not chaotic." We used the first 20 replicates of each model/length/noise combination in the test dataset to train the detection methods. This involved tuning threshold parameters to maximize classification accuracy for the RQA, PE, and HVG methods, and making methodological modifications to maximize classification accuracy for the JLE method. For the JLE method, this included determining the best method for generating confidence intervals (Supplementary Table 2).

Without modifying the methods or tuned threshold parameters, we then applied the trained methods to the remaining 80 replicates of the test dataset. We recorded the overall classification rates for each method in aggregate and for each model, time series length, and noise level.

We also generated two *validation datasets* using independent sets of periodic, chaotic, and stochastic models. Since the purpose of the validation datasets was to evaluate the robustness of our classification rules on a completely novel set of models, the thresholds established on the test data were applied without modification. Validation dataset #1 had the same characteristics (dynamical regimes, time series lengths, levels of observation noise, number of replicates) as the test dataset, but were generated using different models (Supplementary Tables 9-11).

Note that, although the test dataset and validation dataset #1 include models with nonlinearity, process stochasticity, and observation error, it does not include models that have all three simultaneously. Therefore, to test whether classification accuracy extends to this more challenging case, we conducted as second validation test. Validation dataset #2 was constructed using 3 additional simulation models (Supplementary Tables 12-13) with both periodic and chaotic dynamics, crossed with 4 levels of lognormal observation noise (1, 10, 20, 30%) and 4 levels of process noise (0, 10, 20, 30%). Process noise was generated with random perturbations to the dynamics in the discrete time model and random perturbations to a parameter in the continuous time models, and we measured the (relative) level of process noise as $\frac{1}{T}\sum_{i=1}^{T}\frac{\mathrm{std}(f(x_i,\theta))}{f(x_i,\bar{\theta})}$ where T is the length of the time series, $f(x_i,\bar{\theta})$ is the i^{th} point in the time series with no noise and $\mathrm{std}(f(x_i,\theta))$ is the standard deviation at the i^{th} point with noise included. We simulated 100 replicates for each combination of model, dynamical regime (periodic or chaotic), observation noise, and process noise, and used $T = 100$ for all time series.

4 Simulation results

All methods performed better with lower observation noise and longer time series lengths but differed in sensitivity and overall classification accuracy. JLE, RQA, and PE were the most reliable. DLE, HVG, and CDT had misclassification rates of 0.5 or more. Classification results for the simulated test and validation datasets are presented (at various levels of aggregation) in Supplementary Figs. 1-5, Extended Data Fig. 1, Supplementary Table 1, and Table 1.

Performance was reasonably similar across individual models within a given dynamical regime (Supplementary Figs. 2, 4). The methods successfully classified long term trends (e.g. RandomWalkTrend) and seasonal dynamics (e.g. SinForcedAR1) as non-chaotic.

Performance on validation dataset #1 was broadly similar to the results for the test data set (Supplementary Table 1), indicating that the classification thresholds established during training generalize reasonably well to new simulation models. Under validation dataset #2, which included observation noise, process noise, and nonlinearity simultaneously, the performance of all methods was somewhat worse on average. Nevertheless, JLE and PE performed acceptably with overall false positive rates of 10 and 15%, respectively. To provide more context, typical values of observation and process noise in natural populations, as quantified by [143] for large ungulates, range from 0.1 to 17% for observation error (median 1.3%) and 0.6 to 25% for process error (median 2.8%). For fish and plankton time series, average estimates of observation noise are 10 to 15% [144,145]. In our simulations with 10% observation error and 30% process error, we obtained false positive rates of 0.040, 0.043, and 0.45 for JLE, PE, and RQA respectively, with corresponding false negative rates of 0.36, 0.22, and 0.28. Thus, we expect the frequency of chaos estimated for the GPDD, using JLE in particular, to be fairly conservative.

Results by method

The DLE method had the highest true positive rate but also had a very high false positive rate. This method frequently classified stochastic series as chaotic and also struggled to properly classify the periodic series. As this method is known to be sensitive to noise and to be limited to estimating positive LEs[38], this result is not unexpected.

The JLE method had the best overall performance of all of the methods and the best classification accuracy at short time series lengths. Increasing observation noise made it more likely for chaotic series to be classified as non-chaotic, with limited effect on the classification of non-chaotic series. Process noise had a similar effect and also did not increase the false positive rate of chaos detection.

RQA had a lower true positive rate than the Jacobian method and was more sensitive to time series length than to observation noise. RQA tended to misclassify periodic series as chaotic in the presence of high observation noise, but misclassification rates for stochastic series were relatively low. RQA showed the most sensitivity to process noise with higher false positive rates as process noise increased.

For PE, the false positive rate was fairly low and was relatively insensitive to observation noise and time series length. As with RQA, this method tended to misclassify periodic series as chaotic in the presence of high observation noise, but it was not highly sensitive to process noise. Stochastic series were misclassified as chaotic at short time series lengths.

HVG was insensitive to observation noise, and could correctly classify nearly all periodic series, but chaos detection was very sensitive to time series length. This method tended to classify stochastic series as chaotic at longer time series lengths, and chaotic series as non-chaotic at shorter time series lengths.

The chaos DT had the lowest false positive rate but rarely detected chaos when it was present. For long time series (T = 250), the algorithm performed well (comparably to JLE, with similar sensitivity to observation noise), but its ability to detect chaos declined rapidly as time series length decreased.

5 Classifying time series using parametric 1-d models

To test whether the restriction of model form, in addition to restriction of dimensionality, affects the inferred LE, we fit a set of 1-d models, $x_{t+1} = x_t \exp[f(x_t, \boldsymbol{q})]$, to each of the empirical (GPDD) time series, where x_t is population size and $\boldsymbol{q}$ is a vector of parameters (Extended Data Fig. 3). The set of models included those used in other meta-analyses[17–19], and all were capable of generating chaos for some values of $\boldsymbol{q}$. We estimated the parameters by minimizing

$$SS = \sum_{t=1}^{T-1} \left\{ \ln\left(\frac{x_{t+1}}{x_t}\right) - f(x_t, \boldsymbol{q}) \right\}^2$$

using fminsearch in MATLAB R2020b, ignoring zeros. In keeping with the Jacobian-based method for quantifying chaos, the LE for this model was estimated by taking the average of the absolute value of the derivative over the observed states, i.e.

$$\lambda = \frac{1}{T} \sum_{t=1}^{T} f(x_t, \boldsymbol{q}) + \ln\ |1 + x_t f'(x_t, \boldsymbol{q})|$$

where the prime denotes the derivative with respect to *x*. When process noise is present, this approach is more appropriate for characterizing dynamics than computing the LE for the deterministic skeleton[36].

The frequency of chaos in the empirical dataset was 6% or less using this set of 1-d models (Extended Data Fig. 3).

Additional references

86. Munch, S. B., Poynor, V. & Arriaza, J. L. Circumventing structural uncertainty: A Bayesian perspective on nonlinear forecasting for ecology. *Ecol. Complex.* **32**, 134–143 (2017).

87. Fraser, A. M. & Swinney, H. L. Independent coordinates for strange attractors from mutual information. *Phys. Rev. A* **33**, 1134–1140 (1986).

88. Kennel, M. B., Brown, R. & Abarbanel, H. D. I. Determining embedding dimension for phase-space reconstruction using a geometrical construction. *Phys. Rev. A* **45**, 3403–3411 (1992).

89. Marwan, N. How to avoid potential pitfalls in recurrence plot based data analysis. *Int. J. Bifurc. Chaos* **21**, 1003–1017 (2011).

90. Albers, D. J. & Hripcsak, G. Using time-delayed mutual information to discover and interpret temporal correlation structure in complex populations. *Chaos An Interdiscip. J. Nonlinear Sci.* **22**, 13111 (2012).

91. Van Kampen, N. G. *Stochastic processes in physics and chemistry*. vol. 1 (Elsevier, 1992).

92. Stark, J. Delay Embeddings for Forced Systems. I. Deterministic Forcing. *J. Nonlinear Sci.* **9**, 255–332 (1999).

93. Stark, J., Broomhead, D. S., Davies, M. E. & Huke, J. Delay Embeddings for Forced

Systems.II. Stochastic Forcing. *J. Nonlinear Sci.* **13**, 519–577 (2003).

94. Ragwitz, M. & Kantz, H. Markov models from data by simple nonlinear time series predictors in delay embedding spaces. *Phys. Rev. E* **65**, 56201 (2002).

95. Kantz, H. & Ragwitz, M. Phase space reconstruction and nonlinear predictions for stationary and nonstationary Markovian processes. *Int. J. Bifurc. Chaos* **14**, 1935–1945 (2004).

96. Abarbanel, H. D. I. *Analysis of Observed Chaotic Data*. *Institute for Nonlinear Science* (Springer New York, 1996).

97. Kantz, H. & Schreiber, T. *Nonlinear Time Series Analysis*. (Cambridge University Press, 2003).

98. Garland, J., James, R. G. & Bradley, E. Leveraging information storage to select forecast-optimal parameters for delay-coordinate reconstructions. *Phys. Rev. E* **93**, 22221 (2016).

99. Cheng, B. & Tong, H. On consistent nonparametric order determination and chaos. *J. R. Stat. Soc. Ser. B* **54**, 427–449 (1992).

100. Cenci, S., Sugihara, G. & Saavedra, S. Regularized S-map for inference and forecasting with noisy ecological time series. *Methods Ecol. Evol.* **10**, 650–660 (2019).

101. Kantz, H. A robust method to estimate the maximal Lyapunov exponent of a time series. *Phys. Lett. A* **185**, 77–87 (1994).

102. R Core Team. R: A language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria. URL https://www.R-project.org/. (2019).

103. Ushio, M. *et al.* Fluctuating interaction network and time-varying stability of a natural fish community. *Nature* **554**, 360–363 (2018).

104. Bailey, B. A., Ellner, S. & Nychka, D. W. Chaos with confidence: asymptotics and applications of local Lyapunov exponents. in *Nonlinear dynamics and time series: building a bridge between the natural and statistical sciences* (eds. Cutler, C. & Kaplan, D. T.) (American Mathematical Society, 1997). doi:10.1090/fic/011/08.

105. Shintani, M. & Linton, O. Is There Chaos in The World Economy? A Nonparametric Test Using Consistent Standard Errors. *Int. Econ. Rev.* **44**, 331–357 (2003).

106. Bask, M. & Gençay, R. Testing chaotic dynamics via Lyapunov exponents. *Phys. D Nonlinear Phenom.* **114**, 1–2 (1998).

107. Giannerini, S. & Rosa, R. Generating Replications of Chaotic Time Series. *Nonlinear Dynamics. Psychol. Life Sci.* **5**, 77–87 (2001).

108. Ziehmann, C., Smith, L. A. & Kurths, J. The bootstrap and Lyapunov exponents in deterministic chaos. *Phys. D Nonlinear Phenom.* **126**, 49–59 (1999).

109. Wang, L. & Politis, D. N. Asymptotic validity of bootstrap confidence intervals in nonparametric regression without an additive model. *Electron. J. Stat.* **15**, 392–426 (2021).

110. Abarbanel, H. D. I., Brown, R. & Kennel, M. B. Local Lyapunov exponents computed from observed data. *J. Nonlinear Sci.* **2**, 343–365 (1992).

111. Ye, H., Clark, A., Deyle, E. & Munch, S. rEDM: Applications of Empirical Dynamic Modeling from Time Series. https://ha0ye.github.io/rEDM, https://github.com/ha0ye/rEDM. (2019).

112. Poincaré, H. Introduction. *Acta Math.* **13**, 5–7 (1890).

113. Riley, M. A. & Turvey, M. T. Variability and Determinism in Motor Behavior. *J. Mot. Behav.* **34**, 99–125 (2002).

114. Anderson, N. C., Bischof, W. F., Laidlaw, K. E. W., Risko, E. F. & Kingstone, A. Recurrence quantification analysis of eye movements. *Behav. Res. Methods* **45**, 842–856 (2013).

115. Madeo, D., Castellani, E., Santarcangelo, E. L. & Mocenni, C. Hypnotic assessment based on the Recurrence Quantification Analysis of EEG recorded in the ordinary state of consciousness. *Brain Cogn.* **83**, 227–233 (2013).

116. Karagianni, S. & Kyrtsou, C. Analysing the Dynamics between U.S. Inflation and Dow Jones Index Using Non-Linear Methods. *Stud. Nonlinear Dyn. Econom.* **15**, (2011).

117. Zbilut, J. P. Use of Recurrence Quantification Analysis in Economic Time Series. in *Economics: Complex Windows* (eds. Salzano, M. & Kirman, A.) 91–104 (Springer-Verlag, 2005).

118. Dippner, J. W., Heerkloss, R. & Zbilut, J. P. Recurrence quantification analysis as a tool for characterization of non-linear mesocosm dynamics. *Mar. Ecol. Prog. Ser.* **242**, 29–37 (2002).

119. Proulx, R., Côté, P. & Parrott, L. Multivariate recurrence plots for visualizing and quantifying the dynamics of spatially extended ecosystems. *Ecol. Complex.* **6**, 37–47 (2009).

120. Medvinsky, A. B. *et al.* Chaos far away from the edge of chaos: A recurrence quantification analysis of plankton time series. *Ecol. Complex.* **23**, 61–67 (2015).

121. Marwan, N., Carmenromano, M., Thiel, M. & Kurths, J. Recurrence plots for the analysis of complex systems. *Phys. Rep.* **438**, 237–329 (2007).

122. Trulla, L. L., Giuliani, A., Zbilut, J. P. & Webber, C. L. Recurrence quantification analysis of the logistic equation with transients. *Phys. Lett. A* **223**, 255–260 (1996).

123. Marwan, N. CRP Toolbox 5.22. http://tocsy.pik-potsdam.de/CRPtoolbox. (2020).

124. Garland, J., James, R. & Bradley, E. Model-free quantification of time-series predictability. *Phys. Rev. E* **90**, 52910 (2014).

125. Pennekamp, F. *et al.* The intrinsic predictability of ecological time series and its potential to guide forecasting. *Ecol. Monogr.* **89**, 1–17 (2019).

126. Amigó, J. M., Zambrano, S. & Sanjuán, M. A. F. Combinatorial detection of determinism in noisy time series. *EPL (Europhysics Lett.* **83**, 60005 (2008).

127. Riedl, M., Müller, A. & Wessel, N. Practical considerations of permutation entropy: A tutorial review. *Eur. Phys. J. Spec. Top.* **222**, 249–262 (2013).

128. TOCSY - Toolbox for Complex Systems (Recurrence Plots, Cross Recurrence Plots,

System Identification, ACE, Nonlinear Wavelet Analysis, Nonlinear Regression Analysis, Adaptive Filtering, Coupling Direction). https://tocsy.pik-potsdam.de/ (2020).

129. Lacasa, L., Luque, B., Ballesteros, F., Luque, J. & Nuño, J. C. From time series to complex networks: The visibility graph. *Proc. Natl. Acad. Sci.* **105**, 4972–4975 (2008).

130. Zhu, G., Li, Y., Wen, P. & Wang, S. Analysis of alcoholic EEG signals based on horizontal visibility graph entropy. *Brain Informatics* **1**, 19–25 (2014).

131. Suyal, V., Prasad, A. & Singh, H. P. Visibility-Graph Analysis of the Solar Wind Velocity. *Sol. Phys.* **289**, 379–389 (2014).

132. Yu, Z. G., Anh, V., Eastes, R. & Wang, D.-L. Multifractal analysis of solar flare indices and their horizontal visibility graphs. *Nonlinear Process. Geophys.* **19**, 657–665 (2012).

133. Mali, P., Mukhopadhyay, A., Manna, S. K., Haldar, P. K. & Singh, G. Multifractal analysis of charged particle distributions using horizontal visibility graph and sandbox algorithm. *Mod. Phys. Lett. A* **32**, 1750024 (2017).

134. Rong, L. & Shang, P. Topological entropy and geometric entropy and their application to the horizontal visibility graph for financial time series. *Nonlinear Dyn.* **92**, 41–58 (2018).

135. Lacasa, L. & Toral, R. Description of stochastic and chaotic series using visibility graphs. *Phys. Rev. E* **82**, 36120 (2010).

136. Iacobello, G. Fast Horizontal Visibility Graph (HVG) for MATLAB (https://www.mathworks.com/matlabcentral/fileexchange/72889-fast-horizontal-visibility-graph-hvg-for-matlab), MATLAB Central File Exchange. (2020).

137. Theiler, J., Eubank, S., Longtin, A., Galdrikian, B. & Doyne Farmer, J. Testing for nonlinearity in time series: the method of surrogate data. *Phys. D Nonlinear Phenom.* **58**, 77–94 (1992).

138. Jamšek, J., Paluš, M. & Stefanovska, A. Detecting couplings between interacting oscillators with time-varying basic frequencies: Instantaneous wavelet bispectrum and information theoretic approach. *Phys. Rev. E* **81**, 36207 (2010).

139. Schreiber, T. Extremely simple nonlinear noise-reduction method. *Phys. Rev. E* **47**, 2401–2404 (1993).

140. Gottwald, G. A. & Melbourne, I. A new test for chaos in deterministic systems. *Proc. R. Soc. London. Ser. A Math. Phys. Eng. Sci.* **460**, 603–611 (2004).

141. Gottwald, G. A. & Melbourne, I. Testing for chaos in deterministic systems with noise. *Phys. D Nonlinear Phenom.* **212**, 100–110 (2005).

142. Toker, D. The chaos decision tree algorithm. https://doi.org/10.6084/m9.figshare.7476362.v7. (2019).

143. Ahrestani, F. S., Hebblewhite, M. & Post, E. The importance of observation versus process error in analyses of global ungulate populations. *Sci. Rep.* **3**, 3125 (2013).

144. Francis, C. R. I. C., Hurst, R. J. & Renwick, J. A. Quantifying annual variation in catchability for commercial and research fishing. *Fish. Bull.* **101**, 293–304 (2003).

145. Kamarainen, A. M., Rowland, F. E., Biggs, R. & Carpenter, S. R. Zooplankton and the

total phosphorus – chlorophyll a relationship: hierarchical Bayesian analysis of measurement error. *Can. J. Fish. Aquat. Sci.* **65**, 2644–2655 (2008).

146. Hénon, M. A two-dimensional mapping with a strange attractor. *Commun. Math. Phys.* **50**, 69–77 (1976).

147. Zhang, H. *et al.* Complex Dynamics on the Routes to Chaos in a Discrete Predator-Prey System with Crowley-Martin Type Functional Response. *Discret. Dyn. Nat. Soc.* **2018**, 1–18 (2018).

148. Yu, H., Zhao, M., Lv, S. & Zhu, L. Dynamic complexities in a parasitoid-host-parasitoid ecological model. *Chaos, Solitons & Fractals* **39**, 39–48 (2009).

149. Venkatesan, A. & Lakshmanan, M. Interruption of torus doubling bifurcation and genesis of strange nonchaotic attractors in a quasiperiodically forced map: Mechanisms and their characterizations. *Phys. Rev. E* **63**, 26219 (2001).

150. Ikeda, K. Multiple-valued stationary state and its instability of the transmitted light by a ring cavity system. *Opt. Commun.* **30**, 257–261 (1979).

151. Galias, Z. Rigorous investigation of the Ikeda map by means of interval arithmetic. *Nonlinearity* **15**, 1759–1779 (2002).

152. Guevara, M. R. & Glass, L. Phase locking, period doubling bifurcations and chaos in a mathematical model of a periodically driven oscillator: A theory for the entrainment of biological oscillators and the generation of cardiac dysrhythmias. *J. Math. Biol.* **14**, 1–23 (1982).

153. Timmer, J. Power of surrogate data testing with respect to nonstationarity. *Phys. Rev. E* **58**, 5153–5156 (1998).

154. Zhivomirov, H. A method for colored noise generation. *Rom. J. Acoust. Vib.* **15**, 14–19 (2018).

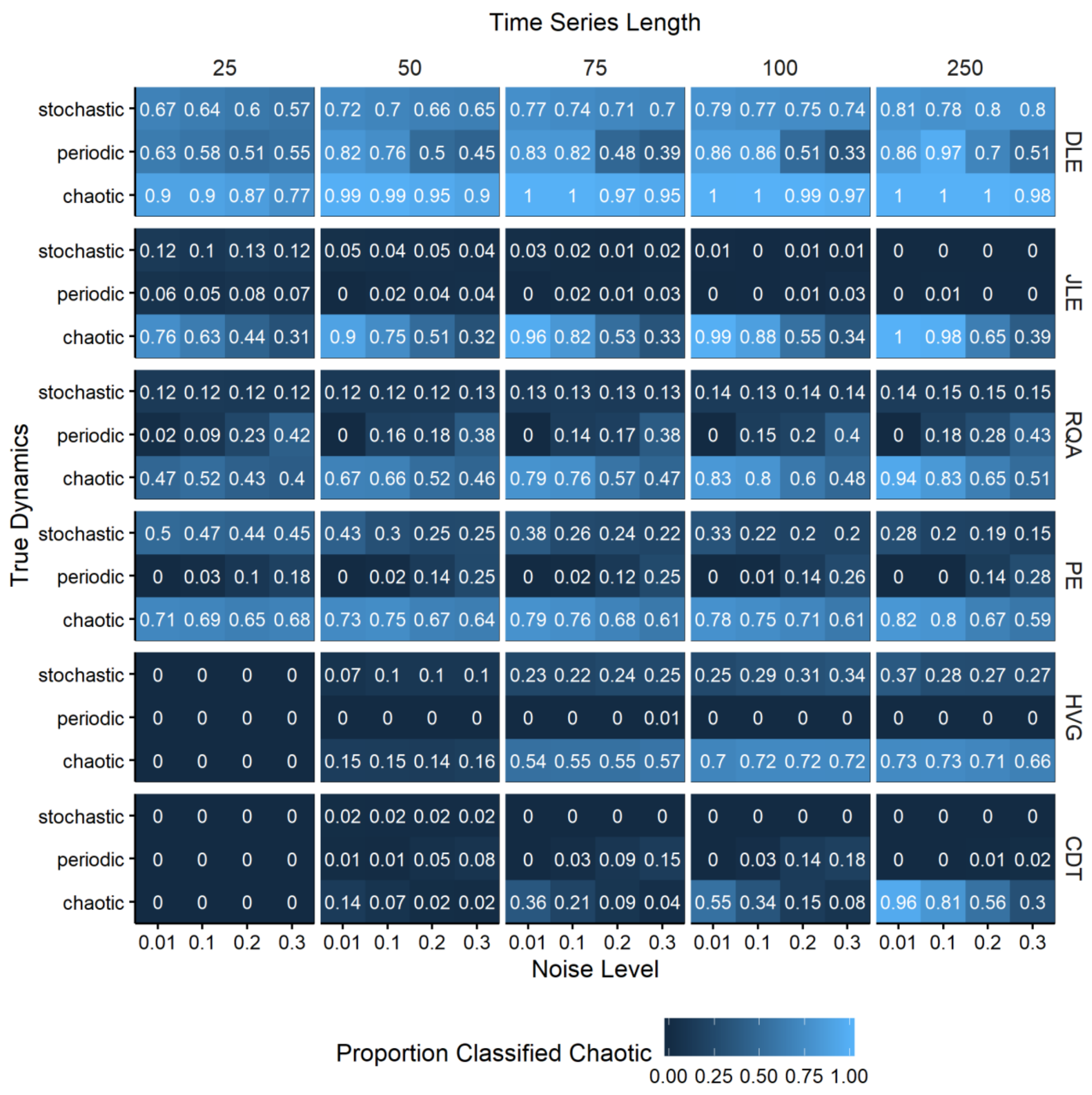
155. Ali, I., Saeed, U. & Din, Q. Bifurcation analysis and chaos control in discrete-time system of three competing species. *Arab. J. Math.* **8**, 1–14 (2019).

156. Hilborn, R. C. *Chaos and nonlinear dynamics: an introduction for scientists and engineers*. (Oxford University Press, 2000).

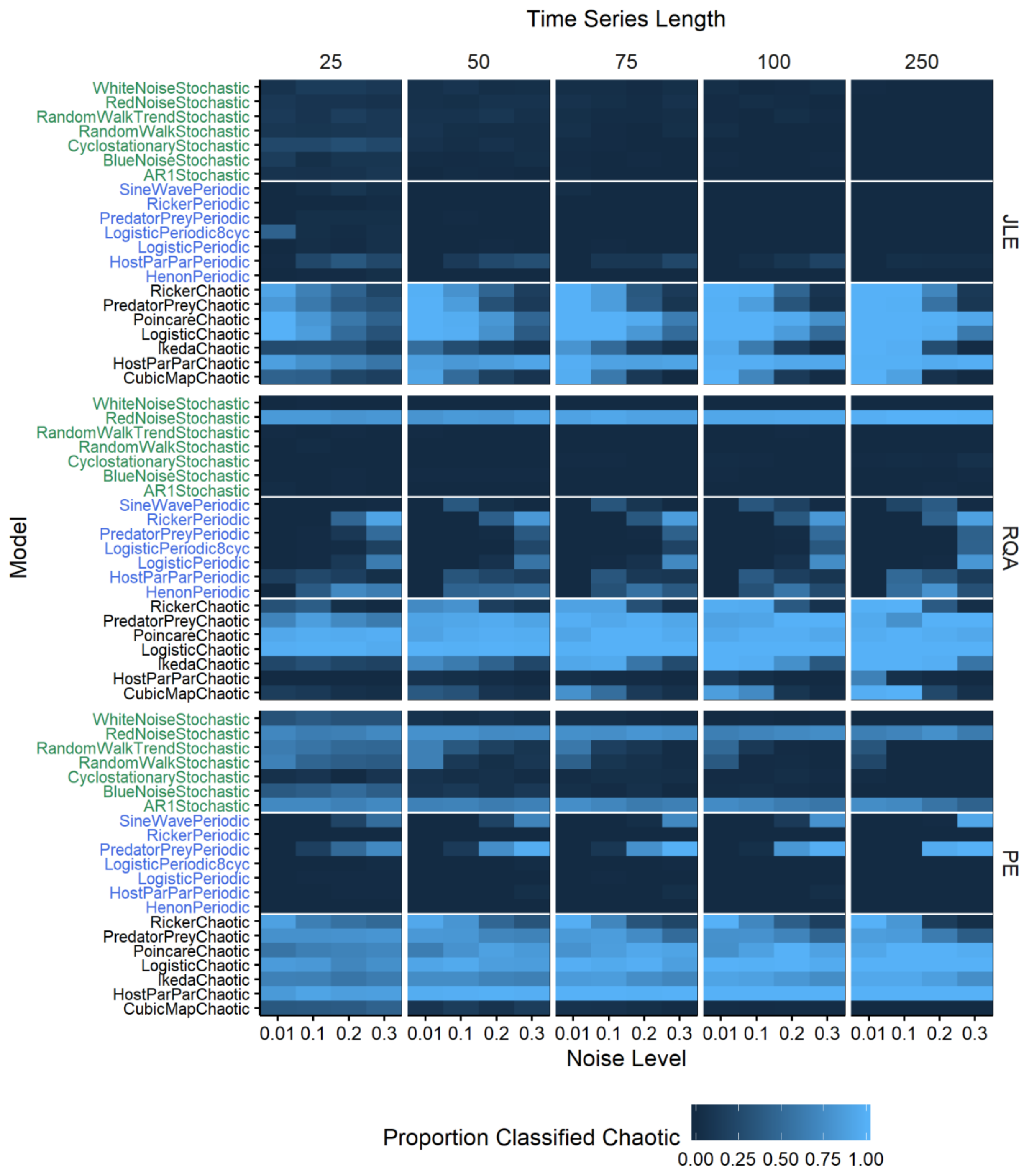
157. Yuan, S., Jiang, T. & Jing, Z. Bifurcation and chaos in the tinkerbell map. *Int. J. Bifurc. Chaos* **21**, 3137–3156 (2011).

158. Turchin, P. & Hanski, I. An Empirically Based Model for Latitudinal Gradient in Vole Population Dynamics. *Am. Nat.* **149**, 842–874 (1997).

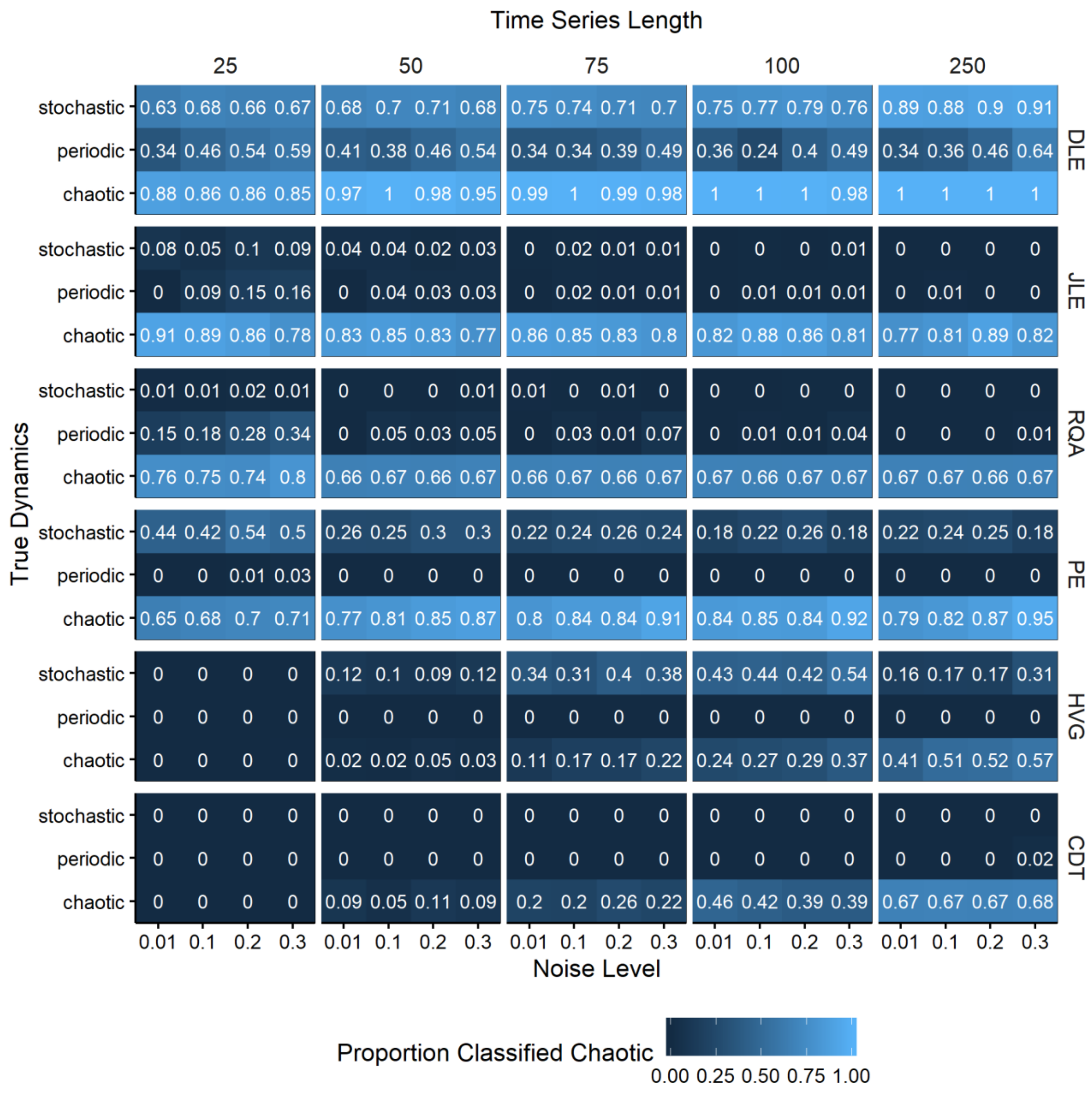
159. Edwards, C. A., Powell, T. A. & Batchelder, H. P. The stability of an NPZ model subject to realistic levels of vertical mixing. *J. Mar. Res.* **58**, 37–60 (2000).



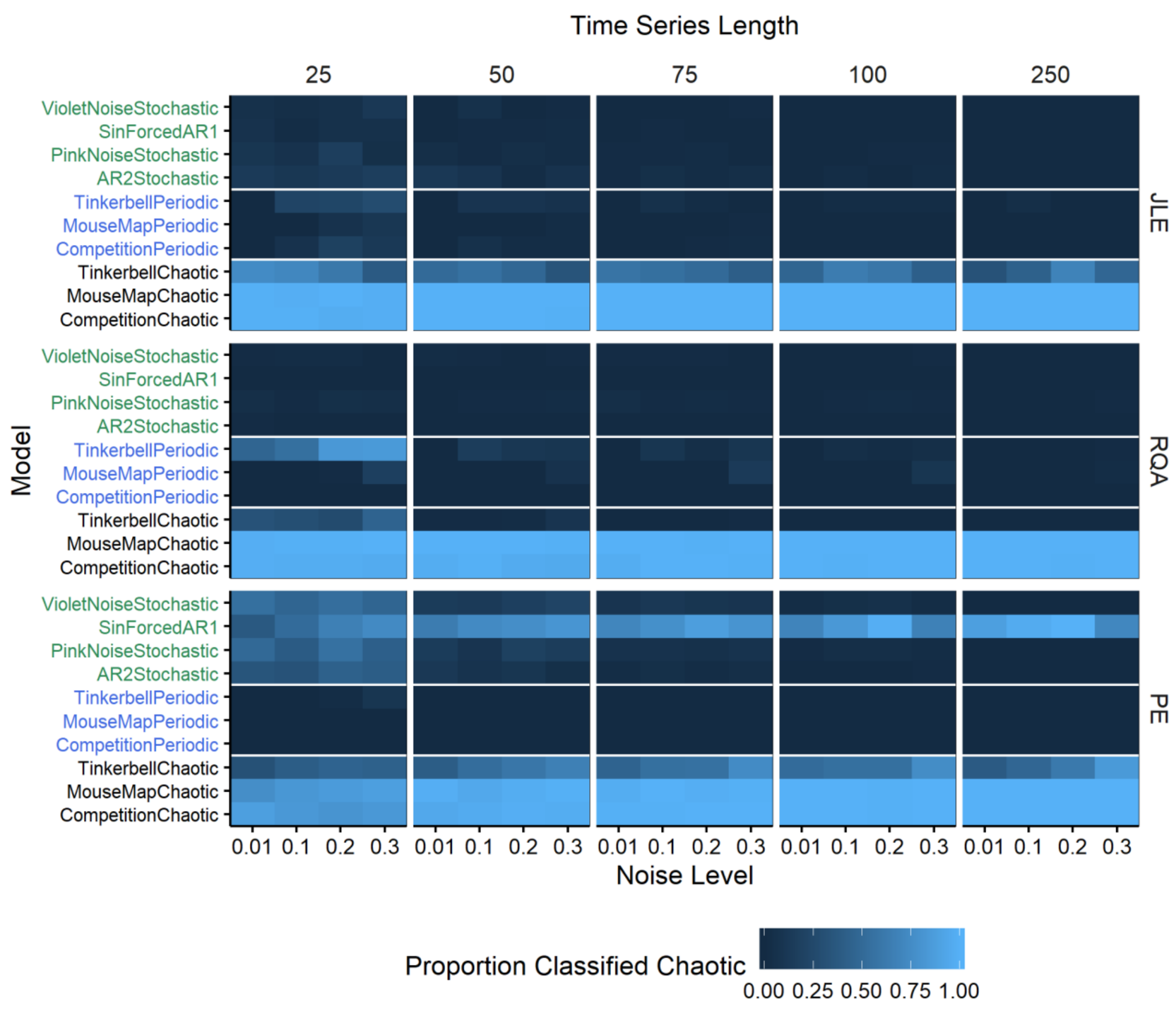
Supplementary Fig. 1. Proportion of simulated time series from the *test* dataset classified as chaotic for all replicates of all models within each dynamical regime, for different levels of observation noise and time series length, for each chaos detection method. DLE = direct Lyapunov exponent, JLE = Jacobian-based Lyapunov exponent, RQA = recurrence quantification analysis, PE = permutation entropy, HVG = horizontal visibility graphs, CDT = chaos decision tree.



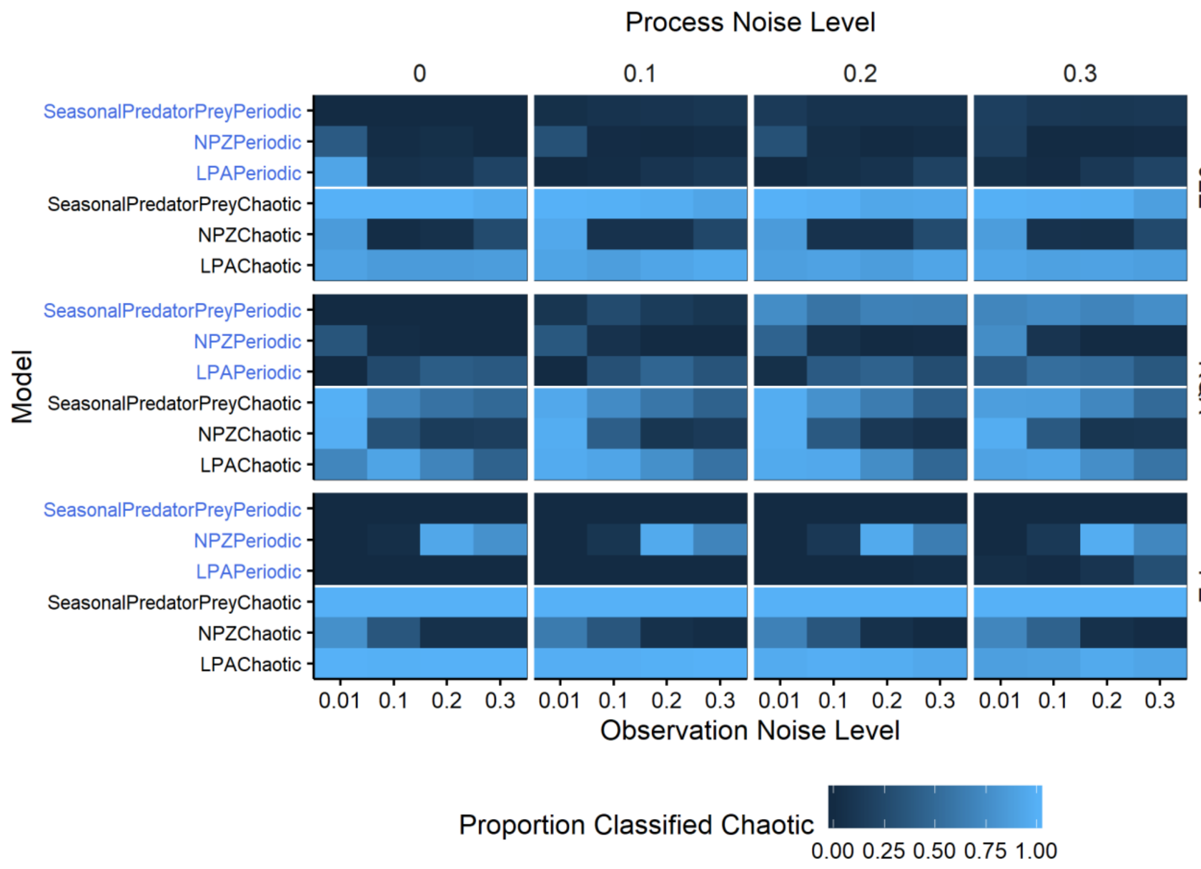
Supplementary Fig. 2. Proportion of simulated time series from the *test* dataset classified as chaotic for all replicates of each individual model, for different levels of observation noise and time series length. Results for the 3 most reliable chaos detection methods are shown. JLE = Jacobian-based Lyapunov exponent, RQA = recurrence quantification analysis, PE = permutation entropy.



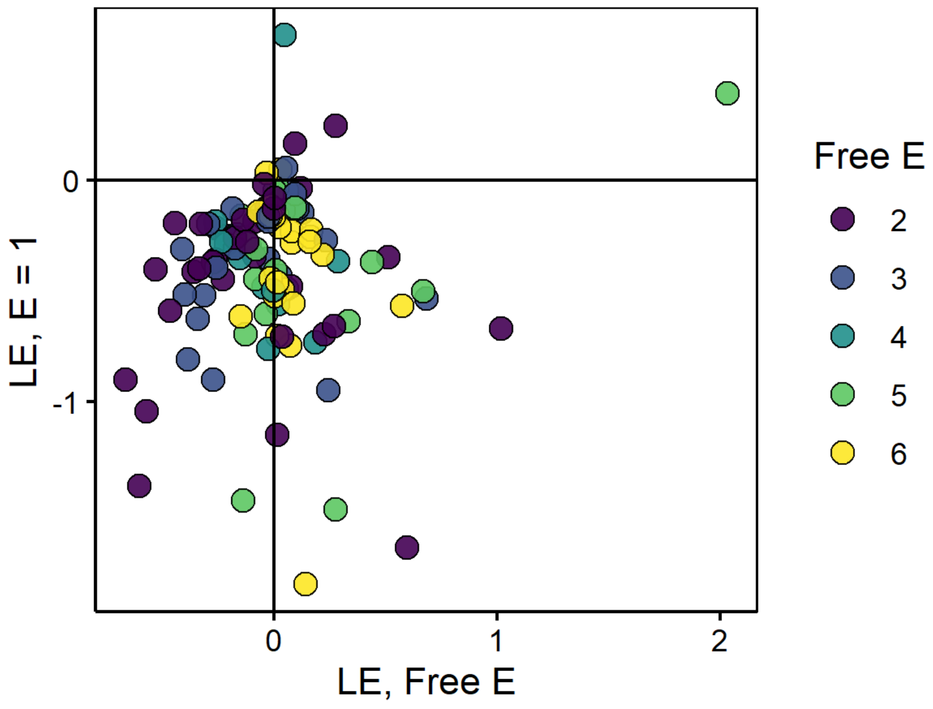
Supplementary Fig. 3. Proportion of simulated time series *validation* dataset #1 classified as chaotic for all replicates of all models within each dynamical regime, for different levels of observation noise and time series length, for each chaos detection method. DLE = direct Lyapunov exponent, JLE = Jacobian-based Lyapunov exponent, RQA = recurrence quantification analysis, PE = permutation entropy, HVG = horizontal visibility graphs, CDT = chaos decision tree.



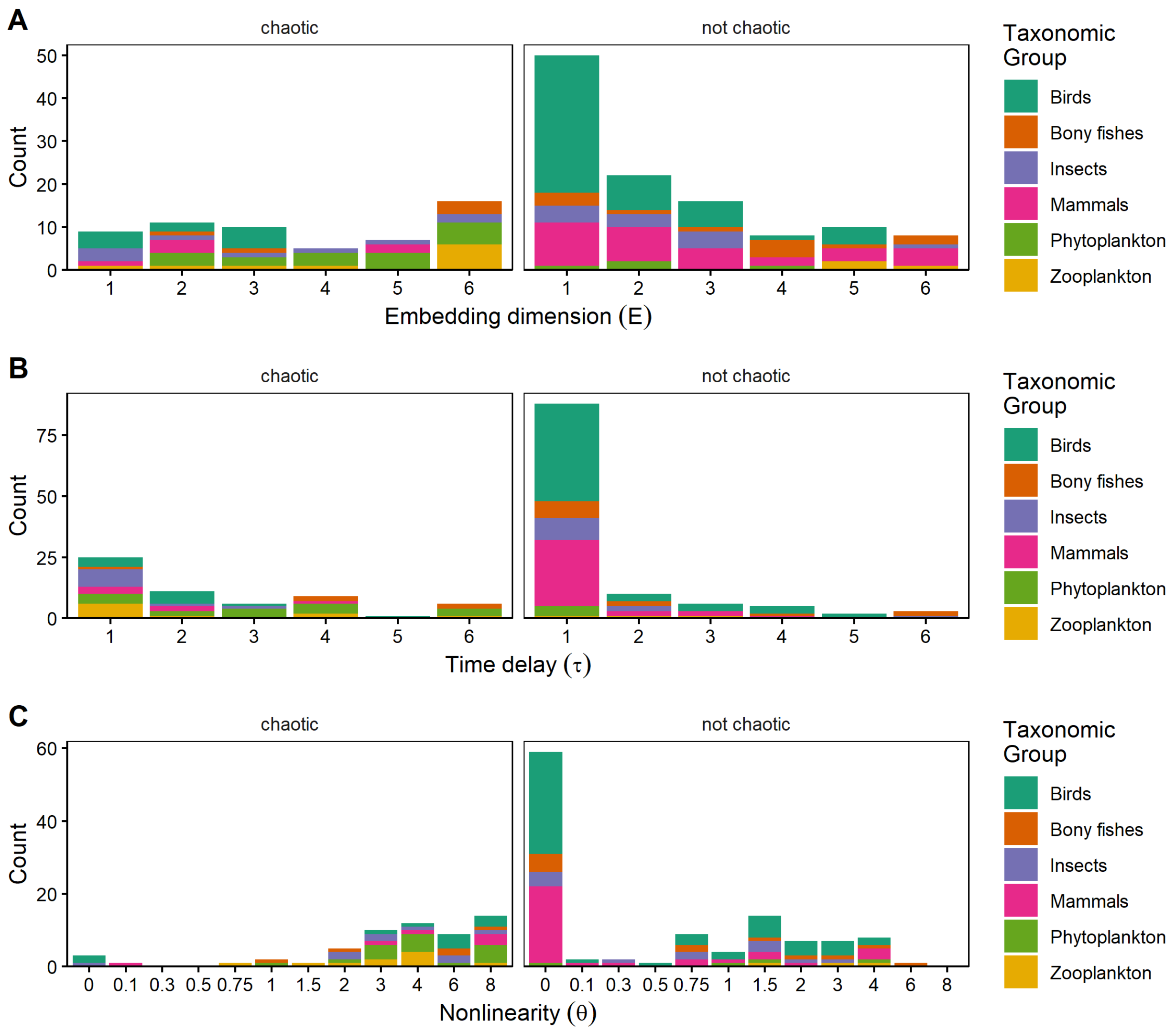
Supplementary Fig. 4. Proportion of simulated time series from *validation* dataset #1 classified as chaotic for all replicates of each individual model, for different levels of observation noise and time series length. Results for the 3 most reliable methods are shown. JLE = Jacobian-based Lyapunov exponent, RQA = recurrence quantification analysis, PE = permutation entropy.



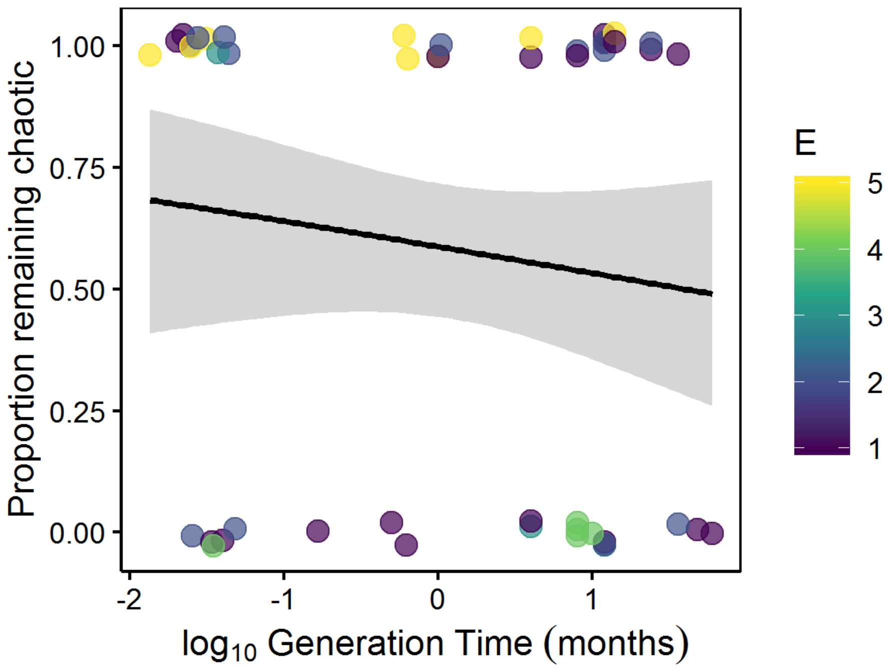
Supplementary Fig. 5. Proportion of simulated times series from *validation* dataset #2 classified as chaotic for all replicates of each individual model, for different levels of observation noise and process noise. Results for the 3 most reliable methods are shown. JLE = Jacobian-based Lyapunov exponent, RQA = recurrence quantification analysis, PE = permutation entropy.



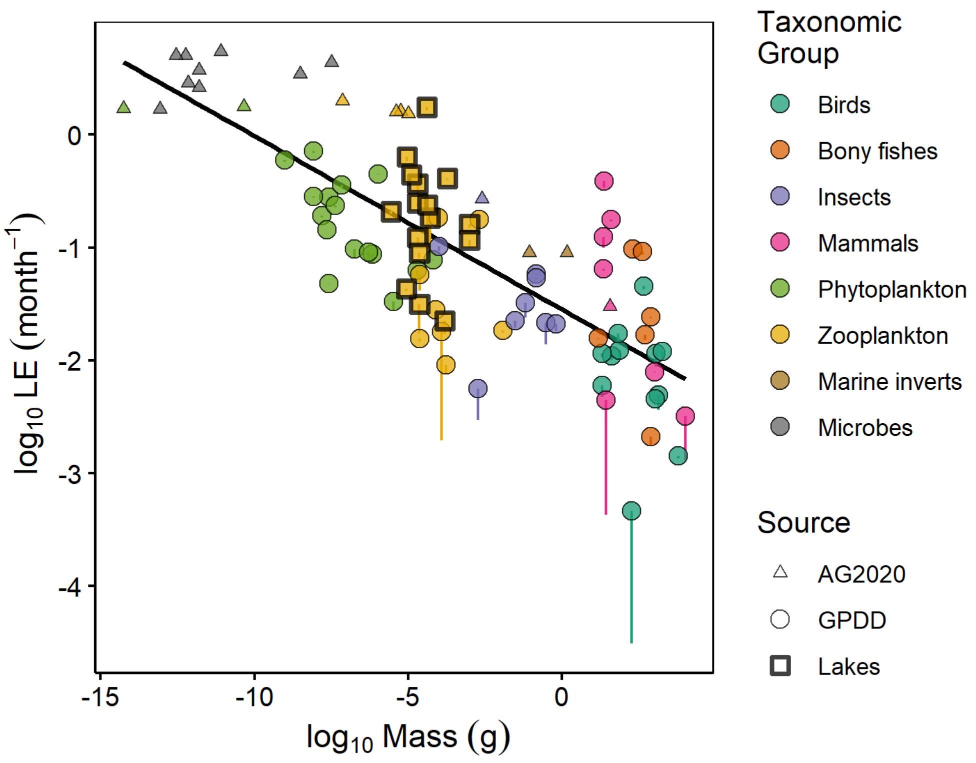
Supplementary Fig. 6. Lyapunov exponent (LE, timestep^{-1}) with unconstrained embedding dimension (Free E) and with embedding dimension fixed to 1 (E=1), excluding time series with Free E = 1, which would fall along 1:1 diagonal.



Supplementary Fig. 7. Distribution of (A) embedding dimension *E*, (B) time delay ***τ***, and (C) local weighting parameter (θ) values by taxonomic group and chaos classification using the Jacobian method.



Supplementary Fig. 8. Proportion of chaotic series that remained classified as chaotic using the Jacobian method when time series were truncated to the last 30 time points, in relation to generation time. The line is a logistic regression, associated band is the 95% confidence interval, and points are vertically jittered to reduce overlap.



Supplementary Fig. 9. Positive Lyapunov exponents (LEs) in relation to body mass, color distinguishing broad taxonomic groups. Includes data from this study (GPDD and supplemental results from 3 lake systems) and positive LEs compiled by [47] (AG2020). The log-log scale is in keeping with prior studies[47]. Note that the lake data (squares) were not used to fit the regression line. Vertical bars are lower confidence intervals.

Supplementary Table 1. False negative rates (FNR) and false positive rates (FPR) for 6 chaos detection methods across simulated datasets. Values in italics indicate misclassification rates >0.5.

Chaos detection method	**Test**		**Validation #1**		**Validation #2**	
	FNR	**FPR**	**FNR**	**FPR**	**FNR**	**FPR**
Direct LE	0.04	*0.68*	0.04	*0.61*	0.23	*0.68*
Jacobian LE	0.35	0.03	0.16	0.03	0.28	0.10
Recurrence quantification analysis	0.38	0.16	0.32	0.03	0.38	0.28
Permutation entropy	0.30	0.20	0.18	0.16	0.25	0.15
Horizontal visibility algorithm	*0.57*	0.09	*0.80*	0.13	*0.54*	0.05
Chaos decision tree	*0.77*	0.02	*0.72*	0.001	*0.65*	0.09

Supplementary Table 2. False negative and false positive rates for several JLE confidence interval estimation methods, which classify time series as chaotic if p(LE>0)>0.95. Results are based on the first 20 replicates of the test dataset and are pooled across all models, time series length, and noise levels.

Method	**FNR**	**FPR**	**FNR + FPR**
1. Point estimate (no confidence interval)	0.28	0.16	0.44
2. Residual bootstrap	0.50	0.07	0.57
3. Asymptotic SE	0.43	0.02	0.45
4. Moving window	0.35	0.03	0.38
5. Importance sampling	0.34	0.10	0.44

Supplementary Table 3. Models used to generate dynamics with a known embedding dimension *E* and time delay $\boldsymbol{\tau}$. Models were generated with random initial conditions and the first 500 points were removed to avoid transients.

Model	**Parameters**	**Known *E***	**Known τ**
$x_t = x_{t-1}e^{r-x_{t-1}}$	$r = 3$	1	1
$x_t = x_{t-2}e^{r-x_{t-2}}$	$r = 3$	1	2
$x_t = x_{t-3}e^{r-x_{t-3}}$	$r = 3$	1	3
$x_t = x_{t-1}e^{r-x_{t-1}-x_{t-2}}$	$r = 3.25$	2	1
$x_t = x_{t-2}e^{r-x_{t-2}-x_{t-4}}$	$r = 3.25$	2	2
$x_t = x_{t-3}e^{r-x_{t-3}-x_{t-6}}$	$r = 3.25$	2	3
$x_t = x_{t-1}e^{r-x_{t-1}-x_{t-2}-x_{t-3}}$	$r = 3.25$	3	1
$x_t = x_{t-2}e^{r-x_{t-2}-x_{t-4}-x_{t-6}}$	$r = 3.25$	3	2
$x_t = x_{t-3}e^{r-x_{t-3}-x_{t-6}-x_{t-9}}$	$r = 3.25$	3	3

Supplementary Table 4. Model formulations used for s-map regression.

Name	**Model**
First difference, abundance	$x_t = x_{t-\tau} + f(x_{t-\tau}, x_{t-2\tau}, \dots, x_{t-E\tau})$ $x_t - x_{t-\tau} = f(x_{t-\tau}, x_{t-2\tau}, \dots, x_{t-E\tau})$
Growth rate, abundance	$x_t = x_{t-\tau} e^{f(x_{t-\tau}, x_{t-2\tau}, \dots, x_{t-E\tau})}$ $\ln\left(\frac{x_t}{x_{t-\tau}}\right) = f(x_{t-\tau}, x_{t-2\tau}, \dots, x_{t-E\tau})$
Growth rate, log abundance	$x_t = x_{t-\tau} e^{f(\ln x_{t-\tau}, \ln x_{t-2\tau}, \dots, \ln x_{t-E\tau})}$ $\ln\left(\frac{x_t}{x_{t-\tau}}\right) = f(\ln x_{t-\tau}, \ln x_{t-2\tau}, \dots, \ln x_{t-E\tau})$

Supplementary Table 5. Leave-one-out prediction R^2 values from the local linear model (JLE) and rates of positive chaos detection in the empirical GPDD dataset using the best model E and $\boldsymbol{\tau}$, and with E or $\boldsymbol{\tau}$ fixed to be either 1 greater or less than the value in the best model. Rate of chaos detection are shown for JLE and RQA.

Model	**mean R^2 for abundance (JLE)**	**median R^2 for abundance (JLE)**	**Prop. chaotic (JLE)**	**Prop. chaotic (RQA)**
Best	0.40	0.35	0.34	0.41
$E+1$	0.31	0.26	0.29	0.36
$E-1$	0.31	0.24	0.23	0.48
$\tau+1$	0.15	0.05	0.26	0.31
$\tau-1$	0.31	0.28	0.26	0.43

Supplementary Table 7. Models used to generate periodic dynamics (test dataset).

Name	Model	Parameters	Reference
Logistic Map (8-cycle)	$x_{t+1} = rx_t(1 - x_t)$	$r = 3.55$	35,56
Logistic Map (3-cycle)	$x_{t+1} = rx_t(1 - x_t)$	$r = 3.828427$	35,56
Ricker Map (2-cycle)	$x_{t+1} = x_t e^{r(1-x_t)}$	$r = 2.2$	85
Henon Map (4-cycle)	$x_{t+1} = 1 - ax_t^2 + bx_{t-1}$	$a = 0.95$ $b = 0.3$	35,146
Sine Wave (12-cycle)	$x_t = a\cos\left(\frac{2\pi}{b}t\right) + a$	$a = 1$ $b = 12$	
Predator-Prey (5-cycle)	$x_{t+1} = x_t + \tau x_t\left(a - x_t - \frac{by_t}{(1+\alpha x_t)(1+\beta y_t)}\right)$ $y_{t+1} = y_t + \tau y_t\left(-c + \frac{dx_t}{(1+\alpha x_t)(1+\beta y_t)}\right)$	$a = 2, b = 2$ $c = 2, d = 1.85$ $\alpha = 0.1, \beta = 0.1$ $\tau = 1.1$	147
Host-Parasitoid-Parasitoid (6-cycle)	$x_{t+1} = x_t e^{r\left(1-\frac{x_t}{K}\right) - ay_t^{-m+1} - bz_t^{-n+1}}$ $y_{t+1} = x_t\left(1 - e^{-ay_t^{-m+1} - bz_t^{-n+1}}\right)\frac{ay_t^{-m+1}}{ay_t^{-m+1} + bz_t^{-n+1}}$ $z_{t+1} = x_t\left(1 - e^{-ay_t^{-m+1} - bz_t^{-n+1}}\right)\frac{bz_t^{-n+1}}{ay_t^{-m+1} + bz_t^{-n+1}}$	$r = 2.5, K = 20$ $a = 0.9, b = 1.12$ $m = 0.7, n = 0.4$	148

Supplementary Table 7. Models used to generate chaotic dynamics (test dataset).

Name	**Model**	**Parameters**	**Reference**
Logistic Map	$x_{t+1} = rx_t(1 - x_t)$	$r = 3.9$	[35,56]
Ricker Map	$x_{t+1} = x_t e^{r(1-x_t)}$	$r = 3.4$	[85]
Cubic Map	$x_{t+1} = f\cos(2\pi\theta_t) - Ax_t + x^3$ $\theta_{t+1} = \theta_t + \omega (\text{mod } 1)$	$f = -0.8$ $A = 1.5$ $\omega = \frac{\sqrt{5}-1}{2}$	[35,149]
Ikeda Map	$x_{t+1} = 1 + u(x_t \cos\theta_t - y_t \sin\theta_t)$ $y_{t+1} = u(x_t \cos\theta_t - y_t \sin\theta_t)$ $\theta_t = 0.4 - \frac{6}{1 + x_t^2 + y_t^2}$	$u = 0.9$	[35,150,151]
Poincare Oscillator	$x_{t+1} = \frac{1}{2\pi}\cos^{-1}\frac{\cos(2\pi x_t)+b}{\sqrt{1+b^2+2b\cos(2\pi x_t)}}(\text{mod } 1)$	$b = 1.13$ $\tau = 0.65$	[152]
Predator-Prey	$x_{t+1} = x_t + \tau x_t\left(a - x_t - \frac{by_t}{(1+\alpha x_t)(1+\beta y_t)}\right)$ $y_{t+1} = y_t + \tau y_t\left(-c + \frac{dx_t}{(1+\alpha x_t)(1+\beta y_t)}\right)$	$a = 2, b = 2$ $c = 2, d = 1.85$ $\alpha = 0.1, \beta = 0.1$ $\tau = 1.27$	[147]
Host-Parasitoid-Parasitoid	$x_{t+1} = x_t e^{r\left(1-\frac{x_t}{K}\right) - ay_t^{-m+1} - bz_t^{-n+1}}$ $y_{t+1} = x_t\left(1 - e^{-ay_t^{-m+1} - bz_t^{-n+1}}\right)\frac{ay_t^{-m+1}}{ay_t^{-m+1} + bz_t^{-n+1}}$ $z_{t+1} = x_t\left(1 - e^{-ay_t^{-m+1} - bz_t^{-n+1}}\right)\frac{bz_t^{-n+1}}{ay_t^{-m+1} + bz_t^{-n+1}}$	$r = 3.4, K = 20$ $a = 0.9, b = 1.12$ $m = 0.7, n = 0.4$	[148]

Supplementary Table 8. Models used to generate stochastic dynamics (test dataset).

Name	Model	Parameters	Reference
AR(1)	$x_{t+1} = c + \phi x_t + \epsilon_t$	$c = 8$ $\phi = 0.8$ $\epsilon \sim N(0,1)$	
Cyclostationary	$x_{t+1} = a_1 x_t + a_2 x_{t-1} + \epsilon_t$	$a_1 = 2\cos\left(\frac{2\pi}{10}\right) e^{-1/50}$ $a_2 = -e^{-1/25}$ $\epsilon \sim N(0,1)$	[35,153]
Random Walk	$x_{t+1} = x_t + \epsilon_t$	$\epsilon \sim N(0,1)$	[35]
Random Walk with Trend	$x_{t+1} = x_t + b + \epsilon_t$	$b = 0.1$ $\epsilon \sim N(0,1)$	[35]
White Noise	$x_t = \epsilon_t$	$\epsilon \sim N(0,1)$	
Red Noise	PSD proportional to $\frac{1}{f^2}$		[154]
Blue Noise	PSD proportional to f		[154]

Supplementary Table 9. Models used to generate periodic dynamics (validation dataset #1).

Name	**Model**	**Parameters**	**Reference**
Three-species competition (4-cycle)	$x_{t+1} = x_t e^{r(1-x_t-ay_t-bz_t)}$ $y_{t+1} = y_t e^{r(1-y_t-az_t-bx_t)}$ $z_{t+1} = z_t e^{r(1-z_t-ax_t-by_t)}$	$r = 2.6$ $a = 0.65$ $b = 0.6$	[155]
Mouse map (2-cycle)	$x_{t+1} = e^{-\alpha x_t{}^2} + \beta$	$\alpha = 6.2$ $\beta = 0$	[156]
Tinkerbell Map (9-cycle)	$x_{t+1} = x_t{}^2 - y_t{}^2 + ax_t + by_t$ $y_t = 2x_t y_t + cx_t + dy_t$	$a = 0.9, b = -0.5$ $c = 1.8, d = 0.5$	[157]

Supplementary Table 10. Models used to generate chaotic dynamics (validation dataset #1).

Name	Model	Parameters	Reference
Three-species competition	$x_{t+1} = x_t e^{r(1-x_t-ay_t-bz_t)}$ $y_{t+1} = y_t e^{r(1-y_t-az_t-bx_t)}$ $z_{t+1} = z_t e^{r(1-z_t-ax_t-by_t)}$	$r = 3$ $a = 0.65$ $b = 0.6$	[155]
Mouse map	$x_{t+1} = e^{-\alpha x_t{}^2} + \beta$	$\alpha = 6.2$ $\beta = -0.5$	[156]
Tinkerbell Map	$x_{t+1} = x_t{}^2 - y_t{}^2 + ax_t + by_t$ $y_t = 2x_t y_t + cx_t + dy_t$	$a = 0.9$, $b = -0.5$ $c = 2.15$, $d = 0.5$	[157]

Supplementary Table 11. Models used to generate stochastic dynamics (validation dataset #1).

Name	**Model**	**Parameters**	**Reference**
AR(2)	$x_{t+1} = \phi_1 x_t + \phi_2 x_{t-1} + \epsilon_t$	$\phi_1 = 0.9,$ $\phi_2 = -0.1,$ $\epsilon \sim N(0,1)$	
Seasonally-forced AR(1)	$x_{t+1} = b\left(x_t - A\sin\left(\frac{2\pi}{12}t\right)\right) + \epsilon_t$	$A = 2, b = 0.8$ $\epsilon \sim N(0,1)$	
Pink Noise	PSD proportional to $\frac{1}{f}$		[154]
Violet Noise	PSD proportional to f^2		[154]

Supplementary Table 12. Models used to generate periodic dynamics with process noise (validation dataset #2).

Name	**Model**	**Parameters**	**Reference**
Seasonal Predator Prey	$\dot{x} = r(1 - A\ \sin 2\pi t))x - rx^2 - \frac{gx^2}{x^2 + h^2} - \frac{axy}{x + d}$ $\dot{y} = s(1 - A\ \sin 2\pi t))y - \frac{sy^2}{x}$	$A = 2.2$ $s = 1.5$ $g\ = 2$ $h\ = 0.13$ $a\ = 7.5$ $d\ = 0.06$ $r \sim N(7, \sigma)$	[158]
Larvae-Pupae-Adult (LPA)	$L_{t+1} = bA_t e^{-c_{el}L_t - c_{ea}A_t} e^{\xi_{1,t}}$ $P_{t+1} = (1 - \mu_l)L_t e^{\xi_{2,t}}$ $\mathrm{A}_{t+1} = [P_t e^{-c_{pa}A_t} + A_t(1 - \mu_a)]e^{\xi_{3,t}}$	$b = 6.598$ $c_{el} = 0.01209$ $c_{ea} = 0.01155$ $c_{pa} = 1$ $\mu_l = 0.2055$ $\mu_a = 0.96$ $\xi_{i,t} \sim N(-\frac{\sigma^2}{2}, \sigma)$	[10]
Nutrient-Phytoplankton-Zooplankton (NPZ)	$\dot{N} = -\frac{v_m\left(1 - A\sin\left(\frac{2\pi}{365}t\right)\right)NP}{k_s + N}e^{kh} + \gamma R_m Z(1 - e^{-\Lambda \mathrm{P}}) + mP + gZ$ $\dot{P} = \frac{v_m\left(1 - A\sin\left(\frac{2\pi}{365}t\right)\right)NP}{k_s + N}e^{kh} - R_m Z(1 - e^{-\Lambda \mathrm{P}}) - mP$ $\dot{Z} = (1 - \gamma)R_m Z(1 - e^{-\Lambda \mathrm{P}}) - gZ$	$v_m = 2$ $k_s = 0.1$ $k\ = 0.06$ $\Lambda\ = 0.2$ $\gamma\ = 0.3$ $m\ = 0.1$ $g\ = 0.2$ $A\ = 0$ $h\ = -35$ $R_m \sim N(0.5, \sigma)$	[159]

Supplementary Table 13. Models used to generate chaotic dynamics with process noise (validation dataset #2).

Name	**Model**	**Parameters**	**Reference**
Seasonal Predator Prey	$\dot{x} = r(1 - A \sin 2\pi t))x - rx^2 - \frac{gx^2}{x^2 + h^2} - \frac{axy}{x + d}$ $\dot{y} = s(1 - A \sin 2\pi t))y - \frac{sy^2}{x}$	$A = 1$ $s = 1.25$ $g = 0$ $h = 0.08$ $a = 710$ $d = 0.04$ $r \sim N(6, \sigma)$	158
Larvae-Pupae-Adult (LPA)	$L_{t+1} = bA_t e^{-c_{el}L_t - c_{ea}A_t} e^{\xi_{1,t}}$ $P_{t+1} = (1 - \mu_l)L_t e^{\xi_{2,t}}$ $\mathrm{A}_{t+1} = [P_t e^{-c_{pa}A_t} + A_t(1 - \mu_a)]e^{\xi_{3,t}}$	$b = 6.598$ $c_{el} = 0.01209$ $c_{ea} = 0.01155$ $c_{pa} = 0.35$ $\mu_l = 0.2055$ $\mu_a = 0.96$ $\xi_{i,t} \sim N(-\frac{\sigma^2}{2}, \sigma)$	10
Nutrient-Phytoplankton-Zooplankton (NPZ)	$\dot{N} = -\frac{v_m\left(1 - A \sin\left(\frac{2\pi}{365}t\right)\right)NP}{k_s + N}e^{kh} + \gamma R_m Z(1 - e^{-\Lambda \mathrm{P}}) + mP + gZ$ $\dot{P} = \frac{v_m\left(1 - A \sin\left(\frac{2\pi}{365}t\right)\right)NP}{k_s + N}e^{kh} - R_m Z(1 - e^{-\Lambda \mathrm{P}}) - mP$ $\dot{Z} = (1 - \gamma)R_m Z(1 - e^{-\Lambda \mathrm{P}}) - gZ$	$v_m = 2$ $k_s = 0.1$ $k = 0.06$ $\Lambda = 0.3$ $\gamma = 0.7$ $m = 0.1$ $g = 0.2$ $A = 1$ $h = 0$ $R_m \sim N(4, \sigma)$	159