Bio-Medical Materials and Engineering 24 (2014) 3557–3568
DOI 10.3233/BME-141182
IOS Press

An efficient estimator of Hurst exponent through an autoregressive model with an order selected by data induction

Yen-Ching Chang*
Department of Medical Informatics, Chung Shan Medical University and Department of Medical Imaging, Chung Shan Medical University Hospital, Taichung City 40201, Taiwan, ROC

Abstract. The discrete-time fractional Gaussian noise (DFGN) has been proven to be a regular process. Therefore, an autoregressive (AR) model of an infinite order can describe DFGN based on Wold and Kolmogorov theorems. A fast estimation algorithm on the Hurst exponent of DFGN or discrete-time fractional Brownian motion (DFBM) has been proposed, but the algorithm did not consider the order selection of AR model. Recently, a Hurst exponent estimator based on an AR model with six existing methods of order selection has been proposed to raise the accuracy of estimating the Hurst exponent. Although the estimation accuracy has been confirmed to be better than the one without order selection, the estimator still requires computing all parameter sets through the Levinson algorithm. In order to lower computational cost, this paper proposes an efficient method of order selection, simply called data induction, which uses simulation data to induce an appropriate threshold of terminating the Levinson algorithm before computing all parameter sets. Experimental results show that the proposed data-induction method has a competitive advantage over six existing methods of order selection in terms of lowering computational cost and raising the accuracy.

Keywords: Fractional Brownian motion, fractional Gaussian noise, Hurst exponent, autoregressive, order selection

1. Introduction

Signals of nature and biology usually exhibit a strong long-term correlation [1–7]. These signals can be differentiated by only one indicator, fractal dimension or Hurst exponent, and thus an increasing number of researchers in statistical signal processing and biomedical engineering are attracted to these areas [8–31]. For example, Lundahl et al. [8] proposed a maximum likelihood estimator to estimate the Hurst exponent of fractional Brownian motion (FBM) and further applied it to the texture of the X-ray images of bones. Jennane et al. studied the fractal analysis of bone X-ray tomographic microscopy projections [17]. Peréz et al. used the estimation of Hurst parameter and fractal dimension to analyze the hemogram components [18]. Esgiar et al. used the fractal dimension to separate the normal images from the cancerous [19]. Chang et al. applied the fractal dimension to analyze the dynamics and synchronization of rhythms urodynamics of female Wistar rats [21]. Lee et al. used the fractal dimension

*Corresponding author: Yen-Ching Chang, Department of Medical Informatics, Chung Shan Medical University and Department of Medical Imaging, Chung Shan Medical University Hospital, Taichung City 40201, Taiwan, ROC. Tel.: +886-4-24730022 ext. 12212; Fax: +886-424730022 ext. 11733; E-mail: nicholas@csmu.edu.tw.

of a skeletonized cerebral surface to analyze the hemispheric asymmetry [22]. Huang and Lee proposed two feature extraction methods to analyze variations of intensity and texture complexity in regions of interest based on fractal dimension [28]. Lin et al. proposed a set of fractal features to automatically distinguish malignant nodules from benign ones based on the fractional Brownian motion model [29].

In order to tell the difference among signals, a simple and meaningful indicator of features is essential, such as Hurst exponent. It is often an effective and efficient way for researchers to model these signals in order to estimate the indicator. Of all models suitable for description, FBM [4,5] is best known and has been adopted by engineers in the field of statistical signal processing. FBM only contains one parameter, called the Hurst exponent; its value is finite between 0 and 1; most importantly, it can characterize a wide range of biomedical and natural signals. Another model is called fractional Gaussian noise (FGN), which is simply an increment process of FBM, or we can say FBM is simply an accumulation process of FGN.

For practical applications, data are often acquired by sampling continuous signals. The sampled FBM is referred to as discrete-time FBM (DFBM), and the sampled FGN as discrete-time FGN (DFGN). The existing methods for estimating the Hurst exponent of DFBM or DFGN include the variance method [1,8,30], box-counting method [10–13], maximum likelihood estimator (MLE) [8], moving-average (MA) method [15], and autoregressive (AR) method [20].

The variance method is usually not reliable unless the order of linear regression for estimating the Hurst exponent is considered. In order to improve the accuracy and efficiency of the variance method, Chang et al. used the autocorrelation function (ACF) to estimate the Hurst exponent and demonstrated that 4-point linear regression is almost the best in most cases [30].

In theory, the MLE is the most accurate, but it requires a large number of repeated matrix operations during estimation and thus its computational cost is particularly high. Therefore, the MLE is normally not recommended to systems that need quick responses. In order to improve the execution efficiency of the MLE, Liu and Chang [14] proposed the MA method for image texture classification, which uses the Wold decomposition theorem to decompose DFGN into two uncorrelated processes: a regular process and a singular process [32–34]. After calculating the ACFs, the Levinson algorithm [35,36] is used to estimate the parameter sets of the AR model and then the algorithm based on the Wold decomposition theorem [37,38] is utilized to estimate the parameter sets of the MA model.

Chang and Chang discovered that if DFGN is a regular process, the procedure of transforming the AR model into the MA one will be redundant. Based on the fact that the ACFs of DFGN are asymptotic to those of fractionally differenced Gaussian noise [39], Chang and Chang comparatively demonstrated that the DFGN is indeed a regular process [20]. Thus, AR method, a more accurate and efficient estimator than MA method, was further proposed.

In theory, an AR model requires an infinite order to describe a regular process, but the given data are always finite. Even worse, much larger errors are produced by higher-order estimates of the ACFs. Therefore, taking order selection into consideration may be a potential solution to raising the accuracy of estimating the Hurst exponent. Based on this intuition, Chang et al. adopted six existing methods of order selection to estimate the Hurst exponent, and experimental results showed that the AR methods with order selection are more accurate than the ones without order selection [31], but it does not cut down computational cost. Therefore, the main objective of the paper is to raise execution efficiency while remaining or even raising the accuracy. Through some realizations of DFGN or DFBM, an appropriate threshold for each chosen data size is induced to terminate the Levinson algorithm before computing all parameter sets. This efficient method of order selection is simply called data induction.

The rest of this paper is organized as follows: Section 2 briefly describes mathematical preliminaries; Section 3 introduces the novel idea of data induction for order selection; then Section 4 discusses experimental results; and finally, Section 5 concludes the paper with some facts.

2. Mathematical preliminaries

This section will introduce some related terminologies and notations for the rest of the paper. Because the Levinson algorithm is important for implementation and comparison, its procedure is completely provided. The notation $\{x(t), t \in \boldsymbol{R}\}$ is used to denote a continuous-time random process, and $\{x[n], n \in \boldsymbol{Z}\}$ a discrete-time process.

2.1. FBM and its variants

FBM is a generalized Brownian motion, first officially proposed by Mandelbrot and Van Ness [1] and denoted by random function $B_H(t,\omega)$, where ω belongs to a sample space Ω, simplified as $B_H(t)$. It can be defined as follows:

$$B_H(0) = b_0,$$
$$B_H(t_2) - B_H(t_1) = \frac{1}{\Gamma(H+1/2)}\left\{\int_{-\infty}^{t_2}(t_2-s)^{H-1/2}\,dB(s) - \int_{-\infty}^{t_1}(t_1-s)^{H-1/2}\,dB(s)\right\}, \tag{1}$$

where H is the Hurst exponent with a value lying between 0 and 1. When H equals to 0.5, FBM will become an ordinary Brownian motion. Because FBM is not a stationary process, i.e., its spectrum is time-dependent, its increment process, called FGN and denoted by $B'_H(t)$, is often considered. FGN is a stationary process, i.e., its spectrum is time-independent, which is expressed as follows:

$$S_{B'_H}(\Omega) = \frac{1}{|\Omega|^{2H-1}}. \tag{2}$$

In real applications, the discrete version of FBM, called DFBM, is considered and denoted by $B_H[n] = B_H(nT_s)$, where T_s denotes the sampling time. The discrete version of FGN, called DFGN, is denoted by $x_H[n] = B_H[n] - B_H[n-1]$, whose ACF, $r_H[k]$, is described as follows:

$$r_H[k] = \frac{V_H}{2}\left(|k+1|^{2H} - 2|k|^{2H} + |k-1|^{2H}\right), \tag{3}$$

where $V_H = \Gamma(1-2H)\cos(H\pi)/H\pi$ [1,9]. The ACF possesses an asymptotical behavior of $k^{2H-2} = k^{-\alpha}$, $\alpha \in (0, 2)$ [5]. Samorodnitsky and Taqqu [5] provided a complete introduction to related topics; Lundahl et al. [8] also summarized some characteristics.

2.2. Levinson algorithm

The Levinson algorithm [35], or Levinson-Durbin recursion [36], is an efficient way of solving the Yule-Walker equation or the Wiener-Hopf equation. The algorithm can compute the parameter sets for all lower-order AR models fitting to the ACFs. If the AR model is finite and free of noise, the correct model order and its parameter sets can be derived by this algorithm according to the minimum variation of the prediction error power. However, if the AR model order is infinite or the model is finite but contaminated by noise, the algorithm will usually need other tools to choose correct or appropriate order. Therefore, the data induction for order selection described in the following section is adopted.

For later needs for notation, comparison and implementation, the Levinson algorithm is summarized as follows. The Levinson algorithm recursively computes the parameter sets $\{a_1[1], \rho_1\}$, $\{a_2[1], a_2[2], \rho_2\}$, …, $\{a_p[1], a_p[2], \ldots, a_p[p], \rho_p\}$. The final set at order p is the desired solution of the Yule-Walker equations. If $x[n]$ is an AR of order p, then $a_p[i] = a[i]$ for $i = 1, 2, \ldots, p$ and $\rho_p = \sigma^2$, where σ^2 is the excitation noise variance. The algorithm is first initialized by

$$a_1[1] = -\frac{r_{xx}[1]}{r_{xx}[0]} \tag{4}$$

and

$$\rho_1 = \left(1 - a_1^2[1]\right) r_{xx}[0], \tag{5}$$

where r_{xx} is the ACF of $x[n]$, and then the following parameter sets are computed by

$$a_k[k] = -\frac{r_{xx}[k] + \sum_{l=1}^{k-1} a_{k-1}[l] r_{xx}[k-l]}{\rho_{k-1}}, \tag{6}$$

$$a_k[i] = a_{k-1}[i] + a_k[k] a_{k-1}[k-i], \; i = 1, 2, \ldots, k-1, \tag{7}$$

and

$$\rho_k = \left(1 - a_k^2[k]\right) \rho_{k-1}, \tag{8}$$

for $k = 2, 3, \ldots, p$.

3. Data induction for order selection and its potential application

Since AR method was shown to be superior to MA method in accuracy and efficiency [20], the order of AR model has been always chosen as the maximum possible order, i.e., the data size minus one. Theoretically, the higher the model order is, the closer to a regular process it is. Practically, larger errors are prone to occur at higher-order estimates because the Hurst exponent estimation through an AR model highly depends on estimated ACFs that are less accurate at higher orders. As a consequence, an AR model of higher order does not improve estimation accuracy; instead, it easily exerts adverse effects on the autoregressive power spectrum estimation and finally leads to worse estimates of the Hurst exponent. The trade-off between theoretical principles and practical limitations is usually intricate. In order to effectively handle the dilemma, a Hurst exponent estimator based on AR power spectrum estimation with order selection has been proposed [31]. Although the existing methods of order selection can work well, they still need to compute all the parameter sets by using the Levinson algorithm in order to find out the best. In order to efficiently select an appropriate order of an AR model, a novel method induced by simulation data, simply called data induction, will be studied

The logic of the data induction for order selection is simple. First, it needs to select a standard curve relatively suitable for different data sizes, where the ordinate of the standard curve is the Hurst exponent and the abscissa is the slope of power spectral densities versus frequencies in a logarithmically scaled way. Second, it needs to find out appropriate thresholds for different data sizes to terminate the Levinson algorithm during the execution.

In this paper, standard curves were obtained by using the true ACFs, and thus the higher the model order is, the closer to a true model it is. Unfortunately, an infinite order cannot be modeled in a finite time-space. Therefore, a standard curve was established in the previous work [31] based on a data size of 1000, 99 points of Hurst exponents beginning from 0.01 to 0.99 with increment 0.01, and a curve-fitting polynomial of degree 4. Through experimental testing, a standard curve based on a data size of 500, 99 points of Hurst exponents, and a curve-fitting polynomial of degree 3 is relatively suitable for different data sizes. It can be expressed by the following equation

$$\hat{H} = c_3 \hat{s}^3 + c_2 \hat{s}^2 + c_1 \hat{s}^1 + c_0, \tag{9}$$

where $c_0 = 0.4990$, $c_1 = -0.4355$, $c_2 = 0.0441$, and $c_3 = 0.0236$.

For appropriate thresholds for different data sizes, three hundred realizations of white Gaussian noise for six data sizes, i.e., $N = 50$, 100, 200, 400, 800, 1600, were generated. Then, the algorithm proposed by Lundahl et al. [8] over $H = 0.1$, 0.2, ..., 0.9 was used for generating 300 realizations of DFGN. The principle is as follows. Let $\{z[n]\}$ be a white Gaussian noise process with zero mean and unit variance, and $\mathbf{z} = [z[1]\, z[2] \cdots z[N]]^T$ be a realization of white Gaussian noise. The covariance matrix of DFGN is defined as $\mathbf{R} = E[\mathbf{x}\mathbf{x}^T]$ or $[\mathbf{R}]_{ij} = r_H[|i-j|]$. Since the covariance matrix is positive definite, it can be decomposed as $\mathbf{R} = \mathbf{L}\mathbf{L}^T$ using Cholesky decomposition, in which $\mathbf{L}$ is a lower triangular matrix. Then, $\mathbf{x} = \mathbf{L}\mathbf{z}$ is a realization of DFGN, where $\mathbf{x} = [x_H[1]\, x_H[2] \cdots x_H[N]]^T$.

Next, twenty-one potential thresholds $\varepsilon = \hat{\rho}_{k-1} - \hat{\rho}_k$ were provided, where $\hat{\rho}_k$ is the estimate of the white noise variance (prediction error power) for order k via the Levinson algorithm, and then their 21 corresponding orders were calculated. These 21 thresholds were chosen as $\varepsilon = 0.01$, $0.01/2^1$, ..., $0.01/2^{20}$, and their 21 corresponding positions were numbered as 1, 2, …, 21, respectively.

Procedure 1: Calculating positions or thresholds of six data sizes

1. Begin with the first data size ($N = 50$).
2. Begin with the first Hurst exponent ($H = 0.1$).
3. Begin with the first realization of white Gaussian noise, $\mathbf{z}$.
4. Calculate the true ACFs from (3); produce the covariance matrix, $\mathbf{R}$; compute the lower triangular matrix, $\mathbf{L}$; generate the realization of DFGN through the formula $\mathbf{x} = \mathbf{L}\mathbf{z}$.
5. Modify each realization into zero mean, i.e., $\mathbf{x}$ was subtracted by its sample mean.
6. Estimate the biased ACFs for $k = 0, 1, \ldots, N-1$ of the following form

$$\hat{r}[k] = \frac{1}{N}\sum_{n=k+1}^{N} x_H[n]x_H[n-k]. \tag{10}$$

7. Begin with the first potential threshold ($\varepsilon = 0.01$).
8. Estimate the parameter sets by using estimated ACFs via the Levinson algorithm until the terminating condition is satisfied, and record its corresponding order and position.
9. Use Eqs. (11) and (12) in the following to calculate the slope of $\{\log(\hat{S}(f_i))\}$ versus $\{\log(f_i)\}$ over $F_M = \{f_1, f_2, \ldots, f_M\}$, where $f_1 = 0.5/M$, $f_i - f_{i-1} = 0.5/M$ for $i = 2, 3, \ldots, M$ with M being selected as 63:

$$\hat{S}(f) = \hat{\rho}_{\hat{p}} / |\hat{A}(f)|^2, \tag{11}$$

where $\hat{\rho}_{\hat{p}}$ is the estimate of the white noise variance (prediction error power) and

$$\hat{A}(f) = 1 + \hat{a}[1]\exp\{-j2\pi f\} + \cdots + \hat{a}[\hat{p}]\exp\{-j2\pi f\hat{p}\}. \tag{12}$$

where $\hat{p}$ is the estimated order of the AR model.

10. Estimate the H through the standard curve, represented by Eq. (9).
11. If the final potential threshold (totally, twenty-one thresholds) was reached, then go to next, else execute the next threshold.
12. If the final realization (totally, three hundred realizations) was reached, then go to next, else execute the next realization.

Table 1

Positions and thresholds of 6 data sizes

	N = 50	N = 100	N = 200	N = 400	N = 800	N = 1600
Position	1	5	9	11	12	14
Threshold	0.01	6.25E-04	3.91E-05	9.77E-06	4.88E-06	1.22E-06

13. If the final Hurst exponent (totally, nine Hurst exponents) was reached, then go to next, else execute the next H.
14. Calculate the averages of mean-squared errors between the estimated H and true H over 300 realizations and nine Hurst exponents.
15. Find out the minimum of 21 thresholds and record the position or its corresponding threshold.
16. If the final data size (totally, six data sizes) was reached, then terminate, else execute the next N.

After finishing Procedure 1, the positions or thresholds of six data sizes were derived and listed in Table 1. Obviously, these thresholds seem to be reasonable because they follow the logic that a lower data size terminates quicker than a higher data size.

After determining thresholds of six data sizes, the thresholds of data sizes lying between 50 and 1600 can be determined by Algorithm 1, described as follows.

Algorithm 1: Determining the threshold of a data size between 50 and 1600

1. Give any data size or number, N.
2. If N equals to 50, then choose $\varepsilon = 0.01$ and stop, else execute the next step.
3. Find out all the numbers (N) larger than the set of 50, 100, 200, 400, 800, and 1600, and then select the largest number, denoted as N_l. Suppose the corresponding threshold and position are denoted as T_l and P_l, respectively, the next number is denoted as N_{l+1}, and its corresponding threshold and position are denoted as T_{l+1} and P_{l+1}, respectively.
4. Calculate the ratio, $r = (N - N_l)/N_l$, where $0 < r \le 1$.
5. Calculate the threshold as $\varepsilon = T_l / 2^{(P_{l+1} - P_l) \times r}$.

For examples, if $N = 100$, then $N_l = 50$, $T_l = 0.01$, $P_l = 1$, $T_{l+1} = 0.01/2^4$, $P_{l+1} = 5$, $r = 1$, and $\varepsilon = 0.01/2^4$; if $N = 150$, then $N_l = 100$, $T_l = 0.01/2^4$, $P_l = 5$, $T_{l+1} = 0.01/2^8$, $P_{l+1} = 9$, $r = 0.5$, and $\varepsilon = 0.01/2^6$.

When the data size is smaller than 50 or larger than 1600, the established thresholds can be extended to determine the corresponding threshold. According to the thresholding trend of Table 1, the threshold is chosen to multiply by the ratio of 50 to the data size (the ratio is larger than 1) when the data size is smaller than 50; relatively, the threshold is chosen to multiply by the ratio of 1600 to the data size (the ratio is smaller than 1) when the data size is larger than 1600. Algorithm 2 described as follows is recommended for the cases of $N < 50$ and $N > 1600$:

Algorithm 2: Determining the threshold of a data size smaller than 50 or larger than 1600

1. If $N < 50$, then $r = 50/N$ and $\varepsilon = r \times 0.01$.

2. If $N > 1600$, then $r = 1600/N$ and $\varepsilon = r \times 0.01/2^{13}$.

For examples, if $N = 25$, then $r = 2$ and $\varepsilon = 2 \times 0.01$; if $N = 3200$, then $r = 0.5$, and $\varepsilon = 0.5 \times 0.01/2^{13}$. For practical applications, a complete procedure for estimating the Hurst exponent is presented as follows:

Procedure 2: Applying the proposed estimator to a practical signal

1. Give any data size or number, N.
2. Determine the threshold according to Algorithm 1 or 2.
3. Extract a section of size N from a practical signal, $x_H[n]$, of interest.
4. Modify $x_H[n]$ into zero mean.
5. Estimate ACFs for $k = 0,1,\ldots,N-1$ by using (10).
6. Estimate the parameter sets by using estimated ACFs via the Levinson algorithm or from (4)–(8) until the threshold is satisfied or the corresponding order, $\hat{p}$, is found out.
7. Estimate the Hurst exponent, H, by using Steps 9 and 10 of Procedure 1.
8. If necessary, calculate the fractal dimension via $D = 2 - H$ [4].

4. Results and discussion

In order to verify the efficiency of the proposed method, a wider range of Hurst exponents and data sizes were considered, including $H = 0.01$, 0.05, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 0.95, and 0.99 (totally, thirteen Hurst exponents) as well as $N = 128$, 256, 512, 1024, 2048, 4096, 8096, and 16384 (totally, eight data sizes). For each data size, one hundred realizations of white Gaussian noise, called Set 1, were generated by a Gaussian random generator.

On the other hand, in order to show the advantage of the proposed method, simply called data induction (DI), six existing order selection methods, that is, the final prediction error (FPE) [31,35,36, 40,41], Akaike information criterion (AIC) [31,35,36,40,41], criterion autoregressive transfer function (CAT) [31,35,40,41], minimum description length (MDL) criterion [31,35,40,41], residual variance (RV) [31,40,41], and Hannan and Quinn (HQ) [40,41], were adopted for comparison. Tables 2-4 show the experimental results performed on Set 1. Table 2 provides the mean mean-squared errors for eight data sizes and seven methods over 13 Hurst exponents and 100 realizations on Set 1. Table 3 lists the mean orders selected by eight data sizes and seven methods over 13 Hurst exponents and 100 realizations on Set 1. Table 4 indicates the mean standard deviations of orders selected by eight data sizes and seven methods over 13 Hurst exponents and 100 realizations on Set 1.

Table 2

Comparison of methods for the averages of mean-squared errors on Set 1

Methods	$N = 128$	$N = 256$	$N = 512$	$N = 1024$	$N = 2048$	$N = 4096$	$N = 8192$	$N = 16384$
DI	2.77E-03	1.21E-03	7.17E-04	4.06E-04	1.90E-04	**1.08E-04**	**6.25E-05**	**3.31E-05**
FPE	3.49E-03	1.34E-03	8.22E-04	4.34E-04	2.13E-04	1.28E-04	6.46E-05	3.57E-05
AIC	3.49E-03	1.34E-03	8.22E-04	4.34E-04	2.13E-04	1.28E-04	6.46E-05	3.57E-05
MDL	7.05E-03	3.18E-03	2.01E-03	9.98E-04	5.84E-04	3.69E-04	2.01E-04	9.46E-05
CAT	3.50E-03	1.35E-03	8.24E-04	4.36E-04	2.14E-04	1.28E-04	6.45E-05	3.57E-05
RV	**2.76E-03**	**1.18E-03**	**6.31E-04**	**3.63E-04**	**1.82E-04**	1.44E-04	9.50E-05	7.84E-05
HQ	4.83E-03	2.11E-03	1.25E-03	6.28E-04	3.18E-04	2.11E-04	1.01E-04	5.25E-05

Table 3

Comparison of methods for means of orders selected on Set 1

Methods	$N = 128$	$N = 256$	$N = 512$	$N = 1024$	$N = 2048$	$N = 4096$	$N = 8192$	$N = 16384$
DI	8.48	16.31	21.95	25.94	35.36	**38.53**	**44.43**	**53.58**
FPE	3.02	4.51	6.64	9.36	12.57	17.69	25.75	36.29
AIC	3.03	4.52	6.64	9.36	12.57	17.69	25.75	36.29
MDL	1.38	1.87	2.69	3.78	5.00	6.93	9.48	12.86
CAT	2.98	4.43	6.54	9.26	12.51	17.67	25.68	36.20
RV	**6.42**	**10.74**	**17.28**	**29.72**	**55.73**	74.40	138.30	205.29
HQ	1.90	2.70	3.81	5.48	7.33	10.01	14.09	19.50

Table 4

Comparison of methods for standard deviations of orders selected on Set 1

Methods	$N = 128$	$N = 256$	$N = 512$	$N = 1024$	$N = 2048$	$N = 4096$	$N = 8192$	$N = 16384$
DI	4.59	10.58	14.43	16.34	23.24	**22.97**	**23.07**	**25.89**
FPE	2.22	2.60	3.49	4.39	4.98	6.08	8.65	10.53
AIC	2.22	2.61	3.49	4.39	4.98	6.08	8.65	10.53
MDL	0.66	0.78	1.00	1.11	1.15	1.47	1.81	2.11
CAT	2.17	2.53	3.38	4.30	4.95	6.08	8.61	10.40
RV	**4.65**	**7.13**	**10.51**	**20.36**	**35.31**	45.65	87.33	123.38
HQ	1.00	1.34	1.50	1.90	2.06	2.50	3.12	3.89

Chang et al. has shown that the AR methods considering six existing order selection methods are more accurate than the original one without considering order selection [31]. Table 2 further shows that the proposed order selection method, denoted as DI, always performs better than these existing methods except for the RV method. Although the RV method is slightly superior to the proposed method at data sizes of 128, 256, 512, 1024, and 2048, it is much worse than the proposed method at data sizes of 4096, 8192, and 16384, even worse than the FPE, AIC, and CAT methods, which might imply that the RV method will work worse as the data size increases. Thus, the proposed method is more reliable in accuracy than the RV method.

All these six existing methods involve the use of the white noise variance, whereas the proposed method is concerned with the difference of the two successive white noise variances. Table 3 shows that the MDL method tends to select lower orders, leading to larger errors; the second is the HQ method. The FPE, AIC, and CAT are similar to one another. These results are consistent with the formulas summarized by Chang et al. [31] and comments proposed by Kay [35] and Haykin [36]. In addition, the proposed and RV methods are more moderate for data sizes smaller than or equal to 2048, but the RV method for data sizes larger than 2048 tends to select a higher order, leading to less accuracy.

On the surface, all the methods with order selection, on average, only use less than 7 percent of each data size to compute the parameter sets as compared to the original one [20]. Indeed, all but the proposed method need to compute the parameter sets of all orders before determining an appropriate order. Thus, no computation cost is saved in the six order selection methods.

On the contrary, the proposed method can determine the threshold for each data size beforehand and the order accordingly, so it only computes the parameter sets until the threshold or the corresponding order is reached. Since the computational cost of a complete Levinson algorithm for order N is $O(N^2)$, at least 99.51 percent of computational cost is saved, i.e., the proposed method only needs $0.0049O(N^2)$ at most. Furthermore, the amount of saving is normally raised as the data size increases.

Table 5

Comparison of methods for the averages of mean-squared errors on Set 2

Methods	$N = 128$	$N = 256$	$N = 512$	$N = 1024$	$N = 2048$	$N = 4096$	$N = 8192$	$N = 16384$
DI	3.53E-03	1.64E-03	8.66E-04	3.85E-04	2.03E-04	**1.31E-04**	**6.04E-05**	**4.03E-05**
FPE	4.16E-03	1.82E-03	9.46E-04	3.98E-04	2.23E-04	1.32E-04	6.37E-05	4.24E-05
AIC	4.15E-03	1.82E-03	9.46E-04	3.98E-04	2.22E-04	1.32E-04	6.37E-05	4.24E-05
MDL	7.47E-03	3.95E-03	2.07E-03	9.29E-04	5.83E-04	3.69E-04	1.87E-04	9.93E-05
CAT	4.14E-03	1.83E-03	9.46E-04	3.99E-04	2.24E-04	1.32E-04	6.37E-05	4.24E-05
RV	**3.50E-03**	**1.61E-03**	**8.14E-04**	**3.37E-04**	**1.69E-04**	1.56E-04	1.26E-04	7.51E-05
HQ	5.24E-03	2.68E-03	1.27E-03	5.62E-04	3.42E-04	2.18E-04	9.80E-05	5.67E-05

Table 6

Comparison of methods for means of orders selected on Set 2

Methods	$N = 128$	$N = 256$	$N = 512$	$N = 1024$	$N = 2048$	$N = 4096$	$N = 8192$	$N = 16384$
DI	8.71	16.75	22.13	25.27	33.32	**39.00**	**46.22**	**53.98**
FPE	3.33	4.67	6.44	9.09	12.53	18.76	25.41	36.18
AIC	3.34	4.69	6.44	9.10	12.54	18.76	25.41	36.18
MDL	1.38	1.89	2.72	3.71	5.06	6.81	9.44	12.84
CAT	3.26	4.55	6.41	9.00	12.47	18.64	25.38	36.15
RV	**6.35**	**11.11**	**17.01**	**28.89**	**48.17**	89.84	133.18	200.22
HQ	1.88	2.69	3.85	5.31	7.34	10.13	14.19	19.23

Table 7

Comparison of methods for standard deviations of orders selected on Set 2

Methods	$N = 128$	$N = 256$	$N = 512$	$N = 1024$	$N = 2048$	$N = 4096$	$N = 8192$	$N = 16384$
DI	4.79	11.10	14.41	15.66	21.40	**23.93**	**24.75**	**26.48**
FPE	2.98	3.07	3.17	4.34	5.12	7.12	7.81	9.99
AIC	2.98	3.12	3.17	4.35	5.13	7.12	7.81	9.99
MDL	0.64	0.82	1.05	1.08	1.20	1.49	1.68	2.14
CAT	2.83	2.79	3.14	4.25	5.06	7.05	7.80	9.97
RV	**4.55**	**7.64**	**10.85**	**19.56**	**32.60**	55.81	82.77	121.18
HQ	1.01	1.27	1.60	1.84	2.05	2.58	3.08	3.73

At data sizes of 4096, 8192, and 16384, the proposed method can save more than 99.99 percent of computational cost of implementing the Levinson algorithm.

In order to confirm that the proposed method is not successful by chance, another set of realizations, called Set 2, were generated. Their results are listed in Tables 5-7. Obviously, the results are consistent with the ones of Set 1.

5. Conclusion

In this paper, a novel method of order selection, called data induction, is proposed to choose an appropriate model order. The data-induction method, on average, can save more than 99 percent of computational cost compared to the original AR method as well as six existing order selection methods. Experimental results show that the proposed data-induction method is more reliable than six existing order selection methods. Taking accuracy and efficiency into consideration, the proposed estimator is more promising and attractive for estimating the Hurst exponent through an AR model. For readers

simply focusing on how to differentiate between biomedical or other signals that can be modeled by DFGN or DFBM, an easy-to-use procedure has been summarized; for readers who want to develop other useful tools for some fields, where a mathematical analysis of a closed-form expression is difficult to achieve, the concept of data induction will be a feasible approach.

References

[1] B.B. Mandelbrot and J.W.V. Ness, Fractional Brownian motions, fractional noises and applications, SIAM Rev. **10** (1968), 422–437.

[2] H.-O. Peitgen and D. Saupe, Fractals in natural: From characterization to simulation, in: The Science of Fractal Images, Springer-Verlag, New York, 1988, pp. 21–70.

[3] B.B. Mandelbrot, Mathematical backup and addenda, in: The Fractal Geometry of Nature, W.H. Freeman and Company, New York, 1983, pp. 349–390.

[4] K. Falconer, Brownian motion and Brownian surfaces, in: Fractal Geometry: Mathematical Foundations and Applications, John Wiley & Sons, New York, 1990, pp. 237–253.

[5] G. Samorodnitsky and M.S. Taqqu, Self-similar process, in: Stable Non-Gaussian Random Processes: Stochastic Models with Infinite Variance, Chapman & Hall, New York, 1994, pp. 309–390.

[6] E.N. Bruce, Scaling and long-term memory, in: Biomedical Signal Processing and Signal Modeling, John Wiley & Sons, New York, 2001, pp. 399–433.

[7] A.P. Pentland, Fractal-based description of natural scenes, IEEE Trans. Pattern Anal. Machine Intell. (PAMI) **6** (1984), 661–674.

[8] T. Lundahl, J. Ohley, S.M. Kay and R. Siffert, Fractional Brownian motion: A maximum likelihood estimator and its application to image texture, IEEE Trans. Med. Image (MI) **5** (1986), 152–161.

[9] P. Flandrin, On the spectrum of fractional Brownian Motions, IEEE Trans. Inform. Theory **35** (1989), 197–199.

[10] N. Sarkar and B.B. Chaudhuri, An efficient approach to estimate fractal dimension of textural images, Pattern Recognit. **25** (1992), 1035–1041.

[11] S.S. Chen, J.M. Keller and R.M. Crownover, On the calculation of fractal features from images, IEEE Trans. Pattern Anal. Mach. Intell. **15** (1993), 1087–1090.

[12] N. Sarkar and B.B. Chaudhuri, An efficient differential box-counting approach to compute fractal dimension of image, IEEE Trans. Syst. Man Cybern. **24** (1994), 115–120.

[13] X.C. Jin, S.H. Ong and Jayasooriah, A practical method for estimating fractal dimension, Pattern Recognit. Lett. **16** (1995), 457–464.

[14] M.S. Taqqu, V. Teverovsky and W. Willinger, Estimators for long-range dependence: An empirical study, Fractals **3** (1995), 785–798.

[15] S.-C. Liu and S. Chang, Dimension estimation of discrete-time fractional Brownian motion with applications to image texture classification, IEEE Trans. Image Process. **6** (1997), 1176–1184.

[16] S. Chang, S.-T. Mao, S.-J. Hu, W.-C. Lin and C.-L. Cheng, Studies of detrusor-sphincter synergia and dyssynergia during micturition in rats via fractional Brownian motion, IEEE Trans. Biomed. Eng. **47** (2000), 1066–1073.

[17] R. Jennane, W.J. Ohley, S. Majumdar and G. Lemineur, Fractal analysis of bone X-ray tomographic microscopy projections, IEEE Trans. Med. Imaging **20** (2001), 443–449.

[18] A. Peréz, C.E. D'Attellis, M. Rapacioli, G.A. Hirchoren and V. Flores, Analyzing blood cell concentration as a stochastic process, IEEE Eng. Med. Biol. Mag. **20** (2001), 170–175.

[19] A.N. Esgiar, R.N.G. Naguib, B.S. Sharif, M.K. Bennett and A. Murray, Fractal analysis in the detection of colonic cancer images, IEEE Trans. Inf. Technol. Biomed. **6** (2002), 54–58.

[20] Y.-C. Chang and S. Chang, A fast estimation algorithm on the Hurst parameter of discrete-time fractional Brownian motion, IEEE Trans. Signal Process. **50** (2002), 554–559.

[21] S. Chang, S.-J. Hu and W.-C. Lin, Fractal dynamics and synchronization of rhythms in urodynamics of female Wistar rats, J. Neurosci. Meth. **139** (2004), 271–279.

[22] J.-M. Lee, U. Yoon, J.-J. Kim, I.Y. Kim, D.S. Lee, J.S. Kwon and S.I. Kim, Analysis of the hemispheric asymmetry using fractal dimension of a skeletonized cerebral surface, IEEE Trans. Biomed. Eng. **51** (2004), 1494–1498.

[23] S. Chang, S.-J. Li, M.-J. Chiang, S.-J. Hu M.-C. Hsyu, Fractal dimension estimation via spectral distribution function and its application to physiological signals, IEEE Trans. Biomed. Eng. **54** (2007), 1895–1898.

[24] S. Chang, M.-C. Hsyu, H.-Y. Cheng and S.-H. Hsieh, Synergic co-activation of muscles in elbow flexion via fractional Brownian motion, Chin. J. Physiol. **51** (2008), 376–386.

[25] S. Chang, M.-C. Hsyu, H.-Y. Cheng, S.-H. Hsieh and C.-C. Lin, Synergic co-activation in forearm pronation, Ann. Biomed. Eng. **36** (2008), 2002–2018.
[26] S. Chang, Physiological rhythms, dynamical diseases and acupuncture, Chin. J. Physiol. **53** (2010), 77–90.
[27] S. Chang, Fractional Brownian motion in biomedical signal processing, physiology, and modern physics, Proceedings of the 5th International Conference on Bioinformatics and Biomedical Engineering (iCBBE'11), Wuhan, China, May 2011.
[28] P.-W. Huang and C.-H. Lee, Automatic classification for pathological prostate images based on fractal analysis, IEEE Trans. Medical Imaging **28** (2009), 1037–1050.
[29] P.-L. Lin, P.-W. Huang, C.-H. Lee and M.-T. Wu, Automatic classification for solitary pulmonary nodule in CT image by fractal analysis based on fractional Brownian motion model, Pattern Recognition **46** (2013), 3279–3287.
[30] Y.-C. Chang, L.-H. Chen, L.-C. Lai and C.-M. Chang, An efficient variance estimator for the Hurst exponent of discrete-time fractional Gaussian noise, IEICE Trans. Fundam. Electron. Commun. Comput. Sci. **E95–A** (2012), 1506–1511.
[31] Y.-C. Chang, L.-C. Lai, L.-H. Chen, C.-M. Chang and C.-C. Chueh, A Hurst exponent estimator based on autoregressive power spectrum estimation with order selection, Bio-Medical Materials and Engineering **24** (2014), 1041–1051.
[32] J.L. Doob, Linear least squares prediction-stationary (wide sense) process, in: Stochastic Processes, John Wiley & Sons, New York, 1953, pp. 560–598.
[33] M.B. Priestley, Prediction, filtering and control, in: Spectral Analysis and Time Series, Volume 2: Multivariate Series, Prediction and Control, Academic Press, New York, 1981, pp. 727–815.
[34] A.N. Shiryaev, Stationary (wide sense) random sequences, in: Probability, 2nd ed., translated by R. P. Boas, Springer-Verlag, New York, 1996, pp. 415–473.
[35] S.M. Kay, Autoregressive spectral estimation: General and Methods, in: Modern Spectral Estimation: Theory & Application, Prentice-Hall, New Jersey, 1988, pp. 153–270.
[36] S. Haykin, Linear prediction and Levinson-Durbin algorithm, in: Modern Filters, Macmillan, New York, 1989, pp. 108–183.
[37] A. Papoulis, Predictable processes and Wold's decomposition: A review, IEEE Trans. Acoust. Speech, Signal Process. **33** (1985), 933–938.
[38] A. Papoulis, Levinson's algorithm, Wold's decomposition, and spectral estimation, SIAM Rev. **27** (1985), 405–441.
[39] M. Deriche and A.H. Tewfik, Maximum likelihood estimation of the parameters of discrete fractionally differenced Gaussian noise process, IEEE Trans. Signal Process. **41** (1993), 2977–2989.
[40] R. Palaniappan, Towards optimal model order selection for autoregressive spectral analysis of mental tasks using genetic algorithm, International Journal of Computer Science and Network Security **6** (2006), 153–162.
[41] A.M. Aibinu, A.R. Najeeb, M.J.E. Salami and A.A. Shafie, Optimal model order selection for transient error autoregressive moving average (TERA) MRI reconstruction method, World Academy of Science, Engineering and Technology **42** (2008), 161–165.