A Fast Clonal Selection Algorithm for Feature Selection in Hyperspectral Imagery

ZHONG Yanfei  ZHANG Liangpei

Abstract  Clonal selection feature selection algorithm (CSFS) based on clonal selection algorithm (CSA), a new computational intelligence approach, has been proposed to perform the task of dimensionality reduction in high-dimensional images, and has better performance than traditional feature selection algorithms with more computational costs. In this paper, a fast clonal selection feature selection algorithm (FCSFS) for hyperspectral imagery is proposed to improve the convergence rate by using Cauchy mutation instead of non-uniform mutation as the primary immune operator. Two experiments are performed to evaluate the performance of the proposed algorithm in comparison with CSFS using hyperspectral remote sensing imagery acquired by the pushbroom hyperspectral imager (PHI) and the airborne visible/infrared imaging spectrometer (AVIRIS), respectively. Experimental results demonstrate that the FCSFS converges faster than CSFS, hence providing an effective new option for dimensionality reduction of hyperspectral remote sensing imagery.

Keywords  hyperspectral; feature selection; artificial immune systems; artificial intelligence

CLC number  TP751

Introduction

Hyperspectral sensors provide a rich source of information to allow accurate separation of land cover classes [1]. Often, several hundred spectral samples are acquired for every pixel. Unfortunately, the number of pixels that are available for training the classifiers is often severely limited, and in combination with the high number of spectral bands, the occurrence of the so-called Hughes phenomenon [2] is almost inevitable, that is, when the number of input features exceeds a given limit for a fixed training-sample size, the classification accuracy will decrease. Furthermore, because hyperspectral sensors acquire images in very narrow spectral channels, the resulting high-dimensional feature sets may contain redundant information. An approach to mitigate these problems is dimensionality reduction, which can generally fall into feature extraction and feature selection [3]. Feature extraction is to find the transform from a higher dimension to a lower dimensional feature space with most of the de-
sired information content preserved [3]. However, since the data after feature extraction have been transformed and are no longer original data, some crucial and critical information may have been compromised and distorted [4].

An alternative to feature extraction is feature selection, which selects a subset from a larger number of features or variables used in classification while maintaining an acceptable classification accuracy [3,5]. Compared to feature extraction, feature selection has an advantage of preserving original information from the data. The sequential forward selection (SFS), sequential backward selection (SBS), sequential forward floating selection (SFFS) and the sequential backward floating selection (SFBS) techniques are the widely used sequential search methods [3,6]. A fast constrained search algorithm for feature selection in hyperspectral remote sensing images has been proposed, which is based on the search for constrained local extremes in a discrete binary space [7]. Some stochastic search algorithms have also been devised for feature selection. Genetic algorithm (GA) is a representative that has been applied to feature selection, which is evolved by genetic operators (e.g., crossover and mutation) to find the optimal feature subset [8].

In a previous work, a new stochastic search strategy was proposed for dimensionality reduction of hyperspectral remote sensing images [9], which is based on the clonal selection algorithm (CSA) [10,11] in artificial immune systems (AIS). AIS, which are inspired by the immune systems, use the immunological properties to support a wide range of applications [12,13]. CSA, derived from the clonal selection theory [14], is an effective method of AIS and successfully applies to pattern recognition, multi-modal optimization and classification [10,11]. In the new strategy, dimensionality reduction is formulated as an optimization problem that searches an optimum with less number of features in a feature space and the hyperspectral imagery is transformed to the immune computation space by the immune encoding and the shape-space model. In line with this novel strategy, a feature subset search algorithm, clonal selection feature selection algorithm (CSFS), and a feature weighting algorithm as well as a clonal selection feature weighting algorithm (CSFW), have been developed [9]. CSFS and CSFW may automatically find out the optimal feature subset by the immune operators, i.e., clone, selection, mutation, to perform the feature selection and dimensionality reduction of hyperspectral remote sensing data. In the previous work [9], although CSFS and CSFW have good results for dimensionality reduction, their computation costs need to be decreased for hyperspectral imagery. In this paper, a fast feature selection algorithm is proposed to improve the CSFS, namely fast clonal selection feature selection algorithm (FCSFS), by designing new immune operators and is compared with the CSFS [9] using the hyperspectral remotely sensed images acquired by the push-broom hyperspectral imager (PHI) and the airborne visible/infrared imaging spectrometer (AVIRIS). Experimental results have shown that the FCSFS may effectively improve the convergence rate of CSFS for decreasing the computational costs, while just having the slight difference of the accuracy, and thus provides an effective option for hyperspectral dimensionality reduction.

The rest of this paper is organized as follows. Section 1 describes the CSFS algorithm. Section 2 reviews the fast CSFS algorithm. In Section 3, the PHI and AVIRIS hyperspectral data used for experiments are described and the experimental results are provided. Finally, Section 4 concludes the paper.

1 Clonal selection feature selection algorithm (CSFS)

In the previous work, a feature selection algorithm based on the clonal selection theory and algorithm [10,11], i.e., CSFS [9], was developed for dimensionality reduction of hyperspectral imagery. The problems of feature selection are described as follows. A hyperspectral remote-sensing dataset $X = \{x^1, x^2, \cdots, x^{N_b}\}^T$ through $N_s$ bands is observed. The objective of feature selection is to reduce the number of features utilized to characterize patterns by selecting, through optimization in terms of a criterion function $F$ (e.g., maximization of a separability index or minimization of an error bound), a good subset $S$ of $N_s$ features,
with \( N_m < N_s \), without significantly degrading the performance of the resulting classifier:

\[
S = \{s^1, s^2, \ldots, s^{N_s}\}^\top, S \subset X
\]  

(1)

The criterion function \( F \) is computed by using a preclassified reference set of patterns (e.g., training set). The value of \( F \) depends on the features included in the subset \( S \), \( F = F(S) \). In CSFS, the average Jeffries-Matusita (JM) distance \[^{[15]}\] was used as the criterion function, which is a common class separability index utilized by the remote sensing community.

The JM distance between classes \( i \) and \( j \) is an affinity/distance measure of separability. The smaller the JM distance, the more difficult it is to separate the classes and vice versa.

To apply the clonal selection algorithm to hyperspectral feature selection, the entire set of features is represented by a discrete binary space. In this search space, each point represents an individual band. Value “0” in the \( j \)-th position indicates that the \( j \)-th feature is not included in the corresponding feature subset; value “1” in the \( j \)-th position indicates that the \( j \)-th feature is included in the corresponding feature subset.

For example, in a simple case with \( N_m = 3 \) and \( N_s = 5 \) features, the binary vector \( b = (1,1,0,0,1) \) indicates the feature subset has the first, second, and fifth features.

\[
b = (1,1,0,0,1) \Leftrightarrow S = \{x^1, x^2, x^5\}
\]  

(5)

The criterion function \( F \) can be viewed as a scalar function defined in the discrete binary space. Without loss of generality, there is a case in which the criterion function has to be maximized. Thus, the dimensionality reduction problem of selecting \( N_m \) out of \( N_s \) features can be suitably formulated as an optimization problem to find the global maximum of the criterion function.

The CSFS algorithm consists of the following steps \[^9\]:

**Step 1  Initialization**

A first antibody population \( (AB) \) including \( N \) antibodies is generated with the value of each bit \( ab^i_j \) in each antibody \( ab_i \in AB \) assigned 1 or 0 according to the number of selected subset features, \( N_m \).

\[
ab^i_j = \begin{cases} 
1, & \text{if } t = k \\
0, & \text{otherwise}
\end{cases}
\]  

(6)

\[
i = 1,2,\ldots,N; t = 1,2,\ldots,N_b
\]

where \( N \) is the number of the initial antibody population, \( N_s \) is the number of features or bands, function \( \text{Irando}(1,N_s) \) returns a random integer value within the range \( [1, N_s] \) using a uniform distribution. Eq. (8) indicates that the sum of selected features is equal to \( N_m \), a user-defined feature subset size.

**Step 2  Cycle of the generations**

1) Calculation of affinity

According to the initial antibody population, the affinity of all \( N \) \( ab \)’s in the antibody population \( AB \) are calculated using the criterion function \( F = F(ab) \) using Eq.(2).

2) Selection

From \( AB \), ‘\( n \)’ highest affinity antibodies are selected to compose a new set \( AB_{sel} \) of high-affinity antibodies and the highest affinity memory cell is found. (memory cell, \( mc \)).

3) Clone
After receiving antibody individuals closer to the solution, the next generation should mainly be derived from the better-fitting individuals. Thus, the \( n \) selected \( ab \)'s are cloned based on their antigenic affinities, generating the clone set \( C \). The total number of clones-generated \( N_c \) is defined as follows:

\[
N_c = \sum_{i=1}^{N_b} \text{round}(\beta \times N)
\]  

(9)

where \( \beta \) is a multiplication factor, \( N \) is the total number of antibodies, \( \text{round}(\cdot) \) is the operator that rounds its argument.

4) Mutation

Provide each \( ab \) in the clone set \( C \) with the opportunity to produce mutated offspring \( C^* \). The higher the affinity, the smaller the mutation rate. The mutation rate of each cloned antibody is calculated adaptively according to the affinity using Eq. (10).

\[
p_m = \exp(-2 \times F'(ab))
\]  

(10)

\[
F'(ab) = \frac{F(ab) - \min(F(ab))}{\max(F(ab)) - \min(F(ab))}
\]  

(11)

where \( p_m \) is the mutation rate of each \( ab \), \( 2 \) is the empirical value to control the decay, and \( F'(ab) \) is the affinity according to Eq. (11). In Eq. (11), the range of the mutation rate \( p_m \) is \((0,1]\).

The mutation process is as follows:

(a) An integer string \( B \) consisting of integer numbers to represent the corresponding selected features is obtained by decoding the antibody \( ab \). For example, as for \( N_m = 5 \) and \( N_s = 10 \), a binary string 1011001010 corresponds to the integer string \( B = \{1,3,4,7,9\} \);

(b) For the integer string \( B \), the non-uniform mutation is used in the mutation procedure, which is one of the most widely used mutation operators. From a point \( B_i \), the mutated point \( B_i' \), is created with mutation rate \( p_m \) as follows:

\[
B_i' = \begin{cases} 
  B_i + \Delta(q,b_i-B_i), & \text{if } \xi \leq 0.5 \\
  B_i + \Delta(q,B_i-a_i), & \text{otherwise}
\end{cases}
\]  

(12)

where \( q \) is the current generation number and \( \xi \) is a uniformly distributed random number between 0 and 1. \( a_i \) and \( b_i \) represent the maximum and minimum of each element in the integer string \( B \) and are equal to \( N_s \) and 1, respectively. In the mutation process, the value of each element is not equal to any other elements existing in the string \( B \). Function \( \Delta(q,u) \) returns a value in the range \([0,u]\) such that \( \Delta(q,u) \) approaches to zero as \( m \) increases. The function is defined as:

\[
\Delta(q,u) = u(1 - r^{\frac{q}{\lambda}}) 
\]  

(13)

where \( r \) is a random value within the range \([0,1]\), \( q \) is the iteration number, \( \lambda \) is the maximal iteration number, and \( \lambda \) is a parameter to decide the nonconforming degree.

5) Re-calculation of affinity

Calculate the affinity \( F'(ab) \) of the matured clones \( C^* \).

6) Re-selection

From the mature clone set \( C^* \), reselect the \( n \) \( ab \)'s with the highest affinity to replace the \( n \) \( ab \)'s with the lowest affinity in \( AB \). Select the highest affinity \( ab \) in \( C^* \) to be a candidate memory cell, \( mc_{\text{candidate}} \). If the affinity of \( mc_{\text{candidate}} \) is higher than the memory cell, \( mc \), then \( mc_{\text{candidate}} \) will replace \( mc \) and become a new memory cell.

7) Displace

In order to replace the \( d \) lowest affinity \( ab \)'s from \( AB \), \( d \) new antibodies are produced by a random process. This step may increase the diversity of the antibody population.

Step 3 Stopping condition

When the number of iterations reaches the user-defined number or the change of memory cell between two consecutive iterations is less than a change threshold, terminate the execution of the algorithm. Otherwise, return to Step 2 until the stop criteria are satisfied.

Finally, the proposed algorithm outputs the value of the memory cell and obtains the subset space through transformation from the binary space.

2 Fast clonal selection feature selection algorithm (FCSFS)

In CSFS algorithm\[^9\], each antibody was mutated with the mutation rate via non-uniform mutation,
which produces new properties of mutated antibodies. Non-uniform mutation operators have the feature of searching the space uniformly at the early stage and very locally at the later stage\cite{16}. As is shown by mathematical analysis\cite{17}, this mutation operator prefers parameter values in the center of the corresponding feasible region, which leads to problems if the optimum is situated near the feasible region’s boundaries.

In this paper, a fast feature selection algorithm based on Cauchy mutation operator is proposed to improve the CSFS, namely fast clonal selection feature selection algorithm (FCSFS). Cauchy mutation is an efficient search operator for a large class of multimodal function optimization problems, which has the ability to escape local minima \cite{18}. Cauchy mutation may increase the probability of finding a near-optimum by long jumps when the distance between the current search point and the optimum is large, but decrease the probability when such distance is small\cite{18}. Because Cauchy mutation has the long jumps, it is more suitable for feature selection in very high-dimensional remote sensing images, e.g., hyperspectral imagery.

The general formula for the probability density function of the Cauchy distribution is defined by

\[
    f(x) = \frac{1}{\pi s^2(1 + ((x-t)/s)^2)}, \quad -\infty < x < \infty
\]

where \( t \) is the location parameter and \( s \) is the scale parameter \cite{19}. The case where \( t=0 \) and \( s=1 \) is called the standard Cauchy distribution. The equation for the standard Cauchy distribution reduces to

\[
    f(x) = \frac{1}{\pi(1+x^2)}
\]

Since the general form of probability density function can be expressed in terms of the standard distribution, all subsequent formulas in the paper are given for the standard form of the function. The corresponding distribution function is

\[
    F(x) = \frac{1}{2} + \frac{1}{\pi} \arctan(x)
\]

The shape of \( f(x) \) resembles that of the Gaussian density function but approaches the axis so slowly that an expectation does not exist\cite{18}. As a result, the variance of the Cauchy distribution is infinite. Fig.1 shows the probability density function of the Cauchy distribution.

The FCSFS uses the following equation to mutate the clonal antibodies

\[
    B_i' = B_i + \sigma'(i)C(0,1) \quad i = 1, 2, \ldots, N_m
\]

\[
    \sigma'(i) = \sigma(i)\exp(\tau N(0,1) + \tau' N(0,1))
\]

for \( i = 1, \ldots, N_m \), where \( B_i, B_i', \sigma'(i) \) and \( \sigma(i) \) denote the \( i \)th component of the integer string \( B \) (see also Eq. (12)), respectively. \( C(0,1) \) denotes a Cauchy random number centered at zero with a scale parameter of one, and is generated anew for each value of \( i \). \( N(0,1) \) denotes a normally distributed one-dimensional random number with mean zero and standard deviation one. \( N(0,1) \) indicates that a different random number is generated for each value of \( i \).

The factors \( \tau \) and \( \tau' \) are commonly set to \( 1/(2\sqrt{n}) \) and \( 1/(\sqrt{2n}) \), respectively\cite{19,20}.

To better show the ability of Cauchy mutation, the Gaussian functions were also shown and compared with Cauchy mutation by plotting them in the same scale in Fig.1. It is clear from Fig.1 that Cauchy mutation is more likely to generate an offspring further away from its parent than Gaussian mutation due to its long flat tails. It is expected to have a higher probability of escaping from a local optimum or moving away from a plateau, especially when the “basin of attraction” of the local optimum or the plateau is large relative to the mean step size. On the other hand, the smaller hill around the center in Fig.1 indicates that Cauchy mutation spends less time in exploiting the local neighborhood and thus has a weaker fine-tuning ability than Gaussian mutation in
small to mid-range regions\textsuperscript{[18]}. The results of the experiments have shown that the Cauchy mutation was successfully utilized to evolutionary programming\textsuperscript{[18]}. That encourages us that FCSFS has a much better performance than CSFS.

3 Experiments and analysis

The FCSFS and CSFS algorithms were tested using different types of hyperspectral remote sensing images acquired by the pushbroom hyperspectral imager (PHI) and the airborne visible/infrared imaging spectrometer (AVIRIS). Consistent comparisons were also carried out among FCSFS and the traditional dimensionality reduction algorithms: SFS (sequential forward selection) and SFFS (sequential forward floating selection).

A. Experiment 1: Xiaqiao PHI

The dataset used in this experiment was acquired from the Xiaqiao test site, a mixed agricultural area in China, using the PHI. 80 bands of the PHI image (340×390 pixels) were tested, and their spectral ranges were from 0.417 to 0.854μm. Fig.2 shows the experimental PHI image cube. The ground truth data were collected as follows. Nine representative classes, i.e., corn1 (713 samples), corn2 (217 samples), corn3 (322 samples), vegetable-sweet potato (464 samples), vegetable-cabbage (253 samples), soil (1368 samples), float grass (220 samples), road (662 samples) and water (659 samples) were considered.

The setting of primary running parameters in CSFS and FCSFS are the same as follows: the number of maximum iterations is equal to 100, antibody population size $N=50$, the number of highest affinity $ab$, $n=50$, clone multiplication factor $\beta=0.02$ (see also Eq. (9)), the number of displace antibody, $d=0$. In order to test these algorithms, approximately half of the available samples were used as the training dataset, whereas another half of the available samples served as the test data set.

The execution time of the algorithms is the important characteristic. In our case, it is in the form of processor ticks (1 millisecond (msec)) spent in the user space. Fig.3 shows the execution times with different feature subset sizes provided by the four algorithms. For every number of selected features from 2 to 79, SFS is the fastest. While the number of selected features is over 67, SFFS is the slowest. Compared with CSFS, FCSFS improves the convergence rate of CSFS with less computational costs. The improvement is more obvious when the number of selected features is changed from 2 to 42.

Fig.3 Execution times required by different algorithms in experiment 1

Fig.4 depicts the values of the criterion function computed on the subsets in the training process provided by SFS, SFFS, CSFS and FCSFS to different numbers of selected features from 2 to 79. All algorithms were described by the relationship curves between the predefined number of features in the subset.
and the corresponding JM distance. To better present the experiment results, the JM distance of SFS was used as a reference, that is, the values of the criterion functions provided by SFFS, CSFS, and FCSFS are divided by the corresponding values obtained by SFS. For example, if the CSFS and SFS provided the same JM distance values, the value on Fig.4 equals 1.

For a more detailed comparison among different algorithms: SFS, SFFS, CSFS, and FCSFS, all the labeled training and test samples were classified using the dimensionality reduction algorithms with the increase of selected feature numbers. The maximum likelihood (ML) classifier was employed in this case. Fig.5 and Fig.6 illustrate the overall accuracy for training data and test data, respectively. For convenience of description, SFS-ML denotes the maximum likelihood classifier with the feature subsets provided by the SFS algorithm. Other notations can be inferred by analogy.

The Hughes phenomenon can be observed in Fig.6: when the number of input features exceeds around 50, the classification accuracy decreases. As can be found from Fig.4, Fig.5 and Fig.6, the SFFS, the proposed CSFS and CSFW algorithms perform better than SFS for all the selected subset features. Comparing the results of CSFS and FCSFS with those of SFFS, the two algorithms make some improvements over SFFS. In particular, when the number of selected features is below 50, the improvement is greater. The highest overall accuracies achieved by CSFS-ML and FCSFS-ML using the test data are respectively 93.07% and 92.97%. Combining with Fig.3, a comparison between the two algorithms based on clonal selection algorithms shows that FCSFS can converge quickly and needs less convergence costs. Although FCSFS decreases some classification accuracy, the differences of the accuracy can be neglected when the number of selected features is larger than 50. In addition, in the experiment, FCSFS presents a tendency towards premature convergence, some runs reaching their peak by around the fifteenth generation. The reasons are as follows: the larger jumps using Cauchy mutation are beneficial when the current solution is far away form the global optimum or a better optimum, but such large jumps near the global optimum point are undesirable. Fortunately, because of the complexity of class distribution in remote sensing imagery, the solution is often far away from the global optimum. Furthermore, the difference of the accuracy should be also acceptable as the results of FCSFS are still better than SFFS and SFS. From the above, it can be concluded that FCSFS may have better potential than CSFS, which can decrease the computation costs with the acceptable accuracy.

B. Experiment 2: Indian Pine AVIRIS

The image data used in this experiment refers to the agricultural area of Indian Pine in the northern part of Indiana [21]. The image (145×145 pixel) was acquired by the airborne visible/infrared imaging spectrometer (AVIRIS) in June 1992 and was downloadable from the website (http://dynamo.ecn.urdue.edu/~biehl/MultiSpec/documentation.html).

The dataset was composed of 220 spectral channels and their spectral ranges were from 0.4 to 2.5 μm in approximately 10 nm bandwidths.

Fig.7 shows the experimental AVIRIS image. Ten
most representative land-cover classes were considered and the reference data were collected, i.e., Corn-notill (C1) (1,434 samples), Corn-min (C2) (834 samples), Grass/Pasture (C3) (497 samples), Grass/Trees (C4) (747 samples), Hay-windrowed (C5) (489 samples), Soybeans-notill (C6) (986 samples), Soybeans-min (C7) (2,468 samples), Soybeans-clean (C8) (614 samples), Woods (C9) (1,294 samples) and Bldg-Grass-ree drives (C10) (380 samples).

In this experiment, the method to select the training and test dataset is the same with experiment 1: approximately half of the available samples were used as the training dataset, whereas the entire set of samples were used as the test dataset in the experiment. The values of the primary parameters were the same with experiment 1.

The execution time is shown in Fig.8 and SFS runs the fastest. SFFS is faster than CSFS for the number of selected features below 25, whereas for larger number of features, over 37, SFFS is the slowest. Compared with CSFS, FCSFS needs less computation costs to convergence for all the selected subset features because the Cauchy mutation has the large jump. When the number of selected features is changed from 3 to 25, FCSFS even has the similar costs with the traditional feature selection algorithm, SFFS.

Fig.9 shows the values of the criterion function computed on the subsets in the training process provided by SFS, SFFS, CSFS and FCSFS. As in Fig. 4, the ratio of JM distance was used for the Y-axis. For a more detailed verification of the above results, the set of the labeled samples was classified using the algorithms: SFS-ML, SFFS-ML, CSFS-ML and FCSFS-ML. As can be seen from Fig.9, SFFS, CSFS, FCSFS make some improvements over SFS for the number of selected features below 24. The improvement for six features is the most significant. In addition, CSFS and FCSFS have a greater improvement than SFFS when the number of selected features is changed from 2 to 24. Compared with CSFS, FCSFS has the lower value of JM distance when the number of selected subset features is changed from 2 to 15, whereas for larger numbers of features, differences can be ignored.

Fig.10 and Fig.11 illustrate the comparative results using the overall accuracy for training data and test data, respectively. It is observed from Fig.10 and Fig.11 that the FCSFS-ML and CSFS-ML classifiers produce better classification results not only on the training data but also on the test data than traditional algorithms, such as SFS-ML, SFFS-ML. SFS-ML and SFFS-ML obtain the similar results for different numbers of selected features. As can be seen from Fig.10, the overall accuracy of the training data provided by FCSFS-ML and CSFS-ML is over 90%,
when the number of selected features is larger than 28 and 27, respectively, whereas SFS-ML and SFFS-ML require 30, 29 features to arrive at the same value, respectively. Compared with CSFS, FCSFS has the lower accuracy when the number of selected features is changed from 2 to 16 and has the similar results when the number of selected features is larger than 16, where the difference of the accuracy is less than 0.05%. As shown in Fig. 11 for test accuracy, FCSFS-ML and CSFS-ML outperform SFS-ML and SFFS-ML. CSFS has some improvements over FCSFS for the number of selected features below 16. When the number of features is larger than 16, the plots of the accuracy using CSFS and FCSFS nearly overlap each other in Fig. 11 and the differences can be ignored.

Comparing the results shown in Figs. 8-11, although FCSFS has the larger difference with lower accuracy than CSFS when the number of selected subset features is changed from 2 to 15, it improve the convergence rate of CSFS and needs less computation costs, which is important to hyperspectral remote sensing image processing. Moreover, the difference should be also acceptable as the results of FCSFS are still better than SFFS and SFS. The above analysis of the experimental results shows that CSFS and FCSFS fare the best among the traditional algorithms tested for dimensionality reduction of hyperspectral remote sensing imagery. In addition, FCSFS improves the convergence rate of CSFS using Cauchy mutation with the acceptable accuracy, which has a slight decrease with CSFS.

4 Conclusion

This paper proposed an FCSFS based on CSFS and Cauchy mutation. The mutation step is critical to the traditional CSFS as it helps escape local optimums, which is complemented by non-uniform mutation in CSFS. Non-uniform mutation operators have the feature of searching the space uniformly at the early stage and very locally at the later stage. However, this mutation operator prefers parameter values in the center of the corresponding feasible region, which leads to problems if the optimum is situated near the feasible region’s boundaries. Cauchy mutation may converge quickly using the long jumps, which increase the probability of finding the near-optimum when the distance between the current search point and the optimum is large. Because Cauchy mutation has the long jumps, FCSFS is more suitable for CSFS than non-uniform mutation in very high-dimensional remote sensing images, e.g., hyperspectral imagery. Experimental results in this study consistently demonstrate that the FCSFS and CSFS provide better results than the traditional SFS and CSFS algorithm. In addition, it is important that FCSFS converges faster than CSFS with the similar feature subset and the acceptable classification accuracy, and provide effective new options for dimensionality reduction of hyperspectral remote sensing imagery. In the experiments, FCSFS sometimes presents a tendency towards premature convergence. Our future work will explore the new method to avoid this premature convergence of FCSFS.
Acknowledgment

The authors would like to thank the research group supervised by Prof. D. Landgrebe, Purdue University, West Lafayette, IN, for providing the free downloads of the AVIRIS image.

References

[1] Jensen A C, Solberg A S (2007) Fast hyperspectral feature reduction using piecewise constant function approximations [J]. *IEEE Geosciences and remote sensing letters*, 4(4): 547-551

[2] Hughes G F (1968) On the mean accuracy of statistical pattern recognizers [J]. *IEEE Trans. Inf. Theory*, IT-14 (1): 55-63

[3] Webb A R (2002) Statistical pattern recognition (2nd ed) [M]. New York: John Wiley & Sons, Inc.

[4] Chang C -I, Wang S (2006) Constrained band selection for hyperspectral imagery [J]. *IEEE Trans. Geosci. Remote Sensing*, 44 (6): 1575-1585

[5] Jain A, Zongker D (1997) Feature selection: evaluation, application, and small sample performance [J]. *IEEE Trans. Pattern Anal. Machine Intell.*, 19: 153-158

[6] Pudil P, Novovicova J, Kittler J (1994) Floating search methods in feature selection [J]. *Pattern Recognit. Lett.*, 15:1 119-1 125

[7] Sebastiano B, Lorenzo L (2001) A new search algorithm for feature selection in hyperspectral remote sensing images [J]. *IEEE Trans. Geosci. Remote Sensing*, 39: 1 360-1 367

[8] Raymer M L, Punch W F, Goodman E D, et al. (2000) Dimensionality reduction using genetic algorithm [J]. *IEEE Trans. Evolutionary Computation*, 4(2): 164-171

[9] Zhang L, Zhong Y, Huang B, et al. (2007) Dimensionality reduction based on clonal selection for hyperspectral imagery [J]. *IEEE Transactions on Geoscience and Remote Sensing*, 45(12): 4 172-4 185

[10] De Castro L N, Von Zuben F J (2000) Clonal selection algorithm with engineering applications [C]. Proc GECCO’s 00, Nevada, USA

[11] De Castro L N, Von Zuben F J (2002) Learning and optimization using the clonal selection principle [J]. *IEEE Transactions on Evolutionary Computation*, 6: 239-250

[12] Dasgupta D (1999) Artificial immune systems and their applications [M]. Berlin: Springer-Verlag

[13] De Castro L N, Timmis J (2002) Artificial immune systems: a new computational intelligence approach [M]. London: Springer-Verlag

[14] Burnet F M (1959) The clonal selection theory of acquired immunity [M]. Cambridge: Cambridge University Press

[15] Bruzzone L, Roli F, Serpico S B (1995) An extension of the Jeffreys-Matusita distance to multiclass cases for feature selection [J]. *IEEE Trans. Geosci. Remote Sensing*, 33: 1 318-1 321

[16] Zhao X, Gao X, Hu Z (2007) Evolutionary programming based on non-uniform mutation [J]. *Applied Mathematics and Computation*, 192: 1-11

[17] Neubauer A (1997) Adaptive non-uniform mutation for genetic algorithms [M]/ Reusch B (Ed.). Computational Intelligence Theory and Applications. Berlin: Springer-Verlag

[18] Yao X, Liu Y, Lin G (1999) Evolutionary programming made faster [J]. *IEEE Transactions on Evolutionary Computation*, 3 (2): 82-102

[19] Feller W (1971) An introduction to probability theory and its applications [M]. New York: John Wiley & Sons

[20] Chellapilla K (1998) Combining mutation operators in evolutionary programming [J]. *IEEE Transactions on Evolutionary Computation*, 2(3): 91-96

[21] Hsieh P F, Landgrebe D (1998) Classification of high dimensional data [D]. Indiana, West Lafayette: School Elect. Comput. Eng., Purdue Univ.