

Insertion of Prior Knowledge into the Multistage Wiener Filter

John D. Hiemstra
SAIC
Chantilly, VA
Email: HiemstraJ@saic.com

J. Scott Goldstein
SAIC
Chantilly, VA
Email: GoldsteinJa@saic.com

Abstract—The multistage Wiener filter (MWF) is a reduced rank adaptive signal processing algorithm applicable to airborne radar space-time adaptive processing (STAP). This paper discusses techniques for augmenting the MWF in order to exploit pre-existing environmental knowledge. We consider *a priori* knowledge such as known fixed interferers as well as the information contained in previously computed adaptive solutions. Two approaches are presented. The first approach is based on existing linear constraint techniques. The second approach is based on non-zero initialization of the CG-MWF implementation of the MWF. We evaluate the performance of both algorithms under ideal conditions, such as perfect knowledge and stationary data, as well as perturbed conditions.

I. INTRODUCTION

The multistage Wiener filter (MWF) is described in [1] in its most general form as an unconstrained filter. This form serves as the basic filtering engine across a whole spectrum of adaptive signal processing applications. In this paper we describe methods for augmenting the MWF in order to take advantage of pre-existing environmental knowledge. We consider two different classes of prior knowledge. In the first case we consider interference sources that are known *a priori* such as might be the case for known fixed transmitters or clutter reflections predicted from digital terrain data bases. We refer to this class of problem as “Knowledge-Aided MWF”. In the second case we take advantage of the information contained in previously computed adaptive solutions to constrain or initialize our new solutions. We refer to this case as “Recursive MWF”.

There are several ways to insert prior knowledge into the MWF. The first approach that we describe uses linear constraints in the context of the generalized sidelobe canceller. We use both directional null constraints and quiescent pattern constraints. We also describe a newer method for inserting knowledge into the MWF. It has recently been shown that the MWF can be implemented via the method of conjugate gradients (CG) [2], [3]. In this implementation, referred to as CG-MWF, the filter is initialized with a starting weight vector. This initial weight vector can be the zero vector if no prior knowledge is available, or it can be nonzero to reflect *a priori* knowledge. We exploit this structure to implement Knowledge-Aided MWF and Recursive MWF. This paper describes and evaluates these algorithms.

II. APPROACH

A. Linearly Constrained MWF

1) *LC-MWF Background*: In this section we review the linear constrained minimum variance (LCMV) beamformer¹ and show how we can similarly apply multiple linear constraints to the multistage Wiener filter. The minimum variance distortionless response (MVDR) beamformer solves the following constrained minimization

$$\mathbf{w}^{(mvdr)} = \arg \min_{\mathbf{w}^H \mathbf{s} = 1} \mathbf{E} [|\mathbf{w}^H \mathbf{x}_{i,n}|^2], \quad (1)$$

where $\mathbf{x}_{i,n}$ are interference-plus-noise training snapshots (i.e., target-free training), \mathbf{s} is the unit-norm steering vector, and there is a single linear constraint $\mathbf{w}^H \mathbf{s} = 1$. The well-known solution to (1) is

$$\mathbf{w}^{(mvdr)} = \frac{\mathbf{R}_x^{-1} \mathbf{s}}{\mathbf{s}^H \mathbf{R}_x^{-1} \mathbf{s}}, \quad (2)$$

where $\mathbf{R}_x = \mathbf{E} [\mathbf{x}_{i,n} \mathbf{x}_{i,n}^H]$ is the covariance matrix.

The MVDR beamformer is a special case of the more general LCMV beamformer which is defined by the following minimization

$$\mathbf{w}^{(lcmv)} = \arg \min_{\mathbf{w}^H \mathbf{C} = \mathbf{g}^H} \mathbf{E} [|\mathbf{w}^H \mathbf{x}_{i,n}|^2], \quad (3)$$

where \mathbf{C} is an $N \times N_c$ constraint matrix that contains N_c constraint vectors, and \mathbf{g} is an $N_c \times 1$ vector that specifies the values of these constraints. The solution to (3) is

$$\mathbf{w}^{(lcmv)} = \mathbf{R}_x^{-1} \mathbf{C} (\mathbf{C}^H \mathbf{R}_x^{-1} \mathbf{C})^{-1} \mathbf{g}. \quad (4)$$

Equation (4) is the direct form processor solution. The LCMV beamformer can also be implemented as a generalized sidelobe canceller (GSC) [5]. The GSC, shown in Figure 1, divides the N -dimensional solution space into a constraint subspace and a subspace that is orthogonal to the constraint subspace. The constraint subspace is defined by the columns of \mathbf{C} . The subspace orthogonal to the constraint subspace is defined by the columns of the blocking matrix \mathbf{B} where

$$\mathbf{B} = \text{null}(\mathbf{C}^H), \quad (5)$$

i.e.,

$$\mathbf{C}^H \mathbf{B} = \mathbf{0}. \quad (6)$$

¹The LCMV beamformer is well known. Our discussion follows [4].

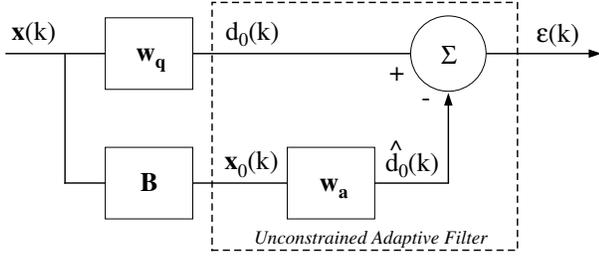


Fig. 1. The Generalized Sidelobe Canceller Implementation of the LCMV Beamformer.

The projection of (4) onto the constraint subspace is given by the quiescent weight vector

$$\mathbf{w}_q = \mathbf{C} (\mathbf{C}^H \mathbf{C})^{-1} \mathbf{g}. \quad (7)$$

Many different types of linear constraints have been developed for the LCMV beamformer. We consider directional constraints and quiescent pattern constraints. The distortionless constraint of the MVDR beamformer, $\mathbf{w}^H \mathbf{s} = 1$, is one example of a directional constraint. We are interested in directional null constraints that enforce beampattern nulls in the direction of known interference sources

$$\mathbf{w}^H \mathbf{s}_j = 0, \quad (8)$$

where \mathbf{s}_j is the steering vector in the direction of the j^{th} interferer. Combining the distortionless constraint with N_j null constraints yields the following \mathbf{C} matrix and \mathbf{g} vector

$$\mathbf{C} = [\mathbf{s} \quad \mathbf{s}_1 \quad \mathbf{s}_2 \quad \cdots \quad \mathbf{s}_{N_j}], \quad (9)$$

$$\mathbf{g}^H = [1 \quad 0 \quad 0 \quad \cdots \quad 0]. \quad (10)$$

We apply directional constraints to Knowledge-Aided LC-MWF.

We are also interested in “quiescent pattern” constraints. There are several different approaches for quiescent pattern constraints. We consider the approach in [6] which utilizes two linear constraints. The first constraint imposes the quiescent response

$$\mathbf{w}^H \bar{\mathbf{w}}_{dq} = 1, \quad (11)$$

where

$$\bar{\mathbf{w}}_{dq} = \frac{\mathbf{w}_{dq}}{\mathbf{w}_{dq}^H \mathbf{w}_{dq}}, \quad (12)$$

and where \mathbf{w}_{dq} is the desired quiescent pattern. The second constraint protects the desired signal from being cancelled by the adaptive filter²

$$\mathbf{w}^H \mathbf{c}_s = 0, \quad (13)$$

where

$$\mathbf{c}_s = \mathbf{P}_{\bar{\mathbf{w}}_{dq}}^\perp \mathbf{s}, \quad (14)$$

²The constraint in (13) protects a point. A broader region in angle/frequency space can be protected using additional constraints as described in [6].

where $\mathbf{P}_{\bar{\mathbf{w}}_{dq}}^\perp$ is the projection matrix orthogonal to $\mathbf{P}_{\bar{\mathbf{w}}_{dq}}$ and where $\mathbf{P}_{\bar{\mathbf{w}}_{dq}}$ is the projection matrix with respect to $\bar{\mathbf{w}}_{dq}$. Thus

$$\mathbf{P}_{\bar{\mathbf{w}}_{dq}}^\perp = \mathbf{I} - \left(\bar{\mathbf{w}}_{dq} (\bar{\mathbf{w}}_{dq}^H \bar{\mathbf{w}}_{dq})^{-1} \bar{\mathbf{w}}_{dq}^H \right). \quad (15)$$

The resulting \mathbf{C} matrix and \mathbf{g} vector are given by

$$\mathbf{C} = [\bar{\mathbf{w}}_{dq} \quad \mathbf{c}_s], \quad (16)$$

$$\mathbf{g}^H = [1 \quad 0]. \quad (17)$$

Note that the upper path of the GSC still contains \mathbf{w}_q computed via (7) based on \mathbf{C} and \mathbf{g} above in (16) and (17) and should not be confused with \mathbf{w}_{dq} or $\bar{\mathbf{w}}_{dq}$. Similarly the blocking matrix is still found from (5). We apply quiescent pattern constraints to Recursive LC-MWF.

The development above was specifically based on the LCMV beamformer defined by the minimization of (3). This is a full rank beamformer — N_c constraints plus $N - N_c$ adaptive degrees-of-freedom. All of the analysis, however, carries over directly to the reduced rank MWF via the GSC architecture. That is, through inspection of Figure 1 we see that we can enforce the linear constraints in the same manner through \mathbf{w}_q and \mathbf{B} and use the MWF for the solution to the unconstrained filter, \mathbf{w}_a , that lies in the subspace orthogonal to the constraint subspace. We refer to the MWF with multiple linear constraints as LC-MWF.

2) *Knowledge-Aided LC-MWF*: When the locations of interference sources are known *a priori* we can apply linear null constraints to the multistage Wiener filter as described above. We refer to this approach as Knowledge-Aided LC-MWF.

3) *Recursive LC-MWF*: We next consider the use of quiescent pattern constraints for Recursive LC-MWF. We process an initial block of K_0 data snapshots (e.g., $K_0=1N$) at time τ_0 using the standard MWF algorithm with only the distortionless constraint. Then we receive another block of K_1 data snapshots (e.g., $K_1=1N$) at time τ_1 . We run the LC-MWF algorithm using the best weight vector computed at time τ_0 as our desired quiescent pattern, i.e., \mathbf{w}_{dq} . We refer to this as the “first recursion” since it is the first time that weight vector feedback was used. We repeat this process as many times as desired using the relation

$$\mathbf{w}_{dq}(\tau_{i+1}) = \mathbf{w}^{(r_{best})}(\tau_i). \quad (18)$$

We refer to the LC-MWF algorithm using (18) as Recursive LC-MWF.

B. The CG-MWF Algorithm

The relationship between the MWF and CG weight vectors was described in [2]. We refer to implementations of the MWF using conjugate gradient techniques as CG-MWF. Here we implement the MWF algorithm using the CG recursions in order to take advantage of a desirable property of CG, namely, the structure allowing for a non-zero initialization of the weight vector. Note that there are many different implementations of the CG algorithm. In connection with (nonsymmetric) least squares problems ($\mathbf{A}\mathbf{x} = \mathbf{b}$) the CG method is applied to the normal equations $\mathbf{A}^H \mathbf{A}\mathbf{x} = \mathbf{A}^H \mathbf{b}$.

One popular and stable CG implementation for the normal equations is conjugate gradient least squares (CGLS) [7]. The Wiener-Hopf equation takes the form of the normal equations for the case of estimated statistics. That is, we solve

$$\hat{\mathbf{R}}_{\mathbf{x}_0} \mathbf{w}_a = \hat{\mathbf{r}}_{\mathbf{x}d}, \quad (19)$$

where

$$\hat{\mathbf{R}}_{\mathbf{x}_0} = \frac{1}{K} \mathbf{X}_0 \mathbf{X}_0^H, \quad (20)$$

and

$$\hat{\mathbf{r}}_{\mathbf{x}d} = \frac{1}{K} \mathbf{X}_0 \mathbf{d}_0^H. \quad (21)$$

Substituting (20) and (21) into (19) yields

$$\mathbf{X}_0 \mathbf{X}_0^H \mathbf{w}_a = \mathbf{X}_0 \mathbf{d}_0^H. \quad (22)$$

We consider CGLS applied to (22) in this paper. The CGLS recursion equations are presented in Table I [7].

TABLE I
CGLS RECURSION EQUATIONS.

| \mathbf{w}_a Computed Via |
|--|
| $\alpha_i = \frac{\ \mathbf{X}_0 \mathbf{t}^{(i-1)}\ ^2}{\ \mathbf{X}_0^H \mathbf{a}^{(i-1)}\ ^2}$ |
| $\mathbf{w}_a^{(i)} = \mathbf{w}_a^{(i-1)} + \alpha_i \mathbf{a}^{(i-1)}$ |
| $\mathbf{t}^{(i)} = \mathbf{t}^{(i-1)} - \alpha_i \mathbf{X}_0^H \mathbf{a}^{(i-1)}$ |
| $\beta_i = \frac{\ \mathbf{X}_0 \mathbf{t}^{(i)}\ ^2}{\ \mathbf{X}_0 \mathbf{t}^{(i-1)}\ ^2}$ |
| $\mathbf{a}^{(i)} = \mathbf{X}_0 \mathbf{t}^{(i)} + \beta_i \mathbf{a}^{(i-1)}$ |
| Initialized With |
| Starting vector $\mathbf{w}_a^{(0)}$ |
| $\mathbf{t}^{(0)} = \mathbf{d}_0^H - \mathbf{X}_0^H \mathbf{w}_a^{(0)}$ |
| $\mathbf{a}^{(0)} = \mathbf{X}_0 \mathbf{t}^{(0)}$ |

1) *Knowledge-Aided CG-MWF*: We now consider the case where *a priori* knowledge of the interference environment is available to the filter. We insert this knowledge using the following relation

$$\mathbf{w}_a^{(0)} = \mathbf{w}_a^{(dq)}, \quad (23)$$

where $\mathbf{w}_a^{(dq)}$ is the desired quiescent pattern that reflects all of the *a priori* knowledge. We refer to CG-MWF initialized with (23) as Knowledge-Aided CG-MWF.

2) *Recursive CG-MWF*: We also consider the case where data observations become available with time. We process an initial block of K_0 data snapshots (e.g., $K_0=1N$) at time τ_0 using the CG-MWF algorithm with $\mathbf{w}_a^{(0)}(\tau_0) = \mathbf{0}$. Then we receive another block of K_1 data snapshots (e.g., $K_1=1N$) at time τ_1 . We run the CG-MWF algorithm again, this time using $\mathbf{w}_a^{(0)}(\tau_1) = \mathbf{w}_a^{(r_{best})}(\tau_0)$, where $\mathbf{w}_a^{(r_{best})}(\tau_0)$ is the best weight vector computed at time τ_0 , i.e., at the best rank. We refer to this second run of the CG-MWF algorithm as the “first

recursion” since it is the first time that weight vector feedback was used. We continue adding additional recursions as desired using the relation

$$\mathbf{w}_a^{(0)}(\tau_{i+1}) = \mathbf{w}_a^{(r_{best})}(\tau_i). \quad (24)$$

We refer to the CG-MWF algorithm using (24) as Recursive CG-MWF.

III. SIMULATION RESULTS

A. Scenario Description

We evaluate the algorithms in this paper for spatial array processing. We consider an adaptive array having 32 elements with half-wavelength spacing, steered to broadside. There are 25 noise jammers at randomly selected locations and power levels, as shown in Figure 2. We consider $K=1N=32$ training samples of target-free data and 1000 Monte Carlo trials.

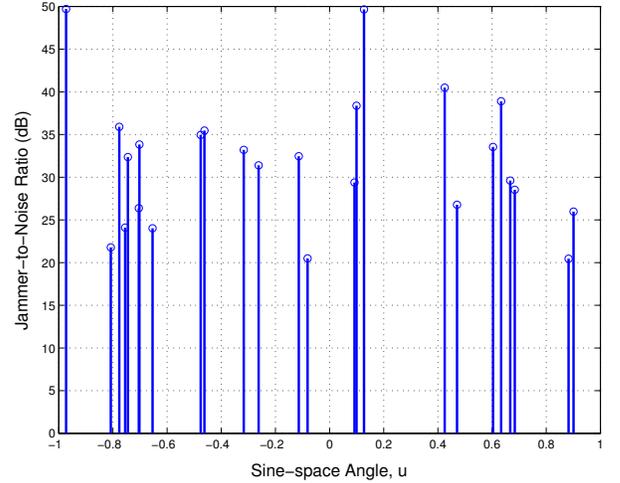


Fig. 2. Jammer Locations and Power Levels.

B. LC-MWF

We first evaluate Knowledge-Aided LC-MWF. We assume that 10 of the 25 jammers come from known directions and place null constraints in these 10 directions. Figure 3 shows the resulting performance under the assumption of perfect knowledge of the jammer directions. We see that the MWF is now able to reach its best performance at rank 15 as opposed to 25 in the baseline case. Thus we have productively partitioned the solution subspace into quiescent and adaptive subspaces due to the high (i.e., perfect) quality of our *a priori* knowledge. We also see in Figure 3 that the mean square error (MSE) for the LC-MWF solution is 2.9 dB better than for the baseline case. This is because the LC-MWF processor has perfect knowledge of 10 of the jammer directions while the baseline MWF processor has to rely on statistical estimates.

Next we examine the more practical case where the *a priori* knowledge is imperfectly known. We perform a sensitivity study and look for the breakpoint in required accuracy. Figure 4 shows the performance for five different error levels in

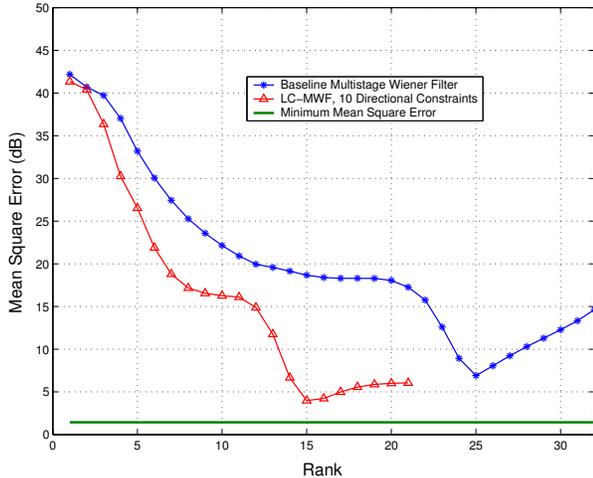


Fig. 3. Knowledge-Aided Linear Constrained MWF, Perfect Knowledge.

the assumed jammer directions, ranging from $1/2$ to $1/100^{th}$ of a half-power beamwidth³. We see that performance is quite sensitive to these errors. Even one-sigma errors of only $1/50^{th}$ to $1/100^{th}$ of a half-power beamwidth provide noticeable performance degradation in both the best MSE and the required number of adaptive stages. By the time the one-sigma errors have reached $1/20^{th}$ of a half-power beamwidth the best MSE has deteriorated by 2 dB relative to the baseline and thus Knowledge-Aided LC-MWF is no longer productive.

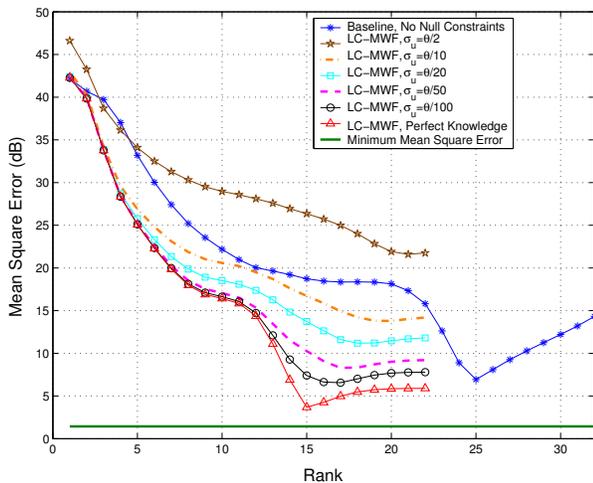


Fig. 4. Sensitivity of Knowledge-Aided LC-MWF to Jammer Direction Errors.

We next evaluate Recursive LC-MWF. Figure 5 shows the results for the baseline, 1st recursion, 2nd recursion, and 10th recursion for stationary data. We see that for all the recursive cases the filter indeed leverages the performance of the previous weight vector and immediately provides excellent

³We refer to the half-power beamwidth of a 32 element array with uniform weighting.

performance. Note that for the 1st recursion the best rank is not literally at a rank of 1. This is consistent with the fact that $\mathbf{w}^{(r_{best})}(\tau_0) \neq \mathbf{w}^{(opt)}$ but rather is the best approximation supported by the specific training samples provided. However, as the number of recursions increases the best rank tends toward 1 and the MSE performance level gradually approaches that of the optimal MMSE performance.

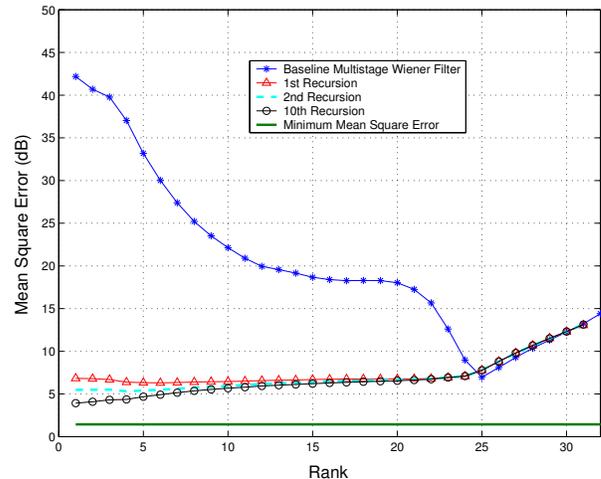


Fig. 5. Recursive MWF using a Quiescent Pattern Constraint, Stationary Data.

We next examine the sensitivity of Recursive LC-MWF to the stationarity of the environment. Clearly if the interference environment changes from one recursion to the next performance will degrade. We examine the gracefulfulness of this degradation. We consider cases where jammers are either deleted or added to the baseline interference environment between times τ_0 and τ_1 . For simplicity we examine performance with just a single recursion.

First we examine performance for the case of deleted jammers. At time τ_0 we implement the standard MWF using observations containing 25 jamming signals at the same locations and power levels as shown in Figure 2. At time τ_1 we execute Recursive LC-MWF constrained by $\mathbf{w}^{(r_{best})}(\tau_0)$ but using observations that reflect the deletion of 0, 1, 2, or 3 jammers. The deletion of jammers has virtually no impact on the performance. Excellent performance is achieved in the first recursion for each of these cases (not shown). This is not a surprising result since the presence of extraneous nulls in the desired quiescent pattern is not disruptive.

In Figure 6 we examine performance for the case of additional jammers. At time τ_0 there are 25 jammers at the same locations and power levels as shown in Figure 2. At time τ_1 we process data that reflects the addition of 0, 1, 2, or 3 jammers. The additional jammers have the same characteristics as the original 25 jammers. That is, they are randomly determined with the restrictions that they lie outside the main beam and have jammer-to-noise ratios between 20 and 50 dB. We see in Figure 6 that the addition of jammers has a negative impact on the performance, as one would expect

since the new jammers are not nulled in the constraint subspace. Additionally, however, we see that the filter requires a full 25-27 adaptive stages in order to recover best performance. Thus an entirely new adaptive subspace solution is needed even though the interference environment has changed only by 1 to 3 jammers. This is an unfortunate property. However, the amount of adaptive processing required is no more than if the standard MWF had been used. Thus in abruptly nonstationary environments we are neither better off nor worse off for having used Recursive LC-MWF.

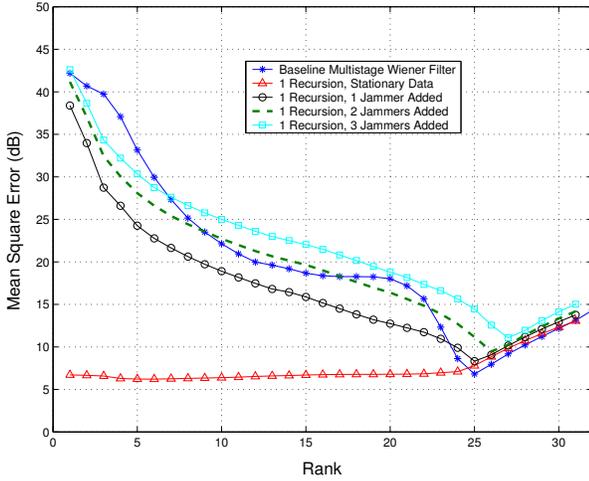


Fig. 6. Sensitivity of Recursive LC-MWF to Nonstationary Data – Certain Jammers Absent at Time τ_0 but Turned On at Time τ_1

C. CG-MWF

We evaluate CG-MWF in a similar manner to LC-MWF. We first examine Knowledge-Aided CG-MWF and consider the case of perfect knowledge of all 25 interference sources (directions and power levels). This is an unrealistic assumption, of course, but represents a bounding case. Figure 7 shows the results. We see that the filter is able to achieve the minimum MSE immediately at rank 1 (no adaptive stages). As adaptive stages are added the performance degrades due to overadaptation.

Next we examine the sensitivity of this approach to imperfections in our knowledge of the interference environment. We search for the breakpoint that determines how accurately *a priori* knowledge must be known in order for it to be productive. We consider two types of imperfections, errors in interference directions and errors in interference power levels. Figure 8 shows the performance as a function of errors in our knowledge of the interferer directions. We consider one-sigma errors of 1/10, 1/20, 1/50, and 1/100 of a half-power beamwidth. We see that the breakpoint appears to be at a one-sigma level of about 1/20th of a half-power beamwidth. When errors are small relative to this level, quiescent pattern initialization is quite productive. The mean square error (MSE) achieved is superior to that achieved in the baseline case and fewer adaptive stages are required. Conversely, when errors

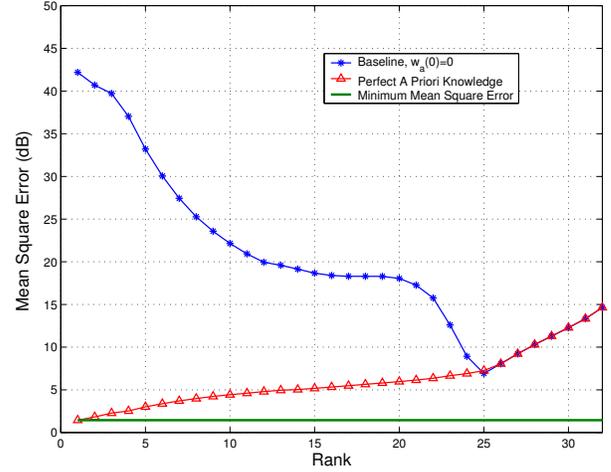


Fig. 7. Performance of Knowledge-Aided CG-MWF with Perfect Knowledge.

are large relative to this level then the benefit of initialization is diminished. We do note, however, that $w_a^{(q)}$ initialization is not counterproductive. Even with large errors the number of required adaptive stages does not exceed that for $w_a^{(0)} = \mathbf{0}$, and the MSE at best rank is the same.

Figure 9 presents the results when there are errors in the interference power levels (e.g., as might be caused by multipath fading). Here we see that the breakpoint appears at a one-sigma error level of between 10 and 15 dB. Below this level the performance advantage is significant. Above this level the utility is diminished. Again we note that the performance with poor *a priori* knowledge in terms of best MSE and required number of stages does not deteriorate relative to the baseline case.

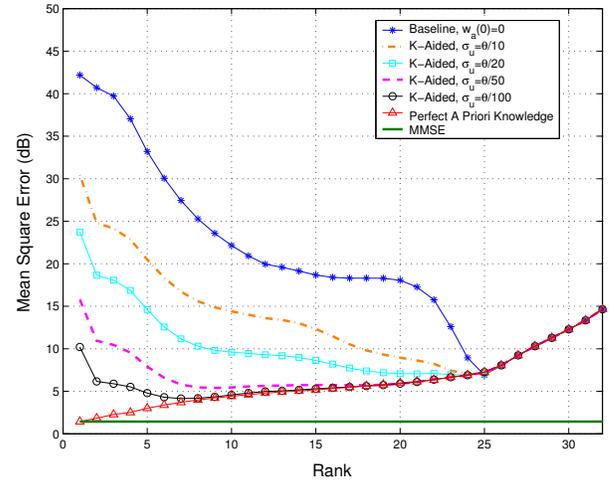


Fig. 8. Sensitivity of Knowledge-Aided CG-MWF to Jammer Direction Errors.

We next evaluate Recursive CG-MWF. Figure 10 shows the results for the baseline, 1st recursion, 2nd recursion, and 10th recursion for stationary data. We see that for all the

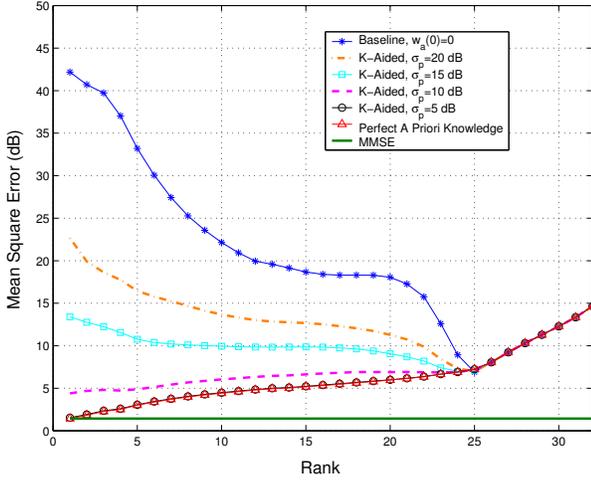


Fig. 9. Sensitivity of Knowledge-Aided CG-MWF to Jammer Power Errors.

recursive cases the filter indeed leverages the performance of the previous weight vector and provides excellent performance with no (or few) adaptive stages required. As the number of recursions increases the best rank tends toward 1 and the MSE performance level gradually approaches that of the optimal MMSE performance.

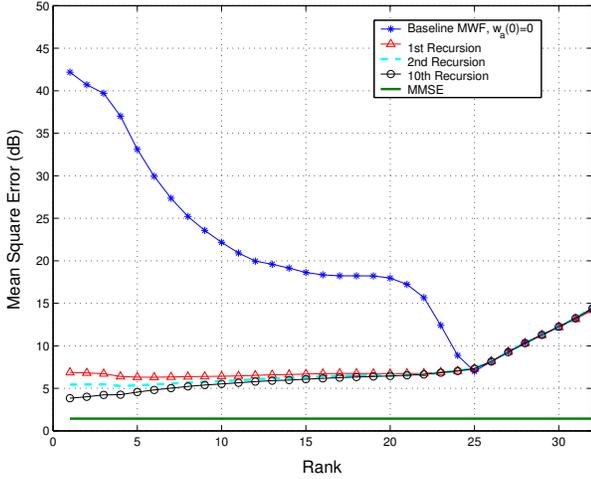


Fig. 10. Performance of Recursive MWF versus the Number of Recursions, Stationary Data.

We next examine the sensitivity of Recursive CG-MWF to the stationarity of the environment. Clearly if the interference environment changes from one recursion to the next performance will degrade. We examine the gracefulness of this degradation. We consider cases where jammers are either deleted or added to the baseline interference environment between times τ_0 and τ_1 . For simplicity we examine performance with just a single recursion.

First we examine performance for the case of deleted jammers. At time τ_0 there are 25 noise jammers at the same locations and power levels as shown in Figure 2. At time

τ_1 we process data that reflects the deletion of 0, 1, 2, or 3 jammers. The initialization at time τ_1 is again with $\mathbf{w}_a^{(0)}(\tau_1) = \mathbf{w}_a^{(r_{best})}(\tau_0)$. The deletion of jammers has virtually no impact on the performance. Excellent performance is achieved with the recursion (not shown). This is not a surprising result since the presence of extraneous nulls in $\mathbf{w}_a^{(0)}(\tau_1)$ is not disruptive.

In Figure 11, however, we examine performance for the case of additional jammers. At time τ_0 there are 25 noise jammers at the same locations and power levels as shown in Figure 2. At time τ_1 we process data that reflects the addition of 0, 1, 2, or 3 jammers. The additional jammers have the same characteristics as the original 25 jammers. That is, they are randomly determined with the restrictions that they lie outside the main beam and have jammer-to-noise ratios between 20 and 50 dB. The initialization at time τ_1 is again with $\mathbf{w}_a^{(0)}(\tau_1) = \mathbf{w}_a^{(r_{best})}(\tau_0)$. We see in Figure 11 that the addition of jammers can have mixed results. Typically, as one would expect, that the performance suffers appreciably due to the introduction of an undernullled interference sources. We see that this is the case with either 2 or 3 new jammers. However, we also see that the case of a single additional jammer had very little negative impact. This is due to a fortuitous random draw of the jammer location/power. We also note that when performance suffers the filter then takes a full 25-28 stages to converge again to its best performance. Thus an entirely new subspace is needed to achieve best performance even though only 2 to 3 new jammers are present. This is no more burdensome, however, then if trivial initialization had been used.

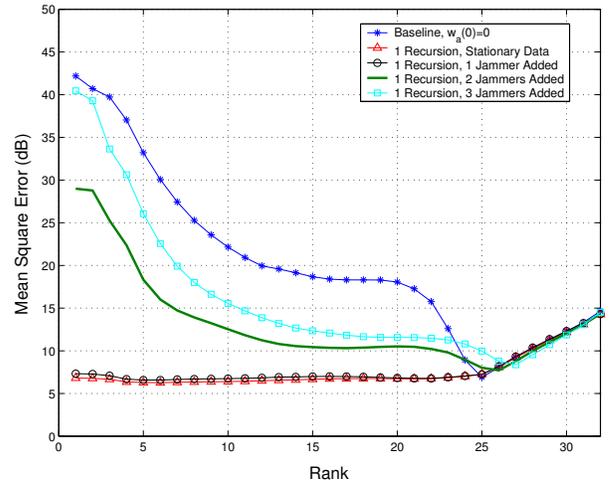


Fig. 11. Sensitivity of Recursive CG-MWF to Nonstationary Data: Certain Jammers Absent at Time τ_0 but Turned On at Time τ_1 .

IV. COMBINING LINEAR CONSTRAINTS WITH CG-MWF

In this paper we describe two different means of augmenting the MWF to take advantage of pre-existing environmental knowledge — LC-MWF and CG-MWF. These two approaches can be combined, of course, to simultaneously implement both Knowledge-Aided and Recursive MWF. For example, *a priori* knowledge can be inserted using linear constraints and recursion can be implemented using CG-MWF. We refer to this as LC-CG-MWF and demonstrate the results in Figure 12 for the baseline scenario under consideration with 10 known interferers. We see that the 10 linear constraints reduce the adaption requirements from 25 to 15 as before. Additionally, upon recursion the performance quickly converges to best MSE. The results in Figure 12 are for perfect knowledge and stationary data. Performance degradations due to imperfections and nonstationarities are not shown but can be inferred from the previous results.

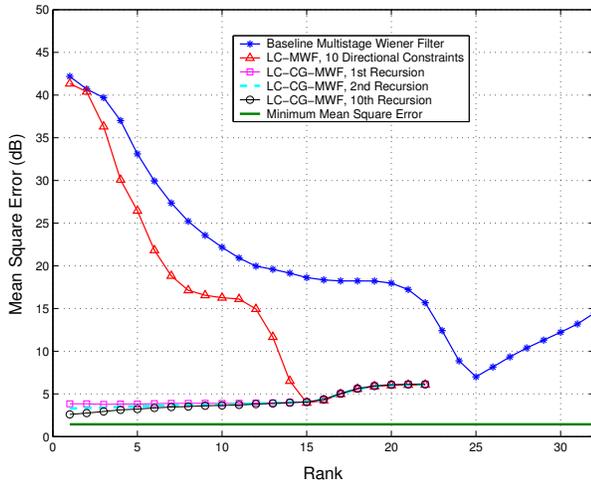


Fig. 12. Combined Knowledge-Aided and Recursive LC-CG-MWF: Perfect Knowledge and Stationary Data.

V. SUMMARY

In this paper we described two methods for augmenting the multistage Wiener filter in order to exploit *a priori* knowledge or take advantage of previously computed adaptive solutions. The first approach was based on existing linear constraint techniques. The second approach used the recently reported CG-MWF implementation of the MWF which affords the opportunity for nonzero weight vector initialization. This nonzero initialization is used to insert either *a priori* jammer knowledge or the weight vector solution from a previous block of data.

We evaluated the above techniques for an adaptive beamforming application. We found that these techniques worked very well for cases with accurate *a priori* knowledge and stationary data. We also evaluated the approaches under imperfect and nonstationary conditions in order to determine the breakpoints where the techniques were no longer productive. For the cases tested we found that prior knowledge

of jammer directions needed to be known to a one-sigma accuracy of about 1/20th of a half-power beamwidth. Similarly prior knowledge of jammer power levels needed to be known to between 10 and 15 dB, one-sigma. For the recursive implementation we found that performance was sensitive to the introduction of new interference sources, and that when this occurred full adaptation to a new subspace was needed. When this occurred, however, the processor simply required the same adaptive rank as it would have required with zero vector initialization. Thus the algorithms were self-correcting.

REFERENCES

- [1] J. S. Goldstein, I. S. Reed, and L. L. Scharf, "A multistage representation of the Wiener filter based on orthogonal projections," *IEEE Trans. Information Theory*, vol. 44, no. 7, pp. 2943–2959, November 1998.
- [2] M. E. Weippert, J. D. Hiemstra, J. S. Goldstein, and M. D. Zoltowski, "Insights from the relationship between the multistage Wiener filter and the method of conjugate gradients," in *Proc. IEEE SAM2002 Workshop*, Arlington, VA, August 2002.
- [3] M. D. Zoltowski, "Conjugate gradient adaptive filtering with application to space-time processing for wireless digital communications," in *Proc. IEEE SAM2002 Workshop*, Arlington, VA, August 2002.
- [4] H. L. Van Trees, *Optimum Array Processing*, Wiley, New York, NY, 2002.
- [5] L. J. Griffiths and C. W. Jim, "An alternative approach to linearly constrained adaptive beamforming," *IEEE Trans. Antennas Propagat.*, vol. AP-30, no. 1, pp. 27–34, January 1982.
- [6] C. Y. Tseng and L. J. Griffiths, "A unified approach to the design of linear constraints in minimum variance adaptive beamformers," *IEEE Trans. Antenn. Propagat.*, vol. 40, no. 12, pp. 1533–1542, December 1992.
- [7] P. C. Hansen, *Rank-Deficient and Discrete Ill-Posed Problems*, SIAM, Philadelphia, PA, 1998.