1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32
33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48
49	50	51	52	53	54	55	56	57	58	59	60	61	62	63	64
65	66	67	68	69	70	71	72	73	74	75	76	77	78	79	80
81	82	83	84	85	86	87	88	89	90	91	92	93	94	95	96
97	98	99	100	101	102	103	104	105	106	107	108	109	110	111	112
113	114	115	116	117	118	119	120	121	122	123	124	125	126	127	128
129	130	131	132	133	134	135	136	137	138	139	140	141	142	143	144
145	146	147	148	149	150	151	152	153	154	155	156	157	158	159	160
161	162	163	164	165	166	167	168	169	170	171	172	173	174	175	176
177	178	179	180	181	182	183	184	185	186	187	188	189	190	191	192
193	194	195	196	197	198	199	200	201	202	203	204	205	206	207	208
209	210	211	212	213	214	215	216	217	218	219	220	221	222	223	224
225	226	227	228	229	230	231	232	233	234	235	236	237	238	239	240
241	242	243	244	245	246	247	248	249	250	251	252	253	254	255	255
254	253	252	251	250	249	248	247	246	245	244	243	242	241	240	239
238	237	236	235	234	233	232	231	230	229	228	227	226	225	224	223
222	221	220	219	218	217	216	215	214	213	212	211	210	209	208	207
206	205	204	203	202	201	200	199	198	197	196	195	194	193	192	191
190	189	188	187	186	185	184	183	182	181	180	179	178	177	176	175
174	173	172	171	170	169	168	167	166	165	164	163	162	161	160	159
158	157	156	155	154	153	152	151	150	149	148	147	146	145	144	143
142	141	140	139	138	137	136	135	134	133	132	131	130	129	128	127
126	125	124	123	122	121	120	119	118	117	116	115	114	113	112	111
110	109	108	107	106	105	104	103	102	101	100	99	98	97	96	95
94	93	92	91	90	89	88	87	86	85	84	83	82	81	80	79
78	77	76	75	74	73	72	71	70	69	68	67	66	65	64	63
62	61	60	59	58	57	56	55	54	53	52	51	50	49	48	47
46	45	44	43	42	41	40	39	38	37	36	35	34	33	32	31
30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15
14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	0

(k)

Fig. 3. (*Continued.*) (k) Split each number obtained by IBZT into two numbers so that each row has 32 numbers instead of 16 numbers and then add the last 16 numbers of each row to the first 16 numbers of the next row to get the final result—the linear convolution of the input data sequence and the impulse response (Box 10).

IV. Summary

It is known that the Fermat number transform (FNT) is not practical for FIR filtering of a long impulse response due to the relationship of word length and transform length. The straightforward method of using short FNT's to compute a long convolution requires an order of N^2 short FNT's. The modified Fermat number transform (MFNT) increases efficiency by a factor of 2; in other words, with the same amount of computations, filter length can be doubled by using the MFNT instead of the FNT. However, it is still not practical to use the MFNT for FIR filtering of a very long impulse response. In this correspondence, the block Z transform (BZT) is introduced. The BZT does not require any multiplies and can reduce the number of right-angle circular convolutions (RCC's) from an order of N^2 in the time domain to an order of N in the transform domain. RCC can be computed efficiently by the MFNT and linear convolution of two sequences can be obtained easily from RCC. Therefore, the BZT-MFNT algorithm is very efficient for FIR filtering of a long impulse response. It is also shown that the BZT-MFNT algorithm can be implemented in a regular structure. A computer program has been developed for implementing the algorithm and a numerical example of running the program is shown in Fig. 3.

References

[1] H. J. Nussbaumer, *Fast Fourier Transform and Convolution Algorithms.* New York: Springer, 1982.

[2] W. Li and A. M. Peterson, "FIR filtering by the modified Fermat number transform," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. 38, no. 9, pp. 1641-1645, Sept. 1990.

[3] T. K. Truong, I. S. Reed, C.-S. Yeh, and H. M. Shao, "A parallel VLSI architecture for a digital filter of arbitrary length using Fermat number transforms," in *Proc. IEEE Int. Conf. Circuits Comput. (ICCC'82)*, (New York), Sept. 28-Oct. 1, 1982, pp. 574-578.

[4] C.-S. Yeh, I. S. Reed, J. J. Chang, and T. K. Truong, "VLSI design of number-theoretic transforms for a fast convolution," in *Proc. IEEE Int. Conf. Comput. Design: VLSI in Comput. (ICCD'83)* (Port Chester, NY), Oct. 31-Nov. 3, 1983, pp. 200-203.

A Modular Structure for Implementation of Linearly Constrained Minimum Variance Beamformers

Tsung-Ching Liu and Barry Van Veen

Abstract—**This correspondence describes a cascade decomposition of the generalized sidelobe canceller (GSC) implementation for linearly constrained minimum variance beamformers. The GSC is initially separated into an adaptive interference cancellation module followed by a nonadaptive beamformer. We prove that the adaptive interference cancellation module can be decomposed into a cascade of first (or higher) order adaptive interference cancellation modules, where the order corresponds to the number of adaptive degrees of freedom represented in the module. This distributes the computational burden associated with determining the adaptive weights over several lower order problems and facilitates simultaneous implementation of beamformers with differing numbers of adaptive degrees of freedom.**

I. Introduction

A popular criterion for choosing the weights in an adaptive beamformer is minimization of output variance (power) subject to

Manuscript received December 28, 1990; revised May 18, 1991. This work was supported in part by the Army Research Office under Grant DAAL03-89-K-0141 and by the National Science Foundation under Award MIP-895 8559.

The authors are with the Department of Electrical and Computer Engineering, University of Wisconsin, Madison, WI 53706.

IEEE Log Number 9101871.

1053-587X/91/1000-2343$01.00 © 1991 IEEE

a set of linear constraints on the weights. The constraints are usually chosen to control the adapted response of the beamformer over specified directions and frequencies [1]. The generalized sidelobe canceller (GSC) [2] implementation decomposes the weight vector into constrained and unconstrained components where the unconstrained components represent the adaptive degrees of freedom in the beamformer. The primary advantage of the GSC is that the weights are adapted using unconstrained adaptive algorithms.

In this correspondence we use the GSC structure to show that linearly constrained minimum variance beamformers can be decomposed into a cascade of adaptive "modules," each of which represents one (or several) adaptive degrees of freedom, followed by a nonadaptive beamformer. This decomposition offers several implementation advantages. First, the total number of adaptive degrees of freedom are distributed throughout the individual modules; hence, the computational burden associated with determining the adaptive weights is distributed over several lower order problems. Second, beamformers with different numbers of adaptive degrees of freedom can be implemented simultaneously in this structure with only minor additional computational cost by applying a nonadaptive beamformer to the output of each module. Similarly, the number of adaptive degrees of freedom can be changed without having to recompute the remaining adaptive weights. Analogous to a lattice filter, the adaptive components of each module depend only on preceding modules in the cascade. Finally, beamformers with different quiescent responses can also be implemented simultaneously without recomputing the adaptive weights.

The outline of this correspondence is as follows. Section II briefly reviews linearly constrained minimum variance beamforming and partially adaptive beamforming. The partially adaptive beamforming results are used to motivate implementation of beamformers with different numbers of adaptive degrees of freedom. The modular structure is derived from the GSC in Section III. Advantages and disadvantages of this structure are discussed in Section IV.

II. Linearly Constrained Minimum Variance Beamforming

Let $\boldsymbol{x}$ be an N dimensional data vector and $\boldsymbol{w}$ the N dimensional vector of beamformer weights so that the beamformer output is given by $y = \boldsymbol{w}^h\boldsymbol{x}$. The linearly constrained minimum variance criterion for choosing the weight vector $\boldsymbol{w}$ is

$$\min_{\boldsymbol{w}} \boldsymbol{w}^h\boldsymbol{R}_x\boldsymbol{w} \quad \text{subject to } \boldsymbol{U}^h\boldsymbol{w} = \boldsymbol{g}. \tag{1}$$

Here $\boldsymbol{R}_x = \mathcal{E}(\boldsymbol{x}\boldsymbol{x}^h)$ is the covariance matrix of the data, $\boldsymbol{U}$ is an N by L constraint matrix and $\boldsymbol{g}$ is an L dimensional response vector. $\boldsymbol{U}$ and $\boldsymbol{g}$ control the beamformer's response over specified regions of frequency and/or direction [1]. The solution to (1) is

$$\boldsymbol{w} = \boldsymbol{R}_x^{-1}\boldsymbol{U}(\boldsymbol{U}^h\boldsymbol{R}_x^{-1}\boldsymbol{U})^{-1}\boldsymbol{g}. \tag{2}$$

The GSC represents the beamformer weights as a sum of two orthogonal components, one that lies in the space spanned by the columns of $\boldsymbol{U}$ and another that lies in the space orthogonal to the columns of $\boldsymbol{U}$. Let the columns of an $N \times (N - L)$ matrix $\overline{\boldsymbol{U}}$ represent a basis for the space orthogonal to the columns of $\boldsymbol{U}$. This implies that $\boldsymbol{U}^h\overline{\boldsymbol{U}} = \boldsymbol{0}$ and that $[\boldsymbol{U}\overline{\boldsymbol{U}}]$ is a nonsingular matrix. Denote the component of $\boldsymbol{w}$ in the space spanned by the columns of $\boldsymbol{U}$ as $\boldsymbol{w}_q$. It is straightforward to show that $\boldsymbol{w}_q = \boldsymbol{U}(\boldsymbol{U}^h\boldsymbol{U})^{-1}\boldsymbol{g}$ in order to satisfy the constraints. The component of $\boldsymbol{w}$ in the space spanned by the columns of $\overline{\boldsymbol{U}}$ is not affected by the constraint; hence we have

$$\boldsymbol{w} = \boldsymbol{w}_q - \overline{\boldsymbol{U}}\boldsymbol{w}_a \tag{3}$$

where the $N - L$ dimensional vector $\boldsymbol{w}_a$ describes the coordinates of $\boldsymbol{w}$ in the space spanned by the columns of $\overline{\boldsymbol{U}}$. $\boldsymbol{w}_a$ is unconstrained and thus represents the available degrees of freedom in $\boldsymbol{w}$. The LCMV problem in (1) is reexpressed in terms of the GSC framework as the unconstrained optimization problem

$$\min_{\boldsymbol{w}_a} (\boldsymbol{w}_q - \overline{\boldsymbol{U}}\boldsymbol{w}_a)^h\boldsymbol{R}_x(\boldsymbol{w}_q - \overline{\boldsymbol{U}}\boldsymbol{w}_a). \tag{4}$$

The solution to (4) is given by

$$\boldsymbol{w}_a = (\overline{\boldsymbol{U}}^h\boldsymbol{R}_x\overline{\boldsymbol{U}})^{-1}\overline{\boldsymbol{U}}^h\boldsymbol{R}_x\boldsymbol{w}_q. \tag{5}$$

A block diagram depicting the GSC is given in Fig. 1. $\overline{\boldsymbol{U}}$ is sometimes termed the "signal blocking matrix" [2] since it prevents the signal from entering the lower path provided the constraints are designed to control the beamformer response to the signal. $\boldsymbol{w}_q$ is known as the quiescent weight vector since under quiescent conditions $\boldsymbol{R}_x = \sigma^2\boldsymbol{I}$ and $\boldsymbol{w} = \boldsymbol{w}_q$.

In a partially adaptive beamformer the number of degrees of freedom are reduced below $N - L$. This is accomplished by inserting an $N - L$ by p matrix $\boldsymbol{T}$ between $\overline{\boldsymbol{U}}$ and $\boldsymbol{w}_a$ in the GSC structure [3] where $p < N - L$. If we define an N by p matrix $\boldsymbol{V} = \overline{\boldsymbol{U}}\boldsymbol{T}$, then the partially adaptive GSC is represented as

$$\boldsymbol{w} = \boldsymbol{w}_q - \boldsymbol{V}\boldsymbol{w}_a \tag{6}$$

where $\boldsymbol{w}_a$ is now a p dimensional adaptive weight vector. The solution for $\boldsymbol{w}_a$ is obtained by replacing $\overline{\boldsymbol{U}}$ with $\boldsymbol{V}$ in (5).

Partially adaptive beamforming is motivated by both computational complexity and performance considerations. The computational burden associated with adaptive algorithms can be a linear, quadratic, or even cubic function of the number of adaptive weights, depending on the algorithm. Note, for example, that the dimension of the matrix inverse in (5) is reduced from $N - L$ to p in the partially adaptive beamformer. There are two aspects of beamformer performance that change as a result of reducing the number of degrees of freedom. A partially adaptive beamformer is generally not capable of achieving the same level of steady state interference cancellation as a fully adaptive beamformer. However, the adaptive convergence characteristics of partially adaptive beamformers are superior. These properties are summarized here to motivate the modular structure.

If the signal and interference/noise are uncorrelated, then the steady state output power due to the interference and noise P_n can be expressed as

$$P_n = \boldsymbol{w}_q^h\boldsymbol{R}_n^{1/2}[\boldsymbol{I} - \boldsymbol{R}_n^{h/2}\boldsymbol{V}(\boldsymbol{V}^h\boldsymbol{R}_n\boldsymbol{V})^{-1}\boldsymbol{V}^h\boldsymbol{R}_n^{1/2}]\boldsymbol{R}_n^{h/2}\boldsymbol{w}_q \tag{7}$$

where $\boldsymbol{R}_n$ is the interference and noise covariance matrix. P_n is thus the norm of the projection of $\boldsymbol{R}_n^{h/2}\boldsymbol{w}_q$ onto the space orthogonal to the space spanned by the columns of $\boldsymbol{R}_n^{h/2}\boldsymbol{V}$. Increasing p, the number of columns in $\boldsymbol{V}$, generally results in reduced P_n.

Define the mean-squared error (MSE) e^2 as the average over time of the difference between the beamformer output when the interference/noise is absent and the beamformer output when the interference/noise is present. Assuming the sample covariance matrix is used to estimate $\boldsymbol{R}_x$, it is shown in [4] that the expected value of the MSE is given by

$$\mathcal{E}(e^2) = P_n(1 - p/M) + \sigma_s^2 p/M \tag{8}$$

where M is the number of data vectors used in the sample covariance matrix and σ_s^2 is the signal power. The first term in (8) is the MSE due to the presence of the interference and the second term is a MSE associated with cancellation of the signal. The signal MSE is only significant for values of M that are not much greater than p. Equations (7) and (8) point to a fundamental conflict in adaptive beamforming. Small p is desirable for rapid convergence,

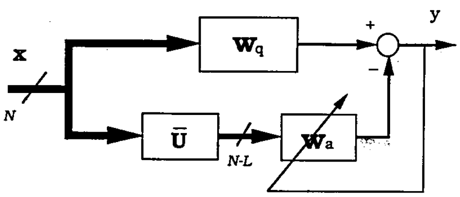

Fig. 1. The generalized sidelobe canceller.

but large p is often necessary to obtain maximum steady state performance potential (minimum P_n).

III. The Modular Decomposition

The discussion in the previous section indicates that it may be advantageous to simultaneously implement beamformers with differing numbers of adaptive weights or perhaps dynamically change the number of adaptive weights in a single beamformer. Both of these desires are feasible using the modular decomposition developed here. Also, as noted in the introduction, this structure distributes the computation of the adaptive weights over several lower order problems.

The optimal partially adaptive GSC weight vector $w = w_q - Vw_a$ is expressed using (5) as $w = (I - V\Pi)w_q$, where

$$\Pi = (V^h R_x V)^{-1} V^h R_x. \quad (9)$$

The matrix Π represents the solution to the optimization problem

$$\min_{\Pi} \mathcal{E}\{|(I - V\Pi)^h x|^2\} \quad (10)$$

or equivalently

$$\min_{\Pi} \operatorname{tr} (I - V\Pi)^h R_x (I - \Pi). \quad (11)$$

$z = (I - V\Pi)^h x$ can be viewed as the output of a bank of beamformers with weight vectors given by the columns of $(I - V\Pi)$; thus, Π minimizes the sum of the powers at the output of the beamformer bank, $\mathcal{E}(z^h z)$. Note that the GSC output is a linear combination of the elements of z: $y = w_q^h z$. A block diagram of this structure is depicted in Fig. 2. The output power is $\mathcal{E}\{|y|^2\} = w_q^h R_z w_q$ where $R_z = R_x - R_x V(V^h R_x V)^{-1} V^h R_x$.

Theorem 3.1: Let V be partitioned as $V = [V_1 \, V_2]$ and define $z_i = (I - V_i\Pi_i)^h z_{i-1}$, $1 \le i \le 2$, with $z_0 = x$. Now,

$$(I - V_1\Pi_1)(I - V_2\Pi_2) = I - V\Pi \quad (12)$$

where Π solves (10) and Π_i solves

$$\min_{\Pi_i} \mathcal{E}\{|(I - V_i\Pi_i)^h z_{i-1}|^2\}, \qquad 1 \le i \le 2. \quad (13)$$

Proof: The solution for Π_1 and Π_2 in (13) are $\Pi_1 = (V_1^h R_x V_1)^{-1} V_1^h R_x$ and $\Pi_2 = (V_2^h R_{z_1} V_2)^{-1} V_2^h R_{z_1}$ where $R_{z_1} = R_x - R_x V_1 (V_1^h R_x V_1)^{-1} V_1^h R_x$. Let

$$V^h R_x V = \begin{bmatrix} V_1^h R_x V_1 & V_1^h R_x V_2 \\ V_2^h R_x V_1 & V_2^h R_x V_2 \end{bmatrix} = \begin{bmatrix} A & D \\ C & B \end{bmatrix} \quad (14)$$

and define $\Delta = B - CA^{-1}D$. The left-hand side (LHS) of (12) is expressed as

$$\text{LHS} = I - V_1\Pi_1 - V_2\Pi_2 + V_1\Pi_1 V_2\Pi_2. \quad (15)$$

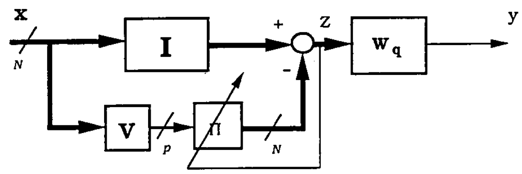

Fig. 2. Separation of adaptive components and w_q described in (9)–(11).

Substituting the solutions for Π_1 and Π_2 and utilizing the terms defined in (14) we obtain

$$\begin{aligned} \text{LHS} = I &- V_1(V_1^h R_x V_1)^{-1} V_1^h R_x \\ &- V_2(V_2^h R_x V_2 - V_2^h R_x V_1 (V_1^h R_x V_1)^{-1} V_1^h R_x V_2)^{-1} V_2^h R_x \\ &+ V_2(V_2^h R_x V_2 - V_2^h R_x V_1(V_1^h R_x V_1)^{-1} V_1^h R_x V_2)^{-1} \\ &\cdot V_2^h R_x V_1 (V_1^h R_x V_1)^{-1} V_1^h R_x \\ &+ V_1(V_1^h R_x V_1)^{-1} V_1^h R_x V_2 \Delta^{-1} V_2^h R_x \\ &- V_1(V_1^h R_x V_1)^{-1} V_1^h R_x V_2 \Delta^{-1} V_2^h R_x V_1 (V_1^h R_x V_1)^{-1} V_1^h R_x \\ = I &- V_1 A^{-1} V_1^h R_x - V_2 \Delta^{-1} V_2^h R_x \\ &+ V_2 \Delta^{-1} C A^{-1} V_1^h R_x + V_1 A^{-1} D \Delta^{-1} V_2^h R_x \\ &- V_1 A^{-1} D \Delta^{-1} C A^{-1} V_1^h R_x. \end{aligned} \quad (16)$$

Substituting (9) for Π on the right-hand side (RHS) of (12) and applying the formula for the inverse of block matrices gives

$$\begin{aligned} \text{RHS} &= I - [V_1 \, V_2] \begin{bmatrix} A^{-1} + A^{-1} D \Delta^{-1} C A^{-1} & -A^{-1} D \Delta^{-1} \\ -\Delta^{-1} C A^{-1} & \Delta^{-1} \end{bmatrix} \cdot \begin{bmatrix} V_1^h R_x \\ V_2^h R_x \end{bmatrix} \\ &= I - [V_1 A^{-1} + V_1 A^{-1} D \Delta^{-1} C A^{-1} - V_2 \Delta^{-1} C A^{-1} | \\ &\qquad - V_1 A^{-1} D \Delta^{-1} + V_2 \Delta^{-1}] \begin{bmatrix} V_1^h R_x \\ V_2^h R_x \end{bmatrix} \\ &= I - V_1 A^{-1} V_1^h R_x - V_2 \Delta^{-1} V_2^h R_x + V_2 \Delta^{-1} C A^{-1} V_1^h R_x \\ &\qquad + V_1 A^{-1} D \Delta^{-1} V_2^h R_x - V_1 A^{-1} D \Delta^{-1} C A^{-1} V_1^h R_x. \end{aligned} \quad (17)$$

Term by term comparison of (16) and (17) establishes (12). □

It is straightforward to see that if V is partitioned as $V = [V_1 \, V_2, \cdots, V_Q]$ and $z_i = (I - V_i\Pi_i)^h z_{i-1}$, $1 \le i \le Q$, with $z_0 = x$, that

$$(I - V_1\Pi_1)(I - V_2\Pi_2) \cdots (I - V_Q\Pi_Q) = I - V\Pi \quad (18)$$

where Π_i solves

$$\min_{\Pi_i} \mathcal{E}\{|(I - V_i\Pi_i)^h z_{i-1}|^2\}, \qquad 1 \le i \le Q. \quad (19)$$

That is,

$$\Pi_i = (V_i^h R_{z_{i-1}} V_i)^{-1} V_i^h R_{z_{i-1}}. \quad (20)$$

A block diagram illustrating the structure implied by (18)–(20) is depicted in Fig. 3. Note that if $R_{z_i} = R_{z_i}^{1/2} R_{z_i}^{h/2}$ we obtain

$$\mathcal{E}(z_i z_i^h) = R_{z_i} = R_{z_{i-1}}^{1/2} \{I - R_{z_{i-1}}^{h/2} V_i (V_i^h R_{z_{i-1}} V_i)^{-1} V_i^h R_{z_{i-1}}^{1/2}\} R_{z_{i-1}}^{h/2}. \quad (21)$$

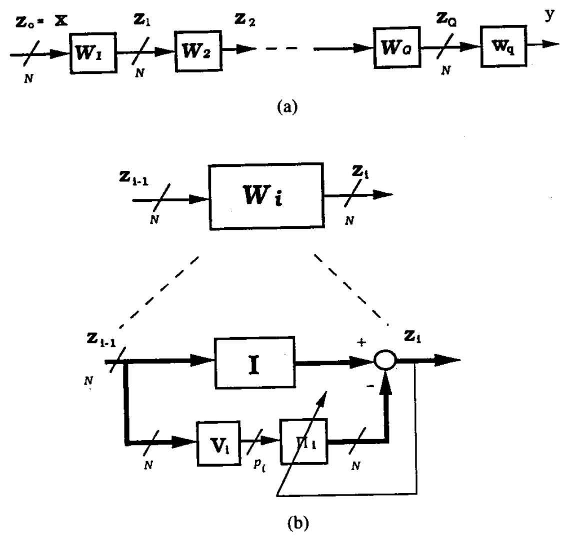

Fig. 3. The modular decomposition of an LCMV beamformer represented in (18)-(20). (a) Cascade structure, (b) individual modules in the cascade; $W_i' = I - V_i \Pi_i$.

The term in brackets is a projection matrix onto a space that is orthogonal to the columns of $R_{z_{i-1}}^{h/2} V_i$. Thus, the ith module removes the components of its input data which lie in the space spanned by $R_{z_{i-1}}^{h/2} V_i$.

IV. Discussion

The dimension of the matrix inverse required to compute the adaptive weights in the ith module is determined by the number of columns in V_i, that is, the number of degrees of freedom associated with each module. If each V_i has only one column, then no matrix inverses are necessary—the p dimensional matrix inverse required in the direct GSC decomposition is reduced to a sequence of p scalar inverses.

Standard adaptive algorithms such as LMS, RLS, and sample matrix inversion (SMI) are easily modified to compute the weights Π_i in each module. Systolic array implementations, e.g., [5], [6], are also easily modified to implement the individual modules. In general the relative computational burdens associated with adaptive algorithms for the modular decomposition structure is different than with the direct GSC implementation. For example, the LMS algorithm is

$$z_i(n) = z_{i-1}(n) - \Pi_i^h(n) u_i(n) \tag{22}$$

$$\Pi_i(n+1) = \Pi_i(n) + \mu u_i(n) z_i^h(n) \tag{23}$$

where $u_i(n) = V_i^h z_{i-1}(n)$, and the SMI algorithm based on M data vectors is

$$\Pi_i = R_i^{-1} r_i \tag{24}$$

$$z_i(n) = z_{i-1}(n) - \Pi_i^h u_i(n), \qquad n = 0, 1, 2, \cdots, M-1 \tag{25}$$

where $R_i = \Sigma_{n=0}^{M-1} u_i(n) u_i^h(n)$, $r_i = \Sigma_{n=0}^{M-1} u_i(n) z_{i-1}^h(n)$. Let the ith stage implement p_i degrees of freedom (V_i is $N \times p_i$). The SMI algorithm (24) and (25) requires on the order of $\Sigma_{i=1}^{Q} p_i^3 + p_i^2 (M + N) + 3NMp_i$ multiplications while the LMS algorithm in (22) and (23) requires about $\Sigma_{i=1}^{Q} 3NMp_i$ multiplications to obtain $z_Q(n)$, $n = 0, 1, 2, \cdots, M-1$. If p_i is sufficiently small relative to M or N, then both LMS and SMI require on the order of $\Sigma_{i=1}^{Q} 3NMp_i$ multiplications. In contrast, SMI requires substantially more computations than LMS in the direct GSC structure.

Note that the proof of equivalence between the direct and modular implementations is applicable when the covariance matrix is replaced by an estimated covariance matrix. This implies that the direct and modular beamformer outputs are identical for adaptive algorithms based on covariance matrix estimates such as RLS and SMI. Of course this implies that both implementations have identical convergence rates, independent of the module ordering. However, the direct and modular beamformer outputs and adaptive convergence rates generally differ if gradient based algorithms, e.g., LMS, are used.

The modular structure facilitates implementation of beamformers with differing numbers of adaptive degrees of freedom. The adaptive components in the ith module are independent of the jth module provided $j > i$. Thus, modules can be added or removed to dynamically change the number of adaptive weights without recomputing the weights in the original or remaining modules. This is analogous to the order recursive property of lattice filters. Alternatively, beam formers with different numbers of adaptive degrees of freedom are simultaneously implemented with a slight increase in computational load by applying the weight vector w_q to each module output. $y_i = w_q^h z_i$ is the output of an adaptive beamformer based on the $\nu_i = \Sigma_{j=1}^{i} p_j$ degrees of freedom represented in the blocking matrix $[V_1 V_2, \cdots, V_i]$. Thus, y_i based on small ν_i can be utilized when rapid convergence is necessary while y_i based on large ν_i can be utilized to obtain good steady state interference cancellation. As a final comment, we note that the weights w_q are decoupled from the adaptive weight determination process in the modular structure. This implies that one can simultaneously implement beamformers with different w_q without changing the adaptive weights. If $V^h w_q = 0$, then w_q represents the quiescent beamformer weight vector. Several different criteria for choosing components of w_q beyond those required by the constraints $U^h w_q = g$ are discussed in [7] and [8].

References

[1] K. M. Buckley, "Spatial/spectral filtering with linearly constrained minimum variance beamformers," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-35, pp. 249-266, Mar. 1987.

[2] L. J. Griffiths and C. W. Jim, "An alternative approach to linearly constrained adaptive arrays," *IEEE Trans. Antennas Propagat.*, vol. AP-30, pp. 27-34, Jan. 1982.

[3] B. D. Van Veen and R. A. Roberts, "Partially adaptive beamformer design via output power minimization," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-35, pp. 1524-1532, Nov. 1987.

[4] B. D. Van Veen, "Adaptive convergence of linearly constrained beamformers based on the sample covariance matrix," *IEEE Trans. Signal Processing*, vol. 39, no. 6, pp. 1470-1473, June 1991.

[5] B. Yang and J. F. Böhme, "On systolic implementation and the numerical properties of a multiple constrained adaptive beamformer," in *Proc. IEEE-ICASSP 89*, Apr. 1989, pp. 2819-2822.

[6] J. G. McWhirter and T. J. Shepherd, "Systolic array processor for MVDR beamforming," *Proc. Inst. Elec. Eng.*, vol. 136, pt. F, no. 2, Apr. 1989.

[7] L. J. Griffiths and K. M. Buckley, "Quiescent pattern control in linearly constrained adaptive arrays," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-35, pp. 917-926, July 1987.

[8] B. D. Van Veen, "Optimization of quiescent response in partially adaptive beamformers," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-38, pp. 471-477, Mar. 1990.

Efficient Parallel Rooting of Complex Polynomials on the Unit Circle

William S. McCormick and James L. Lansford

***Abstract*—An efficient method is presented for rooting complex polynomial equations of any order where the root space is restricted to the unit circle. The method restates the evaluation of polynomials as a recursive algorithm involving only additions. By means of the bilinear transformation, the straight line, uniformly spaced, recursive polynomial evaluation method of Nuttall is extended to the unit circle where the root positions are determined by thresholding. General coefficient transformations are provided along with a comparison to the Horner method.**

I. Introduction

A current need exists [1] for the real-time, high resolution estimates of the center frequencies of multiple incident radar pulses with pulsewidths as narrow as 200 ns. Recently, high resolution [2] parametric methods like the covariance method have been considered for this application. For an Mth order covariance model, the numerical estimation of frequency requires the solution of the Mth order covariance normal equations followed by the rooting of an Mth order polynomial equation.

This correspondence addresses the polynomial equation rooting problem of the covariance algorithm. In general, closed form expressions for the roots of a polynomial equation [3] are computationally intensive, numerically sensitive, and do not exist for orders greater than four. Alternative methods like the iterative techniques are numerically unstable and have varying convergence times [4].

An important and useful property of the covariance method is that the roots (frequencies) of its polynomial equation [2] lie on the unit circle for sufficiently high S/N ratios. In this correspondence, an efficient (few multiplications) approach is presented that evaluates the complex polynomial at prescribed locations on the unit circle and identifies roots by thresholding the modulus of the evaluated polynomial. For a general complex z, the most efficient direct method for polynomial evaluation $f(z)$ is the Horner [5] method which requires M multiplications for each evaluation of a Mth order polynomial or NM multiplications for evaluation at N points on the unit circle. In contrast, by restricting the allowable root space to the unit circle, the method presented in this correspondence restates the polynomial evaluation as the solution of the Mth order recursive relationship involving only additions. The method represents an extension to the unit circle of the recursive method of Nuttall [6], Bose [9], and Nie and Unbehauen [8]. By employing the bilinear transformation, the straight line root space of Nuttall is mapped to the unit circle and a recursive relationship is derived that relates to the unit circle. General relationships between transformed coefficients are included in the correspondence along with an illustrative example and a computational comparison to the Horner method. Scaling and roundoff error propagation are also discussed.

Manuscript received January 25, 1990; revised September 17, 1990. This work was supported by the USAF Aeronautical Laboratories under Contract F33615-86-C-1040.

W. S. McCormick is with Wright State University, Dayton, OH 45435 and Georgia Tech Research Institute, Atlanta, GA 30332.

J. L. Lansford is with Georgia Tech Research Institute, Atlanta, GA 30332.

IEEE Log Number 9101863.

II. Discussion of Algorithm

Given an Mth order polynomial, $f(z) = \Sigma_{i=0}^{M} a_i z_i$, with complex coefficients, a_i, the bilinear transformation will map the unit circle in the z plane on the $j\omega$ imaginary axis of the s plane to give the transformed polynomial

$$F(\omega) = \sum_{i=0}^{M} b_m \omega^m \tag{1}$$

where ω is real and b_m is complex. The expression $F(\omega)$ can be evaluated at uniform points on a straight line (i.e., imaginary axis) in the complex s plane, $\omega_n = \omega_0 + n\Delta\omega$, as [6]

$$Q_M(n) = F(\omega_0 + n\Delta\omega) = \sum_{i=0}^{M} \beta_i n_i \tag{2}$$

where $\{\beta_i\}$ is a function of ω_0, $\Delta\omega$, and $\{b_m\}$. For the important case where $\omega_0 = 0$, β_i equals $b_i \Delta\omega^i$.

From the theory of finite differences [7]–[9], the Mth order backward finite difference, $\nabla^M Q_M(n)$, equals $M!\beta_M$ which leads to the following recursive relationship for $Q_M(n)$:

$$Q_{M-p}(n) = Q_{M-p}(n-1) + Q_{M-p-1}(n)$$
$$p = 0, 1, \cdots, M-2$$

and

$$Q_1(n) = Q_1(n-1) + M!\beta_M. \tag{3}$$

The initial conditions, $Q_{M-p}(0)$, can be shown [6] to equal

$$Q_M(0) = \beta_0$$
$$Q_{M-p}(0) = \beta_0 + \sum_{m=0}^{M} \beta_m \left\{ \sum_{l=1}^{p} (-1)^{m+l} (l)^m \binom{p}{l} \right\}$$
$$p = 1, 2, \cdots, M-1. \tag{4}$$

Given the initial conditions, $Q_M(n)$ can be evaluated over $1 \le n \le N$ with MN additions and no multiplications.

III. Bilinear Transformation

The familiar bilinear transformation [10]

$$z = \frac{1 + s(T/2)}{1 - s(T/2)} \tag{5}$$

maps the imaginary axis $j\omega$ in the s plane to the unit circle $z = \epsilon^{-j\omega' T}$ in the z plane. In order to use the recursive relationship of Section II, the $\Delta\omega$ increments on the imaginary axis of the s plane must be fixed at a constant value which leads to a frequency/angle compression or warping on the unit circle as given by

$$\omega' = \frac{2}{T} \tan^{-1}\left(\frac{\omega T}{2}\right). \tag{6}$$

The resulting nonuniform normalized angle increment, $\Delta\omega'/\Delta\omega$, $(T = 1)$, on the unit circle can be expressed as

$$\frac{\Delta\omega'}{\Delta\omega} \approx \frac{d\omega'}{d\omega} = \frac{4}{4 + (\omega)^2} = \frac{1}{1 + \tan^2(\omega'/2)} \tag{7}$$

which indicates that $0.5\Delta\omega \le (\Delta\omega') \le \Delta\omega$ whenever $0 \le \omega' \le \pi/2$ rad. In other words, the recursive relationship based on the bilinear transformation should limit the range on ω' to 90° in order to avoid compressions of a level $\Delta\omega'/\Delta\omega \le 0.5$; the 90° limit on

1053-587X/91/1000-2347$01.00 © 1991 IEEE