sensors

MDPI

Article

Blind Estimation of the PN Sequence of A DSSS Signal Using A Modified Online Unsupervised Learning Machine

Yangjie Wei *, Shiliang Fang *, Xiaoyan Wang and Shuxia Huang

Key Laboratory of Underwater Acoustic Signal Processing of Ministry of Education, Southeast University, Nanjing 210096, China; xyan@seu.edu.cn (X.W.); 13404133160@163.com (S.H.)

* Correspondence: 230169359@seu.edu.cn (Y.W.); slfang@seu.edu.cn (S.F.); Tel.: +86-137-0518-5218 (Y.W.); +86-182-5196-2822 (S.F.)

Received: 26 November 2018; Accepted: 11 January 2019; Published: 16 January 2019

Abstract: Direct sequence spread spectrum (DSSS) signals are now widely used in air and underwater acoustic communications. A receiver which does not know the pseudo-random (PN) sequence cannot demodulate the DSSS signal. In this paper, firstly, the principle of principal component analysis (PCA) for PN sequence estimation of the DSSS signal is analyzed, then a modified online unsupervised learning machine (LEAP) is introduced for PCA. Compared with the original LEAP, the modified LEAP has the following improvements: (1) By normalizing the system state transition matrices, the modified LEAP can obtain better robustness when the training errors occur; (2) with using variable learning steps instead of a fixed one, the modified LEAP not only converges faster but also has excellent estimation performance. When the modified LEAP is converging, we can utilize the network connection weights which are the eigenvectors of the autocorrelation matrix of the DSSS signal to estimate the PN sequence. Due to the phase ambiguity of the eigenvectors, a novel approach which is based on the properties of the PN sequence is proposed here to exclude the wrong estimated PN sequences. Simulation results showed that the methods mentioned above can estimate the PN sequence rapidly and robustly, even when the DSSS signal is far below the noise level.

Keywords: PN sequence estimation; DSSS signals; PCA; modified LEAP

1. Introduction

Direct sequence spread spectrum (DSSS) signals have been used widely in military as well as civil communication systems for several decades because of their strong anti-interference ability and low probability of interception property [1–7]. In a non-cooperative DSSS communication system, the pseudo-random (PN) sequence of the DSSS signal is a crucial parameter for blind dispreading. Consequently, it is of great significance to estimate it. Many methods for PN sequence estimation of DSSS signals have been published in recent years [8–20].

In Reference [8], Warner et al. firstly used triple correlation function (TCF) to estimate the spreading code; since then, many algorithms based on it have been proposed for PN sequence estimation [9–13]. These TCF-based methods can be used not only for PN sequence estimation, but also for detecting DSSS signals. They can work well in good conditions, but as the environment gets worse, the performance of these algorithms deteriorates sharply.

In Reference [14], Burel et al. introduced an algorithm to PN sequence estimation of the DSSS signal based on eigenvalue decomposition (EVD). In this method, the received DSSS signal is sampled and divided into temporal windows, the size of which is the PN sequence period. Each window provides a vector for the eigenanalysis, then the PN sequence can be estimated by the eigenvectors.

In Reference [15], the sampled DSSS signal was divided into continuous non-overlapping temporal vectors with a width of two periods of PN sequence; then, the PN sequence could be estimated by employing the EVD method after the average correlation matrix was calculated. In Reference [16], Qui et al. proposed a PN sequence estimation method. In this method, the signal was divided into a series of overlapping windows with the width much shorter than the information symbol width, and then the segments of the spreading code were estimated using the EVD method and a complete spreading code was obtained from the estimated segments. Although these EVD based methods [14–18] show excellent performance in low signal to noise ratio (SNR) conditions, they become expensive in terms of computing when the length of the PN sequence is long. In order to solve this problem, many PN sequence estimation methods based on neural network have been proposed.

For example, in Reference [19], Dominique et al. introduced a subspace-based PN sequence estimation algorithm for DSSS signals using a simplified Hebb rule, which can reduce the number of computations required compared to the existing Hebb-based sequence estimator. The main advantage of these Hebb estimators is that the estimator architecture is very simple and can be implemented easily, but the constant small learning steps severely limit their performance.

In Reference [20], Chen et al. proposed a modified online supervised learning machine (LEAP) to extract multiple principal components. In other words, the LEAP can be used for principal component analysis (PCA). This algorithm is adaptive to nonstationary input and requires no knowledge of, or when, the input changes statistically. Since it requires little memory or data storage, the LEAP is very suitable for use in engineering. However, the constant small learning step also severely limits its performance. Although the convergence speed of the LEAP can be accelerated when using a large learning step, the LEAP often fails to converge to the global optimum point and the large learning step may damage the stability of the system. Based on these mentioned above, in this paper, we propose a modified LEAP algorithm and apply it into the PN sequence estimation of the DSSS signal. Compared to the original LEAP, the modified LEAP uses variable learning steps instead of a fixed one, which can greatly improve the convergence performance of the network. Namely, the modified LEAP first makes the network close to the optimum convergence point with large learning steps when the network starts training, then when the network approaches the optimum convergence point, small learning steps are used, thus ensuring the network converges to the best point. Meanwhile, in order to maintain the stability of the network when using variable learning steps, the state transition matrices of the system are normalized. In summary, our main contributions lie in the following folds:

(a) The LEAP algorithm is applied into the field of the PN sequence estimation of DSSS signals and a modified LEAP algorithm is proposed. Compared to the original LEAP algorithm, the modified LEAP algorithm has a better convergence performance due to its use of variable learning steps rather than a fixed one;

(b) Since the phase of the eigenvector can be inverted, the incorrect estimation of the PN sequence of the DSSS signal may be obtained. Based on this, a novel approach which makes full use of the correlation characteristics of the PN sequence is proposed here to solve this problem.

This paper is organized as follows. In Section 2, firstly, the mathematical model of DSSS signals and the principle of PCA for PN sequence estimation are given, then a modified LEAP is introduced. The method for PN sequence estimation and the elimination of phase ambiguity are described in Section 3. The main steps for PN sequence estimation of DSSS signals are described in Section 4. Simulation results are presented in Section 5. Finally, a conclusion is drawn in Section 6.

2. Basic Theories

2.1. DSSS Signal Model

In a DSSS transmission, the symbols are multiplied by a PN sequence, which spreads the bandwidth. In this paper, we use the notations below [14]:

$p(t)$: The convolution of the transmission filter, the channel filter (which represents the channel echoes) and the receiver filter.

$\{c_m, m = 0, 1, 2, \cdots, N-1\}$: The PN sequence.

N: The length of the PN sequence.

T_p: The symbol period.

T_c: The chip period ($T_c = T_p/N$).

$h(t)$: The convolution of the PN sequence with all the filters of the transmission chain (transmitter filter, channel echoes, and receiver filter):

$$h(t) = \sum_{m=0}^{N-1} c_m p(t - mT_c). \tag{1}$$

h: The vector containing the samples of $h(t)$.

$s(t)$: The DSSS baseband signal at the output of the receiver filter:

$$s(t) = \sum_{l=-\infty}^{+\infty} a_l h(t - lT_p). \tag{2}$$

a_l: The message symbols.

$v(t)$: The noise at the output of the receiver filter, which is uncorrelated with the signal.

Then, the baseband signal at the output of the receiver filter can be written as:

$$x(t) = s(t) + v(t). \tag{3}$$

2.2. The Principle of PCA for PN Sequence Estimation

Since many algorithms can estimate T_p and T_c, they are assumed to be obtained in advance in this paper [15]. After being sampled (the sampling rate is $1/T_c$) and passed through an observation window with duration T_p, the received DSSS signal $x(t)$ can produce a series of observed sample vectors after each interval T_p. Let us note $\mathbf{x}(k)$ the content of a window, then the $\mathbf{x}(k)$ can be modeled as:

$$\mathbf{x}(k) = \mathbf{s}(k) + \mathbf{v}(k), k = 1, 2, 3, \cdots, \tag{4}$$

where the dimension of $\mathbf{x}(k)$ is $N = T_p/T_c$, k represents the discrete time. Usually, the observation widow has a random time delay T_x, which is the desynchronization between windows and symbols $\left(0 \le T_x < T_p\right)$. Therefore, $\mathbf{s}(k)$ generally contains two consecutive message symbol bits, and $\mathbf{s}(k)$ can be written as:

$$\mathbf{s}(k) = a_k \mathbf{h}_1 + a_{k+1} \mathbf{h}_2, \tag{5}$$

where a_k, a_{k+1} are the two consecutive message symbols.

$\mathbf{h}_1$ is a vector containing the end (duration $T_p - T_c$) of the spreading waveform $h(t)$, followed by zeros (duration T_x).

$\mathbf{h}_2$ is a vector containing the zeros (duration $T_p - T_x$) followed by the beginning (duration T_x) of the spreading waveform $h(t)$.

Let $\mathbf{e}_i = \mathbf{h}_i / \|\mathbf{h}_i\|, i = 1, 2$, we can obtain:

$$\mathbf{e}_i^T \mathbf{e}_j = \delta(i - j), (i, j = 1, 2), \tag{6}$$

where $e_i, i = 1, 2$ are orthonormalized vectors and $\delta(\cdot)$ is the Dirac function. Then, the $\mathbf{x}(k)$ can be expressed as follows:

$$\mathbf{x}(k) = a_k \|\mathbf{h}_1\| \mathbf{e}_1 + a_{k+1} \|\mathbf{h}_2\| \mathbf{e}_2 + \mathbf{v}(k). \tag{7}$$

Using the equations above, the autocorrelation matrix $\mathbf{R}_x$ of the DSSS signal can be obtained by:

$$\mathbf{R}_x = E\left[\mathbf{x}\mathbf{x}^T\right] = \left[\sigma_n^2 \eta \frac{T_p - T_x}{T_c}\right]\mathbf{e}_1\mathbf{e}_1^T + \left(\sigma_n^2 \eta \frac{T_x}{T_c}\right)\mathbf{e}_2\mathbf{e}_2^T + \sigma_n^2 \mathbf{I}, \tag{8}$$

where $E\{\cdot\}$ denotes expectation, σ_n^2 is the variance of the noise, $\eta = \sigma_s^2/\sigma_n^2$, σ_s^2 is the variance of $\mathbf{s}(k)$, and $\mathbf{I}$ is an identity matrix of dimension $N \times N$. From Equation (8), it is clear that two eigenvalues will be larger than the others when $T_x > 0$, and according to their corresponding eigenvectors, which can be obtained by PCA, the PN sequence can be estimated.

2.3. Mathematical Model of The Modified LEAP

The LEAP is implemented on a neural network with linear units shown in Figure 1. Specifically, in many practical applications, $M \ll N$. Let $\mathbf{x}(k)$ denote the input vector process, then the network's input–output relation can be written as [20]:

$$y_i(k) = \mathbf{w}_i^T(k)\mathbf{x}(k), i = 1, 2, \cdots, M, \tag{9}$$

where $\mathbf{w}_i(k) = [w_{i1}(k), w_{i2}(k), \cdots, w_{iN}(k)]^T$, for $i = 1, 2, \cdots, M$. $\mathbf{x}(k) = [x_1(k), x_2(k), \cdots, x_N(k)]^T$, $\mathbf{y}(k) = [y_1(k), y_2(k), \cdots, y_M(k)]^T$. Here, $x_i(k)$ is the ith input of the network, $y_j(k)$ is the jth output, $w_{ij}(k)$ is the connection weight from the jth input to the ith output neuron, all at discrete time k. Supposing that $\mathbf{R}_x$ has eigenvalues $\lambda_1 > \lambda_2 > \cdots > \lambda_N > 0$ with corresponding normalized eigenvectors $\mathbf{e}_i$, $i = 1, 2, \cdots, N$, then $\mathbf{w}_i(k) \to \mathbf{e}_i$, $E\{y_i^2\} \to \lambda_i$, as $k \to \infty$, for $i = 1, 2, \cdots, M$.

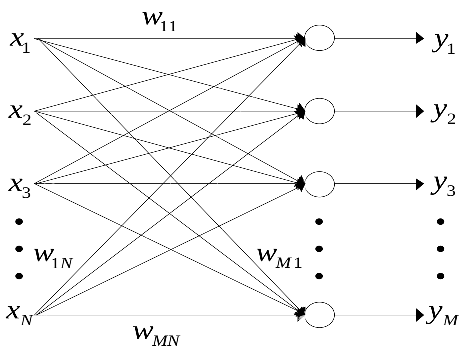

Figure 1. A neural network model of LEAP, an online unsupervised learning machine.

The LEAP for connection weight updating is the following nonlinear non-autonomous dynamical vector difference equations:

$$\mathbf{w}_i(k+1) = \mathbf{w}_i(k) + \beta\left\{\mathbf{B}_i(k)\left[y_i(k)\mathbf{x}(k) - y_i^2(k)\mathbf{w}_i(k)\right] - \mathbf{A}_i(k)\mathbf{w}_i(k)\right\}, \tag{10}$$

for $i = 1, 2, \cdots, M$, and:

$$\mathbf{A}_i(k) = \begin{cases} 0, & i = 1 \\ \sum\limits_{j=1}^{i-1} \mathbf{w}_j(k)\mathbf{w}_j^T(k), & i = 2, 3, \cdots, M \end{cases} \tag{11}$$

$$\mathbf{B}_i(k) = \mathbf{I} - \mathbf{A}_i(k), i = 1, 2, \cdots, M, \tag{12}$$

β is the constant learning step.

In Equation (10), the $\mathbf{A}_i$ and $\mathbf{B}_i$ can be seen as the state transfer matrices of the system, which are important "de-correlation" terms for performing Gram–Schmidt orthogonalization among all connection weights at each iteration. One could think of the term $y_i\mathbf{x}$ as the so-called Hebbian learning, for which the strengthening of the connection weights is proportional to the input–output correlation.

According to the theory in Reference [20], in the original LEAP, it is known that the learning step β should be small enough, otherwise the system performance can be severely degraded and training errors may occur, thus making the system unstable. Namely, there is an equilibrium point: If the learning step β exceeds this point, the system will be unusable. Therefore, in practice, the learning step is set as small as possible in the original LEAP, but it greatly increases the time required for network convergence. On the basis of the reasons above, a modified LEAP is proposed. First, the learning step β is modified to:

$$\beta_i(k+1) = \alpha\beta_i(k) + \gamma(|\lambda_i(k) - \lambda_i(k-1)|/\max\{\lambda_i(k), \lambda_i(k-1)\}), \tag{13}$$

where:

$$\lambda_i(k) = E\left\{y_i^2\right\}, \tag{14}$$

for $i = 1, 2, \cdots, M$, $0 < \alpha < 1$, $\gamma > 0$, $|\cdot|$ denotes taking the absolute value, $\max\{\cdot\}$ denotes taking the maximum value. Where α and γ are the weight coefficients in the variable learning steps and they are similar to the weight coefficients in the variable step size least mean square (LMS) algorithm. Compared to those of the original LEAP algorithm, the learning steps of the modified LEAP algorithm can be adaptively changed according to the output of the network. Namely, when the network starts training, the difference between $\lambda_i(k)$ and $\lambda_i(k-1)$ is large; the network uses large learning steps at this time, and when the network is about to converge, $\lambda_i(k)$ and $\lambda_i(k-1)$ are approximately equal, and the network uses small learning steps in this case. In this way, the modified LEAP can not only have fast convergence speed, but also get good steady-state performance. In Equation (13), the $\max\{\lambda_i(k), \lambda_i(k-1)\}$ is divided in the formula to prevent $|\lambda_i(k) - \lambda_i(k-1)|$ from becoming too large, which may be caused by the training errors etc., thus making the learning steps become too large and seriously affecting the convergence speed of the network.

However, when the variable learning steps are used, since the learning step size of the network of the modified LEAP is slightly large at the beginning of the training, the step size may still exceed the equilibrium point mentioned in Reference [20]. At this time, the uncorrelation between the connection weights of the network may be destroyed, which results in the instability of the system. Therefore, the state transition matrix of the system is normalized here, that is:

$$\mathbf{A}_i(k) = \mathbf{A}_i(k)/\|\mathbf{A}_i(k)\|_F, i = 2, 3, \cdots, M, \tag{15}$$

where $\|\cdot\|_F$ denotes the Frobenius norm. It is obvious that $\|\mathbf{A}i(k)\|_F$ is a compatible matrix norm, then $0 < \lambda(\mathbf{A}_i(k)) \leq 1$, $0 < \lambda(\mathbf{B}_i(k)) \leq 1$ and the normalization does not affect the decorrelation function of the matrices $\mathbf{A}_i$, $\mathbf{B}_i$. Therefore, according to the stability criterion of Liapunov [21,22], we can maintain the system stability, when the training errors damage the uncorrelation between the connection weights $\mathbf{w}_i$.

2.4. Asymptotic Stability Analysis of The Modified LEAP

Since $\beta_i(k)$ is small when $k \to \infty$, and $\mathbf{A}_i(k)_F \geq 1$, we can obtain this approximation:

$$\frac{\beta_i(k)}{\|\mathbf{A}_i(k)\|_F} \approx \beta_i(k), \tag{16}$$

for $i = 2, 3, \cdots, M$. It can be easily shown that [20]:

$$\mathbf{w}_i(k) = \mathbf{e}_i, i = 1, 2, \cdots, M \tag{17}$$

is an equilibrium point of Equation (10). Let $\mathbf{g}_i(k) = \mathbf{w}_i(k) - \mathbf{e}_i$, for $i = 1, 2, \cdots, M$, we can get the following approximations:

$$\mathbf{g}_1(k+1) - \mathbf{g}_1(k) = \varsigma_1 \left[\mathbf{R}_x - \lambda_1 \left(\mathbf{I} + 2\mathbf{e}_1\mathbf{e}_1^T \right) \right] \mathbf{g}_1(k), \tag{18}$$

for $i = 1$.

$$\mathbf{g}_i(k+1) - \mathbf{g}_i(k) = \varsigma_i \left\{ \left[\mathbf{R}_x - \lambda_i (\mathbf{I} + 2\mathbf{e}_i\mathbf{e}_i^T) + \sum_{j=1}^{i-1} (\lambda_i - \lambda_j - 1) \mathbf{e}_j\mathbf{e}_j^T \right] \mathbf{g}_i(k) - \sum_{j=1}^{i-1} \mathbf{e}_j\mathbf{e}_i^T \mathbf{g}_j(k) \right\}, \tag{19}$$

for $i = 2, 3, \cdots, M$. Here, $\varsigma_i = \mu\beta_i(k)$, μ is a positive integer. Then, Equations (18) and (19) can also be written as:

$$\begin{bmatrix} \mathbf{g}_1(k+1) \\ \mathbf{g}_2(k+1) \\ \mathbf{g}_3(k+1) \\ \vdots \\ \mathbf{g}_M(k+1) \end{bmatrix} = \begin{bmatrix} \mathbf{I} + \varsigma_1\mathbf{D}_1 & 0 & 0 & 0 & 0 \\ -\varsigma_2\mathbf{e}_1\mathbf{e}_2^T & \mathbf{I} + \varsigma_2\mathbf{D}_2 & 0 & 0 & 0 \\ -\varsigma_3\mathbf{e}_1\mathbf{e}_3^T & -\varsigma_3\mathbf{e}_2\mathbf{e}_3^T & \mathbf{I} + \varsigma_3\mathbf{D}_3 & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ -\varsigma_M\mathbf{e}_1\mathbf{e}_M^T & -\varsigma_M\mathbf{e}_2\mathbf{e}_M^T & -\varsigma_M\mathbf{e}_3\mathbf{e}_M^T & \cdots & \mathbf{I} + \varsigma_M\mathbf{D}_M \end{bmatrix} \begin{bmatrix} \mathbf{g}_1(k) \\ \mathbf{g}_2(k) \\ \mathbf{g}_3(k) \\ \vdots \\ \mathbf{g}_M(k) \end{bmatrix}, \tag{20}$$

where:

$$\mathbf{D}_i = \begin{cases} \mathbf{R}_x - \lambda_1(\mathbf{I} + 2\mathbf{e}_1\mathbf{e}_1^T), i = 1 \\ \mathbf{R}_x - \lambda_i(\mathbf{I} + 2\mathbf{e}_i\mathbf{e}_i^T) + \sum_{j=1}^{i-1} (\lambda_i - \lambda_j - 1)\mathbf{e}_j\mathbf{e}_j^T, i = 2, 3, \cdots, M \end{cases} \tag{21}$$

and because:

$$\mathbf{I} = \sum_{i=1}^{N} \mathbf{e}_i\mathbf{e}_i^T, \tag{22}$$

$\mathbf{D}_1$ can be written as:

$$\mathbf{D}_1 = \sum_{i=1}^{N} \lambda_i\mathbf{e}_i\mathbf{e}_i^T - \lambda_1 \sum_{i=1}^{N} \mathbf{e}_i\mathbf{e}_i^T - 2\lambda_1\mathbf{e}_1\mathbf{e}_1^T \tag{23}$$

According to Equation (23), it is obvious that $\mathbf{D}_1$'s eigenvectors are $\{\mathbf{e}_1, \mathbf{e}_2, \cdots \mathbf{e}_N\}$ with corresponding eigenvalues:

$$\{-2\lambda_1, (\lambda_2 - \lambda_1), \cdots, (\lambda_N - \lambda_1)\}. \tag{24}$$

Similarly, we can know that all $\mathbf{D}_i, i = 2, 3, \cdots, M$ have the same eigenvectors $\{\mathbf{e}_1, \mathbf{e}_2, \cdots \mathbf{e}_N\}$, and their corresponding eigenvalues are:

$$\{\overbrace{-1, \cdots, -1}^{i-1}, -2\lambda_i, \overbrace{(\lambda_{i+1} - \lambda_i), \cdots, (\lambda_N - \lambda_i)}^{N-i}\}, \tag{25}$$

On the basis of Equations (24) and (25), it is clear that all $\mathbf{D}_i$'s eigenvalues are negative. The magnitudes of all eigenvalues of $\mathbf{I} + \varsigma_i\mathbf{D}_i$ will be less than 1, if ς_i is small enough, for $i = 1, 2, \cdots, M$. Therefore, the equilibrium point given by Equation (17) is asymptotically stable in the sense of Liapunov [21,22], i.e., there exists a neighborhood of the point that any solution initially in this neighborhood will converge to the equilibrium point as $k \to \infty$.

3. PN Sequence Estimation and The Elimination of Phase Ambiguity

When the modified LEAP is converging, which means:

$$|\lambda_i(k) - \lambda_i(k-1)| < \varepsilon, \tag{26}$$

where ε represents a threshold for judging whether the network is converging, $0 < \varepsilon \ll 1$, $i = 1, 2, \cdots, M$, then concatenating $\mathbf{e}_1(k)$ and $\mathbf{e}_2(k)$. Because the phase of eigenvectors can be inverted, four estimated sequences $\mathbf{b}_i, i = 1, 2, \cdots, 4$ can be obtained. Then, the estimated PN sequence of the DSSS signal can be calculated by:

$$\mathbf{PN}_i = \mathrm{sgn}(\mathbf{b}_i), i = 1, 2, \cdots, 4, \tag{27}$$

where:

$$\mathrm{sgn}(x) = \begin{cases} 1, & x > 0 \\ 0, & x = 0 \\ -1, & x < 0 \end{cases}. \tag{28}$$

$\hat{\mathbf{PN}}_i$ is a vector of length N. In order to select the true estimated PN sequence from $\hat{\mathbf{PN}}_i, i = 1, 2, \cdots, 4$, the correlation characteristics of the PN sequence of the DSSS signal can be used. Namely, the PN sequence has good autocorrelation and bad cross-correlation. Hence, let us define a correlation factor ψ:

$$\psi_i = \sum_{\tau=0}^{N-1} \left| \widehat{\mathbf{PN}}_i(n) \widehat{\mathbf{PN}}_i^{*}(n - \tau) \right|, \tag{29}$$

where $\tau = 0, 1, 2, \cdots, N-1$ denotes the discrete time delay and $i = 1, 2, \cdots, 4$, * denotes the conjugate operation. Obviously, the smaller the ψ is, the worse the cross-correlation of the PN sequence is. Then, the correct index of the true estimated PN sequence among $\hat{\mathbf{PN}}_i, i = 1, 2, \cdots, 4$ can be calculated by:

$$id = \arg \min_{i=1,2,\cdots,4} [\psi_i], \tag{30}$$

where $\min\{\cdot\}$ denotes taking the minimum value. Then, according to Equation (30), the true estimated PN sequence is $\hat{\mathbf{PN}}_{id}$. In addition, in the absence of other prior information, the overall phase ambiguity of the $\hat{\mathbf{PN}}_{id}$ cannot be eliminated, which means the true PN sequence could be either $\hat{\mathbf{PN}}_{id}$ or $-\hat{\mathbf{PN}}_{id}$. For example, if we know that this PN sequence is the m-sequences, then we can eliminate the overall phase ambiguity of the estimated PN sequence according to the equalization characteristics of the m-sequences; namely, in the m-sequences, the number of 1 is one more than the number of -1.

4. The Main Steps for PN Sequence Estimation

To be more specific, the main steps involved in this paper for PN sequence estimation of the DSSS signal are summarized as follows:

Step 1. Sample the received DSSS signal, then obtain $\mathbf{x}(k), k = 1, 2, 3, \cdots$. Meanwhile, in order to improve the system robustness, the neural network input $\mathbf{x}(k)$ should be normalized as follows:

$$\mathbf{x}(k) = \mathbf{x}(k) / \|\mathbf{x}(k)\|, \tag{31}$$

where:

$$\|\mathbf{x}(k)\| = \sqrt{\mathbf{x}^T(k)\mathbf{x}(k)}. \tag{32}$$

Step 2. Setting the initial value of $\mathbf{w}_i, i = 1, 2, \cdots, M$, which are often random numbers between -1 and 1, then normalizing:

$$\mathbf{w}_i = \mathbf{w}_i / \|\mathbf{w}_i\|, i = 1, 2, \cdots, M. \tag{33}$$

Step 3. According to Equations (10)–(15), updating the weight vectors $\mathbf{w}_i, i = 1, 2, \cdots, M$.

Step 4. Extracting the eigenvectors corresponding to the largest and second largest eigenvalues, when the neural network is converging. Subsequently, concatenating the two eigenvectors, then according to Equations (26)–(30), the PN sequence of the DSSS signal can be estimated. Then, the blind PN sequence estimation method of the DSSS signal proposed in this paper is derived.

5. Simulations and Analysis

To verify the capability of the proposed method, simulation results are presented in this section. Here, the DSSS signal is generated using a random sequence of length 31 (it is one of the m-sequences); then, for completeness, we shall set $N = M = 31$. The symbols belong to a BPSK constellation (binary phase shift keying). The noise is additive white Gaussian noise, which is uncorrelated with the DSSS signal. $T_x/T_c = 10$. Figure 2 shows the true PN sequence of the DSSS signal.

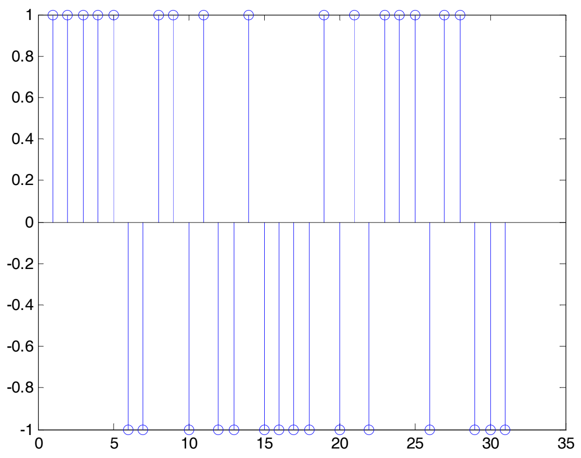

Figure 2. The true pseudo-random (PN) sequence of the direct sequence spread spectrum (DSSS) signal.

The estimated eigenvalues of the autocorrelation matrix of the DSSS signal are shown in Figure 3, and the normalized eigenvectors $\mathbf{e}_1$, $\mathbf{e}_2$. corresponding to the largest and second largest eigenvalues are shown in Figure 4a,b. Both of them are estimated by the modified LEAP $(\alpha = 0.9, \gamma = 2)$ and the SNR is -5 dB.

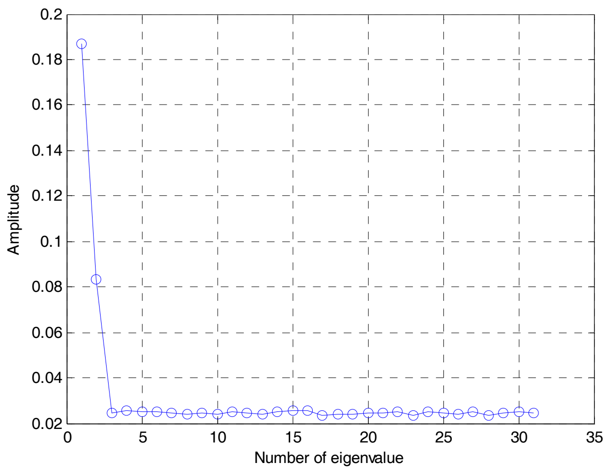

Figure 3. The estimated eigenvalues of the autocorrelation matrix of the DSSS signal.

Then, the two eigenvectors shown in Figure 4 are concatenated. Because the phase of eigenvectors can be reversed, we can get two different estimated sequences $\mathbf{b}_1$ and $\mathbf{b}_2$. shown in Figure 5 (here, we regard $\mathbf{b}$ and $-\mathbf{b}$ as the same). Then, according to Equation (27), the estimated PN sequences are:

$$\widehat{\mathbf{PN}}_i = \mathrm{sgn}(\mathbf{b}_i), i = 1, 2, \tag{34}$$

which are shown in Figure 6. On the basis of Equation (29), in this simulation, $\psi_1 = 102$, $\psi_2 = 94$. Since $\psi_1 > \psi_2$, the true estimated PN sequence is $\hat{\mathbf{PN}}_2$, which is the same as the true PN sequence of the DSSS signal shown in Figure 2. By now, the above simulation results show the validity of the proposed method, even with very low SNR.

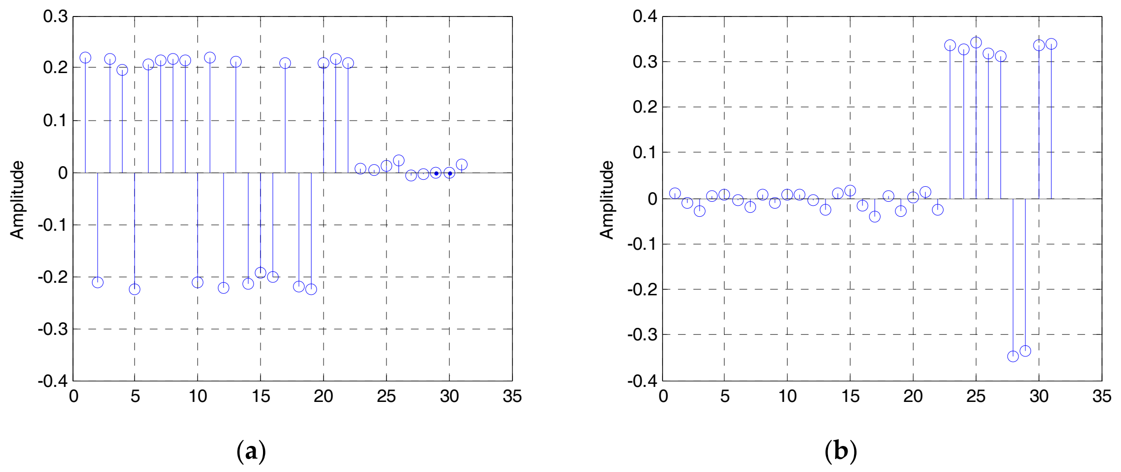

Figure 4. The estimated eigenvectors: (**a**) The eigenvector corresponding to the largest eigenvalue; (**b**) the eigenvector corresponding to the second largest eigenvalue.

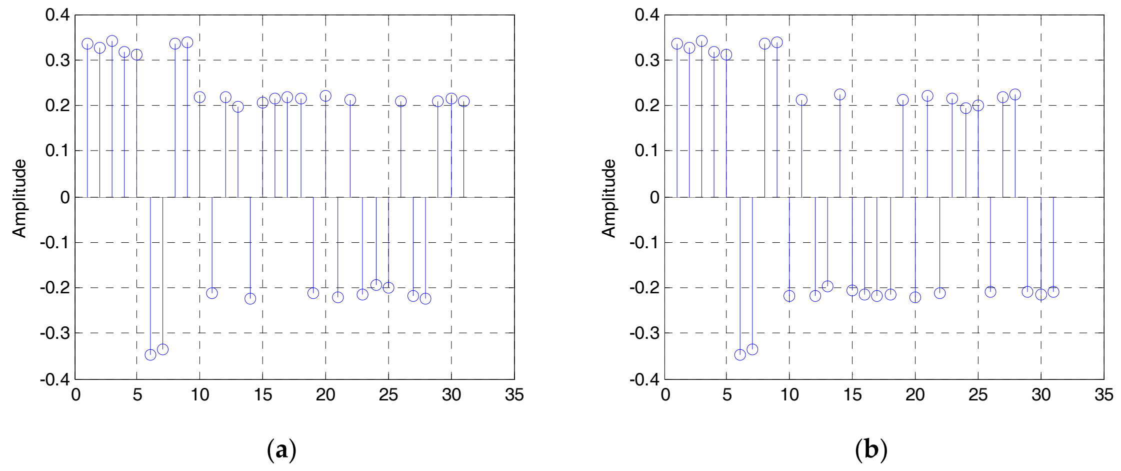

Figure 5. The estimated sequences: (**a**) $\mathbf{b}_1$; (**b**) $\mathbf{b}_2$.

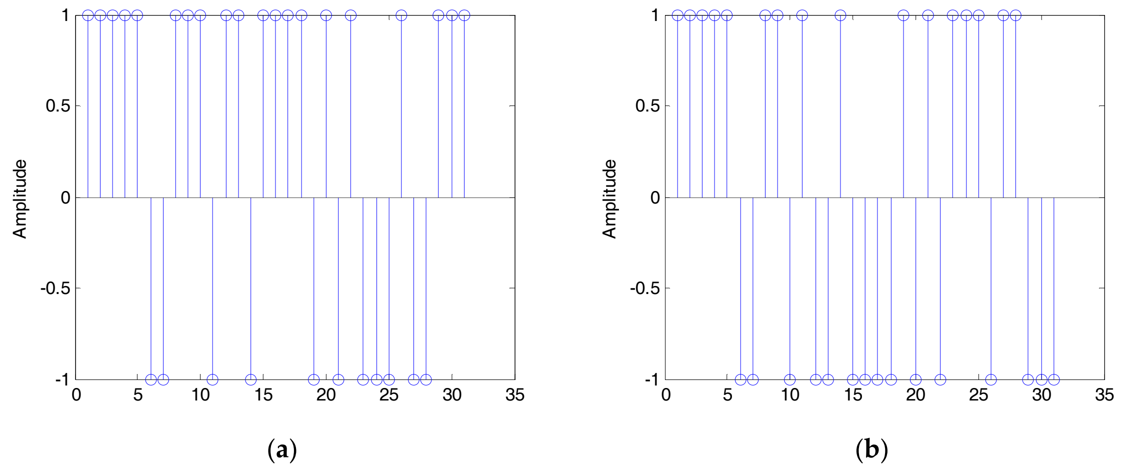

Figure 6. The estimated PN sequences: (**a**) $\hat{\mathbf{PN}}_1$; (**b**) $\hat{\mathbf{PN}}_2$.

Figure 7 shows the relationship between the number of iterative steps required by the modified LEAP and the original LEAP to make the network converge at different learning steps as the SNR changes. Figure 8 shows the relationship between the correct estimation probability of the PN sequence and SNRs, when using the modified LEAP and the original LEAP at different learning steps as well as the TCF- and EVD-based methods. Both tests use 1000 Monte Carlo simulations.

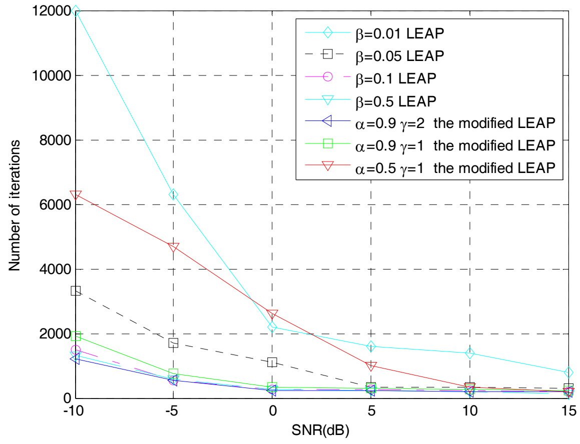

Figure 7. Number of iterations for network convergence of the original LEAP and the modified LEAP sunder different SNRs.

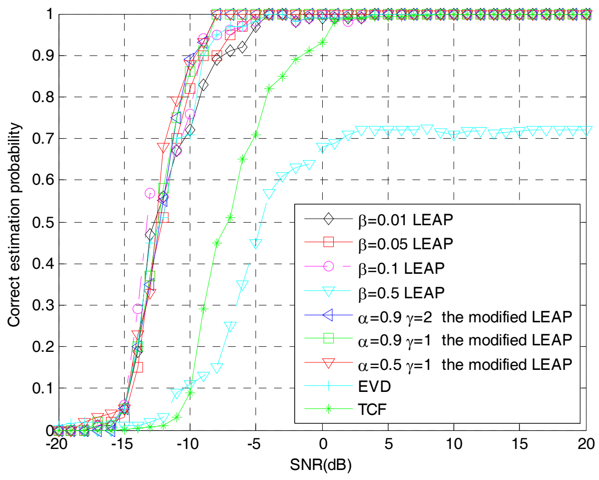

Figure 8. Correct PN sequence estimation probability of different methods under different SNRs.

It can be seen from Figures 7 and 8 that when the learning step is set to $\beta = 0.01$ and $\beta = 0.05$ in the original LEAP, the network can stably converge to the optimum point due to the small step size, but the number of iterations required for network convergence is relatively large, which is similar to the modified LEAP when $\alpha = 0.5, \gamma = 1$. When the learning step is set to $\beta = 0.5$ in the original LEAP, although the large learning step increases the convergence speed of the network, the correct estimation probability of the PN sequence is seriously reduced, because the network cannot converge to the optimum point. When the learning step of the original LEAP is set to $\beta = 0.1$, the original LEAP can not only converge rapidly, but also get good estimation performance, which is similar to the modified LEAP when $\alpha = 0.9,\ \gamma = 1$ or $\alpha = 0.9,\ \gamma = 2$. However, in practical applications, it is difficult to choose a suitable learning step when using the original LEAP, but in the modified LEAP, you just need to set the weight coefficients α and γ slightly larger, then the network will first approach the optimum point with a large step size and get to the optimum point with a small step size, which can not only make the network converge rapidly, but also obtain a high correct estimation probability of the PN sequence of the DSSS signal. Moreover, according to Figure 8, it is obvious that when the

LEAP-based methods can converge correctly, their performance is comparable to that of the EVD-based method and is superior to that of the TCF-based method. The reason is twofold. First, since the LEAP neural network is actually a principal component analysis network, the principle of the LEAP-based methods and the EVD-based method is the same, which means that their performance is comparable. Second, when using the TCF-based method for PN sequence estimation, the peaks of the TCF need to be accurately searched [8], which is difficult to achieve in low SNR environment. Therefore, the TCF-based method has poor performance when the SNR is low.

6. Conclusions

A blind PN sequence estimation method of the DSSS signal using a modified LEAP is proposed in this paper. Compared to the original LEAP, the modified LEAP makes it easier to set the suitable learning step size to obtain good convergence performance. When the modified LEAP is converging, the PN sequence of the DSSS signal can be estimated by the connection weights of the network. These weights are the eigenvectors of the autocorrelation matrix of the DSSS signal. Because of the phase ambiguity of the eigenvectors, a novel approach which is based on the characteristics of the PN sequence of the DSSS signal is also proposed here to exclude the wrong estimated PN sequences. As shown in the simulations, the proposed methods mentioned above can quickly estimate the PN sequence of the DSSS signal in a low SNR environment.

7. Patents

F.S.; Y.W.; X.W.; C.Z. PN sequence estimation of DSSS Signals based on a variable step size online unsupervised learning machine. China Patent 201810861275.2, 1 August 2018.

Author Contributions: The work presented in this paper has been carried out in collaboration with all authors. Y.W. designed and implemented the improved method. S.F., X.Y., and S.H. discussed the results with Y.W. and refined the overall manuscript.

Funding: This work was supported in part by National Natural Science Foundation of China under Grant no.11604048, and in part by the National Natural Science Foundation of China under Grant no.11704069, and in part by the Fundamental Research Funds for the Central Universities 2242016K30013, and in part by the National Natural Science Foundation of China under Grant no.11574048, and in part by the National Natural Science Foundation of China under Grant no.11674057.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. De Gaudenzi, R.; Giannetti, F.; Luise, M. Signal recognition and signature code acquisition in CDMA mobile packet communications. *IEEE Trans. Veh. Technol.* **1998**, *47*, 196–208. [CrossRef]
2. Pickholtz, R.; Schilling, D.; Milstein, L. Theory of spread-spectrum communications. *IEEE Trans. Commun.* **1982**, *30*, 855–884. [CrossRef]
3. Dillard, R.A. Detectability of Spread Spectrum Signals. *IEEE Trans. Aerosp. Electron. Syst.* **1979**, *15*, 526–537. [CrossRef]
4. Mou, Q.; Wei, P.; Tai, H.M. Invariant detection for short-code QPSK DS-SS signals. *Signal Process.* **2010**, *90*, 1720–1729. [CrossRef]
5. Rappaport, T.S. Spread spectrum signal Acquisition: Methods and technology. *IEEE Commun. Mag.* **1984**, *22*, 6–21. [CrossRef]
6. Flikkema, P.G. Spread-spectrum techniques for wireless communication. *IEEE Signal Process. Mag.* **1997**, *14*, 26–36. [CrossRef]
7. Davisson, L.D.; Flikkema, P.G. Fast single-element PN acquisition for the TDRSS MA system: Methods and technology. *IEEE Trans. Commun.* **1988**, *36*, 1226–1235. [CrossRef]
8. Warner, E.S.; Mulgrew, B.; Grant, P.M. Triple correlation analysis of m sequences. *Electron. Lett.* **1993**, *29*, 1755–1756. [CrossRef]
9. Gu, X.; Zhao, Z.; Shen, L. Blind estimation of pseudo-random codes in periodic long code direct sequence spread spectrum signals. *IET Commun.* **2016**, *10*, 1273–1281. [CrossRef]

10. Hill, P.C.J.; Comley, V.E.; Adams, E.R. Techniques for detecting and characterising covert communication signals. In Proceedings of the IEEE International Conference on Military Communications, Monterey, CA, USA, 3–5 November 1997.
11. Adams, E.R.; Gouda, M.; Hill, P.C.J. Statistical techniques for blind detection & discrimination of m-sequence codes in DS/SS systems. In Proceedings of the IEEE 5th International Symposium on Spread Spectrum Techniques and Applications, Sun City, South Africa, 4 September 1998.
12. Adams, E.R.; Gouda, M.; Hill, P.C.J. Detection and characterisation of DS/SS signals using higher-order correlation. In Proceedings of the IEEE 3th International Symposium on Spread Spectrum Techniques and Applications, Mainz, Germany, 25 September 1996.
13. Gouda, M.; Ali, Y. M-sequence Triple Correlation Function Co-set Summing and Code Image Print (CIP). In Proceedings of the IEEE 11th International Conference on Computer Modelling and Simulation, Cambridge, UK, 25–27 March 2009.
14. Burel, G.; Bouder, C. Blind estimation of the pseudo-random sequence of a direct sequence spread spectrum signal. In Proceedings of the 21st Century Military Communications. Architectures and Technologies for Information Superiority, Los Angeles, CA, USA, 22–25 October 2000.
15. Zhang, T.; Mu, A. A modified eigen-structure analyzer to lower SNR DS-SS signals under narrow band interferences. *Digit. Signal Prog.* **2008**, *18*, 526–533. [CrossRef]
16. Qiu, P.Y.; Huang, Z.T.; Jiang, W.L.; Zhang, C. Blind multiuser spreading sequences estimation algorithm for the direct-sequence code division multiple access signals. *IET Signal Process.* **2010**, *4*, 465–478. [CrossRef]
17. Zhang, T.; Zhang, A.M.C. Analyze the eigen-structure of DS-SS signals under narrow band interferences. *Digit. Signal Prog.* **2006**, *16*, 746–753. [CrossRef]
18. Qui, P.Y.; Huang, Z.T.; Jiang, W.L.; Zhang, C. Improved blind-spreading sequence estimation algorithm for direct sequence spread spectrum signals. *IET Signal Process.* **2008**, *2*, 139–146. [CrossRef]
19. Dominique, F.; Reed, J.H. Subspace based PN code sequence estimation for direct sequence signals using a simplified Hebb rule. *Electron. Lett.* **1997**, *33*, 1119–1120. [CrossRef]
20. Chen, H.; Liu, R.W. An On-Line Unsupervised Learning Machine for Adaptive Feature Extraction. *IEEE Trans. Circuits Syst.* **1994**, *41*, 87–98. [CrossRef]
21. Rohrer, R. Transformation Techniques for the Application of Liapunov's Direct Method to Linear Systems. *IEEE Trans. Circuit Theory* **1964**, *11*, 171–173. [CrossRef]
22. Deville, Y. A unified stability analysis of the Herault-Jutten source separation neural network. *Signal Process.* **1996**, *51*, 229–233. [CrossRef]