Bernstein polynomials for adaptive evolutionary prediction of short-term time series

Kristina Lukoseviciute[a], Rita Baubliene[a], Daniel Howard[b], Minvydas Ragulskis[a,*]

[a] *Research Group for Mathematical and Numerical Analysis of Dynamical Systems, Kaunas University of Technology, Studentu 50–147, Kaunas LT–51368, Lithuania*
[b] *Howard Science Limited, Malwern, U.K.*

Abstract

We introduce a short-term time series prediction model by means of evolutionary algorithms and Bernstein polynomials. This adapts Bernstein-type algebraic skeletons to extrapolate and predict short time series. A mixed smoothing strategy is used to achieve the necessary balance between the roughness of the algebraic prediction and the smoothness of the moving average. Computational experiments with standardized real world time series illustrate the accuracy of this approach to short-term prediction.

Keywords: Bernstein polynomial, time series prediction, evolutionary algorithms

1. Introduction

Forecasting is a modelling challenge that relies on a time series analysis. Its aim is to identify a model in time-stamped data presumably generated by some process. Extrapolating by means of this model it makes reliable predictions for unseen data. Recent decades have delivered various models and techniques that are suited to long-term or short-term time series forecasting [1]. Unfortunately, the sheer amount of data needed for training,

*Corresponding author.
Email addresses: kristina.lukoseviciute@ktu.lt (Kristina Lukoseviciute), rita.palivonaite@ktu.lt (Rita Baubliene), dr.daniel.howard@gmail.com (Daniel Howard), minvydas.ragulskis@ktu.lt (Minvydas Ragulskis)
URL: www.personalas.ktu.lt/∼mragul (Minvydas Ragulskis)

validating and testing mostly renders long-term time series analysis implausible. Yet, a one-step forward future horizon is adequate for short-term time series forecasting [1] delivering methods which are widely used in high frequency time series analysis with intra-daily data values [2]. Short-term time series predictors are used in finance [3, 4, 5]; electricity demand and the associated price forecasting problem [6, 7, 8]; wind power; passenger demand [9] and many other applications.

Time series forecasting techniques can be coarsely grouped into classical linear modelling, such as simple exponential smoothing [10], Holt-Winter's methods [11] or Autoregressive Integrated Moving Average (ARIMA) [12], and modern non-linear modelling that is based on soft computing. The latter includes regime-switching models comprising a wide variety of threshold autoregressive models [13, 14, 15]: self exciting models [15, 16, 17], smooth transition models [18] and continuous-time models [19, 20]. Hybrid forecasting methods combine regression, data smoothing, and other techniques to produce forecasts that make up for the comparative deficiencies of individual methods.

A large number of linear and non-linear methods of forecasting appear in the literature, with some methods claiming to do a better job than others under competing assumptions, for example: when given only a short series of input data, or if applied to long-term forecasting [1]. The literature [3]– [23] covers a wide-variety of techniques that include various flavours of signal processing, support vector machines, ARIMA, Artificial Neural Network (ANN) and Evolutionary Algorithms (EA).

The reader may wonder why there is a continued and strong interest in a plethora of algorithms. The no-free-lunch theorems [24, 25] lead to the conclusion that a problem can always be found to defeat any algorithm. Indeed, practical interest in the development of new and hybrid algorithms is warranted because of this reality that no single method will outperform all others in every single situation. At the same time, as Stafford Beer once observed [26], problems of practical interest cannot take an algorithm completely by surprise because the regularities that they comprise are of this world. Real-world problems have neither been designed nor contrived to defeat a popular algorithm. A taxonomy of practical problems, therefore, exists, and it motivates the search for improved algorithms that suit different classes of problems.

In our earlier work [27, 28, 29], special EA schemes for the identification of near-optimal algebraic skeleton sequences based on Prony interpolants

(represented as linear recurrence sequences (LRS) in the discrete case) are developed. The high variability of Prony interpolants is managed by means of external [27], internal [28] and mixed smoothing strategies [29]. Such Prony polynomial based techniques are shown to be well applicable for the prediction of stationary short-term time series - but are not well suited for such highly variable time series as Odonovan and Montgome [30].

A natural question arises as to whether some other algebraic interpolants, covering a wider class of functions than the Prony polynomials, might be of benefit to short-term time series forecasting applications. This paper aims to investigate this question with Bernstein polynomials [31] and is structured as follows. Preliminaries on forecasting techniques based on Prony polynomials are given in the second section. Advances over research in [27, 28, 29] are presented in the third section. Evolutionary algorithms for the identification of a near-optimal set of corrections are developed in the fourth section, the validation of the model is performed in the fifth section, computational experiments with standard real world time series are presented in the sixth section, and concluding remarks are given in the final section.

2. Preliminaries

The short overview of time series prediction based on Prony polynomials [27, 28, 29] in this section, helps to introduce the method that uses Bernstein polynomials.

2.1. The order of a sequence

Let us consider an order n LRS with constant coefficients:

$$x_k = a_{n-1}x_{k-1} + a_{n-2}x_{k-2} + \ldots + a_0 x_{k-n};\ k = 0, 1, \ldots; \tag{1}$$

where coefficients $a_j, j = 0, 1, \ldots, n-1$ are constants. The initial conditions $x_k, k = 0, 1, \ldots, n-1$ uniquely determine the evolution of this LRS [32]. The auxiliary polynomial to Eq. (1) reads

$$P(\rho) = \rho^n - a_{n-1}\rho^{n-1} - a_{n-2}\rho^{n-2} - \ldots - a_0, \tag{2}$$

where ρ is the root of the characteristic equation. The LRS takes the form

$$x_j = \mu_1\rho_1^j + \mu_2\rho_2^j + \ldots + \mu_n\rho_n^j \tag{3}$$

if all n roots of Eq. (2) $\rho_1, \rho_2, \ldots, \rho_n$ are distinct and coefficients are determined to fit the initial conditions of the recurrence. If some roots coincide, then the recurrence takes the form:

$$x_j = \sum_{k=1}^{r} \sum_{l=0}^{n_k-1} \mu_{kl} \binom{j}{l} \rho_k^{j-1} \tag{4}$$

where r is the number of distinct roots, n_k is the multiplicity index of the k-th root; $n_1 + n_2 + \ldots + n_k = n$. If the order of LRS is not known in advance, the algorithm for the reconstruction of the model of LRS from a sequence $(x_j)_{j=0}^{+\infty}$ is more complex. The Hankel transform of $(x_j)_{j=0}^{+\infty}$ produces the sequence $(h_j)_{j=0}^{+\infty}$ where $h_j = \det(H_j)$ and $H_j = (x_{k+l-2})_{1 \le k,l \le (j+1)}$ is a Hankel matrix – catalecticant matrix of dimension $(j+1) \times (j+1)$. If there exists an $n \ge 1$ such that $h_n \neq 0$ but $h_k = 0$ for all $k > n$, then $(x_j)_{j=0}^{+\infty}$ is an LRS and its order is n, and the auxiliary Eq. (2) now reads:

$$\det \begin{bmatrix} x_0 & x_1 & \ldots & x_n \\ x_1 & x_2 & \ldots & x_{n+1} \\ & & \ldots & \\ x_{n-1} & x_n & \ldots & x_{2n-1} \\ 1 & \rho & \ldots & \rho^n \end{bmatrix} = 0. \tag{5}$$

This linear system of algebraic equations has a unique solution because $h_n \neq 0$ [33]. How can one build a model of the process using Eq. (4) if the observed sequence is not an algebraic equation? The idea behind the algebraic prediction technique is based on the identification of the skeleton algebraic sequences and is presented in [27]. Such a concept is based on the assumption that many time series are contaminated with additive noise. This is a strong reason why central to algebraic prediction models is the detection of a base skeleton algebraic sequence in the time series data that removes this additive noise. Further modifications are presented in [28, 29].

2.2. Algebraic prediction, external smoothing (APES)

APES is presented in [27]. Let $2n+1$ observations be available for building the model of the process: $(x_k)_{k=0}^{2n}$; where x_{2n} is the value of the observation at the present time. The assumption made that the sequence consists of the addition of noise to some algebraic progression means that the determinant $d_n \neq 0$. The goal now becomes to identify a vector of corrections

$(\varepsilon_0, \varepsilon_1, \ldots, \varepsilon_{2n})$ such that determinant $\tilde{d}_n$ of differences to these corrections is minimized:

$$\tilde{d}_n = \det \begin{bmatrix} x_0 - \varepsilon_0 & x_1 - \varepsilon_1 & \cdots & x_n - \varepsilon_n \\ x_1 - \varepsilon_1 & x_2 - \varepsilon_2 & \cdots & x_{n+1} - \varepsilon_{n+1} \\ & & \cdots & \\ x_n - \varepsilon_n & x_{n+1} - \varepsilon_{n+1} & \cdots & x_{2n} - \varepsilon_{2n} \end{bmatrix} \tag{6}$$

Corrections are identified before any predictions are made with the goal of minimizing any distortions of the original time series and so the fitness function which is the subject of the evolutionary pressure as applied by the EA [27] is given by:

$$F_e\left(\varepsilon_0, \varepsilon_1, \ldots, \varepsilon_{2n}\right) = \frac{1}{a|\tilde{d}_n| + \sum_{k=0}^{2n} \lambda_k \left|\varepsilon_k\right|} \tag{7}$$

where the penalty constant, a, is chosen to reflect a desired balance between the magnitude of the determinant and the sum of weighted corrections; λ_k define the tolerance corridor for corrections ε_k. The closer is the element to the last time point, the higher is its weight – and the lower is the variability of its correction.

The evolutionary computation strategy developed in [27] averages 100 reconstructed algebraic skeletons for every single prediction – a single step prediction horizon – to discover a near-optimal vector of corrections.

2.3. Algebraic prediction, internal smoothing (APIS)

An alternative forecasting strategy for short time series in [28] assumes that $2n$ observations are available: $(x_k)_{k=0}^{2n-1}$; and x_{2n-1} is the value of the observation at the current time. Algebraic equation $d_n = 0$ uniquely determines x_{2n}, the element that follows in this sequence. However, such straightforward computations cannot produce satisfactory forecasts. Thus, instead of trying to build such a direct algebraic model into the future, a conciliation between the variability of the skeleton algebraic sequences and the smoothness of the averaged estimates is introduced:

$$F_i\left(\varepsilon_0, \varepsilon_1, \ldots, \varepsilon_{2n-1}\right) = \frac{1}{a \sum \lambda_k \left|\varepsilon_k\right| + \left|\tilde{x}_{2n} - \bar{x}_{2n}\right|}, \tag{8}$$

where $\tilde{x}_{2n}$ is determined from

$$\det \begin{bmatrix} x_0 - \varepsilon_0 & x_1 - \varepsilon_1 & \cdots & x_n - \varepsilon_n \\ & & \cdots & \\ x_n - \varepsilon_n & x_{n+1} - \varepsilon_{n+1} & \cdots & \tilde{x}_{2n} \end{bmatrix} = 0 \qquad (9)$$

and the quantity of interest, the smoothed moving average of $(x_k - \varepsilon_k)_{k=0}^{2n-1}$, we now denote as $\bar{x}_{2n}$. An evolutionary computation strategy is developed in [28] which allows for the identification of a near-optimal vector of corrections by averaging 100 reconstructed algebraic skeletons for every single prediction (the prediction horizon is 1 step).

The principal differences between APIS and APES are in the extrapolation schemes. The value of the determinant $\tilde{d}_n$ is minimized, but not driven down to zero in APES. APES cost function (Eq. (7)) is constructed in such a way that both the total perturbation of the sequence and the absolute value of $\tilde{d}_n$ are minimized simultaneously. A large number of extrapolations are performed for different perturbations (in the same observation window). Then the predictions are averaged – hence the algorithm is given by name: "external smoothing".

On the contrast, Eq. (9) in APIS is exact. The only unknown in Eq. (9) is $\tilde{x}_{2n}$ – if only all corrections are pre-determined. The sequence is perturbed and extrapolated in APIS. But the perturbation is constructed in such a way that the extrapolation would be not far from the moving average – hence the name "internal smoothing". Note that measures related to the prediction errors are not present in the cost functions neither in APES nor in APIS.

2.4. Algebraic prediction, mixed smoothing (APMS)

An improved prediction strategy for short time series is proposed in [29]. Let $2n + 1$ observations be available for building the model of the process: $(x_k)_{k=0}^{2n}$; where x_{2n} is the value of the observation at the current time. The fitness function for the set of corrections $(\varepsilon_k)_{k=0}^{2n}$ is given by:

$$F_m(\varepsilon_0, \varepsilon_1, \ldots, \varepsilon_{2n}) = \frac{1}{\sum_{k=0}^{2n} |\varepsilon_k| + a\hat{E} + b\,|\hat{x}_{2n+1} - \bar{x}_{2n+1}|}; a, b > 0; \qquad (10)$$

where parameters a and b are penalty proportions between and balancing different terms in the denominator of the fitness function; $\hat{E} = \sqrt{\frac{1}{2n+1}\sum_{k=0}^{2n}(\hat{x}_k - x_k)^2}$ is the error computed between observations $(x_k)_{k=0}^{2n}$ and the reconstructed algebraic skeleton. The algebraic prediction $\hat{x}_{2n+1}$ can be considered to be the

extrapolation of the algebraic skeleton: $\hat{x}_j = \sum_{r=1}^{l} \sum_{k=0}^{m_r-1} \mu_{rk} \binom{j}{k} \rho_r^{j-k}$. It is important to note that the identification of a near-optimal vector of corrections in APMS is obtained without any external averaging (the prediction horizon is 1 step).

As mentioned previously, we now replace Prony polynomials by Bernstein polynomials. However, it seems that a straightforward replacement of Prony interpolants by Bernstein interpolants is impossible. Why? All target functions in [27, 28, 29] are based on determinants of catalectican Hankel matrices constructed from the original time series. Such an approach is completely natural for Prony interpolants since Prony eigenvalues for a particular algebraic skeleton are computed from the Hankel matrix of an appropriate order. From the physical point of view, Prony polynomials describe a linear mixture of exponentially decaying (or growing) harmonics. Conversely, Bernstein polynomials describe a much richer variety of functions; for example, Prony polynomials cannot describe a sequence of factorials. However, determinants of Hankel matrices are neither related to the order of Bernstein polynomials nor to any parameters of Bernstein interpolants. Hence, a new strategy is required for building the target functions for such a time series forecasting approach.

3. Construction of a predictor based on Bernstein polynomials

The basis of Bernstein polynomials is developed mainly in the approximation theory of computer aided geometric design [34, 35]. Bernstein polynomials are widely used in various fields of mathematics owing to their simple probabilistic behavior, mostly in the development of nonparametric regression models [36, 37, 38, 39], fuzzy logic based regression procedures [40], the analysis of distribution and density functions [41, 42, 43], and various approximation applications [44, 45]. Bernstein polynomials are used for analysis and forecasting of chaotic time series: the Bernstein Neural Network (BNN) is used to forecast the chaotic wind power series [46]. An algorithm based on the band-limited signal extrapolation by truncated Bernstein polynomials is proposed in [47]. This scheme utilizes a finite number of equally spaced samples of the given function and provides a time-limited polynomial approximation. Another algorithm for local prediction of chaotic sequences with variable frame length is presented in [48]. This interpolation method is used to increase the available sample data, then the chaotic dynamical system is modelled by using the least square algorithm based on the Bernstein po-

lynomial. The optimal frame length and interpolation points are optimized in every frame in order to improve the prediction performance [48].

This paper proposes a completely different approach based on the extrapolation of the Bernstein scheme with mixed smoothing. The structure of the proposed algorithm is discussed in detail in the following sections.

3.1. Straightforward extrapolation of the the Bernstein scheme

Polynomials are very useful tools because they are defined in simple terms; can be calculated quickly on computer systems; and represent a huge variety of functions [31]. Bernstein polynomials remain popular for practical computations. For a given time series $x_0, x_1, \ldots, x_n$ the Bernstein polynomial of degree n can be defined as:

$$B_n(t) = \sum_{k=0}^{n} \binom{n}{k} (1-t)^{(n-k)} t^k x_k \tag{11}$$

where t is time constructed on the uniform grid at nodes $t_0 = 0$, $t_1 = \frac{1}{n}, t_2 = \frac{2}{n}, \ldots, t_n = 1$. The Bernstein interpolant can be extrapolated for the following time as $t_{n+1} = 1 + \frac{1}{n}$:

$$B_n\left(1+\frac{1}{n}\right) = \sum_{k=0}^{n} \binom{n}{k} \left(-\frac{1}{n}\right)^{(n-k)} \left(1+\frac{1}{n}\right)^{k} x_k \tag{12}$$

Example 1. Assume the time series to be $x_0 = 2$, $x_1 = 1.74$, $x_2 = 1.5$, $x_3 = 1.3$ and $x_4 = 1.17$. This time series is generated with polynomial $x(t) = 0.1t^3 - 0.8t + 3$ with $t = \left\{0, \frac{1}{3}, \frac{2}{3}, 1, \frac{4}{3}\right\}$. A Bernstein polynomial of 3-rd degree extrapolates using equation (12) to predict $\hat{x}_4$:

$$\hat{x}_4 = \binom{3}{0}\left(-\frac{1}{3}\right)^3 x_0 + \binom{3}{1}\left(-\frac{1}{3}\right)^2\left(\frac{4}{3}\right) x_1 + \binom{3}{2}\left(-\frac{1}{3}\right)\left(\frac{4}{3}\right)^2 x_2 + \binom{3}{3}\left(\frac{4}{3}\right)^3 x_3 = 1.33$$

The inaccuracy of this scheme is noticeable when comparing x_4 to $\hat{x}_4$ values.

3.2. The improved scheme

It is submitted that use of Bernstein polynomials of higher degree $n+1$ yields a better behaved time series extrapolation. Let us assume that $B_{n+1}(0) = x_0$, $B_{n+1}\left(\frac{1}{n}\right) = x_1$, $B_{n+1}\left(\frac{2}{n}\right) = x_2$,...,$B_{n+1}\left(1+\frac{1}{n}\right) = x_{n+1}$. Variable x_{n+1} can be expressed from the following equality:

$$\begin{aligned}&\binom{n+1}{0}(1-t_{n+1})^{n+1}t_{n+1}^{0}x_0+\binom{n+1}{1}(1-t_{n+1})^{n}t_{n+1}x_1+\ldots\\&+\binom{n+1}{n}(1-t_{n+1})t_{n+1}^{n}x_n+\binom{n+1}{n+1}(1-t_{n+1})^{0}t_{n+1}^{n+1}x_{n+1}=x_{n+1}.\end{aligned}\tag{13}$$

Denoting $\hat{x}_{n+1} = x_{n+1}$ to be the predicted value, this is obtained as:

$$\hat{x}_{n+1}=\frac{\sum_{k=0}^{n}\binom{n}{k}(1-t_{n+1})^{n+1-k}t_{n+1}^{k}x_k}{1-t_{n+1}^{n+1}},t_{n+1}=1+\frac{1}{n}.\tag{14}$$

Example 2. Using the same time series as that of Example 1, the improved extrapolation of Eq. (14) predicts $\hat{x}_4$ as:

$$\hat{x}_4=\frac{\binom{4}{0}\left(-\frac{1}{3}\right)^4x_0+\binom{4}{1}\left(-\frac{1}{3}\right)^3\frac{4}{3}x_1+\binom{4}{2}\left(-\frac{1}{3}\right)^2\left(\frac{4}{3}\right)^2x_2+\binom{4}{3}\left(-\frac{1}{3}\right)\left(\frac{4}{3}\right)^3x_3}{1-\left(\frac{4}{3}\right)^4}=1.13.$$

Example 2 demonstrates that the improved scheme outperforms a straightforward extrapolation of the Bernstein scheme. However, the variability of the predicted time series is still high. A natural resolution would be to use a smoothing technique similar to the one presented in [29]. Yet the algebraic predictor used in [29] is now replaced by the improved Bernstein predictor scheme. Moreover, the target function used for the evolutionary optimization in [29] needs to be completely re-designed.

3.3. Bernstein polynomial prediction with mixed smoothing (BPPMS)

An evolutionary scheme is used in [29] to identify the algebraic skeleton sequence in the observation window of the predicted time series by removing the unknown additive noise. The idea is based on the assumption that the time series comprises some sort of deterministic skeleton describing the dynamics of the time series which is contaminated by the additive noise.

Figure 1: Schematic diagram illustrating the proposed method of prediction where thick dots denote the original time series; stars denote the corrected time series; B_{n+1} the improved Bernstein forecasting for the original time series; B^{ε}_{n+1} the improved Bernstein forecasting for the corrected time series, and MA_{n+1} the moving average forecasting for the corrected time series.

Let us denote $\tilde{x}_k = x_k - \varepsilon_k; k = 0,1,2,\ldots$ as the corrected values of the sequence (ε_k are unknown corrections). The F-measure as in [49] becomes the fitness measure of the GA that identifies predictive patterns in the sequence of events [50].

The F-measure consists of two parts that embody different objectives: PRECISION, the model precision, requires from the model that it faithfully reconstruct the last known time series values and RECALL requires that the prediction repeats past dynamical behavior:

$$F\left(\varepsilon_0, \varepsilon_1, \ldots, \varepsilon_n\right) = \frac{\left(\gamma^2 + 1\right) \cdot PRECISION \cdot RECALL}{\gamma^2 \cdot PRECISION + RECALL}. \tag{15}$$

In equation (15) the value γ controls the relative importance of precision to recall. If $\gamma = 0$ then the fitness function evaluates only the PRECISION part. If $\gamma = \infty$ then the fitness function evaluates the RECALL values only. In our case we build PRECISION and RECALL functions in such a way that the minimal value of the fitness function is reached when the corrections are small and the improved Bernstein extrapolation (through points $\tilde{x}_k$) is close to the moving average prediction (also based on $\tilde{x}_k$):

$$PRECISION = \frac{1}{n-2} \sum_{i=1}^{n-1} \left| x_i - \hat{x}_i \right|, \tag{16}$$

$$RECALL = \alpha \sum_{i=0}^{n} |\varepsilon_i| + \left| MA_{n+1} - B^{\varepsilon}_{n+1} \right|; \alpha > 0; \tag{17}$$

where an array ε_0, ε_1, ..., ε_n represents near-optimal corrections of the original time series; the MA_{n+1} notation stands for the moving average through the last s time series values; B^{ε}_{n+1} stands for the improved Bernstein extrapolation through last $n+1$ values of $\tilde{x}_k$; parameter α determines the penalty proportion between the sum of weighted corrections and the difference of forecasts based on MA_{n+1} and B^{ε}_{n+1}. Note that we do not use weighting of corrections (the closer is the element to present time – the higher its weight) in Eq. (17) both not to add to the complexity of the algorithm and because such a weighting is more suitable to the moving average side of the technique [51]. The synchronization of weightings between such corrections and the MA is a topic of future research. Fig. 1 involves the aforementioned concepts and offers clarity through diagram.

It is the same improved Bernstein construction of Eq. (14) that estimates inner values of the time series $\hat{x}_0$,$\hat{x}_1$,..., $\hat{x}_n$ in Eq (16), the PRECISION part:

$$\binom{n+1}{0}(1-t_k)^{n+1} t_k^0 x_0 + \binom{n+1}{1}(1-t_k)^n t_k x_1 + \ldots + \binom{n+1}{n}(1-t_k) t_k^n x_n + \binom{n+1}{n+1}(1-t_k)^0 t_k^{n+1} x_{n+1} = x_k, t_k = \frac{k}{n}, k = 0,1,,\ldots, n. \tag{18}$$

Thus,

$$\hat{x}_k = \frac{\sum_{i=0}^{n} \binom{n+1}{i}(1-t_k)^{n+1-i} t_k^i \tilde{x}_i + t_k^{n+1}\hat{x}_{n+1} - \binom{n+1}{k}(1-t_k)^{n+1-k} t_k^k \tilde{x}_k}{\left(1 - \binom{n+1}{k}(1-t_k) t_k^k\right)}, \quad t_k = \frac{k}{n}, k = 0,,1,\ldots, n. \tag{19}$$

Note that: $\hat{x}_0 = \tilde{x}_0$, $\hat{x}_n = \tilde{x}_n$, and only for this reason are inner interpolated values $\hat{x}_0, \hat{x}_1,\ldots, \hat{x}_n$ included in Eq. (16). The overarching philosophy of the proposed method is to determine a near-optimal skeleton sequence, i.e., the identification of those corrections that are hypothesized to reflect

Table 1: Comparison of fitness function components in different algebraic predictors.

Fitness function component Model	APES	APIS	APMS	BPPMS
Error norm	-	-	+	+
External smoothing	+	+	-	-
Internal smoothing	-	+	+	+
Determinant of Hankel matrices	+	-	-	-
Corrections (algebraic skeleton)	+	+	+	+

a perturbation to the underlying time series, and to extrapolate that sequence such that the predicted element resides near the MA forecast of the non-perturbed sequence.

Table 1 gives differences between the fitness functions of APES [27], APIS [28], APMS [29] and the BPPMS. Such fitness functions involve: a smoothing method; the determinant of the Hankel matrix; an error norm; and the correction to the algebraic skeleton. Note that all fitness functions exploit the latter since this correction is key to all of the aforementioned models.

A model based on the Bernstein polynomials is more flexible as compared to one based on Prony polynomials. We do not need to make an assumption that the time series is a recurrence sequence contaminated by the additive noise, we do not need to reconstruct neither the near-optimal order of the sequence nor the roots of its polynomials. This flexibility allows to describe a much wider class of interpolants, and to produce better forecasts.

3.4. Numerical example

The improved Bernstein prediction with mixed smoothing (BPPMS) is illustrated by a simple numerical example. Let us keep with the same 4 observation values of the 3-rd degree polynomial: $x_0 = 2$, $x_1 = 1.74$, $x_2 = 1.5$, $x_3 = 1.3$ and the extrapolated value $x_4 = 1.17$ for $t = \left\{0, \frac{1}{3}, \frac{2}{3}, 1, \frac{4}{3}\right\}$.

Let us assume the following set of corrections: $\{\varepsilon_0, \varepsilon_1, \varepsilon_2, \varepsilon_3\} = \{0.01, -0.01, -0.005, 0.01\}$. The corrected observations ($\tilde{x}_k = x_k + \varepsilon_k$) read: $\tilde{x}_0 = 2.01$, $\tilde{x}_1 = 1.73$, $\tilde{x}_2 = 1.495$ and $\tilde{x}_3 = 1.31$.

Let us fix the observation window for the moving average algorithm $s = 2$ that yields $\bar{x}_4 = \frac{x_2 + x_3}{2} = 1.40$. Now, $\hat{x}_4$ can be extrapolated according to Eq. (14):

$$\hat{x}_4 = \frac{\binom{4}{0}\left(-\frac{1}{3}\right)^4 \tilde{x}_0 + \binom{4}{1}\left(-\frac{1}{3}\right)^3 \frac{4}{3}\tilde{x}_1 + \binom{4}{2}\left(-\frac{1}{3}\right)^2 \left(\frac{4}{3}\right)^2 \tilde{x}_2 + \binom{4}{3}\left(-\frac{1}{3}\right)\left(\frac{4}{3}\right)^3 \tilde{x}_3}{1 - \left(\frac{4}{3}\right)^4} = 1.2429.$$

Without loss of generality, $\gamma = 1$. According to Eq. (16) and Eq. (17), $PRECISION = 0.2146$ and $RECALL = 0.0921$. Now, the fitness function (Eq. (15)) reads: $F\,(0.01,-0.01,-0.005,0.01) = 0.1289$.

Note that this result is produced by a randomly chosen set of corrections. Although deterministic strategies for optimization of ε_k may be used, full sorting on a regular grid with a very coarse step in a domain $\varepsilon_k = -0.01 : 0.001 : 0.01$, $k = 0$, 1, 2, 3 would already result in 194481 sets of corrections! Evolutionary optimization techniques are needed for identification of the near optimal set of corrections.

The cost function (Eq. (14)) is defined in closed form and the parameters are continuous. It is differentiable almost everywhere in the multi-dimensional space of corrections (except for modulus boundaries). Thus, a plausible approach to the optimization of the cost function are gradient-based techniques but the topology of the cost function of this primitive example comprising the 4-dimensional space of corrections is already very complex corrections are implicitly embedded into precision and recall in an intricate form.

Deterministic optimization leads to the convergence towards a local extrema not only on the boundary of the feasible set, but also on one of the axis of the correction space (where one correction is large and all others tend to zero). For example, the starting point $\{0.01,\ -0.01,\ -0.005,\ 0.01\}$ results in $\{0,\ -0.1,\ 0,\ 0\}$ and the strategy for the selection of the starting point is completely unclear so it appears sensible to exploit evolutionary algorithms to identify near optimal set of corrections.

3.5. Time series forecasting algorithm

The proposed time series forecasting algorithm consists of three parts: pre-processing, one step forward prediction and parameter re-estimation.

A. Pre-processing

The degree n of the Bernstein polynomial and the smoothing parameter s must be identified prior to time series forecasting. Pre-processing involves

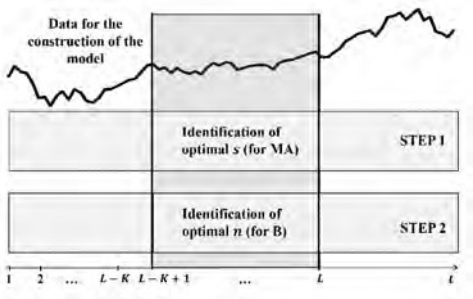

Figure 2: Model preprocessing. Step 1: identify smoothing parameter s for moving average (MA) forecasting; Step 2: identify Bernstein polynomial parameters n.

L time series observations. It consists of two steps: model development and error testing. If K is a number of observations smaller than L, then $L-K$ observations develop the model while K observations compute the Root Mean Square Error of prediction (RMSE). That is (see also Figure 2):

Step 1. Identification of smoothing parameter s for moving average (MA) forecasting: the optimal observation window $s = 1, 2, ..., (L-K)$ is identified for MA forecasting (in terms of RMSE).

Step 2. Identification of the Bernstein polynomial degree n: a one-step forward prediction algorithm evaluates model accuracy within the range of the testing set. RMSE between the predicted $\hat{x}_{L-K+1},\ldots,\hat{x}_L$ and observation values $x_{L-K+1},\ldots,x_L$ is minimized in respect of n.

B. One-step forward prediction algorithm.

Given time series x_0,x_1,x_2,..., n and s.

(1) Compute MA_{n+1} for $(x_k)_{k=n-s+1}^{n+1}$.

(2) Use EA to find a near optimal set of corrections $(\varepsilon_k)_{k=0}^{n}$.

(3) Reconstruct $\hat{x}_{n+1} = B_{n+1}^{\varepsilon}$ using $(\varepsilon)_{k=0}^{n}$.

(4) Shift the observation window by 1 step forward and return to step (1).

C. Parameter re-estimation.

It is clear that the preselected degree of the Bernstein polynomial $n+1$ cannot remain fixed for long time series realizations; the model governing the evolution of the time series must be updated adaptively. We introduce the maximal level of the acceptable absolute error level of the prediction δ as a level-set measure to adapt the forecasting model to dynamical variation of the time series. The degree of the Bernstein polynomial must be changed when absolute errors of the prediction exceed δ. This adaptive selection of the degree of the polynomial reads:

(1) Fix δ.

(2) Execute One-Step forward prediction algorithm and compute $\hat{x}_M$.

(3) Compute the absolute error $E = |x_M - \hat{x}_M|$.

(4) If $E \leq \delta$ move the forecasting window one step ahead and repeat steps (2) and (3). If $E > \delta$ terminate forecasting and run the pre-processing with time series values $\{x_{M-L}, x_{M-L+1}, x_M\}$. Fix new value n and proceed with steps (2) and (3) for time series further values $x_{M+1}, x_{M+2}, \ldots$

Such adaptive time series forecasting can significantly improve forecasting results.

4. Evolutionary algorithms for the identification of a near-optimal set of corrections

4.1. Training and validation of the BPPMS model with DJIA time series

The task of finding a near-optimal set of corrections is well suited to GA. The GA functional and operational parameters are identical to those in [29], namely, the crossover coefficient $\kappa = 0.7$, the mutation parameter $\mu = 0.1$ and the arithmetic crossover parameter $\beta = 0.2$ (Figure 3). Each GA experiment runs 40 generations. The initial population comprises 50 chromosomes randomly generated; values of parameters of genes are limited to the interval $[-0.2; 0.2]$.

Fitness function Eq. (15)-(17) has two parameters (α and γ) predefined for each computational experiment. Training executes 100 parallel independent runs for each set of parameters α and γ over grid $\alpha = \{0.1, 0.2, \ldots, 1\}$, $\gamma = \{0.1, 0.2, \ldots, 1, 1.5, 2, 3, 4, 5, 10, 20\}$ and $n = 5, 10, 15$ and 20 and computes mean of the RMSE of each parallel independent run.

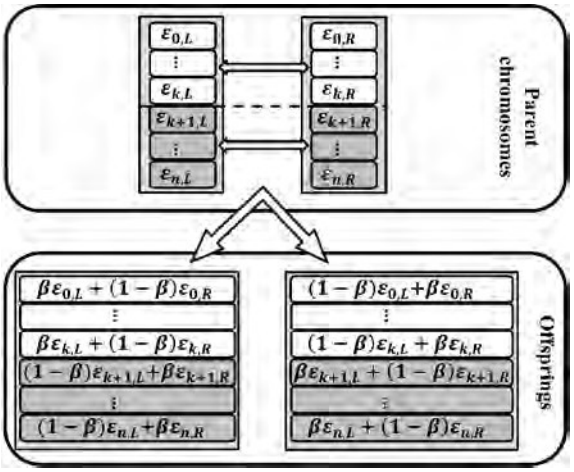

Figure 3: Schematic diagram of the arithmetic crossover procedure (every gene represents a single correction).

All computations use Dow Jones Industrial Average (DJIA) time series (2000-01-01 to 2013-08-27 monthly index observations on of 11 US stocks) [52] normed into the range $[-1; 1]$, pre-processed for $\{x_0,\ldots,x_{30}\}$. The model builds on the initial 21 observations, the RMSE computed for the last 10 observations.

The lowest RMSE value is at $n+1 = 11$, $s = 1$, $\alpha = 0.2$ and $\gamma = 2$. We fix α and γ – but the Bernstein polynomial degree $n+1$ and the smoothing window s chosen for each individual time series. The set of corrections discovered by the GA builds the time series predict candidate.

Examination of the prediction residuals in the training set delivers the required verification of the model accuracy. Residuals must not differ significantly from pure random errors with zero mean [12]. The auto-correlation function (ACF) and the partial auto-correlation function (PACF) are both zero at all lags for the white noise. In practice, it means that all the ACF and PACF values of residuals should be within the confidence bounds. It is also important to check if residuals are normally distributed and uncorrelated. Several diagnostic statistics of the residuals examine the goodness of fit to the model of the historical data. For instance, the Jarque-Bera test [53] is a test of null hypothesis that residuals should have normal distribution and the skewness and kurtosis both equal to zero. We test the randomness of the residuals for the BPPMS model on prediction errors produced during the pre-processing. The residuals of the ACF and the PACF do not show any

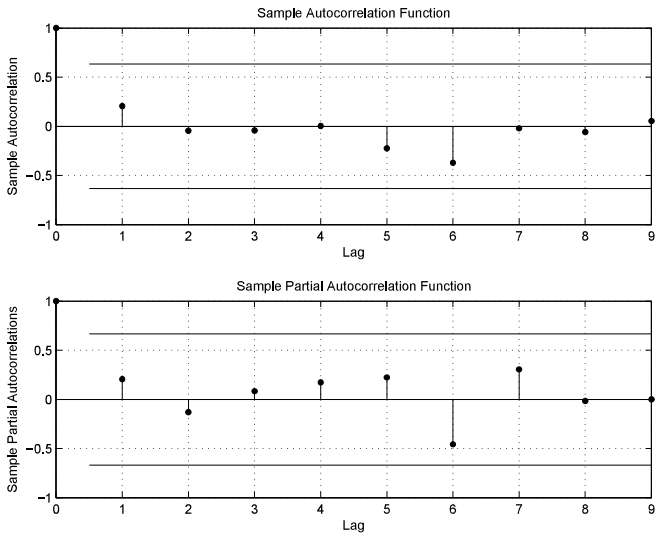

Figure 4: The ACF and PACF plots of residuals of the BPPMS forecasting model for the training set forecasts of the Dow Jones industrial Average time series. The results show that the selected BPPMS model is adequate: the ACF and the PACF values of residuals are within the confidence bounds – the residuals are uncorrelated and can be considered as white noise.

significantly correlated coefficients (Fig. 4). The Jarque-Bera test indicates that the hypothesis that the residuals are normally distributed is not rejected. These results enable us to consider that the selected model is adequate.

5. Model validation based on DJIA time series

The previous section discussed optimal parameter choice. The smallest RMSE values are obtained with $n = 10$, but is using the same degree of Bernstein polynomial most accurate in all considered time series? For example, Fig. 5 (A) demonstrates poor forecasting results in the segment [89; 140]. In this section a strategy of adaptive selection of parameter n predicts different parts of time series. This is based on the idea of algebraic segmentation of short non-stationary time series [54]. The error level δ is the key parameter preselected. Error level δ (preselected in training) and training set values x_{21},x_{22},...,x_{30} are shown in Fig. 5 (A). It follows that an adaptive selection of δ (as detailed in [54]) can improve upon the forecasting with $n = 10$ in segment [89; 140].

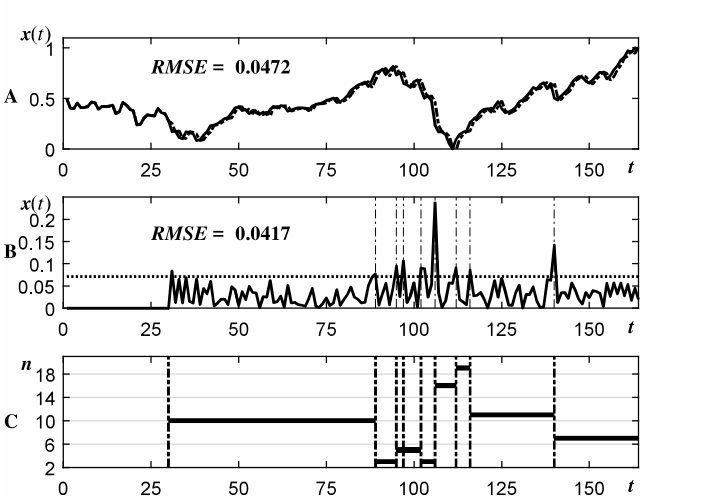

Figure 5: The Dow Jones Industrial Average (DJIA) time series (solid line) and BPPMS forecasts (dashed line) (A); Bernstein polynomial forecasting absolute errors, when polynomial degree is fixed and equal to 11 ($n+1=11$) (B); Bernstein polynomial forecasting absolute errors with adaptive polynomial degree (ABPPMS) for tolerance error level $\delta = 0.07$ (C).

Figure 5 (B) gives results on DJIA time series with adaptive polynomial degree selection. The polynomial degree $n+1$ is recalculated and the set of preceding values is retrained when the forecast error reaches the δ value. Fig. 5 (B) demonstrates that the forecast errors (at $n = 10$) are lower than the $\delta = 0.07$ until x_{89}. As noted previously, we re-train the model at x_{89}. We reset all Bernstein values n = 2,3,...,20 and repeat forecasting for 100 times; the predicted value x_{89} is taken to be the average value of all 100 trials. The n value is set to 3 (the degree of Bernstein polynomial is equal to $n+1=4$). The process is repeated again. As we can see in Fig. 5 (C) – to obtain satisfactory results the polynomial degree has changed 8 times, where forecasting errors were higher than $\delta = 0.07$: at points 89, 95, 97, 102, 106, 112, 116, 140. Respectively, the degree of Bernstein polynomial at relevant segments has changed to values: $n+1=11$ at segment [31, 89], $n+1=4$ at [90,95], $n+1=6$ at [96,97], $n+1=6$ at [98, 102], $n+1=4$ at [103,106], $n+1=17$ at [107, 112], $n+1=20$ at [113, 116], $n+1=12$ at [117, 140], and $n+1=8$ at [141, 164]. The applied strategy improved upon the forecasts and the RMSE value was reduced from $RMSE = 0.0472$ with $n = 10$ to $RMSE = 0.0417$.

Table 2: The comparison of RMSE errors for DJIA time series.

	MA(1)	SES(0.99)	ARIMA(1,1,0)	APES	APIS	APMS	BPPMS	ABPPMS	NAR-NN
RMSE	0.0450	0.0451	0.0536	0.2696	0.0500	0.0446	0.0472	0.0417	0.0480
MAE	0.0321	0.0322	0.0400	0.1189	0.0405	0.0321	0.0293	0.0280	0.0340

5.1. Comparison to other forecasting methods

We continue experiments with DJIA time series. We compare the performance of ABPPMS (Adaptive Bernstein polynomial prediction mixed smoothing) model with MA ($s = 1$), SES ($\alpha = 0.99$), ARIMA(0,1,1), APES ($n = 3$), APIS ($n = 3$, $s = 3$), and APMS ($n = 4$, $s = 1$) models forecasting results. All these models were analysed in [29] and the sets of parameters with lowest RMSE values were preselected on the same training set of time series data values and the forecast is executed from value No 31. Additionally MAE (mean absolute errors) metrics were computed for accuracy analysis. All computational results and best performances of the methods are presented in Table 2. Note that BPPMS prediction errors do not outperform MA (with window $s = 1$), SES ($\alpha = 0.99$) and APMS ($n = 4$, $s = 1$) models. However, the adaptive polynomial selection methodology (ABPPMS) based on error level δ outperforms all these methods. That can be explained by the fact that ABPPMS finds a near-optimal conciliations between the variability of polynomial algebraic extrapolation and the smoothness of the averaging window. On the other hand, one needs to remember that although the search for a best time series forecasting method continues, it is agreeable that no single method will outperform all others in all situations. Computational experiments with other time series are presented in further sections.

The performance of ABPPMS is also compared to nonlinear autoregressive neural network (NAR-NN) [55, 56]. Initially, the optimal architecture of NAR-NN model for DJIA series was determined (4 delays and 7 neurons in the hidden layer) – the criterion for the optimal architecture was the lowest prediction error in the training set. Bayesian regularization training technique [57] was used for training NAR-NN. The training of the network was performed using 90 initial data points of DJIA series (the network was not able to capture the dynamics of the time series from a smaller amount of data points) – 70 % of these points were used for training, 15 % – for validation, and 15 % – for testing (the last $132 - 90 = 42$ data points were not used in the training). One-step ahead DJIA series forecasting with NAR-NN (using all available data points) resulted into $RMSE = 0.0480$ and $MAE = 0.0340$. It appears that ABPPMS outperforms NAR-NN for DJIA series.

Table 3: The assessment of the computational complexity of BPPMS.

The degree of Bernstein polynomial, n	1	2	3	4	5	6	10	15	20
Time, sec	2.66	7.06	19.36	51.16	104.28	129.23	268.94	521.43	847.37

5.2. Computational complexity of BPPMS

ABPPMS is a one step forward prediction method. The model must be updated for every time-step – thus the computational complexity of BPPMS is an important feature to be considered. Moreover, it is important to determine the computational complexity of a problem when GAs are being used [58]. The performance of GAs is usually measured by the number of fitness function evaluations done during the course of a run. For fixed population sizes (the usual case in GA implementations) the number of fitness function evaluations is given by the product of the population size and the number of generations [59].

The computational complexity of BPPMS is assessed by measuring the execution times for one step ahead prediction of DJIA series on an Intel (R) Core TM i5 CPU 2.67 GHz personal computer. An arbitrarily generated population of 50 chromosomes is fixed for all experiments. The crossover coefficient κ is set to 0.7; the mutation parameter μ – to 0.1; the arithmetic crossover coefficient β – to 0.2. Each GA experiment is executed for 40 generations. The Bernstein polynomial degree is incrementally changed from 1 to 20 for every consecutive experiment. The execution times are presented in Table 3.

Computational experiments demonstrate a polynomial (not exponential) relationship between the degree of Bernstein polynomial and the execution time (Fig. 6). Least square approximation yields a parabola with a rather small quadratic coefficient (Fig. 6). That is a rather favorable result in respect to the efficiency of computational implementations of BPPMS.

6. Computational experiments with other time series

6.1. S&P 500 times series forecasting

The S&P 500 is the stock market index of 500 leading US shares. It covers 75 % of US equities with common stock listed on the NYSE or NASDAQ. The time series considered spans the daily index for the period 2013-01-02 to 2013-06-18. The initial training process uses the first 20 values of this time series, with the next 10 values of the time series reserved for model

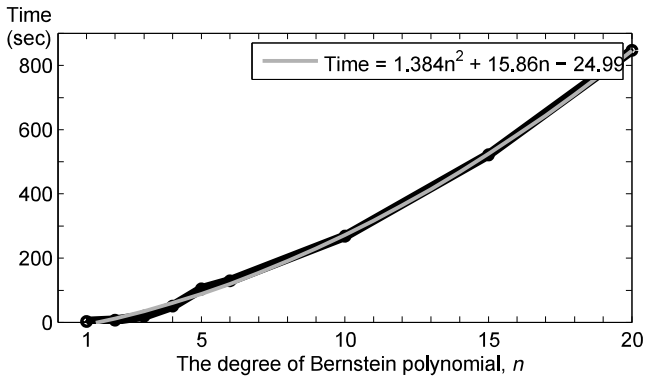

Figure 6: The assessment of the computational complexity of BPPMS.

Table 4: The comparison of RMSE errors for S&P 500 time series.

	MA(1)	SES(0.9)	ARIMA(0,1,1)	APES	APIS	APMS	BPPMS	ABPPMS	NAR-NN
RMSE	0.0477	0.0476	0.0484	0.2266	0.1671	0.0445	0.0487	0.0414	0.0471
MAE	0.0448	0.0445	0.0440	0.1697	0.1720	0.0414	0.0313	0.0302	0.357

validation. A Bernstein polynomial degree of $n+1=5$ (parameter $n=4$) with s=1 attains lowest RMSE. BPPMS forecasts are shown in Fig. 7 (A) (S&P 500 time series values – solid line, BPPMS forecasts – dashed line) with RMSE value equal to 0.0487. Forecasting results with segmentation based on different Bernstein polynomial degree selection reduce the RMSE value to 0.0414. The selected absolute error level $\delta = 0.095$ is exceeded 6 times and for this reason six different training processes are executed to obtain the result, they are: $n = 15$ at segment [38,71]; $n = 6$ at [72,104]; $n = 12$ at [105,117]; $n = 14$ at [118,129], $n = 14$ at [130,165], and $n = 6$ at [165,168]. The ABPPMS forecasting results of different degree Bernstein polynomials and appropriate segments are illustrated in Fig. 7 (B) and (C).

A comparison to other methods of prediction of RMSE shows that BPPMS outperforms only some of the algebraic methods. As an example, it outperforms APES ($n = 3$) and APIS ($n = 3,\ s = 3$), but concede to MA(1), SES($\alpha = 0.9$), ARIMA(0,1,1), APMS($n = 3,\ s = 1$). BPPMS also outperforms NAR-NN (3 delays, 9 neurons in the hidden layer, first 120 data points from 168 used for training). But it is obvious that ABPPMS obtains the lowest RMSE value (Table 3).

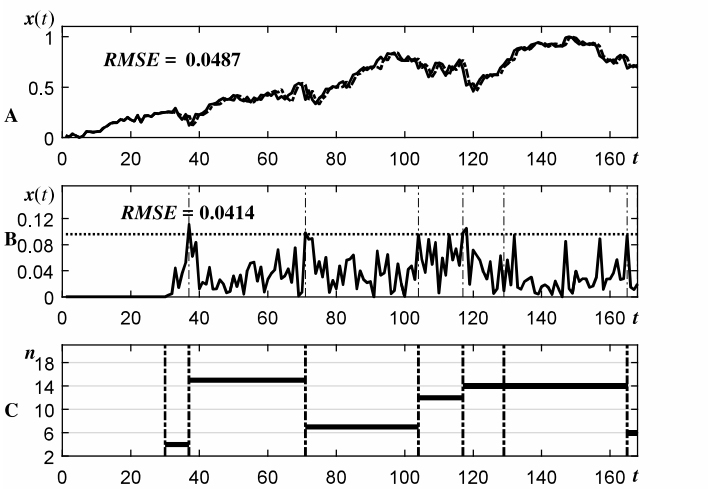

Figure 7: S&P 500 time series (solid line) and BPPMS forecasts (dashed line) (A); Bernstein polynomial forecasting absolute errors when polynomial degree is fixed and equal to 5 ($n = 4$) (B); Bernstein polynomial forecasting absolute errors with adaptive polynomial degree (ABPPMS) for tolerance error level $\delta = 0.095$ (C).

6.2. STLFSI times series forecasting

The STLFSI (St. Louis Fed Financial Stress Index) measures the degree of financial stress in the markets. This economic indicator is built from 18 weekly data series that capture some aspects of financial stress and respectively react to it. The time series used spans the index from 1993-12-31 to 2007-12-30.

The STLFSI time series training set yields the lowest RMSE value for a Bernstein polynomial of degree $n + 1 = 8$ with the MA part $s = 1$. To a tolerance level of $\delta = 0.035$ these segments are identified as: $n = 7$ at [31 56], $n = 11$ at [57 76], $n = 10$ at [77 84], $n = 10$ at [85 89], $n = 3$ at [90 93], $n = 6$ at [94 103], $n = 14$ at [104 164], and $n = 15$ at [165 167] (Fig. 8).

Comparison with other methods forecasting RMSE reveals that ABPPMS produces the best results with $RMSE = 0.0228$, although; MA(1), SES(0.99) and APMS methods yield better results than BPPMS (see Table 5). NAR-NN (4 delays, 8 neurons in the hidden layer, 190 data points from 236 used for training) outperforms BPPMS but does not outperform ABPPMS (Table 5).

6.3. Odonovan times series forecasting

The Odonovan1.dat time series reports on 70 consecutive yields of batch chemical processes [30]; all 70 are normalized to interval [0; 1]. The initial

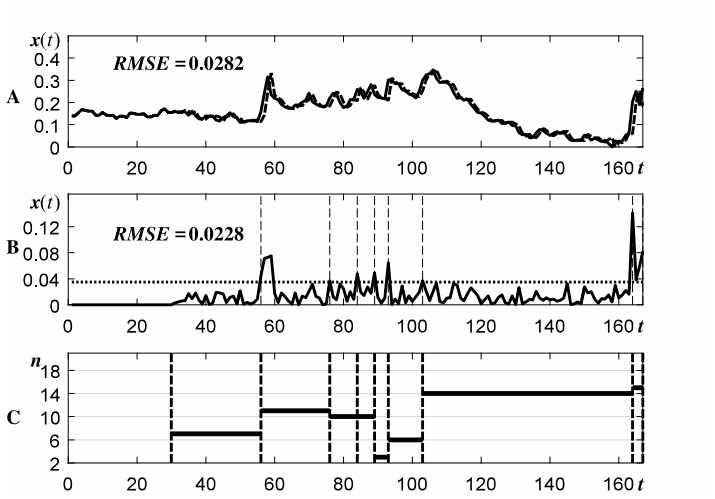

Figure 8: STLFSI time series (solid line) and BPPMS forecasts (dashed line) (A); Bernstein polynomial forecasting absolute errors, when polynomial degree is fixed and equal to 8 ($n = 7$) (B); Bernstein polynomial forecasting absolute errors with adaptive polynomial degree (ABPPMS) for tolerance error level $\delta = 0.035$ (C).

Table 5: The comparison of RMSE errors for STLFSI time series.

	MA(1)	SES(0.99)	ARIMA(0,1,1)	APES	APIS	APMS	BPPMS	ABPPMS	NAR-NN
RMSE	0.0268	0.0268	0.0311	0.5098	0.0324	0.0276	0.0282	0.0228	0.0282
MAE	0.0207	0.0208	0.0251	0.2226	0.0248	0.0218	0.0225	0.0135	0.0186

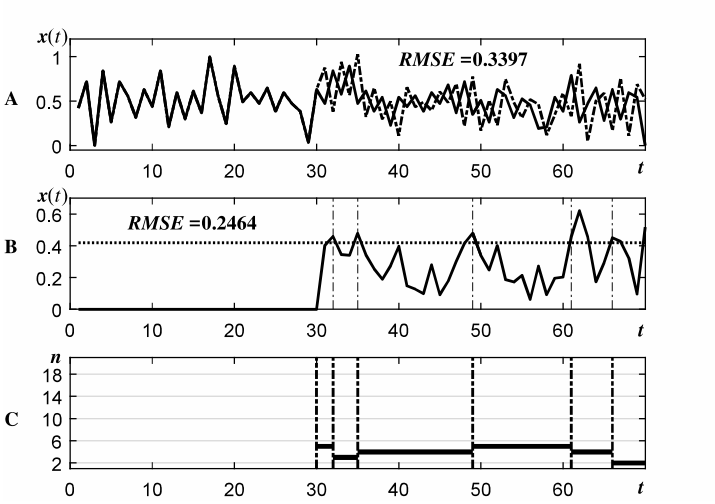

Figure 9: Odonovan time series (solid line) and BPPMS forecasts (dashed line) (A); Bernstein polynomial forecasting absolute errors, when polynomial degree is fixed and equal to 6 ($n = 5$) (B); Bernstein polynomial forecasting absolute errors with adaptive polynomial degree (ABPPMS) for tolerance error level $\delta = 0.41$ (C).

Table 6: The comparison of RMSE errors for Odonovan time series.

	MA(2)	SES(0.1)	ARIMA(1,0,0)	APES	APIS	APMS	BPPMS	ABPPMS	NAR-NN
RMSE	0.1869	0.1758	0.1819	0.7668	0.1728	0.1855	0.3397	0.2464	0.1582
MAE	0.3695	0.3272	0.3266	1.0516	0.3326	0.3587	0.1759	0.1687	0.1214

training process delivers the lowest RMSE value. This equals 0.3397 for a Bernstein polynomial of degree $n + 1 = 6$ and MA parameter set to $s = 4$. The segmentation procedure uses an error level of $\delta = 0.41$. This yields 6 different segments: $n = 5$ at [31,32], $n = 3$ at [33, 35] , $n = 4$ at [36,49], $n = 5$ at [50,61], $n = 4$ at [62,66], and $n = 2$ at [67,70]. This procedure reduced RMSE error to value 0.2464 (Fig. 9). However, NAR-NN (2 delays, 4 neurons in the hidden layer, 50 data points from 100 used for training) outperforms both BPPMS and ABPPMS (Table 6).

Given that BPPMS and ABPPMS are only superior to the APES method, the proposed predictor appears not best suited at forecasting the Odonovan time series.

6.4. Montgome times series forecasting.

The Montgome8.dat time series consists of 100 positive elements normalized to the unit interval [0; 1]. Over the training set, the smallest RMSE are obtained with BPPMS parameters set to $n = 4$ and $s = 4$. In forecas-

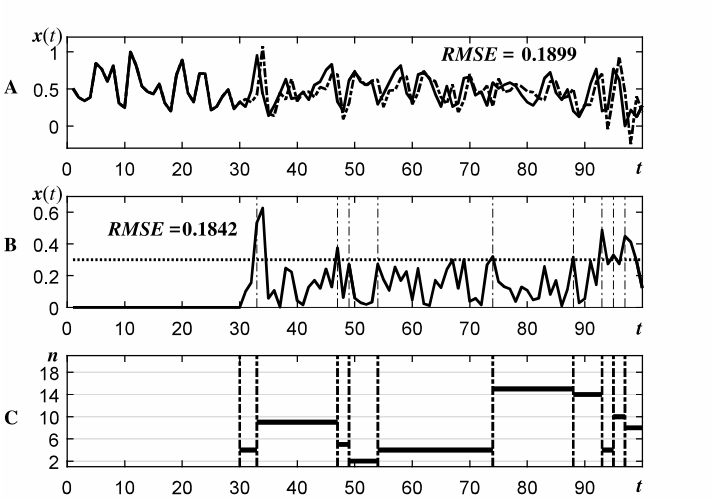

Figure 10: Montgome time series (solid line) and BPPMS forecasts (dashed line) (A); Bernstein polynomial forecasting absolute errors, when polynomial degree is fixed and equal to 5 ($n = 4$) (B); Bernstein polynomial forecasting absolute errors with adaptive polynomial degree (ABPPMS) for tolerance error level $\delta = 0.3$ (C).

Table 7: The comparison of RMSE errors for Montgome time series.

	MA(2)	SES(0.1)	ARIMA(1,0,0)	APES	APIS	APMS	BPPMS	ABPPMS	NAR-NN
RMSE	0.2152	0.2094	0.2106	0.6589	0.2129	0.3407	0.1899	0.1842	0.1752
MAE	0.2510	0.2515	0.2545	0.6054	0.2529	0.3377	0.1234	0.1215	0.1398

ting, $RMSE = 0.1899$ for BPPMS. For ABPPMS, and $RMSE = 0.1842$ with segments: $n = 4$ at [31,33], $n = 9$ at [34,47], $n = 5$ at [48,49], $n = 2$ at [50,54], $n = 4$ at [55,74], $n = 15$ at [75,88], $n = 14$ at [89,93], $n = 4$ at [94,95], $n = 10$ at [96,97], and $n = 8$ at [98,100] with error level $\delta = 0.3$ (Fig. 10).

It can be noted that both BPPMS and ABPPMS methods outperform all other methods (including NAR-NN with 2 delays, 4 neurons in the hidden layer, 50 data points from 100 for training) in respect of MAE values (Table 7).

7. Concluding remarks

A different approach to algebraic prediction based on Bernstein polynomials with mixed smoothing procedure (BPPMS) is proposed in this paper. Extrapolations of Bernstein-type skeletons are successfully employed to forecast real world time series. Evolutionary algorithms are used to identify a near optimal algebraic skeleton and to conciliate its variability with the

smoothness on moving averaging. Moreover, models of Bernstein-type skeletons are selected using a fully automatic and adaptive procedure. The current degree of Bernstein polynomial is changed if the measure representing the error of prediction exceeds a preset level. Computational experiments with real world time series indicate that such an approach can significantly improve forecasting for many examples of non-stationary time series. Moreover, the ability to generate a large number of near-optimal algebraic skeleton sequences from a very limited amount of data available in a fixed observation window allows to apply BPPMS for very short time-series.

The degree n of the Bernstein polynomial and the smoothing parameter s are identified for every individual time series. However, parameters α (the balance between the corrections and the smoothness of the prediction) and γ (the relative importance of precision to recall) are fixed only for DJIA series and used for all other time series in our paper. In fact, the forecasting results for other time series could be improved if only parameters α and γ would be preselected individually for every time series (what would require thorough computations in the preprocessing stage).

It appears that time series prediction techniques based on adaptive identification of Bernstein-type algebraic skeletons can improve forecasting results produced by Prony polynomials. This result is not astonishing as the algebraic variability generated by Bernstein polynomials is much wider compared to Prony polynomials. However, the ability of an adaptive selection of the Bernstein-type model can serve as an effective computational tool for time series segmentation applications - what is a definite objective of future research.

Finally, it can be noted that the presented approach (based on the construction of near-optimal perturbations) in principle could be adapted for a completely different time series analysis problem – filtering out the additive noise from the underlying unknown skeleton sequence. But the development of such a filter is a completely different problem, and it also remains a definite objective of future research.

Acknowledgments

Financial support from the Lithuanian Science Council under project No. MIP-078/2015 is acknowledged.

References

[1] E. Parras-Gutierrez, V.M. Rivas, M. Garcia-Arenas, M.J. del Jesus, Short, medium and long term forecasting of time series using the L-Co-R algorithm, Neurocomputing 128(2014)433–446.

[2] M.J. Tao, Y.Z. Wang, Q.W. Yao, J. Zou, Large Volatility Matrix Inference via Combining Low-Frequency and High-Frequency Approaches, Journal of the American Statistical Association 106(495) (2011) 1025–1040.

[3] Q. Cao, K.B. Leggio, M.J. Schniederjans, A comparison between Fama and Frenchś model and artificial neural networks in predicting the Chinese stock market, Computers & Operations Research 32(10) (2005) 2499–2512.

[4] K. Lam, K.C. Lam, Forecasting for the generation of trading signals in financial markets, Journal of Forecasting 19(1) (2000) 39–52.

[5] M. Podsiadlo, H. Rybinski, Financial time series forecasting using rough sets with time-weighted rule voting, Expert Systems With Applications 66 (2016) 219–233.

[6] J.W. Taylor, L.M. de Menezes, P.E. McSharry, A comparison of univariate methods for forecasting electricity demand up to a day ahead, International Journal of Forecasting 22(1) (2006) 1–16.

[7] G.A. Darbellay, M. Slama, Forecasting the short-term demand for electricity – Do neural networks stand a better chance?, International Journal of Forecasting 16(1) (2000) 71–83.

[8] F.J. Nogales, J. Contreras, A.J. Conejo, R. Espínola, Forecasting Next-Day Electricity Prices by Time Series Models, IEEE Transactions on power systems 17(2) (2002) 342–348.

[9] S. Kim, D.H. Shin, Forecasting short-term air passenger demand using big data fromsearch engine queries, Automation in Construction 70 (2016) 98–108.

[10] R. Brown, Statistical Forecasting for Inventory Control, McGraw–Hill, New York, 1959.

[11] P. Winters, Forecasting sales by exponentially weighted moving averages, Manage. Sci. 6(3) (1960) 324–342.

[12] G.E.P. Box, G.M. Jenkins, Time Series Analysis: Forecasting and Control, Holden–Day, San Francisco, 1976.

[13] M. S. Choi, J. A. Park, S. Y. Hwang, Asymmetric GARCH processes featuring both threshold effect and bilinear structure, Statistics & Probability Letters 82(3) (2012) 419–426.

[14] E. Amiri, Forecasting daily river flows using nonlinear time series models, Journal of Hydrology 527 (2015) 1054–1072.

[15] D. Aydin, O. I. Guneri, Time Series Prediction Using Hybridization of AR, SETAR and ARM Models, International Journal of Applied Science and Technology 5(6) (2015) 87–96.

[16] Y. Cai, Forecasting for quantile self-exciting threshold autoregressive time series models, Biometrika 97(1) (2010) 199–208.

[17] D. E. Allen, M. McAleer, S. Peiris, A. K. Singh, Nonlinear Time Series and Neural-Network Models of Exchange Rates between the US Dollar and Major Currencies, Risks 4(1), 7 (2016) doi:10.3390/risks4010007

[18] C. W. S. Chen, Z. Wang, S. Sriboonchitta, S. Lee, Pair trading based on quantile forecasting of smooth transition GARCH models, The North American Journal of Economics and Finance 39 (2017) 38–55.

[19] P. Brockwell, R.Hyndman, On continuous-time threshold autoregression, Int.J. Forecast.8(2)(1992)157–173.

[20] Y. Syu, J. Y. Kuo, Y. Y. Fanjiang, Time Series Forecasting for Dynamic Quality of Web Services: An Empirical Study, The Journal of Systems & Software (2017), doi: 10.1016/j.jss.2017.09.011.

[21] M. Khashei, M. Bijari, A novel hybridization of artificial neural networks and ARIMA models for time series forecasting, Applied Soft Computing 11(2) (2011) 2664–2675.

[22] E. Gomez-Ramirez, K. Najim, E. Ikonen, Forecasting time series with a new architecture for polynomial artificial neural network, Applied Soft Computing 7(4) (2007) 1209–1216.

[23] A. Bahrammirzaee, A comparative survey of artificial intelligence applications in finance: artificial neural networks, expert system and hybrid intelligent systems, Neural Computing & Applications 19(8) (2010) 1165–1195.

[24] D. H. Wolpert, W. G. Macready, No Free Lunch Theorems for Optimization, IEEE Transactions on Evolutionary Computation 67(1) (1997) 67–82.

[25] D. H. Wolpert, What the no free lunch theorems really mean, Ubiquity 2013 (2013) DOI: 10.1145/2555235.2555237.

[26] S. Beer, Decision and Control: The Meaning of Operational Research and Management Cybernetics Stafford Beer Classic Library, John Wiley & Sons, Ltd. (UK) (1994) isbn10 | asin: 0471948381. print isbn13: 9780471948384. ebook isbn13: 9780585224848.

[27] M. Ragulskis, K. Lukoseviciute, Z. Navickas, R. Palivonaite, Short-term time series forecasting based on the identification of skeleton algebraic sequences, Neurocomputing 74 (2011) 1735–1747.

[28] R. Palivonaite, M. Ragulskis, Short-term time series algebraic forecasting with internal smoothing, Neurocomputing 127 (2014) 161–171.

[29] R. Palivonaite, K. Lukoseviciute, M. Ragulskis, Short-term time series algebraic forecasting with mixed smoothing, Neurocomputing 171 (2016) 854–865.

[30] R. J. Hyndman, Time Series Data Library, http://datamarket.com/data/list/? q=provider:tsdl.

[31] M. I. Bhatti, P. Bracken, Solutions of differential equations in a Bernstein polynomial basis, J. Comput. Appl. Math. 205 (2007) 272–280.

[32] A.V.M.V.L. Kurakin, A.S. Kuzmin, A.A. Nechavev, Linear complexity of polinear sequences, J. Math. Sci. 76 (1995)2793–2915.

[33] Z. Navickas, L. Bikulciene, Expressions of solutions of ordinary differential equations by standard functions, Mathematical Modeling and Analysis 11 (2006) 399–412.

[34] R. T. Farouki, The Bernstein polynomial basis: a centennial retrospective. Comput Aided Geom 29(6) (2012) 379–419.

[35] J.M. Pena, Shape Preserving Representations in Computer-Aided Geometric Design, Nova Science Publ., New York, 1999.

[36] S.M. Curtis, S.K. Ghosh, A variable selection approach to monotonic regression with Bernstein polynomials, J. Appl. Stat. 38 (2011) 961–976.

[37] M. Osman, S.K. Ghosh, Nonparametric regression models for right-censored data using Bernstein polynomials, Comput. Statist. Data Anal. 56 (2012) 559–573.

[38] J. Wang, S.K. Ghosh, Shape restricted nonparametric regression with Bernstein polynomials, Comput. Statist. Data Anal. 56 (2012) 2729–2741.

[39] M.C. Edwards, R. Meyer, N. Christensen, Bayesian semiparametric power spectral density estimation with applications in gravitational wave data analysis, Phys. Review D 92(2) (2015) DOI: 10.1103/PhysRevD.92.064011.

[40] A.F.R.L. de Hierro, J. Martınez-Moreno, C. Aguilar Penac, C. Roldan, A fuzzy regression approach using Bernstein polynomials for the spreads: Computational aspects and applications to economic models, Mathematics and Computers in Simulation 128 (2016) 13–25.

[41] G.J. Babu, A.J. Canty, Y.P. Chaubey, Application of Bernstein Polynomials for smooth estimation of a distribution and density function, Journal of Statistical Planning and Inference, 105(2) (2002) 377–392.

[42] G.J. Babu, Y.P. Chaubey, Smooth estimation of a distribution and density function on a hypercube using Bernstein polynomials for dependent random vectors, Statistics & Probability Letters 76(9) (2006) 959–969.

[43] A. Leblanc, On the boundary properties of Bernstein polynomial estimators of density and distribution functions, J. Statist. Plann. Inference 142 (2012) 2762–2778.

[44] D. Howard, J. Kolibal, A. Brezulianu, Engineering presentation of the stochastic interpolation framework and its applications, Soft Comp. 15 (2011) 79–87.

[45] G. M. Phillips, Interpolation and Approximation by Polynomials, New York: Springer-Verlag, 2003, 247–290.

[46] Cong Wang, Hongli Zhang, Wenhui Fan, Xiaochao Fan, A new wind power prediction method based on chaotic theory and Bernstein Neural Network, Energy 117(1) (2016) 259–271.

[47] C.R. Giardina, Band-limited signal extrapoloation by truncated Bernstein polynomials, J. of Mathematical Analysis and Applications 104 (1984) 264–273.

[48] J. Li, J. Lin, An algorithm of local prediction for chaotic sequences with variable frame length, J. of Electronics (China) 29 (2012) 345–352.

[49] C.J. van Rijsbergen, Information Retireval, Butterworths, London, 1979.

[50] G.M. Weiss, Timeweaver: a Genetic Algorithm for Identifying Predictive Patterns in Sequences of Events, In Proceedings of the Genetic and Evolutionary Computation Conference (1999) 718–725.

[51] M. Landauskas, Z. Navickas, A. Vainoras, M. Ragulskis, Weighted moving averaging revisited: an algebraic approach, Comp. Appl. Math. (2016) doi:10.1007/s40314–016–0309–9.

[52] Federal reserve bank of st. Louis http://research.stlouisfed.org/fred2/series/ STLFSI/downloaddata.

[53] C.M. Jarque, A.K. Bera, A Test for Normality of Observations and Regression Residuals, International Statistical Review. 55(2) (1987) 163–172.

[54] R. Palivonaite, K. Lukoseviciute, M. Ragulskis, Algebraic segmentation of short nonstationary time series based on evolutionary prediction algorithms, Neurocomputing 121 (2013) 354–364.

[55] K. Benmouiza, A. Cheknane, Forecasting hourly global solar radiation using hybrid k-means and nonlinear autoregressive neural network models, Energy Conversion and Management 75 (2013) 561–569.

[56] I.S. Markham, T.R. Rakes, The effect of sample size and variability of data on the comparative performance of artificial neural networks and regression, Comput Oper Res 25 (1998) 251–263.

[57] F. D. Foresee, M. T. Hagan, Gauss-Newton approximation to Bayesian learning, Proceedings of the International Joint Conference on Neural Networks 3 (1997) 1930–1935.

[58] B. Rylander, J. Foster, Computational complexity and genetic algorithms, Proc. of the World Science and Engineering Society's Conference on Soft Computing, Advances in Fuzzy Systems and Evolutionary Computation (2001) 248–253.

[59] F.G. Lobo, D.E. Goldberg, M. Pelikan, Time complexity of genetic algorithms on exponentially scaled problems, In Proceedings of the 2nd Annual Conference on Genetic and Evolutionary Computation (GECCO'00), L. Darrell Whitley, David E. Goldberg, Erick Cantú-Paz, Lee Spector, and Ian C. Parmee (Eds.), Morgan Kaufmann Publishers Inc., San Francisco, CA, USA (2000) 151–158.