University of Leeds

SCHOOL OF COMPUTER STUDIES

RESEARCH REPORT SERIES

Report 93.9

Some Further Results of Three Stage ML Classification Applied to Remotely Sensed Images[1]

by

N B Venkateswarlu, S. Balaji,[2]
P S V S K Raju[2] & R D Boyle
Division of Artificial Intelligence

March 1994

[1] to appear in Pattern Recognition

[2] Computer Science, BITS, Pilani, India

Abstract

Recently, a three stage Maximum Likelihood (TSML) classifier[1] has been proposed to reduce the computational requirements of the ML classification rule. Some modifications are proposed here further to improve this fast algorithm. Winograd's method is proposed for use with range calculations, and is also used with Lower Triangular and Unitary canonical form approaches [2] in calculating quadratic forms. New types of range are derived by expanding the discriminant function which are then used with a TSML algorithm to identify their usefulness in eliminating groups at stages I & II. The use of pre-calculated values is proposed to obviate some multiplications while calculating the ranges. Further, threshold logic [3] is used with an old and a modified TSML classifier and its effectiveness observed in further reducing computation time. Performance of the old and the modified TSML algorithms is studied in detail by varying the dimensionality and number of samples. For the purpose of experiment, 6 channel thematic mapper (TM) and randomly generated 12 dimensional data sets are used. A maximum speed-up factor of 4-8 is observed with these data sets. These experiments are also repeated with modified maximum likelihood and Mahalanobis distance classifiers to inspect CPU time requirements.

Keywords

Quadratic Form Range	Classification
Thresholds	Unitary Canonical Form
Winograd's Method	Partial Sum
Speed-up	Lower Triangular Canonical Form

1 INTRODUCTION

In recent decades, analysis of earth resources has generally been carried out with the help of remotely sensed images from Earth resources satellites such as LANDSAT MSS, TM, SPOT, IRS, TIMS, etc.,. The analysis of the remotely sensed data is usually achieved by machine-oriented pattern recognition techniques, of which one of the most widely used is classification based on maximum likelihood (ML), assuming Gaussian distribution of the data. A serious problem of the ML classification rule is its long processing time. This computational cost may become an important problem if remotely sensed data (images) of a large area are to be analysed, which is a common feature in Geographic Information Systems (GIS) development and analysis. Further, the problem will be exacerbated in the case of data from future multichannel satellites such as the 192 channel HIRIS (High resolution Infrared Imaging Spectrometer) and 224 channel AVIRIS (Airborne Visible and Infrared Imaging Spectrometer). Moreover, the fine pixel and grey level resolution of future satellites will further exacerbate this problem. Analysis of multi temporal images and analysis aided by ancillary data demands additional computational efforts with an ML algorithm.

Efforts to reduce processing time have been pursued in two ways; hardware related developments and software oriented developments. Hardware solutions include the use of enhanced numeric co-processors, use of processors with increased clock speed, and use of parallel computer architectures. Settle and Briggs [(4)] and Fu [(5, 6)] implemented an ML classifier on parallel computer systems to speed-up the image classification task. Swain et al., [(7)] implemented a contextual classification algorithm for remotely sensed imagery on a multi processor system. Similarly, Garg et al., [(8)] implemented the ML classification rule on a 4 transputer array.

A number of authors have proposed numerous software related methods to reduce the computational requirements of an ML classifier. Feiveson[(3)] proposed a procedure to reduce computing time by employing thresholding logic. Here, using the training data, thresholds between pairs of groups are calculated before the classification task. For a random pixel vector, the most likely class is selected based upon some prediction (for example, based upon maximum a priori probability or autocorrelation), and for that group a probability density function (pdf) is calculated. If this pdf value is greater than the threshold between the most likely group and any other group, calculation of the pdf for the other group is omitted, resulting in a reduction of computing time. Performance of this method will depend on the accuracy of prediction. Feiveson[(3)] observed a speed-up of 2 with 4 channel LARS C1 Flight data on a Univac 1180 computer system.

Eppler[(2)] proposed some methods to reduce the number of computations while classifying a pixel vector by using fewer computations in calculating the pdf of each group. In his algorithm, he proposed the quadratic term in the pdf to be a monotonically increasing sum of squares by representing the inverse covariance matrix in two types of canonical form, known as Unitary and Lower Triangular canonical forms (UCF and LTCF). This formulation gave the advantage of discontinuing the computations of a particular class hypothesis when the partial sum exceeds the smallest value obtained for other classes already tested. Eppler[2] observed a speed-up of 6.7 with 4 channel LARS C1 flight data. Kriegler[9] also proposed a method similar to Eppler[2] based on the inverse of the correlation matrix and his procedure is reported to be existing in hardwired fashion in the MIDAS image processing system.

Some authors employed feature selection algorithms to reduce the computational requirements of the ML algorithm.[10, 11, 12, 13] It was also observed that the classification accuracy is also increased by the use of transformed variables.[14] Further, decision tree classifiers are proposed to reduce the processing time.[15, 16, 17, 18] In some of these decision tree implementations, groups are eliminated using a smaller number of features at each node in the tree.

Odell and Duran,[19] Mather,[20] Bolstad and Lillesand,[21] and Ahearn and Wee[22] proposed methods based on look-up table operations. In a large image there will be certain combinations of pixel values which occur numerous times, and this redundancy is exploited in the development of look-up table algorithms. Shlien[23] found that the number of unique pixel vectors in a LANDSAT MSS image is surprisingly small, being measured in the order of thousands. If these unique pixels are classified by using an ML algorithm and this information is stored in a look-up table, it can be effectively used while classifying each of the image pixels by simply reading its classification result from the look-up table if it matches with any of the unique pixels. The main drawback of this procedures is the requirement of additional memory to store look-up tables. However, this is not going to be a serious problem in future due to developments in memory related technology.

Lee and Landgrebe[24] proposed a multistage ML classification algorithm in which some groups are terminated at each stage using a small number of variables (bands) using some type of threshold information. Experiments show a speed-up of 3-7 depending on the number of classes and features. Minskii and Chizhevskii[25] and Venkateswarlu and Raju[1] proposed fast ML algorithms which use the ranges of a quadratic form.[26] Recently, Venkateswarlu and Raju[27, 28] proposed some fast ML algorithms utilising Winograd's logic, threshold logic[3].

In the present work, some modifications are proposed to a TSML classifier[1] and its performance is compared with the TSML classifier and ML algorithms. It was observed that it is not always possible to use a LTCF approach at stage III in the TSML algorithm because of difficulties in the calculation of the lower triangular matrix from the relationship given by Eppler[2]. To cope with this type of group, the UCF approach[2] is identified as useful in the study, but the computational complexity of this algorithm in calculating quadratic terms is identified to be the same as the direct matrix multiplication approach.[2] However, this method gives the possibility of using a partial sum approach[29] which is not possible in the direct approach. Further, the number of multiplications used in calculating quadratic forms using a LTCF and a UCF are reduced by using a mathematical development known as Winograd's method, with the help of which the number of multiplications is reduced by approximately half. This method is also used to calculate ranges of each group's discriminant functions with less computations compared to the direct approach. New ranges of the discriminant function are derived by expanding the discriminant function equation; these ranges are used in developing another new TSML classifier and its performance is studied relative to both the original TSML classifier(1) and the ML classifier. Computational time requirements of the proposed algorithms are compared with ML algorithm with the help of remotely sensed and simulated data sets. Thresholds(3) are used before the first stage of the TSML classifier and it is observed that they further reduce the CPU time requirements. Performance of these algorithms is studied by varying the dimensionality and number of samples on two computer systems (PC/AT and PC/XT). This logics are also used with modified maximum likelihood (MML) and Mahalanobis distance classifiers to reduce their CPU time requirements.

2 MAXIMUM LIKELIHOOD CLASSIFIER

Let $\omega_1, \ldots, \omega_m$ denote m distinct populations (classes) with known d dimensional probability density functions $p_1(X), \ldots, p_m(X)$, respectively. The a priori probabilities that an observation is selected from populations $\omega_1, \ldots, \omega_m$ are denoted by $q_1, \ldots, q_m$ respectively. According to the Bayesian ML classification rule,(30) assuming equal costs for misclassifications, a random vector X (of dimension d) is classified as class ω_k if

$$q_k p_k(X) = \max\{q_i p_i(X)\} \; for \; i = 1, \ldots, m \tag{1}$$

Assuming equal a priori probabilities for all the classes, decision rule (1) becomes (see Swain and Davis(30) for derivation) :

$X \in \omega_k$ if

$$d_k(X) = \min\{d_i(X)\}, i = 1, \ldots, m \tag{2}$$

Here

$$\begin{aligned} d_k(X) &= B_k + Q_k(X) && (3) \\ Q_k(X) &= (X - M_k)^T \textstyle\sum_k^{-1} (X - M_k) && (4) \\ B_k &= \ln |\textstyle\sum_k| \end{aligned}$$

Here, $M_k, \sum_k$ are the mean vector and covariance matrices of the k^{th} class which are calculated from the training data. $\sum_k$ is a symmetric positive definite matrix. $\sum_k^{-1}, |\sum_k|$ are the inverse and determinant of the covariance matrix $\sum_k$.

In equation (3), $d_k(X)$ is generally called the discriminant function and $Q_k(X)$ the quadratic term. Calculation of the quadratic term makes the ML classification rule computationally inefficient. Direct calculation of the quadratic term for a group requires *d(d+1)* multiplications and about the same order of additions and subtractions.[31] So, classification of a pixel vector X into one of the m groups necessitates $md(d+1)$ multiplications. This direct implementation is common in most image processing software systems.[31]

3 DEVELOPMENT OF FAST ML ALGORITHMS

Here, in the development of fast ML classifiers, linear algebraic rules related to positive definite matrices and quadratic forms are used . The LTCF and UCF approaches are used in the calculation of quadratic terms which facilitates the use of partial sum logic. [29, 32, 33] A linear algebra theorem which defines the ranges of the quadratic form is used as a prime group elimination criterion. Ranges of the quadratic term are used for group elimination in the search process instead of the actual quadratic term values; details are mentioned in the following. By the use of a LTCF approach, the number of multiplications required to calculate the quadratic term may be reduced by approximately one half, but the computational complexity of the UCF approach is identified to be the same as the literal ML algorithm. In order further to reduce the computational requirements of these two approaches, Winograd's method is proposed.

Some of the proposed algorithms are heuristic in nature and some have strong mathematical support. In the case of heuristic algorithms it is not possible to estimate the theoretical speed-up, whereas it is possible with methods which have mathematical support. For these algorithms the theoretical speed-up is defined as the ratio of the number of multiplications required to classify a pixel vector with an ML algorithm to the number of multiplications required for the proposed algorithm for the same operation. It is not possible to count (estimate) the number of multiplications for heuristic algorithms such as partial sum logic because the effectiveness of these algorithms depends on the nature of the data (classes). The theoretical speed-up will give an estimate of the efficiency of the algorithm. In order to discover the actual advantage of the proposed algorithm, some data will be classified with the ML and proposed methods, and the ratio of CPU time requirements of these methods provides the actual effectiveness (actual or observed speed-up) of the proposed algorithm. This actual speed-up can be used to decide which logic is superior in reducing the CPU time requirements of ML classification rule.

3.1 Lower Triangular Canonical Form (LTCF) Approach

The matrix $\sum_k^{-1}$ can be represented in terms of a lower triangular matrix L_k as:[2]

$$\textstyle\sum_k^{-1} = L_k^T L_k \tag{5}$$

Using this lower triangular matrix, the quadratic function in equation (4) can be calculated with equations (6), (7) and (8).

$$Y_k = \left\{ \begin{array}{c} y_{k1} \\ y_{k2} \\ . \\ . \\ . \\ y_{kd} \end{array} \right\} \quad (6a)$$

$$Y_k = L_k^T(X - M_k) = \left\{ \begin{array}{c} V_{k1}^T \\ V_{k2}^T \\ . \\ . \\ . \\ V_{kd}^T \end{array} \right\} (X - M_k) \quad (6b)$$

$$y_{kK} = V_{kK}^T(X - M_k) \quad (7)$$

$$Q_k(X) = \sum_{K=1}^{d} y_{kK}^2 \quad (8)$$

Here, V_{kK}^T is the K^{th} row of the lower triangular matrix L_k . As, V_{kK}^T is zero beyond the K^{th} element, the number of multiplications involved in computing y_{kK} is K only. That is, calculation of one group's discriminant function requires only *0.5d(d+1)+d* multiplications. For m classes, the number of multiplications will be equal to *m(0.5d(d+1)+d)*, which is approximately half of the direct calculation through equation (4). So, the theoretical speed-up of this method can be given as:

$$Theoretical\ Speed - up = \frac{2(d+1)}{d+3}$$

3.2 Unitary Canonical Form (UCF) Approach

It is well known that a positive definite matrix such as $\sum_k$ can be decomposed (Singular Value Decomposition) into:(2)

$$\sum_k = U_k^T D_k U_k \quad (9)$$

where,

U_k = a matrix whose i^{th} column is the eigenvector which corresponds to i^{th} largest eigenvalue of $\sum_k$

D_k = a diagonal matrix whose diagonal elements are eigenvalues of $\sum_k$ and $D_{k,j,j} < D_{k,i,i}\ for\ all\ j > i.$

Using the U_k , the quadratic function (equation (4)) can be calculated as follows:[2]

$$Y_k = \left\{ \begin{array}{c} y_{k1} \\ y_{k2} \\ . \\ . \\ . \\ y_{kd} \end{array} \right\} \tag{10a}$$

$$Y_k = U_k^T(X - M_k) = \left\{ \begin{array}{c} V_{k1} \\ V_{k2} \\ . \\ . \\ . \\ V_{kd} \end{array} \right\} (X - M_k) \tag{10b}$$

$$y_{kK} = V_{kK}^T(X - M_k) \tag{11}$$

$$Q_k(X) = \sum_{K=1}^{d} \frac{y_{kK}^2}{D_{k,K,K}} \tag{12}$$

Here, V_{kK}^T is the K^{th} column (i.e K^{th} eigen vector of $\sum_k^{-1}$) of matrix U_k. By use of this method, the calculation of a quadratic form demands the same number of multiplications as that of the ML algorithm. That is, this method demands *d(d+1)* multiplications to calculate the quadratic form. However, this method allows the use of a partial sum (truncation) approach.[2, 1] In the following, eigenvectors are assumed to be normalised with corresponding eigenvalues to eliminate the division operation in equation (12). That is :

$$V_{kK}^T = \frac{V_{kK}^T}{\sqrt{D_{k,K,K}}} \tag{13}$$

3.3 Winograd's Method

Consider the identity:

$$x_1y_1 + x_2y_2 = (x_1 + y_2)(x_2 + y_1) - x_1x_2 - y_1y_2 \tag{14}$$

Winograd's identity is an expansion of equation (14) for the even number $n = 2k$ of pairwise products (say multiplication of two vectors X and Y of size n) and is given as :

$$X^TY = \sum_{i=1}^{2k} x_iy_i = \sum_{u=1}^{k}(x_{2u-1}+y_{2u})(x_{2u}+y_{2u-1}) - \sum_{u=1}^{k} x_{2u}x_{2u-} - \sum_{u=1}^{k} y_{2u}y_{2u-1} \tag{15}$$

Here, x_i, y_i are elements of vectors X and Y. If n is odd, simply apply the method in equation (15) by adding an extra zero element to both vectors. It has been shown that methods based on three or more pairwise products will not give results better than two pairwise products.[34] In the following pages, small letters in the equations where Winograd's method is used signify that they are elements of vectors or matrices of the respective capital letters.

3.4 Quadratic Form Range Theorem

According to Graybill,[26] if A is a symmetric matrix of dimension d x d with characteristic roots $E_1 > E_2 > \ldots > E_d$, then

$$E_d \leq \frac{X^T A X}{X^T X} \leq E_1 \; for \; any \; vector \; X \neq 0 \tag{16}$$

Here, $X^T A X$ is a quadratic form.

From equation(16), $E_d X^T X, E_1 X^T X$ can be inferred as the range of the quadratic term $X^T A X$.

Similarly, the range of the quadratic term in equation(4) can be represented as:

$$E_{k,d}(X - M_k)^T (X - M_k) \leq Q_k(X) \leq E_{k,1}(X - M_k)^T (X - M_k) \tag{17}$$

Here, $E_{k,d}, E_{k,1}$ are the lowest and highest eigenvalues of the k^{th} group's inverse covariance matrix ($\sum_k^{-1}$). So the range of the discriminant function in equation (3) can be represented as:[25]

$$B + E_{k,d}(X - M_k)^T (X - M_k) \leq d_k(X) \leq B + E_{k,1}(X - M_k)^T (X - M_k) \tag{18}$$

This can be also denoted as :

$$\begin{aligned} B_k + q_{k,min}(X) \leq d_k(X) \leq B_k + q_{k,max}(X), \; or \\ d_{k,min}(X) \leq d_k(X) \leq d_{k,max}(X) \end{aligned} \tag{19}$$

By using the lower triangular matrix and range rule of the discriminant function, a fast ML classification rule is developed with three stages[1]); at each stage some groups are eliminated from the search process. It is observed that in this TSML algorithm some groups will be eliminated just by using ranges, and only for groups which are not eliminated by these ranges will discriminant functions be calculated. If all groups are eliminated using only ranges then a good speed-up is observed; otherwise the calculation of ranges becomes a burden and the algorithm performs worse than the original ML algorithm.

For example, consider a 10 group problem in 4 channel MSS data analysis. In order to classify a sample (pixel vector), approximately 200 multiplications (10.4.(4+1)) are needed for the ML algorithm. Suppose now that the TSML method is applied on a pixel vector and 3 groups are eliminated up to the second stage. That is, by using around 64 multiplications (the sum of the total number of multiplications for both range calculations and the actual group discriminant function calculation of the most probable group using the LTCF approach) only three groups are eliminated from the search process. In order to classify the pixel, an additional 84 multiplications are needed assuming LTCF is used in the third stage. So in total about 150 multiplications are used, less than 200 and in this example the TSML classifier is advantageous in classifying this pixel. If a direct matrix multiplications approach is used at the third stage then the total number of multiplications becomes 184, still less than 200. Thus, here the use of ranges does not reduce the classification time appreciably. One can say that the performance of this TSML is a function of the number of groups eliminated up to stage 2 and the remaining groups for which actual discriminant function is required. So TSML algorithm can further be improved by proposing efficient methods to calculate ranges and quadratic terms. In the following some efficient methods are proposed to calculate these ranges and quadratic terms.

3.5 Winograd's Method with Quadratic Form Range Theorem

From equation (19) one can infer :

$$\begin{aligned} q_{k,min} &= E_{k,d}[(X - M_k)^T (X - M_k)] \\ &= E_{k,d}[X^T X - X^T W_k + M_k^T M_K] \end{aligned} \tag{20}$$

where,

$$W_k = 2.M_k$$

By applying Winograd's method, equation (20) can be written as:

$$\begin{aligned} & E_{k,d}\ [\sum_{u=1}^{l} (x_{2u-1} + x_{2u})(x_{2u} + x_{2u-1}) \\ & - \sum_{u=1}^{l} x_{2u} x_{2u-1} - \sum_{u=1}^{l} x_{2u-1} x_{2u} \\ & - \sum_{u=1}^{l} (x_{2u-1} + w_{k,2u})(x_{2u} + w_{k,2u-1}) \\ & + \sum_{u=1}^{l} x_{2u-1} x_{2u} + \sum_{u=1}^{l} w_{k,2u-1} w_{k,2u} + M_k^T M_k\ ] \end{aligned} \tag{21}$$

where d is assumed to be an even number and $l = d/2$.

By eliminating common terms, (21) may be written as :

$$E_{k,d}\ [\ \sum_{u=1}^{l}(x_{2u-1}+x_{2u})(x_{2u}+x_{2u-1}) - \sum_{u=1}^{l}(x_{2u-1}+w_{k,2u})(x_{2u}+w_{k,2u-1}) - \sum_{u=1}^{l}x_{2u-1}x_{2u}+\sum_{u=1}^{l}w_{k,2u-1}w_{k,2u}+M_k^TM_k\ ] \tag{22}$$

The first term in equation (22) can be written as $\sum_k^{-1}(x_{2u}+x_{2u-1})$. As x_{2u-1}, x_{2u} are elements of a pixel vector, X their values usually lie between 0-255 (grey scale). As x_{2u-1}, x_{2u} values lie between 0-255 their sum lie in between 0-510. So, instead of calculating the square of their sum, if a look-up table (say TAB) is prepared such that squares of each of the values in between 0-510 are stored in that table (array) then they can be used while calculating the first term for any random sample with any group.

The second term is independent of group parameters such as the mean vector, so once calculated it can be used with all $q_{k,min},\ k=1,\ldots,m$. Moreover, this term can be used with the actual discriminant function calculation with UCF approach aided by Winograd's approach. The last two terms are independent of the pixel vector X. So, once they are calculated for a group they will not change. Effectively, calculation of $d_{k,min}$ involves calculation of the middle term for which *d/2 (l)* multiplications are needed. The above derivation can also be applied for the effective calculation of $q_{k,max}$ and the final equation for the same can be given as :

$$q_{k,max} = E_{k,1}\ [\ \sum_{u=1}^{l}(x_{2u-1}+x_{2u})(x_{2u}+x_{2u-1}) - \sum_{u=1}^{l}(x_{2u-1}+w_{k,2u})(x_{2u}+w_{k,2u-1}) - \sum_{u=1}^{l}x_{2u-1}x_{2u}+\sum_{u=1}^{l}w_{k,2u-1}w_{k,2u}+M_k^TM_k\ ] \tag{23}$$

or

$$q_{k,max} = q_{k,min}E_{k,maxmin}$$

where,

$$E_{k,maxmin} = \frac{E_{k,1}}{E_{k,d}} \tag{24}$$

(the ratio of maximum and minimum eigenvalues of $\sum_k^{-1}$)

By using the above maximum and minimum ranges of the discriminant functions, groups will be eliminated from the search process at stages I and II in the TSML algorithm.(1) By using the above equations, the number of multiplications required to calculate ranges is reduced to half compared to earlier approach.

3.6 Winograd's Approach with UCF Approach

By expanding terms, the quadratic form in equation (4) can be written as:

$$Q_k(X) = X^T \textstyle\sum_k^{-1} X - X^T W_k + M_k^T \textstyle\sum_k^{-1} M_k \qquad (25)$$

Here,

$$W_k = 2.\textstyle\sum_k^{-1} M_k$$

In this equation, the last term is independent of X, and its calculation is required only once. Here, Winograd's method can be effectively used while calculating the second and first terms. In the following, equation (25) is referred to as the expanded discriminant function.

Using the UCF approach (equations (11) - (13)), this first term can be written as :

$$X^T \textstyle\sum_k^{-1} X = \sum_{i=1}^{d} [V_{ki}^T.X]^2 \qquad (26)$$

Here, V_{ki}^T is the i^{th} eigenvector of the k^{th} group which is normalised with the corresponding eigenvalue. Calculation of the quadratic term in equation (26) involves the scalar product $V_{ki}^T X$. By using Winograd's method, $V_{ki}^T X$ can be written as :

$$V_{ki}^T X = \sum_{u=1}^{l} (x_{2u-1} + v_{k,i,2u})(x_{2u} + v_{k,i,2u-1}) - \sum_{u=1}^{l} x_{2u} x_{2u-1} - \sum_{u=1}^{l} v_{k,i,2u} v_{k,i,2u-1} \qquad (27)$$

As the square of $V_{ki}^T.X$ is needed in equation (26), calculation of all the terms in equation (27) is required. However, the second term in equation (27) is known from calculating the ranges of the discriminant function (see section 3.5). As the last term is independent of X its calculation is required only once for each group. Effectively, calculation of $V_{ki}^T X$ using equation (27) involves the calculation of only the first term, for which only *d/2 (l)* multiplications are needed.

Similarly, the calculation of $X^T W_k$ in equation (25) using Winograd's method can be written as :

$$X^T W_k = \sum_{u=1}^{l}(x_{2u-1}+w_{k,2u})(x_{2u}+w_{k,2u-1}) - \sum_{u=1}^{l} x_{2u}x_{2u-1} - \sum_{u=1}^{l} w_{k,2u}w_{k,2u-1} \tag{28}$$

The second term in this equation can be seen as a common term for all the groups, and is not required to be calculated. Further, the last term is independent of X and need only be calculated once. So, the calculation of $X^T W_k$ using this equation involves the calculation of the first term only, and for this *d/2* multiplications are sufficient.

By combining equations (27) and (28), the effective discriminant function can be written as :

$$\begin{aligned} Q_k(X) \quad = \quad & \sum_{i=1}^{d}[\sum_{u=1}^{l}(x_{2u-1}+v_{k,i,2u})(x_{2u}+v_{k,i,2u-1}) \\ & - \sum_{u=1}^{l} x_{2u}x_{2u-1} - \sum_{u=1}^{l} v_{k,i,2u-1}v_{k,i,2u}\ ]^2 \\ & - \sum_{u=1}^{l}(x_{2u-1}+w_{k,2u})(x_{2u}+w_{k,2u-1}) \\ & + \sum_{u=1}^{l} w_{k,2u-1}w_{k,2u} + M_k^T \sum_{k}^{-1} M_k \end{aligned} \tag{29}$$

In summary, the total number of multiplications used while calculating the discriminant function with equation (29) can be given as *d [d/2 +1] + d/2 = d(d+2)/2* and so, the theoretical speed-up of this algorithm can be given as:

$$Theoretical\ Speed-up \quad = \quad \frac{2(d+1)}{(d+2)}$$

3.7 Winograd's Approach with LTCF Approach

As mentioned earlier, the quadratic form ($Q_k(X)$) can be written in the form of equation (25). The first term in this equation can be calculated with a LTCF approach discussed above. Further, this calculation can be improved by using Winograd's method.

Using the LTCF approach the first term in equation (25) can be written as :

$$X^T \textstyle\sum_k^{-1} X \quad = \quad \displaystyle\sum_{i=1}^{d}[V_{ki}^T X]^2 \tag{30}$$

Here, V_{ki}^T is the i^{th} row of the lower triangular matrix (L_k) calculated from equation (5). As mentioned earlier, $V_{ki}^T X$ requires i multiplications because of the zero valued elements in vector V_{ki}^T after its i^{th} element.

By applying Winograd's method to pairs of elements, the quadratic form can be represented in a LTCF approach as :

$$\begin{aligned} = \; & [\, l_{k,1,1}x_1 \,]^2 + [\, (l_{k,2,1} + x_2)(l_{k,2,2} + x_1) - \underline{x_1x_2} \\ & \overline{l_{k,2,1}l_{k,2,2}} \,]^2 + [\, (l_{k,3,1} + x_2)(l_{k,3,2}x_1) - \underline{x_1x_2} \\ & -\overline{l_{k,3,1}l_{k,3,2}} + l_{k,3,3}x_3 \,]^2 + \\ & [\sum_{u=1}^{2}(x_{2u-1} + l_{k,4,2u})(x_{2u} + l_{k,4,2u-1}) \\ & \underline{-\sum_{u=1}^{2} x_{2u}x_{2u-1}} - \overline{\sum_{u=1}^{2} l_{k,4,2u-1}l_{k,4,2u}} \,]^2 \\ & + \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\ & + \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\ & +[\sum_{u=1}^{h}(x_{2u-1} + l_{k,d,2u})(x_{2u} + l_{k,d,2u-1}) \\ & \underline{-\sum_{u=1}^{h} x_{2u}x_{2u-1}} - \overline{\sum_{u=1}^{h} l_{k,d,2u-1}l_{k,d,2u}} \,]^2 \end{aligned} \quad (31)$$

In the above expansion, d is assumed to be even. If d is odd, then the last term can be written as :

$$\begin{aligned} & [\sum_{u=1}^{h}(x_{2u-1} + l_{k,d,2u})(x_{2u} + l_{k,d,2u-1}) \\ & \underline{-\sum_{u=1}^{h} x_{2u}x_{2u-1}} - \overline{\sum_{u=1}^{h} l_{k,d,2u-1}l_{k,d,2u}} + l_{k,d,d}x_d \,]^2 \end{aligned} \quad (32)$$

Here, h = d div 2 (i.e, d = 2h+1)

In equation (31), the terms underlined can be seen as common for all the groups for that pixel X. In fact, while calculating the term $\sum_{u=1}^{l} x_{2u}x_{2u-1}$ in equation (31), these values which are available from calculation of ranges using Winograd's approach can be used. Thus, no extra computations are needed to calculate these terms. The terms with over lines are independent of X, and are required to be calculated only once.

Effectively, the total number of multiplications used while calculating the expansion in equation (31) can be represented as a series, $1,\ 1,\ 2,\ 2,\ 3,\ 3, \ldots,$. Assuming d extra multiplications for the calculation of squares, the total number of multiplications can be given as $= \frac{d(d+8)}{4}$. So, the theoretical speed-up of this LTCF method aided by Winograd's method can be given as :

$$Theoretical\ Speed-up \;\; = \;\; \frac{4(d+1)}{(d+8)}$$

See Figure 1 for the theoretical speed-up details of LTCF, UCF, UCF with Winograd's Method and LTCF with Winograd's method for varying dimensionality (number of bands). By observing the respective equations for speed-up, one can find that UCF aided by Winograd's approach is almost same as the LTCF approach.

Using the modified approach to calculate ranges and with small modification in the logic at stages I & II a modified TSML classifier is proposed. The main difference between this method and the previous TSML method can be seen as follows. In this method, the most likely group (at Stage I) will be found from minimum ranges of the groups only, whereas in the old method the most likely group is found using maximum ranges. Thus this modified method obviates the calculation of maximum ranges of the discriminant function. Further, it uses Winograd's approach to calculate ranges.

Modified Three Stage ML Classifier

1. Calculate $M_k, \sum_k, \sum_k^{-1}, |\sum_k|, E_{k,d}, E_{k,1},\ k = 1, \ldots, m$ using the training data.
2. Read new random pattern vector X.

   Stage I

3. Calculate $\sum_{u=1}^{l}(x_{2u-1} + x_{2u})$ and $\sum_{u=1}^{l} x_{2u-1}x_{2u}$ (Here, l = d/2).
4. Calculate minimum ranges of the discriminant function (in equation(18)) for each group $(d_{k,min}(X))$.
5. Find minimum of $d_{k,min}(X), k = 1, \ldots, m$. Call the corresponding group number *Iclass*; *Iclass* is the most probable class.
6. Let $A = \{j/j = 1, \ldots, m,\ and\ j = Iclass\}$; that is, A contains the numbers of groups other than *Iclass*.
7. For each $j \epsilon A$, remove j from A if $d_{j,min}(X) > d_{Iclass,max}(X)$. If A is empty, then classify X as group *Iclass* and go to Step 2.
8. Otherwise, calculate actual discriminant function value for group *Iclass*, i.e, $d_{Iclass}(X)$, using the LTCF approach aided by Winograd's method.

   Stage II

9. For each $j \epsilon A$, remove j from A if $d_{j,min}(X) > d_{Iclass}(X)$. If A is empty, then classify X as *Iclass* and go to Step 2.

   Stage III

10. For each $j \epsilon A$, calculate the actual discriminant function $d_j(X)$ using LTCF aided by Winograd's method . If this $d_j(X) > d_{Iclass}(X)$, then remove j from A; otherwise *Iclass* =*j*, then remove j from A. Finally, if A is empty then X is classified as group *Iclass*. Go to step 2.

3.8 Ranges of The Expanded Discriminant Function

From the quadratic forms range theorem, the range of the expanded function in equation (25) can be written as :

$$E_{k,d}(X)^T(X) + C_k \quad \leq \quad X^T \textstyle\sum_K^{-1} X + C_k \leq E_{k,1}(X)^T(X) + C_k \qquad (33)$$

where,

$$C_k \quad = \quad -X^T W_k + M_k^T \textstyle\sum_k^{-1} M_k$$

By normalising the above ranges with X X, the ranges can be re-written as :

$$E_{k,d} + \overline{C}_k \quad \leq \quad \frac{X^T \sum_k^{-1} X}{X^T X} + \overline{C}_k \leq E_{k,1} + \overline{C}_k \qquad (34)$$

where,

$$\overline{C}_k \quad = \quad \frac{C_k}{X^T X}$$

In equation (33), the ranges of the quadratic form are similar to those defined in the TSML Classifier.(1) Calculation of ranges with equation (34) obviates one multiplication compared to equation (33) . The ranges in equation(18) necessitates the calculation of the Euclidean distance $(X - M_k)^T(X - M_k)$, and is not used at all in the algorithm. That is, some groups will be eliminated based upon this range at Stages I and II. If the groups are not eliminated even after the second stage, the actual discriminant function is required to be calculated,(1) but by observing the ranges in equation (34), it can be seen that the ranges of any group can be found by calculating $\overline{C}_k$ and $X^T X$. Moreover, $X^T X$ is common for all the groups, and can be calculated simply from the look-up table (TAB) which is discussed in the following. Further, $\overline{C}_k$ can be used at Stage III, in calculating actual discriminant function values. As, the $\overline{C}_k$ values are used both at stages I & III, the computational advantage is expected to be more if these ranges are used with TSML. Further, while calculating the term $X^T W_k \; in \; \overline{C}_k$ Winograd's approach can be used.

4 MODIFIED MAXIMUM LIKELIHOOD CLASSIFIER

Chang and Dwyer,(35) Venkateswarlu,(13) and Oza and Sharma(36) observed that elimination of the term $\ln |\sum_k|$ from the conventional ML discriminant function gives better results than the original ML algorithm. This classification rule is:

$X \epsilon\, \omega_k$ if

$$d_k(X) = \min\{d_i(X)\}, i = 1, \ldots, m \qquad (35)$$

Here,

$$d_k(X) \quad = \quad Q_k(X)$$

In this classification rule also, calculation of the term Q (X) is very demanding. Methods discussed above can be used with this classification rule to reduce its CPU time requirements.

Similarly, we identified that the proposed logics can also be used with the Mahalanobis distance classification rule(37) to reduce classification time.

5 EXPERIMENTS & DISCUSSION

Experimental work has been carried out on two PC systems, namely PC/XT and PC/AT working under the DOS operating system. Both systems have a Math Co-processor. Separate modules are written for each of the methods to observe the CPU time for classification of any data set, I/O operations are eliminated and the time observed is strictly for the classification of the data. In the tables, CPU time is given in seconds. Data used in this study can be summarised as :

a. Six channel TM data of sample set size 700 with 7 ground cover classes. Group parameters such as mean vector and covariance matrices are given in Table. 1.

b. A Randomly generated 12 dimensional data set of size 5000.

By combining two or more of the above proposals a number of fast ML algorithms are proposed which we expect to be better than the original ML algorithm. Methods studied in this work are:

Method 1 Conventional ML algorithm in which a direct matrix multiplication approach is used for the calculation of the quadratic term in the discriminant function. This is the most commonly implemented form of the ML decision rule for which approximately *d(d+1)* multiplications and additions are needed to classify a random pixel vector.(31)

Method 2 Old TSML classifier.(1)

Method 3 Expanded discriminant function approach aided by the UCF approach to calculate the $X^T \sum_k^{-1} X$ term.

Method 4 Expanded discriminant function approach aided by UCF and Winograd's approach in calculating the $X^T \sum_k^{-1} X$ term.

Method 5 Expanded discriminant function approach aided by LTCF approach to calculate $X^T \sum_k^{-1} X$ term.

Method 6 Expanded discriminant function approach aided by LTCF and Winograd's approaches in calculating the $X^T \sum_k^{-1} X$ term.

Method 7 Modified TSML classifier (see the algorithm) in which Winograd's method is used in calculating ranges of the discriminant function and Method 4 at stage III in calculating the actual discriminant function.

Method 8 Modified TSML classifier (similar to Method 7) with Method 6 to calculate the actual discriminant function at stage III.

Method 9 Method 7 with partial sum approach at stage III.

Method 10 Method 8 with partial sum approach at stage III.

Method 11 Modified TSML classifier with the third stage as same as the old TSML classifier. That is, at stage III the $(X - M_i) \sum_i^{-1}(X - M_i)$ term will be used instead of the expanded discriminant function. Here, the LTCF approach aided by the partial sum approach is used.

Method 12 Method 11 with UCF approach and partial sum logic in calculating the $(X - M_i) \sum_i^{-1}(X - M_i)$ term at stage III.

Method 13 Method 11 aided by threshold logic of Feiveson.[3] Here, an unclassified pixel vector will first be assumed most probably to the previous classified pixel's group and the minimum range of the discriminant function will be calculated for that group. Groups will be eliminated using this minimum range value and thresholds. Later, a modified TSML classifier (method 11) will be applied to the rest of the groups (See Figure 2 for STAGE I of this algorithm and STAGE II & III are same as method 11).

Method 14 The old TSML classifier with thresholds (similar to Method 13).

Method 15 The modified TSML classifier which uses the ranges of the expanded discriminant function (see section 3.8 for description).

Method 16 In the case of image classification, a small modification in the old TSML classifier will further reduce the time requirements. This modification is at stage I. Instead of calculating minimum ranges of all the groups, we can first calculate the minimum range of the most likely group (the group of the previous classified pixel) and while calculating the ranges of the other groups a partial sum approach[29, 32, 33] can be used. This reduces the computations spent in the ranges calculation and minimum group calculation in the TSML classifier. See Figure 3 for modified algorithm with this modification.

Method 17 While calculating ranges of a discriminant function the Euclidean distance $(X - M_k)^T(X - M_k)$ is used. In reality M_k will consist of real valued elements. If M_k is assumed to have integer valued elements (by taking nearest integer value) we find that the elements of $(X - M_k)$

will have values between -255 and 255, because elements of X and rounded M_k will have elements between 0 and 255 for 8 bit images. So, once squares of all values between -255 and 255 are calculated and stored in a look-up table, the distance $(X - M_k)^T(X - M_k)$ can be approximated with the help of pre-calculated squares from the array. This approach is denoted as pre-calculated squares approach in the following pages.

This approach is used with the Old Three Stage ML classifier (see APPENDIX). Further, we feel that Euclidean distances can also be calculated from the method suggested by Cannon[(38)] which also uses a type of look-up table.

Method 18 Method 11 with pre-calculated squares in the calculation of ranges instead of Winograd's approach.

Method 19 Pre-calculated squares approach is used with Method 16. Here the partial sum approach is also used.

Method 20 At Stage I of the modified three stage classifier (Method 11) a pixel is assumed to belong to the previous classified pixel's group. Here it is not advantageous to use the partial sum approach while calculating the Euclidean distance with Winograd's method.

Method 21 MML classifier

Method 22 Method 11 with MML classifier

Method 23 Method 2 with MML classifier

Method 24 Method 17 with MML classifier

Method 25 Method 16 with MML classifier

Method 26 Method 18 with MML classifier

Method 27 Mahalanobis distance classifier

Method 28 Method 11 with Mahalanobis distance classifier

The first data set is classified with all methods and the CPU time is observed (Table. 2). Further, each method's (up to Method 20) classification accuracy is the same, and is shown in Table (3). It is observed that Winograd's method does not give a considerable reduction in CPU time with methods 4 and 6. Moreover, it is identified that only careful implementation of equations based on Winograd's method give good results. However, the reduction in CPU time is not significant. From the derivations we find that Winograd's method only reduces the number of multiplications while calculating the quadratic function in equation (26), and the number of additions is not reduced. If care is not taken in converting the equations into code, the number of additions will increase and the algorithm will show undesirable results. We believe that the performance of methods 4 and 6 which uses Winograd's method will also depend on the computer hardware. If a system's ALU has larger execution times for multiplication than addition, these algorithms are expected to give a perceptible speed-up. However, as these algorithms use fewer computations (assuming additions cost the same as respective old algorithms) they will definitely show a reduction in CPU time on any computer system. Even in those processors that can execute multiplications and additions in comparable time, these algorithms will work equally well.

It is observed that Method 11 performs better than Method 2 and also than any other Methods 6-12. It is observed that the expanded discriminant function approach with partial sum logic does not give better results. The expanded discriminant function is does not show any improvement with partial sum logic. That is, the full discriminant function is needed with this approach as there is no early termination of the quadratic function observed with the partial sum approach.

Further, it is observed that the threshold logic with Method 2 and Method 11 gives good results. See Table (4) for details of thresholds of the first data set. These thresholds are calculated from the algorithm given in Feiveson.[(3)] With the both the methods threshold logic gave better results.

The New TSML classifier (Method 15) which uses ranges of the expanded discriminant function does not give good results. It is identified that no group is eliminated at both the first and second stages by the use of these ranges. For a random pixel, ranges used in this old TSML classifier and ranges of the expanded discriminant function (Section 3.8) are shown in Table (5). We find that the ranges based on the expanded discriminant function are too overlapped; no group can be eliminated by the use of the corresponding minimum value shown in the same table, whereas ranges in the old three stage classifier are found to be useful in eliminating some groups (for the example shown in the table all the groups will be eliminated by the ranges only) with the help of the minimum value shown. Thus. Method 15 does not show encouraging results.

Method 16, which assumes an unclassified pixel belongs to the previous pixels group in stage I is found to give an appreciable reduction in computer time. In reality autocorrelation between pixels in most of the remotely sensed images will be high, so we believe that for images this method will give still better results. Further, it is possible to use the partial sum approach while calculating the Euclidean distances (see Figure 3).

Method 17 uses pre-calculated values to calculate Euclidean distances in the old TSML algorithm. This is found to be give encouraging results, and has shown encouraging results with the modified TSML algorithm too, but the reduction in time is not appreciable. There is no change in the classification accuracy by rounding the mean values and using this pre-calculated squares in this method.

Method 19, which uses the autocorrelation assumption is also found to give good results with this data. Off all the methods, Method 11 gives the best results with the selected data set. However, the methods 16, 17, 18, and 19 also gave similar results.

Methods 1, 2 , 11, 16, 17 and 18 are taken to study their performance under varying dimensionality and sample set size. For this purpose a randomly generated 12 dimensional data set mentioned above is used. The number of dimensions is varied from 4 to 12 with steps of 2 and CPU time requirements are observed on both XT and AT systems for a fixed sample set size of 2600. In the case of experiments on PC/XT, we did not observe the CPU time for method 1 for all the dimensions as it is so time consuming. It is observed that the modified TSML classifier give better results than the old three stage classifier (see Figure 4). However, improvement is not considerable and we believe this may be due to the nature of the groups which we have selected in generating the data. The pre-calculated squares with the old TSML algorithm show better results for any dimension (Figure 5), but the same pre-calculated squares approach with Method 11 shows similar results to that of Method 11 (see Figure 6). Further, the number of samples are varied from 1000 to 5000 for all 4, 6, and 12 dimensions and the CPU time is observed for all the algorithms on the PC/AT (see Figures 7-10). It is observed that the proposed modification for the TSML algorithm shows good results. In Figures 7-10 we have shown the performance of some methods only. In order to observe clearly the relative advantage of each of the methods, the same analysis is carried out on PC/XT. See Tables. 6-8 for CPU time observations for both varying dimensionality and sample set size on PC/XT. From these tables one can find the relative advantage of each method very clearly. Figure 7 shows how the pre-calculated squares approach is useful with old TSML algorithm under varying sample set size. In order to identify the effectiveness of the proposed methods compared to the old TSML algorithm, we have prepared Table (9). Here, the derivation of a table element follows: Consider 1'st row 3'rd column element, i.e 1.62. This is calculated either from the Table (6) or (7). From Table (6), one can observe that for a sample set of size 2600, the old TSML classifier takes 520 seconds of CPU time. For the same operation, Method 16 (best of all the proposed methods) takes 318 seconds of CPU time. The ratio of 520 and 318 is 1.62, given in Table (9). This value explains the effectiveness of this proposed logic with the old TSML algorithm. From the same table one can observe the actual speed-up of the TSML algorithm with the proposed modification for different dimensions and sample set sizes. In this table, speed-up is calculated based on the best algorithm for a given dimension and sample set size. We find that in most cases the autocorrelation assumption shows better results and the pre-calculated squares approach with the modified three stage ML algorithm (Method 18) shows some improvement

Methods 21-26 are also studied under varying dimensionality and sample set size. Figure 11 shows performance of these algorithms under varying dimensionality. Similarly, Figures 12-14 show how these algorithms behave under varying sample set size. One can also observe here the modified approach showing better results, and identify the autocorrelation assumption as useful with this MML classifier also. It is further observed that the use of thresholds shows a Speed-up of 4 to 8 for different dimensions with both the ML and MML algorithms. Method 11 with the Mahalanobis distance classifier show good result (See Table. 2)

We believe that the look-up table approaches[20, 21] can be directly used with this modified TSML classifier. For example, if one finds that the modified TSML classifier and look-up table approaches give CPU time reductions of 3 and 4 respectively compared to the ML algorithm, then a combination of these two methods will be expected to give a reduction of 12 compared to the literal ML algorithm. As future research, we feel one can work in this direction. In this study we have considered the performance of the modified TSML classifier up to 12 dimensions only. One could study the performance of this algorithm with 192 channel HIRIS data and 224 channel AVIRIS data with the number of groups in the order of hundreds. Further, one can also study the order in which terms in the LTCF approach are calculated to use the partial sum approach effectively.[2] This may be more helpful in the case of future satellite data such as HIRIS and AVIRIS.

Further, we believe that with high dimensional images such as AVIRIS, HIRIS, this modified TSML algorithms may give some additional implementation problems such as:

a. Difficulties in calculating eigenvalues and

b. Difficulties while calculating thresholds between pairs of groups using the method of Feiveson.[3]

At present, we are trying to see the effectiveness and difficulties of all these methods with 224 channel AVIRIS and other multi channel satellite images.

6 CONCLUSIONS

Some modifications are proposed to improve further the TSML algorithm. Winograd's method is proposed for use with range calculations, and the same method is used with LTCF and UCF approaches[2] in calculating the quadratic form. New types of ranges are derived from the expanded discriminant function and used with TSML algorithm to identify their usefulness in reducing CPU time. Pre-calculated values are proposed for use while calculating the Euclidean distance in ranges calculation and the effectiveness of this logic is studied. It is observed that by assuming the rounded values of group means in the pre-calculated squares approach, the classification efficiency of the algorithms does not change. Further, threshold logic[3] is used with old and modified TSML classifiers and its effectiveness is observed. Performance of old and modified TSML algorithms is studied in detail by varying the dimensionality and number of samples. For the purpose of experimentation, a 6 channel thematic mapper (TM) and a randomly generated 12 dimensional data sets are used. Proposed fast algorithms are also used with MML and Mahalanobis distance classifiers.

References

(1) N.B. Venkateswarlu and P.S.V.S.K. Raju, Three Stage ML Classifier, Pattern Recognition, 24, 1113-1116 (1991).

(2) W.Eppler, Canonical analysis for increased classification speed and channel selection, IEEE Transactions on Geoscience and Electronics, 14:1, 26-33 (1976).

(3) A.H. Feiveson, Classification by Thresholding, IEEE Transactions on Pattern Analysis and Machine Intelligence, 5:1, 48-54, (1983).

(4) J.J. Settle, and S.A. Briggs, Fast Maximum Likelihood classification of remotely sensed imagery, International Journal of Remote Sensing, 8:5, 723-734 (1987).

(5) K.S. Fu, Special computer Architectures for Pattern Recognition, CRC Press, Florida, (1982).

(6) K.S. Fu. VLSI for Pattern Recognition and Image Processing, Springer-Verlag, New York (1984).

(7) P.H. Swain , H.Y. Siegel and B.W. Smith, Contextual Classification of Multispectral Remote Sensing data Using a Multi Processor System, IEEE Transactions on Geoscience and Remote Sensing, 18, 197-203 (1980).

(8) H.K. Garg , G. Pyne, I.C. Matieda, and C.V.S. Prakash, Implementation of ML classification Algorithm on 4 Transputer System using Parallel "C", Technical Report, Space Applications Center, Ahmedabad, India, July (1990).

(9) F. Kriegler et al., Multivariate Interactive Digital Analysis System (MIDAS) : A New Fast multispectral recognition System, Proc. Conference on Machine Processing of Remote Sensed data, Purdue University IEEE catalogue No. 73, CHO. 834-2GE, pp 2B-O-2B-13, Oct 16-18 (1973).

(10) H.M. Kalayeh, M.J. Muasher and D.A. Landgrebe, Feature Selection With Limited Training Samples, IEEE Transactions on Geoscience and Remote Sensing, 21:4, 434-437 (1983).

(11) M.J. Muasher and D.A. Landgrebe, The K-L Expansion as an Effective Feature Ordering Technique for Limited Training Sample Size, IEEE Transactions on Geoscience and Remote Sensing, 21:4, 437-441 (1983).

(12) D.G. Goodenough, P.M. Narendra and K.O'Neill, Feature Subset Selection In Remote Sensing, Canadian Journal of Remote Sensing, 4:2, 143-145 (1978).

(13) N.B. Venkateswarlu, Evaluation of Pattern Classifiers for Pattern Recognition, Masters Thesis Submitted to Indian Institute of Technology, Kanpur, India, (1988).

(14) B.F. Merembeck and B.J. Turner, Directed canonical Analysis and The performance of Classifiers under its Associated Linear Transformation, IEEE Transactions on Geoscience and Remote Sensing, 18, 190-196 (1980).

(15) P.H. Swain and H. Hauska, The Decision Tree Classifier : Design and Potential, IEEE Transactions on Geoscience Electronics, 15, 142-152 (1977).

(16) B. Kim and D.A . Landgrebe, Hierarchical Classifiers Design in High-Dimensional Numerous Class Classes, IEEE Transactions on Geoscience and Remote Sensing, 29, 518-528 (1991).

(17) Y. IIkura and Y. Yasuoka, Utilisation of a best linear discriminant function for designing the binary tree, International Journal of Remote Sensing, 12:1, 55-67 (1991).

(18) S. Fujimura and H. Hanaizumi, Hierarchical Algorithms for the Classification of Remotely Sensed Multispectral Images, IEICE Transactions on Part. E, 74, 295-301 (1991).

(19) P.L. Odell and B.S. Duran, Comparison of Some Classification Techniques, IEEE Transactions on Computers, 23, 891-896 (1974).

(20) P.M. Mather, A computationally efficient maximum likelihood classifier employing priori probabilities for remotely sensed data, International Journal of Remote Sensing, 6:2, 369 - 379, (1985).

(21) P.V. Bolstad and T.M. Lillesand, Rapid Maximum Likelihood Classification, Photogrammetric Engineering and Remote Sensing, 57, 67-74 (1991).

(22) S.C. Ahearn and C. Wee, Data Space Volumes and Classification Optimisation of SPOT and Landsat TM Data, Photogrammetric Engineering and Remote Sensing, 57, 61-65 (1991).

(23) R. Shlien, Practical Aspects related to Automatic Classification of Landsat Imagery using Look-Up tables, Canada Center for Remote Sensing, Research report 75-2, Ottawa, Canada (1975).

(24) D. Lee and D.A. Landgrebe, Fast Likelihood Classification , IEEE Transactions on Geoscience and Remote Sensing, 29, 509-517 (1991).

(25) D.E. Minskii and A.M. Chizhevskii, Fast Bayes classification of Multispectral Images, Soviet Journal of Remote Sensing, 1, 69-79 (1982).

(26) F.A. Graybill, Introduction to matrices with applications in Statistics, Wadsworth , California (1969).

(27) N.B. Venkateswarlu and P.S.V.S.K. Raju, Winograd's method: a perspective for some PR Problems, Pattern Recognition Letters, 15, 105-109 (1994).

(28) N.B. Venkateswarlu and P.S.V.S.K. Raju, Fast ML classifier for Remotely Sensed Images, Communicated to International Journal of Remote Sensing, May (1991).

(29) N.B. Venkateswarlu and P.S.V.S.K. Raju, A new fast classifier for remotely sensed imagery, International Journal of Remote Sensing, 14, 383-393 (1993).

(30) P.H. Swain and S.M.Davis, Remote Sensing The quantitative approach, Mc-Graw Hill, New York (1978).

(31) J.A. Richards, Remote Sensing Digital Image Analysis, Springer Verlag, Berlin, (1986).

(32) J.A. Bryant, Fast Classifier for Image Data, Pattern Recognition, 22:1, 45-48 (1989).

(33) M.E. Hodgson, Reducing Computational requirements of the minimum distance classifier, Remote Sensing of Environment, 25, 117-128 (1988).

(34) R. Harter, The Optimality of Winograd's Formula, Communications of ACM, 15:5, 352-353 (1972).

(35) J.K. Chang and S.J. Dwyer, New multiclass classification method : Modified maximum likelihood decision rule, Proceedings of the first International Joint Conference on Pattern Recognition, Washington D.C., October, 334-339 (1973).

(36) M.P. Oza and S.A. Sharma, A Comparative Study of Supervised Classifiers on a subscene in Junagadh district, Gujarat, Journal of the Indian Society of Remote Sensing, 18:3, 18-24 (1990).

(37) R.O. Duda, and P.E. Hart, Pattern Classification and Scene analysis, Wiley-InterScience, New York (1973).

(38) R.L. Cannon, J.V. Dave and J.C. Bezdek, Efficient Implementation of Fuzzy C-Means Clustering Algorithms, IEEE Transactions on Pattern Analysis and Machine Intelligence, 8, 248-255 (1986).

Table 1: Mean vector and Covariance Matrices of TM data

Group	Mean Vector	Covariance	Matrix				
1	65.13	7.9703	3.4829	6.5158	15.4918	5.4058	2.6761
	28.11		3.2083	4.7205	11.2476	4.2226	1.5072
	24.53			10.4674	17.3653	10.1343	3.8382
	82.11				100.6353	14.3713	1.5719
	58.05					32.8482	11.2350
	18.17						7.4796
2	73.51	7.5305	4.1215	5.2481	3.6783	10.8313	5.8669
	35.73		4.2828	4.6073	4.0132	11.4198	5.7362
	34.39			7.0796	1.7717	12.8133	7.6058
	82.93				39.3770	20.7186	9.2679
	94.34					91.8638	37.4228
	35.38						21.8762
3	66.30	4.3378	-0.2788	0.1368	-0.5809	1.9365	1.8311
	31.54		3.0153	0.7368	4.1987	1.2867	-0.4440
	22.48			4.2405	-8.1517	1.4787	0.9722
	131.03				104.6501	7.4427	-4.7778
	73.55					9.1121	2.4913
	19.22						8.5003
4	66.36	3.1820	-0.2275	0.3140	0.0950	0.0502	0.3846
	35.13		1.1730	0.8996	-1.8379	0.0334	0.1971
	25.05			2.6536	-6.2290	-0.0692	-0.0318
	116.44				77.1313	6.4668	-3.7385
	71.50					6.8893	1.6561
	19.00						2.9950
5	70.65	5.7881	1.7422	2.1404	-4.6257	0.2185	2.7876
	32.60		2.4045	1.0251	-1.0262	1.4398	0.9375
	27.77			3.5357	-2.7430	2.1431	2.6628
	84.77				47.3721	10.3990	-7.9306
	74.37					14.7411	4.0040
	25.77						9.1151
6	71.25	5.5429	0.4363	0.0104	2.2993	-0.6800	-1.1679
	30.48		2.6170	1.3100	-6.5523	4.4931	3.1821
	25.40			3.6932	-12.5703	7.9940	5.9032
	107.40				65.7621	-34.9278	-28.7912
	64.21					37.9286	24.9381
	17.68						20.3340
7	63.44	12.8483	8.1494	9.7692	14.5699	7.7148	4.8247
	27.56		8.0880	9.7044	8.9883	10.4924	5.8384
	23.45			15.4034	5.1695	17.0818	10.1263
	78.63				107.1971	-7.3615	-19.3642
	71.84					37.8487	19.0122
	23.50						17.2225

Table 2: Observations with TM data

Method	CPU PC/AT (Seconds)	time PC/XT (Minutes)
Method 1	380	7.95
Method 2	200	5.65
Method 3	375	8.15
Method 4	365	8.05
Method 5	220	6.45
Method 6	204	6.25
Method 7	194	5.95
Method 8	145	4.85
Method 9	193	5.80
Method 10	142	4.65
Method 11	135	3.75
Method 12	190	5.75
Method 13	51	1.20
Method 14	60	2.65
Method 15	400	8.55
Method 16	132	2.95
Method 17	133	3.75
Method 18	132	3.70
Method 19	130	3.70
Method 20	133	3.70
Method 21	378	7.70
Method 22	190	5.50
Method 23	134	3.60
Method 24	133	3.50
Method 25	132	3.50
Method 26	134	3.55
Method 27	370	7.73
Method 28	125	3.55

Table 3: Confusion Matrix for Literal Maximum Likelihood classifier for TM data

Group	Percent correct classification	Number of samples used	Classified as group 1	2	3	4	5	6	7
1	96.5	226	218	0	0	0	1	1	6
2	100.0	81	0	81	0	0	0	0	0
3	92.2	128	0	1	118	7	1	1	0
4	97.2	36	0	0	0	35	1	0	0
5	88.5	35	0	0	0	2	31	2	0
6	93.6	47	0	0	0	0	3	44	0
7	93.9	147	1	0	0	0	2	6	138
Percent	Mis-classification	=	5.0						

Table 4: Thresholds for TM data

Group	1	2	3	4	5	6	7
1		-14.20	-14.82	-15.71	-12.38	-11.03	-12.45
2	-14.20		-15.79	-16.47	-11.37	-11.22	-12.82
3	-14.82	-15.79		-8.21	-11.98	-11.77	-13.42
4	-15.71	-16.47	-8.21		-12.91	-13.93	-17.30
5	-12.38	-11.37	-11.98	-12.91		-9.91	-12.03
6	-11.03	-11.22	-11.77	-13.93	-9.91		-12.95
7	-12.45	-12.82	-13.42	-17.30	-12.03	-12.95	

Table 5: Details of Ranges for a sample vector

Group	Ranges of equation (18) Min	discriminant and Max	function equation (34) Min	in Max
1	5.79	85.6	-.054	1.226
2	24.58	3382.19	-0.039	1.419
3	26.99	1659.6	-0.076	0.42
4	35.91	3577.42	-0.194	1.52
5	10.37	383.02	-0.071	0.50
6	9.39	1090.91	-0.13	0.89
7	7.9	319.91	-0.029	0.97
Actual discriminant function value of minimum group (Most Probable Group)		7.46		0.00434

Table 6: Observations with varying dimensions for a constant sample set size of 2600 on PC/XT

Dimension	Method					
	2	11	17	18	16	1
4	520	370	363	348	318	730
6	760	580	534	550	478	1235
8	1020	835	740	815	702	–
10	1325	112	975	1095	940	–
12	1655	1485	1260	1445	1240	–

Table 7: Observations with varying sample set size for 4 channel MSS data on PC/XT

Number of	Method				
Samples	2	11	17	18	16
1000	200	142	140	135	123
1500	301	214	214	207	183
2600	520	370	363	348	318
3500	700	478	487	473	428
4500	901	635	628	606	559
5000	1001	700	700	670	687

Table 8: Observations with varying sample set size for 6 channel TM data on PC/XT

Number of	Method				
Samples	2	11	17	18	16
1000	293	223	205	214	185
1500	440	335	308	321	280
2600	1000	580	536	550	478
3500	1022	782	718	750	700
4500	1360	1055	—	—	—
5000	1455	1190	—	—	—

Table 9: Speed-Up vs Dimension

Dimension	PC/AT with Sample set Size 2600	PC/AT with Sample Set Size 5000	PC/XT Sample Set Size 2600
4	1.56	1.60	1.62
6	1.54	1.59	1.62
12	1.50	1.40	—

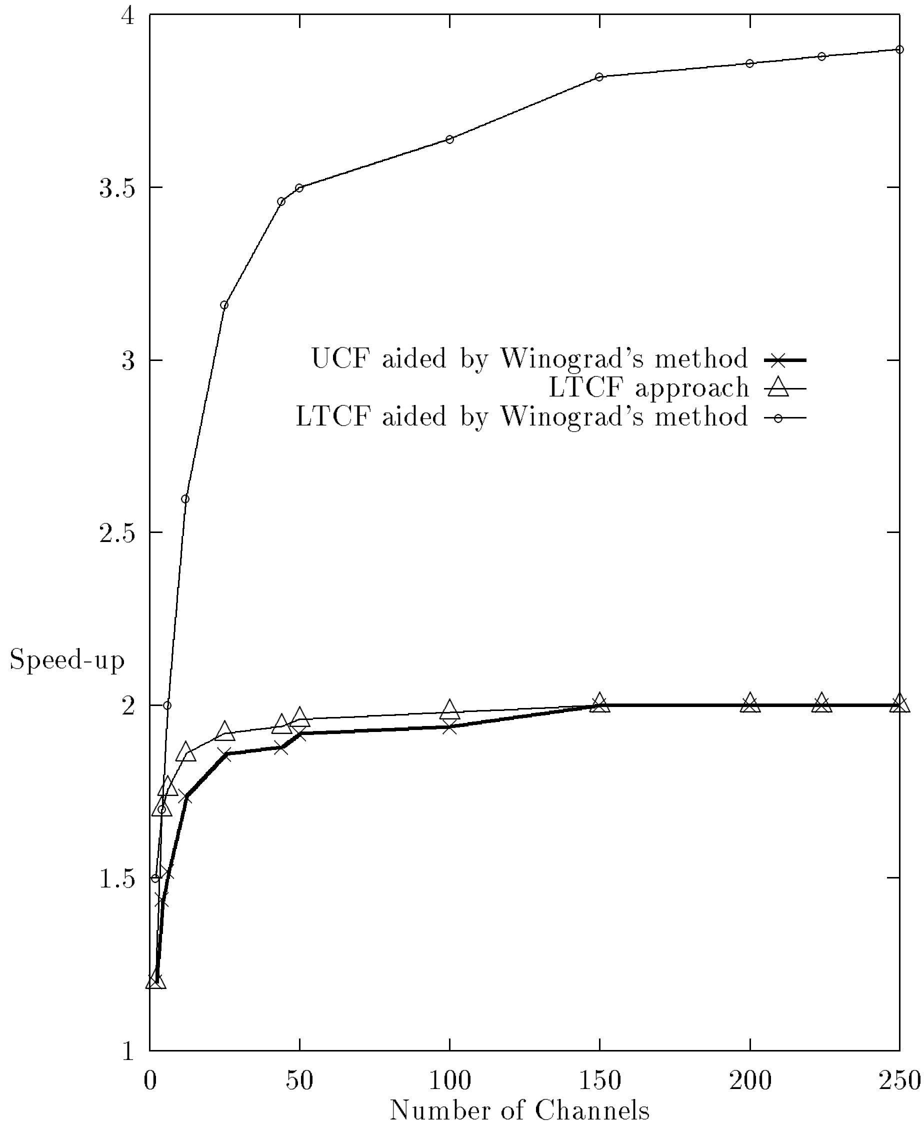

Figure.1 Theoretical Speed-up of different methoods in calculating quadratic form.

STAGE I

Assume the pixel to be classified most probably belongs to the previous classified pixel's group (say group is *Iclass* and remove *Iclass* from A). Calculate the Euclidean distance between mean vector of *Iclass* and X which is

$$dis_{Iclass}(X) = (X - M_{Iclass})^T (X - M_{Iclass})$$

a. Remove each group j from A if

$$E_{Iclass,d} * dis_{Iclass}(X) > \tau_{Iclass,j}$$

b. Remove each group j from A if

$$dis_j(X) > g_{Iclass,j} * dis_{Iclass}(X)$$

c. Remove each group j from A if

$$dis_j(X)\gamma_{j,Iclass} < dis_{Iclass}(X)$$

and if group j is removed then modify

$$dis_{Iclass}(X) = d_j(X) and Iclass = j$$

Here

$$\begin{aligned} dis_j(X) &= (X - M_j)^T (X - M_j) \\ \gamma_{i,k} &= \frac{E_{i,1}}{E_{k,d}} \\ \tau_{Iclass,j} &= \text{threshold between groups } Iclass \text{ and } j.^{(3)} \end{aligned}$$

These γ, τ matrices will be calculated before the classification. Moreover, one can use the partial sum approach of Venkateswarlu and Raju,[29] Bryant[32] and Hodgson[33] with groups elimination with the above mentioned Euclidean distances.

Figure. 2. Old TSML classifier with threshold logic aided by Autocorrelation Assumption

Stage I

Assume the pixel to be classified most probably belongs to the previous classified pixels group (say group is *Iclass* and remove *Iclass* from A). Calculate the Euclidean distance between mean vector of *Iclass* and X which is

$$dis_{Iclass}(X) = (X - M_{Iclass})^T (X - M_{Iclass})$$

a. Remove each group j from A if

$$dis_j(X) > \gamma_{Iclass,j} * dis_{Iclass}(X)$$

b. Remove each group j from A if

$$dis_j(X) * \gamma_{Iclass,j} < dis_{Iclass}(X)$$

and modify

$$dis_{Iclass}(X) = dis_j(X) \; and \; Iclass = j$$

Here

$$\begin{aligned} dis_j(X) &= (X - M_j)^T (X - M_j) \\ \gamma_{i,k} &= \frac{E_{i,1}}{E_{k,d}} \end{aligned}$$

This γ matrix can be calculated before the classification. Moreover, one can use the partial sum approach of Venkateswarlu and Raju,[29] Bryant[32] and Hodgson[33] with groups elimination with the above mentioned Euclidean distances.

Figure. 3. Old TSML classifier with Autocorrelation Assumption

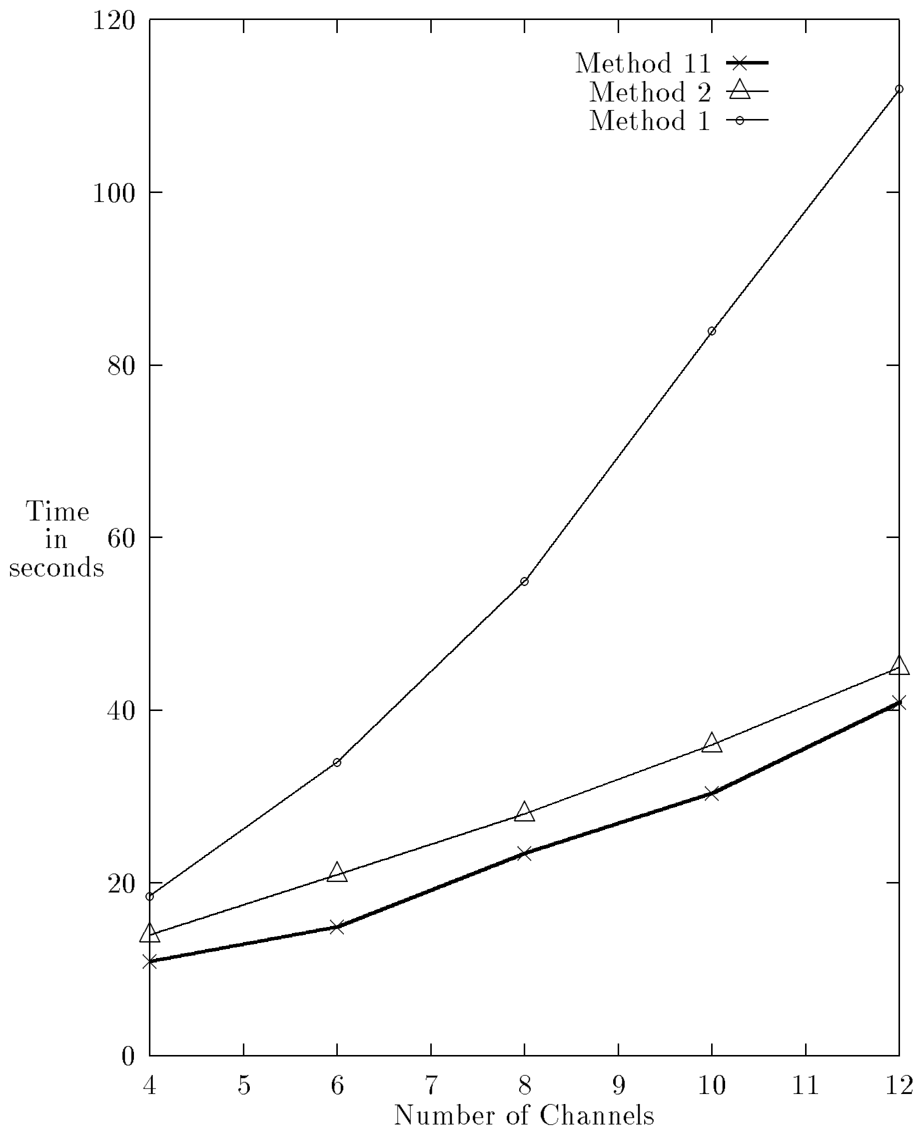

Figure.4 Time vs Dimensionality observations on PC/AT for a sample set size of 2600.

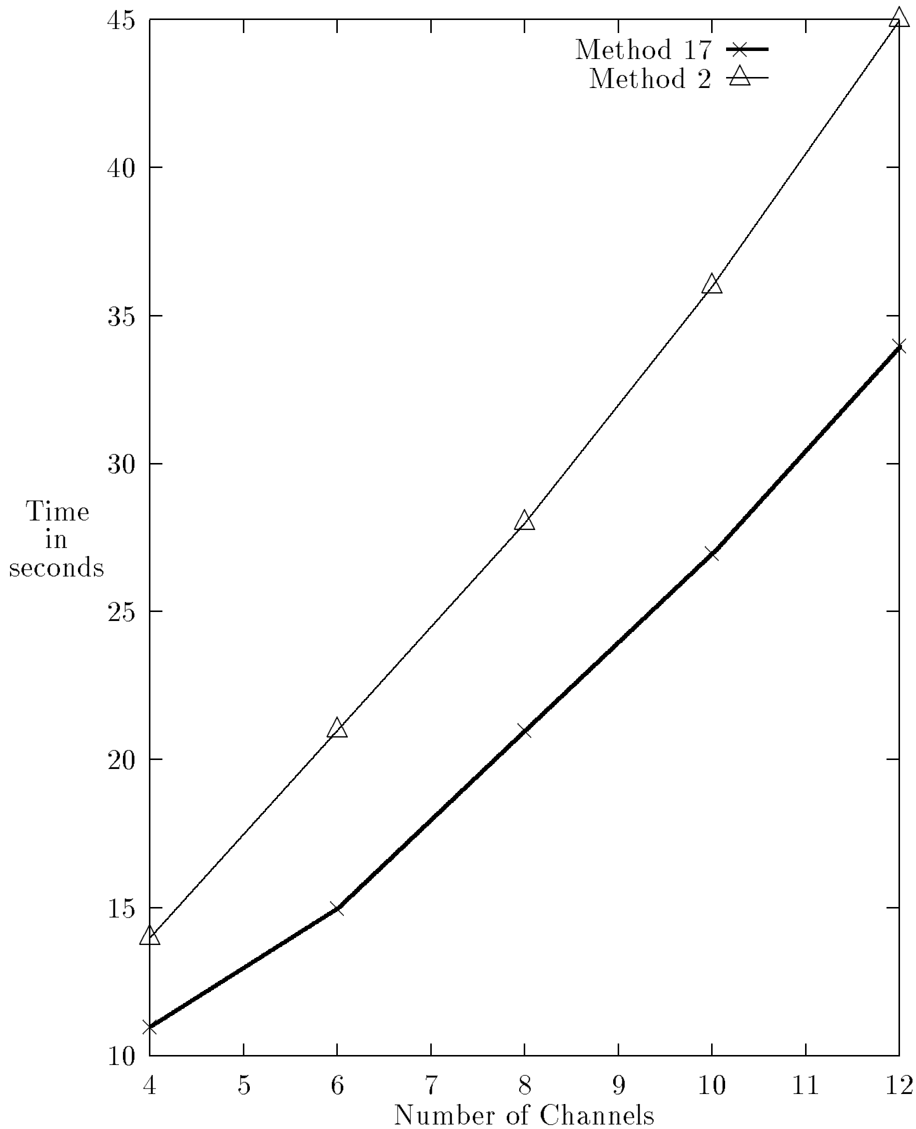

Figure.5 Time vs Dimensionality observations on PC/AT for a sample set size of 2600.

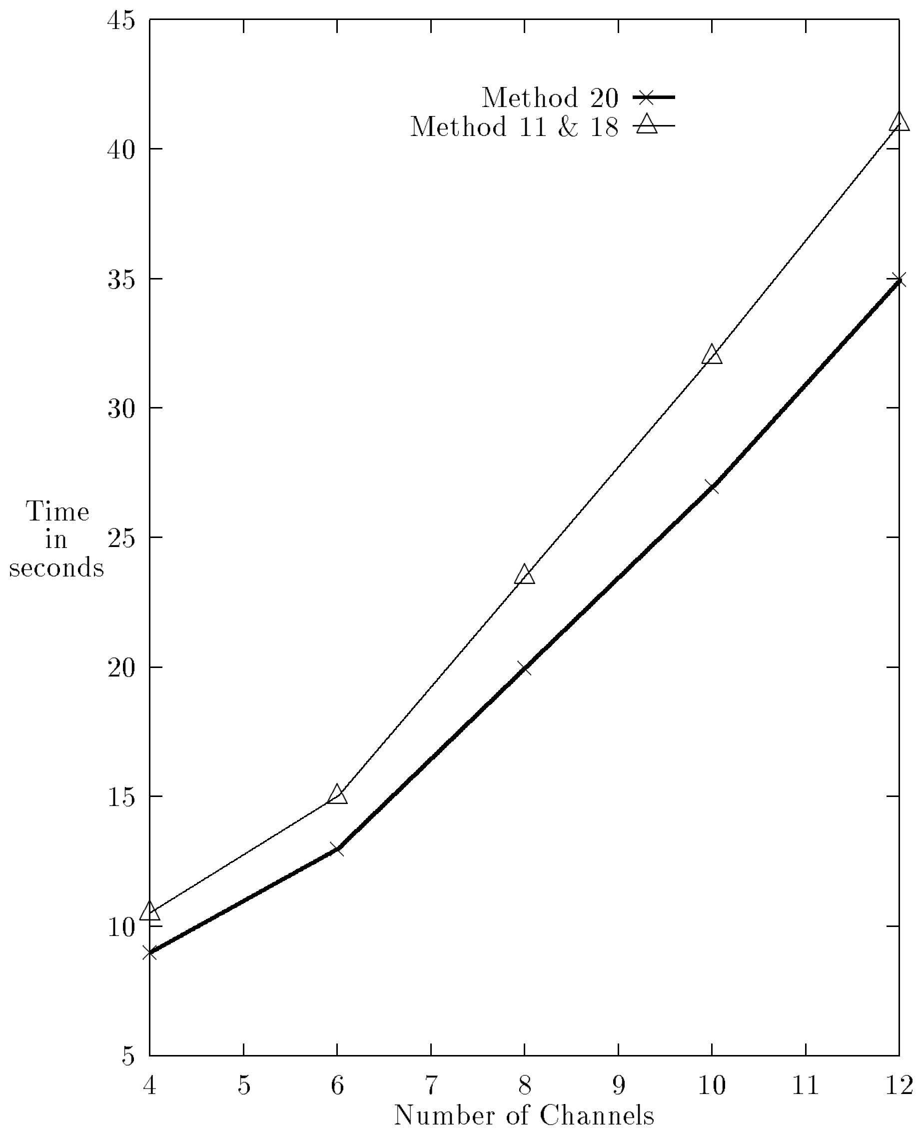

Figure.6 Time vs Dimensionality observations on PC/AT for a sample set size of 2600.

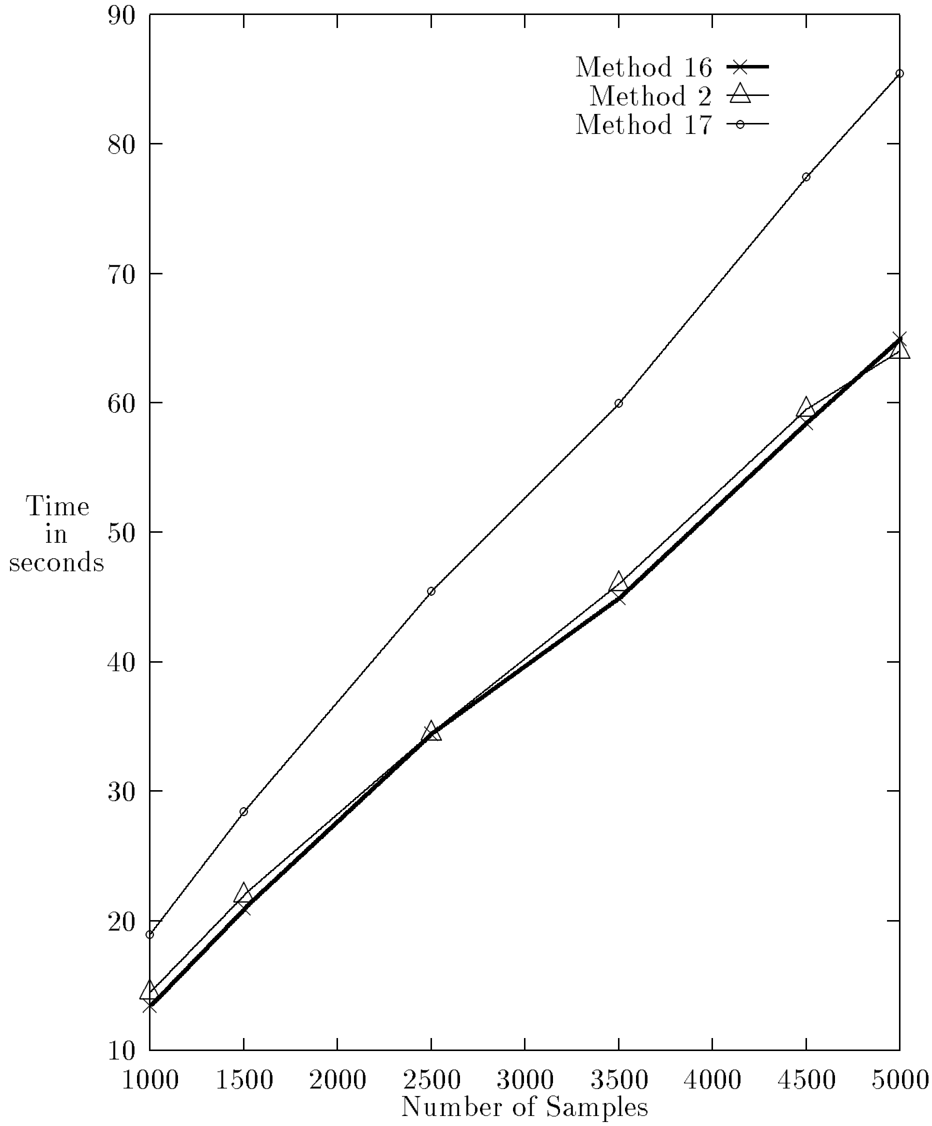

Figure.7 Time vs Sample set size observations on PC/AT for 12 band data.

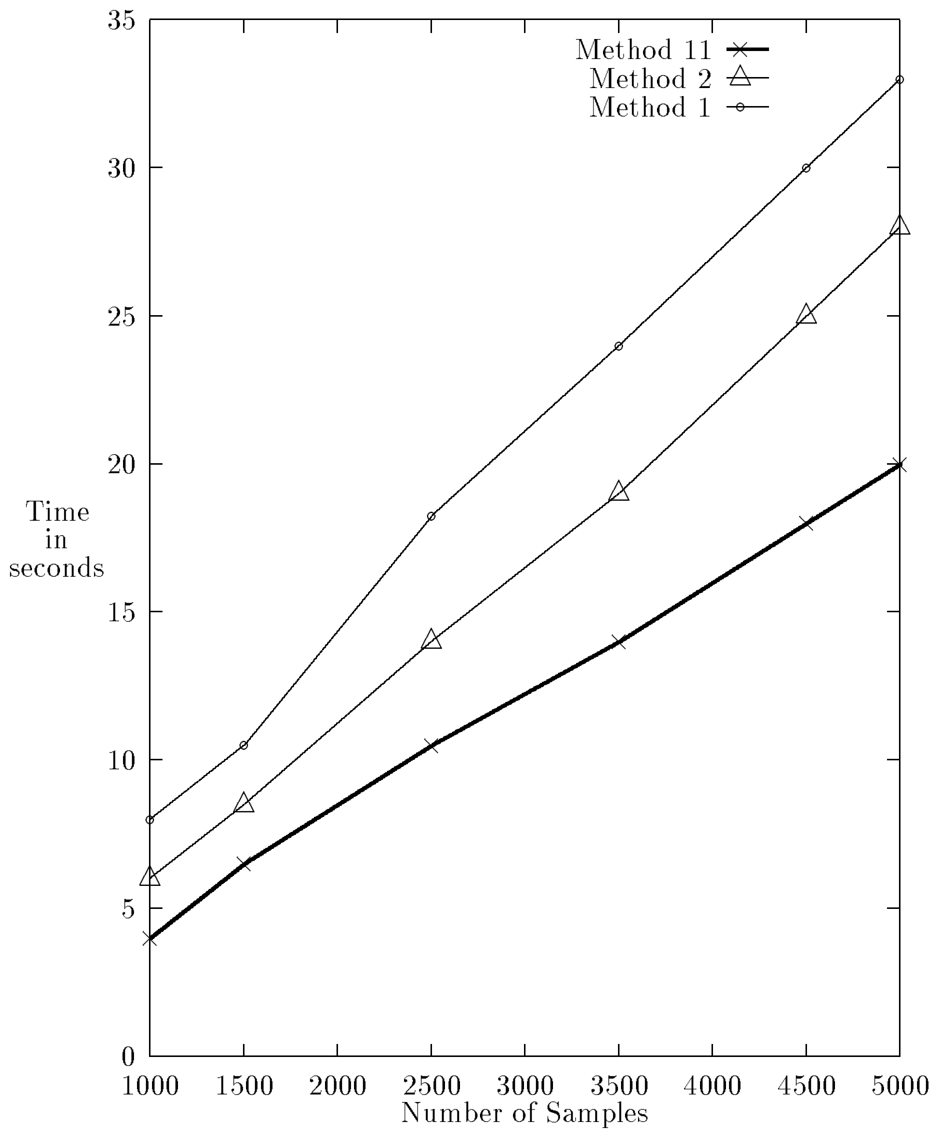

Figure.8 Time vs Sample set size observations on PC/AT for 4 band data.

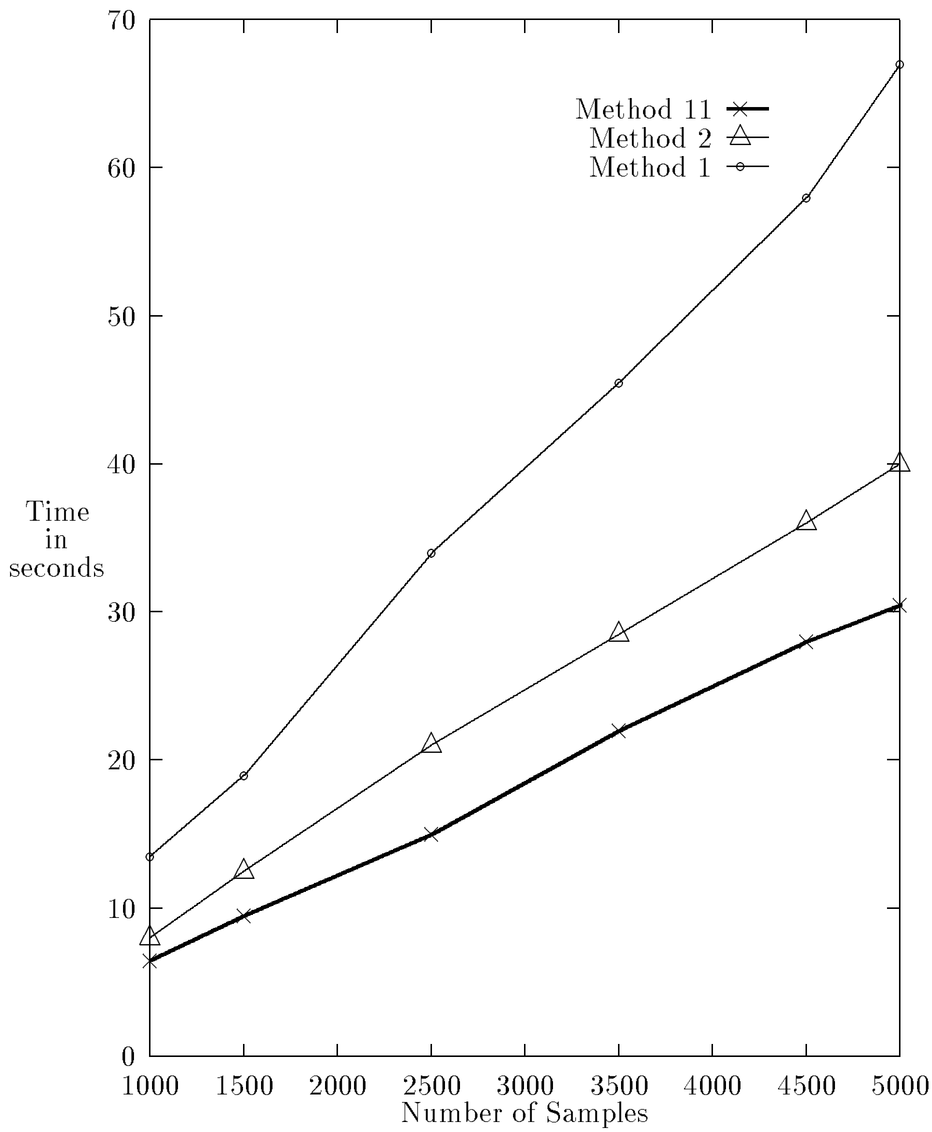

Figure.9 Time vs Sample set size observations on PC/AT for 6 band data.

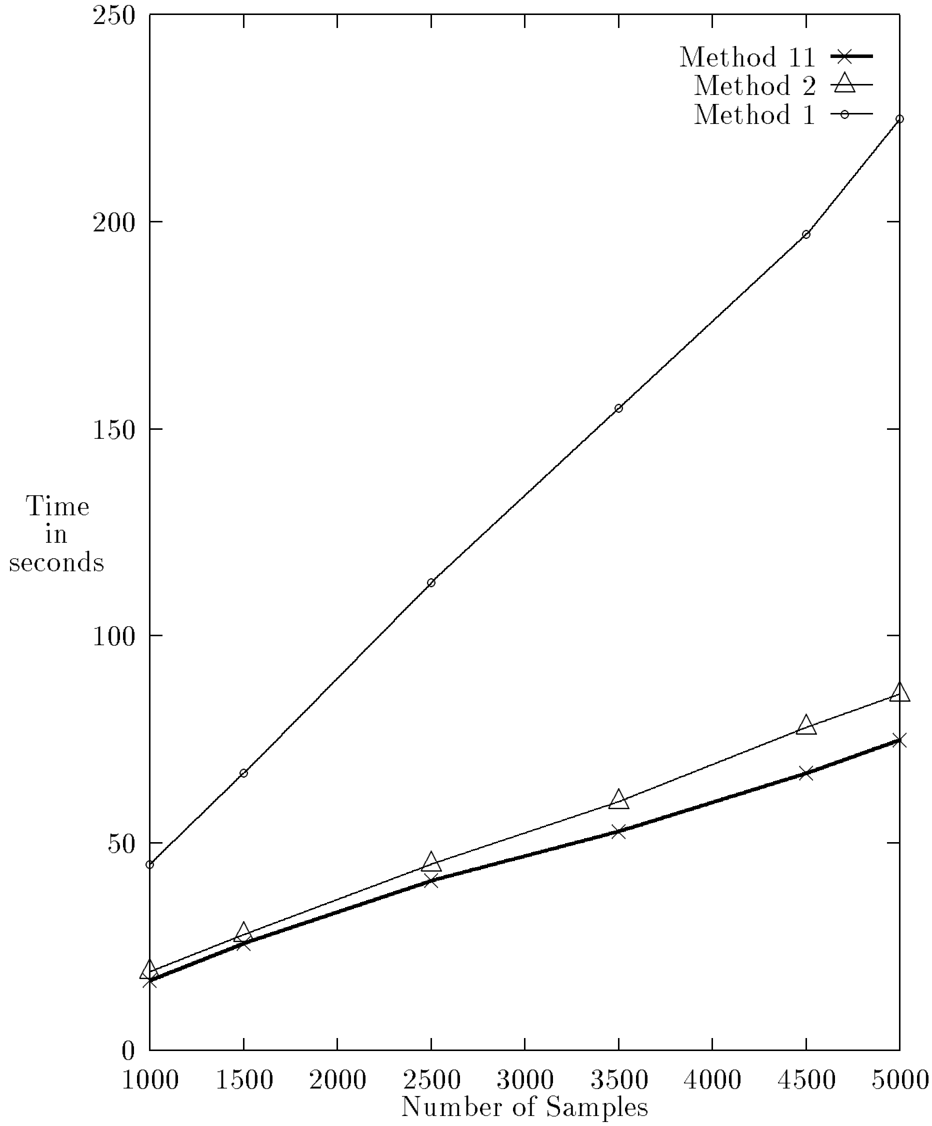

Figure.10 Time vs Sample set size observations on PC/AT for 12 band data.

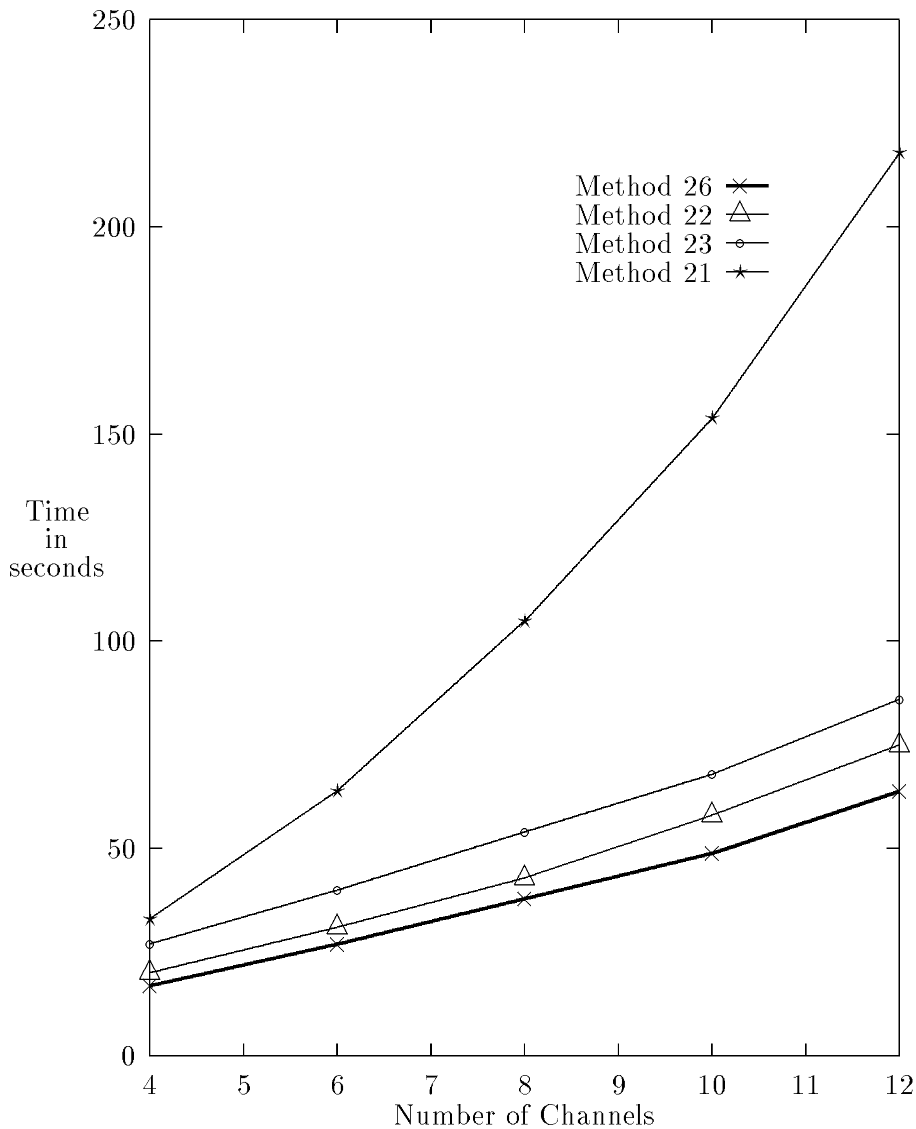

Figure.11 Time vs Dimensionality observations on PC/AT for MML classifier.

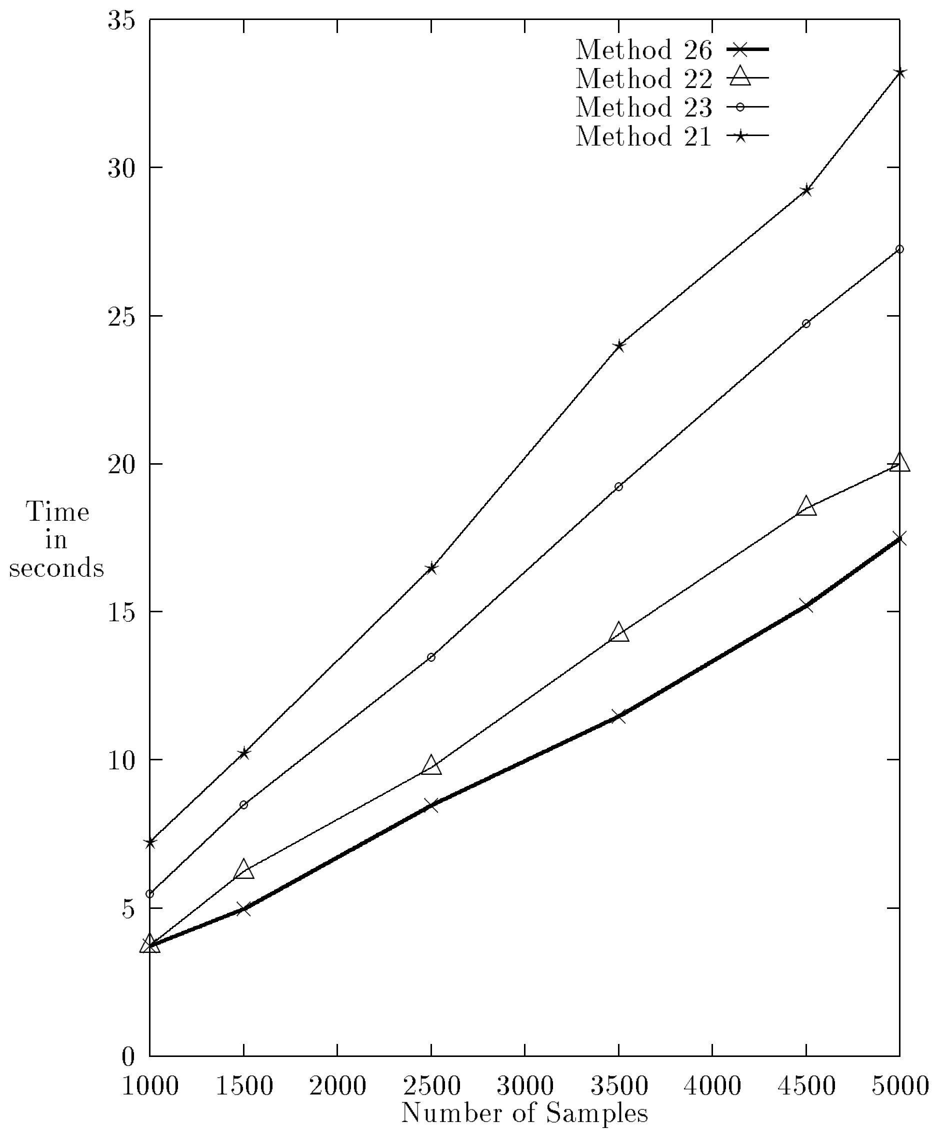

Figure.12 Time vs Sample set size observations on PC/AT for 4 band data.

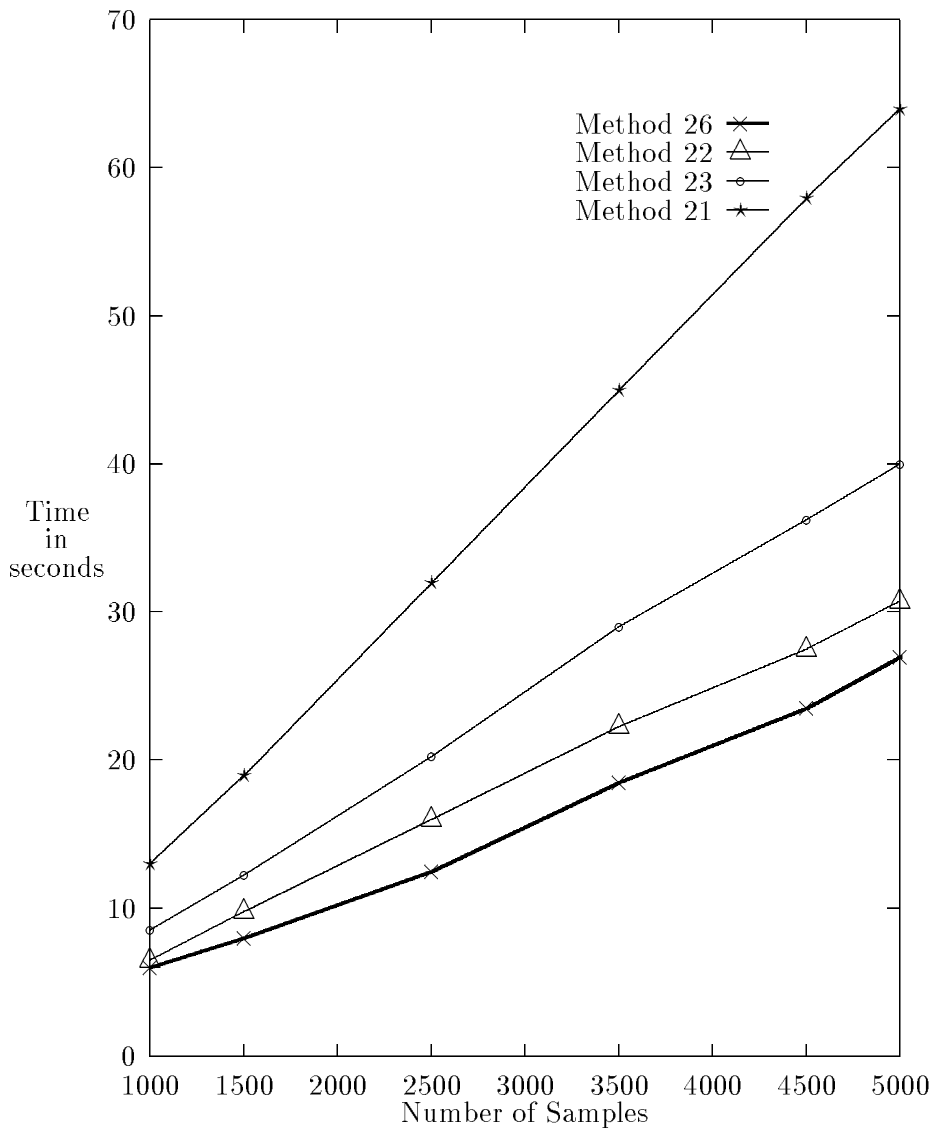

Figure.13 Time vs Sample set size observations on PC/AT for 6 band data.

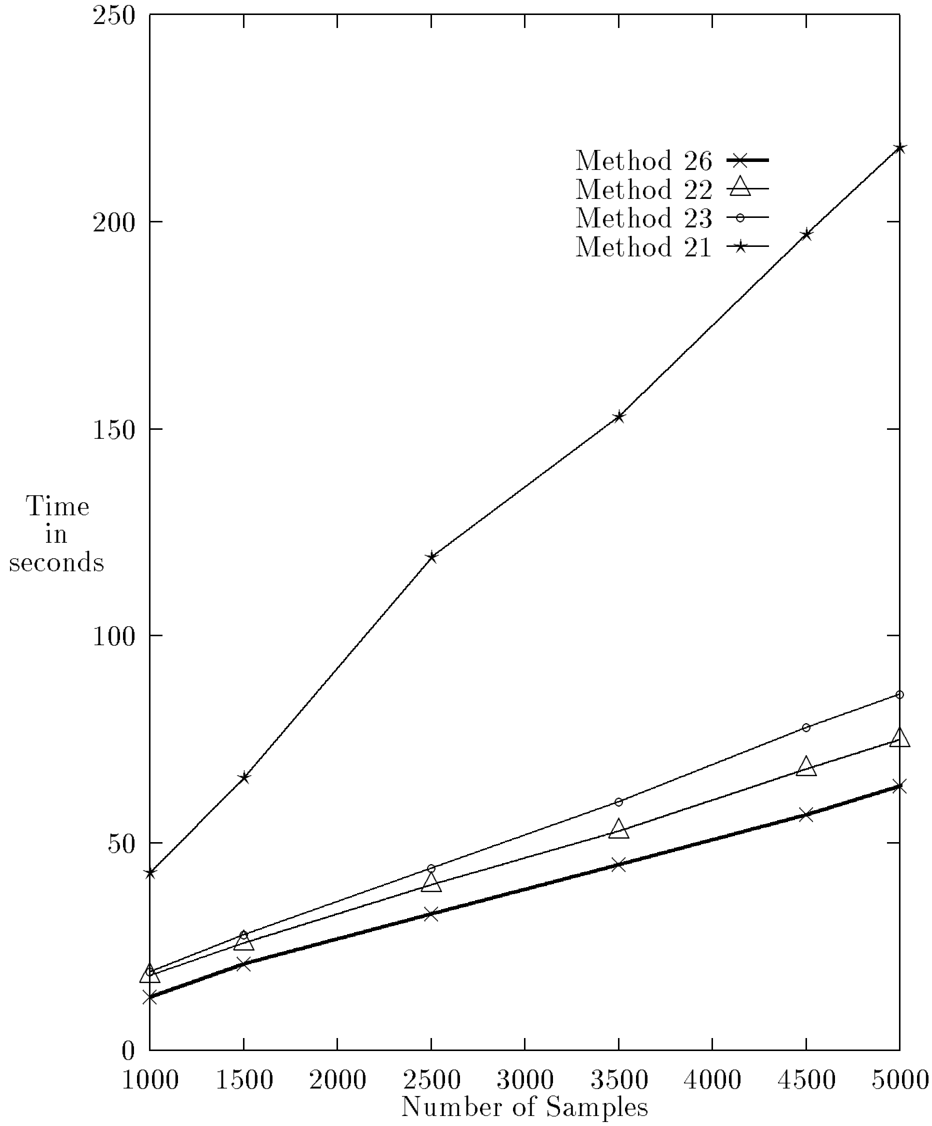

Figure.14 Time vs Sample set size observations on PC/AT for 12 band data.

APPENDIX
THREE STAGE ML CLASSIFIER (Venkateswarlu and Raju[1])

1. Calculate $M_k, \sum_k, \sum_k^{-1}, |\sum_k|, E_{k,d}, E_{k,1},\ k = 1, \ldots, m$ using the training data.

2. Read new random pattern vector X.

   Stage I

3. Calculate ranges of discriminant function using equation (18) for each group. Say, $d_{k,min}(X), d_{k,max}(X)$ are minimum and maximum of the discriminant function of k^{th} group.

4. Find minimum of $d_{k,max}(X), k = 1, \ldots, m$, and label the corresponding group number as *Iclass*. *Iclass* is the most probable class.

5. Let $A = \{j/j = 1, \ldots, m,\ and\ j \neq Iclass\}$. (That is, set A contains the numbers of groups other than *Iclass*).

6. For each $j \epsilon A$, remove j from$A$ if $d_{j,min}(X) > d_{Iclass,max}(X)$. If A is empty, then classify X as group *Iclass* and go to Step 2.

   Stage II

7. Otherwise calculate actual discriminant function value for group *Iclass*, say it is $d_{Iclass}(X)$, using the LTCF approach.

8. For each $j \epsilon A$, remove j from A if $d_{j,min}(X) > d_{Iclass}(X)$. If A is an empty, then classify X as belonging to *Iclass* and go to Step 2.

   Stage III

9. For each $j \epsilon A$, calculate the actual discriminant function $d_j(X)$ using the LTCF approach. If this $d_j(X) > d_{Iclass}(X)$, then remove j from A. Otherwise *Iclass* =j, then remove j from A. Finally If A is empty then X is classified as group *Iclass*. Go to step 2.