Expert Systems With Applications

Prediction After a Horizon of Predictability Non-Predictable Points and Partial Multi-Step Prediction for Chaotic Time Series

--Manuscript Draft--

Manuscript Number:	
Article Type:	Full length article
Keywords:	Chaotic time series; multi-step prediction; predictive clustering; predictable and non-predictable points; a horizon of predictability
Corresponding Author:	Vasilii A. A. Gromov, Ph.D., D.SC National Research University Higher School of Economics: Nacional'nyj issledovatel'skij universitet Vyssaa skola ekonomiki Moscow, RUSSIAN FEDERATION
First Author:	Vasilii A. A. Gromov, Ph.D., D.SC
Order of Authors:	Vasilii A. A. Gromov, Ph.D., D.SC
	Philip S. Baranov, MSc
	Alexandr Yu. Tsybakin
Abstract:	The paper introduces several novel strategies for multi-step prediction of chaotic time series. Generalized z-vectors (irregular embeddings), comprising non-successive observations, make it possible to obtain a fairly large set of possible predicted values for each point to be predicted. Upon examining such a set, it may possible either to calculate a unified ('final') predicted value for a particular point (thus making it a 'predictable' point) or not (in which case the point becomes 'unpredictable'). With non-predictable points, the multi-step prediction process is, in fact, a two-objective problem: on one hand, we aim to minimise the total number of unpredictable points; on the other – we try to minimise the average error among the predictable ones. The main difference between the strategies presented in this paper and their classical counterparts is the concept of 'unpredictable' points – i.e., skipping points where the predicted value will clearly be wrong and calculating predictions only for the 'good' points. The results of the present research indicate that, for such algorithms, the number of non-predictable points grows exponentially with a prediction horizon; however, an average error for predictable points tends to remain constant and relatively small up to the horizon of predictability and sometimes even farther (for benchmark and real-world time series).

Prediction After a Horizon of Predictability: Non-Predictable Points and Partial Multi-Step Prediction for Chaotic Time Series

Vasilii A. Gromov, Philip S. Baranov, and Alexandr Yu. Tsybakin

HSE University

Pokrovskii Boulevard, 11, Moscow, Russian Federation

Vasilii A. Gromov, stroller@rambler.ru (corresponding author:)

Philip S. Baranov, pbaranov1306@gmail.com

Alexandr Yu. Tsybakin, a.tsby@yandex.ru

Abstract. The paper introduces several novel strategies for multi-step prediction of chaotic time series. Generalized z-vectors (irregular embeddings), comprising non-successive observations, make it possible to obtain a fairly large set of possible predicted values for each point to be predicted. Upon examining such a set, it may possible either to calculate a unified ('final') predicted value for a particular point (thus making it a 'predictable' point) or not (in which case the point becomes 'unpredictable'). With non-predictable points, the multi-step prediction process is, in fact, a two-objective problem: on one hand, we aim to minimise the total number of unpredictable points; on the other – we try to minimise the average error among the predictable ones. The main difference between the strategies presented in this paper and their classical counterparts is the concept of 'unpredictable' points – i.e., skipping points where the predicted value will clearly be wrong and calculating predictions only for the 'good' points. The results of the present research indicate that, for such algorithms, the number of non-predictable points grows exponentially with a prediction horizon; however, an average error for predictable points tends to remain constant and relatively small up to the horizon of predictability and sometimes even farther (for benchmark and real-world time series).

Keywords. Chaotic time series, multi-step prediction, predictive clustering, predictable and non-predictable points, a horizon of predictability.

I. INTRODUCTION

Chaotic systems, both social and natural, have a widespread currency. This may explain a constantly increasing number of prediction algorithms for chaotic time series. It is worth stressing however, that while the algorithms for a single-step-ahead prediction (Aghabozorgi et al., 2015) appear to be amazingly efficient;

their counterparts for a multi-step ahead (*MSA*) prediction are still in their infancy. One may attribute this fact to an exponential growth of an average prediction error with a prediction horizon, that is the number of steps ahead to be predicted.

This exponential growth reflects Lyapunov instability, immanent to any chaotic system. Namely, according to Lyapunov instability definition, however small is any initial difference between two neighbouring trajectories, the difference grows exponentially with time, and its exponent equals to the highest Lyapunov exponent (Kantz & Schreiber, 2003; Malinetskii & Potapov, 2000). Laypunov instability also leads to another concept, that of a horizon of predictability (sometimes also referred to as Lyapunov time (Bezruchko & Smirnov, 2010)). For a given observational error $\varepsilon(0)$ and the maximum

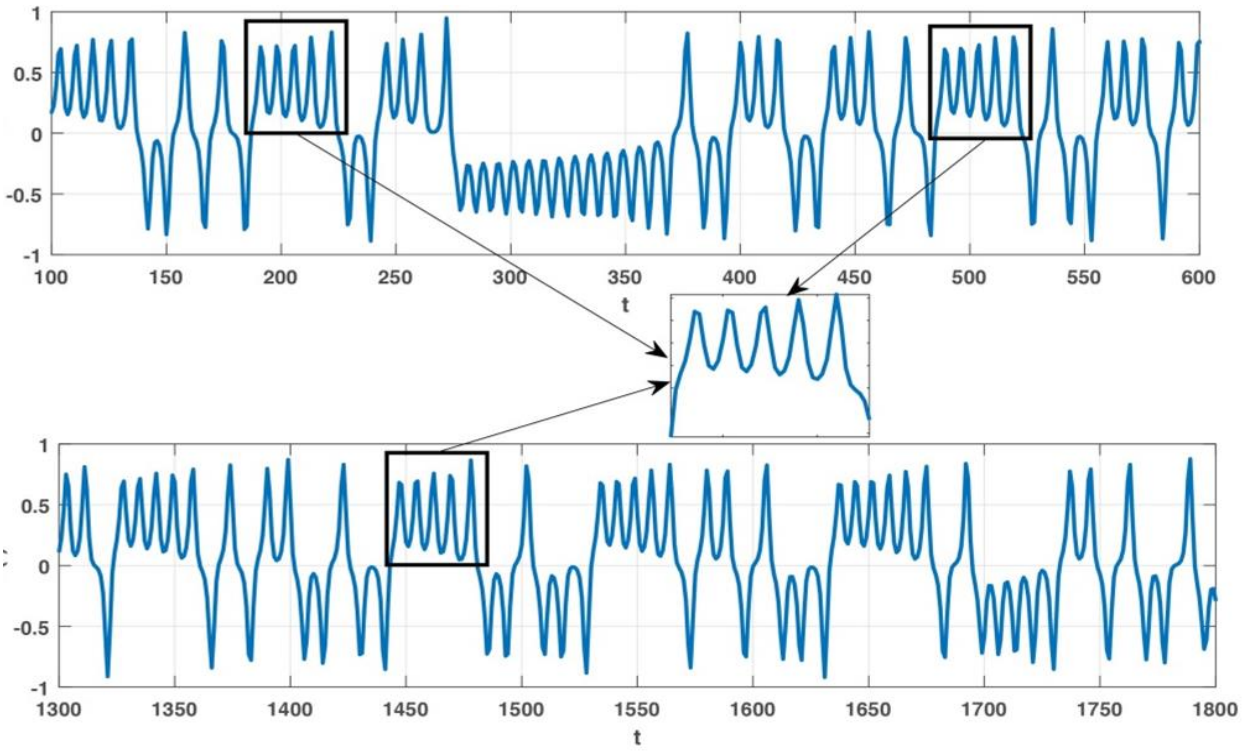


Fig. 1. Diagrammatical representation of the way algorithm searches for similar series' sections (top), obtains a cluster center (inset in the middle), and makes predictions when it finds a similar section in a test set (bottom).

prediction error ε_{max}, the exponential growth mentioned above satisfies$(T) = \exp(\lambda T)$, thereby giving a horizon of predictability $T{\sim}1/\lambda \times ln\left(\varepsilon_{max}/\varepsilon(0)\right)$ (Kantz & Schreiber, 2003; Malinetskii & Potapov, 2000). In the context of a multi-step ahead prediction, this implies that the smallest difference between true and predicted values in an intermediate point (that is the point between the last observable point and the point to be predicted) – inevitable for any prediction method – triggers exponential error growth in all subsequent intermediate points and in the point to be predicted itself. Therefore, a horizon of predictability is an upper theoretical boundary for the number of steps ahead to be predicted for any

prediction algorithm (for given $\varepsilon(0)$ and ε_{max}). A horizon of predictability should not be confused with a prediction horizon, which is a mere number of steps ahead the algorithm makes its prediction. Usually, a prediction horizon h is essentially lesser than a horizon of predictability T: $T >> h$.

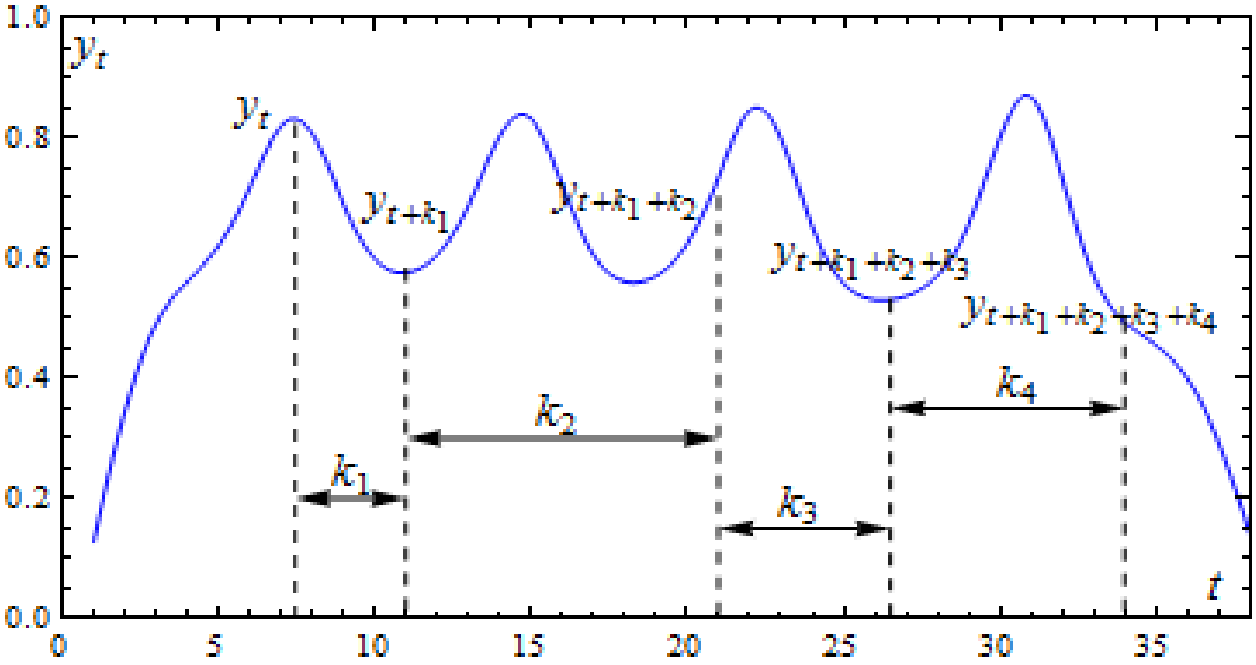


Fig. 2. A sample vector $(y_t, y_{t+k_1},\ldots,y_{t+k_1+\ldots+k_{L-1}})$, generated for a pattern $k_1,\ldots,k_{L-1}, k_i \in N$.

Nonetheless, this apparently inevitable exponential growth only seems to be so. Predictive clustering (Blockeel & De Raedt, 1998; Gromov & Borisenko, 2015) furnishes several tools that, applied together, make it possible to avoid it.

First, predictive clustering uses motifs – repeated sequences of data in a series (with some small variations). If a section of a time series resembles first part of a motif 'closely enough', it is likely that the subsequent points of a series will be close to the subsequent points of the motif, allowing one to use motifs to predict next points of the time series. Motifs can be obtained by clustering vectors of non-successive observations of a series into z-vectors (irregular embeddings (Small, 2005)) and calculating their respective centers. It is worth stressing that most available prediction methods attempt to construct a unified, global prediction model, whereas predictive clustering methods, vice versa, build up a set of separate, local prediction models, motifs (cf. global vs. local learning (Ben Taieb et al., 2010; Gromov, 2019)).

Second, predictive clustering allows one to develop the concept of non-predictable points in a natural way (Gromov & Borisenko, 2015). Initially, an unpredictable point was the fone for which the algorithm failed to find a corresponding motif that would be close enough and based on which the possible value of the point could be estimated (a set of possible predicted values is empty for it. It is worthy to emphasize that if one introduces the concept of non-

predictable points (non-predictables), one thereby renders the problem two-objective: one must minimise simultaneously the number of non-predictable points and the average error over predictable ones. In the conventional statement, the first objective is discarded – predictions are made for every single point. Introduction of unpredictable points proved to have a positive effect on the results – it is better for algorithm to skip some points along the way for which it is unable to find a proper predicted value rather than forcing it regardless of whether this predicted value will be accurate or not. This approach is actually quite natural in some areas: for example, traders are not compelled to buy and/or sell at every single moment – instead, they may choose only those moments in time that the algorithm has recognized as predictable.

In the context of dynamical systems (Kantz & Schreiber, 2003; Malinetskii & Potapov, 2000), each cluster of z-vectors (by definition, it includes similar sections of a trajectory) corresponds to a particular area of a strange attractor. Furthermore, areas with high values of an invariant measure are associated with large clusters (i.e., with frequently observed motifs). Similarly, areas with smaller values of the invariant measure are associated with smaller clusters (i.e., rarely observed motifs). Consequently, as a training set increases – so does the number of clusters, while the number of unpredictable points, as well as the average error of the predictable ones, decrease. A large-scale simulation for the Lorenz series (Gromov, 2019) supports this conclusion.

Third, Gromov and Borisenko (2015) use generalized z-vectors – vectors comprised of non-successive observations according to certain predefined patterns (irregular embeddings (Small, 2005)). A pattern is defined as a sequence of distances between positions of observations in a series, such that these (non-successive) observations become successive in a generated vector. This concatenated vector generalizes a conventional z-vector (Kantz & Schreiber, 2003; Malinetskii & Potapov, 2000), which corresponds to the pattern $(1,1,..,1)$ ($L-1$ times). Figure 1 presents a sample vector constructed according to the pattern $k_1, \dots, k_{L-1}, k_i \in N$ and shows the way z-vectors are clustered into motifs and how the motifs can be used to obtain predictions for the time series at hand. The process of generating a z-vector can be visualized as placing a 'comb' with L teeth where tooth L is 'broken' and the distances between the remaining $L-1$ teeth correspond to some pattern. Moving the comb along the series, we, for each position t, obtain a vector of observations, to be under the teeth of the comb, $\left(y_t, y_{t+k_1}, \dots, y_{t+k_1+\dots+k_{L-1}}\right)$. To put it differently, for a given pattern, a conventional sliding window is replaced by a sliding comb with the majority of its teeth broken off. A set of such vectors comprises a sample corresponding to a pattern of $k_1, \dots, k_{L-1}, k_i \in N$. Each such sample is clustered separately.

Fourth, due to the fact that the number of patterns can be significant, so can the number of possible predicted values at each point; however, they are not always statistically independent. This makes it possible to design a great deal of various algorithms to determine a unified prediction value for a set of possible prediction values (for instance, by averaging them). It also becomes possible to extend the concept of the unpredictable points: apart from the points that do not have a corresponding motif that would be close enough, we can also consider points for which it is impossible to calculate a unified prediction value based on the set of possible predicted values to be unpredictable as well. An example case of such a point would be a point which set of predicted values consists of two equally sized clusters with different centres. It is worth stressing that in our previous paper (Gromov & Borisenko, 2015) we employed non-predictable points of the first kind only (a point is non-predictable if there are no motifs close enough to make predictions for it). In the present paper, we employ both kinds of non-predictable points – and this appears to make it possible for the proposed algorithm to predict for so many steps ahead.

It is worth noting that since patterns with a distance between penultimate and ultimate teeth greater than 1, $k_{L-1} > 1$, exist, the fact that a position t is non-predictable does not imply that positions $t + 1$, $t + 2$, etc. would be non-predictable too. This fact is of fundamental importance for multi-step ahead prediction: If one needs to predict a point at a position $t + h$, one would use the iterative strategy (or rather its modification based on patterns of non-successive observations). Namely, one predicts values for intermediate positions between t and $t + h$ step by step, identifying unpredictable points along the way and stepping over them. This results in unpredictable sections between predictable positions. We used to refer to the former as a ‘bog’, and to the latter as ‘hillocks’. The entire prediction process is likened to jumping from a hillock to hillock in order to navigate a swamp (or to put it differently – using stepping stones).

While the algorithm cannot produce a prediction at each point (thus the word ‘partial’ in the title), it can predict values up to the horizon of predictability and sometimes even further. While the algorithm cannot produce a prediction at each point (thus the word ‘partial’ in the title), it can predict values up to the horizon of predictability and sometimes even further. This paper presents several methods of identifying unpredictable points and corresponding strategies for the multi-step ahead prediction as well as demonstrates partial predictions beyond the horizon of predictability for the benchmark and real-world time series.

The rest of the paper is organized as follows. The next section reviews recent advances in the field; the third section formally states the problems under study; the fourth outlines clustering method, and methods employed to obtain predictions and also identify non-predictable points; the fifth section provides the prediction results for a time series generated by the Lorenz system, and a time

series of hourly load values in Germany. Finally, the last section presents conclusions.

II. RELATED WORKS

Recent papers that concern themselves with chaotic time series prediction are quite numerous (Aghabozorgi et al., 2015; Bao et al., 2014; Ben Taieb et al., 2010; Gromov & Borisenko, 2015; Sangiorgio & Dercole, 2020; Ye & Dai, 2019) but most of them only deal with one-step ahead prediction. At the same time, the number of papers tackling the problem of the multi-step ahead prediction (MSA) is considerably lower. This could be attributed to the exponential growth of the average prediction error with an increasing prediction horizon.

An MSA prediction algorithm for chaotic time series usually consists of two parts, responsible collectively for the prediction quality. The first is a technique used for a one- or few-step-ahead prediction. The second is a strategy utilized to 'assemble' one- or few-step ahead predictions to a multi-step ahead prediction.

Algorithms used for a one-step-ahead prediction employs concepts of nearly all fields of data mining and machine learning: support vector regression and its modifications (Bao et al., 2014), multilayer perceptron (Chandra et al., 2017; Mirzaee, 2009), cluster centres in predictive clustering (Gromov et al., 2017; Gromov & Konev, 2017; Gromov & Shulga, 2012; Martínez-Álvarez et al., 2011), LSTM neural networks (Sangiorgio & Dercole, 2020), Voronoi regions partitioning (Kurogi et al., 2014), ridge polynomial neural networks (Waheeb & Ghazali, 2016), wavelet neural networks (Guntu et al., 2020; Ong & Zainuddin, 2019), dilated convolution networks (R. Wang et al., 2020), to name just a few. Dilated convolution networks could be viewed as counterparts of patterns of non-successive observations (see below).

Another factor of fundamental importance is the strategies of MSA prediction. The problem can be solved using two basic strategies – the iterated (recursive) strategy and the direct (Ben Taieb et al., 2012; Cox, 1961) strategy. The iterated one implies that the multi-step ahead prediction process is carried out step-by-step, calculating new predicted values based on the predictions made in the previous steps. The direct strategy aims for getting the results immediately, without calculating predicted values for the intermediate positions. The strategy applies prediction techniques for various prediction horizons, thereby providing multiple predictions for a position to be predicted. Taieb, Sorjamaa, and Bontempi (2010) review different methods based on these two basic strategies. Sangiorgio and Dercole (2020) apply both strategies with multilayer perceptrons and LSTM nets employed as tools to predict for a one step ahead.

Unfortunately, MSA methods designed in the framework of the aforementioned strategies are, in one way or another, amenable to 'a curse of an exponential growth,' discussed above. This gives rise to a number of hybrid

strategies aiming to resolve this problem. Ben Taieb et al. (2012), in their brilliant review, compare two basic and three novel strategies (DirRec, MIMO, and DIRMO). DirRec (Direct + Recursive) combines two basic strategies to the effect that it uses the direct approach to predict values, but the number of inputs is enlarged iteratively to include values for just predicted positions. When it comes to the MIMO (Multiple Input Multiple Output) strategy (Universit & Bontempi, 2014), an array of values is produced for the positions between the observed values and a prediction horizon inclusive instead of only a single value corresponding to the prediction horizon. This allows any correlations that the time series in question might have to be stored in the predicted values.

Chaotic time series prediction methods that rely upon reservoir computing would mark a different approach (Jaeger, 2004; Lu et al., 2017). Lu et al. (2017) indicate that the method demonstrates excellent results for small prediction horizons; however, their performance severely worsens when applied to larger horizons. Nevertheless, a predicted sequence 'behaves like the measurements from a typical trajectory on the attractor' and this approach converges quickly (Canaday et al., 2018). DIRMO (DirRec + MIMO) divides the series up to a prediction horizon into blocks and applies MIMO strategy separately to each block. Authors test these five strategies on a reasonably large sample of different time series (NN5 competition), which reflects various irregularities inherent to time series.

Bao, Xiong, and Hu (2014) compare the efficiency of the iterated, direct, and MIMO strategies by performing a one-step-ahead prediction using a modified support vector regression. According to the authors, the MIMO compares favourably with two other strategies (all other factors being equal).

Multiple-task learning can be viewed as a kind of strategy for multi-step ahead prediction. Chandra, Ong, and Goh (2017) propose an algorithm to determine a neural network structure to solve MSA prediction problems; their approach can be considered as a combination of the direct and iterated strategies. Wang et al. (2019) utilize a neural model in order to combine periodic approximations for longer periods and machine learning models for shorter periods. This could be viewed as a separate strategy when dealing with data with a marked periodicity. Ye and Dai (2019) employ multi-task learning for multi-step prediction; it is possible to view multi-task learning as a kind of MSA predicted strategy. Kurogi, Shigematsu, and Ono (2014) make use of an out-of-bag model for selecting models for multi-step prediction. Authors aim at predicting a chaotic series as far as it is possible (the largest possible prediction horizon) while retaining reasonable accuracy. Authors present results for the Lorenz series. The present paper discusses a few novel strategies; the main difference from their classical counterparts is non-predictable points, and consequently, an ability to

not account for predictions at intermediate positions that are clearly erroneous (Gromov & Borisenko, 2015).

III. PROBLEM STATEMENT

The paper considers a h -step-ahead prediction problem for a chaotic time series $Y = \{y_0, y_1, \ldots, y_t, \ldots\}$, $h \in N, h > 0$. We assume that all transient processes in the system that generate the time series have been completed, and the time series reflects the trajectory movement in the neighbourhood of a strange attractor, however complex it can be. The second assumption is that the series meets Takens' theorem conditions and, consequently, it is possible analyse the attractor structure using time series observations (Kantz & Schreiber, 2003; Malinetskii & Potapov, 2000).

A series is divided into two parts: Y_1 and Y_2, with $Y_1 = Y_1(t) = \{y_0, y_1, \ldots, y_t\}$ being an observable part, used as a training set a prediction model and Y_2 being a test set); $Y = Y_1 \cup Y_2$, $Y_1 \cap Y_2 = \emptyset$. When the algorithm predicts a value at the $t + h$ position of the test set, it possesses information about observations from $t - s + 1$ up to t inclusively, it *does not* possess information observations about positions $[t + 1, \ldots, t + h - 1]$. $\hat{y}_{t+h} = \hat{y}_{t+h}(\{y_t, y_{t-1}, \ldots, y_{t-s+1}\})$, where s is a parameter of the algorithm.

Predictive-clustering algorithms and a large number of used patterns make it possible to build up a set of possible predicted values $\hat{S}_{t+h} = \left\{\hat{y}_{t+h}^{(1)}, \ldots, \hat{y}_{t+h}^{(N_{t+h})}\right\}$ for each position to be predicted. N_{t+h} is the number of possible predicted values found by an algorithm, where $\hat{y}_{t+h}^{(i)}, i = 1..N_{t+h}$ is the ith predicted value. A set $\hat{S}_{t+h}^{(p)}$ is defined as $\hat{S}_{t+h}^{(p)} = \{\hat{S}_{t+h}, \ldots, \hat{S}_{t+h-p+1}\}$ and is comprised of sets of possible predicted values for the position $t + h$ itself and for $p - 1$ preceding positions, with p being a parameter of the algorithm. It usually takes values $p = 1$ or $p = h$. For $p = 1$, the set $\hat{S}_{t+h}^{(p)}$ consists of the set of possible predicted values for the position $t + h$ only; for $p = h$, it consists of sets for the position $t + h$ itself and for all intermediate positions between the last observable point and the position $t + h$. We denote an algorithm that builds up sets of possible predicted values $\hat{S}_{t+h}$ as f_h: $\hat{S}_{t+h} = f_h(\{y_t, \ldots, y_{t-s+1}\})$.

The concept of non-predictable points implies that one applies two operators to the set $\hat{S}_{t+h}^{(p)}$. The first checks whether the position is predictable:

$$\zeta\left(\hat{S}_{t+h}^{(p)}\right) = \begin{cases} 1, if\ \ position\ \ is\ predictable \\ 0, otherwise \end{cases} \quad (1)$$

where $\zeta(\emptyset) = 0$ for any function ζ.

The second determines a unified predicted value for a set of possible predicted values, provided the position appears to be predictable:

$$\hat{y}_{t+h} = g\left(\hat{S}_{t+h}^{(p)}\right) \quad (2)$$

In these terms, a multi-step ahead prediction problem can be stated as a two-objective optimization problem:

$$I_1 = min \sum_{t+h \in Y_2} \left(1 - \zeta\left(\hat{S}_{t+h}^{(p)}\right)\right) \quad (3)$$

$$I_2 = min(1/|Y_2|) \sum_{t+h \in Y_2} \zeta\left(\hat{S}_{t+h}^{(p)}\right) \left\| g\left(\hat{S}_{t+h}^{(p)}\right) - y_{t+h} \right\|^2 \quad (4)$$

For both functionals, one sums over all observations of a test set. The first functional minimises the number of non-predictable points, while the second one minimizes an average error for the predictable ones. In this paper, authors attempt to solve the two-objective optimization problem (3), (4).

IV. PREDICTION ALGORITHM

This section describes the proposed prediction algorithm. The first subsection deals with a way to generate samples from the time series according to predefined patterns; the second one talks about the employed clustering techniques. The third, fourth, and fifth provide ways of generating a set of possible predicted values, quality measures of identification of unpredictable points, and techniques of identification of unpredictable points respectively. Each technique in this section is denoted by a label in square brackets (for example, [p]) for ease of reference in the next section – Numerical Results.

2.1. Training set

The series is considered to be normalized. The concept of patterns is used to generate samples. A pattern is defined as an array of distances between positions of observations such that these (non-successive) observations become successive in a sample vector. Thus, each pattern is a $L-1$-dimensional vector of integers $(k_1, k_2, \ldots, k_{L-1})$, $1 \le k_j \le K_{max}$; the parameter K_{max} dictates the

maximum distance between positions of observations that become neighbours in the vector to be generated. Thereby, the quantity $(L-1)K_{max}$ symbolizes a kind of memory depth. $\aleph(L, K_{max})$ denotes a set of all possible patterns of length L. The vector thus concatenated generalizes a conventional z-vector (Kantz & Schreiber, 2003; Malinetskii & Potapov, 2000), which corresponds to the pattern (1, 1, …, 1) ($L-1$ times).

For example, let us consider a four-point pattern (2, 3, 4). For this pattern, the two first vectors of a training set would be (y_0, y_2, y_5, y_9) and (y_1, y_3, y_6, y_{10}); the last one vector would then be $(y_{t-9}, y_{t-7}, y_{t-4}, y_t)$, where y_t is the last observable value.

Conventionally, one used, for predictive clustering, vectors that concatenated successive observations (z-vectors); they proved less efficient (Gromov & Borisenko, 2015) than those based on the vectors concatenated according to various patterns, generalized z-vectors, discussed above. One may attribute this to the fact that vectors of non-successive observations are able to store information about salient observations (minima, maxima, tipping points, and so forth), as well as correlations between them. Figure 2 above shows a pattern superimposed on a time series in order to generate a sample vector.

It is possible to use either the entire set of all combinatorically possible patterns $\aleph(L, K_{max})$ for generating training sets or its part consisting of randomly chosen patterns. The patterns should be chosen before simulation starts. A set of used patterns is denoted as $\beta\aleph(L, K_{max})$, where β is a percentage of patterns used. In particular, $100\%\aleph(L, K_{max})$ corresponds to the case of all possible patterns being used.

Training sets are generated independently for each pattern $\alpha \in \beta\aleph(L, K_{max})$. Motifs may be extracted from each training set by two possible ways. The first one utilizes vectors of a training set as motifs [*p*] (pointwise); this approach bears a close resemblance to lazy learning (Ben Taieb et al., 2012). The second implies that a sample is clustered, and centres of the clusters are used as the motifs in question [*cl*] (cluster); except as otherwise noted, the clustering technique is the modified Wishart (please, refer to the next subsection). Despite its drawbacks, the first method compares favourably with the second in terms of required computations.

We denote a set of motifs corresponding to a pattern $\alpha = \left(k_1^{(\alpha)}, \ldots, k_{L-1}^{(\alpha)}\right)$, $\alpha \in \beta\aleph(L, K_{max})$ as $\Psi_\alpha = \{C_\alpha\}$, $C_\alpha = \left(\eta_1^{(\alpha)}, \ldots, \eta_{L-1}^{(\alpha)}\right)$. The motifs obtained this way can be used only with the respective pattern α, which indicates distances between positions in a series, such that values $\eta_1^{(\alpha)}, \ldots, \eta_{L-1}^{(\alpha)}$ may occupy. A set of all used motifs is denoted as $\Psi = \{(\alpha, \Psi_\alpha)\}$, $\alpha \in \beta\aleph(L, K_{max})$.

2.2. Clustering technique

As the trajectory of a system moves through the same area of an attractor repeatedly, similar sequences (associated with that area) can be repeatedly encountered in the time series. Revealing and describing such areas, as well as developing simple prediction models for each of them, makes it possible to predict chaotic time series up to a considerable limit (Gromov & Shulga, 2012). The clustering method presented below is employed to collect sequences belonging to the same cluster.

We employ the Wishart clustering technique (Bock, 1970; Wishart, 1969) as modified by Lapko and Chentsov (Lapko & Chentsov, 2000) to cluster vectors. The method employs graph theory concepts and a non-parametric probability density function estimator of r -nearest neighbours. Some difficulties associated with the application of the algorithm to forecasting problems are discussed in (Gromov & Borisenko, 2015).

An estimated saliency for a point x is defined as $p(x) = r/(V_r(x)n)$, where $V_r(x)$ and $d_r(x)$ are the volume and radius of the minimum hypersphere with its centre at point x containing at least r observations. The method relies upon a similarity relation (proximity) graph $G(Z_n, U_n)$, vertices of which correspond to samples and edges defined as $U_n = \{(x_{i,}x_j): d(x_{i,}x_j) \leq d_r(x_i), i \neq j\}$. $G(Z_q, U_q)$ is a generated subgraph for graph $G(Z_n, U_n)$, with a vertex set $Z_q = \{x_j, j = 1..q\}$ and an edge set that comprises all edges from U_n, such that their vertices belong to the set Z_q. $w(x_q)$ is a cluster number (label) for x_q. A cluster c_l ($l > 0$) is said to be a height-significant one (with respect to height value $\mu > 0$) if $max_{x_i,x_j \in c_l}\{|p(x_i) - p(x_j)|\} \geq \mu$.

Thus, the algorithm consists of the following steps:

1. Determine distances $d_r(x_q)$ for each sample to its r-nearest neighbor and sort the samples in ascending order according to their respective $d_r(x_q)$;
2. Set $q = 1$;
3. For each subgraph $G(Z_q, U_q)$ the following alternatives are possible:
   3.1. If a vertex x_q is an isolated vertex of $G(Z_q, U_q)$, then start a new cluster;
   3.2. If a vertex x_q is connected to vertices of clusters $l_1, l_2, \ldots, l_a, a > 1$:
      3.2.1. If all a clusters are completed, then set $w(x_q) = 0$.
      3.2.2. Otherwise, determine the number of significant clusters $\kappa(\mu) \leq a$:

3.2.2.1. If $\kappa(\mu) > 1$ or $l_1 = 0$, then set $w(x_q) = 0$, label significant clusters as completed and delete insignificant clusters, setting $w(x_q) = 0$ for all samples that belong to them.

3.2.2.2. Otherwise, merge clusters $l_2, \ldots, l_a$ into l_1, setting $w(x_q) = l_1$ for the sample itself and for all samples belonging to them.

4. Set $q = q + 1$. If $q \leq n$, then go to step 3.

We used the following values: $r = 11$, $\mu = 0.2$.

2.3. Sets of possible predicted values ($\hat{S}_{t+h}^{(p)}$)

For a given pattern $\alpha \in \beta\aleph(L, K_{max})$, $\alpha = \left(k_1^{(\alpha)}, \ldots, k_{L-1}^{(\alpha)}\right)$, we consider a set of all corresponding motifs Ξ_α and, using them, builds up a set of possible predicted values $\hat{S}_{t+h}^{(p,\alpha)}$ associated with the pattern α. Namely, for a given position to be predicted $t+h$, we compose a vector from time series observations according to the pattern α: $C = \left(y_{t+h-k_{L-1}^{(\alpha)}}, y_{t+h-k_{L-1}^{(\alpha)}-k_{L-2}^{(\alpha)}}, \ldots, y_{t+h-k_{L-1}^{(\alpha)}-k_{L-2}^{(\alpha)}-\ldots-k_1^{(\alpha)}}\right)$. All elements of the vector C are assumed to be known – they may be either observed or already predicted values. Then if the Euclidean distance between a vector C and a truncated motif $TruncC_\alpha$, $C_\alpha \in \Xi_\alpha$ (a truncated motif comprises all elements but the last one, $TruncC_\alpha = \left(\eta_1^{(\alpha)}, \ldots, \eta_{L-2}^{(\alpha)}\right)$ for $C_\alpha = \left(\eta_1^{(\alpha)}, \ldots, \eta_{L-1}^{(\alpha)}\right)$), then the last element of the motif C_α, $\eta_{L-1}^{(\alpha)}$, is a reasonable estimate of the value to be predicted.

Respectively, the set $\hat{S}_{t+h}^{(p,\alpha)}$ is determined as $\hat{S}_{t+h}^{(p,\alpha)} = \left\{\eta_{L-1}^{(\alpha)} : \rho(C, TruncC_\alpha) \leq \varepsilon\right\}$, where ε is a small threshold. In turn, a set of possible predicted values for a position $t+h$ is a union of the sets $\hat{S}_{t+h}^{(p,\alpha)}$ over all possible patterns, $\hat{S}_{t+h}^{(p)} = \bigcup_{\alpha \in \beta\aleph} \hat{S}_{t+h}^{(p,\alpha)}$. The unified predicted value $\hat{y}_{t+h}$ is calculated as a function of the set of possible predicted values for this position $\hat{S}_{t+h}^{(p)}$.

Previously predicted values at intermediate positions $\hat{y}_{t+i}$ are used as inputs for new iterations of the algorithm until some prediction horizon h is reached. Let us call each iteration a 'prediction operation' – a generalized step in

the iterated strategy. Once again, it is not necessary to make predictions for all intermediate positions – some may be skipped.

A repeated application of this procedure produces an algorithm that implements an operator f_h: $\hat{S}_{t+h} = f_h(\{y_t, \ldots, y_{t-s+1}\})$, which yields a set of possible predicted values for a position $t+h$ to be predicted. Since the number of patterns is rather large, so the size of the set may be. This makes it possible to design various algorithms to determine a unified predicted value and identify non-predictable points. These algorithms are discussed in greater detail later.

Apart from sets of possible predicted values, we can also calculate a set of weights $\Omega_{t+h} = \left\{\omega_{t+h}^{(i)}\right\}, i = 1..N_h$, which characterize comparative significance of possible predicted values $\hat{y}_{t+h}^{(i)}$. We utilize several alternatives to determine weights, based upon the following ideas:

1. The first technique relies on the notion of 'prediction length' required to obtain a predicted value $\hat{y}_{t+h}^{(i)}$. The algorithm introduces some error with each run of the prediction operation, therefore making the result less reliable the more times the prediction operation has been performed. Since the distances between the first and last positions for various patterns may differ significantly for large h, the number of prediction operations carried out to obtain various $\hat{y}_{t+h}^{(i)}$ values may significantly differ as well. To calculate this quantity, we use the following recurrence relation. We assign unity weights $\omega_i = 1$ to observed values; then, if a possible predicted value $\hat{y}_{t+h}^{(i)}$ is obtained using a pattern $\alpha = \left(k_1^{(\alpha)}, \ldots, k_{L-1}^{(\alpha)}\right)$ and the values $y_{t+h-k_{L-1}^{(\alpha)}}, y_{t+h-k_{L-1}^{(\alpha)}-k_{L-2}^{(\alpha)}}, \ldots, y_{t+h-k_{L-1}^{(\alpha)}-k_{L-2}^{(\alpha)}-\ldots-k_1^{(\alpha)}}$ (observed or predicted) are used as inputs for this prediction operation have weights $\omega_{t+h-k_{L-1}^{(\alpha)}}, \omega_{t+h-k_{L-1}^{(\alpha)}-k_{L-2}^{(\alpha)}}, \ldots, \omega_{t+h-k_{L-1}^{(\alpha)}-k_{L-2}^{(\alpha)}-\ldots-k_1^{(\alpha)}}$, then one calculates the weight of these value as an average of these weights times a step-down factor λ:

$$\omega_{t+h}^{(i)} = \lambda/(L-1)\sum_{l=1}^{L-1}\omega_{t+h-\sum_{j=1}^{l}k_j^{(\alpha)}} \qquad (5)$$

We assign average weight of all possible predicted values that take part in its calculation to the unified predicted value. Thus, the step-down factor λ ensures that values calculated with a larger number of prediction operations receive smaller weights. In the simulation, λ usually takes up a value of $\lambda = 0.99$ [*wl*] (weighted sum, length).

2. The second technique suggests that a weight ω_i is inversely proportional to the distance between observed values and the centre of the cluster chosen to make the prediction:

$$\omega_i = \left(\varepsilon - \rho(C, TruncC_\alpha)\right)/\varepsilon \quad (6)$$

, where ε is a small parameter used as a threshold for building up a set of possible predicted values. In the simulation, ε usually equals $\varepsilon = 0.05$ (for a normalized series) [*wd*] (weighted sum, distance).

3. The third approach suggests that a weight ω_i is a product of weights of the first and second approaches [*w*] (weighted sum, combined).

2.4. A unified predicted value ($\hat{y}_{t+h} = g\left(\hat{S}_{t+h}^{(p)}\right)$ (1))

The basic dichotomy for algorithms when determining a unified predicted value is whether $p = 1$ or $p > 1$. If $p = 1$, the unified predicted value is determined using a set of possible predicted values associated with the current position, $\hat{y}_{t+h} = g\left(\hat{S}_{t+h}^{(p)}\right) \equiv g\left(\hat{S}_{t+h}\right)$ [*S*] (set). If $p > 1$, it also employs the sets associated with preceding positions [*T*] (trajectory).

For the case of $p = 1$, several techniques to determine the unified predicted value for a set of possible predicted values $\hat{S}_{t+h}$ are possible:

- Calculating an average value: $\hat{y}_{t+h} = \sum_{i=1}^{N_{t+h}} \hat{y}_{t+h}^{(i)} / N_{t+h}$. [*av*] (an average)
- Calculating a weighted average value: $\hat{y}_{t+h} = \sum_{i=1}^{N_{t+h}} \omega_i \hat{y}_{t+h}^{(i)} / \sum_{j=1}^{N_{t+h}} \omega_j$ [*wl, wd*, or *w*] (please, see above).
- Clustering a set $\hat{S}_{t+h}$ in order to select the largest cluster Q_{j^*}: $\hat{S}_{t+h} = \bigcup_j Q_j$, $Q_i \cap Q_j = \emptyset$, $q_j = \left|Q_j\right|$, $j^* = \underset{j}{argmax} q_j$. The modified Wishart (Wishart, 1969) and DBSCAN (Maronna, 2016) are employed. The latter gives a good account of itself for one-dimensional data, that is fully applicable to a set $\hat{S}_{t+h}$. Both algorithms do not require *a priori* information about the true number of clusters. The unified predicted value is calculated as $\hat{y}_{t+h} = \sum_{\hat{y}_{t+h}^{(i)} \in Q_{j^*}} \hat{y}_{t+h}^{(i)} / \left|Q_{j^*}\right|$ [*cm*] (maximum cluster).

4. This technique is similar to the previous one with one exception: we cluster only those elements of $\hat{S}_{t+h}$, which weights are greater than some threshold ω_0. Alternatively, one may discard υ percent of observations with the least weights, regardless of the weights absolute values [*cw*] (a maximum cluster, weighted).

5. The two last techniques can be readily extended to the fuzzy sets. Normalized weights could be viewed, quite naturally, as values of a membership function to belong to a set of possible predicted values. A clustering algorithm should be replaced by some algorithm that can cluster fuzzy data [*fs*] (a fuzzy set).

6. Roulette wheel. Similarly to the previous techniques, we cluster a set of possible predicted values, but probabilistically choose the cluster which centre will be used as the unified predicted value. Probability for a cluster to be chosen is

directly proportional to the number of elements it contains or to the normalized sum of weights associated with its elements [*rw*] (a roulette wheel).

7. Most frequent value. We divide a range of values of the series in question, into disjoint intervals of equal lengths ε: $Q_i \cap Q_j = \emptyset$, $q_j = \left|Q_j\right|$, $j^* = \underset{j}{argmax} q_j$. An average value calculated over the elements of the most 'populated' interval, $\hat{y}_{t+h} = \sum_{\hat{y}_{t+h}^{(i)} \in Q_{j^*}} \hat{y}_{t+h}^{(i)} / \left|Q_{j^*}\right|$, is the unified predicted value [*mf*] (the most frequent).

8. Randomly perturbed most frequent value. This approach is essentially the same as the previous one, with the exception of adding a normally distributed noise to the average value: $\hat{y}_{t+h} = \sum_{\hat{y}_{t+h}^{(i)} \in Q_{j^*}} \hat{y}_{t+h}^{(i)} \left|Q_{j^*}\right| + \zeta(\Delta)$, where $\zeta(\Delta)$ is a normally distributed random variable with zero mean and variance $\Delta \geq 0$ [*mp*] (the most frequent, perturbed).

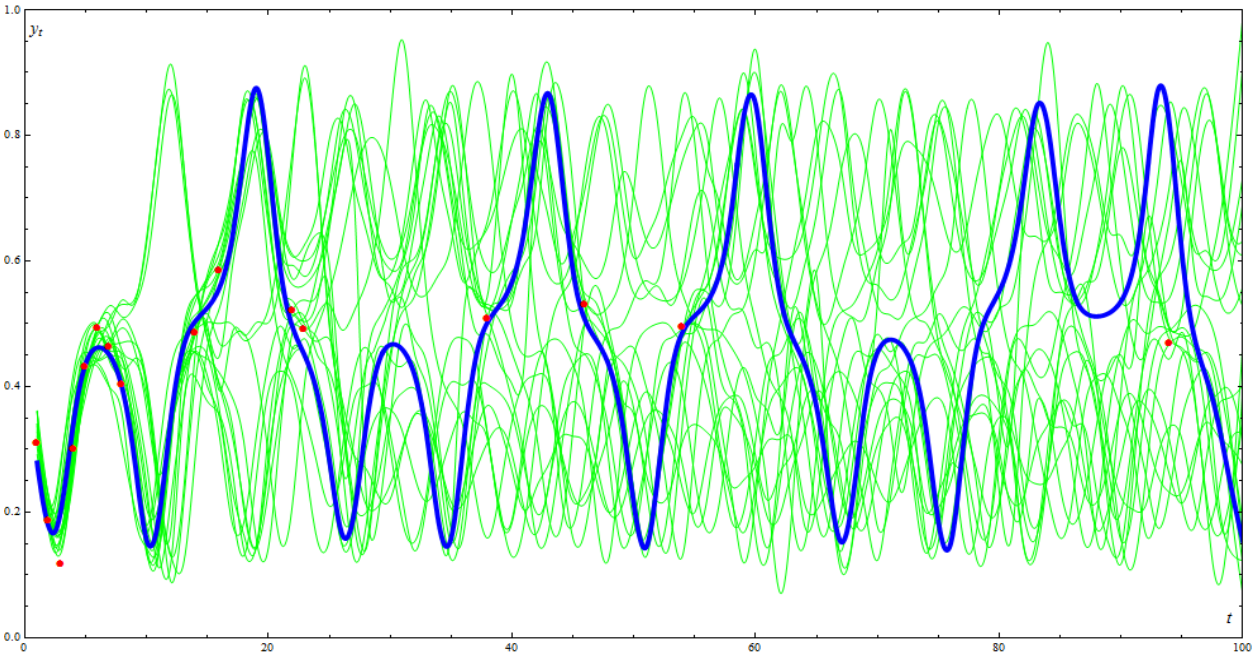


Fig. 3. A set of possible predicted trajectories for the Lorenz series. Blue solid line represents a true trajectory, while the thin green lines represent possible trajectories. Red points are the predictions made by the algorithm.

When $p > 1$, we utilize the concept of a 'possible predicted trajectory.' A possible predicted trajectory is defined as a sequence of possible predicted values at positions (not necessarily successive) preceding a prediction horizon and at the prediction horizon itself. In the framework of this approach, instead of producing sequence of sets of possible predicted values, the prediction method produces a set of possible trajectories that end in a prediction horizon. For some methods, the trajectory is built by adding a random component at each position, thus making the maximum number of trajectories a parameter of the prediction

algorithm. For other methods, several values are chosen at each point instead of a single unified prediction value, thereby making a 'fan' of trajectories. Obviously, the number of trajectories grows exponentially, and in order to avoid combinatorial explosion, quantitative constraints should be introduced (this could be visualized as placing the trajectories inside some kind of a 'pipe').

$\hat{\xi}_{t+h}^{(s)} = \left(\hat{y}_{t+1}^{(s)}, \ldots, \hat{y}_{t+h}^{(s)}\right)$ denotes an s th possible predicted trajectory; $\hat{\xi}_{t+h}^{(s)}(i) = \hat{y}_{t+i}^{(s)}$, its value at an ith position; S_{max}, the maximum number of trajectories. $\hat{\Xi}_{t+h}$ denotes a set of all trajectories that end at a position $t+h$; $\hat{\Xi}_{t+h}(i)$, a set of their values at an ith position. Quite naturally, $\hat{S}_{t+h} \equiv \hat{\Xi}_{t+h}(h)$. It should be noted that that since a trajectory may not contain all successive points, generally a set $\hat{\Xi}_{t+h}$ does not coincide with a set $\hat{S}_{t+h}^{(p)}$.

In the case of $p > 1$, some techniques to build up sets of possible predicted trajectories and then determine a unified predicted value are possible:

1. Randomly perturbed trajectories. This technique employs a conventional iterative strategy (predicted values are calculated step by step at all intermediate positions). A unified predicted value is calculated for each trajectory independently, with the employment of its own sets of possible predicted values. This value is equal to a perturbed average value over elements of the largest cluster (cf. the technique № 8 of the previous list): $\hat{y}_{t+h} = \sum_{\hat{y}_{t+h}^{(i)} \in Q_{j^*}} \hat{y}_{t+h}^{(i)} / \left|Q_{j^*}\right| + \varsigma(\Delta)$, where $\varsigma(\Delta)$ is a normally distributed random variable with zero mean and variance $\Delta \geq 0$. In general, this technique closely resembles the respective technique for $p = 1$, but it requires that the possible predicted trajectory be built S_{max} times. Figure 3 shows a number of possible predicted trajectories for the Lorenz series [*tp*] (a trajectory, perturbed).
2. Trajectories that rely on a choice of several clusters (the garden of forking trajectories; a fan of trajectories). We cluster the sets of possible predicted values at each intermediate position and considers the following centres of the clusters similarly to the second technique of the previous list:
$\hat{S}_{t+k} = \bigcup_j Q_j$, $Q_i \cap Q_j = \emptyset$. In this case, however, we choose several centres, thus making trajectories to branch. Usually, the first centre is that of the largest cluster with the rest being centres of the second largest cluster, the third, etc. and/or being centres of randomly chosen clusters. If all clusters are chosen randomly, we choose among clusters larger than a certain predefined threshold q_{min}. Regardless of the way the centres are chosen, each one becomes the next element of a separate trajectory. In order to cut the number of the trajectories employed, we introduce constraints based on their weights. A weight of a trajectory is defined as a product of weights of its elements, which may be calculated by any of the three ways discussed above. In other words, a 'fan' of

trajectories is transformed into a 'pipe.' For the cutting procedure, the current length (the number of elements) of trajectories should exceed a certain predefined threshold. As above, one may exclude trajectories with weights less than a certain value ω_0 or υ percent of 'lightest' trajectories, regardless of their actual weights [*tp*] (a trajectory, multiple clustering).

3. Trajectories that rely on reduced initial information. The technique combines, in a sense, iterated and direct strategies (Ben Taieb et al., 2012). It builds principal trajectory, starting at point t as well a series of additional trajectories starting at points $t-1$, $t-2$,…,$t-a$, where a is a parameter. For an additional trajectory, observed values after its starting point are neglected and replaced by predicted values, calculated by the prediction algorithm in a usual way [*tr*] (a trajectory, reduced information).

For all techniques, a unified predicted value is calculated as an average over the last values of the above trajectories: $\hat{y}_{t+h} = 1/(S_{max}\sum_{j=1}^{S_{max}\Sigma(h)}\hat{\xi}_{t+h}^{(j)})$. It is worth noting that the second technique appears to be the most efficient, at the sacrifice, quite naturally, of computational efficiency.

Interestingly, for the predictive-clustering algorithms discussed in this study, a set of non-predictable points appears to closely approximate a set of points for which the same algorithm without a procedure that identifies non-predictable points fails to make at least a satisfactory prediction. This observation provides the basis for a strategy to design various non-predictable-points identifying techniques.

2.5. Quality measures of unpredictable points identification algorithms

The prediction method under discussion utilizes various techniques for identifying unpredictable points. They can either use sets of possible predicted values for a position to be predicted $\hat{S}_{t+h}^{(p)}$ $(p=1)$ or a set of possible predicted trajectories that end at this point $\hat{\hat{\Xi}}_{t+h}$ $(p>1)$. For each prediction horizon h we may consider a set $NP(t,m,h) = \left\{t+i: t+m \le t+i \le t+h \;\&y_{t+i} \in Y_2 \& \zeta\left(\hat{S}_{t+i}^{(p)}\right)=0\right\}$; then its set of non-predictable intermediate position is defined as $NIP(t,h) = NP(t,1,h)$. A set of all non-predictable points of a test set is defined as

$$NP(h) = \bigcup_{t+h:y_{t+h}\in Y_2} NP(t,h,h) \equiv \left\{t+h:\ y_{t+h} \in Y_2 \& \zeta\left(\hat{S}_{t+h}^{(p)}\right)=0\right\} \quad .$$

This set consists of all positions of a test set such that the prediction algorithm fails to predict, when it predicts h steps ahead. It is important to note that a point may belong to the set of non-predictable points if:

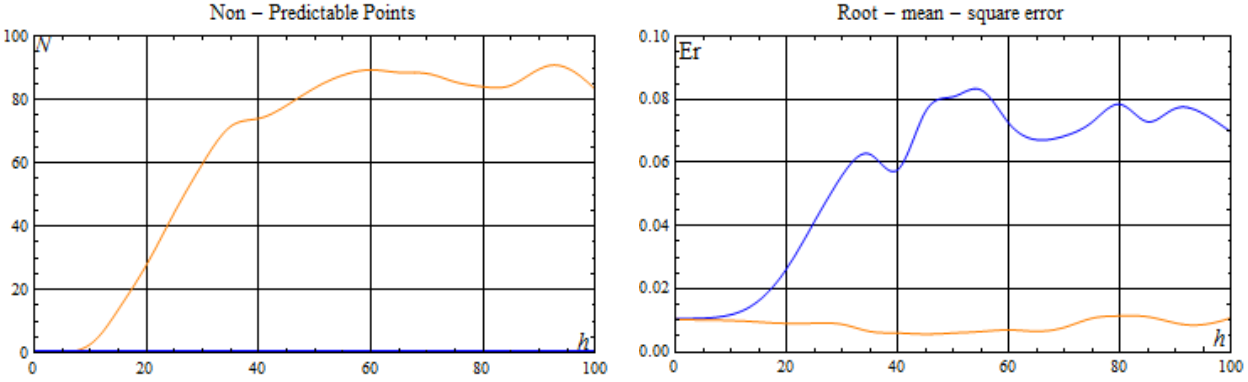
1. Its set of possible predicted values is empty;

2. This set comprises predicted values such that an adequate prediction value is impossible to obtain based on them.

The functional (3) in effect minimizes the size of this set.

Sets like this depend heavily on the algorithm used for a one-step-ahead prediction, f_h, a value of p, and a technique used to identify non-predictable points ζ. To design baselines for non-predictable-points identifying algorithms – and to compare them (for a givenf_h) – we consider two extreme cases. For the first one, we do not employ a non-predictable-points identifying algorithm altogether: $\zeta(S) \equiv 1, NP(h) \equiv \emptyset, \ \forall h \in N$ by default. It means the prediction algorithm uses the closest motif regardless of how remote it is [*fp*] (forced prediction). A kind of light version of this baseline does not use motifs that are farther than a certain threshold [*ab*]. This version (the first extreme case) serves as a baseline for other.

For the second extreme case, we assume that the prediction algorithm possesses *a priori* information about observed values at intermediate positions. More precisely, if the difference between its unified predicted value and the true value at the same position is more than ε (a small parameter then the algorithm marks this point as unpredictable. Note that the algorithm *does not* replace its predictions by the true values – instead, it simply does not take clearly erroneous predictions into account during next prediction operations. Such points comprise, for a given prediction algorithm, its set of the ground-truth non-predictable points. Namely, for each point $t + h$ that we are trying to predict, we can build up the set $GTNP(t, m, h) = \{t + i: t + m \leq t + i \leq t + h \ \& y_{t+i} \in Y_2 \& \rho(y_{t+i}, \hat{y}_{t+i}) \geq \varepsilon\}$, where $\hat{y}_{t+i} = g\left(\hat{S}_{t+i}^{(p)}\right)$ is, as usual, a unified predicted value calculated for the set of possible predicted values $\hat{S}_{t+i}^{(p)}$. Hereinafter, the distance is Euclidean. Then, similarly to the above, its set of the ground-truth non-predictable intermediate points is defined as $GTNIP(t, h) = GTNP(t, 1, h) = \{t + i: t + 1 \leq t + i \leq t + h \ \& y_{t+i} \in Y_2 \& \rho(y_{t+i}, \hat{y}_{t+i}) \geq \varepsilon\}$ and the set of the ground-truth non-predictable points for a test set, as $GTNP(h) = \bigcup_{t+h: y_{t+h} \in Y_2} GTNP(t, h, h) \equiv \{t + h: \ y_{t+h} \in Y_2 \& \rho(y_{t+i}, \hat{y}_{t+i}) \geq \varepsilon\}$.



a b

Fig. 4. The number of non-predictable points (a) and RMSE (b) vs. a prediction horizon for the two extreme cases. The blue curves correspond to the first extreme case (intermediate non-predictable points are not identified at all); orange, to the second one (intermediate non-predictable points are identified with *a priori* information).

Attempting to predict at $t + h$ and excluding all intermediate points that fall into the set $GTNIP(t, h)$, leads to the second extreme case – a prediction method that uses an algorithm identifying unpredictable points using *a priori* information (in our internal team slang, this algorithm is named 'the daemon' – Socrates is known to have had his own daemon, who gave him advice in his most painful situations Wide-raging simulation reveals that the results of such an algorithm are nearly independent of a threshold value ε. Quite naturally, there is no such things for real-world algorithms: they do not possess information about the true values for intermediate positions y_{t+i} , and those pieces of information they have at their disposal are either sets of possible predicted values $\hat{S}_{t+i}^{(p)}$ or sets of possible predicted trajectories $\hat{\Xi}_{t+i}$. Nonetheless, this algorithm is extremely useful in determining the lower error boundary reachable by any other prediction algorithm. Algorithms identifying unpredictable points should be developed in a way that would make their results as close as possible to this boundary. Graphs in Fig. 4 exemplify dependences of the number of non-predictable points (Fig. 4 *a*) and the average error on predictable points (Fig. 4 *b*) on the prediction horizon h for the two extreme cases. The blue curves correspond to the first extreme case (intermediate non-predictable points are not identified at all), and orange, to the second (intermediate non-predictable points are identified with *a priori* information). It can be seen tha

t

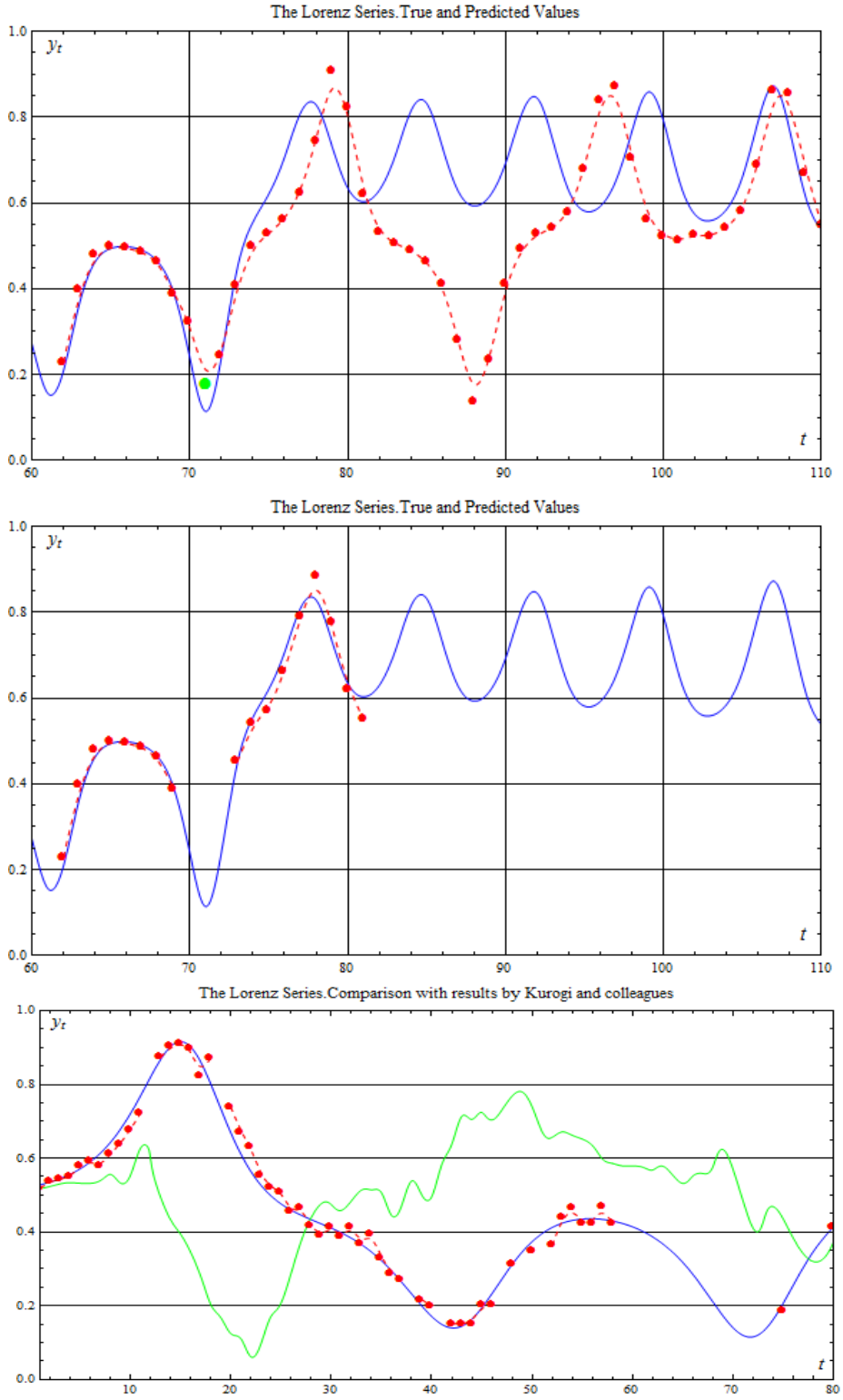


Fig. 5. True (blue solid) and intermediate predicted (red dashed) values for the Lorenz series for the first (a) and second (b) extreme cases. A green disk indicates a point that the algorithm that uses *a priori* information identifies as non-predictable.
(c) – the Lorenz series: true trajectory (blue curve) and predicted points (red dashed curve with disks). A green line presents results by Kurogi and colleagues (2014).

in the first case the number of unpredictable points is equal to zero and the prediction error grows exponentially with h. The second extreme algorithm

demonstrates the opposite: the number of non-predictable points grows exponentially with h, while the prediction error function remains mostly constant, bounded, and rather small. It is obvious that the first algorithm minimizes the functional (3), neglecting the functional (4); the second one minimizes the functional (4), neglecting the functional (3). Figures 5a and 5b display the true values (blue solid) and intermediate predicted values (red dashed) for the Lorenz series for the first (5 a) and second (5 b) extreme cases. From the latter subfigure we notice that the second algorithm does not make a prediction for all points, while making rather accurate predictions where it 'decides' to predict. On the other hand, we notice that in the first case the predicted trajectory diverges from the true one just after the point (a green disk in Fig. 5 a) that is identified as non-predictable by the second algorithm – the first algorithm must predict at this point (as with any point), and this immediately ruins the MSA prediction process, causing the predicted trajectory to diverge from the true one.

Any other prediction algorithm is a trade-off between these two extreme algorithms. In the context of multi-step ahead prediction, the second case is essentially more interesting, since it allows one to predict a fairly large number of steps ahead. Consequently, all algorithms discussed below attempt to follow suit with this very algorithm (in the team's internal slang, we referred to them as 'approximating the daemon' :).)

Figure 5 c displays the true Lorenz series (blue), predicted values by Prof. Kurogi and colleagues (green), and the values predicted by the algorithm discussed in the present paper (red dashed line with disks). We should stress that at $t = 12$ the trajectory predicted by Prof. Kurogi and colleagues diverges from the true trajectory, while our algorithm identifies several non-predictable points. This makes it possible to predict essentially farther and find 'hillocks' of predictable points farther down from the last known point.

Common sense (and large-scale simulation) suggests that such an algorithm will work efficiently if and only if a set of identified non-predictable points $NP(h)$ is sufficiently close to the set of the ground-truth non-predictable points $GTNP(h)$. Symmetric difference of the two sets constitutes an auxiliary quality measure for a technique identifying non-predictable points:

$$I_3 = |GTNP(h)\ \Delta\ NP(h)| \tag{7}$$

Two opposite situations can be observed:

1. The difference $|GTNP(h)\ /\ NP(h)|$ is large, whereas the difference $|NP(h)\ /\ GTNP(h)|$ is small;
2. The difference $|NP(h)\ /\ GTNP(h)|$ is large, whereas the difference $|GTNP(h)\ /\ NP(h)|$ is small.

The first case seems to correlate with an overly optimistic algorithm while the second correlates with an overly conservative one.

2.6. Algorithms to identify non-predictable points (function ζ (1))

These algorithms show a basic dichotomy between those that employ a set of possible predicted values $\hat{S}_{t+h}$ ($p = 1$) and those that employ a set of possible predicted trajectories $\hat{\Xi}_{t+h}$ ($p > 1$). In order to gain a better insight into the identification algorithms, we should consider a 'paragon' of a predictable point. We believe that its set of possible predicted values should consist of a single compact cluster, which includes an overwhelming majority of elements, and a number of small clusters and/or individual elements. On the contrary, non-predictable points are associated either with a continuum of possible predictable values not amenable to clustering or, vice versa, with several approximately equinumerous compact clusters. In terms of probability distributions, a predictable point correlates with a unimodal symmetric distribution with relatively small variance, while the unpredictable one correlates either with a nearly uniform distribution or with a distribution with several pronounced modes.

In order to characterize unpredictable points, we employ the following quantities:

- Multimodality – a percent of possible predicted values remote from the main mode;
- Difference between α and $1 - \alpha$ percentiles;
- Second, third, and fourth central moments, as well as the kurtosis of the distribution and its entropy.

We performed large-scale simulation on the validation set Y_3 by calculating sets of unpredictable points $NP(h)$ and the ground-truth unpredictable points $GTNP(h)$ for each feature separately. A comparison of these sets makes it possible to estimate the best threshold values for each feature. It also reveals that each feature (except entropy) by itself does not lead to larger values of objective (7) [*en*] (entropy).

In terms of clustering of a set of possible predicted values, we employ the following quantities:

- Number of clusters;
- Percent of possible predicted values that fall into the largest cluster;
- Difference between the percentage of values belonging to the largest and smallest clusters.

Similarly to the previous case, it is revealed that these quantities do not 'work' separately either. Consequently, the main course would be to use them together with quantities based on the distribution of possible predicted values.

We employ the following methods to separate the features mentioned above into regions corresponding to predictable and unpredictable points:

1. Logistic regression [*lr*];
2. Support vector machine [*sv*];
3. Decision tree c 4.5 [*dt*];
4. K-nearest neighbours clustering [*kn*];

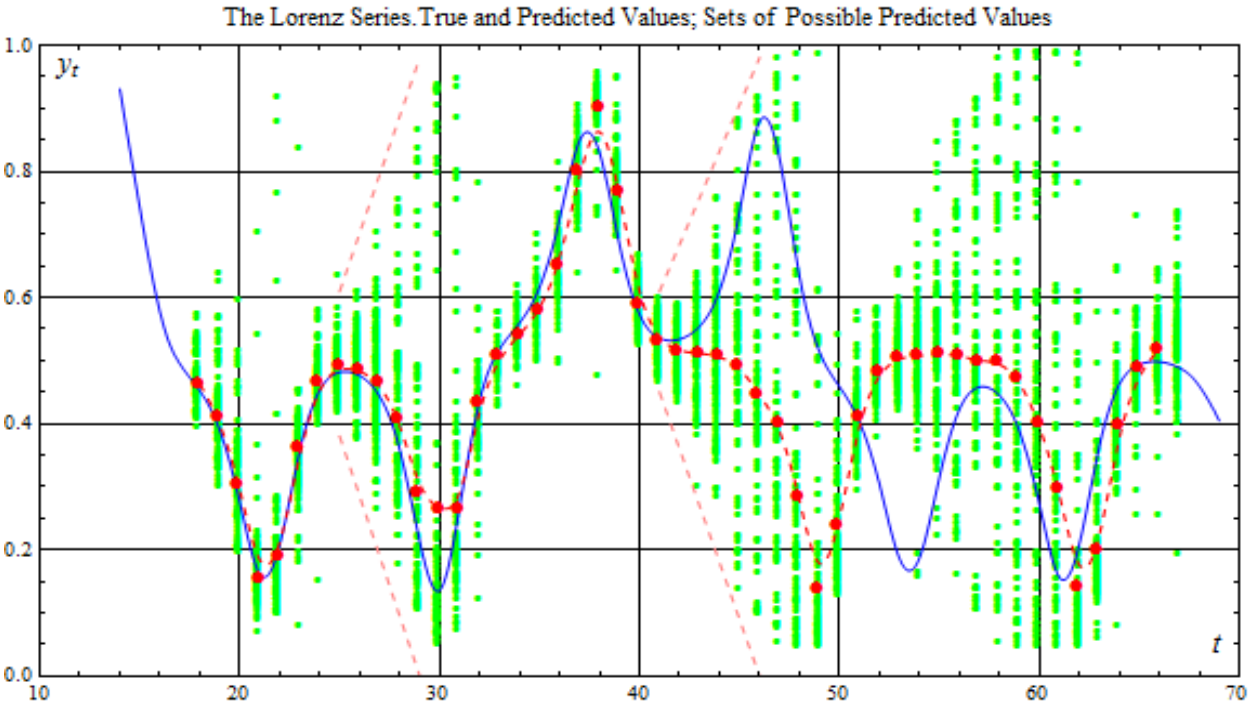


Fig. 6. Divergence of true (solid blue lines) and predicted (dashed red lines) trajectories correlates with a monotonic spread growth (indicated by pink dashed lines) of sets of possible predicted values (Each set is green disks with the same x-coordinate).

5. Multilayer perceptron with a single layer and 8 neurons in it [*mp*].

Mathematically, in order to apply this approach, a third validation set Y_3, independent of the training set Y_1 and the testing set Y_2 should be introduced:: $Y_1 \cap Y_3 = \emptyset$, $Y_2 \cap Y_3 = \emptyset$. When training these models, we label the validation set using the set of ground-truth unpredictable points. Since the number of unpredictable points is larger than the number of predictable ones for larger prediction horizons, a sample is balanced by SMOTE methods (synthetic minority over-sampling technique) (Chawla et al., 2002). This method generates new elements in the neighbourhood of the smaller cluster.

In addition to the above classifying methods, we employ their ensembles:

1. AdaBoost (Adaptive Boosting) – an algorithm implying that each subsequent classifier is trained on the elements incorrectly classified by the current classifier [*al* or *as*] (adaptive boosting for logistic regression or support vector machine, respectively);
2. Stacking – an algorithm implying that several classifiers are applied in order to produce a feature space for a combined algorithm (logistic regression) [*sl*] (stacking, logistic regression);

3. Voting – an algorithm that assigns a label to each element by the classifiers' votes [*vl*] (voting, logistic regression).

For the second approach, when analysing a set of possible predicted trajectories $\hat{\Xi}_{t+h}$, trajectories converging at a certain position indicates predictability. Plots in Fig. 5 exemplify the set of possible predicted trajectories (thin green lines); a solid blue line shows the true trajectory for comparison; red disks denote predictions made by the algorithm (all other points are identified as unpredictable). From this figure we notice that predictable points are separated by sections of non-predictable points. On the other hand, predictable points are associated with regions where possible predicted trajectories converge. It is worth noting that predicted values are in fairly good agreement with the true ones at the points where algorithms identified as predictable: the results shown in the figure correspond to a conservative variant of the algorithm; the algorithm seeks to minimize average error over the predictable points (the objective I_2 (4)), and to a lesser degree the number of non-predictable points (objective I_1 (3)). Figure 6 displays a true trajectory, a predicted trajectory, and sets of possible predicted values at intermediate positions. The figure illustrates the fact that a large spread (larger than some average value) of a set of possible predicted points suggests the existence of a divergence between true and predicted trajectories. On the other hand, several spaced out clusters may be observed at points like this instead of a single compact cluster.

These observations were utilized to develop several techniques to identify non-predictable points:

1. We apply the prediction algorithm to a validation set Y_3 in order to calculate an average spread over sets of possible predicted values $\kappa_{avg} = \sum_{t+h \in Y_3} \left| max\, \hat{\Xi}_{t+h}(h) - min\, \hat{\Xi}_{t+h}(h) \right| / |Y_3|$. Then the algorithm recognizes a point as non-predictable, if its spread exceeds this average value times some factor $\vartheta \kappa_{avg}$ [*ls*] (large spread).
2. Unlike the previous case, one considers not the spread itself $\kappa(t+i) = \left| max\, \hat{\Xi}_{t+h}(h) - min\, \hat{\Xi}_{t+h}(h) \right|$, but its variation over some successive positions. If it increases monotonically, then the first point is classified as non-predictable. Simulation suggests that the best option here is to consider three successive points. It may be noted (refer to Fig. 6) that the divergence of true and predicted trajectories correlates with a monotonic spread growth (indicated by pink dashed lines) [*rg*] (rapid growth). One may also note growth of the number of possible predicted value clusters obtained with clustering algorithms (DBSCAN [*rd*] or the Wishart clustering [*rw*]).
3. For the trajectories relying on reduced initial information, we may compare the last value of a trajectory starting at a position t, $\hat{y}_{t+h} = \hat{\xi}^{(0)}_{t+h}(h)$, with the weighted average of the last values for trajectories starting at positions $t-1$, $t-$

2,…,$t-a$: $\tilde{y}_{t+h}=\sum_{j=1}^{a}\omega_j\hat{\xi}_{t+h+j}^{(j)}(h)$. If $|\hat{y}_{t+h}-\tilde{y}_{t+h}|$ is greater than small ε, then the point is recognized as non-predictable. The weights are $\omega_j=1/2^j$; hence trajectories starting from positions closer to t possess larger weights [*ca*] (compare with average).

4. Another variant, designed for the trajectories that rely on reduced initial information implies comparing an averaged modulus of the difference between the last points of a trajectory starting at a position t and a trajectory starting at other positions $\frac{1}{a}\sum_{j=1}^{a}\left|\hat{\xi}_{t+h+j}^{(0)}(h)-\hat{\xi}_{t+h+j}^{(j)}(h)\right|$ with ε [*ad*] (averaged difference).

5. Similarly to the previous case, we calculate a weighted average of the differences $\sum_{j=1}^{a}\omega_j\left|\hat{\xi}_{t+h+j}^{(0)}(h)-\hat{\xi}_{t+h+j}^{(j)}(h)\right|$ with weights equal to $\omega_j=2(a-j+1)/(a(a+1))$ [*wa*] (weighted average).

6. It is also possible to apply all techniques from the previous list to a set of final points of the trajectories $\hat{\Xi}_{t+h}(h)$ [*wd*] (weighted, difference).

For comparison, tables of the next section include results for the extreme cases. Namely, a dash symbolizes that non-predictable points are not identified altogether, predicted values are forcibly calculated at all intermediate points; *ab* (absent) implies that a set of non-predictable points consists only of the points that the algorithm failed to find motifs close enough ($\hat{S}_{t+h}^{(p)}=\emptyset$), sets of possible predicted values are not analysed (the first extreme case); *id* (ideal), sets of possible predicted values are constructed using *a priori* information (the second extreme case).

V. NUMERICAL RESULTS

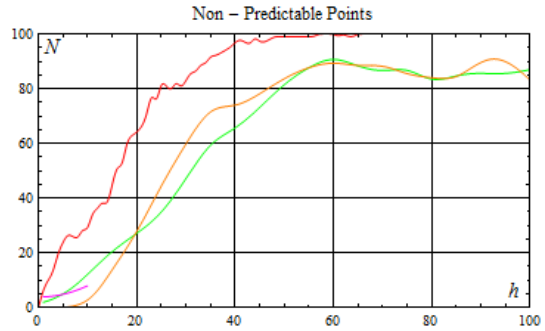
The approach discussed in the previous section is applied to both benchmark (Lorenz) and the real (hourly electricity load) time series. The clustering algorithm is applied to generate samples using all possible patterns of four elements ($L=4$) with the maximum distance between neighbouring positions in the pattern equal to K_{max} =10. Thus, the total number of patterns amounts to $|\aleph(L,K_{max})|$=1000. If the case when algorithm does not use all possible patterns, the percentage of used patterns is equal to 4 %, $|\beta\aleph(L,K_{max})|$=40.

We utilize two types of plots for each series. The first type (see Appendix A) shows a predicted trajectory up to a prediction horizon (a dashed red line with disks of predicted values); a true trajectory is shown for comparison (a solid blue line). For the predicted trajectory, the first presented value corresponds to a one-step ahead prediction, the second – to a two-step ahead prediction, and so on Plots of the second type display an averaged (over a test set) error measure (the number of non-predictable points, the root-mean-square error [RMSE], the mean

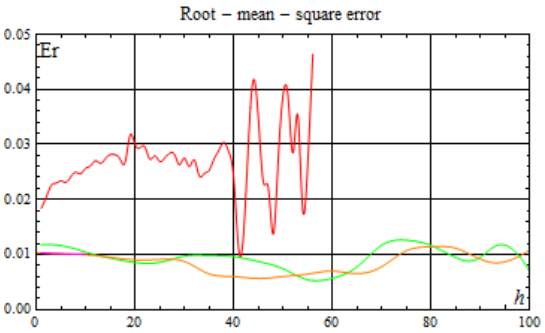
absolute percentage error [MAPE]) against a prediction horizon. Each such figure also contains, for comparison, the respective plots for two extreme cases (refer to Section *Quality measures for algorithms to identify non-predictable points*) for the algorithm that does not identify non-predictable points (blue) and the one that identifies them using *a priori* information (orange line normally, or red in case of 4% of patterns being used). The two plots make a 'fork' that usually bounds from below and above plots of all other discussed algorithms.

These results are also presented in tabular form for several typical *prediction horizons* h. We used two types of tables. The first four columns present characteristics of the method used and are the same for both types; namely, the first column indicates a percentage of employed patterns β and the way the algorithm obtains motifs (refer to the Section *A training set*). The second column indicates whether the algorithm builds up a set of possible predicted values or predicted trajectories and the way it calculates a unified predicted value (refer to the Sections *Sets of possible predicted values* and *A unified predicted value*). The third column provides information on the way the algorithm identifies non-predictable points (refer to the Section *Algorithms to identify non-predictable points*). For information on symbolic labels used in these columns, please refer to the previous section (mentioned in the square brackets after the corresponding technique) or to tables in Appendix B. The fourth column indicates the figure that displays results obtained using the respective variant of the algorithm and the colour used to plot the respective curves. Finally, all other columns present error measures for prediction horizons $h = 1$, 10, 50 and 100 (the number of non-predictable points, the mean absolute percentage error, and the root-mean-square error). It is worth noting that the first ten lines correspond to the extreme cases (refer to the Section *Quality measures for algorithms to identify non-predictable points*).

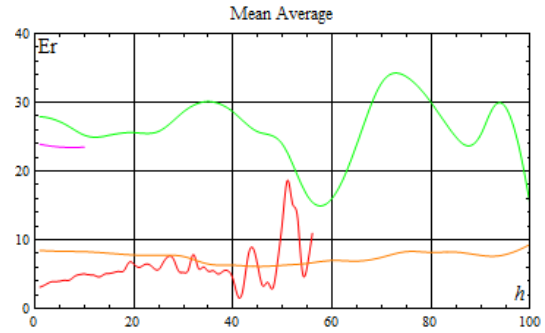
Tables of the second type present information on the sets of non-predictable points for various variants of the method considered. The first columns are the same as the respective columns of the tables of the first type. Other columns indicate, ($h = 1$, 10, 50 and 100) $Re\,c\,all = TP/(TP + FN)$, $Pr\,e\,cision = TP/(TP + FP)$ and F-measure $F = 2\,Pr\,e\,cision * Re\,c\,all/(Precision + Recall)$ for the same prediction horizons.



a



b



c

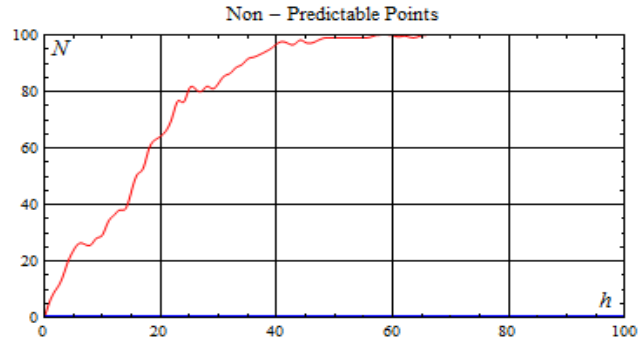
Fig. 7. The second extreme case. Results for various versions of algorithms that identify non-predictable points with *a priori* information. The number of non-predictable points (a), RMSE (b), and MAPE (c) vs. a prediction horizon Green and red curves correspond to the method that employ the Wishart clustering (100% and 4%, respectively); orange and magenta, to those that does not employ clustering algorithms (100% and 4%, respectively).

The Lorenz system (Atlee Jackson, 1985; Malinetskii & Potapov, 2000) with standard 'chaotic' parameters $\sigma = 10, b = 8/3, r = 28$, integrated using the Runge-Kutta fourth-order method (an integration step is equal to $\Delta t = 0.1$), yields a time series hereinafter referred to as the Lorenz series. The series in question is a typical chaotic series; it is used as a conventional benchmark to test forecasting procedures for chaotic time series.

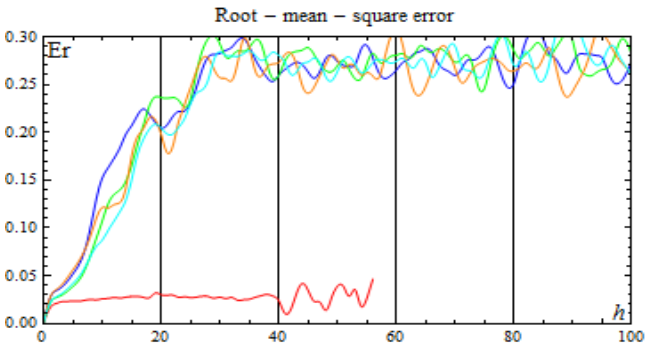
To distinguish chaotic and noisy time series, Rosso et al. (Rosso et al., 2007) proposed checking the position of a pair of quantities – entropy and complexity – on the appropriate plane. They calculated these quantities for the Lorenz series and got 0.68 and 0.45, thus making it possible to classify this series as chaotic. On the other hand, the highest Lyapunov exponent λ for this series, calculated using the Eckman algorithm (Eckmann et al., 1986), is equal to 0.92, which is in a good agreement with results by Malinetskii and Potapov (Malinetskii & Potapov, 2000) (see p. 217). A strictly positive value of this quantity also speaks in favour of the series inherently chaotic nature.

Numerical error for the fourth-order Runge-Kutta method amounts to $\varepsilon(0) = \Delta t^4 = 10^{-4}$. If the maximum prediction error is chosen as $10^{-1} = \varepsilon_{max}$, then estimating a horizon of predictability would yield $T \sim \frac{1}{\lambda} ln\left(\varepsilon_{max}/\varepsilon(0)\right) = 7.5$, or (in integration steps) 75 time series observations.

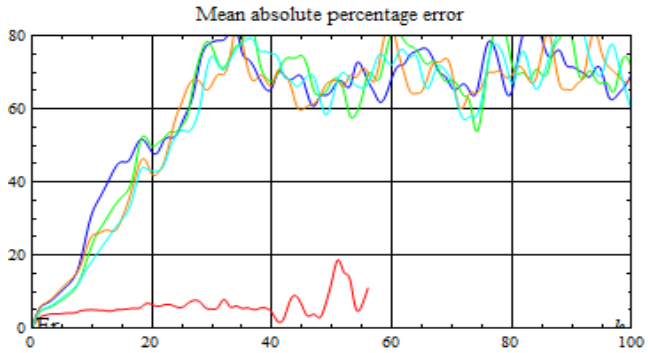
For the Lorenz series, the first 3000 observations are discarded in order to ensure that trajectory moves in the neighbourhood of the respective strange attractor. The test set for the series consists of 1000 observations; the training set



a



b



c

Fig. 8. The number of non-predictable points (a), RMSE (b), and MAPE (c) vs. a prediction horizon (4% of patterns are used). Values are calculated (without identifying non-predictable points) as an average of possible predicted values (blue) or a weighted average with weights inversely proportional to the distance to a cluster centre; weights are proportional to the predictive length and combined weights (green, orange, and cyan lines respectively). Red curves correspond to the algorithm that identifies non-predictable points using *a priori* information.

consists of 10000. Table I and II are the first and second type tables for the Lorenz series.

Large-scale simulation reveals that for the first extreme case (a point is recognized as non-predictable only if there are no motifs close enough to make prediction, $\hat{S}_{t+h}^{(p)} = \emptyset$) both the number of non-predictable points and average errors for predictable ones grow exponentially with a prediction horizon. Figure 7 demonstrates the number of non-predictable points (7a), RMSE (7b), and MAPE (7c) as a function of a prediction horizon in order to compare results obtained for the methods utilizing 100% and 4% of patterns. The green and red curves correspond to the method that employ the Wishart clustering (100% and 4%, respectively); orange and magenta correspond to those that do not employ any clustering algorithms (100% and 4%, respectively). Such results allow us to conclude that the prediction quality increases with the number of used patterns: the percent of non-predictable points is nearly zero for a larger prediction horizon, whereas the error measures for predictable points are comparable.

Figure 8 presents the results for multi-step ahead prediction (the number of non-predictable points (8a), RMSE (8b), and MAPE (8c)); all methods does not identify non-predictable points altogether (4% of patterns is used). A unified predicted value is determined as an average (blue lines) [*av*] or a weighted average with weights inversely proportional to a distance to a cluster centre [*wd*], weights proportional to predictive length [*wl*], and combined weights [*w*] (green, orange, and cyan lines, respectively). For comparison, red lines represent the second extreme case (non-predictable points are identified with the employment of *a priori* information). It may be concluded that the way the algorithm calculates a unified prediction value has little effect on the rate of this growth. Consequently, of crucial importance is the algorithm used to identify non-predictable points. In what follows, we discuss results for various identification techniques.

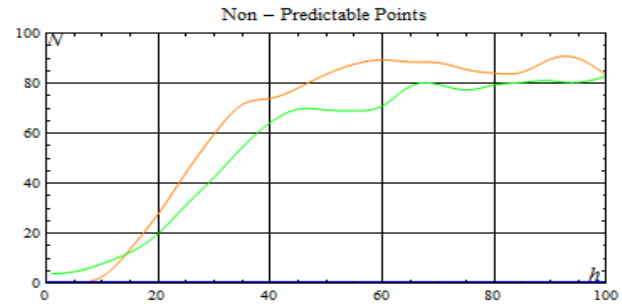
The first identification technique employs entropy of possible predicted values distribution. Simulation reveals that the optimal threshold between the entropies of predictable and non-predictable points is equal to $\Delta = 0.1$. Figure A1 (refer to Appendix A) portrays a typical empirical distribution for entropy for the ground-truth non-predictable points $GTNP(50)$ and its complement to a testing set $Y_2 \backslash GTNP(50)$. It is quite obvious that entropy values correlating with the ground-truth non-predictable points and those correlating with predictable ones differ significantly. Unfortunately, it appears that it is impossible to design an efficient test based solely on entropy values.

A number of identification techniques apply various machine learning algorithms to feature spaces build for sets of possible predicted values. Figure A9 summarizes the following results:

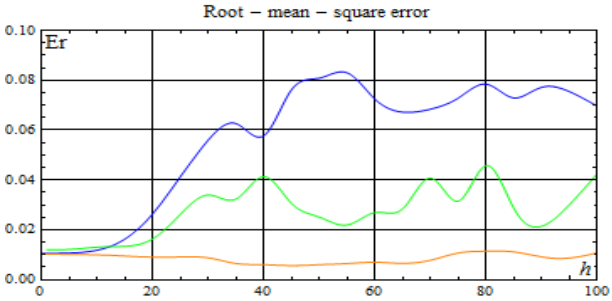
- Red curves correspond to the unpredictable points identified using *a priori* information;
- Blue points are not identified at all;
- The green ones are identified using a logistic regression classifier [*lr*];
- Orange ones – using an SVM classifier [*sv*];
- Cyan – decision tree [*dt*];
- Black – k-nearest neighbour [*kn*];
- Grey – multi-layer perceptron classifier[*mp*];
- Magenta – Adaboost ensemble of LogReg classifiers [*al*];
- Yellow – Adaboost ensemble of SVM classifiers [*as*];
- Pink – voting of LogReg classifiers [*vl*];
- Orange –stacking of LogReg classifiers [*sl*].

A multi-layer perceptron contains 8 hidden layers and utilizes sigmoidal activation function; support vector machine employs polynomial kernel functions; C4.5 employs entropy criterion without any restriction on tree depth; k-nearest neighbour value is equal to 3.

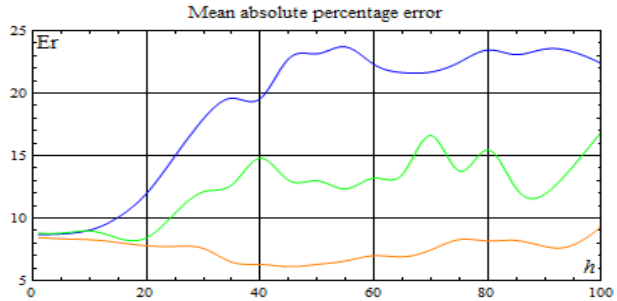
Among all considered machine learning methods, the ones based on logistic regression [*lr*] (green curves) and Adaboost ensemble of them [*al*] (magenta curves) stand out. They have a nearly constant dependence of both error measures on non-predictable points on the horizon, and their error measures for predictable points are comparable with those for the algorithm that used *a priori* information to identify non-predictable points (the second extreme case, red curves). Unfortunately, it also demonstrates an exponential growth of the number of non-predictable points with a rate larger than that of the second extreme case. The method that employs a multi-layer perceptron [*mp*] (gray curve), vice versa, demonstrates a moderate rate of growth of the number of non-predictable points with rapid error growth. Figures A2-A5 show predicted trajectories compared with a true one for these non-predictable-points identifying techniques. All methods that use an ensemble of classifiers to distinguish predictable and non-predictable points show comparable efficiency. Figure 10 displays results for the algorithms that use growth of the standard deviation value [*rg*] (green); of the number of DBSCAN clusters [*sd*] (orange); of the number of Wishart clusters [*sw*] (cyan) over three consecutive points to identify non-predictable points. Unfortunately, these approaches do not demonstrate desirable behavior. Figure A6 demonstrates true and predicted trajectories for this case.



a

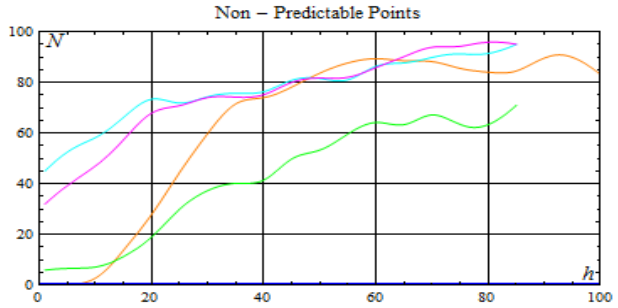


b

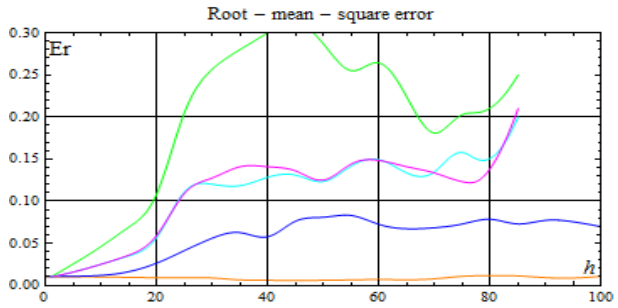


c

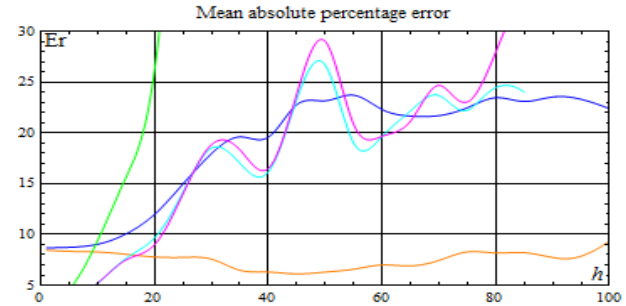
Fig. 9. The number of non-predictable points (a), RMSE (b), and MAPE (c) vs. a prediction horizon for trajectories with random perturbation (100 % of patterns are used; *no clustering technique is applied*). The algorithm identifies non-predictable points when possible predicted trajectories diverge. Orange curves correspond to the algorithm that identifies non-predictable points using *a priori* information; blue, to the one that does not identify such points.



a



b



c

Fig. 10. The number of non-predictable points (a), RMSE (b), and MAPE (c) vs. a prediction horizon for trajectories with random perturbation (100 % of patterns are used; *the Wishart clustering*). The algorithm identifies non-predictable points if possible predicted trajectories diverge. Orange curves correspond to the algorithm that identifies non-predictable points using *a priori* information; blue, to the one that does not identify such points.

In order to implement an approach using branching trajectories (the 'pipe' concept), we cluster points using DBSCAN algorithm and then select those clusters that appear to be larger than q_{min}= 5% of the total number of possible predicted values. Of these clusters, we chose two, using the following strategies:

1. Choosing the largest and the second largest clusters;
2. Choosing two randomly chosen clusters;
3. Choosing the largest and a number of randomly chosen clusters.

The maximum allowed number of trajectories (pipe diameter) is equal to 20. It was ascertained that exponential error growth of the number of predictable points is peculiar to this technique, though it is not as steep as in the previous case.

Simulation reveals that the methods based upon sets of predictable trajectories compare favourably to those based upon sets of predictable points. Figure 9 exhibits results for trajectories with random perturbation [*tp*]. The presented results correspond to a variance of the perturbation equal to 0.05. In order to identify non-predictable points, the algorithm clusters a set of final points of the predicted trajectories. If a size of the largest cluster is less than 4% of the total number of the final points, then the algorithm recognizes this point as non-predictable. To cluster points, we use DBSCAN with the following parameters: eps = 0.01, min_samples = 5. (Orange curve corresponds to the method that does not identify non-predictable points; it differs significantly from the respective curves of the previous plots (the red ones), since the algorithm builds up sets of possible predictable values in different manner). Figure A8 presents typical true and predicted trajectories for this case.

An inspection of the figure indicates that the algorithm predicts values at positions up to $t + 100$, that significantly exceeds 75 – an estimated horizon of predictability for the Lorenz series (see above). This justifies the title of the present article. Figure A8 (refer to Appendix A) shows typical curves of error measures against the number of possible trajectories (100 % of patterns are used; no clustering technique is applied). The algorithm identifies non-predictable points when possible predicted trajectories diverge. Orange curves correspond to the algorithm that identifies non-predictable points with *a priori* information; blue, to the one that does not identify such points ($\Delta = 0.033$). Results for the methods that use reduced initial information are presented in Fig. 10.

Table I. The Lorenz series. Non-predictable points and error measures

β %	PVs	NP	Figs	$h = 1$			$h = 10$			$h = 50$			$h = 100$		
motif	UPV	T		NP %	MA %	RMS	NP %	MA %	RMS	NP %	MA %	RMS	NP %	MA %	RMS
100 cl	*S av*	*ab*	-	0	8.7	0.011	0	8.8	0.013	0	22.5	0.08	0	22.4	0.07
100 cl	*S av*	*id*	7, green	2	8.9	0.011	12	8.3	0.01	82	7.2	0.007	82	9.5	0.01
100 p	*T av*	*ab*	4, blue	0	2.0	0.04	0	10	0.13	0	22	0.29	0	-	-
100 p	*T av*	*id*	7, orange	5	2.0	0.03	27	2	0.03	99	3	0.03	-	-	-
4 cl	*S av*	*ab*	-	0	5.7	0.03	0	32	0.154	0	63	0.281	0	69.7	0.28
4 cl	*S av*	*id*	7, red	7	3.1	0.018	28	5	0.025	99	7	0.029	100	-	-
4 p	*T av*	*id*	7, magenta	92	13.7	0.013	89	51.5	0.02	79	75.4	0.06	93	121	0.1
4 cl	*S wd*	*ab*	8, green	0	4.5	0.023	0	23.1	0.106	0	64.2	0.261	0	70.7	0.28
4 cl	*S wl*	*ab*	8, orange	0	5.7	0.03	0	25.9	0.124	0	67.2	0.285	0	68.6	0.26
4 cl	*S w*	*ab*	8, cyan	0	4.5	0.023	0	18.8	0.085	0	56.3	0.25	0	59.1	0.26
4 cl	*S av*	*en*	-	28	3.0	0.016	21	18.4	0.089	36	51.0	0.269	38	63.9	0.29
4 cl	*S w*	*lr*	9, green	18	2.2	0.014	57	6.3	0.038	100	-	-	100	-	-
4 cl	*S w*	*sv*	9, orange	24	2.1	0.014	46	6.0	0.039	97	52.1	0.326	100	-	-
4 cl	*S w*	*dt*	9, cyan	43	2.6	0.015	44	7.2	0.062	86	55.9	0.258	97	48.1	0.24
4 cl	*S w*	*kn*	9, black	44	4.0	0.019	42	6.0	0.042	84	73.7	0.319	**94**	**26.5**	**0.14**
4 cl	*S w*	*m*	9, grey	28	2.2	0.014	29	8.0	0.048	68	46.6	0.195	87	68.0	0.25
4 cl	*S w*	*al*	9, magenta	28	2.0	0.013	79	4.7	0.026	100	-	-	100	-	-

4 cl	*S w*	*as*	9, yellow	22	2.3	0.014	43	6.0	0.04	90	33.3	0.243	100	-	-
4 cl	*S w*	*vl*	9, pink	26	2.2	0.015	27	12.5	0.063	48	48.6	0.249	60	53.5	0.22
4 cl	*S w*	*sl*	9, orange	26	2.1	0.014	48	11.7	0.061	98	109.	0.371	100	-	-
4 cl	*S w*	*rg*	10, green	0	4.5	0.023	43	6.9	0.057	84	56.7	0.296	97	259.	0.45
4 cl	*S w*	*rd*	10, orange	0	4.5	0.023	20	23.0	0.105	31	58.6	0.260	59	77.7	0.27
4 cl	*S w*	*rw*	10, cyan	0	4.5	0.023	5	19.1	0.086	11	73.7	0.289	15	62.1	0.28
100 p	*T tp*	*rd*	11, green	0	8.3	0.012	0	9.0	0.028	84	13.0	0.042	**83**	**17.0**	**0.04**

The first column indicates a percentage of used patterns β and the way the algorithm obtains motifs (refer to the Section *A training set*). The second column indicates whether the algorithm builds up a set of possible predicted values or predicted trajectories and how it calculates a unified predicted value (refer to the Sections *Sets of possible predicted values* and *A unified predicted value*). The third column provides information on the way the algorithm identifies non-predictable points (refer to the Section *Algorithms to identify non-predictable points*. For information on symbolic labels used in these columns, please, refer to the previous section (in square brackets after the corresponding technique) or to the tables in Appendix B. The fourth column indicates the figure that displays results obtained using the respective variant of the algorithm and the colour used to plot the respective curves. Finally, all other columns present error measures for prediction horizons $h = 1$, 10, 50 and 100 (the number of non-predictable points, the mean absolute percentage error, and the root-mean-square error). Bold green colour indicates the best results after a horizon of predictability obtained with sets and trajectories of possible predicted values.

Table II. The Lorenz series. Sets of non-predictable points

β % motif	PVs UPV	NPT	Figs	$h = 1$			$h = 10$			$h = 50$			$h = 100$		
				Rcl	Prc	F	Rcl	Prc	F	Rcl	Prc	F	Rcl	Prc	F
100 cl	*S av*	*ab*	-	0.8	0.74	0.77	0.69	0.46	0.55	-	-	-	-	-	-
100 cl	*S av*	*id*	7, green	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0
100 p	*T av*	*ab*	4, blue	1.0	0.73	0.84	1.0	0.73	0.84	0.03	1.0	0.05	-	-	-
100 p	*T av*	*id*	7, orange	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0
4 cl	*S av*	*ab*	-	1	0.93	0.96	1	0.72	0.84	1	0.01	0.02	-	-	-
4 cl	*S av*	*id*	7, red	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0
4 p	*T av*	*id*	7, magenta	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0
4 cl	*S wd*	*ab*	8, green	1	0.93	0.96	1	0.72	0.84	1	0.01	0.02	-	-	-
4 cl	*S wl*	*ab*	8, orange	1	0.93	0.96	1	0.72	0.84	1	0.01	0.02	-	-	-
4 cl	*S w*	*ab*	8, cyan	1	0.93	0.96	1	0.72	0.84	1	0.01	0.02	-	-	-
4 cl	*S av*	*en*	-	0.59	0.94	0.73	0.79	0.73	0.76	0	0	0	-	-	-
4 cl	*S w*	*lr*	9, green	0.78	0.99	0.87	0.28	0.83	0.42	-	-	-	-	-	-
4 cl	*S w*	*sv*	9, orange	0.71	1	0.83	0.57	0.77	0.66	0	0	0	-	-	-
4 cl	*S w*	*dt*	9, cyan	0.41	0.97	0.58	0.5	0.8	0.62	0	0	0	-	-	-
4 cl	*S w*	*kn*	9, black	0.42	0.98	0.59	0.61	0.78	0.69	0	0	0	-	-	-
4 cl	*S w*	*mp*	9, grey	0.7	0.97	0.81	0.89	0.77	0.83	1	0.01	0.03	-	-	-
4 cl	*S w*	*al*	9, magenta	0.72	1	0.84	0.2	0.94	0.34	-	-	-	-	-	-
4 cl	*S w*	*as*	9, yellow	0.77	0.97	0.86	0.6	0.83	0.69	0	0	0	-	-	-
4 cl	*S w*	*vl*	9, pink	0.73	0.97	0.83	0.85	0.77	0.81	1	0.01	0.03	-	-	-
4 cl	*S w*	*sl*	9, orange	0.71	1	0.83	0.57	0.77	0.66	0	0	0	-	-	-
4 cl	*S w*	*rg*	10, green	1	0.93	0.96	0.68	0.86	0.76	0	0	0	-	-	-
4 cl	*S w*	*rd*	10, orange	1	0.93	0.96	0.68	0.86	0.76	0	0	0	-	-	-
4 cl	*S w*	*rw*	10, cyan	1	0.93	0.96	0.93	0.71	0.8	1	0.01	0.02	-	-	-
100 p	*T tp*	*rd*	11, green	1.0	0.73	0.84	1.0	0.73	0.84	0.03	1.0	0.05	**0.05**	**0.8**	**0.1**

Legends of the first 4 columns are the same as in Table I. Other columns indicate results for the prediction horizons of $h = 1$, 10, 50, 100, Recall, Precision, and F -measure.

The second series considered is a real-world time series (hourly load values in Germany, from 23:00 12/31/2014 to 14:00 20/02/2016 https://www.entsoe.eu/data/power-stats/). A pairing of entropy and complexity for this series amounts to (0.499, 0.372), indicating chaoticity. Its highest

Lyapunov exponent amounts to $0.125 > 0$, which supports the hypothesis that the series is chaotic.

All of the aforementioned methods were used in the analysis of this time series; results are demonstrated in tables 3 and 4, which are similar to tables 1 and 2 for the Lorenz series. We present essentially lesser number of plots for the sake of conciseness.

Table III. The electricity load series. Non-predictable points and error measures

β %	PVs	NP	Figs	$h = 1$			$h = 10$			$h = 50$			$h = 100$		
motif	UPV	T		NP %	MA %	RMS	NP %	MA %	RMS	NP %	MA %	RMS	NP %	MA %	RMS
100 cl	*S av*	*ab*	-	0	14.0	0.007	0	22.3	0.019	0	35.6	0.077	0	42.1	0.083
100 cl	*S av*	*id*	7, green	22	10.4	0.005	32	11.4	0.004	64	12.0	0.005	57	9.2	0.006
100 p	*T av*	*ab*	4, blue	0	17.8	0.01	0	37	0.035	0	44.3	0.08	0	45.7	0.09
100 p	*T av*	*id*	7, orange	17	9.3	0.005	28	10.8	0.005	61	11.4	0.006	63	10.6	0.005
4 cl	*S av*	*ab*	-	0	4.9	0.031	0	31.0	0.154	0	48.4	0.268	0	38.7	0.272
4 cl	*S av*	*id*	7, red	8	3.7	0.022	66	4.6	0.027	100	-	-	100	-	-
4 p	*T av*	*id*	7, magenta												
4 cl	*S wd*	*ab*	8, green	0	3.9	0.025	0	23.0	0.120	0	43.0	0.249	0	33.4	0.241
4 cl	*S wl*	*ab*	8, orange	0	4.9	0.032	0	25.2	0.127	0	41.0	0.236	0	40.4	0.283
4 cl	*S w*	*ab*	8, cyan	0	3.9	0.025	0	19.0	0.102	0	39.8	0.233	0	39.4	0.277
4 cl	*S av*	*en*	-	47	2.3	0.016	31	18.6	0.117	48	43.0	0.270	72	39.5	0.304
4 cl	*S w*	*lr*	9, green	24	2.8	0.018	62	24.7	0.161	100	-	-	100	-	-
4 cl	*S w*	*sv*	9, orange	28	2.6	0.017	70	10.1	0.092	100	-	-	100	-	-
4 cl	*S w*	*dt*	9, cyan	61	2.0	0.016	91	4.8	0.025	100	-	-	100	-	-
4 cl	*S w*	*kn*	9, black	56	2.6	0.017	96	5.2	0.037	100	-	-	100	-	-
4 cl	*S w*	*mp*	9, grey	27	2.8	0.020	54	16.5	0.112	95	18.3	0.121	100	-	-
4 cl	*S w*	*al*	9, magenta	39	3.2	0.021	63	24.1	0.162	100	-	-	100	-	-
4 cl	*S w*	*as*	9, yellow	15	3.1	0.02	35	16.8	0.108	76	38.0	0.216	**92**	**34.3**	**0.322**
4 cl	*S w*	*vl*	9, pink	27	2.8	0.02	53	16	0.111	92	29.5	0.2	100	-	-
4 cl	*S w*	*sl*	9, orange	30	2.5	0.017	75	5.7	0.047	100	-	-	100	-	-
4 cl	*S w*	*rg*	10, green	0	3.9	0.025	11	19.3	0.117	55	43.0	0.244	73	40.4	0.315
4 cl	*S w*	*rd*	10, orange	0	3.9	0.025	9	21.4	0.115	47	45.2	0.228	59	42.9	0.317
4 cl	*S w*	*rw*	10, cyan	0	3.9	0.025	4	17.8	0.101	7	47.6	0.255	19	37.3	0.257
100 p	*T tp*	*rd*	11, green	0	10.3	0.005	0	11.7	0.006	43	34.1	0.05	**51**	**33**	**0.034**

Same legend as in Table I.

Table IV. The electricity load series. Sets of non-predictable points

β %	PVs	NP	Figs	$h = 1$			$h = 10$			$h = 50$			$h = 100$		
motif	UPV	T		Rcl	Prc	F	Rcl	Prc	F	Rcl	Prc	F	Rcl	Prc	F
100 cl	*S av*	*ab*	-	1.0	0.78	0.88	1.0	0.68	0.81	1.0	0.36	0.53	1.0	0.43	0.6
100 cl	*S av*	*id*	7, green	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0
100 p	*T av*	*ab*	4, blue	1	0.83	0.91	1	0.72	0.84	1	0.39	0.56	1	0.37	0.54
100 p	*T av*	*id*	7, orange	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0
4 cl	*S av*	*ab*	-	1	0.92	0.96	1	0.34	0.5	-	-	-	-	-	-
4 cl	*S av*	*id*	7, red	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0
4 p	*T av*	*id*	7, magenta	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0
4 cl	*S wd*	*ab*	8, green	1	0.92	0.96	1	0.34	0.5	-	-	-	-	-	-
4 cl	*S wl*	*ab*	8, orange	1	0.92	0.96	1	0.34	0.5	-	-	-	-	-	-
4 cl	*S w*	*ab*	8, cyan	1	0.92	0.96	1	0.34	0.5	-	-	-	-	-	-
4 cl	*S av*	*en*	-	0.54	0.94	0.69	0.56	0.28	0.37	-	-	-	-	-	-
4 cl	*S w*	*lr*	9, green	0.83	1	0.9	0.7	0.63	0.67	-	-	-	-	-	-
4 cl	*S w*	*sv*	9, orange	0.78	1	0.88	0.62	0.7	0.67	-	-	-	-	-	-
4 cl	*S w*	*dt*	9, cyan	0.42	1	0.6	0.12	0.45	0.19	-	-	-	-	-	-
4 cl	*S w*	*kn*	9, black	0.5	1	0.67	0.18	0.75	0.29	-	-	-	-	-	-
4 cl	*S w*	*mp*	9, grey	0.78	1	0.88	0.76	0.57	0.65	-	-	-	-	-	-
4 cl	*S w*	*al*	9, magenta	0.6	0.95	0.74	0.17	0.86	0.3	-	-	-	-	-	-
4 cl	*S w*	*as*	9, yellow	0.91	0.98	0.95	0.88	0.46	0.6	-	-	-	-	-	-
4 cl	*S w*	*vl*	9, pink	0.79	0.97	0.88	0.76	0.58	0.66	-	-	-	-	-	-

4 cl	*S w*	*sl*	9, orange	0.76	1	0.86	0.56	0.86	0.68	-	-	-	-	-	-
4 cl	*S w*	*rg*	10, green	1	0.92	0.96	0.97	0.37	0.54	-	-	-	-	-	-
4 cl	*S w*	*rd*	10, orange	1	0.92	0.96	0.94	0.35	0.51	-	-	-	-	-	-
4 cl	*S w*	*rw*	10, cyan	1	0.92	0.96	1	0.36	0.52	-	-	-	-	-	-
100 p	*T tp*	*rd*	11, green	1.0	0.83	0.91	0.82	0.7	0.76	0.05	0.14	0.08	**0.11**	**0.67**	**0.19**

Same legend as in Table II.

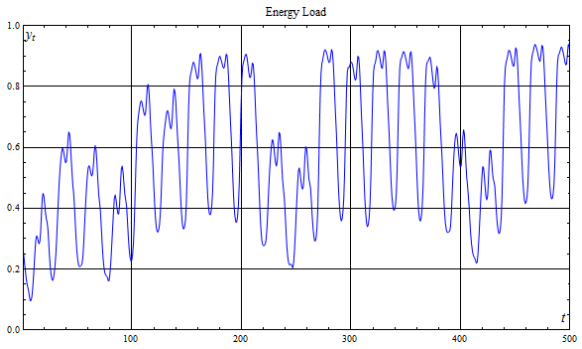
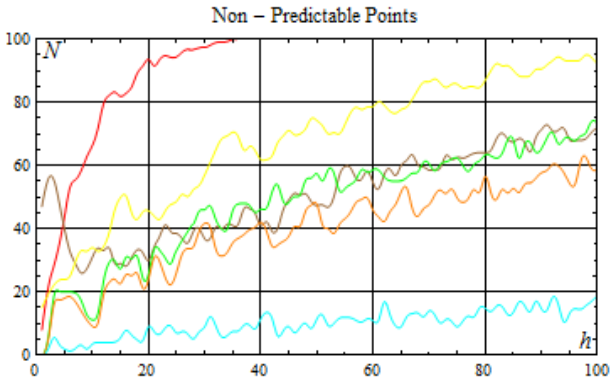


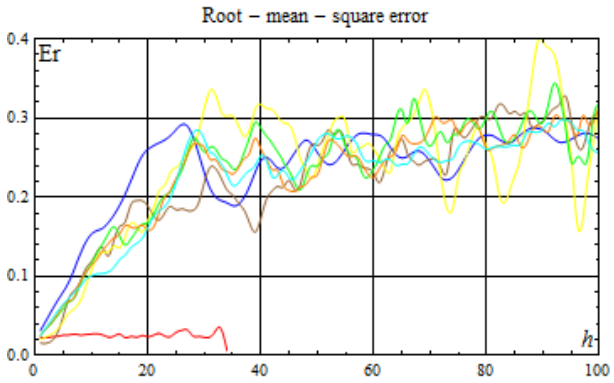
Fig. 11. The energy load series, typical section.

Figure 11 demonstrates a section of the time series. Figure 12 demonstrates correlation between the number of non-predictable points and errors along with the corresponding prediction horizon for using trajectories of possible forecasted values [*rd*]. Figure 13 shows the same graphs when using a set of possible predicted values. Both graphs correspond to the best ways of calculating final predicted values and identifying unforecastable points amongst their respective classes. The best method is the pointwise using 100% of templates with perturbed trajectories (*100p tp*).

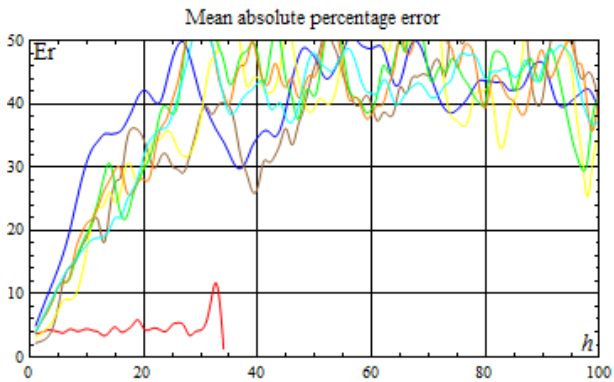
We have recreated a simulation by Prof. Kurogi and his colleagues (2014) (multi-step ahead prediction of chaotic time series and out-of-bag estimate of the prediction performance for model selection). Figure 5c above displays the true Lorenz series (blue), predicted values by Prof. Kurogi and colleagues (green), and the values predicted by the algorithm discussed in the present paper (red dashed line with disks). The algorithm discussed appears to provide reliable predicted values at positions essentially farther (from the last observed value) than that by Prof. Kurogi and his colleagues, but it does not make predictions for all positions.



a

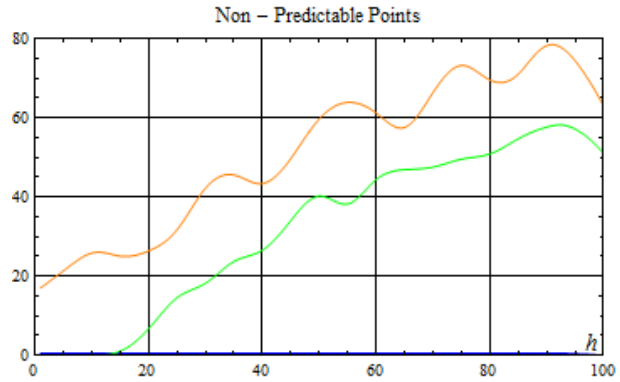


b

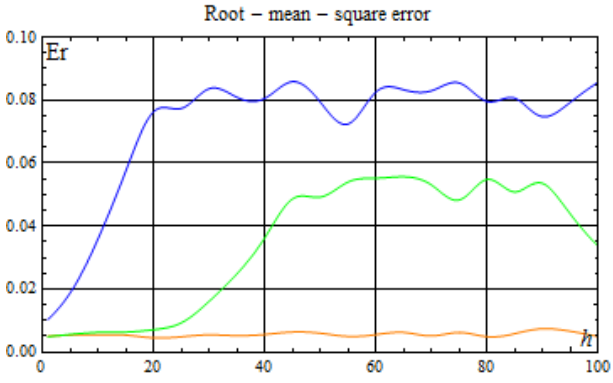


c

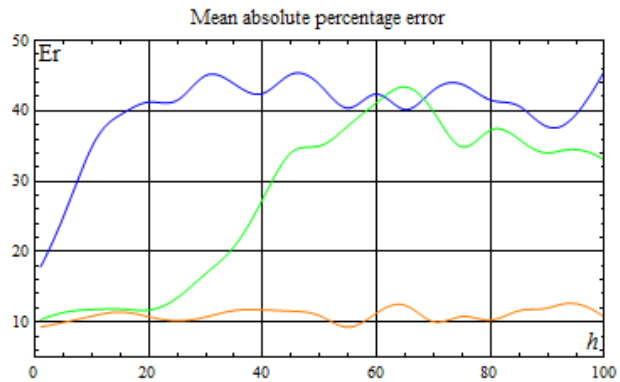
Fig. 12. The number of non-predictable points (a), RMSE (b), and MAPE (c) vs. a prediction horizon for hourly load values in Germany (4% of patterns are used). The techniques identifies non-predictable points based on growth of the spread [*rg*] (green); of the number of DBSCAN clusters [*sd*] (orange); of the number of Wishart clusters [*sw*] (cyan) over three consecutive points; large entropy values [*en*] (brown); AdaBoost of SVM classifiers[as] (yellow). Red curves correspond to the algorithm that identifies non-predictable points with *a priori* information; blue, to the one that does not identify such points.



a



b



c

Fig. 13. Energy series. The number of non-predictable points (a), RMSE (b), and MAPE (c) vs. a prediction horizon for trajectories with random perturbation (100 % of patterns are used; no clustering technique is applied). The algorithm identifies non-predictable points when possible predicted trajectories diverge. Orange curves correspond to the algorithm that identifies non-predictable points with *a priori* information; blue, to the one that does not identify such points.

VI. CONCLUSIONS

Several conclusions may be reached:

The paper discusses several novel strategies for multi-step prediction of chaotic time series. Generalized z-vectors, comprising non-successive observations, make it possible to obtain a fairly large set of possible predicted values for each point to be predicted. Upon examining such a set, it may be ascertained whether it is possible to produce a unified predicted value for it or not (whether the point is predictable or non-predictable), and determine this unified value if it is predictable. With non-predictable points, one may state the partial multi-step prediction problem as a two-objective problem: the first functional minimizes the number of non-predictable points, while the second minimizes the average error for predictable ones.

It appears that for such algorithms the number of non-predictable points grows exponentially with a prediction horizon, but an average error for predictable points remains nearly constant and rather small. The strategies discussed allow a predictive-clustering algorithm to predict positions after a horizon of predictability for a benchmark (the Lorenz) and a real-world time series.

It was concluded that the exact procedure of calculating the final forecasted value has little effect on the accuracy of forecasting, while an effective method of identifying non-predictable points dramatically expands the horizon of prediction. It is worth stressing that that this effect is possible only due to the fact that an algorithm examines sets of non-predictable points and determines whether it possible to calculate a unified prediction value for it (the second kind of non-predictable points).

Large-scale simulation reveals that methods based upon sets of predictable trajectories compare favourably to those based upon sets of predictable points in terms of the ultimate prediction quality. This approach allowed for making some predictions after the horizon of predictability for the corresponding series.

The strategies allow one to work with missing data in quite a natural way.

VII. ACKNOWLEDGEMENTS

The authors are deeply indebted to Mr. Joel Cumberland, HSE for the manuscript proof-reading and language editing. Authors are indebted to colleagues from Supercomputer Modelling Unit, HSE for access to the supercomputer and valuable advice.

The article was prepared within the framework of the HSE University Basic Research Program.

VIII. AUTHORS' CONTRIBUTIONS.

The contributors of this paper are as follows:

- Vasilii A. Gromov: conceptualization, methodology; project administration; supervision; writing - original draft;
- Philip S. Baranov: investigation, software, resources, formal analysis writing, review & editing;
- Alexandr Yu. Tsybakin: investigation, software, formal analysis.

REFERENCES

Aghabozorgi, S., Seyed Shirkhorshidi, A., & Ying Wah, T. (2015). Time-series clustering - A decade review. *Information Systems*, *53*, 16–38. https://doi.org/10.1016/j.is.2015.04.007

Atlee Jackson, E. (1985). The lorenz system: I. the global structure of its stable manifolds. *Physica Scripta*, *32*(5), 469–475. https://doi.org/10.1088/0031-8949/32/5/001

Bao, Y., Xiong, T., & Hu, Z. (2014). Multi-step-ahead time series prediction using multiple-output support vector regression. *Neurocomputing*, *129*, 482–493. https://doi.org/10.1016/j.neucom.2013.09.010

Ben Taieb, S., Bontempi, G., Atiya, A. F., & Sorjamaa, A. (2012). A review and comparison of strategies for multi-step ahead time series forecasting based on the NN5 forecasting competition. *Expert Systems with Applications*, *39*(8), 7067–7083. https://doi.org/10.1016/j.eswa.2012.01.039

Ben Taieb, S., Sorjamaa, A., & Bontempi, G. (2010). Multiple-output modeling for multi-step-ahead time series forecasting. *Neurocomputing*, *73*(10–12), 1950–1957. https://doi.org/10.1016/j.neucom.2009.11.030

Bezruchko, B. P., & Smirnov, D. A. (2010). Extracting Knowledge From Time Series. In *Springer Series in Synergetics*. http://link.springer.com/10.1007/978-3-642-12601-7

Blockeel, H., & De Raedt, L. (1998). Top-down induction of first-order logical decision trees. *Artificial Intelligence*, *101*(1–2), 285–297. https://doi.org/10.1016/S0004-3702(98)00034-4

Bock, H. H. (1970). Automatische Klassifikation. In *Statistische Methoden II* (pp. 36–80). https://doi.org/10.1007/978-3-642-88253-1_10

Canaday, D., Griffith, A., & Gauthier, D. J. (2018). Rapid time series prediction with a hardware-based reservoir computer. *Chaos*, *28*(12). https://doi.org/10.1063/1.5048199

Chandra, R., Ong, Y. S., & Goh, C. K. (2017). Co-evolutionary multi-task learning with predictive recurrence for multi-step chaotic time series prediction. *Neurocomputing*, *243*, 21–34. https://doi.org/10.1016/j.neucom.2017.02.065

Chawla, N. V., Bowyer, K. W., Hall, L. O., & Kegelmeyer, W. P. (2002). SMOTE: Synthetic minority over-sampling technique. *Journal of Artificial Intelligence Research*, *16*, 321–357. https://doi.org/10.1613/jair.953

Cox, D. R. (1961). Prediction by Exponentially Weighted Moving Averages and Related Methods. *Journal of the Royal Statistical Society: Series B (Methodological)*, *23*(2), 414–422. https://doi.org/10.1111/j.2517-6161.1961.tb00424.x

Eckmann, J. P., Kamphorst, S. O., Ruelle, D., & Ciliberto, S. (1986). Liapunov exponents from time series. *Physical Review A*, *34*(6), 4971–4979. https://doi.org/10.1103/PhysRevA.34.4971

Gromov, V. A. (2019). Chaotic Time Series Prediction: Run for the Horizon. *International Conference on Software Testing, Machine Learning and Complex Process Analysis (TMPA-2019)*.

Gromov, V. A., & Borisenko, E. A. (2015). Predictive clustering on non-successive observations for multi-step ahead chaotic time series prediction. *Neural Computing and Applications*, *26*(8), 1827–1838. https://doi.org/10.1007/s00521-015-1845-8

Gromov, V. A., & Konev, A. S. (2017). Precocious identification of popular topics on Twitter with the employment of predictive clustering. *Neural Computing and Applications*, *28*(11), 3317–3322. https://doi.org/10.1007/s00521-016-2256-1

Gromov, V. A., & Shulga, A. N. (2012). Chaotic time series prediction with employment of ant colony optimization. *Expert Systems with Applications*, *39*(9), 8474–8478. https://doi.org/10.1016/j.eswa.2012.01.171

Gromov, V. A., Voronin, I. M., Gatylo, V. R., & Prokopalo, E. T. (2017). Active cluster replacement algorithm as a tool to assess bifurcation early-warning signs for von Karman equations. *Artificial Intelligence Research*, *6*(2), 51. https://doi.org/10.5430/air.v6n2p51

Guntu, R. K., Yeditha, P. K., Rathinasamy, M., Perc, M., Marwan, N., Kurths, J., & Agarwal, A. (2020). Wavelet entropy-based evaluation of intrinsic predictability of time series. *Chaos*, *30*(3). https://doi.org/10.1063/1.5145005

Jaeger, H. (2004). Harnessing Nonlinearity: Predicting Chaotic Systems and Saving Energy in Wireless Communication. *Science*, *304*(5667), 78–80. https://doi.org/10.1126/science.1091277

Kantz, H., & Schreiber, T. (2003). *Nonlinear Time Series Analysis* (2nd ed.). Cambridge University Press. https://doi.org/10.1017/CBO9780511755798

Kurogi, S., Shigematsu, R., & Ono, K. (2014). Properties of direct multi-step ahead prediction of chaotic time series and out-of-bag estimate for model selection. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, *8835*, 421–428. https://doi.org/10.1007/978-3-319-12640-1_51

Lapko, A., & Chentsov, S. (2000). *Nonparametric Information Processing Systems*. Nauka.
Lu, Z., Pathak, J., Hunt, B., Girvan, M., Brockett, R., & Ott, E. (2017). Reservoir observers: Model-free inference of unmeasured variables in chaotic systems. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, *27*(4), 041102. https://doi.org/10.1063/1.4979665
Malinetskii, G., & Potapov, A. (2000). *Modern Problems of Nonlinear Dynamics*. Editorial URSS.
Maronna, R. (2016). Charu C. Aggarwal and Chandan K. Reddy (eds.): Data clustering: algorithms and applications. *Statistical Papers*, *57*(2), 565–566. https://doi.org/10.1007/s00362-015-0661-7
Martínez-Álvarez, F., Troncoso, A., Riquelme, J. C., & Aguilar Ruiz, J. S. (2011). Energy time series forecasting based on pattern sequence similarity. *IEEE Transactions on Knowledge and Data Engineering*, *23*(8), 1230–1243. https://doi.org/10.1109/TKDE.2010.227
Mirzaee, H. (2009). Long-term prediction of chaotic time series with multi-step prediction horizons by a neural network with Levenberg-Marquardt learning algorithm. *Chaos, Solitons and Fractals*, *41*(4), 1975–1979. https://doi.org/10.1016/j.chaos.2008.08.016
Ong, P., & Zainuddin, Z. (2019). Optimizing wavelet neural networks using modified cuckoo search for multi-step ahead chaotic time series prediction. *Applied Soft Computing Journal*, *80*, 374–386. https://doi.org/10.1016/j.asoc.2019.04.016
Rosso, O. A., Larrondo, H. A., Martin, M. T., Plastino, A., & Fuentes, M. A. (2007). Distinguishing noise from chaos. *Physical Review Letters*, *99*(15). https://doi.org/10.1103/PhysRevLett.99.154102
Sangiorgio, M., & Dercole, F. (2020). Robustness of LSTM neural networks for multi-step forecasting of chaotic time series. *Chaos, Solitons and Fractals*, *139*. https://doi.org/10.1016/j.chaos.2020.110045
Small, M. (2005). Applied nonlinear time series analysis : applications in physics, physiology and finance. In *World Scientific series in nonlinear science, Series A* (Issue 52).
Universit, G. B., & Bontempi, G. (2014). Long term time series prediction with multi- input multi-output local learning Long Term Time Series Prediction with. *Proc. 2nd ESTSP*, *January 2008*, 129–138.
Waheeb, W., & Ghazali, R. (2016). Multi-step time series forecasting using ridge polynomial neural network with error-output feedbacks. *Communications in Computer and Information Science*, *652*, 48–58. https://doi.org/10.1007/978-981-10-2777-2_5
Wang, K., Li, K., Zhou, L., Hu, Y., Cheng, Z., Liu, J., & Chen, C. (2019). Multiple convolutional neural networks for multivariate time series prediction.

Neurocomputing, *360*, 107–119. https://doi.org/10.1016/j.neucom.2019.05.023

Wang, R., Peng, C., Gao, J., Gao, Z., & Jiang, H. (2020). A dilated convolution network-based LSTM model for multi-step prediction of chaotic time series. *Computational and Applied Mathematics*, *39*(1). https://doi.org/10.1007/s40314-019-1006-2

Wishart, D. (1969). Numerical classification method for deriving natural classes. *Nature*, *221*(5175), 97–98. https://doi.org/10.1038/221097a0

Ye, R., & Dai, Q. (2019). MultiTL-KELM: A multi-task learning algorithm for multi-step-ahead time series prediction. *Applied Soft Computing Journal*, *79*, 227–253. https://doi.org/10.1016/j.asoc.2019.03.039

Appendix A

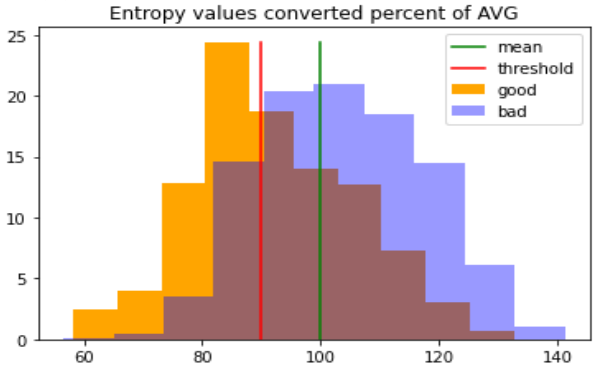


Fig. A1. A typical empirical distribution for entropy for the ground-truth non-predictable points (orange) and its complement to a testing set (blue) for the Lorenz series; brown colour denotes an intersection. Green line stands for a mean value for non-predictable points; red, for a threshold between predictable and non-predictable points.

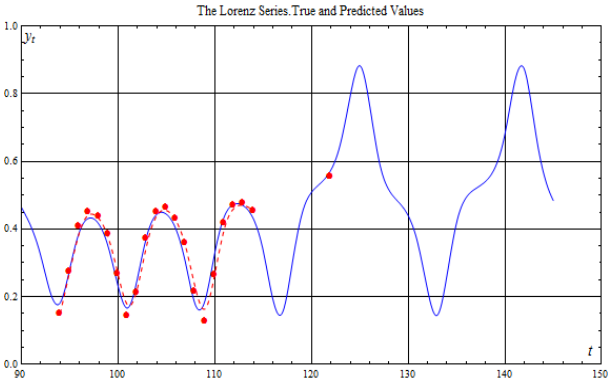


Fig. A2. The Lorenz series: true trajectory (blue curve) and predicted points (red dashed curve with disks). The algorithm identifies non-predictable points using logistic regression [*lr*].

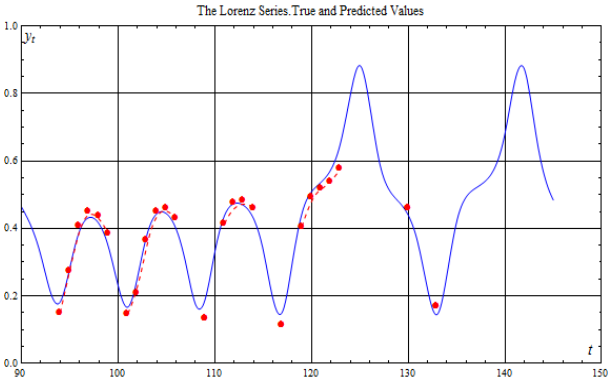


Fig. A3. The Lorenz series: true trajectory (blue curve) and predicted points (red dashed curve with disks). The algorithm identifies non-predictable points using multi-layer perceptron [*mp*].

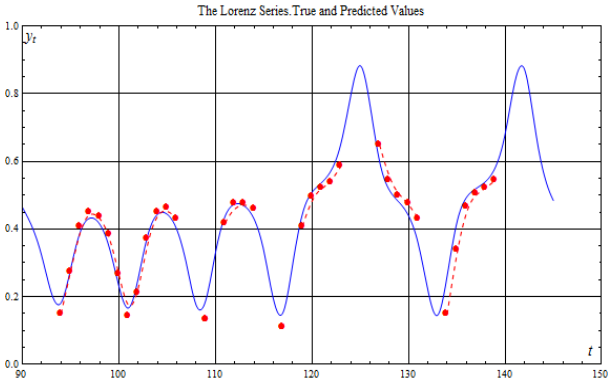


Fig. A4. The Lorenz series: true trajectory (blue curve) and predicted points (red dashed curve with disks). The algorithm identifies non-predictable points using AdaBoost that combines support vector machines [*as*].

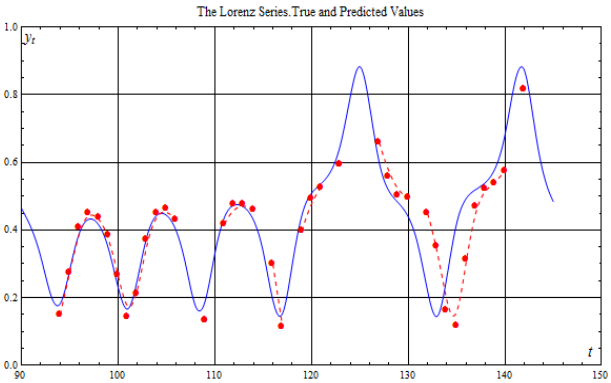


Fig. A5. The Lorenz series: true trajectory (blue curve) and predicted points (red dashed curve with disks). The algorithm identifies non-predictable points using stacking of logistic regressions [*sl*].

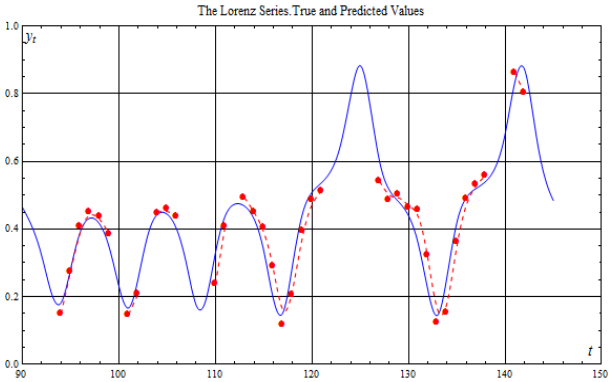


Fig. A6. The Lorenz series: true trajectory (blue curve) and predicted points (red dashed curve with disks). The algorithm identifies non-predictable points using growth of spread [*rg*].

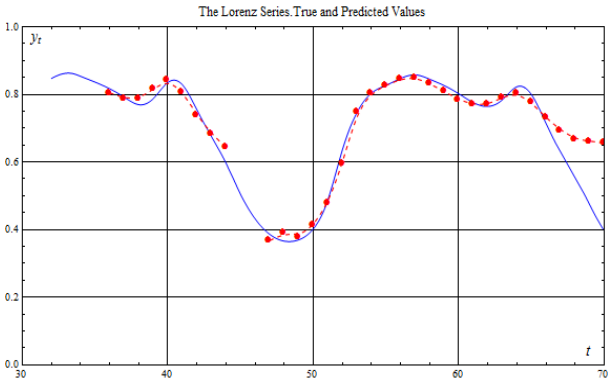


Fig. A7. German energy time series: true trajectory (blue curve) and predicted points (red dashed curve with disks). The algorithm identifies non-predictable points using AdaBoost of SVM classifier [*as*].

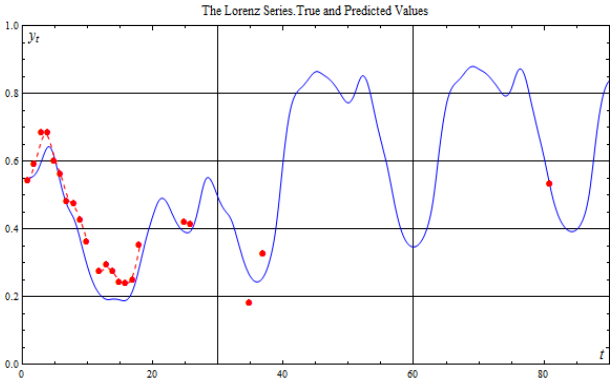
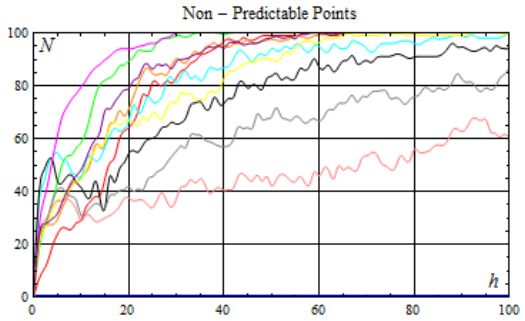
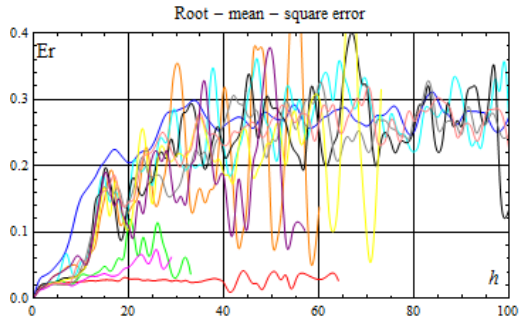


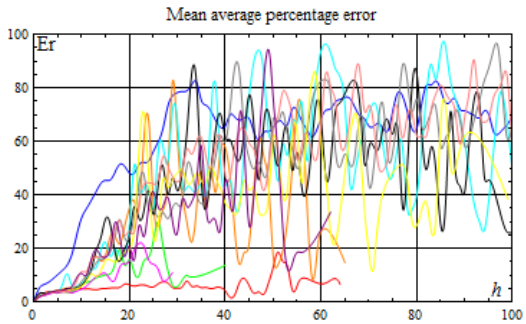
Fig. A8. German energy time series: true trajectory (blue curve) and predicted points (red dashed curve with disks). The algorithm employs possible predicted trajectories [*T*] identifies non-predictable points using spread growth [*rg*].



a

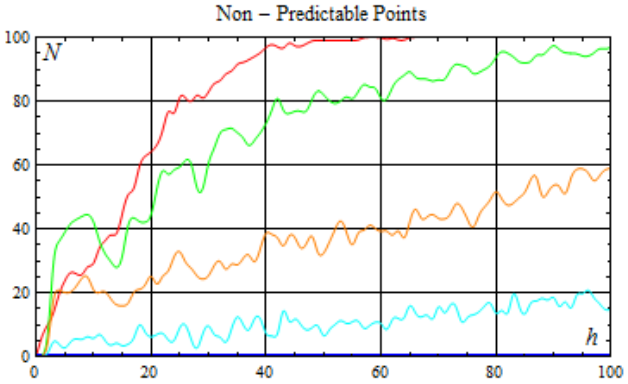


b

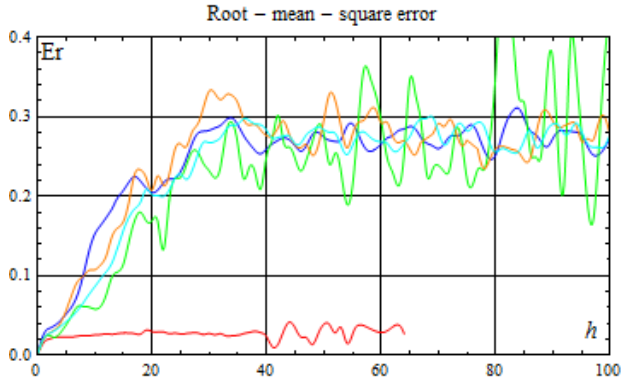


c

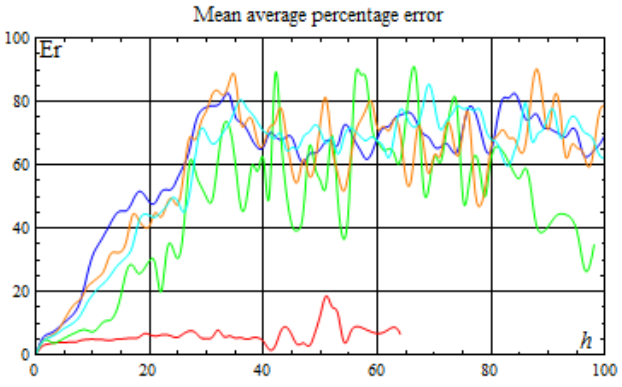
Fig. A9. The number of non-predictable points (a), RMSE (b), and MAPE (c) vs. a prediction horizon (4% of patterns are used). The techniques identify non-predictable points with logreg clasification [*lr*] (green); support vector machine [*sv*] (orange); decision tree [*dt*] (cyan); knn [*kn*] (black); multilayer perceptron [*mp*] (gray); adaboost logreg [*al*] (magenta); adaboost svm [*as*] (yellow); voting [*vl*] (pink); stacking [*sl*] (orange). Red curves correspond to the algorithm that identifies non-predictable points with *a priori* information; blue, to the one that does not identify such points.



a



b



c

Fig. A10. The number of non-predictable points (a), RMSE (b), and MAPE (c) vs. a prediction horizon (4% of patterns are used). The techniques identify non-predictable points based on growth of the spread [*rg*] (green); of the number of DBSCAN clusters [*sd*] (orange); of the number of Wishart clusters [*sw*] (cyan) over three consecutive points. Red curves correspond to the algorithm that identifies non-predictable points with *a priori* information; blue, to the one that does not identify such points.

Appendix B

Table B1. Abbreviations for methods of *A training set*

p	**Pointwise**	1
cl	**Cluster**	2

Table B2. Abbreviations for methods of *A unified predicted value* (algorithm types)

T	**Trajectory**. One uses a set of possible predicted trajectories	1
S	**Set**. One uses a set of possible predicted values	2

Table B3. Abbreviations for methods of *A unified predicted value* (calculation methods)

av	**Average.** The value is an average	1.1
wl	**Weighted sum, length.** The value is a weighted average with weights determined by a prediction length	1.2
wd	**Weighted sum, distance.** Weights are determined by a distance to a cluster centre	1.2
w	**Weighted sum, combined.** Weights are determined by a product of the two previous weights	1.2
cm	**Cluster max.** The value is a centre of the largest cluster	1.3
cw	**Cluster max, weighted.** The value is a centre of the largest cluster, and clustering is performed for possible predicted values with relatively large weights	1.4
fs	**Fuzzy set.** Denotes fuzzy clustering	1.5
rw	**Roulette wheel.** The value is the centre of the cluster chosen probabilistically	1.6
mf	**Most frequent.** The value is the most frequent possible predicted value	1.7
mp	**Most frequent, perturbed.** The most frequent possible predicted value with perturbation	1.8
tp	**Trajectory, perturbed.** The value is an average over the last points of the perturbed trajectories	2.1
tm	**Trajectory, multiple clustering.** The value is an average over the last points of the branching trajectories	2.2
tr	**Trajectory, reduced information.** The value is an average over the last points of the trajectories that rely on reduced initial information	2.3

Table B4. Abbreviations for methods of *Algorithms to identify non-predictable points*

fp	**Forced prediction**. Non-predictable points are not identified altogether, predicted values are forcibly calculated at all intermediate points	-
ab	**Absent.** Implies that a set of non-predictable points consists only of the points that the algorithm failed to find motifs close enough, sets of possible predicted values are not analysed (the first extreme case)	-
id	**Ideal.** Sets of possible predicted values are constructed using *a priori* information (the second extreme case)	-
en	**Entropy.** Implies that predictable and non-predictable points are separated using entropies of their possible predicted values distributions	-
lr	**Logistic regression.** One applies the classifier to features of possible predicted values sets	1.1
sv	**Support vector machine.** One applies a support vector machine	1.2
dt	**Decision tree.** One applies a decision tree C4.5	1.3
kn	**K-nearest neighbours.** One applies a k-nearest neighbours classifier	1.4
mp	**Multi-layer perceptron.** One applies a multi-layer perceptron	1.5
al	**AdaBoost logistic regression.** Classifiers (logistic regressions) are assembled using AdaBoost	2.1
as	**AdaBoost SVM.** Classifiers (SVM) are assembled using AdaBoost	2.1
sl	**Stacking logistic regression.** Classifiers (logistic regression) are assembled by stacking	2.2
vl	**Voting logistic regression.** Classifiers (logistic regression) are assembled by voting	2.3
ls	**Large spread.** Implies that the point is non-predictable if the spread of its possible predicted values exceeds an average spread	3.1
rg	**Rapid growth.** The spread grows monotonically at several consecutive points	3.2
rd	**Rapid growth. DBSCAN.** A monotonic growth of the number of possible predicted value clusters	3.2

	obtained with DBSCAN	
rw	**Rapid growth. The Wishart clustering.** A monotonic growth of the number of possible predicted value clusters obtained with the Wishart clustering	3.2
ca	**Compare with average.** One compares the principle predicted trajectory with a weighted average of other predicted trajectories	3.3
ad	**Average, difference.** One calculates an average of modulus of a difference	3.4
wa	**Weighted average**. One calculates a weighted average of modulus of a difference	3.5
wd	**Weighted compare.** One calculates a weighted average of modulus of a difference	3.6

- Prediction after the horizon of predictability for chaotic time series
- Predictable and non-predictable points
- Identifying non-predictable points
- Results for benchmark and real-world chaotic time series

ORCID

Vasilii A. Gromov

0000-0001-5891-6597

Declaration of interests

☒ The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

☐The authors declare the following financial interests/personal relationships which may be considered as potential competing interests:
