Stability Analysis of Predictor Based Least Squares Algorithm and Finite Precision Arithmetic Error Effects

Youhua Wang and Kenji Nakayama

Department of Electrical and Computer Engineering

Kanazawa University, Kanazawa 920, Japan

Abstract— *The numerical property of the recursive least squares (RLS) algorithm has been extensively studied. However, very few investigations are reported concerning the numerical behavior of the predictor based least squares (PLS) algorithms that provide the same least square solutions as the RLS algorithm. This paper studies the numerical property of the backward PLS (BPLS) algorithm. First, the stability of the BPLS algorithm is verified by using state space method. Then, finite-precision arithmetic error effects on both the BPLS and the RLS algorithms are presented through computer simulations. Some important results are obtained, which demonstrate that the BPLS algorithm appears quite robust to round-off errors and provides a much more accuracy and stable numerical performance than that of the RLS algorithm under finite-precision implementation.*

1. INTRODUCTION

In solving the least square problem for transversal adaptive filters, the recursive least squares (RLS) algorithms are well known. The principle of the RLS algorithms is based on the so-called matrix inversion lemma in order to get the recursive equations. The RLS algorithms are characterized by a fast convergence rate and a high computational load. As concern to the numerical property, much research has been done. The results show that divergence phenomenon may occur if finite-precision arithmetic is used or the input signal is ill conditioned [1]-[4].

Another approach for solving the least square problem is to use the fast RLS (FRLS) algorithms. The principle of these algorithms is different from the RLS algorithms in that the relations of the forward and backward predictors and the gain vector are exploited, resulting in a fast convergence rate with much less computation. However, the numerical instability problem of the FRLS algorithms is so serious that they cannot be continuously used in real applications, especially under finite-precision implementation [5],[6].

The reason for the instability of the FRLS algorithms is that the hyperbolic rotation (causing the eigenvalues to go out of the unit circle) has to be operated on the backward predictor to get the recursive equation for computing the gain vector [5]. So if we assume the recursion involve both order- and time-update, the least square solution can be obtained by using either forward or backward predictor. Therefore, the stable structure of both forward and backward predictors are remained. This leads to the algorithms we called the predictor based least square (PLS) algorithms [7].

Although the PLS algorithms can be easily derived from the FRLS algorithms, very few investigations concerning their numerical properties are reported in the literature. In Ref.[7], we have introduced the PLS algorithms and investigated preliminarily their numerical performances. It has been shown that the PLS algorithms perform more stable than the RLS algorithm when the order of adaptive filter is large and the forgetting factor is small. Furthermore, since the symmetric property of the input correlation matrix is exploited, the computational load of the PLS algorithms is less than 50% of that of the RLS algorithm. Nevertheless, the investigation presented there was carried out by using a 32-bit floating-point arithmetic, the stability of the PLS algorithms in a finite word-length implementation remains unknown.

In this paper, the numerical property of the backward PLS (BPLS) algorithm is further studied. First, the BPLS algorithm is given in Sec.2 for convenience of analysis. In Sec.3, the stability of the BPLS algorithm is analized by using state space method. A state space model is set up for representing the relation between the order-update of the gain vector and the time-update of the tap-weight vector of the backward predictor. The stability of the BPLS algorithm under finite-precision implementation is also analized. These analyses are confirmed through computer simulations in Sec.4. A floating-point arithmetic with various word-length is used for simulations. Finite-precision arithmetic error effects on both the BPLS and the RLS algorithms are investigated. Two and three dimensional views of

0-7803-3679-8/96/$5.00 © 1996 IEEE

the eigenvalues of the state space model in finite-precision implementation are also presented. Finally, the conclusion is made in Sec.5.

2. Backward PLS Algorithm

Experiments show that the numerical behavior of the forward PLS (FPLS) algorithm is very similar to that of the BPLS algorithm. So only the BPLS algorithm is studied in this paper. For convenience of analysis, we write the BPLS algorithm below.

$$\psi_m(n) = \mathbf{c}_m^T(n-1)\mathbf{u}_m(n) + u(n-m) \quad (1)$$

$$B_m(n) = \lambda B_m(n-1) + \gamma_m(n)\psi_m^2(n) \quad (2)$$

$$\gamma_{m+1}(n) = \frac{\lambda B_m(n-1)}{B_m(n)}\gamma_m(n) \quad (3)$$

$$\mathbf{c}_m(n) = \mathbf{c}_m(n-1) - \psi_m(n)\mathbf{k}_m(n) \quad (4)$$

$$\mathbf{k}_{m+1}(n) = \begin{bmatrix} \mathbf{k}_m(n) \\ 0 \end{bmatrix} + \frac{\gamma_m(n)\psi_m(n)}{B_m(n)} \begin{bmatrix} \mathbf{c}_m(n) \\ 1 \end{bmatrix} \quad (5)$$

$$\alpha(n) = d(n) - \mathbf{w}_M^T(n-1)\mathbf{u}_M(n) \quad (6)$$

$$\mathbf{w}_M(n) = \mathbf{w}_M(n-1) + \alpha(n)\mathbf{k}_M(n) \quad (7)$$

where $\psi_m(n)$ is the backward a priori prediction error, $B_m(n)$ is the minimum power of the backward prediction error, $\gamma_m(n)$ is the conversion factor, $\mathbf{k}_m(n)$ is the gain vector, $\mathbf{c}_m(n)$ is the tap-weight vector of the backward predictor and $\mathbf{u}_m(n)$ is the tap-input vector, $\alpha(n)$ is the a priori estimation error, $d(n)$ is the desired signal, $\mathbf{w}_M(n)$ is the tap-weight vector of the adaptive filter.

To initialize the BPLS algorithm at time $n = 0$, set $\mathbf{c}_m(0) = \mathbf{0}_m$, $B_m(0) = \delta$, $\mathbf{k}_m(0) = \mathbf{0}_m$ and $\gamma_m(0) = 1$, where $m = 1, 2, \cdots M-1$, M is the order of the adaptive filter, δ is a small positive constant.

At each iteration $n \geq 1$, generate the first-order variables as follows:

$$\mathbf{k}_1(n) = \frac{u(n)}{\Phi_1(n)} \quad (8)$$

$$\gamma_1(n) = \frac{\lambda\Phi_1(n-1)}{\Phi_1(n)} \quad (9)$$

where $\Phi_1(n)$ is the first-order of the input correlation matrix that satisfies

$$\Phi_1(n) = \lambda\Phi_1(n-1) + u^2(n) \quad (10)$$

where $\Phi_1(0) = \delta$.

3. Stability Analysis of BPLS Algorithm

In this section, the stability of the BPLS algorithms is analyzed by using state space method. The analysis is based on the following conclusion [8]:

The linear time invariant discrete system in the following state space form

$$\mathbf{x}(n+1) = \mathbf{A}\mathbf{x}(n) \quad (11)$$

is stable (but not asymptotically stable) if and only if the eigenvalues of $\mathbf{A}$, $\lambda_j (j = 1, 2, \cdots J)$ satisfy

$$|\lambda_j| \leq 1 \quad (12)$$

Applying the conclusion to the linear time variant discrete system expressed by

$$\mathbf{x}(m+1, n+1) = \mathbf{A}(m, n)\mathbf{x}(m, n) \quad (13)$$

we get the stability condition as $|\lambda_j(m, n)| \leq 1$, where $j = 1, 2, \cdots J$, $m = 1, 2, \cdots M$, $n = 1, 2, \cdots$, $\lambda_j(m, n)$ are the eigenvalues of $\mathbf{A}(m, n)$.

The state space variables we choose for the BPLS algorithm are the gain vector and the tap-weight vector of the backward predictor. If the order- and time-update of these variables are stable, then we can say that the BPLS algorithm is stable.

To set up the state space model for the BPLS algorithm, we substitute Eq.(4) into Eq.(5) and use Eq.(2) to write

$$\mathbf{k}_{m+1}(n) = \frac{\lambda B_m(n-1)}{B_m(n)} \begin{bmatrix} \mathbf{k}_m(n) \\ 0 \end{bmatrix} + \frac{r_m(n)\psi_m(n)}{B_m(n)} \begin{bmatrix} \mathbf{c}_m(n-1) \\ 1 \end{bmatrix} \quad (14)$$

Let $\mathbf{k}_{m+1}^f(n)$ denote the vector of the first m elements of $\mathbf{k}_{m+1}(n)$, then Eq.(14) can be rewritten as

$$\mathbf{k}_{m+1}^f(n) = \frac{\lambda B_m(n-1)}{B_m(n)}\mathbf{k}_m^f(n) + \frac{r_m(n)\psi_m(n)}{B_m(n)}\mathbf{c}_m(n-1) \quad (15)$$

The state space representation of the BPLS algorithm can be expressed by combining Eqs.(15) and (4), which is written as

$$\begin{bmatrix} \mathbf{k}_{m+1}^f(n) \\ \mathbf{c}_m(n) \end{bmatrix} = \begin{bmatrix} \frac{\lambda B_m(n-1)}{B_m(n)}\mathbf{I}_m & \frac{\gamma_m(n)\psi_m(n)}{B_m(n)}\mathbf{I}_m \\ -\psi_m(n)\mathbf{I}_m & \mathbf{I}_m \end{bmatrix} \cdot \begin{bmatrix} \mathbf{k}_m^f(n) \\ \mathbf{c}_m(n-1) \end{bmatrix} \quad (16)$$

where $\mathbf{I}_m$ is an m-by-m identity matrix.

For convenience of analysis, we write Eq.(16) in the scalar form as

$$\begin{bmatrix} k_{m+1,j}^f(n) \\ c_{m,j}(n) \end{bmatrix} = \begin{bmatrix} \frac{\lambda B_m(n-1)}{B_m(n)} & \frac{\gamma_m(n)\psi_m(n)}{B_m(n)} \\ -\psi_m(n) & 1 \end{bmatrix} \cdot \begin{bmatrix} k_{m,j}^f(n) \\ c_{m,j}(n-1) \end{bmatrix} \quad (17)$$

where the subscripts m, j are the m-th order and the j-th element, respectively. In practical computations, we begin with $n = 1$ and $m = 1 \cdots M-1, j \leq m$, then $n = 2 \cdots$.

The characteristic equation of Eq.(17) is

$$\lambda^2(m,n) - (\beta_1 + 1)\lambda(m,n) + \beta_1 + \beta_2 = 0 \quad (18)$$

where $\beta_1 = \dfrac{\lambda B_m(n-1)}{B_m(n)}$ and $\beta_2 = \dfrac{\gamma_m(n)\psi_m^2(n)}{B_m(n)}$.

From Eq.(2), we know that $\beta_1 + \beta_2 = 1$. Apparently, this result is also true in finite-precision implementation. Hence the two roots of Eq.(18) are

$$\lambda_{1,2}(m,n) = \frac{\beta_1 + 1}{2} \pm \frac{\sqrt{(\beta_1+1)^2 - 4}}{2} \quad (19)$$

In Appendix, we show that $0 \leq \beta_1 \leq 1$ is satisfied in both infinite and finite precision arithmetic. So two cases should be considered:

1. If $\beta_1 = 1$, then $(\beta_1+1)^2-4 = 0$, so $\lambda_1(m,n) = \lambda_2(m,n) = 1$.
2. If $0 \leq \beta_1 < 1$, then $(\beta_1+1)^2-4 < 0$, resulting in two complex roots:

$$\lambda_{1,2}(m,n) = \frac{\beta_1 + 1}{2} \pm j\frac{\sqrt{4-(\beta_1+1)^2}}{2} \quad (20)$$

Eq.(20) is equivalent to

$$\lambda_{1,2}(m,n) = e^{\pm j\theta} \quad (21)$$

with $\theta = sin^{-1}\Big(\dfrac{\sqrt{4-(\beta_1+1)^2}}{2}\Big)$ or $\theta = cos^{-1}\Big(\dfrac{\beta_1+1}{2}\Big)$.

So in both cases, we get

$$|\lambda_1(m,n)| = |\lambda_2(m,n)| = 1 \quad (22)$$

It is interesting to note that even though the state space model of the BPLS algorithm is time-varying, the two eigenvalues are independent of order m and time n and will always locate on the unit circle. We also note that the round-off errors produced by finite-precision implementation will not cause the eigenvalues to go out of the unit circle as long as the condition $0 \leq \beta_1 \leq 1$ is held. In another word, the round-off errors will cause the eigenvalues of Eq.(19) only to change their positions on the unit circle but without going out of it. These conclusions are also supported by various simulation results as shown in Sec.4. Therefore the stability of the BPLS algorithm is verified. The stability of the FPLS algorithm can be proved following the same procedure.

Like the RLS algorithm, the BPLS algorithm will be unstable when the forgetting factor $\lambda = 1$. This is because the value of $\gamma_m(n)\psi_m^2(n)$ in Eq.(2) is non-negative. If $\lambda = 1$, then the value of $B_m(n)$ will be increased without bound and eventually leads to the overflow of these variables.

4. Finite Precision Arithmetic Error Effects

Extensive studies have been made concerning the numerical stability of the RLS algorithm. However, much of them considers only single or local round-off error effects [2]-[4]. In this section, a complete finite-precision implementation will be carried out on both the BPLS and the RLS algorithms in order to give a fare comparison for these two algorithms.

An adaptive equalizer is employed for the simulation. Its block diagram is shown in Fig.1. The blocks enclosed by the dashed line are implemented by using a floating-point arithmetic that consists of an 8-bit exponent and a variable mantissa (including a sign bit). The block labeled Q quantize double-precision input data into finite-precision ones that are used in the adaptive filter algorithm. The impulse response of the channel is [9]

$$h(n) = \begin{cases} \frac{1}{2}\{1 + cos(\frac{2\pi}{W}(n-2))\} & n = 1,2,3 \\ 0 & otherwise \end{cases} \quad (23)$$

where W controls the eigenvalue spread χ produced by the channel. We choose $W = 3.5(\chi \approx 47)$. White noise and speech signal are used as the input $u(n)$ to the channel. the additive noise $v(n)$ is a white noise with zero mean. The variances of the channel output and $v(n)$ are unity and 0.001, respectively. The desired signal $d(n)$ is obtained from $u(n)$ after a delay of seven samples. The adaptive equalizer has 11 taps.

The RLS algorithm used in the simulation is the stable version (version II) as shown in Ref.[9]. The initial parameter $\delta = 0.1$ and the forgetting factor

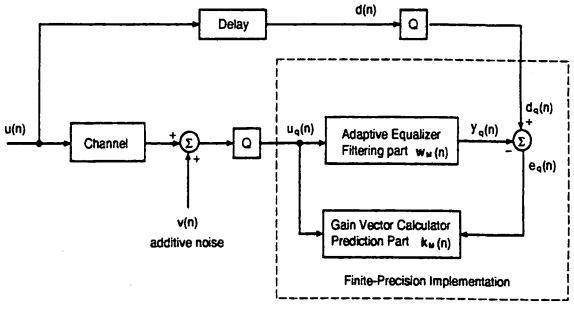

Fig. 1. Block diagram of adaptive equalizer

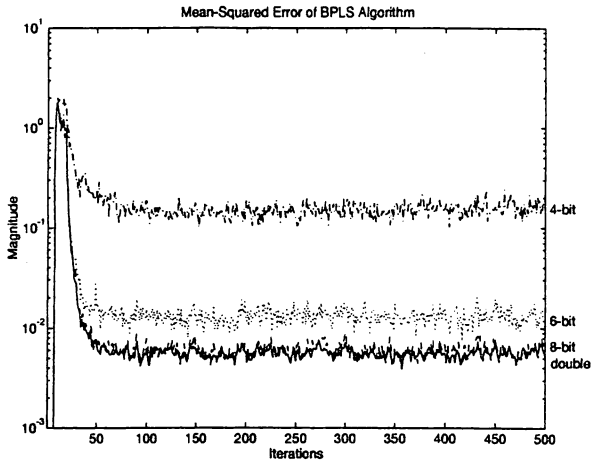

Fig. 2. Finite-precision MSE of BPLS algorithm (white noise input). Double precision (solid line), 8-bit (dashed line), 6-bit (dotted line) and 4-bit (dashdot line).

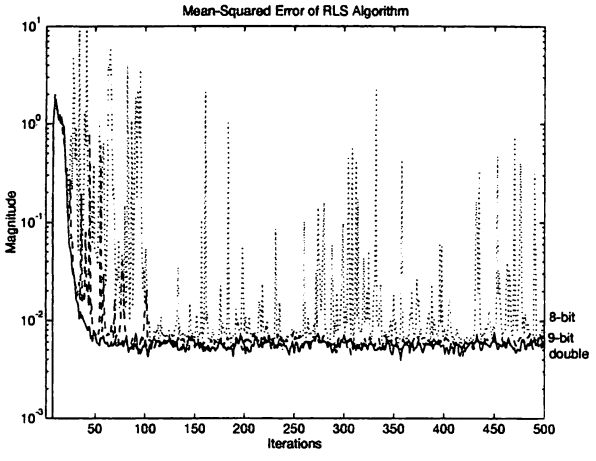

Fig. 3. Finite-precision MSE of RLS algorithm (white noise input). Double precision (solid line), 9-bit (dashed line) and 8-bit (dotted line).

$\lambda = 0.95$ are used in both the RLS and the BPLS algorithms.

For white noise input, the simulation results are shown in Fig.2, 3 and Table 1. For speech signal input, the speech signal is shown in Fig.4 and the simulation results are shown in Fig.5, 6 and Table 2. The results are obtained by ensemble averaging of 200 independent trials.

For the simulation results of the BPLS algorithm, it has been confirmed that the eigenvalues $\lambda_{1,2}(m,n)$ $(m = 1, 2, \cdots M, n = 1, 2, \cdots 500)$ of the state space model are all on the unite circle. Figure 7 and 8 show only one of the results, which indicate the positions of the eigenvalues with respect to time recursions. From Eq.(20), the real part and the imaginary part of the eigenvalues can

TABLE 1

MSE of BPLS and RLS Algorithms (white noise input)

Mantissa Bits	BPLS Algorithm	RLS Algorithm
Double precision	0.005522	0.005522
16	0.005523	0.005523
10	0.005600	0.005611
9	0.005710	0.006966
8	0.005959	unstable
6	0.013074	unstable
4	0.157987	unstable

be computed as

$$Re(\lambda_{1,2}(m,n)) = \frac{1}{2}\Big(\frac{\lambda B_m(n-1)}{B_m(n)} + 1\Big) \tag{24}$$

$$Im(\lambda_{1,2}(m,n)) = \pm\frac{1}{2}\sqrt{4 - \Big(\frac{\lambda B_m(n-1)}{B_m(n)} + 1\Big)^2} \tag{25}$$

Since the difference between $\lambda_1(m,n)$ and $\lambda_2(m,n)$ is a sign in the imaginary part, only $\lambda_1(m,n)$ is shown in the figure. As expected, the finite-precision implementation does not make the eigenvalues go out of the unite circle even though the speech signal is used as input. Figure 7 also shows that most of the eigenvalues are concentrated on the unit circle near the real axis. This is because the forgetting factor $\lambda = 0.95$ chosen for the simulations is near to 1, the real part of the eigenvalues in Eq.(24) tends to unity and the imaginary part in Eq.(25) tends to zero.

These results clearly demonstrate that in both white noise and speech signal input cases, the numerical performance of the BPLS algorithm is very robust to finite word-length implementations. On the other hand, the RLS algorithm is seriously affected by finite-word length as well as the characteristic of the input signal. This is consistent with the analysis provided in Ref.[10] in which

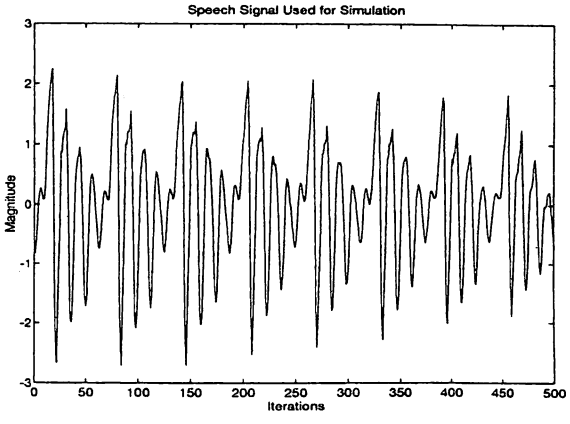

Fig. 4. Speech signal used for simulation.

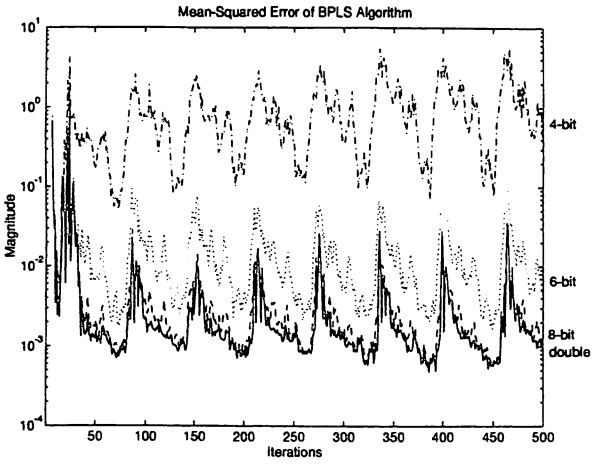

Fig. 5. Finite-precision MSE of BPLS algorithm (speech signal input). Double precision (solid line), 8-bit (dashed line), 6-bit (dotted line) and 4-bit (dashdot line).

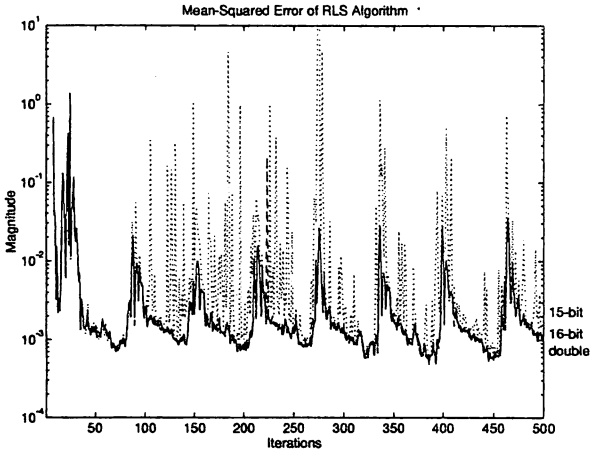

Fig. 6. Finite-precision MSE of RLS algorithm (speech signal input). Double precision (solid line), 16-bit (dashed line) and 15-bit (dotted line)

it was shown that when computing in an m-digit arithmetic, the numerical problem of the conventional RLS algorithm will occur if the eigenvalue spread of the inverse correlation matrix $\chi(\Phi^{-1}(n))$ approaches 2^m. It was also pointed out that the use of the square root RLS algorithms will double the arithmetic precision, therefore the numerical problem will not occur until 2^{2m}. So we may conjecture that the numerical performance of the BPLS algorithm is also superior to that of the square root RLS algorithms.

5. Conclusion

The numerical property of the BPLS algorithm has been further studied. First, the stability of BPLS algorithm in finite-precision implementation is analyzed by using the state space method. A state space model has been set up for representing the relation between the order- and time-update of the gain vector and the tap-weight vector of the backward predictor. It has been shown that the eigenvalues of the state space model are independent of order and time recursions of the BPLS algorithm and always locate on the unit circle. So the BPLS algorithm is stable. The stability of the BPLS algorithm has been confirmed through computer simulations on a variety of floating-point word-length implementations. Some important results have been obtained, which clearly indicate that the BPLS algorithm provides a much more accurate and stable least square solution compared with the RLS algorithm. We believe that it is very promising that the use of the RLS algorithm and its square root versions may be replaced by the PLS algorithms.

TABLE 2

MSE of BPLS and RLS Algorithms (speech signal input)

Mantissa Bits	BPLS Algorithm	RLS Algorithm
Double precision	0.002599	0.002599
16	0.002618	0.003324
15	0.002631	unstable
8	0.003293	unstable
6	0.014401	unstable
4	1.175265	unstable

Appendix

To show $0 \leq \beta_1 = \dfrac{\lambda B_m(n-1)}{B_m(n)} \leq 1$, we write Eq.(2) as

$$\begin{aligned} B_m(n) &= \lambda B_m(n-1) + \gamma_m(n)\psi_m^2(n) \\ &= \lambda^n \delta + \sum_{i=1}^{n} \lambda^{n-i}\gamma_m(i)\psi_m^2(i) \qquad (A.1) \end{aligned}$$

where λ is the forgetting factor and δ is a small positive constant. From Eq.(A.1), we can see that the value of $B_m(n)$ will never be negative if $\gamma_m(i) \geq 0$. In another word, if $\gamma_m(i) \geq 0$, then $0 \leq \dfrac{\lambda B_m(i-1)}{B_m(i)} \leq 1$. $\gamma_m(i)$ is computed by using Eq.(3), which involves only an order-update recursion. So at every time-update recursion, we begin from $\gamma_1(i) = \dfrac{\lambda \Phi_1(i-1)}{\Phi_1(i)}$ (see Eq.(9)). Apparently, $0 \leq \gamma_1(i) \leq 1$. Hence, $B_1(i) \geq 0$ or $0 \leq \dfrac{\lambda B_1(i-1)}{B_1(i)} \leq 1$. Then we compute $\gamma_2(i)$ from Eq.(3) as $\gamma_2(i) = \dfrac{\lambda B_1(i-1)}{B_1(i)}\gamma_1(i)$. From finite-

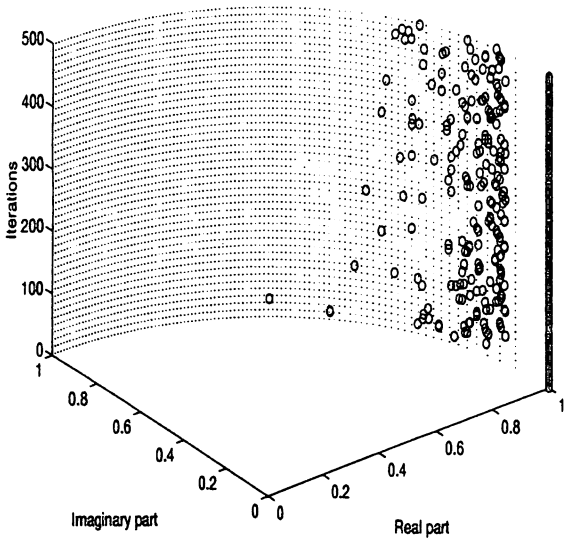

Fig. 7. Positions of eigenvalues (shown by circles) on the unit circle (three dimensional view). Simulation conditions: Speech signal input, 6-bit mantissa, $\lambda_1(M,n)$, where $M = 11$ and $n = 1, 2, \cdots 500$.

precision arithmetic we know that for multiplication with fractions, overflow can never occur since the product of two fractions is a fraction. Furthermore, if fractions are greater or equal to zero, then the product can never be negative. Therefore, $0 \leq \gamma_2(i) \leq 1$ is true. Following this procedure, we can conclude that $0 \leq \gamma_m(i) \leq 1$ is always satisfied despite finite-precision implementation.

ACKNOWLEDGMENT

The authors wish to thank Mr. T. Yoshida of Matsushita Communication Kanazawa R&D Labs. for help in finite-precision simulations and Dr. K. Ikeda of Kanazawa University for useful discussions.

REFERENCES

[1] F. Ling and G. Proakis, "Numerical accuracy and stability: two problems of adaptive estimation algorithms caused by round-off error", *Proc. ICASSP'84*, pp.30.3.1-30.3.4, 1984.

[2] S. Ljung and L. Ljung, "Error propagation properties of recursive least-squares adaptation algorithms," *Automatica*, vol.21, pp.157-167, 1985.

[3] M.H. Verhaegen, "Round-off error propagation in four generally-applicable, recursive, least-squares estimation schemes," *Automatica*, vol.25, no.3 pp.437-444, 1989.

[4] G.E Bottomley and S.T. Alexander, "A theoretical basis for the divergence of conventional recursive least squares filters," *Proc. ICASSP'89*, pp.908-911, 1989.

[5] J.M. Cioffi, "Limited-precision effects in adaptive filtering," *IEEE Trans. Circuits and Systems*, CAS-34, pp.821-833, 1987.

[6] D.W. Lin, "On digital implementation of the fast Kalman algorithm," *IEEE Trans. Acoust., Speech, Signal Processing*, vol.ASSP-32, no.5, pp.998-1005, 1984.

[7] Y. Wang and K. Nakayama, "A stable least square algorithm based on predictors and its application to fast newton transversal filters," *Trans. IEICE*, vol.E78-A, no.8, pp.999-1003, 1995.

[8] T. Kailath, *Linear System*, Prentice Hall, 1980.

[9] S. Haykin, *Adaptive Filter Theory*, Second Edition, Prentice Hall, 1991.

[10] R.W. Stewart and R. Chapman, "Fast stable Kalman filter algorithms utilising the square root," *Proc. ICASSP'90*, pp.1815-1818, 1990.

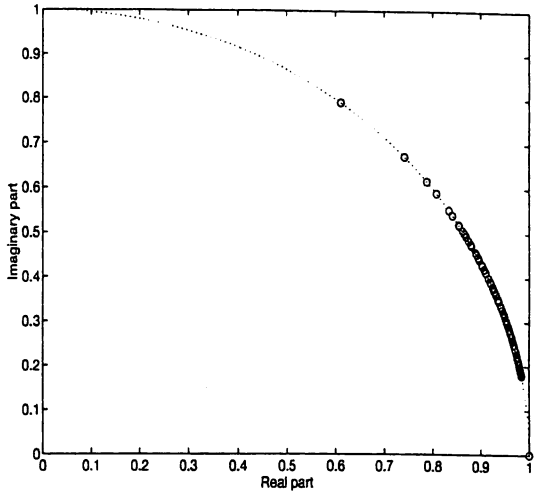

Fig. 8. Positions of eigenvalues (shown by circles) on the unit circle (two dimensional view of Fig.7).