Generalized Relational Tensors For Irregularly Sampled Time Series: Methods To Store, Re-generate, And Predict

Vasilii Gromov (✉ stroller@rambler.ru)

National Research University Higher School of Economics

Anastasia Necheporenko

National Research University Higher School of Economics

Andrei Gaisin

École Polytechnique

Ilya Volkov

National Research University Higher School of Economics

Stanislav Diner

National Research University Higher School of Economics

Research Article

Keywords: Chaotic time series, graph representation of a series, ant colony optimization, irregularly sampled time series

Posted Date: September 17th, 2021

DOI: <https://doi.org/10.21203/rs.3.rs-853110/v1>

License: This work is licensed under a Creative Commons Attribution 4.0 International License.

[Read Full License](#)

Generalized Relational Tensors for Irregularly Sampled Time Series: Methods to Store, Re-generate, and Predict

Vasilii A. Gromov^{*a}, Anastasia I. Necheporenko^a,

Andrei G. Gaisin^b, Ilya D. Volkov^a, and Stanislav S. Diner^a

^a School of Data Analysis and Artificial Intelligence, HSE University, Pokrovskii boulevard, 11, Moscow, Russian Federation

^b Department of Computer Science, Ecole Polytechnique, Palaiseau Cedex, Paris, France

* Corresponding author (vgromov@hse.ru; stroller@rambler.ru)

Vasilii A. Gromov 0000-0001-5891-6597

Stanislav S. Diner 0000-0002-4760-646X

Abstract

The paper deals with a generalized relational tensor, a novel discrete structure to store information about a time series, and algorithms (1) to fill the structure, (2) to generate a time series from the structure, and (3) to predict a time series, for both regularly and irregularly sampled time series. The algorithms combine the concept of generalized z-vectors with ant colony optimization techniques. In order to estimate quality of the storing/re-generating procedure, a difference between characteristics of the initial and regenerated time series is used. The structure allows working with a multivariate time series, with an irregularly sampled time series, and with a number of series as well. For chaotic time series, a difference between characteristics of the initial time series (the highest Lyapunov exponent, the auto-correlation function) and those of the time series re-generated from a structure is used to assess the effectiveness of the algorithms in question. The approach has shown fairly good results for periodic and benchmark chaotic time series and satisfactory results for real-world chaotic data.

Keywords Chaotic time series; graph representation of a series; ant colony optimization; irregularly sampled time series.

Declarations

Funding. The article was prepared within the framework of the HSE University Basic Research Program, HSE University, Moscow, Russian Federation.

Conflicts of interest/Competing interests. The authors declare that there is no conflict of interest regarding the publication of this paper.

Availability of data and material. Data available on request from the authors.

Code availability. Code is available on the author's request.

Authors' contributions. All authors contribute equally to this work.

Ethics approval. Not applicable.

Consent to participate. Not applicable.

Consent for publication. Not applicable.

1. Introduction

Discrete structures used to represent chaotic time series make it possible to study the series with the employment of complex networks theory, thus allowing deeper insight into series dynamics. A novel discrete structure, a generalized relation tensor, is discussed in the paper. The structure can store information about a time series, generate a time series from itself, and forecast a time series as well. In particular, this allows one to estimate the extent in which structure is able to store the information about coded time series, comparing characteristics of the initial and re-generated time series. A generalized relation tensor relies on the concept of a generalized z-vector, a z-vector that comprises non-successive observations of a time series (irregular embedding). The tensor is readily applicable to both regularly and irregularly sampled time series (with missing data). To design fast and efficient algorithms to fill, re-generate, and predict with the tensor we utilize ant colony optimization (distributed artificial intelligence approach).

The idea to represent a time series using some discrete structure (which is of particular value for chaotic time series) has made great progress over the past few decades [7, 24, 26, 37, 39]. We should emphasize that regardless of a structure or method used any employed approach can be considered as a way of representing a potentially infinite time series with the use of a finite number of real numbers. “Time series analysis is essentially data compression” [6, 50].

The gained experience for the application of such discrete structures revealed some demands. Firstly, it is necessary sometimes to restrict the number of vertices and edges of a used discrete structure. Such restriction excludes the approaches where the number of vertices equals to the number of observations. The latter is particularly the case for chaotic systems, where it is supposed that the series is long enough to attain regions of strange attractor with relatively small invariant measure values [20, 33]. This brought into existence a number of methods that rely on the concept of z-vector (regular embedding); please, for example, refer to [40, 43]; a z-vector is a sequence of successive observations of a time series (a segment of a time series). Its length is bounded from below by the minimum embedding dimension required to produce a delay embedding such that topological equivalence is guaranteed (please, refer to Takens’s theorem); in actual practice, one should apply sophisticated methods to estimate this length. Secondly, such algorithm is practicable only if it can proceed irregularly sampled data [2]. This implies that one should modify the concept of irregular embedding [47], that it is one should employ generalized z-vectors, composed of non-successive observations. In that case a sliding window employed to obtain z-vectors is replaced by a set of sliding combs with some teeth broken off (patterns). One should stress here that the Takens’s theorem stipulates the length of z-vector, not the way one embeds a time series into a space of generalized z-vectors that is state (embedding) vectors obtained with irregular embedding [47]. On the other hand, Gromov and Borisenko [16] show that the generalized z-vectors may be more efficient for regularly sampled series likewise. Finally, the third demand is that such structure must be able to generate a time series. This makes possible to compare some salient characteristics of original and re-generated time series in order to estimate, whether the filled structure is consistent with the time series used to fill it [39].

To sum up, to the best of authors’ knowledge, none of the available discrete structures makes it possible to re-generate a series using the filled structure only, without resort to observations of the initial time series. More to the point, they are rarely able to predict a series. This fact cast some doubts on results obtained with the employment of such structures: if it is impossible to restore a series from a discrete structure, it is impossible to assess whether the results of its inspection correspond to the chaotic time series under study or to another chaotic time series. On the other hand, real-world chaotic time series makes it necessary to deal with both series with all observations available and series with large amount of missing data.

The generalized relational tensor approach, discussed in the present paper, makes it possible to store information about a time series, generate a time series from itself, and forecast a time series as well. It allows working with irregularly sampled time series.

The novel structure compares favourably with analogous discrete structures used to represent a time series:

1. Maybe, the most important point is that the tensor generates a series with approximately the same ‘chaotic’ characteristics as the original one without resort to observations of the original series. Hence one may be sure that he analyses a discrete structure associated with this very series, not with other chaotic series.
2. The number of elements in the tensor is determined by the number of possible generalized z-vectors (see below), and hence it is rather large. In order to avoid combinatorial explosion, we propose to use algorithms based on ant colony optimization method; this makes possible to develop efficient algorithms to re-generate a time series from the tensors and predict it.
3. The model allows one to deal with both regularly and irregularly sampled data.
4. The model is seemed not to exhibit overtraining effect: as a sample size exceeds certain threshold, algorithm performance does not deteriorate.

The model seems to combine advantages of recurrence networks (the possibility to work with multivariate and irregularly sampled data) with those of coarse-graining transition networks (relatively small number of nodes even if the length of the series is high, the ability to re-generate a series from a structure).

The rest of the paper is organized as follows. The next section reviews structures utilized to represent information about a time series and methods to fill them. The third section states the problem; the fourth one introduces the concept of a generalized relational tensor. The methods employed are discussed in greater detail in the fifth section. The sixth section provides results for noisy periodic time series, standard chaotic time series, and real-world chaotic data. Finally, the last section presents conclusions.

2. Literature review

Zou et al. [50] review various discrete structures used to represent a non-Markovian time series: recurrence networks, visibility graphs, and transition networks. Recurrence networks [37, 50] employ recurrent methods to fill graphs.

The popular structure used to represent information about a time series is a visibility graph [26, 31]. Flanagan and Lacasa [12] consider visibility graphs associated with financial time series as complex networks. The Kullback-Leibler divergence calculated for distribution functions associated with a horizontal visibility graph (*HVG*) and a visibility graph allows estimation of the irreversibility of the respective time series. Gonçalves et al. [16] propose a way to extract information from *HVG*, based on amplitude differences, and show that this approach results in better characterization of real-world dynamical systems (El-Nino Southern Oscillation), making possible to represent in a proper way features important for climatologist. In addition, it requires significantly shorter time series.

Zhuang, Small, and Feng [49] also deal with visibility graphs as complex networks. The authors address the problem of community detection for these graphs in order to reveal some properties of the time series. Lan et al. [27] proposes a fast method to build a visibility graph for a given series. Li et al. [28] construct visibility graphs for the time series of fractional Brownian motions; while Gonçalves et al. [16] estimate information measures for the same series with the employment of its visibility graph. Bezsudnov and Snarskii [4] propose a parametric visibility graph (the generalization of a conventional one) and examine the influence of the view angle (the parameter) on distributions associated with the graph. Extension that is able to cope with non-stationary time series is discussed in paper by Gao et al. [13]. Budroni et al. [7] utilize a specific graph, named a chaotic graph, to describe a time series.

Transition networks algorithms employ the transition matrix between various elements of a series (observations, motifs, etc.) in order to describe the system dynamics. For instance, McCullough et al. [39] construct the ordinal networks with the employment of the Bandt and Pompe [3] representation for chaotic time series. The algorithms for compression and re-generation are also presented; in order to assess the quality of the compression characteristics of chaotic time series such as the highest Lyapunov exponent, correlation integral, etc. were used. Unfortunately, re-generation algorithm employs, in a sense, observations of the original time series. Sakellariou et al. [46] discuss algorithms to construct an ordinal network and to explore structures thereof (associated with regular and chaotic time series) as complex networks; this allows finding some intriguing relations between series characteristic and ordinal network features. Keller and Sinn [21] employ ordinal patterns in order to design an efficient way to estimate dynamical systems complexity (with the employment Kolmogorov-Sinai entropy). Amigo et al. [2] proves equivalency of metric and permutation entropy rates discrete-time stationary stochastic processes that makes possible to use the later as an estimate for the former.

Nicolis, Cantú, and Nicolis [41] discuss non-Markovian transition networks used for chaotic time-series representation. This approach can be easily extended to work with irregularly sampled data [25, 38, 45]. Kulp [24] modifies a method originally proposed by Wiebe and Virgin [48]; the method employs series spectrum. In Ref. [18], authors utilize multigraph to represent information about time series. Mutua, Gu, and Yang [36] employ (unique) adjacent matrices of z -vectors (graphlets) as nodes, while the edges are established if the second z -vector follows the first. Raut and Raeth [43] construct a complex networks in order to test whether or not a time series is nonlinear.

It is worth noting that one may relate the problem to represent a chaotic time series using the generalized relation tensors, proposed in this paper, with a popular problem to design recommendation system. Particularly, it is germane to compare with (decentralised and collaborative) clustering bandits algorithms [29]. Indeed, information that a single ‘ant’ receives during one step (in the present paper) corresponds to information than a single agent receives at one time round (in clustering bandits); a tensor element that filled by various ‘ants’ at various moments correspond to a cluster of bandits and so on. Korda, Szorenyi and Li [23] discuss distributed confidence ball algorithms for solving linear bandit problems in peer to peer networks. Authors assume that all bandits solve the same problem; this seems limit utility of the paper for our purpose. Hao et al. [19], to solve a similar problem, employs clustering to group users in ad-hoc social network environments. Mahadik et al. [32] deals with a scalable algorithm for the problem, DistCLUB. Gentile, Li and Zapella [14] investigate on-line clustering; the paper presents a strict analysis of the problem. Li, Karatzoglou and Gentile [30] examine collaborative effects. One should emphasise that, for all above papers, a payoff received by agent is determined by a linear function (usually, noised),

whereas, for the present paper, a payoff is nonlinear and determined by a series in question. This fact interferes with direct comparison of algorithm for the two problems.

We should draw a sharp distinction line between the method we use and such popular approach as a Poisson point process [9, 11, 15, 22, 34]: We employ not a single Poisson point process, but rather a set of all possible Poisson point processes, squeezed into a single discrete structure.

To sum up, despite the variety of methods to represent information about time series one can hardly find the one that re-generates a series using the filled structure only, without resort to observations of the initial time series. Another demand is that it be able to proceed with series with large amount of missing data.

3. Problem statement

For a given set X of time series of the following type:

$x = (x(t_0), x(t_1), \dots, x(t_{D_j}))$, where $x(t_i) \in R^K$ (D_j is the lengths of j th series of X ; these lengths may be different, but assumed to be large enough) and a given set of discrete structures Λ that are able to represent information about a time series, $L: X \rightarrow \Lambda$ maps a time series into a structure that represents it and $L^*: \Lambda \rightarrow X$ generates a series from a structure (the inverse mapping Campanharo et al. [8]). The set Λ may include either identical structures corresponding to different hyperparameters values, or various ones – the only requirement is the ability to implement mappings L and L^* algorithmically. Another considered object is a set of characteristics of a time series $M(x), x \in X$. For a chaotic time series, this set may include the Lyapunov exponents, the generalized entropies, fractal spectra, etc. [20, 33, 39]. One can also utilize averaged prediction errors if the algorithm is able to predict.

The goal is the following: for a given time series $x \in X$ find a structure $\lambda^* \in \Lambda$ such that

$$\lambda^* = \left\| \frac{M(x) - M(L^*(\lambda))}{M(x)} \right\|, \quad (1)$$

provided the number of parameters of $\lambda \in \Lambda$ is less than some threshold λ_{max} (defined by user)

$$|\lambda| \leq \lambda_{max}. \quad (2)$$

The expression $\frac{M(x) - M(L^*(\lambda))}{M(x)}$ symbolizes that the components used to calculate the norm are relative differences between the respective components of the vector M (characteristics of a time series) computed for the initial and re-generated series.

It implies that one attempts to develop a structure such that the initial and re-generated series are as close as possible in terms of a chosen set of time series characteristics $M(x), x \in X$. Comparing the characteristics of the initial time series with those of the re-generated time series, one should use sections of the initial time series that were not used to fill the structure in order to test its ability to generalize, not only to store information. It is possible to seek the minimum of (1) calculated either for a separate series or for a set of series $X' \subset X$ in average.

4. Generalized relational tensor

Before introducing a formal definition of a generalized relational tensor, we discuss a concept of an observation pattern. A pattern (an irregular time delay embedding scheme) is defined as a pre-set sequence of distances between positions of observations such that these (non-successive) observations are to be placed on the successive positions in a newly generated sample vector. For example, let us consider a four-point pattern (2, 3, 4). For this pattern, the two first vectors of a training set are $(x(t_0), x(t_2), x(t_5), x(t_9))$ and $(x(t_1), x(t_3), x(t_6), x(t_{10}))$; the last one is $(x(t-9), x(t-7), x(t-4), x(t))$, where $x(t)$ is the last observable value.

The vector, thus concatenated, generalizes a conventional embedding vector (z-vector) [20, 33], which corresponds to the pattern (1, 1, ..., 1) (m times). Thus, each pattern is a $S - 1$ -dimension integer vector $(p_1, \dots, p_{S-1})$, $p_j \in \{1, \dots, P_{max}\}, j = 1..S - 1$; the parameter P_{max} dictates the maximum distance between positions of observations that become successive in the vector to be generated. Thereby, the quantity $S \cdot P_{max}$ refers to a kind of the memory depth. $\aleph(S, P_{max})$ denotes a set of all possible patterns of the specified length S . Figure 1 diagrammatically shows a pattern superimposed on a time series in order to generate a sample vector. This implies that one moves a sliding comb with some teeth broken off (the distance between ‘extant’ teeth are $p_1, \dots, p_{S-1}$) along a series to obtain samples of generalized z-vectors. One should stress that embedding vectors (z-vectors) are usually composed of successive observations. Generalized z-vectors are composed of non-successive observations according to a certain pattern (irregular embedding scheme).

The generalized z-vectors proved to be efficient prediction tools in the framework of predictive clustering approach [1, 5, 36]. Predictive clustering implies that in order to predict a given position, one should seek in series for sequences similar to that immediately preceding the position in question. The final observations of such sequences are used as predictions. To implement this idea, one clusters all possible short sections of the time series and utilizes centers of the clusters (motifs) as prediction tools. Gromov and Borisenko [17] show that if one employs motifs corresponding to various patterns (generalized z-vectors), one may essentially improve prediction

Fig. 1. A pattern superimposed on the Lorenz time series in order to generate a sample vector. The top subfigure (a) shows a motif $(v_1, v_2, \dots, v_s)$; generated according to a pattern $(p_1, p_2, \dots, p_{s-1})$; the bottom subfigure (b) illustrates how a motif are determined: similar sections from various part of a series are averaged to create a motif

quality. Hence, the discrete structure that in point of fact compresses information about such motifs has aroused considerable interest.

The algorithm considered in the present paper, in contrast to known methods used to reveal motifs in time series, does not store the motifs separately, but “compresses” them into the tensor in order to reduce the number of parameters stored.

To define the generalized relational tensor, we assume that the time series observations belong to the interval $[-1, 1]$. The interval is divided into N equal subintervals; all observations of the time series are K -dimensional.

If elements of an observation $x(t_i)$ belong to intervals $i_1, i_2, \dots, i_K, i_k \in 1..N, k = 1..K$, then it corresponds to an index vector $I = (i_1, i_2, \dots, i_K)$. A set of (non-successive) observations then yields a concatenation of S K -dimensional vectors $I_j = (i_{j1}, i_{j2}, \dots, i_{jK})$.

Definition

The tensor $T \in \Lambda$ is said to correspond to the set of all patterns of the length S $\aleph(S, P_{max})$ and to K -dimensional observations, if its multi-index is a concatenation of S K -dimensional vectors $I_j = (i_{j1}, i_{j2}, \dots, i_{jK})$ and a $S - 1$ -dimensional vector $P \in \aleph(S, P_{max})$: $I = \bigcup_{j=1}^S I_j \cup P$. Indices $i_{jk}, j = 1..S, k = 1..K$ take the values $i_{jk} \in \{1, \dots, N\}$.

Definition

The generalized relational tensor T is the tensor T corresponding to $\aleph(S, P_{max})$ such that if one assigns to its multi-index values $I^* = (i_{jk} = i_{jk}^*, p_j = p_j^*)$, then one considers S observations with positions separated by distances $p_1^*, p_2^*, \dots, p_{S-1}^*$ and respective values belonging to the subintervals $i_{1k}^*, i_{2k}^*, \dots, i_{Sk}^*, k = 1..N$. The value of the respective tensor element is an estimated probability to encounter (in the time series in question) observations that belong to intervals $I_j^* = (i_{j1}^*, \dots, i_{jK}^*)$ at positions separated by distances $p_1^*, p_2^*, \dots, p_{S-1}^*$, respectively.

In the context of predictive clustering, that is if one does not “compress” motifs into a single generalized relational tensor, but rather considers all clusters for all patterns (as in Gromov and Borisenko [16]), each cluster serves as a simple statistical model, associated with a number of time series sections or, to put it differently, with a certain region of its strange attractor. A set of values of a particular component of cluster elements corresponds to a sample probability density function for a certain random variable, and the cluster itself encodes statistical relations among such variables. Naturally, the motifs (clusters’ centers) with lengths $S > 2$ make sense only if the time series in question is essentially non-Markovian. In the present paper, we consider relational tensors, which compress the information stored in the motifs into a single object. The tensor naturally comprises significantly less parameters at the cost of partially lost information. On the other hand, if one sets $S = 2$, then the relation tensor becomes a conventional transition matrix of a Markov chain.

5. Method to generate generalized relational tensor

To generate a generalized relational tensor for a time series, we employ modified ant colony optimization [10]. The central idea of the algorithm (which imitate the way ant colonies forage for food items in the wild) is to move in a search space and increase weights of all edges in the path, provided the path appears good - “an ant” deposits a certain amount of pheromone along the path in terms of the algorithm, and the amount is directly proportional to the goodness of the path. On the other hand, the probability for an ant to select an edge is directly proportional to the amount of pheromone already deposited. This algorithm combines probabilistic search, typical for evolutionary algorithms, with rather high speed of search, typical for classical optimization techniques. It is worth stressing that brute force algorithm to generate the tensor is computationally prohibitive.

The parameters of the algorithm in question, besides N, S, P_{max} , and K described above, are the following:

- The initial amount of pheromone deposited on all elements of the tensor, p_0 .
- The number of transitions that an ant makes before it deposits pheromone for the next time (during a single step of algorithm), $Q_1, Q_1 \leq S - 1$; usually, $Q_1 = S - 1$.
- The amount of pheromone that an ant deposits along the completed path, Δp .
- The total number of transitions that an ant is able to accomplish as it moves along a time series, $Q_2, Q_2 \gg Q_1$. Usually, $Q_2 \sim D$ and, therefore, an ant does not stop moving unless it reaches the end of the series.
- The amount of pheromone “evaporating” from all tensor elements (all tensor elements are decreased by the same value) after each step, $-\Delta p_a$ ($\Delta p_a \ll \Delta p$).

The procedure *Quantify* for a set of (normalized) real-valued numbers $x_{ik}, k = 1..K$, returns a set of intervals to which they belong. The procedure *ConstTensor(a)* fills all elements of the tensor with a given value a ; the operator $|T|_+$ replaces all negative elements of the tensor T with zeroes.

For the algorithm to generate the relation tensor, a time series serves as input data, while filled generalized relational tensor is its output data.

Algorithm 1 (to generate a generalized relational tensor)

All elements of the tensor T are filled with the value p_0 before the first step:

$T \leftarrow \text{ConstTensor}(p_0)$.

A step of the algorithm.

1. A random position in the time series is selected ($t_0 \leftarrow \text{Random}, t \leftarrow t_0$).
2. An ant makes Q_1 transitions. For arbitrary $q, q = 1..Q_1$, it makes the following:

$$2.1 I_q^* \leftarrow \text{Quantify}(x(t))$$

2.2 The next position is chosen probabilistically where the probability to transit to (“leap for”) p_q^* positions ahead is determined as

$$P(p_q = p_q^*) = \frac{\Sigma_{(I_{q+1}, \dots, I_S, p_{q+1}, \dots, p_{S-1})} T(I_1^*, \dots, I_q^*, I_{q+1}, \dots, I_S, p_1^*, \dots, p_{q-1}^*, p_q^*, p_{q+1}, \dots, p_{S-1})}{\Sigma_{(I_{q+1}, \dots, I_S, p_q, \dots, p_{S-1})} T(I_1^*, \dots, I_q^*, I_{q+1}, \dots, I_S, p_1^*, \dots, p_{q-1}^*, p_q, \dots, p_{S-1})}; t \leftarrow t + p_q. \quad (3)$$

3. After Q_1 transitions, the algorithm performs the following operations:

- 3.1 A pheromone is deposited along an ant’s path

$$T(I_1^*, \dots, I_S^*, p_1^*, \dots, p_{S-1}^*) \leftarrow T(I_1^*, \dots, I_S^*, p_1^*, \dots, p_{S-1}^*) + \Delta p. \quad (4)$$

- 3.2 A pheromone evaporates from all elements of the tensor

$$T \leftarrow |T - \text{ConstTensor}(\Delta p_a)|_+. \quad (5)$$

After that, a new ant is placed randomly in this time series.

Possible termination criterion can be defined by:

1. The maximum total number of ants’ steps along time series considered.
2. Small overall rate of changes of the tensor elements during several successive steps.

To generate a time series with the employment of a given generalized relational tensor, we employed the Algorithm 2. (It is suggested that $S - 1$ random observations of the initial time series are taken as initial conditions.) The parameter of the algorithm in question is Q_3 – the total number of transitions an ant can make before it is replaced by another ant.

Algorithm 2 (to generate a time series)

1. An ant is placed on the tensor element defined by the initial conditions (for the first ant) or by already filled positions (for all other ants). In the latter case, from a set of already filled observations the algorithm selects the one that immediately precedes the first unfilled one. It becomes the next start position.
2. Both position to which an ant transits and the interval to which the value of respective observation belongs are determined probabilistically as

$$P(I_S = I_S^*, p_{S-1} = p_{S-1}^*) = \frac{T(I_1^*, \dots, I_S^*, p_1^*, \dots, p_{S-2}^*, p_{S-1}^*)}{\Sigma_{(I_S, p_{S-1})} T(I_1^*, \dots, I_{S-1}^*, I_S, p_1^*, \dots, p_{S-2}^*, p_{S-1})}. \quad (6)$$

3. If the position selected is already filled, then go to 1, else continue $t \leftarrow t + p_S$.
4. The ant moves till the distance between the first position it took and the last generated is lesser than Q_3 .
5. If some time series positions are unfilled, then place a new ant to the position of an already filled observation that immediately precedes the first unfilled one and go to 1.

The algorithm outlined above gives the number of an interval only. One can calculate the concrete value as the center of the interval, a realization of the random variable uniformly distributed over this interval, or averaged observations that belong to this interval. The last option appears to be the best, but it naturally requires that these averaged values be calculated when the tensor is generated.

We should stress that while the number of parameters is comparatively large, for the most of them there exist a kind of ‘default’ values in the ACO theory, and we did not attempt to perturb them. What actually works is a pair S, P_{max} that determines ‘lengths’ of sliding combs, an initial amount of pheromone p_0 , its rate of evaporation $-\Delta p_a$, and the number of ants. As for the first parameter S , it should be equal to 3 or 4, since smaller values are not sufficient to allow definite conclusions for chaotic series, and larger values lead to combinatorial explosion. P_{max} (a maximum number of steps between neighbouring comb teeth) is reasonable to choose near 10 in order to grasp typical motifs of series; $(S - 1)P_{max}$ is a kind of memory depth. p_0 and $-\Delta p_a$ are not independent parameters; their ratio should be large enough to allow ants make a large number of movements. An alternative strategy implies that

one do not apply evaporation operation ($\Delta p_a = 0$), but subtract p_0 when all ants' movement are finished. As for the number of ants, the more is the better.

The filled generalized relational tensor can be used for time series forecasting. As the predicted value for the position t one chooses the most probable $I_S: T(I_1^*, \dots, I_{S-1}^*, I_S, p_1, \dots, p_{S-1})$

(with $I_{S-q}^* = \text{Quantify}(x(t - \sum_{j=1}^q p_{S-j}))$, $q = 1..S-1$) out of all possible patterns of $\aleph(S, P_{max})$.

It is worth noting that the algorithms prove to be efficient, provided that transient processes are completed, and thereby the time series at hand is associated with trajectories belonging to the neighborhood of a (strange) attractor, or, in any case, the time series is ε -stationary [42].

6. Numerical results

The algorithms described above were applied to generate generalized relational tensors for a noisy periodic time series, the standard chaotic time series (generated by the Lorenz system and by Mackey-Glass equations), and the real-world chaotic data (hourly load values in Germany, from 23:00 12/31/2014 to 14:00 20/02/2016, <https://www.entsoe.eu/data/power-stats/>).

Figures 2 a and 2 b show the initial non-noisy periodic time series (with the step length equal to $h = 0.1$) and the one that was generated by its generalized relational tensor. Training and testing sets comprise 100000 and 10000 observations, respectively.

Normally distributed noise, with mean 0 and variance $\xi = 0.5$, is added to observations of the initial (periodic) time series; the series is the values of a periodic function calculated. Figure 3 graphically exhibits average squared

Fig. 2. The initial (non-noisy) periodic time series (a) and the one generated by the respective relational tensor (b) (noise variance $\xi = 0.5$). The centers of the intervals are used as the values of the generated series. The average squared error is $\varepsilon = 0.02$.

Fig. 3. Noisy periodic time series. Average squared error vs. pattern length S . $N = 20$.

error vs. pattern length S for $N = 20$ (for $K = 1, p_0 = 0.1, \Delta p = 0.1, \Delta p_a = 0.0001$).

Furthermore, the algorithm was checked on the Lorenz time series. The Lorenz series is defined by the following system of differential equations:

$$\begin{cases} \dot{x}_1 = \sigma(x_2 - x_1) \\ \dot{x}_2 = x_1(r - x_3) - x_2 \\ \dot{x}_3 = x_1x_2 - bx_3 \end{cases}$$

where its parameters take conventional “chaotic” values $\sigma = 10, r = 28, b = 8/3$.

Mackey-Glass time series defined by differential equation:

$$\dot{x} = \frac{\beta x_\tau}{1 + x_\tau} - \gamma x, \gamma, \beta, n > 0$$

where $\beta=0.2, \gamma=0.1, \tau=22, n=10$ are standard values of parameters and x_τ represents the value of the variable x at time $(t-\tau)$.

To obtain the series under consideration, the Lorenz system is integrated with the employment of the fourth-order Runge-Kutta method; integration step is equal to $h = 0.1$. Training and testing sets comprise 100000 and 10000 observations, respectively.

As far as the series in question is chaotic, the pointwise comparison of the initial and re-generated series is irrelevant. To assess the tensor, we compare the following characteristics of these series: partial auto-correlation functions, the highest Lyapunov exponents, spectra of generalized entropies, and entropy-complexity pair. To estimate the highest Lyapunov exponent, TISEAN package [20, 33] is used. The spectrum of the generalized entropies is computed with the employment of the method based on the box-counting [20, 33]. To compare, Malinetskii and Potapov [33] (see also Kantz and Schreiber [20]) estimate the highest Lyapunov exponent for this series as $\lambda_1 = 0.91$; the monograph also discusses its full Lyapunov spectrum and the spectrum of the generalized entropies. For entropy-complexity pair calculation, we use the method discussed in [35, 44].

Figure 4 shows the results corresponding to the initial (Lorenz) time series (the left column) and the one generated by the respective generalized relational tensor (the right column). Subfigures 4 a and 4 b show typical sections of the series; c and d display partial auto-correlation functions; e and f present the spectra of the generalized entropies. The figure corresponds to the optimal values $K = 3, N = 200, P_{max} = 2, S = 4, p_0 = 0.1, \Delta p =$

Fig. 4. The initial (Lorenz time) series (the left column) and the one generated by the respective relational tensor (the right column). A and b show typical sections of the series; c and d display partial auto-correlation functions; e and f present the spectra of the generalized entropies. $K = 3, N = 200, P_{max} = 2, S = 4, p_0 = 0.1, \Delta p = 0.1, \Delta p_a = 0.00001$.

$$0.1, \Delta p_a = 0.00001.$$

The calculated highest Lyapunov exponent for the initial series is $\lambda_1 = 0.9$, while for the re-generated series it is $\lambda_1 = 0.91$, thus the relative error does not exceed 1.2%. The relative error for the first ten points of the auto-correlation function amounts to 21.8%, for the spectra of the generalized entropies the error equals to 3%. In particular, the estimated KS-entropies of the initial and re-generated series equal to $K_2 = 0.92$ and $K_4 = 0.94$, respectively. The entropy-complexity pair for the original time series is (0.53; 0.43); for the re-generated one (0.51; 0.43). The relative errors are, respectively, 3% and 0.08%.

Fig. 5. Typical section of the Lorenz series (blue solid line) and the respective predicted values (red dashed line) for 10 steps ahead prediction

prediction. The average relative prediction error amounts to 35%. The results are considered to be satisfactory for multi-step ahead prediction due to the exponential growth of an error intrinsic to chaotic time series.

To examine the algorithm as a means to proceed irregularly sampled data, ten percent (of randomly chosen) observations is dropped from the Lorenz sample. To this end, we use several dropping patterns corresponding to various probability distributions; please, refer to Sakellariou et al. [45] for details. Table 1 comprises

Fig. 6. The initial German hourly load time series and the one generated by the respective relational tensor. a and b show typical sections of the series; c and d display partial auto-correlation functions. $K = 3$, $N = 200$, $p_{max} = 2$, $S = 4$, $p_0 = 0.1$, $\Delta p = 0.1$, $\Delta p_a = 0.00001$.

MPEs for entropy, complexity, and the highest Lyapunov exponent as well. It makes possible to conclude that the tensor proposed is able to re-generate time series with missing observations.

Table 1. Uniform and Non-uniform sampling. Discrepancies between characteristics of the original and re-generated series.

Sampling	Entropy, MPE %	Complexity, MPE %	HLE (TISEAN), MPE %
Uniform	3	0.08	1.92
Poisson	15.5	1.3	5.48
Pareto	11.5	3.9	14.52
Gamma	15.4	1.3	14.53

Scalability of the algorithm in question is seemed to be of fundamental importance. In order to check it, we performed a large-scale simulation, for a sample size ranging from 10^5 to 10^{10} . Figure 7 displays the respective results. On the abscissa axis, for all three subfigures, a logarithm of a sample size is plotted. On the ordinate axis, Fig. 7a displays the number of non-zero elements of a tensor; Fig. 7b, a relative error for the largest Lyapunov exponent; Fig 7c, a relative error for position on entropy-complexity plane. Figure 1a shows that the number of non-zero tensor elements grows exponentially. The fact conforms to chaotic nature of the series. Namely, the longer a sample size, the more probability for the respective trajectory section to visit rarely visited regions of a strange

Fig. 7. Scalability of the algorithm. The Lorenz series. On the abscissa axis, for all three subfigures, a logarithm of a sample size is plotted (the sample size ranges from 10^5 to 10^{10} . On the ordinate axis, Fig. 7a displays the number of non-zero elements of a tensor; Fig. 7b, a relative error for the largest Lyapunov exponent; Fig 7c, a relative error for position on entropy-complexity plane

Fig. 8. Typical section of energy consumption time series (blue solid line) and the respective predicted values (red dashed line) for 10 steps ahead prediction.

attractor, those with small invariant measure, and consequently, the more typical sequences (motifs) is added to the tensor. Meanwhile, the relative error (relation of difference between quantity for re-generated and original time series to that for original time series) is nearly constant for increasing sample size, starting from a certain threshold. Quite clearly, that there is no need to enlarge the sample size over this threshold and the algorithm is scalable in this sense.

Another issue of importance is how hyper-parameters affect algorithm efficiency. To address the problem, we also performed wide-ranging simulation. Resultant plots are presented in Appendix A. It was ascertained that the number of tensor dimensions S relatively weakly influence the algorithm performance. The optimum value amounts to 3. A value of Q also scarcely affects the algorithm performance. Vice versa, the number of intervals of the range N of values strongly affects the performance. For example, for the Lorenz series, as the number of intervals ranges from

20 to 100, the relative error for the highest Lyapunov exponent ranges from 14% to 25%, and its counterpart for position on the entropy-complexity ranges from 3% to 10%. The best value for all series under study appears 60 intervals. The algorithm performance also depends strongly upon pheromone evaporation Δp_a , with the best value equal to $10^{-6} * \Delta p$. Pheromone change Δp and P_{max} affect it fairly weakly. The algorithm performance was also tested on the Mackey-Glass chaotic time series, with similar results.

Finally, the algorithm was applied to analyse chaotic real-world data: namely, energy consumption in Germany from 23:00 12/31/2014 to 14:00 20/02/2016; <https://www.entsoe.eu/data/power-stats/>). The series is chaotic as its highest Lyapunov exponent is positive; $\lambda_1 = 0.12$, the value was estimated using TISEAN package. Figure 6 shows the results corresponding to the initial time series (the left column) and the one generated by the respective generalized relational tensor (the right column). Figures 6a and 6b show the typical sections of the series; c and d display partial auto-correlation functions. The figure corresponds to the same values $K = 1, N = 200, P_{max} = 2, S = 4, p_0 = 0.1, \Delta p = 0.1, \Delta p_a = 0.00001$. The highest Lyapunov exponents for the initial and re-generated time series are $\lambda_1 = 0.12$ and $\lambda_1 = 0.18$ respectively. An entropy-complexity pairs are (0.5; 0.38) and (0.53; 0.37) that yields relative errors 4.8% and 2.1%, respectively.

The average relative error for the first ten points of the auto-correlation function is 20.5%. In order to test the algorithm's ability to generalize, we compared values of these results with those calculated for a section of the series that has not been used to fill the tensor – the respective errors appear to be 7.2% and 19.7%.

Figure 8 shows typical section of this series and the respective predicted values for 10 steps ahead prediction. The average relative prediction error amounts to 45%.

We would like to discuss limitations of the proposed method. The method implies that one seeks for motifs of the time series in question and aggregates them in a single discrete structure, and both its advantages and disadvantages rely on this fact. Apparently, this algorithm proves efficient provided data are not so noisy. On the controversy, if the noise amplitude is large, the number non-zero tensor elements grow exponentially, and the algorithm performance deteriorates.

On the completion, we would like to elaborate on potential applications and future directions. At our sight, the discrete structure that represents chaotic time series with a guarantee that its chaotic characteristics are preserved has lots of potential applications. First of all, it seems reasonable to examine these structures corresponding to time series readily interpretable in the context of their subject areas (biomedical, energy, weather time series and so forth). In such a case such structure may help reveal hidden relations and relate them to the laws that govern the subject area. In particular, it seems possible to identify time series belonging to different classes with the employment of the corresponding tensors. The second possible direction is to adjust the tensor under study and modify appropriate algorithms in order to tackle time series with a pronounced noise term (financial, for example).

Finally, the third direction is to predict many steps ahead. It is worth noting that there is a number of fairly efficient methods to predict to one step ahead. Unfortunately, when it comes to many steps, the prediction methods are hardly available due to Lyapunov instability of trajectories of chaotic dynamic systems. Meanwhile, some papers (associated mainly with predictive clustering) that discuss many-steps prediction methods [17] make it possible to expect new results in this complex and important problem. The new algorithm offers to compose motifs from non-successive observations according to certain patterns. Main drawback of this approach is an exponential growth of the number of motifs and, consequently, of computation time. From this angle, the relation tensor squeezes all these motifs into relatively compact structure. Of course, it partially loses pieces of information and deteriorates prediction quality. It would be interesting to estimate this deterioration for real-world time series.

7. Conclusions

In this paper, we discuss a generalized relational tensor approach to represent the information about a time series. The paper deals with a relational tensor to store information about a time series, to re-generate a time series using the stored information, and to predict them, for both regular and chaotic time series.

The approach combines advantages of recurrence networks (the possibility to work with multivariate and regularly sampled data) with those of coarse-graining transition networks (relatively small number of vertices even if the length of the series is high, the ability to re-generate a series from a structure).

For chaotic time series, a difference between characteristics of the initial time series (the highest Lyapunov exponent, the auto-correlation function, pair of entropy and complexity) and those of the time series re-generated from a structure is used to assess the effectiveness of the algorithm in question. The approach has shown fairly good results for periodic and benchmark chaotic time series and satisfactory results for real chaotic data.

It means that if one considers properties of such discrete structure, then one may be sure that he works with a structure corresponding with time series of interest.

The structure allows readily proceeding irregularly sampled series.

References

1. Aghabozorgi S, Shirkhorshidi AS, Wah TY (2015) Time-series clustering – A decade review. *Inf Syst* 23: 16–38. <http://doi.org/10.1016/j.is.2015.04.007>
2. Amigo JM, Kennel MB, Kocarev L (2005) The permutation entropy rate equal the metric entropy rate for ergodic information sources and ergodic dynamical systems. *Physica D* 210: 77–95. <https://doi.org/10.1016/j.physd.2005.07.006>
3. Bandt C, Pompe B (2002) Permutation Entropy: A Natural Complexity Measure for Time Series. *Phys Rev Lett* 88. <https://doi.org/10.1103/PhysRevLett.88.174102>
4. Bezsudnov IV, Snarskii AA (2014) From the Time Series to the Complex Networks: The Parametric Natural Visibility Graph. *Physica A* 414: 53–60. <https://doi.org/10.1016/j.physa.2014.07.002>
5. Blockeel H, De Raedt L, Ramon J (1998) Top-down induction of clustering trees. *Proceedings of the 15th Int Conf*: 55–63
6. Bradley E, Kantz H, Nonlinear Time-series Analysis Revisited (2015) *Chaos* 25. <https://doi.org/10.1063/1.4917289>
7. Budroni MA, Tiezzi E, Rustici M (2010) On Chaotic Graphs: A Different Approach for Characterizing Aperiodic Dynamics. *Physica A* 389: 3883–3891. <http://doi.org/10.1016/j.physa.2010.05.049>
8. Campanharo ASLO, Sirer MI, Malmgren RD, Ramos FM, Amaral LAN (2011) Duality between Time Series and Networks. *PLoS ONE* 6(8). <https://doi.org/10.1371/journal.pone.0023378>
9. Chen J, Micheas AC, Holan SH (2020) Bayesian modeling and decision theory for non-homogeneous Poisson point processes. *Spatial Statistics*, 36, 100412. <https://doi.org/10.1016/j.spasta.2020.100412>
10. Dorigo M, Gambardella LM (1997) Ant Colony System: A Cooperative Learning Approach to the Traveling Salesman Problem. *IEEE Transactions on Evolutionary Computation* 1: 53–66. <https://doi.org/10.1109/4235.585892>
11. Eckardt M, González JA, Mateu J (2021) Graphical modelling and partial characteristics for multitype and multivariate-marked spatio-temporal point processes, *Computational Statistics & Data Analysis*, 156, 107139. <https://doi.org/10.1016/j.csda.2020.107139>
12. Flanagan R, Lacasa L (2016) Irreversibility of Financial Time Series: A Graph theoretical Approach. *Phys Lett A* 380: 1689–1697. <https://doi.org/10.1016/j.physleta.2016.03.011>
13. Gao ZK, Cai Q, Yang YX, Dang WD (2017) Time-dependent Limited Penetrable Visibility Graph Analysis of Nonstationary Time Series. *Physica A* 476: 43–48. <https://doi.org/10.1016/j.physa.2017.02.038>
14. Gentile C, Li S, Zappella G (2014) Online Clustering of Bandits. *31st International Conference on Machine Learning* 3
15. Ghazal MA, Mitwalli A (2019) Time series with Poisson point process. *Applied Mathematics and Computation*, 150(1): 149-157. [https://doi.org/10.1016/S0096-3003\(03\)00205-4](https://doi.org/10.1016/S0096-3003(03)00205-4)
16. Gonçalves BA, Carpi L, Rosso OA, Ravetti MG (2016) Time Series Characterization via Horizontal Visibility Graph and Information Theory. *Physica A* 464: 93–102. <https://doi.org/10.1016/j.physa.2016.07.063>
17. Gromov VA, Borisenko EA (2015) Chaotic Time Series Prediction and Clustering Methods. *Neural Computing and Applications* 2: 307–315
18. Gromov VA, Shulga AN (2012) Chaotic Time Series Prediction with Employment of Ant Colony Optimization. *Expert Systems with Applications* 39 (9): 8474–8478
19. Hao F, Li S, Min G, Kim H, Yau SS and Yang LT (2015) An Efficient Approach to Generating Location-Sensitive Recommendations in Ad-hoc Social Network Environments. *IEEE Transactions on Services Computing* 8(3): 520-533 <https://doi.org/10.1109/TSC.2015.2401833>.

20. Kantz H, Schreiber T (2004) Nonlinear Time Series Analysis. University Press, Cambridge. <https://doi.org/10.1017/CBO9780511755798>
21. Keller K, Sinn M (2009) A standardized approach to the Kolmogorov–Sinai entropy. *Nonlinearity* 22: 2417–2422. <http://doi.org/10.1088/0951-7715/22/10/006>
22. Komori O, Eguchi S, Saigusa Y, Kusumoto B, Kubota Y (2019) Sampling bias correction in species distribution models by quasi-linear Poisson point process. *Ecological Informatics*. <https://doi.org/10.1016/j.ecoinf.2019.101015>
23. Korda N, Szorenyi B, Li S (2016) Proceedings of the 33rd International Conference on International Conference on Machine Learning 48.
24. Kulp CW (2013) Detecting chaos in irregularly sampled time series. *Chaos* 23, 033110. <https://doi.org/10.1063/1.4813865>
25. Kulp CW, Chobot JM, Niskala BJ, Needhammer CJ (2016) Using Forbidden Ordinal Patterns to Detect Determinism in Irregularly Sampled Time Series. *Chaos* 26 (2), 023107. <https://doi.org/10.1063/1.4941674>
26. Lacasa L, Luque B, Ballesteros F, Luque J, Nuño JC (2008) From Time Series to Complex Networks: The Visibility Graph. *PNAS* 105: 4972–4975. <https://doi.org/10.1073/pnas.0709247105>
27. Lan X, Mo H, Chen S, Liu Q, Deng Y (2015) Fast transformation from time series to visibility graphs. *Chaos* 25, 083105. <https://doi.org/10.1063/1.4927835>
28. Li R, Wang J, Yu H, Deng B, Wei X, Chen Y (2016) Fractal Analysis of the Short Time Series in a Visibility Graph Method. *Physica A* 450: 531–540. <https://doi.org/10.1016/j.physa.2015.12.071>
29. Li S (2016) The Art of Clustering Bandits. Dissertation, University of Insubria
30. Li S, Alexandros K, Gentile C (2016) Collaborative Filtering Bandits. Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval. <https://doi.org/10.1145/2911451.2911548>
31. Luque B, Lacasa L, Ballesteros F, Luque J (2009) Horizontal Visibility Graphs: Exact Results for Random Time Series. *Physical Review E* 80, 046103. <https://doi.org/10.1103/PhysRevE.80.046103>
32. Mahadik K, Wu Q, Li S, Sabne A (2020) Fast distributed bandits for online recommendation systems. Proceedings of the 34th ACM International Conference on Supercomputing. <https://doi.org/10.1145/3392717.3392748>
33. Malinetskii GG, Potapov AP (2002) Modern Problems of Non-linear Dynamics. Editorial URSS, Moscow (in Russian)
34. Marmarelis VZ, Berger TW (2005) General methodology for nonlinear modeling of neural systems with Poisson point-process inputs, *Mathematical Biosciences* 196(1): 1–13. <https://doi.org/10.1016/j.mbs.2005.04.002>
35. Martin MT, Plastino A, Rosso OA (2006) Generalized statistical complexity measures: Geometrical and analytical properties. *Physica A* 369: 439–462. <https://doi.org/10.1016/j.physa.2005.11.053>
36. Martinez-Alvarez F, Troncoso A, Riquelme JC (2011) Energy time series forecasting based on pattern sequence similarity. *IEEE Trans. Knowl. Data Eng.* 23: 1230–1243. <http://doi.org/10.1109/TKDE.2010.227>
37. Marwan N, Donges JF, Zou Y, Donner RV, Kurths J (2009) Complex Network Approach for Recurrence Analysis of Time Series. *Phys Lett A* 373: 4246–4254. <https://doi.org/10.1016/j.physleta.2009.09.042>
38. McCullough M, Sakellariou K, Stemler T, Small M (2016) Counting Forbidden Patterns in Irregularly Sampled Time Series. I. The Effects of Under-sampling, Random Depletion, and Timing Jitter. *Chaos* 26 (12), 123103. <https://doi.org/10.1063/1.4968551>

39. McCullough M, Sakellariou K, Stemler T, Small M (2017) Regenerating Time Series from Ordinal Networks. *Chaos* 27(3), 035814. <https://doi.org/10.1063/1.4978743>
40. Mutua S, Gu C, Yang H (2016) Visibility graphlet approach to chaotic time series. *Chaos* 26, 053107. <https://doi.org/10.1063/1.4951681>
41. Nicolis G, Cantú AG, Nicolis C (2005) Dynamical Aspects of Interaction Networks. *Int J of Bifurcation and Chaos* 15(11): 3467–3480. <https://doi.org/10.1142/S0218127405014167>
42. Orlov YN, Osminin KP (2011) Nonstationary Time Series: Prediction Methods with Examples of Financial and Raw Materials Markets. Editorial URSS, Moscow (in Russian)
43. Raut I, Raeth C (2016) Surrogate-assisted network analysis of nonlinear time series. *Chaos* 26, 103108. <https://doi.org/10.1063/1.4964646>
44. Rosso OA, Larrondo HA, Martin MT, Plastino A, Fuentes MA (2007) Distinguishing Noise from Chaos. *Physical Review Letters* 99, 154102. <https://doi.org/10.1103/PhysRevLett.99.154102>
45. Sakellariou K, McCullough M, Stemler T, Small M (2016) Counting Forbidden Patterns in Irregularly Sampled Time Series. II. Reliability in the Presence of Highly Irregular Sampling. *Chaos* 26(12), 123104. <https://doi.org/10.1063/1.4970483>
46. Sakellariou K, Stemler T, Small M (2019) Markov modeling via ordinal partitions: An alternative paradigm for network-based time-series analysis. *Phys. Rev. E* 100, 062307. <https://doi.org/10.1103/PhysRevE.100.062307>
47. Small M (2005) *Applied Nonlinear Time Series Analysis: Applications in Physics, Physiology and Finance*. World Scientific Publishing, Singapore. <https://doi.org/10.1142/5722>
48. Wiebe R, Virgin LN (2012) A heuristic method for identifying chaos from frequency content. *Chaos* 22, 013136. <https://doi.org/10.1063/1.3675624>
49. Zhuang E, Small M, Feng G (2014) Time Series Analysis of the Developed Financial Markets' Integration Using Visibility Graphs. *Physica A* 410: 483–49. <https://doi.org/10.1016/j.physa.2014.05.058>
50. Zou Y, Donner RV, Marwan N, Donges JF, Kurths J (2019) Complex Network Approaches to Nonlinear Time Series Analysis. *Physics Reports* 787: 1–97. <https://doi.org/10.1016/j.physrep.2018.10.005>

Appendix A

Supplementary Fig. 1. A relative error for position on entropy-complexity plane (a) and for the largest Lyapunov exponent (b) for different values of the parameter N

Entropy-Complexity, $P_{\max}$; $N=60$, $S=3$, $\Delta p=p_0$, $\Delta p_a=10^{-7}\Delta p$, $Q=S-1$

a

LLE, $P_{\max}$; $N=60$, $S=3$, $\Delta p=p_0$, $\Delta p_a=10^{-7}\Delta p$, $Q=S-1$

b

Supplementary Fig. 2. A relative error for position on entropy-complexity plane (a) and for the largest Lyapunov exponent (b) for different values of the parameter $P_{\max}$

Entropy-Complexity, S ; $N=60$, $P_{\max}=2$, $\Delta p=p_0$, $\Delta p_a=10^{-7}\Delta p$, $Q=S-1$

a

LLE, S ; $N=60$, $P_{\max}=2$, $\Delta p=p_0$, $\Delta p_a=10^{-7}\Delta p$, $Q=S-1$

b

Supplementary Fig. 3. A relative error for position on entropy-complexity plane (a) and for the largest Lyapunov exponent (b) for different values of the parameter S

Entropy-Complexity, Δp ; $N=60$, $P_{\max}=2$, $S=3$, $\Delta p_a=10^{-7}\Delta p$, $Q=S-1$

a

LLE, Δp ; $N=60$, $P_{\max}=2$, $S=3$, $\Delta p_a=10^{-7}\Delta p$, $Q=S-1$

b

Supplementary Fig. 4. A relative error for position on entropy-complexity plane (a) and for the largest Lyapunov exponent (b) for different values of the parameter Δp

a

b

Supplementary Fig. 5. A relative error for position on entropy-complexity plane (a) and for the largest Lyapunov exponent (b) for different values of the parameter Δp_a

a

b

Supplementary Fig. 6. A relative error for position on entropy-complexity plane (a) and for the largest Lyapunov exponent (b) for different values of the parameter Q