6 Constant Modulus Beamforming

ALLE-JAN VAN DER VEEN and AMIR LESHEM

Delft University of Technology
Delft, The Netherlands

School of engineering, Bar-Ilan university
52900 Ramat-Gan, Israel

Algorithms for blind source separation aim to compute beamformers that select a desired source while suppressing interfering sources, without specific knowledge of the sources or the channel. The preceding chapters have described algorithms based on direction finding: sources are separated based on differences in spatial signature vectors (array response vectors). Such algorithms need to know the parametric structure of the array response, therefore they rely on calibrated arrays. A complementary class of algorithms uses the structural properties of the source modulation, and try to reconstruct, at the output of the beamformer, a signal that has this structure. A widely used property for this is based on the fact that many sources are phase modulated, therefore have a constant modulus. The related Constant Modulus Algorithms (CMAs) are studied in this chapter.

6.1 INTRODUCTION

In wireless communications, an elementary beamforming problem arises when a number of sources at distinct locations transmit signals at nominally the same carrier frequency and in the same time slot. The signals are received by the base station, which is assumed here to contain an array of antennas. By linearly combining the antenna outputs, the objective is to separate the signals and remove the interference from the other signals. In many cases of channel estimation and source separation, training sequences are available: a segment of the signal of interest which is known. In this chapter, we consider "blind" algorithms: a blind beamformer is to compute the proper weight vectors $\mathbf{w}_i$

from the measured data only, *without* detailed knowledge of the signals and the channel. It can do so by comparing properties of the signal at the output of the beamformer to properties that the desired source signal would have at this point.

For example, if we know that the desired source has an amplitude $|s_k|$ constant to 1 for every sample index k (such a signal is called *constant modulus* or CM), then we can test this property for the output signal y_k of the beamformer, and define an error equal to the modulus difference $|y_k|^2 - 1$, as in figure 6.1(a). Alternatively, we can estimate the best signal that has this property based on the output of the beamformer, i.e., $\hat{s}_k = \frac{y_k}{|y_k|}$, and give an error equal to $\hat{s}_k - y_k$. Here, $\hat{s}_k$ is regarded as a good estimate of the source signal, and it is used as a reference signal instead of s_k. This is an elementary form of *decision feedback*, which could be refined further if we know that the source belongs to a certain alphabet, e.g., $\{\pm 1\}$ for BPSK or $\{\pm 1, \pm j\}$ for QPSK. See figure 6.1(b).

Throughout most of this chapter we assume a stationary situation with essentially no delay spread (as compared to the inverse of the signal bandwidths), so that no equalization is required. With d sources and M receive antennas, the situation is described by the simple data model

$$\mathbf{x}_k = \mathbf{A}\mathbf{s}_k + \mathbf{n}_k \tag{6.1}$$

where the vector $\mathbf{x}_k$ is a stacking of the M antenna outputs $(x_i)_k$ at discrete time k, $\mathbf{s}_k$ is a stacking of the d source signals $(s_i)_k$, and $\mathbf{A}$ is the $M \times d$ array response matrix which describes the linear combinations of the signals as received by the antennas. The M-dimensional vector $\mathbf{n}_k$ is the additive noise. A beamformer $\mathbf{w}$ takes a linear combination of the antenna outputs, which is written as an inner product $y_k = \mathbf{w}^{\mathrm{H}}\mathbf{x}_k$, where $^{\mathrm{H}}$ denotes the complex conjugate transpose. The beamforming problem is to find weight vectors, one for each source, such that $\mathbf{w}_i^{\mathrm{H}}\mathbf{x}_k = (s_i)_k$ is equal to one of the original sources, without interference from the others. For this to be possible, we need to make (at least) the following assumptions, which are valid throughout the chapter:

1. $M \geq d$: more antennas than sources,

2. $\mathbf{A}$ has full column rank: its columns are linearly independent.

3. The power of the sources is absorbed in $\mathbf{A}$ (or a separate diagonal factor $\mathbf{B}$), therefore we may assume that all sources have equal, unit power.

4. $\mathbf{n}_k$ is white Gaussian noise, with covariance matrix $\mathrm{E}(\mathbf{n}_k\mathbf{n}_k^{\mathrm{H}}) = \sigma^2\mathbf{I}$, where E denotes the expectation operator.

Constant Modulus algorithms have been widely studied. The specific aim of this chapter is to look at algorithms that find the complete set of *all* beamformers (one for each impinging signal). The original Constant Modulus Algorithm (CMA) [1] can find only a single signal. A 1992 extension (the CM

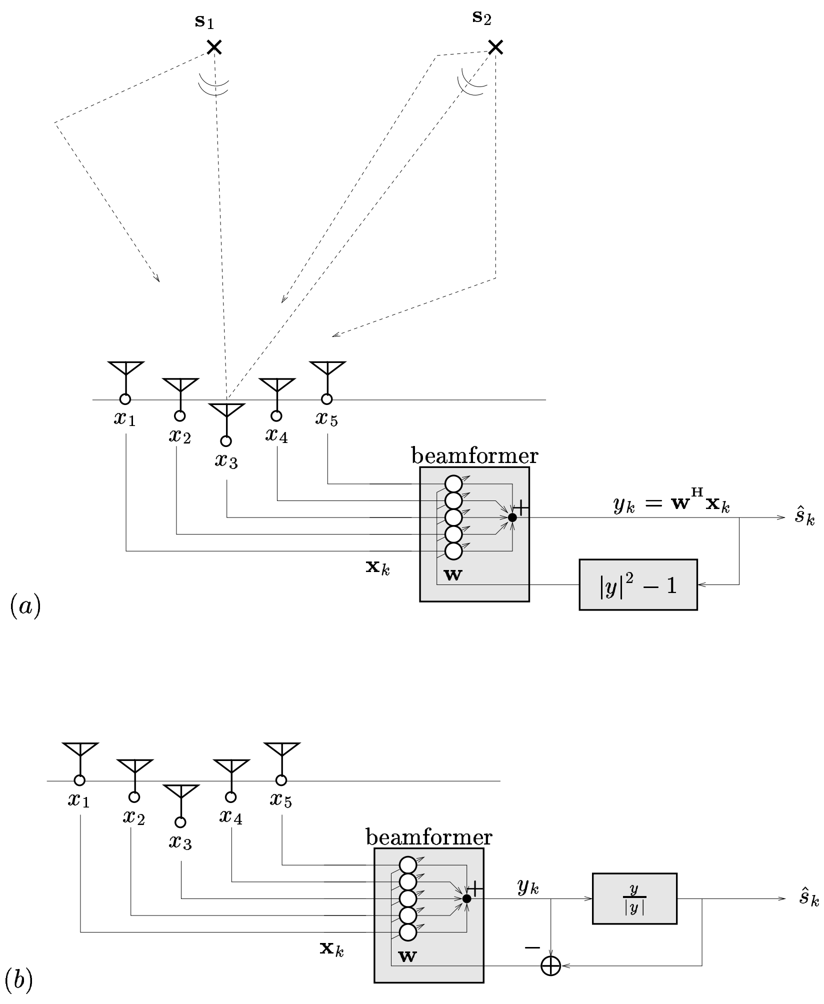


Fig. 6.1 Blind adaptive beamforming structures: (a) based on modulus error, (b) based on estimated output error.

Array [2, 3]) is a multistage algorithm based on successive cancellation; it has a poor performance. Currently there are two types of algorithms that can find all signals:

1. the "Algebraic CMA" (ACMA) [4] and similar algorithms such as JADE [5]: this is a non-iterative block algorithm acting on a batch of data which computes jointly beamforming vectors for all constant modulus sources as the solution of a joint diagonalization problem;
2. the "Multi-user Kurtosis" (MUK) algorithm [6]: an adaptive algorithm which can be viewed as a bank of CMAs with an orthogonality constraint between the beamformers.

Both algorithms are based on similar cost functions leading to 4-th order equations for the beamformer coefficients, and in both algorithms, prewhitening plays an important role. Our aim is to put the two algorithms in a common framework so that they can be compared. To this end, we derive an block-iterative version and an adaptive version of ACMA.

In a second part of the chapter, we study the application of CMA algorithms to *direction finding*. In these approaches, we use both the structure of the source $\mathbf{s}_k$ and the parametric structure of $\mathbf{A}$, namely each column of $\mathbf{A}$ is a vector on the array manifold associated to a certain direction-of-arrival (DOA). By combining both properties, increased estimation accuracy and robustness of the beamformers is obtained.

Notation We adopt the following notation:

$\bar{\cdot}$	complex conjugation,
$^{\mathrm{T}}$	matrix or vector transpose,
$^{\mathrm{H}}$	matrix or vector complex conjugate transpose,
$\dagger$	matrix pseudo-inverse (Moore-Penrose inverse),
$\underline{\cdot}$	prewhitened data,
$\mathbf{0}$	vector of all 0s,
$\mathbf{1}$	vector of all 1s,
$\mathrm{E}(\cdot)$	mathematical expectation operator,
$\hat{\cdot}$	estimated value of a variable,
$\mathrm{diag}(\mathbf{a})$	a diagonal matrix constructed from the vector $\mathbf{a}$,
$\mathrm{vec}(\mathbf{A})$	stacking of the columns of $\mathbf{A}$ into a vector,
$\odot$	Schur-Hadamard product (entrywise multiplication),
$\otimes$	Kronecker product,
$\circ$	Khatri-Rao product (column-wise Kronecker product):

$$\mathbf{A} \circ \mathbf{B} := [\mathbf{a}_1 \otimes \mathbf{b}_1 \quad \mathbf{a}_2 \otimes \mathbf{b}_2 \quad \cdots\,]\,. \tag{6.2}$$

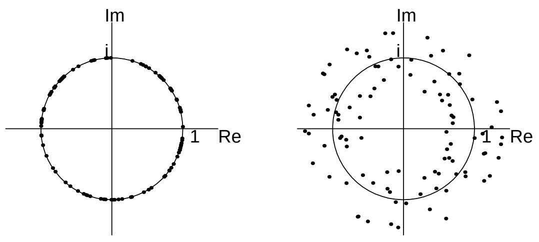


Fig. 6.2 (*a*) Single CM signal, (*b*) sum of two CM signals.

Notable properties are, for matrices $\mathbf{A}$, $\mathbf{B}$, $\cdots$ and vectors $\mathbf{a}$, $\mathbf{b}$ of compatible sizes,

$$\begin{aligned} \mathrm{vec}(\mathbf{a}\mathbf{b}^{\mathrm{H}}) &= \bar{\mathbf{b}} \otimes \mathbf{a} & (6.3)\\ (\mathbf{A}\otimes\mathbf{B})(\mathbf{C}\otimes\mathbf{D}) &= \mathbf{AC}\otimes\mathbf{BD} & (6.4)\\ \mathrm{vec}(\mathbf{ABC}) &= (\mathbf{C}^{\mathrm{T}}\otimes\mathbf{A})\mathrm{vec}(\mathbf{B}) & (6.5)\\ \mathrm{vec}(\mathbf{A}\,\mathrm{diag}(\mathbf{b})\,\mathbf{C}) &= (\mathbf{C}^{\mathrm{T}}\circ\mathbf{A})\mathbf{b}\,. & (6.6) \end{aligned}$$

6.2 THE CONSTANT MODULUS ALGORITHM

As mentioned, many communication signals have a constant modulus property. For such signals, the amplitude $|s_k|$ is a constant, typically normalized to 1, and all information is carried in the phase. If we have a single source s_k, and plot the (complex) samples in the complex plane, then all samples will lie on the unit circle, see figure 6.2. On the other hand, if we have the sum of two sources, $(s_1)_k + \alpha(s_2)_k$, then the samples will in general not lie on a circle, unless $\alpha = 0$ (or if there are very special relations between the two sources—this is not possible if the two sources are independent). If $\alpha \neq 0$, then the received samples will be on a donut-shaped annulus.

The idea of modulus restoral is to play with the weights of a beamformer $\mathbf{w}$ until the output $y_k = \hat{s}_k = \mathbf{w}^{\mathrm{H}}\mathbf{x}_k$ has the same property, $|\hat{s}_k| = 1$, for all k. If that is the case, the output signal will be equal to one of the original sources [1], up to an unknown phase factor which cannot be established blindly.

6.2.1 CMA cost function

Popular implementations of such a property restoral algorithm are found by writing down a suitable cost function and minimizing it using stochastic gradient-descent techniques. For example, for a sample vector $\mathbf{x}_k$ we can consider as cost function the expected deviation of the squared modulus of the output signal $y_k = \mathbf{w}^{\mathrm{H}}\mathbf{x}_k$ to a constant, say 1:

$$J(\mathbf{w}) \;=\; \mathrm{E}(|y_k|^2 - 1)^2 = \mathrm{E}(|\mathbf{w}^{\mathrm{H}}\mathbf{x}_k|^2 - 1)^2 \,. \tag{6.7}$$

This so-called CMA(2,2) cost function is simply a positive measure of the average amount that the beamformer output y_k deviates from the unit modulus condition. The objective in choosing $\mathbf{w}$ is to minimize J and hence to make y_k as close to a constant modulus signal as possible. Without additive noise, if we manage to achieve $J(\mathbf{w}) = 0$ then $\mathbf{w}$ reconstructs one of the sources.

A closed-form solution which will minimize the CM cost function (6.7) appears to be impossible because it is a fourth-order function with a more complicated structure. However, there are many ways in which we can iteratively search for the minimum of J. The simplest algorithm follows from a stochastic gradient-descent, similar to the derivation of the LMS algorithm by Widrow [7]. In this case, we update $\mathbf{w}$ iteratively, with small steps into the direction of the negative gradient,

$$\mathbf{w}^{(k+1)} \;=\; \mathbf{w}^{(k)} - \mu\,\nabla(J_k)$$

where μ is a small step size, and $\nabla(J_k) \equiv \nabla_{\bar{\mathbf{w}}} J(\mathbf{w}^{(k)})$ is the gradient vector of $J(\mathbf{w})$ with respect to the entries of $\bar{\mathbf{w}}$ (treated independently from $\mathbf{w}$), evaluated at the current value of $\mathbf{w}$. Using complex calculus and the fact that $|y_k|^2 = y_k\bar{y}_k = \mathbf{w}^{\mathrm{H}}\mathbf{x}_k\mathbf{x}_k^{\mathrm{H}}\mathbf{w}$, it can be verified that the gradient is given by

$$\begin{aligned}
\nabla(J_k) \;&=\; \mathrm{E}\{(|y_k|^2 - 1)\cdot\nabla(\mathbf{w}^{\mathrm{H}}\mathbf{x}_k\mathbf{x}_k^{\mathrm{H}}\mathbf{w})\} \\
&=\; 2\mathrm{E}\{(|y_k|^2 - 1)\cdot\mathbf{x}_k\mathbf{x}_k^{\mathrm{H}}\mathbf{w}\} \\
&=\; 2\mathrm{E}\{(|y_k|^2 - 1)\bar{y}_k\mathbf{x}_k\}\,.
\end{aligned}$$

Replacing the expectation by an instantaneous estimate, as in LMS, shows that we can find a minimizer $\mathbf{w}$ iteratively via

$$\mathbf{w}^{(k+1)} \;=\; \mathbf{w}^{(k)} \;-\; \mu\mathbf{x}_k\,\bar{z}_k\,, \qquad y_k := \mathbf{w}^{(k)H}\mathbf{x}_k\,, \quad z_k := (|y_k|^2 - 1)y_k \tag{6.8}$$

(absorbing the factor 2 in μ). This iteration is called the Constant Modulus Algorithm (CMA, Treichler Agee and Larimore 1983 [8, 1, 3]) and was first introduced for the case of blind equalization. It has its roots in the work of Sato [9] and Godard [10]. See [11, 12, 13] for overviews and a historical perspective.

In comparison to the LMS algorithm, we see that the role of the update error (in the LMS equal to the output error $\epsilon_k = y_k - s_k$) is here played by z_k. In the LMS, we need a reference signal s_k to which we want to have the output converge. In CMA, however, the reference signal is not necessary, we use the a priori information that $|y_k| = 1$ in the absence of interfering signals.

We need to select a suitable step size μ and an initial point $\mathbf{w}^{(0)}$ for the iteration. Unlike LMS, we cannot choose $\mathbf{w}^{(0)} = \mathbf{0}$ since this precisely selects a local maximum of the cost function, but any other random vector will do. The maximal step size has not been theoretically derived. Because the cost function involves fourth order moments, the gradient is much steeper away from the optimum in comparison to LMS, and the maximal μ that still guarantees stability is smaller.

The constant modulus property holds for all phase-modulated and frequency modulated signals, and for several types of signals in the digital domain, such as frequency-shift keying (FSK), phase-shift keying (PSK), binary-shift keying (BPSK) and 4-QAM. For digital signals, the fact that the source symbols are selected from a finite alphabet is an even stronger property that can very well be exploited. Even for sources that are not constant modulus, such as multi-level constellations (higher order QAM), the CMA can be successfully applied. The algorithm has been widely used in modem equalization.

6.2.2 Variants of the adaptive CMA

Instead of the cost function in (6.7), a slightly different cost function is also often considered, namely the "CMA(1,2)" cost function

$$J(\mathbf{w}) \;=\; \mathrm{E}(|y_k| - 1)^2 = \mathrm{E}(|\mathbf{w}^{\mathrm{H}}\mathbf{x}_k| - 1)^2 \,. \tag{6.9}$$

The associated update rule is given by [2]

$$\mathbf{w}^{(k+1)} \;:=\; \mathbf{w}^{(k)} - \mu \mathbf{x}_k \, \bar{z}_k \,, \qquad \text{where} \quad y_k := \mathbf{w}^{(k)H}\mathbf{x}_k \,, \quad z_k = y_k - \frac{y_k}{|y_k|} \,. \tag{6.10}$$

In this case, the update error that controls the iteration is $(y - \frac{y}{|y|})$. Compared to the LMS, we see that $\frac{y}{|y|}$ plays the role of desired signal, see also figure 6.1(b). Ideally, y_k is constant modulus and the error is zero. An advantage of this iteration is that the role of μ is more closely related to that of LMS, facilitating its analysis close to convergence. It also allows to pose a "Normalized CMA" [14] similar to the Normalized LMS by Goodwin [15],

$$\mathbf{w}^{(k+1)} \;:=\; \mathbf{w}^{(k)} - \frac{\mu}{\|\mathbf{x}_k\|^2}\mathbf{x}_k \, \bar{z}_k \,, \tag{6.11}$$

where μ is made data scaling independent by dividing by the instantaneous input power. For the NLMS, this modification was obtained by computing the optimal stepsize which would minimize the instantaneous error; it is known that $0 < \mu < 2$ is required for stability, although one would take $\mu \ll 1$ to obtain a sufficiently smooth performance. The same range holds for NCMA. A better but slightly more complicated "orthogonalization" version of CMA was considered in [2] and became known as orthogonal CMA (OCMA, see also [16]),

$$\mathbf{w}^{(k+1)} \;:=\; \mathbf{w}^{(k)} - \mu \hat{\mathbf{R}}_k^{-1}\mathbf{x}_k \, \bar{z}_k \,, \tag{6.12}$$

where $\hat{\mathbf{R}}_k$ is an estimate of the data covariance matrix $\mathbf{R} = \mathrm{E}(\mathbf{x}_k\mathbf{x}_k^{\mathrm{H}})$. Usually $\hat{\mathbf{R}}_k$ is estimated by a sliding window estimate

$$\hat{\mathbf{R}}_k = \lambda\hat{\mathbf{R}}_{k-1} + (1-\lambda)\mathbf{x}_k\mathbf{x}_k^{\mathrm{H}}\,, \tag{6.13}$$

where $0 < \lambda < 1$ determines the exponential window size (the effective window size is defined as $1/(1-\lambda)$). The inverse of $\hat{\mathbf{R}}_k$ can be efficiently updated using techniques known from the RLS algorithm [15].

There are a few other CMAs, in particular the LS-CMA, which will be discussed in section 6.4.2.

6.2.3 The CM Array

The CMA gives only a single beamformer vector. This is sufficient for blind equalization applications, where the received signal consists of several temporal shifts of the same CM signal, and we do not have to recover all of them. In contrast, the beamforming problem frequently asks for all possible weight vectors that give back linearly independent CM signals, which is usually much harder.

If we initialize the CMA with a random vector $\mathbf{w}^{(0)}$, then the CMA tends to converge to the strongest signal. However, this cannot be guaranteed: for an initialization close enough to a weak signal, the algorithm converges to that weaker signal.[1] This gives one way to find all signals: use various initializations and determine if independent signals have been found.

A somewhat more robust algorithm is the so-called multi-stage CMA, also called the CM Array. It was introduced in [2, 3], with analysis appearing in [17, 18, 19]. The output of a first CMA stage results in the detection of the first CM signal, and gives an estimate $\hat{s}_1(k)$. This signal can be used as a reference signal for an LMS algorithm to estimate the corresponding array response vector $\hat{\mathbf{a}}_1$, an estimate of the first column of $\mathbf{A}$. The resulting update rule is

$$\hat{\mathbf{a}}_1^{(k+1)} = \hat{\mathbf{a}}_1^{(k)} \;+\; \mu_{lms}[\mathbf{x}(k) - \hat{\mathbf{a}}_1^{(k)}\hat{s}_1(k)]\,\bar{\hat{s}}_1(k)\,.$$

We can then subtract the estimated source signal from the original data sequence,

$$\mathbf{x}_1(k) = \mathbf{x}(k) - \hat{\mathbf{a}}_1^{(k)}\hat{s}_1(k)$$

and feed the resulting filtered data to a second CMA stage in order to detect a possible second CM signal. This can be repeated until all signals have been found. See figure 6.3.[2]

[1]During the early days of CMA, only the reception of the strongest signal was desired, and convergence to another signal was regarded as mis-convergence.

[2]The algorithm used in the CM Array in [17, 18, 19] is in fact based on the CMA(1,2) as in equation (6.10).

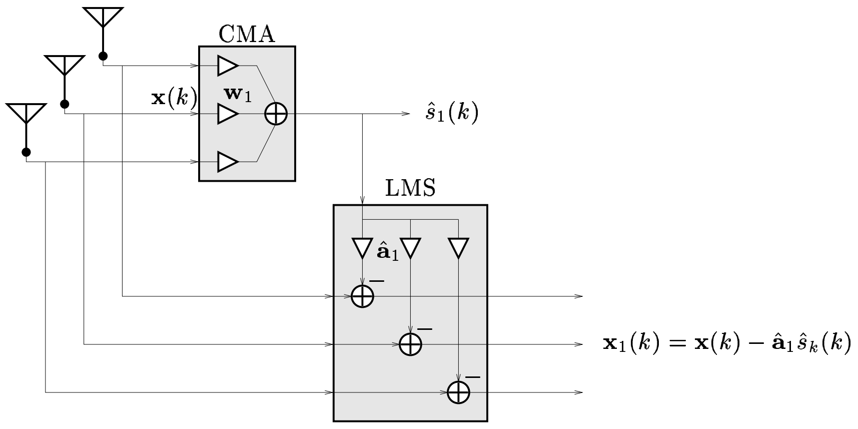


Fig. 6.3 The CM Array.

A problem with this scheme is that the LMS algorithm can converge only after the first CMA has sufficiently converged. In the mean time, the second CMA may have converged to the same signal as the first CMA (especially if it is strong), and if the first LMS is not completely removing this signal, the second stage will stay at this signal. Thus, it may happen that the same signal is found twice, and/or that not all signals are found. A related problem is that the CMA converges to a point close to the Wiener solution. Hence, the estimate $\hat{s}_1(k)$ will always contain components of the other signals as well, causing misadjustment in later stages. An analysis of the situation is given in [17, 19].

Another problem is that the convergence speed may be slow (several hundreds of samples), since we have a cascade of adaptive algorithms. It has been proposed to use direction-finding algorithms such as MUSIC first to initialize the cascade. An alternative approach is to augment the cost function with additional terms that express the independence of the output signals, e.g., by putting a constraint on the cross-correlation of the recovered signals [20, 21]. Algorithms for this are discussed in section 6.4. Some of the problems are alleviated by prewhitening, which is discussed in the next section.

6.3 PREWHITENING AND RANK REDUCTION

6.3.1 Rank reduction

At this point, let us first make a small extension to our notation. Starting from the data model $\mathbf{x}_k = \mathbf{A}\mathbf{s}_k + \mathbf{n}_k$, we assume that we have received N sample vectors, $k = 1, \cdots, N$. It is often convenient to collect the data in matrices:

$$\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_N]\,, \qquad \mathbf{S} = [\mathbf{s}_1, \mathbf{s}_2, \cdots, \mathbf{s}_N]\,,$$

and likewise for a noise matrix $\mathbf{N}$, so that the model becomes

$$\mathbf{X} = \mathbf{AS} + \mathbf{N}\,.$$

The objective is to recover all beamformers $\mathbf{w}_i$, $i = 1, \cdots, d$, one for each source. These can also be collected into a matrix $\mathbf{W}$,

$$\mathbf{W} = [\mathbf{w}_1, \cdots, \mathbf{w}_d]\,, \qquad M \times d\,.$$

In the noise-free case, we would like to achieve $\mathbf{W}^{\mathrm{H}}\mathbf{X} = \mathbf{S}$. However, with no other knowledge about $\mathbf{S}$ but the constant modulus property, there are two causes of non-uniqueness:

1. The ordering of sources is arbitrary: we do not know which is "source number 1", etcetera. Therefore the ordering of beamformers (columns of $\mathbf{W}$) is arbitrary: $\mathbf{W}$ will always have a permutation ambiguity.
2. The solution for each beamformer can be found only up to an arbitrary phase, since the CM cost function is phase-blind.

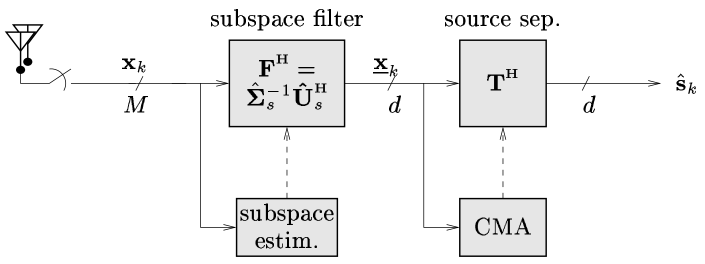
This kind of non-uniqueness is common to all blind source separation problems. In the estimation of a beamformer, there may be one other cause of non-uniqueness. Namely, additional "nullspace solutions" exist if the received data matrix $\mathbf{X}$ is rank deficient: in this case there are beamformers $\mathbf{w}_0$ such that $\mathbf{w}_0^{\mathrm{H}}\mathbf{X} = \mathbf{0}^{\mathrm{H}}$. Such solutions can be added to any beamformer $\mathbf{w}_i$ and cause non-uniqueness: two linearly independent beamformers ($\mathbf{w}_i$ and $\mathbf{w}_i + \mathbf{w}_0$) may still reconstruct the same signal. This is clearly undesired if our aim is to reconstruct all independent signals, because the simplest way to detect if independent signals have been obtained is to verify the linear independence of the corresponding beamformers.

Nullspace solutions exist if the number of sensors is larger than the number of sources ($\mathbf{A}$ tall), and if $\mathbf{A}$ is not full column rank. The former is simply treated by a prefiltering operation that reduces the number of rows of $\mathbf{X}$ from M to d, as we discuss here, whereas the latter case is hopeless, at least for linear receivers.

We will use the underscore (_) to denote prefiltered variables. Thus, let $\underline{\mathbf{X}} := \mathbf{F}^{\mathrm{H}}\mathbf{X}$ where $\mathbf{F} : M \times d$ is the prefilter. Then

$$\underline{\mathbf{X}} = \underline{\mathbf{A}}\mathbf{S} + \underline{\mathbf{N}}\,, \qquad \text{where} \quad \underline{\mathbf{A}} := \mathbf{F}^{\mathrm{H}}\mathbf{A}\,, \ \ \underline{\mathbf{N}} := \mathbf{F}^{\mathrm{H}}\mathbf{N}\,.$$

This is essentially the same model as before, except $\underline{\mathbf{X}}$ has only d channels and $\underline{\mathbf{A}} :\ d \times d$ is square. The blind beamforming problem is now replaced by finding a separating beamforming matrix $\mathbf{T} : d \times d$ with columns $\mathbf{t}_i$, acting on $\underline{\mathbf{X}}$. After $\mathbf{T}$ has been found, the beamforming matrix on the original data will be $\mathbf{W} = \mathbf{FT}$. The associated processing structure is illustrated in figure 6.4.



subspace filter
source sep.
$\mathbf{x}_k$
M
$\mathbf{F}^{\mathrm{H}} = \hat{\mathbf{\Sigma}}_s^{-1}\hat{\mathbf{U}}_s^{\mathrm{H}}$
$\underline{\mathbf{x}}_k$
d
$\mathbf{T}^{\mathrm{H}}$
d
$\hat{\mathbf{s}}_k$
subspace estim.
CMA

Fig. 6.4 Blind beamforming prefiltering structure.

For the purpose of dimension reduction, there are many suitable $\mathbf{F}$: the only requirement is that $\mathbf{F}^{\mathrm{H}}\mathbf{A}$ should be full rank. To avoid noise enhancement, we also want it to be well conditioned. This leads to the choice of $\mathbf{F}$ to have orthogonal columns that together span the column span of $\mathbf{A}$.

6.3.2 Whitening

Assume that the noise is white i.i.d. with covariance matrix $\mathbf{R_n} = \mathrm{E}(\mathbf{nn}^{\mathrm{H}}) = \sigma^2\mathbf{I}$. We can choose $\mathbf{F}$ such that the resulting data matrix $\underline{\mathbf{X}}$ is white, as follows. Let $\hat{\mathbf{R}}_{\mathbf{x}} = \frac{1}{N}\sum_{k=1}^{N}\mathbf{x}_k\mathbf{x}_k^{\mathrm{H}} = \frac{1}{N}\mathbf{XX}^{\mathrm{H}}$ be the noisy sample data covariance matrix, with eigenvalue decomposition

$$\hat{\mathbf{R}}_{\mathbf{x}} \;=\; \hat{\mathbf{U}}\hat{\mathbf{\Sigma}}^2\hat{\mathbf{U}}^{\mathrm{H}} = [\hat{\mathbf{U}}_s \quad \hat{\mathbf{U}}_n]\begin{bmatrix}\hat{\mathbf{\Sigma}}_s^2 & \\ & \hat{\mathbf{\Sigma}}_n^2\end{bmatrix}\begin{bmatrix}\hat{\mathbf{U}}_s^{\mathrm{H}} \\ \hat{\mathbf{U}}_n^{\mathrm{H}}\end{bmatrix}. \tag{6.14}$$

Here, $\hat{\mathbf{U}} = [\hat{\mathbf{U}}_s \;\; \hat{\mathbf{U}}_n]$ is $M\times M$ unitary, and $\hat{\mathbf{\Sigma}}^2$ is $M\times M$ diagonal ($\hat{\mathbf{\Sigma}}$ contains the singular values of $\mathbf{X}/\sqrt{N}$). The d largest eigenvalues are collected into a diagonal matrix $\hat{\mathbf{\Sigma}}_s^2$ and the corresponding d eigenvectors into $\hat{\mathbf{U}}_s$ (they span the "signal subspace"). In this notation, define $\mathbf{F}$ as

$$\mathbf{F} = \hat{\mathbf{U}}_s\hat{\mathbf{\Sigma}}_s^{-1}\,. \tag{6.15}$$

This prewhitening is such that $\underline{\hat{\mathbf{R}}}_{\mathbf{x}} := \frac{1}{N}\underline{\mathbf{XX}}^{\mathrm{H}}$ is unity: $\underline{\hat{\mathbf{R}}}_{\mathbf{x}} = \mathbf{I}$, and at the same time it reduces the dimension of $\underline{\mathbf{X}}$ from M rows to d rows.

An important reason to choose a prefilter that whitens the output is that the resulting $\underline{\mathbf{A}}$ in the whitened domain is approximately unitary. Indeed, let $\mathbf{A} = \mathbf{U}_A\mathbf{\Sigma}_A\mathbf{V}_A^{\mathrm{H}}$ be an "economy-size" SVD of $\mathbf{A}$ ($\mathbf{U}_A$ is a submatrix of a unitary matrix and has size $M\times d$, $\mathbf{V}_A$ is size $d\times d$ and is unitary, and $\mathbf{\Sigma}_A$ is $d\times d$ diagonal and contains the singular values), then for a large number of samples, $\hat{\mathbf{R}}_{\mathbf{x}} \approx \mathbf{R}_{\mathbf{x}}$, where

$$\mathbf{R}_{\mathbf{x}} = \mathbf{AA}^{\mathrm{H}} + \sigma^2\mathbf{I} = [\mathbf{U}_A \quad \mathbf{U}_A^{\perp}]\begin{bmatrix}\mathbf{\Sigma}_A^2+\sigma^2\mathbf{I} & \\ & \sigma^2\mathbf{I}\end{bmatrix}\begin{bmatrix}\mathbf{U}_A^{\mathrm{H}} \\ (\mathbf{U}_A^{\perp})^{\mathrm{H}}\end{bmatrix}$$

so that we can identify $\mathbf{U}_s = \mathbf{U}_A$ and $\mathbf{\Sigma}_s^2 = \mathbf{\Sigma}_A^2 + \sigma^2\mathbf{I}$. Therefore,

$$\underline{\mathbf{A}} = \mathbf{F}^{\mathrm{H}}\mathbf{A} = \mathbf{\Sigma}_s^{-1}\mathbf{U}_s^{\mathrm{H}}\mathbf{U}_A\mathbf{\Sigma}_A\mathbf{V}_A^{\mathrm{H}} = (\mathbf{\Sigma}_s^{-1}\mathbf{\Sigma}_A)\mathbf{V}_A^{\mathrm{H}} .$$

This shows that $\underline{\mathbf{A}}$ is unitary if $(\mathbf{\Sigma}_A^2 + \sigma^2\mathbf{I})^{-1/2}\mathbf{\Sigma}_A = \mathbf{I}$, or a scalar multiple of $\mathbf{I}$, which is the case if there is no noise or if the singular values of $\mathbf{A}$ are all the same (then $\mathbf{A}$ has orthonormal columns, which corresponds to well-separated equal-powered sources). If this is not the case, then $\underline{\mathbf{A}}$ is only approximately unitary, but always better conditioned than $\mathbf{A}$ itself (the conditioning of a matrix is the ratio of its largest and smallest singular value, preferably this is a number close to 1).

It has sometimes been suggested to use a slightly different prewhitening filter, $\mathbf{F} = \hat{\mathbf{U}}_s\hat{\mathbf{\Sigma}}_A^{-1}$, where $\hat{\mathbf{\Sigma}}_A = (\hat{\mathbf{\Sigma}}_s^2 - \sigma^2\mathbf{I})^{1/2}$ is an estimate of $\mathbf{\Sigma}_A$. This would yield a unitary $\underline{\mathbf{A}}$. Care has to be taken to ensure that $\hat{\mathbf{\Sigma}}_s^2 - \sigma^2\mathbf{I}$ is positive, e.g., by replacing σ^2 by a data-dependent estimate.

In the noise-free case, the optimal beamformer is $\mathbf{T} = \underline{\mathbf{A}}^{-H}$. The importance of having a unitary $\underline{\mathbf{A}}$-matrix is that the corresponding beamformer is $\mathbf{T} = \underline{\mathbf{A}}$ is also unitary, hence has maximally independent columns. If our aim is to find all independent sources in the data, it is very convenient to know that the beamformers are orthogonal: if we have found one solution $\mathbf{t}_1$, we can constrain the other solutions to be orthogonal to it, which is a simpler constraint than requiring them to be linearly independent. If $\underline{\mathbf{A}}$ is only approximately unitary, e.g., in the presence of noise, then the orthogonality condition on $\mathbf{T}$ has to be relaxed to being "well-conditioned".

It is well recognized in adaptive filtering that moving to the whitened domain improves the convergence speed: this is often limited by the conditioning of the data covariance matrix, which becomes optimal after whitening. (A disadvantage is that the noise is not longer white nor spatially independent.) This is the motivation of introducing the factor $\mathbf{R}_{\mathbf{x}}^{-1}$ in the OCMA in equation (6.12).

6.3.3 Convergence to the Wiener beamformer

In a stochastic context, the Wiener beamformer is defined as the solution to the linear minimum mean square error (LMMSE) problem

$$\mathbf{w} = \arg\min_{\mathbf{w}} \mathrm{E}|\mathbf{w}^{\mathrm{H}}\mathbf{x}_k - s_k|^2 ,$$

where s_k is known to the receiver. The solution is straightforward to derive,

$$\begin{aligned}\mathrm{E}|\mathbf{w}^{\mathrm{H}}\mathbf{x}_k - s_k|^2 &= \mathbf{w}^{\mathrm{H}}\mathrm{E}(\mathbf{x}_k\mathbf{x}_k)\mathbf{w}^{\mathrm{H}} - \mathbf{w}^{\mathrm{H}}\mathrm{E}(\mathbf{x}_k\bar{s}_k) - \mathrm{E}(s_k\mathbf{x}_k^{\mathrm{H}})\mathbf{w} + \mathrm{E}(|s_k|^2) \\ &= \mathbf{w}^{\mathrm{H}}\mathbf{R}_{\mathbf{x}}\mathbf{w} - \mathbf{w}^{\mathrm{H}}\mathbf{r}_{\mathbf{x}s} - \mathbf{r}_{\mathbf{x}s}^{\mathrm{H}}\mathbf{w} + r_s \\ &= (\mathbf{w} - \mathbf{R}_{\mathbf{x}}^{-1}\mathbf{r}_{\mathbf{x}s})^{\mathrm{H}}\mathbf{R}_{\mathbf{x}}(\mathbf{w} - \mathbf{R}_{\mathbf{x}}^{-1}\mathbf{r}_{\mathbf{x}s}) + r_s - \mathbf{r}_{\mathbf{x}s}^{\mathrm{H}}\mathbf{R}_{\mathbf{x}}^{-1}\mathbf{r}_{\mathbf{x}s} ,\end{aligned}$$

where $\mathbf{R}_\mathbf{x}$ is the data covariance matrix, and $\mathbf{r}_{\mathbf{x}s}$ is the correlation of the received data with the transmitted signal. Therefore, the optimal beamformer is given by $\mathbf{w} = \mathbf{R}_\mathbf{x}^{-1}\mathbf{r}_{\mathbf{x}s}$. Similarly, if a vector $\mathbf{s}_k$ of d sources is specified, the collection of beamformers is $\mathbf{W} = \mathbf{R}_\mathbf{x}^{-1}\mathbf{R}_{\mathbf{xs}}$, where $\mathbf{R}_{\mathbf{xs}} = \mathrm{E}(\mathbf{x}_k\mathbf{s}_k^{\mathrm{H}})$. Assuming the noise is independent from the signals, and the sources are i.i.d., we obtain $\mathbf{R}_{\mathbf{xs}} = \mathbf{A}$ and

$$\mathbf{W} = \mathbf{R}_\mathbf{x}^{-1}\mathbf{A}\,. \tag{6.16}$$

In a deterministic context, the Wiener beamformer based on sample data is derived similarly as the solution to the Least Squares problem

$$\hat{\mathbf{W}} = \arg\min_{\mathbf{W}} \; \| \mathbf{W}^{\mathrm{H}}\mathbf{X} - \mathbf{S} \|_{\mathrm{F}}^2 = (\mathbf{S}\mathbf{X}^\dagger)^{\mathrm{H}} = (\tfrac{1}{N}\mathbf{X}\mathbf{X}^{\mathrm{H}})^{-1}\tfrac{1}{N}\mathbf{X}\mathbf{S}^{\mathrm{H}}\,. \tag{6.17}$$

(The subscript $_\mathrm{F}$ indicates the Frobenius norm.) As $N \to \infty$, the deterministic Wiener beamformer converges to (6.16). The importance of the Wiener beamformer is that it can be shown to maximize the Signal to Interference and Noise Ratio (SINR) among linear receivers.

Suppose we act in the whitened domain: $\underline{\mathbf{R}}_\mathbf{x} = \mathbf{I}$. The Wiener solution is then $\mathbf{T} = \underline{\mathbf{A}}$: each column $\mathbf{t}_i$ of the beamformer is equal to the whitened direction vector (a matched spatial filter). If we go back to the resulting beamformer $\mathbf{w}_i$ acting on the original (unwhitened) data matrix $\mathbf{X}$, we find (for $i = 1, \cdots, d$)

$$\mathbf{t}_i = \underline{\mathbf{a}}_i = \mathbf{F}^{\mathrm{H}}\mathbf{a}_i \qquad \Rightarrow \qquad \mathbf{w}_i = \mathbf{F}\mathbf{t}_i = \mathbf{F}\mathbf{F}^{\mathrm{H}}\mathbf{a}_i = \mathbf{R}_\mathbf{x}^{-1}\mathbf{a}_i\,.$$

If the whitening did not involve a dimension reduction, the last step is clear. But even if $\mathbf{F}$ is defined to reduce the dimension to the signal subspace the result holds (assuming the noise is white i.i.d.), since

$$\mathbf{R}_\mathbf{x}^{-1} = \mathbf{U}\mathbf{\Sigma}^{-2}\mathbf{U}^{\mathrm{H}} \;=\; \mathbf{U}_s\mathbf{\Sigma}_s^{-2}\mathbf{U}_s^{\mathrm{H}} + \sigma^{-2}\mathbf{U}_n\mathbf{U}_n^{\mathrm{H}} = \mathbf{F}\mathbf{F}^{\mathrm{H}} + \sigma^{-2}\mathbf{U}_n\mathbf{U}_n^{\mathrm{H}}$$

whereas $\mathbf{U}_n^{\mathrm{H}}\mathbf{a}_i = \mathbf{0}$. Therefore, the beamformer in the original domain is equal to the Wiener beamformer (6.16). In general, this is a very attractive property.

Several algorithms can be shown to converge to the true $\mathbf{a}$-vector or $\mathbf{A}$-matrix asymptotically, in case the number of samples $N \to \infty$ or in case the signal to noise ratio SNR $\to \infty$. Such algorithms, when applied in the whitened domain, converge to Wiener beamformers. A caveat in the proof is that, in the whitened domain, the noise covariance $\sigma^2\mathbf{F}\mathbf{F}^{\mathrm{H}}$ is in general not white anymore whereas many algorithms are based on this assumption. (For well separated and equal powered sources, the resulting noise covariance is still white.)

In summary, the prewhitening serves two purposes:

1. The dimension reduction (from number of antennas M to number of sources d) avoids the existence of additional "nullspace solutions", which facilitates the reconstruction of all independent signals.
2. The prewhitening will improve the conditioning of the problem: after prewhitening, $\underline{\mathbf{A}}$ is approximately unitary, therefore the beamformers in the whitened domain are approximately orthogonal. This greatly facilitates the convergence of iterative algorithms to independent solutions, and sometimes also guarantees the convergence to beamformers that are close to the Wiener beamformer.

6.4 MULTI-USER CMA TECHNIQUES

6.4.1 OCMA and MUK

The CMA(2,2) cost function was shown in equation (6.7). The corresponding adaptive algorithm (stochastic gradient) is

$$\mathbf{w}_{k+1} = \mathbf{w}_k - \mu \mathbf{x}_k \bar{z}_k\,, \qquad z_k = (|y_k|^2 - 1) y_k$$

where $y_k = \mathbf{w}_k^{\mathrm{H}} \mathbf{x}_k$ is the output of the beamformer using its current estimate, and μ is a small step size. We also introduced the OCMA in equation (6.12) [2], which premultiplied the update term by $\hat{\mathbf{R}}_{\mathbf{x}}^{-1}$ to make the algorithm independent of the scaling of the data:

$$\mathbf{w}_{k+1} = \mathbf{w}_k - \mu \hat{\mathbf{R}}_{\mathbf{x}}^{-1} \mathbf{x}_k \bar{z}_k\,, \qquad z_k = (|y_k|^2 - 1) y_k\,. \tag{6.18}$$

We can directly interprete this algorithm in terms of our prewhitening step. Indeed, define $\mathbf{t} = \hat{\mathbf{R}}_{\mathbf{x}}^{1/2} \mathbf{w}$ and $\underline{\mathbf{x}} = \hat{\mathbf{R}}_{\mathbf{x}}^{-1/2} \mathbf{x}$. Premultiplying (6.18) with $\hat{\mathbf{R}}_{\mathbf{x}}^{1/2}$ leads to

$$\mathbf{t}_{k+1} = \mathbf{t}_k - \mu \underline{\mathbf{x}}_k \bar{z}_k\,, \qquad z_k = (|y_k|^2 - 1) y_k$$

where $y_k = \mathbf{w}_k^{\mathrm{H}} \mathbf{x}_k = \mathbf{t}_k^{\mathrm{H}} \underline{\mathbf{x}}_k$. Therefore, OCMA is equal to the ordinary CMA, but in the whitened domain. The algorithm is easily modified to update d beamformers in parallel:

$$\mathbf{T}_{k+1} = \mathbf{T}_k - \mu \underline{\mathbf{x}}_k \mathbf{z}_k^{\mathrm{H}}\,, \qquad \mathbf{z}_k = (\mathbf{y}_k \odot \bar{\mathbf{y}}_k - \mathbf{1}) \odot \mathbf{y}_k$$

where $\mathbf{y}_k = \mathbf{W}_k^{\mathrm{H}} \mathbf{x}_k = \mathbf{T}_k^{\mathrm{H}} \underline{\mathbf{x}}_k$ and $\odot$ denotes the Schur-Hadamard product (entrywise multiplication). In spite of its appearance, the beamformers are updated independently, and there is no guarantee that they converge to independent solutions. However, since $\mathbf{T}$ is supposed to be almost orthogonal and therefore well-conditioned (linearly independent columns), it is straightforward to recondition $\mathbf{T}$ after each update.

A simple technique to restore the linear independence of the solutions is to compute a singular value decomposition of $\mathbf{T}$ as $\mathbf{T} = \sum \sigma_j \mathbf{u}_j \mathbf{v}_j^{\mathrm{H}}$, and to

```
Given $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, \cdots]$ and a stepsize $\mu$, compute beamformers $\mathbf{W}_k = \mathbf{F}_k\mathbf{T}_k$,
output vector $\mathbf{s}_k = \mathbf{W}_k^{\mathrm{H}}\mathbf{x}_k$:

Initialize prewhitening filter $\mathbf{F}$ using $m$ prior input samples; $\mathbf{T} = \mathbf{I}_{d\times d}$.

For $k = 1, 2, \cdots$ do
    1. Update $\mathbf{F}$, the prewhitening filter, using $\mathbf{x}_k$ (fig. 6.11)            $(4dM + 3d^2)$
       $\underline{\mathbf{x}} = \mathbf{F}^{\mathrm{H}}\mathbf{x}_k$, the prewhitened input vector
    2. $\mathbf{y} = \mathbf{T}^{\mathrm{H}}\underline{\mathbf{x}}$, the current beamformer output vector            $(d^2)$
       $\mathbf{z} = \mathbf{y} \odot \bar{\mathbf{y}} \odot \mathbf{y}$ (MUK-update)
          or $\mathbf{z} = (\mathbf{y} \odot \bar{\mathbf{y}} - \mathbf{1}) \odot \mathbf{y}$ (OCMA-update)            $(3d)$
       $\mathbf{T} = \mathbf{T} - \mu\underline{\mathbf{x}}\mathbf{z}^{\mathrm{H}}$            $(d^2)$
       $\mathbf{T} = \mathrm{reorth}(\mathbf{T})$, equation (6.20); or $\mathbf{T} = \mathrm{recond}(\mathbf{T})$, equation (6.19)   $(d^3)$
    3. $\hat{\mathbf{s}}_k = \mathbf{y}$
                                                                      $4dM + d^3 + 5d^2$
```

Fig. 6.5 Multi-user CMAs: MUK and OCMA algorithm (in brackets the complexity of a single update step).

replace the singular values of $\mathbf{T}$ that are below some threshold (e.g., smaller than 0.5) by 1:

$$\mathbf{T}' = \mathrm{recond}(\mathbf{T}) := \sum \sigma_j' \mathbf{u}_j \mathbf{v}_j^{\mathrm{H}} \qquad \text{where} \quad \sigma_j' = \begin{cases} 1, & \sigma_j < 0.5 \\ \sigma_j, & \sigma_j \geq 0.5 \end{cases} \tag{6.19}$$

Experience by simulations shows that (i) this reconditioning step is very effective, and (ii) with good SNR it is rarely needed, even if sources are closely spaced. The reasons for this have to do with the good conditioning of the problem after prewhitening: the columns of the desired solution are almost orthogonal.

A similar algorithm was proposed more recently, called the Multi-User Kurtosis Algorithm (MUK) [6]. MUK is not specifically targeted for CM signals, but aims to separate statistically independent non-Gaussian signals by maximizing the absolute value of the kurtosis $K(y)$ of the output, where $K(y) = \mathrm{E}(|y|^4) - 2[\mathrm{E}(|y|^2)]^2$. This is a Shalvi-Weinstein cost function [22] (in this seminal paper, criteria were introduced which involve only the computation of the second- and fourth-order moments and do not pose restrictions on the probability distribution of the input sequence). For sources with a negative kurtosis, this leads to [6]

$$\mathbf{T}_{k+1} = \mathbf{T}_k - \mu\underline{\mathbf{x}}_k\mathbf{z}_k^{\mathrm{H}}\,, \qquad \mathbf{z}_k = \mathbf{y}_k \odot \bar{\mathbf{y}}_k \odot \mathbf{y}_k\,.$$

```
Given data $\mathbf{X}$, compute beamformers $\mathbf{W}$ and output $\mathbf{S} = \mathbf{W}^{\mathrm{H}}\mathbf{X}$:
  1. SVD: $\mathbf{X} =: \hat{\mathbf{U}}\hat{\mathbf{\Sigma}}\hat{\mathbf{V}}$                                   $(M^2N)$
     Prewhitening filter: $\mathbf{F} = \hat{\mathbf{U}}_s\hat{\mathbf{\Sigma}}_s^{-1}/\sqrt{N}$
     Prefiltering: $\underline{\mathbf{X}} := \mathbf{F}^{\mathrm{H}}\mathbf{X} = \hat{\mathbf{V}}_s\sqrt{N}$
     Data matrix inverse: $\underline{\mathbf{X}}^{\dagger} = \hat{\mathbf{V}}_s^{\mathrm{H}}/\sqrt{N}$
     Initialize: $\mathbf{T} = \mathbf{I}_{d\times d}$
  2. Iterate until convergence:
        $\mathbf{S} = \mathbf{T}^{\mathrm{H}}\underline{\mathbf{X}}$                                          $(d^2N)$
        $\mathbf{S} = [\frac{s_{ij}}{|s_{ij}|}]_{ij}$                                       $(dN)$
        $\mathbf{T} = (\mathbf{S}\underline{\mathbf{X}}^{\dagger})^{\mathrm{H}}$                                       $(d^2N)$
        $\mathbf{T} = \mathrm{recond}(\mathbf{T})$,   see equation (6.19)               $(d^3)$
  3. $\mathbf{W} = \mathbf{F}\mathbf{T}$
```

Fig. 6.6 LS-CMA in the whitened domain.

In [6], a condition that $\mathbf{T}$ should be orthogonal (rather than well-conditioned) is maintained. This can be formulated as an orthogonal Procrustes problem,

$$\mathbf{T}' = \arg\min_{\mathbf{T}'^{\mathrm{H}}\mathbf{T}'=\mathbf{I}} \|\mathbf{T} - \mathbf{T}'\|_{\mathrm{F}}^2$$

of which the solution is [23]

$$\mathbf{T} =: \sum_j \sigma_j \mathbf{u}_j \mathbf{v}_j^{\mathrm{H}} \qquad \Rightarrow \qquad \mathbf{T}' = \mathrm{reorth}(\mathbf{T}) := \sum_j \mathbf{u}_j \mathbf{v}_j^{\mathrm{H}}\,. \tag{6.20}$$

In [6], this solution is approximated by a QR factorization (this implements a sequential reorthogonalization where each column of $\mathbf{T}'$ is made orthogonal to the preceding columns). This does not optimize (6.20) and therefore may be less effective, but is computationally simpler. Alternatively, orthogonality of weight vectors may be enforced during adaptive processing by perturbation analysis of small rotations, as illustrated, e.g., in [24] and in section 6.6.

Both algorithms are summarized in figure 6.5.

6.4.2 Least Squares CMA

In the context of block-iterative algorithms, many other solutions are possible. We may formulate the multi-user constant modulus problem as a Least Squares problem

$$\hat{\mathbf{S}} = \arg\min_{\mathbf{S},\mathbf{W}} \|\mathbf{W}^{\mathrm{H}}\mathbf{X} - \mathbf{S}\|^2\,, \qquad \mathbf{S} \in \mathcal{CM}, \mathbf{W} \text{ full rank}$$

where $\mathcal{CM}$ denotes the set of constant modulus signals,

$$\mathcal{CM} = \{\, \mathbf{S} \,\big|\, |s_{ij}| = 1\,,\ \text{all } i,j \,\}\,.$$

With an initial value for $\mathbf{W}$, we can set up an Alternating Least Squares procedure as before: (*i*) find the best matching $\mathbf{S} \in \mathcal{CM}$, i.e., project $\mathbf{W}^{\mathrm{H}}\mathbf{X}$ onto the set of CM signals, (*ii*) find the best matching $\mathbf{W}$, i.e., $\mathbf{W}^{\mathrm{H}} = \mathbf{S}\mathbf{X}^{\dagger}$, Note that $\mathbf{W}^{\mathrm{H}}\mathbf{X} = \mathbf{S}\mathbf{X}^{\dagger}\mathbf{X}$, so that the algorithm alternates between projecting $\mathbf{S}$ onto the row span of $\mathbf{X}$, and projecting it onto the set $\mathcal{CM}$. The latter projection is simply a scaling of each entry of $\mathbf{S}$ to unit-norm:

$$\hat{\mathbf{S}} = \left[\frac{s_{ij}}{|s_{ij}|}\right]_{ij}\,.$$

This algorithm was derived from the CMA by Agee [25], and called the LS-CMA. A similar algorithm is well-known in the field of optics for solving the phase-retrieval problem, where it is called the Gerchberg-Saxton algorithm (GSA) [26]. The relation was pointed out in [27].

As stated in section 6.2, CMAs have significant problems in finding all independent sources. E.g., LS-CMA has no capability to ensure that $\mathbf{W}$ has full rank. As in the preceding section, this problem can be largely avoided by working in the whitened domain, for which $\mathbf{T}$ is nearly orthogonal. In the whitened domain, after $\mathbf{T}$ is computed, we need to verify its conditioning and possibly improve it, as in equation (6.19). This leads to the algorithm stated in figure 6.6. Although as a block algorithm it does not need many samples, the convergence can be slow: typically 20 iterations or more are needed, especially for small N. The reconditioning escape is typically needed only about once during the iterations. A performance simulation will follow later in section 6.5.5.

6.5 THE ANALYTICAL CMA[4]

In the preceding sections, we have discussed some of the adaptive and iterative CMAs that have appeared in the literature. They have been derived as solutions of certain optimization problems, e.g., the CMA(2,2) or CMA(1,2) cost function. It is interesting to note that the CMA(2,2) also admits an approximate solution that can be computed in closed form. In fact, given a block of $N > d^2$ samples, the complete collection of beamformers for all sources can be computed as the solution of a generalized eigenvalue problem. The algorithm is called the Analytical CMA (ACMA) [4].

[4]The material in sections 6.5.1–6.5.3 was presented in similar form in [28, 29].

6.5.1 Reformulation of the CMA(2,2) optimization problem

The CMA(2,2) cost function (6.7) was expressed in a stochastic framework. Given a finite batch of N data samples, it is more convenient to pose the cost function as a Least Squares problem:

$$\mathbf{w} = \arg\min_{\mathbf{w}} \frac{1}{N}\sum_k (|\hat{s}_k|^2 - 1)^2\,, \qquad \hat{s}_k = \mathbf{w}^{\mathrm{H}}\mathbf{x}_k\,. \tag{6.21}$$

The solution of this problem coincides with that of (6.7) as $N \to \infty$.

Using the Kronecker product properties (6.3) and (6.5), we can write $|\hat{s}_k|^2$ as

$$|\hat{s}_k|^2 = \mathbf{w}^{\mathrm{H}}(\mathbf{x}_k\mathbf{x}_k^{\mathrm{H}})\mathbf{w} = (\bar{\mathbf{x}}_k \otimes \mathbf{x}_k)^{\mathrm{H}}(\bar{\mathbf{w}} \otimes \mathbf{w})\,.$$

We subsequently stack the rows $(\bar{\mathbf{x}}_k \otimes \mathbf{x}_k)^{\mathrm{H}}$ of the data into a matrix $\mathbf{P}$ (size $N \times M^2$). Referring to the definition of the Khatri-Rao product in (6.2), we see that $\mathbf{P} = [\bar{\mathbf{X}} \circ \mathbf{X}]^{\mathrm{H}}$. Also introducing $\mathbf{y} = \bar{\mathbf{w}} \otimes \mathbf{w}$ and $\mathbf{1} = [1, \cdots, 1]^{\mathrm{T}}$, we can write (6.21) as

$$\begin{aligned} \frac{1}{N}\sum_k (|\hat{s}_k|^2 - 1)^2 &= \frac{1}{N}\sum_k \left[(\bar{\mathbf{x}}_k \otimes \mathbf{x}_k)^{\mathrm{H}}(\bar{\mathbf{w}} \otimes \mathbf{w}) - 1\right]^2 \\ &= \frac{1}{N}\|\, \mathbf{P}\mathbf{y} - \mathbf{1}\,\|^2\,. \end{aligned} \tag{6.22}$$

Thus, the CMA(2,2) optimization problem asks for the Least Squares solution of a linear system of equations, subject to a quadratic constraint, namely $\mathbf{y} = \bar{\mathbf{w}} \otimes \mathbf{w}$.

The linear system can be conveniently rewritten as follows. Let $\mathbf{Q}$ be any unitary matrix such that $\mathbf{Q}\mathbf{1} = \sqrt{N}\left[\begin{smallmatrix}1\\ \mathbf{0}\end{smallmatrix}\right]$, for example found by computing a QR factorization of $[\mathbf{1}\ \ \mathbf{P}]$. Apply $\mathbf{Q}$ to $[\mathbf{1}\ \ \mathbf{P}]$ and partition the result as

$$\mathbf{Q}[\mathbf{1}\ \ \mathbf{P}] =: \sqrt{N}\begin{bmatrix} 1 & \hat{\mathbf{p}}^{\mathrm{H}} \\ \mathbf{0} & \mathbf{G} \end{bmatrix}. \tag{6.23}$$

Then

$$\mathbf{P}\mathbf{y} = \mathbf{1} \quad \Leftrightarrow \quad \mathbf{Q}[\mathbf{1}\ \ \mathbf{P}]\begin{bmatrix} -1 \\ \mathbf{y} \end{bmatrix} = \mathbf{0} \quad \Leftrightarrow \quad \left\{ \begin{array}{lcl} \hat{\mathbf{p}}^{\mathrm{H}}\mathbf{y} & = & 1 \\ \mathbf{G}\mathbf{y} & = & \mathbf{0} \end{array} \right. \tag{6.24}$$

and therefore (6.22) can be written as

$$\frac{1}{N}\sum_k (|\hat{s}_k|^2 - 1)^2 = |\,\hat{\mathbf{p}}^{\mathrm{H}}\mathbf{y} - 1\,|^2 + \|\,\mathbf{G}\mathbf{y}\,\|^2\,. \tag{6.25}$$

The second term of this expression requires $\mathbf{y} = \bar{\mathbf{w}} \otimes \mathbf{w}$ to be in the nullspace of matrix $\mathbf{G}$, as much as possible. The first term puts a constraint which avoids the trivial solution $\mathbf{y} = \mathbf{0}$. By squaring (6.23) to eliminate $\mathbf{Q}$, we obtain explicit expressions for $\hat{\mathbf{p}}$ and a matrix $\hat{\mathbf{C}} := \mathbf{G}^{\mathrm{H}}\mathbf{G}$:

$$\begin{aligned}
\hat{\mathbf{p}} &= \tfrac{1}{N}\mathbf{P}^{\mathrm{H}}\mathbf{1} = \tfrac{1}{N}\textstyle\sum_k \bar{\mathbf{x}}_k \otimes \mathbf{x}_k && (6.26)\\
\hat{\mathbf{C}} &:= \mathbf{G}^{\mathrm{H}}\mathbf{G} = \tfrac{1}{N}\mathbf{P}^{\mathrm{H}}\mathbf{P} - \tfrac{1}{N}\mathbf{P}^{\mathrm{H}}\mathbf{1}\cdot\tfrac{1}{N}\mathbf{1}^{\mathrm{H}}\mathbf{P} && (6.27)\\
&= \tfrac{1}{N}\textstyle\sum_k (\bar{\mathbf{x}}_k\otimes\mathbf{x}_k)(\bar{\mathbf{x}}_k\otimes\mathbf{x}_k)^{\mathrm{H}} - \left[\tfrac{1}{N}\sum_k \bar{\mathbf{x}}_k\otimes\mathbf{x}_k\right]\left[\tfrac{1}{N}\sum_k \bar{\mathbf{x}}_k\otimes\mathbf{x}_k\right]^{\mathrm{H}}.
\end{aligned}$$

The former expression shows that $\hat{\mathbf{p}} = \frac{1}{N}\sum \bar{\mathbf{x}}_k \otimes \mathbf{x}_k = \mathrm{vec}(\frac{1}{N}\sum \mathbf{x}_k\mathbf{x}_k^{\mathrm{H}}) = \mathrm{vec}(\hat{\mathbf{R}}_{\mathbf{x}})$, where $\hat{\mathbf{R}}_{\mathbf{x}}$ is the sample covariance matrix of the data. Thus, (for $\mathbf{y} = \bar{\mathbf{w}}\otimes\mathbf{w}$)

$$\hat{\mathbf{p}}^{\mathrm{H}}\mathbf{y} = \mathbf{w}^{\mathrm{H}}\hat{\mathbf{R}}_{\mathbf{x}}\mathbf{w} \qquad (6.28)$$

and we see that the condition $\hat{\mathbf{p}}^{\mathrm{H}}\mathbf{y} = 1$ in (6.24) is equal to requiring that the average output power of the beamformer is 1.

Returning to (6.25), let $\hat{\mathbf{y}}$ be the (structured) minimizer of this expression, and define $\beta = \hat{\mathbf{p}}^{\mathrm{H}}\hat{\mathbf{y}}$. Equation (6.28) shows that it is the output power of the beamformer corresponding to $\hat{\mathbf{y}}$ and hence $\beta > 0$. Regarding β as some known fixed constant, we can add a condition that $\hat{\mathbf{p}}^{\mathrm{H}}\mathbf{y} = \beta$ to the optimization problem without changing the outcome:

$$\begin{aligned}
\hat{\mathbf{y}} &= \underset{\substack{\mathbf{y}=\bar{\mathbf{w}}\otimes\mathbf{w}\\ \hat{\mathbf{p}}^{\mathrm{H}}\mathbf{y}=\beta}}{\arg\min}\ |\hat{\mathbf{p}}^{\mathrm{H}}\mathbf{y}-1|^2 + \|\,\mathbf{G}\mathbf{y}\,\|^2 = \underset{\substack{\mathbf{y}=\bar{\mathbf{w}}\otimes\mathbf{w}\\ \hat{\mathbf{p}}^{\mathrm{H}}\mathbf{y}=\beta}}{\arg\min}\ |\beta-1|^2 + \|\,\mathbf{G}\mathbf{y}\,\|^2\\
&= \underset{\substack{\mathbf{y}=\bar{\mathbf{w}}\otimes\mathbf{w}\\ \hat{\mathbf{p}}^{\mathrm{H}}\mathbf{y}=\beta}}{\arg\min}\ \|\,\mathbf{G}\mathbf{y}\,\|^2 .
\end{aligned}$$

Since β is real and positive, replacing β by 1 will only scale the solution $\hat{\mathbf{y}}$ to $\frac{1}{\beta}\hat{\mathbf{y}}$, and does not affect the fact that it has a Kronecker structure. Therefore, it is possible to drop the first term in the optimization problem (6.25) and replace it by a constraint $\hat{\mathbf{p}}^{\mathrm{H}}\mathbf{y} = 1$.

This constraint in turn motivates in a natural way the choice of a prewhitening filter $\mathbf{F} = \hat{\mathbf{U}}_s\hat{\boldsymbol{\Sigma}}_s^{-1}$ as given in (6.15). Indeed, we derived in (6.28) that $\hat{\mathbf{p}}^{\mathrm{H}}\mathbf{y} = \mathbf{w}^{\mathrm{H}}\hat{\mathbf{R}}_{\mathbf{x}}\mathbf{w}$. If we change variables to $\underline{\mathbf{x}} = \mathbf{F}^{\mathrm{H}}\mathbf{x}$ and $\mathbf{w} = \mathbf{F}\mathbf{t}$, then $\underline{\hat{\mathbf{R}}}_{\mathbf{x}} = \mathbf{I}$ and

$$\hat{\mathbf{p}}^{\mathrm{H}}\mathbf{y} = \mathbf{w}^{\mathrm{H}}\hat{\mathbf{R}}_{\mathbf{x}}\mathbf{w} = \mathbf{t}^{\mathrm{H}}\mathbf{t} = \|\mathbf{t}\|^2 .$$

Moreover, $(\bar{\mathbf{t}}\otimes\mathbf{t})^{\mathrm{H}}(\bar{\mathbf{t}}\otimes\mathbf{t}) = \bar{\mathbf{t}}^{\mathrm{H}}\bar{\mathbf{t}}\otimes\mathbf{t}^{\mathrm{H}}\mathbf{t} = \|\mathbf{t}\|^4$. It thus follows that $\hat{\mathbf{p}}^{\mathrm{H}}\mathbf{y} = 1 \Leftrightarrow \|\,\bar{\mathbf{t}}\otimes\mathbf{t}\,\| = 1$. Hence, up to a scaling which is not important, the CMA(2,2) optimization problem is equivalent to solving

$$\mathbf{t} = \underset{\substack{\mathbf{y}=\bar{\mathbf{t}}\otimes\mathbf{t}\\ \|\mathbf{y}\|=1}}{\arg\min}\ \|\,\underline{\mathbf{G}\mathbf{y}}\,\|^2 = \underset{\substack{\mathbf{y}=\bar{\mathbf{t}}\otimes\mathbf{t}\\ \|\mathbf{y}\|=1}}{\arg\min}\ \mathbf{y}^{\mathrm{H}}\underline{\hat{\mathbf{C}}}\mathbf{y} \qquad (6.29)$$

and setting $\mathbf{w} = \mathbf{F}\mathbf{t}$.

As discussed in section 6.3, it is important to also incorporate a dimension reduction in the prewhitening filter. Without dimension reduction, the nullspace of $\mathbf{G}$ and $\mathbf{C}$ contain additional vectors which do not correspond to valid solutions. The prefilter $\mathbf{F}$ reduces the dimension of $\mathbf{x}_k$ from M to d, and the corresponding matrix $\underline{\mathbf{P}} := [\underline{\bar{\mathbf{X}}} \circ \underline{\mathbf{X}}]^{\mathrm{H}}$ has size $N \times d^2$. We assume that $N > d^2$ so that $\underline{\mathbf{P}}$ is 'tall', and $\underline{\mathbf{G}}$ and $\underline{\hat{\mathbf{C}}}$ have a well-defined (i.e., overdetermined) nullspace. It can be shown [30] that for sufficiently large N (at least $N > d^2$) the nullspace has dimension d and does not contain other solutions.

6.5.2 Solving the CMA(2,2) optimization problem

To solve the CMA(2,2) optimization problem in (6.29), it is required to numerically optimize the minimization problem and find d independent solutions. The solutions will be unit-norm vectors $\mathbf{y}$ that have the required Kronecker structure and minimize $\| \underline{\mathbf{G}}\mathbf{y} \|^2$. With noise, the solutions will not exactly be in the approximate nullspace of $\underline{\mathbf{G}}$ since in general this space will not contain vectors with the Kronecker structure.

Instead of solving this problem, an alternative approach is to first find an orthonormal basis $\mathbf{Y} = [\mathbf{y}_1, \cdots, \mathbf{y}_d]$ for the the d-dimensional approximate nullspace of $\underline{\mathbf{G}}$ (or $\underline{\hat{\mathbf{C}}}$),

$$\mathbf{Y} = \arg\min_{\mathbf{Y}^{\mathrm{H}}\mathbf{Y}=\mathbf{I}} \| \underline{\mathbf{G}}\mathbf{Y} \|_{\mathrm{F}}^2 = \arg\min_{\mathbf{Y}^{\mathrm{H}}\mathbf{Y}=\mathbf{I}} \sum \mathbf{y}_i^{\mathrm{H}} \underline{\hat{\mathbf{C}}} \mathbf{y}_i \,, \tag{6.30}$$

whose solution is the set of d least dominant eigenvectors of $\underline{\hat{\mathbf{C}}}$. We can subsequently look for a set of d unit-norm vectors $\bar{\mathbf{t}}_i \otimes \mathbf{t}_i$ that best spans the same subspace,

$$\mathbf{T} = \arg\min_{\mathbf{T},\mathbf{\Lambda}} \| \mathbf{Y} - (\bar{\mathbf{T}} \circ \mathbf{T})\mathbf{\Lambda} \|_{\mathrm{F}}^2 \,.$$

where $\mathbf{T} = [\mathbf{t}_1, \cdots, \mathbf{t}_d]$ is the set of beamformers in the whitened domain, $\bar{\mathbf{T}} \circ \mathbf{T} := [\bar{\mathbf{t}}_1 \otimes \mathbf{t}_1, \cdots, \bar{\mathbf{t}}_d \otimes \mathbf{t}_d]$ denotes the Khatri-Rao product, and $\mathbf{\Lambda}$ is a full rank $d \times d$ matrix that relates the two bases of the subspace.

Alternatively, we can formulate this as a joint diagonalization problem, since (using (6.3)–(6.6))

$$\| \mathbf{Y} - (\bar{\mathbf{T}} \circ \mathbf{T})\mathbf{\Lambda} \|_{\mathrm{F}}^2 \;=\; \sum_i \| \mathbf{y}_i - (\bar{\mathbf{T}} \circ \mathbf{T})\boldsymbol{\lambda}_i \|_{\mathrm{F}}^2 \;=\; \sum_i \| \mathbf{Y}_i - \mathbf{T}\mathbf{\Lambda}_i\mathbf{T}^{\mathrm{H}} \|_{\mathrm{F}}^2$$

where $\boldsymbol{\lambda}_i$ is the i-th column of $\mathbf{\Lambda}$, $\mathbf{\Lambda}_i = \mathrm{diag}(\boldsymbol{\lambda}_i)$ is a diagonal matrix constructed from this vector, and $\mathbf{Y}_i$ is a matrix obtained by unstacking $\mathbf{y}_i$ into a square $d \times d$ matrix such that $\mathrm{vec}(\mathbf{Y}_i) = \mathbf{y}_i$; we have also used (6.6). The latter equation shows that all $\mathbf{Y}_i$ can be diagonalized by the same matrix $\mathbf{T}$. The resulting joint diagonalization problem is a generalization of the standard eigenvalue decomposition problem, and can be solved iteratively (see section 6.5.4 ahead).

The preceding two-step approach gives an approximate but closed-form solution to (6.29); the corresponding algorithm is called the Analytical CMA

```
Given data $\mathbf{X}$, compute beamformers $\mathbf{W}$ and output $\hat{\mathbf{S}} = \mathbf{W}^{\mathrm{H}}\mathbf{X}$:

1. SVD: $\mathbf{X} =: \hat{\mathbf{U}}\hat{\mathbf{\Sigma}}\hat{\mathbf{V}}$                                              $(M^2N)$
   Prefiltering: $\underline{\mathbf{X}} = \hat{\mathbf{\Sigma}}_s^{-1}\hat{\mathbf{U}}_s^{\mathrm{H}}\mathbf{X} = \hat{\mathbf{V}}_s$
   $\hat{\underline{\mathbf{C}}} = \frac{1}{N}\sum(\bar{\underline{\mathbf{x}}}_k \otimes \underline{\mathbf{x}}_k)(\bar{\underline{\mathbf{x}}}_k \otimes \underline{\mathbf{x}}_k)^{\mathrm{H}} - [\frac{1}{N}\sum\bar{\underline{\mathbf{x}}}_k \otimes \underline{\mathbf{x}}_k][\frac{1}{N}\sum\bar{\underline{\mathbf{x}}}_k \otimes \underline{\mathbf{x}}_k]^{\mathrm{H}}$   $(d^4N)$
   EVD of $\hat{\underline{\mathbf{C}}}$, let $\{\mathbf{y}_i\}$ be the $d$ least dominant eigenvectors.   $(\le d^6)$

2. $\mathbf{Y}_i = \mathrm{vec}^{-1}\mathbf{y}_i \;\; (i = 1, \cdots, d)$
   Find $\mathbf{T}$ to jointly diagonalize $\{\mathbf{Y}_i\}$ as $\mathbf{Y}_i = \mathbf{T}\mathbf{\Lambda}_i\mathbf{T}^{\mathrm{H}} \; (i = 1, \cdots, d)$   $(d^4)$

3. Scale each column of $\mathbf{T}$ to norm 1.
   Set $\mathbf{W} = \hat{\mathbf{U}}_s\hat{\mathbf{\Sigma}}_s^{-1}\mathbf{T}$ and $\hat{\mathbf{S}} = \mathbf{T}^{\mathrm{H}}\underline{\mathbf{X}}$   $(d^2N)$
```

Fig. 6.7 Summary of ACMA.

(ACMA) [4]. An important advantage of this algorithm is that it provides the complete set of beamformers as the solution of the joint diagonalization problem. A second advantage is that in the noise-free case and with $N \geq d^2$, the algorithm produces the exact solution $\mathbf{W} = \mathbf{A}^{\dagger\mathrm{H}}$. The algorithm is summarized in figure 6.7. The main complexity is in the construction of $\hat{\underline{\mathbf{C}}}$.

6.5.3 Asymptotic analysis of ACMA

A detailed performance analysis of ACMA has appeared in [29, 31]. The most important finding is that asymptotically, the solution converges to the Wiener beamformer, for a large number of samples or for a high SNR. The derivation is summarized in this subsection.

Recall the definition of $\hat{\underline{\mathbf{C}}}$ in equation (6.27). To analyze the nullspace of $\hat{\underline{\mathbf{C}}}$, we consider a large number of samples, so that $\hat{\underline{\mathbf{C}}}$ converges to $\underline{\mathbf{C}}$,

$$\underline{\mathbf{C}} = \mathrm{E}(\bar{\underline{\mathbf{x}}} \otimes \underline{\mathbf{x}})(\bar{\underline{\mathbf{x}}} \otimes \underline{\mathbf{x}})^{\mathrm{H}} \; - \; \mathrm{E}\,[\bar{\underline{\mathbf{x}}} \otimes \underline{\mathbf{x}}]\,\mathrm{E}\,[\bar{\underline{\mathbf{x}}} \otimes \underline{\mathbf{x}}]^{\mathrm{H}} \, . \tag{6.31}$$

where $\underline{\mathbf{x}}_k = \underline{\mathbf{A}}\mathbf{s}_k + \underline{\mathbf{n}}_k$. Thus, $\underline{\mathbf{C}}$ involves both 4th order and 2nd order statistics of the data.

In general, for a zero mean random vector $\mathbf{x} = [x_i]$, the fourth order cumulant matrix is defined as

$$\mathbf{K}_{\mathbf{x}} = \sum_{abcd} (\mathbf{i}_b \otimes \mathbf{i}_a)(\mathbf{i}_c \otimes \mathbf{i}_d)^{\mathrm{H}} \, \mathrm{cum}(x_a, \bar{x}_b, x_c, \bar{x}_d) \, , \tag{6.32}$$

where $\mathbf{i}_i$ is the ith column of the identity matrix $\mathbf{I}$, and, for circularly symmetric variables, the cumulant function is defined by

$$\mathrm{cum}(x_a, \bar{x}_b, x_c, \bar{x}_d) = \mathrm{E}(x_a\bar{x}_b x_c\bar{x}_d) - \mathrm{E}(x_a\bar{x}_b)\mathrm{E}(x_c\bar{x}_d) - \mathrm{E}(x_a\bar{x}_d)\mathrm{E}(\bar{x}_b x_c) \, .$$

Therefore, (6.32) can be written compactly as

$$\mathbf{K}_{\mathbf{x}} = \mathrm{E}(\bar{\mathbf{x}} \otimes \mathbf{x})(\bar{\mathbf{x}} \otimes \mathbf{x})^{\mathrm{H}} - \mathrm{E}(\bar{\mathbf{x}} \otimes \mathbf{x})\mathrm{E}(\bar{\mathbf{x}} \otimes \mathbf{x})^{\mathrm{H}} - \mathrm{E}(\bar{\mathbf{x}}\bar{\mathbf{x}}^{\mathrm{H}}) \otimes \mathrm{E}(\mathbf{x}\mathbf{x}^{\mathrm{H}}) \qquad (6.33)$$

Cumulants have several important properties: for sums of independent sources they are additive, the 4th order cumulant of Gaussian sources is $\mathbf{K}_{\mathbf{n}} = 0$, the 4th order cumulant of independent CM sources is $\mathbf{K}_{\mathbf{s}} = -\mathbf{I}$. For the data model $\underline{\mathbf{x}}_k = \underline{\mathbf{A}}\mathbf{s}_k + \underline{\mathbf{n}}_k$, it thus follows that

$$\mathbf{K}_{\underline{\mathbf{x}}} = [\underline{\bar{\mathbf{A}}} \otimes \underline{\mathbf{A}}](-\mathbf{I})[\underline{\bar{\mathbf{A}}} \otimes \underline{\mathbf{A}}]^{\mathrm{H}}$$

and hence by combining (6.31) with (6.33), $\underline{\mathbf{C}}$ is given by[5]

$$\begin{aligned} \underline{\mathbf{C}} &= \mathbf{K}_{\underline{\mathbf{x}}} + \mathrm{E}(\underline{\bar{\mathbf{x}}}\underline{\bar{\mathbf{x}}}^{\mathrm{H}}) \otimes \mathrm{E}(\underline{\mathbf{x}}\underline{\mathbf{x}}^{\mathrm{H}}) \\ &= [\underline{\bar{\mathbf{A}}} \circ \underline{\mathbf{A}}](-\mathbf{I})[\underline{\bar{\mathbf{A}}} \circ \underline{\mathbf{A}}]^{\mathrm{H}} + \mathbf{I}\,. \end{aligned} \qquad (6.34)$$

where we also used that after prewhitening $\mathrm{E}(\underline{\mathbf{x}}\underline{\mathbf{x}}^{\mathrm{H}}) = \mathbf{I}$. Consequently, the CMA(2,2) cost function (6.29) becomes asymptotically

$$\begin{aligned} \underset{\mathbf{y}=\bar{\mathbf{t}}\otimes\mathbf{t},\|\mathbf{y}\|=1}{\arg\min}\ \mathbf{y}^{\mathrm{H}}\underline{\mathbf{C}}\mathbf{y} &= \underset{\mathbf{y}=\bar{\mathbf{t}}\otimes\mathbf{t},\|\mathbf{y}\|=1}{\arg\min}\ \mathbf{y}^{\mathrm{H}}\left\{[\underline{\bar{\mathbf{A}}} \circ \underline{\mathbf{A}}](-\mathbf{I})[\underline{\bar{\mathbf{A}}} \circ \underline{\mathbf{A}}]^{\mathrm{H}} + \mathbf{I}\right\}\mathbf{y} \\ &= \underset{\mathbf{y}=\bar{\mathbf{t}}\otimes\mathbf{t},\|\mathbf{y}\|=1}{\arg\max}\ \mathbf{y}^{\mathrm{H}}\left\{[\underline{\bar{\mathbf{A}}} \circ \underline{\mathbf{A}}][\underline{\bar{\mathbf{A}}} \circ \underline{\mathbf{A}}]^{\mathrm{H}}\right\}\mathbf{y}\,. \end{aligned} \qquad (6.35)$$

ACMA first constructs a basis $\{\mathbf{y}_1, \cdots, \mathbf{y}_d\}$ for the d nullspace vectors of $\underline{\mathbf{C}}$ without the constraint $\mathbf{y} = \bar{\mathbf{t}} \otimes \mathbf{t}$, which clearly are given by

$$\mathrm{span}\{\mathbf{y}_1, \cdots, \mathbf{y}_d\} = \mathrm{span}[\underline{\bar{\mathbf{A}}} \circ \underline{\mathbf{A}}] = \mathrm{span}\{\underline{\bar{\mathbf{a}}}_1 \otimes \underline{\mathbf{a}}_1, \cdots, \underline{\bar{\mathbf{a}}}_d \otimes \underline{\mathbf{a}}_d\}\,.$$

As a second step, the joint diagonalization procedure is used to replace the unstructured basis by one that has the required Kronecker product structure, i.e., d independent vectors of the form $\bar{\mathbf{t}} \otimes \mathbf{t}$ within this column span. From the above equation, we see that the unique solution is $\bar{\mathbf{t}}_i \otimes \mathbf{t}_i = \underline{\bar{\mathbf{a}}}_i \otimes \underline{\mathbf{a}}_i$ (up to a scaling to make $\mathbf{t}_i$ have unit norm), and thus

$$\mathbf{t}_i = \underline{\mathbf{a}}_i\,, \qquad i = 1, \cdots, d\,.$$

Similar to section 6.3.3, the solution in the original (unwhitened) domain is $\mathbf{w}_i = \mathbf{R}_{\mathbf{x}}^{-1}\mathbf{a}_i$. We have just shown that as $N \to \infty$, the beamformers provided by ACMA converge to the Wiener receivers (6.16). This is partly due to the choice in prewhitening scheme, and partly due to the two-step solution of the CMA(2,2) problem (6.29). In general, the exact solution of CMA(2,2) does not lead to the Wiener solution, although it has already been observed by

[5]This analysis is not valid for BPSK sources, because they are not circularly symmetric. For such sources, the ACMA has to be modified [32].

Godard that that the solutions are often close. Quantitative evidence for this is given in [33, 34, 35].

6.5.4 Joint diagonalization via iterative projection

Since the problem of joint diagonalization shows up in several other signal processing applications, it has actually been well studied over the past decade [36, 37, 38, 39, 5, 40, 41, 42, 43, 44, 4, 45, 46]. The algorithms are iterative, e.g., using extensions of Jacobi iterations or QZ iterations. Generally the performance of these algorithms is satisfactory although the speed of convergence is often linear, and without an exact solution (i.e., in the noisy case) the convergence to local minima cannot be avoided. The reason for good performance is that a good starting point is available from the solution for only two matrices, which reduces to a standard eigenvalue problem.

The joint diagonalization step in the ACMA can be written as

$$\mathbf{T} = \arg\min_{\|\mathbf{t}_i\|=1} \| \mathbf{V}_n - (\bar{\mathbf{T}} \circ \mathbf{T})\boldsymbol{\Lambda} \|_{\mathrm{F}}^2 \qquad (6.36)$$

where $\mathbf{V}_n$ is an orthogonal basis for the nullspace of $\mathbf{G}$. The unit-norm constraint on the columns of $\mathbf{T}$ avoids a scaling indeterminacy between $\mathbf{T}$ and $\boldsymbol{\Lambda}$. Since $\boldsymbol{\Lambda}$ is invertible, the problem can also be written as

$$\mathbf{T} = \arg\min_{\substack{\|\mathbf{t}_i\| = 1 \\ \mathbf{M}\text{ invertible}}} \| \mathbf{V}_n\mathbf{M} - (\bar{\mathbf{T}} \circ \mathbf{T}) \|_{\mathrm{F}}^2 \, . \qquad (6.37)$$

This is not entirely equivalent, but almost so since $\boldsymbol{\Lambda}$ is close to unitary. However, a condition that $\mathbf{M}$ is invertible is needed to avoid finding repeated solutions $\mathbf{t}_i = \mathbf{t}_j$.

As inspired by [44], the problem (6.37) can be solved iteratively using Alternating Least Squares: (*i*) given an initial value for $\mathbf{T}$, solve for $\mathbf{M}$; (*ii*) for the new value of $\mathbf{M}$, solve for $\mathbf{T}$. Both steps are simple to implement: for fixed $\mathbf{T}$, solving for $\mathbf{M}$ gives

$$\mathbf{M} = \mathbf{V}_n^{\mathrm{H}}(\bar{\mathbf{T}} \circ \mathbf{T}) \, ,$$

whereas for fixed $\mathbf{M}$, solving for $\mathbf{T}$ gives

$$\mathbf{T} = \arg\min_{\|\mathbf{t}_i\|=1} \|\mathbf{V}_n\mathbf{M} - (\bar{\mathbf{T}} \circ \mathbf{T})\|^2 = \arg\min_{\|\mathbf{t}_i\|=1} \|\mathbf{Y} - (\bar{\mathbf{T}} \circ \mathbf{T})\|^2 \, ,$$

where $\mathbf{Y} = \mathbf{V}_n\mathbf{M} = \mathbf{V}_n\mathbf{V}_n^{\mathrm{H}}(\bar{\mathbf{T}} \circ \mathbf{T})$ is the projection of the previous solution onto the estimated subspace. The problem decouples into solving for the individual columns of $\mathbf{T}$:

$$\mathbf{t}_i = \arg\min_{\|\mathbf{t}_i\|=1} \|\mathbf{y}_i - (\bar{\mathbf{t}}_i \otimes \mathbf{t}_i)\|^2 = \arg\min_{\|\mathbf{t}_i\|=1} \|\mathbf{Y}_i - \mathbf{t}_i\mathbf{t}_i^{\mathrm{H}}\|^2 \, ,$$

```
Given data $\mathbf{X} = [\mathbf{x}_1, \cdots, \mathbf{x}_N]$, compute beamformer $\mathbf{W}$ and output $\mathbf{s}_k = \mathbf{W}_k^{\mathrm{H}}\mathbf{x}_k$:

1. Compute $\hat{\mathbf{R}}_{\mathbf{x}} = \frac{1}{N}\sum_k \mathbf{x}_k\mathbf{x}_k^{\mathrm{H}}$                                   $(M^2N)$
   Compute the EVD $\hat{\mathbf{R}}_{\mathbf{x}} = \mathbf{U}\mathbf{\Sigma}^2\mathbf{U}^{\mathrm{H}}$                                  $(M^3)$
   Set the prewhitening filter $\mathbf{F} = \hat{\mathbf{U}}_s\hat{\mathbf{\Sigma}}_s^{-1/2}$                              $(dM)$
2. Prefilter the data, $\underline{\mathbf{x}}_k = \mathbf{F}^{\mathrm{H}}\mathbf{x}_k$                                      $(dMN)$
   $\hat{\underline{\mathbf{C}}} = \frac{1}{N}\sum_k (\bar{\underline{\mathbf{x}}}_k \otimes \underline{\mathbf{x}}_k)(\bar{\underline{\mathbf{x}}}_k \otimes \underline{\mathbf{x}}_k)^{\mathrm{H}} - \mathrm{vec}(\hat{\underline{\mathbf{R}}}_{\mathbf{x}})\mathrm{vec}(\hat{\underline{\mathbf{R}}}_{\mathbf{x}})^{\mathrm{H}}$    $(d^4N)$
3. Compute $\mathbf{V}_n$, the nullspace of $\hat{\underline{\mathbf{C}}}$                                  $(\leq d^6)$
4. Initialize $\mathbf{T} = \mathbf{I}$
   Until convergence:
       $\mathbf{M} = \mathbf{V}_n^{\mathrm{H}}(\bar{\mathbf{T}} \circ \mathbf{T})$                                        $(d^4)$
       $\mathbf{M}' = \mathrm{recond}(\mathbf{M})$, equation (6.19)                             $(d^3)$
       $\mathbf{Y} = \mathbf{V}_n\mathbf{M}'$                                               $(d^4)$
       $\mathbf{t}_i = \pi_1(\mathbf{y}_i), \quad i = 1, \cdots, d$, equation (6.38)                $(d^4)$
5. Set $\mathbf{W} = \mathbf{F}\mathbf{T}$ and $\hat{\mathbf{s}}_k = \mathbf{W}^{\mathrm{H}}\mathbf{x}_k$                                   $(dMN)$
```

Fig. 6.8 ACMA using iterative implementation of the joint diagonalization.

where $\mathbf{Y}_i = \mathrm{vec}^{-1}(\mathbf{y}_i)$. The solution is given by an SVD of $\mathbf{Y}_i$ and retaining the dominant component, i.e.,

$$\mathbf{Y}_i =: \sum_j \sigma_j \mathbf{u}_j \mathbf{u}_j^{\mathrm{H}}, \qquad \mathbf{t}_i = \mathbf{u}_1 =: \pi_1(\mathbf{y}_i)\,. \tag{6.38}$$

We will denote this "projection onto rank-1" operation by $\mathbf{t}_i = \pi_1(\mathbf{y}_i)$.

The algorithm thus projects a prior solution $\bar{\mathbf{t}}_i \otimes \mathbf{t}_i$ onto the estimated subspace $\mathbf{V}_n$, and then "projects" the result back onto the Kronecker structure (the latter projection is non-linear). After each projection, the error is decreasing, therefore the algorithm converges monotonically to a local minimum. As with most alternating projection algorithms, the speed of convergence is only linear. Nonetheless, experience shows that in practice only a few iterations are needed (two or three): the minimum error depends on the noise level and is quickly reached.

The columns of $\mathbf{T}$ are processed independently. Therefore, there is a risk that they converge to the same solution. To avoid this situation, the independence of the solutions has to be monitored and corrected, if needed. The easiest point to do this is to compute an SVD of $\mathbf{M}$ and recondition it if needed (see equation (6.19)). The motivation for this is that, after prewhitening, $\mathbf{T}$ is nearly unitary, hence the columns of $\bar{\mathbf{T}} \circ \mathbf{T}$ are nearly orthogonal. Therefore, $\mathbf{M} = \mathbf{V}_n^{\mathrm{H}}(\bar{\mathbf{T}} \circ \mathbf{T})$ is expected to be nearly unitary: its singular values are close to 1. On the other hand, if two columns of $\mathbf{T}$ converge to the same solution, then a singular value of $\mathbf{M}$ gets close to 0.

The resulting iterative ACMA is summarized in figure 6.8. Also the computational complexity of each step is indicated. Most of the work is done in two steps: the SVD of $\mathbf{X}$ which has a complexity of M^2N, and the construction of $\underline{\hat{\mathbf{C}}}$, which has a complexity of d^4N. The joint diagonalization step has a complexity of order d^4 (independent of its implementation), which can be neglected. Therefore, the total complexity is about $(M^2 + d^4)N$.

The computational complexity of the LS-CMA (figure 6.6) is dominated by the two large inner products, $\mathbf{T}^{\mathrm{H}}\mathbf{X}$ and $\mathbf{S}\underline{\mathbf{X}}^{\dagger}$, each of complexity d^2N. The whitening step has a complexity of M^2N. With 25 iterations, the total complexity is about $(M^2+50d^2)N$. In comparison, the ACMA is more efficient than the LS-CMA algorithm if $d \leq 7$.

6.5.5 Comparison of ACMA to LS-CMA

Some performance results for the block-iterative methods are shown in figure 6.9. In the simulations, we took a uniform linear array with $M = 4$ antennas spaced at half wavelengths, and $d = 3$ constant-modulus sources with directions $[-10°, 20°, 30°]$ and amplitudes [1, .8, .9], respectively. (Recall that the algorithms do not use knowledge of the array structure—the above information just serves to specify the simulation data.) The noise power is determined by the Signal to Noise Ratio (SNR), which is the SNR per antenna for the strongest user.

We compare the performance of the prewhitened LS-CMA as in figure 6.6 (25 iterations), ACMA as in figure 6.7, and iterative ACMA as in figure 6.8 (2 or 3 iterations). For reference, we also show the performance of the sample data LMMSE receiver, $\hat{\mathbf{W}} = (\mathbf{S}\mathbf{X}^{\dagger})^{\mathrm{H}}$ with known $\mathbf{S}$. The performance measure is the residual signal to interference plus noise ratio (SINR) at the output of the beamformers. We only consider the SINR of the worst output channel. The graphs show the performance as a function of Signal to Noise Ratio (SNR), number of samples N, and angular separation between the 2nd and 3rd source, respectively.

It is seen in figure 6.9 that all algorithms converge to the LMMSE receiver for sufficiently large N or SNR and source separation. The panel at the right shows the fraction of times that not all independent beamformers were recovered (this was established by first verifying for each beamformer which signal it recovers most strongly; these "failures" were omitted from the SINR statistics). The algorithms have similar failure rate performance, which is very small for sufficiently large N, SNR and source separation.

Figure 6.10 shows the convergence of the CMA(2,2) cost function (averaged over all sources and over 800 monte-carlo runs), for various SNR levels. It is seen that the LS-CMA converges much slower, and for large SNR may never reach the minimum cost. Also note that LS-CMA does not aim to minimize

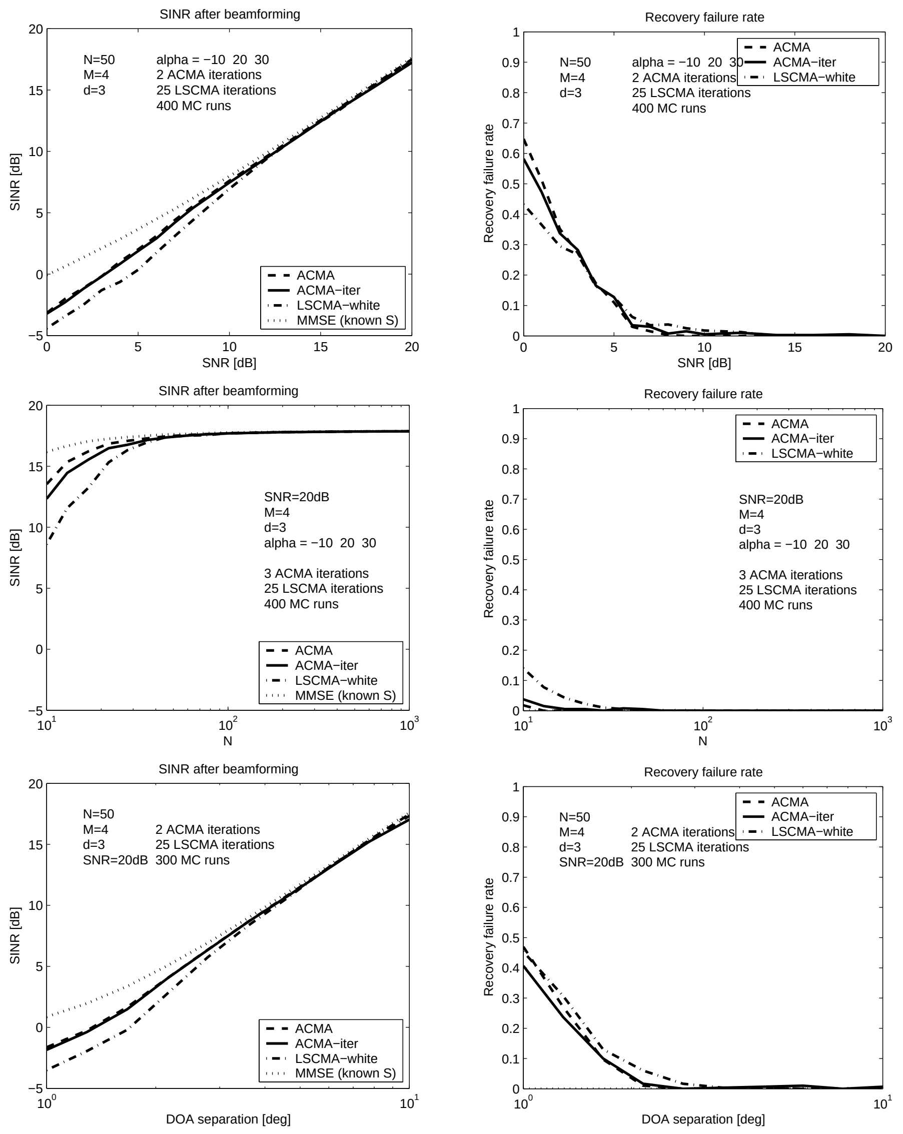


Fig. 6.9 SINR performance and failure rate of ACMA, iterative ACMA, and whitened LS-CMA, as function of SNR, N and DOA separation.

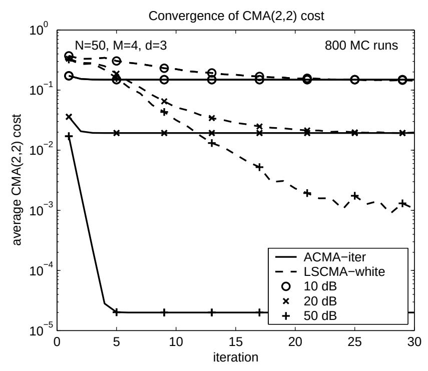


Fig. 6.10 Average convergence of iterative ACMA and whitened LS-CMA iterations, for various SNR levels.

the CMA(2,2) cost, rather it was derived to minimize the CMA(1,2) cost [25]. This explains that the convergence graph in figure 6.10 can be non-monotonic.

6.6 ADAPTIVE PREWHITENING

In the previous sections, the importance of the prewhitening step was highlighted. This has been recognized in the blind source separation community, and algorithms have been devised to produce the filtering matrix $\mathbf{F}$ adaptively. In particular, the prewhitening step requires tracking the inverse of a Cholesky factor of the data covariance matrix $\mathbf{R_x}$. Most of the proposed prewhitening algorithms, however, consider the case $d = M$ and therefore do not implement the dimension reduction step. With dimension reduction, also the dominant subspace has to be tracked: the d-dimensional column span of $\mathbf{F}$ should span the d-dimensional principal column span of the data matrix $\mathbf{X}$. Again, there are many adaptive algorithms for subspace tracking, the most popular being the PAST algorithm [47] and derivatives of it (a 1990 comprehensive overview of subspace tracking algorithms is [48]). Apparently the only paper where these two steps are explicitly combined is a conference paper by Douglas [49]. The derivation is as follows (throughout, it is assumed that the subspace dimension d is known and fixed).

As explained in [47], PAST is based on the idea that the minimizer $\mathbf{U}$ of

$$J(\mathbf{U}) = \mathrm{E}\,\|\mathbf{x} - \mathbf{U}\mathbf{U}^{\mathrm{H}}\mathbf{x}\|^2\,, \qquad \mathbf{U}:\; M \times d$$

is equal to an orthonormal matrix whose columns span the principal subspace of $\mathrm{E}(\mathbf{x}\mathbf{x}^{\mathrm{H}})$. To map this into a practical algorithm, a number of modifications are made. First, the problem is converted into a Least Squares problem, Given $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, \cdots]$ and an exponential forgetting factor $0 < \lambda < 1$, adaptively compute a prewhitening filter $\mathbf{F}_k = \alpha_k \mathbf{U}_k \mathbf{W}_k$, and a prewhitened output $\underline{\mathbf{X}} = [\underline{\mathbf{x}}_1, \underline{\mathbf{x}}_2, \cdots]$, where $\underline{\mathbf{x}}_k = \mathbf{F}_k^{\mathrm{H}} \mathbf{x}_k$, $\mathbf{U}_k$ is an orthonormal basis tracking the d-dimensional column span of $\mathbf{X}$, and $\alpha_k^2 \mathbf{W}_k \mathbf{W}_k^{\mathrm{H}} \to (\mathbf{U}^{\mathrm{H}} \mathbf{R}_{\mathbf{x}} \mathbf{U})^{-1}$

```
Initialize:
$\mathbf{U} = \mathbf{I}_{m\times d}$, $\mathbf{W} = \delta^{-1}\mathbf{I}_{d\times d}$, where $\delta$ is very small
$\alpha^2 = 0$

Update:
for $k = 1, 2, \cdots$ do
  1. $\mathbf{v} = \mathbf{U}^{\mathrm{H}}\mathbf{x}$                                              $(dM)$
     $\mathbf{y} = \mathbf{W}^{\mathrm{H}}\mathbf{v}$                                              $(d^2)$
     $\mathbf{u} = \mathbf{W}\mathbf{y}$                                                $(d^2)$
  2. $\mathbf{k} = \frac{1}{\lambda + \|\mathbf{y}\|^2}\mathbf{u}$                                     $(d)$
     $\mathbf{e} = \mathbf{x} - \mathbf{U}\mathbf{v}$                                              $(dM)$
     $\mathbf{z} = \mathbf{e} - \frac{\|\mathbf{e}\|^2}{2}\mathbf{U}\mathbf{k}$                                      $(dM)$
     $\mathbf{U} = \mathbf{U} + \frac{1}{1 + .25\|\mathbf{e}\|^2\|\mathbf{k}\|^2}\mathbf{z}\mathbf{k}^{\mathrm{H}}$                    $(dM)$
  3. $\zeta = \frac{1}{\lambda + \|\mathbf{y}\|^2 + \sqrt{\lambda}\sqrt{\lambda + \|\mathbf{y}\|^2}}$                    $(1)$
     $\mathbf{W} = \frac{1}{\sqrt{\lambda}}(\mathbf{W} - \zeta\mathbf{u}\mathbf{y}^{\mathrm{H}})$                          $(d^2)$
  4. $\alpha^2 = \lambda\alpha^2 + 1$
     $\underline{\mathbf{x}}_k = \alpha\mathbf{y}$
                                                               $\overline{4dM + 3d^2}$
```

Fig. 6.11 Adaptive prewhitening filter [49].

$\min \sum_i \|\mathbf{x}_i - \mathbf{U}\mathbf{U}^{\mathrm{H}}\mathbf{x}_i\|^2$. Next, to obtain an adaptive algorithm, the contributions of the received samples to the cost function are exponentially scaled down by a scalar λ, where $0 < \lambda \leq 1$, which gives after k samples

$$J(\mathbf{U}_k) = \sum_{i=1}^{k} \lambda^{k-i} \|\mathbf{x}_i - \mathbf{U}_k \mathbf{U}_k^{\mathrm{H}} \mathbf{x}_i\|^2 .$$

Finally, to enable a simple solution to this problem, the vector $\mathbf{U}_k^{\mathrm{H}}\mathbf{x}_i$ which depends on the unknown $\mathbf{U}_k$ is replaced by $\mathbf{v}_i = \mathbf{U}_{i-1}^{\mathrm{H}}\mathbf{x}_i$, which depends on the previous estimate of $\mathbf{U}$ and does not adapt with k. Thus, the cost function used in PAST for subspace tracking is

$$J(\mathbf{U}_k) = \sum_{i=1}^{k} \lambda^{k-i} \|\mathbf{x}_i - \mathbf{U}_k \mathbf{v}_i\|^2 .$$

This has the form of a standard adaptive Least Squares problem. The solution $\mathbf{U}_k$ minimizing this cost is

$$\mathbf{U}_k = \hat{\mathbf{R}}_{\mathbf{xv}}(k)\, \hat{\mathbf{R}}_{\mathbf{v}}^{-1}(k)$$

where

$$\hat{\mathbf{R}}_{\mathbf{xv}}(k) = \sum_{i=1}^{k} \lambda^{k-i} \mathbf{x}_i \mathbf{v}_i^{\mathrm{H}} , \qquad \hat{\mathbf{R}}_{\mathbf{v}}(k) = \sum_{i=1}^{k} \lambda^{k-i} \mathbf{v}_i \mathbf{v}_i^{\mathrm{H}} \tag{6.39}$$

and can be found by the RLS-like algorithm [47]

$$\begin{array}{lcll}
\mathbf{v}_k & = & \mathbf{U}_{k-1}^{\mathrm{H}} \mathbf{x}_k & \\
\mathbf{e}_k & = & \mathbf{x}_k - \mathbf{U}_{k-1} \mathbf{v}_k & \text{(the subspace error)} \\
\mathbf{k}_k & = & \dfrac{\hat{\mathbf{R}}_{\mathbf{v}}^{-1}(k-1)\, \mathbf{v}_k}{\lambda + \mathbf{v}_k^{\mathrm{H}} \hat{\mathbf{R}}_{\mathbf{v}}^{-1}(k-1) \mathbf{v}_k} & \text{(Kalman gain)} \\
\hat{\mathbf{R}}_{\mathbf{v}}^{-1}(k) & = & \frac{1}{\lambda}[\hat{\mathbf{R}}_{\mathbf{v}}^{-1}(k-1) - \mathbf{k}_k \mathbf{v}_k^{\mathrm{H}} \hat{\mathbf{R}}_{\mathbf{v}}^{-1}(k-1)] & \\
\mathbf{U}_k & = & \mathbf{U}_{k-1} + \mathbf{e}_k \mathbf{k}_k^{\mathrm{H}} . &
\end{array} \tag{6.40}$$

This is the PAST algorithm for principal subspace tracking; its complexity is of order dM. Similar to other subspace tracking algorithms of this complexity, the update of $\mathbf{U}_{k-1}$ consists of a rank-1 update in the direction of the subspace error vector $\mathbf{e}_k$, i.e., the component of $\mathbf{x}_k$ outside the estimated subspace.

Although $\mathbf{U}_k$ will converge to an orthonormal matrix for $k \to \infty$ and $\lambda = 1$, this does not ensure the orthonormality of $\mathbf{U}_k$ at every step. As shown in [24], the update step for $\mathbf{U}_k$ can be slightly modified with a second-order term such that the update takes the form of a Householder transformation, which ensures that $\mathbf{U}_k^{\mathrm{H}}\mathbf{U}_k = \mathbf{U}_{k-1}^{\mathrm{H}}\mathbf{U}_{k-1}$. Indeed, let

$$\mathbf{z}_k = \mathbf{e}_k - \frac{\|\mathbf{e}_k\|^2}{2} \mathbf{U}_{k-1} \mathbf{k}_k$$

and define the Householder transformation

$$\mathbf{U}_k = \left(\mathbf{I} - 2\frac{\mathbf{z}_k\mathbf{z}_k^{\mathrm{H}}}{\|\mathbf{z}_k\|^2}\right)\mathbf{U}_{k-1}$$

then (since $\mathbf{e}_k^{\mathrm{H}}\mathbf{U}_{k-1} = \mathbf{0}^{\mathrm{H}}$)

$$\mathbf{U}_k = \mathbf{U}_{k-1} + \frac{\mathbf{z}_k\mathbf{k}_k^{\mathrm{H}}}{1 + 0.25\|\mathbf{e}_k\|^2\|\mathbf{k}_k\|^2}$$

which is close to the update in (6.40), but ensures the orthonormality of $\mathbf{U}_k$ provided the iteration is started with an orthonormal matrix $\mathbf{U}_0$.

The output of the subspace tracking step is a sequence of d-dimensional vectors $\mathbf{v}_k = \mathbf{U}_{k-1}^{\mathrm{H}}\mathbf{x}_k$, where $\mathbf{U}_k$ is an $M \times d$-dimensional matrix whose columns span the estimated principal subspace of $\mathbf{X}$. At convergence, the updates on $\mathbf{U}_k$ are only small, of order $\|\mathbf{e}_k\|^2$. Therefore, $\mathbf{v}_k$ is approximately stationary and $\hat{\mathbf{R}}_{\mathbf{v}}(k)$ is a meaningful estimate of the covariance matrix of $\mathbf{v}_k$.

Subsequently, as described in [49], $\mathbf{v}_k$ can be whitened by an adaptive filter $\mathbf{W}_k$ of size $d \times d$ such that $\mathbf{y}_k := \mathbf{W}_{k-1}^{\mathrm{H}}\mathbf{v}_k$ has covariance matrix $\mathrm{E}(\mathbf{y}_k\mathbf{y}_k^{\mathrm{H}}) = \mathbf{I}$, i.e., $\mathbf{W}_k^{\mathrm{H}}\mathbf{R}_{\mathbf{v}}(k)\,\mathbf{W}_k = \mathbf{I}$ or, with estimated quantities,

$$\mathbf{W}_k\mathbf{W}_k^{\mathrm{H}} = \hat{\mathbf{R}}_{\mathbf{v}}^{-1}(k)\,. \tag{6.41}$$

Therefore, $\mathbf{W}_k$ is a square-root factor of $\hat{\mathbf{R}}_{\mathbf{v}}^{-1}(k)$. Since $\hat{\mathbf{R}}_{\mathbf{v}}^{-1}(k)$ is obtained via a rank-1 update (6.40), a rank-1 update for $\mathbf{W}_k$ can be found in closed form as well [50]. Indeed, substituting (6.41) in the update for $\hat{\mathbf{R}}_{\mathbf{v}}^{-1}(k)$ gives

$$\mathbf{W}_k\mathbf{W}_k^{\mathrm{H}} = \frac{1}{\lambda}\mathbf{W}_{k-1}\left(\mathbf{I} - \frac{\mathbf{y}_k\mathbf{y}_k^{\mathrm{H}}}{\lambda + \|\mathbf{y}_k\|^2}\right)\mathbf{W}_{k-1}^{\mathrm{H}} \tag{6.42}$$

where $\mathbf{y}_k = \mathbf{W}_{k-1}^{\mathrm{H}}\mathbf{v}_k$. Let $\mathbf{B}_k$ be a symmetric square-root factor of the term in braces, then (6.42) implies

$$\mathbf{W}_k = \frac{1}{\sqrt{\lambda}}\mathbf{W}_{k-1}\mathbf{B}_k\,.$$

The square-root factor $\mathbf{B}_k$ is found as

$$\mathbf{B}_k = \left(\mathbf{I} - \frac{\mathbf{y}_k\mathbf{y}_k^{\mathrm{H}}}{\|\mathbf{y}_k\|^2}\right) + \sqrt{\frac{\lambda}{\lambda + \|\mathbf{y}_k\|^2}} \cdot \frac{\mathbf{y}_k\mathbf{y}_k^{\mathrm{H}}}{\|\mathbf{y}_k\|^2}\,,$$

and together this gives the following update [50]:

$$\begin{aligned}
\mathbf{u}_k &= \mathbf{W}_{k-1}\mathbf{y}_k \\
\zeta_k &= \frac{1}{\|\mathbf{y}_k\|^2}\left(1-\sqrt{\frac{\lambda}{\lambda+\|\mathbf{y}_k\|^2}}\right) = \frac{1}{\lambda+\|\mathbf{y}_k\|^2+\sqrt{\lambda}\sqrt{\lambda+\|\mathbf{y}_k\|^2}} \\
\mathbf{W}_k &= \tfrac{1}{\sqrt{\lambda}}(\mathbf{W}_{k-1}-\zeta_k\mathbf{u}_k\mathbf{y}_k^{\mathrm{H}})\,.
\end{aligned}$$

A final point is that, due to the definition of $\hat{\mathbf{R}}_{\mathbf{v}}(k)$ in (6.39), it converges to $1/(1-\lambda)\mathbf{R}_{\mathbf{v}}$ instead of $\mathbf{R}_{\mathbf{v}}$. Rather than modifying all the above equations to take this into account, we can simply scale the resulting whitened output vector $\mathbf{y}_k$ by a factor α_k, where α_k^2 is computed recursively in a similar way as $\mathbf{R}_{\mathbf{v}}$, namely

$$\alpha_k^2 = \lambda\alpha_{k-1}^2 + 1\,, \qquad \alpha_0 = 0\,.$$

The resulting algorithm is summarized in figure 6.11. Its complexity is of order dM. The output of the filter is $\underline{\mathbf{x}}_k = \alpha_k \mathbf{W}_k^{\mathrm{H}}\mathbf{U}_k^{\mathrm{H}}\mathbf{x}_k$.

6.7 ADAPTIVE ACMA

Recall figure 6.8, which summarizes the iterative version of ACMA. To make this block-algorithm adaptive, the following ingredients are needed:

1. Adaptive implementation of the prewhitening filter $\mathbf{F}$,
2. Adaptive tracking of the nullspace of $\underline{\hat{\mathbf{C}}}$,
3. Adaptive update of the joint diagonalization, or alternatively, of the rank-1 mapping of each subspace vector.

The adaptive prewhitening was discussed in section 6.6.

For the second item, ideally we would track the unwhitened $\hat{\mathbf{C}}$ and apply a prewhitening operation $\mathbf{F}$ to $\hat{\mathbf{C}}$ after each update to find a consistent estimate of $\underline{\hat{\mathbf{C}}}$ and its nullspace. An update of $\hat{\mathbf{C}}$ is straightforward to derive from its definition in equation (6.27), but it would cost order M^4 per update, which is too much. Applying $(\overline{\mathbf{F}}_k \otimes \mathbf{F}_k)$ to the left and right of this matrix after each update would be even more costly. Therefore, we have to assume that the prewhitening filter $\mathbf{F}_k$ changes only slowly, and track the whitened $\underline{\hat{\mathbf{C}}}$ using whitened update vectors $\bar{\underline{\mathbf{x}}}_k \otimes \underline{\mathbf{x}}_k$. This leads to a complexity of order d^4, which is still too much in comparison to the existing adaptive CMAs which have complexity d^2. Thus, we will avoid to construct and store $\underline{\hat{\mathbf{C}}}$, but will directly update its nullspace using the update vectors of $\underline{\hat{\mathbf{C}}}$.

6.7.1 Adaptive tracking of $\underline{\hat{\mathbf{C}}}$

First we derive an update equation for $\underline{\hat{\mathbf{C}}}$. So far, $\underline{\hat{\mathbf{C}}}$ was defined only in terms of a given batch of N samples, but we would like to convert this into an exponential window (λ-scaling). As before, let $\mathbf{P}$ be an $N \times d^2$ dimensional matrix with rows $\mathbf{y}_k^{\mathrm{H}} := (\bar{\underline{\mathbf{x}}}_k \otimes \underline{\mathbf{x}}_k)^{\mathrm{H}}$, where $\underline{\mathbf{x}}_k$ is the received (whitened) sample at time k. For simplicity of notation, we will drop the underscores in this subsection from now on, since all data will be in the whitened domain. According to equation (6.27)

$$\hat{\mathbf{C}} := \frac{1}{N}\mathbf{P}^{\mathrm{H}}\mathbf{P} - \frac{1}{N^2}\mathbf{P}^{\mathrm{H}}\mathbf{1}\,\mathbf{1}^{\mathrm{H}}\mathbf{P}\,.$$

Comparing this matrix to

$$\mathbf{M} := \sum_{k=1}^{N} \begin{bmatrix} 1 \\ \mathbf{y}_k \end{bmatrix} [1 \quad \mathbf{y}_k^{\mathrm{H}}] = [\mathbf{1} \quad \mathbf{P}]^{\mathrm{H}} [\mathbf{1} \quad \mathbf{P}] = \begin{bmatrix} \mathbf{1}^{\mathrm{H}}\mathbf{1} & \mathbf{1}^{\mathrm{H}}\mathbf{P} \\ \mathbf{P}^{\mathrm{H}}\mathbf{1} & \mathbf{P}^{\mathrm{H}}\mathbf{P} \end{bmatrix} \qquad (6.43)$$

we see that $N\hat{\mathbf{C}}$ is equal to the Schur complement $\mathbf{M}_{2,2} - \mathbf{M}_{2,1}\mathbf{M}_{1,1}^{-1}\mathbf{M}_{1,2}$.

An adaptive update rule for $\hat{\mathbf{C}}$ can be derived from an adaptive update rule for $\mathbf{M}$, where we scale previous estimates with λ, with $0 < \lambda < 1$. Thus let $\mathbf{M}_k$ and $\mathbf{C}_k$ be defined as

$$\mathbf{M}_k := (1-\lambda)\sum_{i=1}^{k} \lambda^{k-i} \begin{bmatrix} 1 \\ \mathbf{y}_i \end{bmatrix} [1 \quad \mathbf{y}_i^{\mathrm{H}}] =: \begin{bmatrix} \alpha_k & \mathbf{p}_k^{\mathrm{H}} \\ \mathbf{p}_k & \mathbf{N}_k \end{bmatrix}, \qquad \mathbf{C}_k := \mathbf{N}_k - \frac{\mathbf{p}_k\mathbf{p}_k^{\mathrm{H}}}{\alpha_k}$$

then $\mathbf{M}_k$ is an unbiased estimate of $\mathbf{M}$, due to multiplication with the factor $(1-\lambda)$, and $\mathbf{C}_k$ is an unbiased estimate of $\hat{\mathbf{C}}$. The update rule for $\mathbf{M}_k$ which follows from this equation is

$$\mathbf{M}_k = \lambda\mathbf{M}_{k-1} \;+\; (1-\lambda)\begin{bmatrix} 1 \\ \mathbf{y}_k \end{bmatrix} [1 \quad \mathbf{y}_k^{\mathrm{H}}]$$

so that

$$\begin{bmatrix} \alpha_k & \mathbf{p}_k^{\mathrm{H}} \\ \mathbf{p}_k & \mathbf{N}_k \end{bmatrix} = \begin{bmatrix} \lambda\alpha_{k-1} + (1-\lambda) & \lambda\mathbf{p}_{k-1}^{\mathrm{H}} + (1-\lambda)\mathbf{y}_k^{\mathrm{H}} \\ \lambda\mathbf{p}_{k-1} + (1-\lambda)\mathbf{y}_k & \lambda\mathbf{N}_{k-1} + (1-\lambda)\mathbf{y}_k\mathbf{y}_k^{\mathrm{H}} \end{bmatrix}$$

and

$$\begin{aligned} \mathbf{C}_k &= \mathbf{N}_k - \frac{\mathbf{p}_k\mathbf{p}_k^{\mathrm{H}}}{\alpha_k} \\ &= \lambda\mathbf{C}_{k-1} + \tfrac{\lambda}{\alpha_{k-1}}\mathbf{p}_{k-1}\mathbf{p}_{k-1}^{\mathrm{H}} + (1-\lambda)\mathbf{y}_k\mathbf{y}_k^{\mathrm{H}} \\ &\quad - \tfrac{1}{\alpha_k}[\lambda\mathbf{p}_{k-1} + (1-\lambda)\mathbf{y}_k][\lambda\mathbf{p}_{k-1} + (1-\lambda)\mathbf{y}_k]^{\mathrm{H}} \\ &= \lambda\mathbf{C}_{k-1} + [\mathbf{y}_k \quad \mathbf{p}_{k-1}] \begin{bmatrix} (1-\lambda) - \frac{(1-\lambda)^2}{\alpha_k} & -\frac{\lambda(1-\lambda)}{\alpha_k} \\ -\frac{\lambda(1-\lambda)}{\alpha_k} & \frac{\lambda}{\alpha_{k-1}} - \frac{\lambda^2}{\alpha_k} \end{bmatrix} \begin{bmatrix} \mathbf{y}_k^{\mathrm{H}} \\ \mathbf{p}_{k-1}^{\mathrm{H}} \end{bmatrix}. \end{aligned}$$

Using $\alpha_k = \lambda\alpha_{k-1} + (1-\lambda)$, it follows that the 2×2 matrix in the update is actually of rank 1, namely equal to

$$\frac{1-\lambda}{\alpha_k}\begin{bmatrix} \alpha_k - (1-\lambda) & -\lambda \\ -\lambda & \frac{\lambda}{\alpha_{k-1}} \end{bmatrix} = \frac{\alpha_{k-1}}{\alpha_k}\lambda(1-\lambda)\begin{bmatrix} 1 & -\frac{1}{\alpha_{k-1}} \\ -\frac{1}{\alpha_{k-1}} & \frac{1}{\alpha_{k-1}^2} \end{bmatrix}$$

so that we remain with a one-dimensional update

$$\begin{aligned} \mathbf{C}_k &= \lambda\mathbf{C}_{k-1} + \tfrac{\alpha_{k-1}}{\alpha_k}\lambda(1-\lambda)\cdot(\mathbf{y}_k - \mathbf{p}_{k-1}/\alpha_{k-1})(\mathbf{y}_k - \mathbf{p}_{k-1}/\alpha_{k-1})^{\mathrm{H}} \\ &=: \lambda\mathbf{C}_{k-1} + \beta_k\mathbf{c}_k\mathbf{c}_k^{\mathrm{H}}\,. \end{aligned} \tag{6.44}$$

Therefore, the vector by which to update $\mathbf{C}_{k-1}$ is equal to a scaling of the modified data vector

$$\mathbf{c}_k := \underline{\bar{\mathbf{x}}}_k \otimes \underline{\mathbf{x}}_k \; - \; \mathbf{p}_{k-1}/\alpha_{k-1} \tag{6.45}$$

where $\mathbf{p}_{k-1}$ and α_{k-1} are updated as

$$\mathbf{p}_k = \lambda\mathbf{p}_{k-1} + (1-\lambda)\underline{\bar{\mathbf{x}}}_k \otimes \underline{\mathbf{x}}_k\,, \qquad \alpha_k = \lambda\alpha_{k-1} + (1-\lambda)\,. \tag{6.46}$$

Note that $\mathbf{p}_k$ is an exponentially-weighted unbiased estimate of the mean of $\underline{\bar{\mathbf{x}}}_k \otimes \underline{\mathbf{x}}_k$, which matches its use in equation (6.26), whereas α_k converges to 1 and is only relevant during the initialization phase of the algorithm.

6.7.2 Adaptive tracking of the nullspace of $\underline{\hat{\mathbf{C}}}$

Subspace tracking is well studied in signal processing, and a variety of algorithms has been derived. Algorithms can be classified in several ways:

1. *Subspace:* Principal components versus minor subspace (or nullspace) tracking, or both,
2. *Rank determination:* a specified subspace dimension or a specified error threshold,
3. *Complexity:* for d-dimensional subspaces in an M-dimensional space, algorithms of order M^2d, d^2M, down to dM have been reported
4. *Strategy:* exact (deterministic), gradient descent (stochastic), etc.

The paper [48] gives an overview; the most reliable algorithm seems to be Karasalo's, with complexity d^2M.

In the case of an adaptive version for ACMA, we have to track the nullspace of $\underline{\hat{\mathbf{C}}}$, which is a d-dimensional nullspace in a d^2-dimensional space. The lowest complexity that can be achieved for this case is of order d^3. To remain competitive with CMA and MUK, a higher complexity cannot be tolerated. Therefore only nullspace tracking algorithms of order dM are applicable. The most popular algorithms in this context are based on the PAST algorithm [47],

```
For a sequence of $M$-dimensional vectors $\mathbf{c}_1, \mathbf{c}_2, \cdots$, compute adaptive estimates
of an orthonormal basis $\mathbf{V}_k$ of the $d$-dimensional nullspace:

Initialize $\mathbf{V}_1 = [\mathbf{I}_{d\times d} \quad \mathbf{0}_{d\times(m-d)}]^{\mathrm{H}}$

for $k = 1, 2, \cdots$ do
        $\mathbf{y} = \mathbf{V}_k^{\mathrm{H}}\mathbf{c}_k$                                   $(dM)$
        $\mathbf{z} = \mathbf{V}_k\mathbf{y}$                                    $(dM)$
        $\mathbf{p} = \mathbf{c}_k - \mathbf{z}$
        $\beta = \frac{1}{\|\mathbf{c}_k\|^2 - \|\mathbf{y}\|^2 + 0.2}$
        $\phi = \frac{1}{\sqrt{1 + \beta^2\|\mathbf{p}\|^2\|\mathbf{y}\|^2}}$
        $\tau = \frac{\phi - 1}{\|\mathbf{y}\|^2}$
        $\mathbf{u} = -\tau/\beta\mathbf{z} + \phi\mathbf{p}$
        $\mathbf{V}_{k+1} = \mathbf{V}_k - 2\frac{\mathbf{u}\mathbf{u}^{\mathrm{H}}}{\|\mathbf{u}\|^2}\mathbf{V}_k$                  $(2dM)$
                                                          $4dM$
end
```

Fig. 6.12 Adaptive nullspace tracking using NOOJA [51].

which as described in section 6.6 is derived from an iterative optimization of the cost function

$$
\begin{aligned}
J(\mathbf{V}) &= \mathrm{E}\|\mathbf{c} - \mathbf{V}\mathbf{V}^{\mathrm{H}}\mathbf{c}\|^2 \\
&= \mathrm{tr}(\mathbf{C}) - 2\mathrm{tr}(\mathbf{V}^{\mathrm{H}}\mathbf{C}\mathbf{V}) + \mathrm{tr}(\mathbf{V}^{\mathrm{H}}\mathbf{C}\mathbf{V}\mathbf{V}^{\mathrm{H}}\mathbf{V})
\end{aligned}
\tag{6.47}
$$

where $\mathbf{V}$ is the estimated subspace, $\mathbf{c}$ is the data vector, in our context given by equation (6.45), and $\mathbf{C} = \mathrm{E}(\mathbf{c}\mathbf{c}^{\mathrm{H}})$ is the matrix from which the subspace has to be determined. Minimization of the cost function will produce an orthogonal basis for the principal subspace, whereas maximization leads to a basis of the nullspace (in this case normalization constraints on $\mathbf{V}$ are needed to avoid trivial solutions).

As discussed in section 6.6, the PAST algorithm follows from an Alternating Least Squares implementation of the optimization problem: compute the vector $\mathbf{y}_k = \mathbf{V}_{k-1}^{\mathrm{H}}\mathbf{c}_k$ using the previous estimate of $\mathbf{V}$, then optimize $\sum_{i=1}^{k} \lambda^{k-i}\|\mathbf{c}_i - \mathbf{V}\mathbf{y}_i\|^2$ over $\mathbf{V}$, where λ is the forgetting factor. This resulted in an RLS-type algorithm.

In an alternative approach in [51], a gradient descent algorithm is derived from (6.47):

$$
\mathbf{V}_k := \mathbf{V}_{k-1} - \beta_k \nabla_J(\mathbf{V}_{k-1}) \tag{6.48}
$$

where β_k is a step size and $\nabla_J(\mathbf{V}_{k-1})$ the gradient of $J(\mathbf{V})$ evaluated at $\mathbf{V}_{k-1}$:

$$\nabla_J(\mathbf{V}) = (-2\mathbf{C} + \mathbf{C}\mathbf{V}\mathbf{V}^{\mathrm{H}} + \mathbf{V}\mathbf{V}^{\mathrm{H}}\mathbf{C})\mathbf{V}\,.$$

By taking $\beta_k < 0$, the cost function is maximized and an estimate of the nullspace is obtained. In [51], $\mathbf{V}$ is constrained to be orthonormal ($\mathbf{V}^{\mathrm{H}}\mathbf{V} = \mathbf{I}$), and a variable step size β_k is selected such that it optimizes the cost in every step (since the cost function (6.47) with $\mathbf{V}$ replaced by $\mathbf{V}_k$ from (6.48) is quadratic in β_k, it can be computed in closed form). Subsequently, the gradient is approximated by replacing $\mathbf{C}$ by an estimate based on the current sample, $\mathbf{c}_k\mathbf{c}_k^{\mathrm{H}}$, which leads to

$$\mathbf{V}_k := \mathbf{V}_{k-1} - \beta_{\mathrm{opt},k}(-\mathbf{c}_k + \mathbf{V}_{k-1}\mathbf{y}_k)\mathbf{y}_k^{\mathrm{H}}\,. \tag{6.49}$$

After this update step, the new basis $\mathbf{V}_k$ needs to be orthonormalized again, which is done by setting

$$\mathbf{V}_k := \mathbf{V}_k(\mathbf{V}_k^{\mathrm{H}}\mathbf{V}_k)^{-1/2}\,.$$

Since (6.49) represents a rank-1 update, the normalization can be computed efficiently in closed form using a Householder reflection, similar as in section 6.6. The resulting algorithm is called "Normalized Orthogonal OJA" (NOOJA) [51], and is summarized in figure 6.12. It is interesting to note that it is of complexity $4dM$, and scaling-independent: if the input data $\mathbf{c}_k$ is multiplied by a scalar, then the resulting $\mathbf{V}_k$ is unchanged. Although there are a few alternative algorithms for minor subspace tracking [52, 53, 54, 55], this algorithm currently seems to be one of the fastest and more reliable (e.g., some of the other algorithms perform poor for high SNR). A disadvantage is that the subspace estimate remains jittery in steady state, because the large stepsize tends to emphasize the instantaneous noise. Therefore, instead of taking the maximal stepsize $\beta_{\mathrm{opt},k}$, typically a fraction of this step size is used.

6.7.3 Adaptive update of the joint diagonalization

Using the preceding nullspace tracking algorithm in our application, we can update the basis $\mathbf{V}_n$ of the nullspace of $\hat{\underline{\mathbf{C}}}$ given the update vector $\mathbf{c}_k$ in (6.45). The final step is an efficient implementation of the joint diagonalization. After the subspace update, the iterative ACMA (figure 6.8) uses the previous estimate of the beamformers, $\mathbf{T}$, and continues with the following three steps:

$$\begin{array}{rcl}\mathbf{M} & = & \mathbf{V}_n^{\mathrm{H}}(\bar{\mathbf{T}} \circ \mathbf{T}) \\ \mathbf{Y} & = & \mathbf{V}_n\mathbf{M} = \mathbf{V}_n\mathbf{V}_n^{\mathrm{H}}(\bar{\mathbf{T}} \circ \mathbf{T}) \\ \mathbf{t}_i & = & \pi_1(\mathbf{y}_i), \quad i = 1, \cdots, d\,.\end{array}$$

This projects $\bar{\mathbf{T}} \circ \mathbf{T}$ onto the estimated subspace, resulting in $\mathbf{Y}$, and subsequently maps the columns of $\mathbf{Y}$ back to the Kronecker-product structure, resulting in new estimates of the columns $\mathbf{t}_i$ of $\mathbf{T}$. However, the complexity of the projection is too high (order d^4 instead of d^3).

Therefore, the following modification is introduced: instead of updating the basis $\mathbf{V}_n$, we compute $\bar{\mathbf{T}} \circ \mathbf{T}$ and regard it as the current estimate of the subspace basis (i.e., set $\mathbf{V}_n = \bar{\mathbf{T}} \circ \mathbf{T}$). Using this basis, the subspace update is performed, giving $\mathbf{Y}$, and then the result is mapped back to the Kronecker-product structure. In this context, the update performed by the NOOJA algorithm is interpreted as a Householder reflection which tries to make $\bar{\mathbf{T}} \circ \mathbf{T}$ orthogonal to the current update vector $\mathbf{c}_k$.

The last step of the algorithm is the mapping of the columns $\mathbf{y}_i$ of $\mathbf{Y}$ to a Kronecker-product structure, $\mathbf{y}_i =: \bar{\mathbf{t}}_i \otimes \mathbf{t}_i$, or equivalently,

$$\mathbf{Y}_i = \mathbf{t}_i \mathbf{t}_i^{\mathrm{H}}$$

where $\mathrm{vec}(\mathbf{Y}_i) = \mathbf{y}_i$. An SVD can be used to estimate $\mathbf{t}_i$ as the dominant singular vector, as was done in the nonadaptive version in figure 6.8, but it would cost order d^3 per subspace vector, or d^4 in total. Since we need only the dominant singular vector, we can instead apply a power iteration [23]. The general form of an iterative step in the power iteration is

$$\mathbf{v}_{k+1} = \mathbf{Y}\mathbf{v}_k\,, \qquad k = 1, 2, \cdots$$

where the iteration is initialized by a randomly selected vector $\mathbf{v}_0$. However, the best choice of an initial point is the previous estimate for $\mathbf{t}_i$, and in this case, a single step of the iteration is sufficient to give a good improvement of the estimate. The complexity of one update step is d^2 per subspace vector, or d^3 in total.

6.7.4 Summary of the resulting algorithm

The resulting algorithm is summarized in figure 6.13. As indicated, the complexity of the algorithm is of order $4dM + 6d^3$. This is comparable to the complexity of OCMA and MUK (figure 6.5), which was computed as order $4dM + d^3 + 5d^2$, where the term d^3 is contributed by the reorthogonalization step, which perhaps can be implemented cheaper. Therefore, the complexity is at most a factor d worse, where d is small (typically $d \leq 5$).

6.7.5 Comparison of MUK with Adaptive-ACMA

To compare the performance of the MUK algorithm with the Adaptive-ACMA derived in this section, we show two sets of simulations. In the first, a stationary scenario is considered, whereas in the second, the source powers and array response vectors are time varying.

```
Given data $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, \cdots]$, compute beamformers $\mathbf{W}_k = \mathbf{F}_k\mathbf{T}_k$ and output
$\hat{\mathbf{s}}_k = \mathbf{W}_k^{\mathrm{H}}\mathbf{x}_k$:

Initialize prewhitening filter $\mathbf{F}$ using about $5M$ prior input samples;
$\mathbf{T} = \mathbf{I}_{d\times d}$, $\mathbf{p} = \mathbf{0}, \alpha = 0$

for $k = 1, 2, \cdots$ do
  1. Update $\mathbf{F}$, the prewhitening filter, using $\mathbf{x}_k$ (fig. 6.11)        $(4dM + 3d^2)$
     $\underline{\mathbf{x}} = \mathbf{F}^{\mathrm{H}}\mathbf{x}_k$, the prewhitened input vector
  2. Compute the update vector $\mathbf{c}$ for $\hat{\underline{\mathbf{C}}}$:                            $(d^2)$
          $\mathbf{c} = \bar{\underline{\mathbf{x}}} \otimes \underline{\mathbf{x}} - \mathbf{p}/\alpha$
          $\mathbf{p} = \lambda\mathbf{p} + (1-\lambda)\bar{\underline{\mathbf{x}}} \otimes \underline{\mathbf{x}}$
          $\alpha = \lambda\alpha + (1-\lambda)$
     Compute $\mathbf{Y} = \bar{\mathbf{T}} \circ \mathbf{T}$                                      $(d^3)$
     Regard $\mathbf{Y}$ as a basis of the nullspace of $\hat{\underline{\mathbf{C}}}$, and
     update it using $\mathbf{c}$ (fig. 6.12)                                $(4d^3)$
  3. for $i = 1, \cdots, d$ do                                         $(d^3)$
          $\mathbf{Y}_i = \mathrm{vec}^{-1}(\mathbf{y}_i)$
          $\mathbf{t}_i = \mathbf{Y}_i\mathbf{t}_i$     (one step of a power iteration)
          $\mathbf{t}_i = \mathbf{t}_i/\|\mathbf{t}_i\|$
     end
  4. $\mathbf{T} = \mathrm{recond}(\mathbf{T})$,    equation (6.19)                       $(d^3)$
  5. $\hat{\mathbf{s}}_k = \mathbf{T}^{\mathrm{H}}\underline{\mathbf{x}}$                                            $(d^2)$
                                                                 $4dM + 6d^3$
```

Fig. 6.13 Adaptive implementation of ACMA.

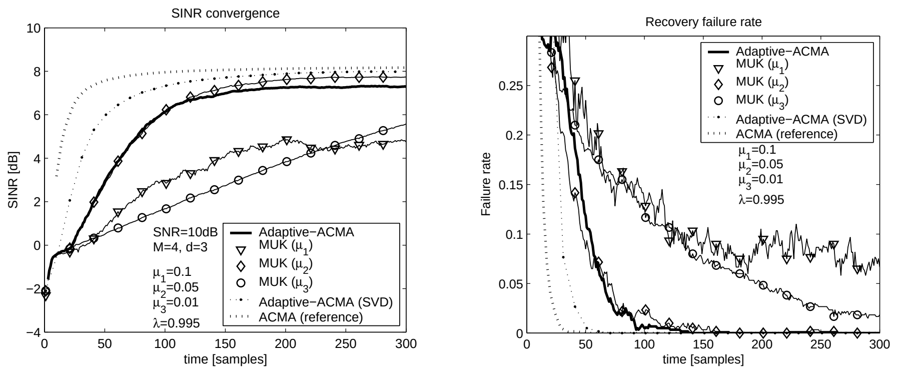


Fig. 6.14 Average convergence and failure rate of MUK and Adaptive-ACMA—stationary channel.

6.7.5.1 Stationary channel Entirely similar to section 6.5.5, we take a uniform linear array with $M = 4$ antennas, $d = 3$ constant-modulus sources with directions $[-10°, 20°, 30°]$ and amplitudes [1, .8, .9]. The SNR is 10 dB per antenna for the strongest user.

Figure 6.14(a) shows the worst SINR among the users, averaged over 600 monte-carlo runs, as a function of the sample index. Since not always all independent users are recovered, figure 6.14(b) shows the percentage of failed cases. These cases are not used in the SINR statistics. The dotted reference curve is formed by ACMA acting on a growing block of samples (no forgetting factor λ).

For Adaptive-ACMA, the first 20 samples are used to initialize the adaptive prewhitening filter. The convergence speed depends on λ, the plot shows the results for $\lambda = 0.995$. A smaller λ gives faster initial convergence but a lower steady-state performance. The final SINR level is also determined by the nullspace tracking algorithm. NOOJA with optimal step size is a "greedy" algorithm which quickly tracks the subspace, but this also results in a more noisy output. Instead of the optimal step size $\beta_{\text{opt},k}$, we have used $0.3\beta_{\text{opt},k}$ which gives a smoother performance at the expense of initial tracking speed. To verify that the nullspace tracking error is a limiting factor in the steady state, we also implemented a version where $\hat{\mathbf{C}}_k$ is formed as in (6.44) and its nullspace computed using SVD (dotted reference curve "Adaptive-ACMA(SVD)" in the figure; for this algorithm also the rank-1 truncation is performed using SVD).

For MUK, the performance behavior dependents on λ due to the adaptive prewhitening, and also strongly depends on the value of the step size μ, therefore three values are shown; the value $\mu = 0.05$ gives a remarkably similar performance to Adaptive-ACMA in this scenario both in convergence speed

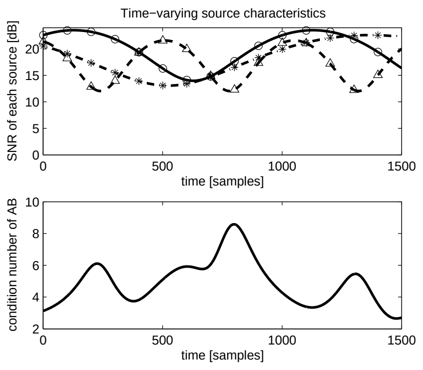


Fig. 6.15 (a) SNR and (b) conditioning of a time-varying channel (example).

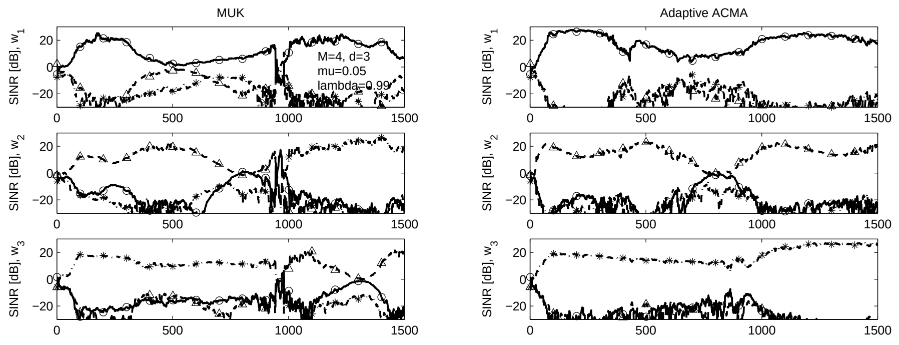


Fig. 6.16 Example tracking behavior of MUK and adaptive-ACMA—time-varying channel. For each beamformer the output SINR corresponding to each source is shown.

and in asymptotic SNR. The experience is that for higher SNRs, Adaptive-ACMA will outperform MUK by a few dB.

6.7.5.2 Time-varying channel To test the tracking behavior of the adaptive algorithms, the preceding scenario is made time-varying. Specifically, the source amplitudes β_i are varied in sinusoidal patterns with randomly selected periods, with a maximum of 3 periods over the simulated time interval ($N = 1500$ samples). An example is shown in figure 6.15(a). The direction vectors $\{\mathbf{a}_i\}$ of each source are not selected on an array manifold, instead each entry of each $\mathbf{a}_i$ is a unimodular complex number with a linear phase progression, causing at most one cycle per interval. The condition number of the resulting channel matrix $\mathbf{AB} = [\mathbf{a}_1\beta_1, \cdots, \mathbf{a}_d\beta_d]$ is plotted in figure 6.15(b).

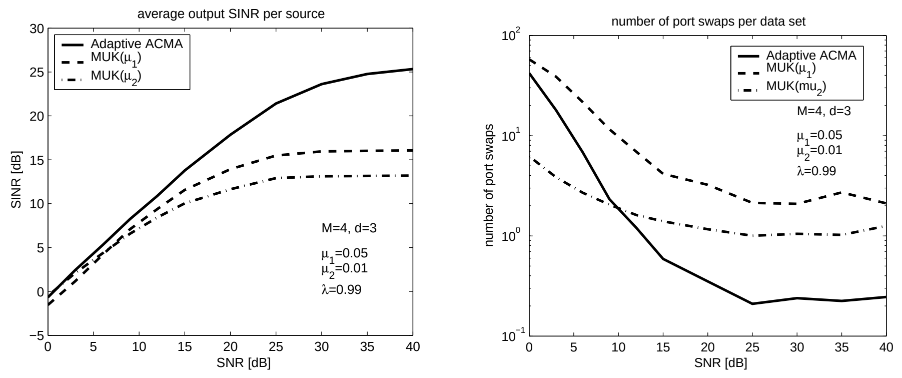


Fig. 6.17 Performance of MUK and adaptive-ACMA in a time-varying scenario: (a) average output SINR, (b) average number of port swaps per interval (1500 samples).

The output SINR for each beamformer is shown in figure 6.16 for MUK and Adaptive ACMA, respectively. Each panel corresponds to a beamformer $\mathbf{w}_i$, $i = 1, \cdots, d$, and the jth curve in a panel corresponds to the SINR behavior of the jth source, i.e., related to $\mathbf{a}_j\beta_j$. For this example, MUK experienced a case of "port swapping" around sample number 1000, i.e., a beamformer suddenly starts to track a different source. This can occur if two sources come too close or if the scenario changes faster than the algorithm can track. The fluctuations in the output SINR of the tracked source obviously also follow the fluctuations in the input SNR.

Figure 6.17(a) shows the average output SINR of each source for MUK and Adaptive ACMA as a function of SNR, where the average is over time, over the three sources, and over 300 monte carlo runs, each with different randomly varying channels (there has been no attempt to detect and remove "failed cases"). Finally, figure 6.17(b) shows the average number of times that a port swap occurred in a data run of N samples. The performance of MUK is sensitive to the choice of stepsize μ: a larger μ gives faster tracking and resulted in better output SINR, but also gave more port swaps: usually at least once per data set. Both algorithms are performance limited at high SNR due to the time-variation in the observation window, but overall, Adaptive-ACMA is better in this particular scenario. Since this is a new algorithm, it is unclear whether this is observation holds in general.

6.8 DOA ASSISTED BEAMFORMING OF CONSTANT MODULUS SIGNALS

In some applications the sensor array is assumed to be calibrated, i.e., it is known parametrically as a function of a vector parameter $\boldsymbol{\theta}$. One typical example is when $\boldsymbol{\theta}$ represents the direction of arrivals (DOAs) of the received signals. DOA estimation has been thoroughly studied for arbitrary signals. For a good overview of DOA estimation for arbitrary signals we refer the reader to [56] and [57], or indeed some of the other chapters in this book. When the signals have a constant modulus the number of nuisance parameters (i.e., the signal parameters) is reduced by a factor of two, since the amplitudes are constant. As we will see this enables a much more accurate estimation of the direction of arrival of the signals and consequently leads to a better separation.

The literature on DOA estimation of CM signals is relatively sparse. The most common approach is to separate the sources based on the CM property, followed by estimating the direction of each recovered signal. Initially this has been done using the CM Array [58, 19, 59], but as mentioned in preceding sections, the CM Array is recursive in nature and requires many hundreds of samples to obtain convergence. Very good results on small numbers of samples have been obtained by first estimating the channel matrix using ACMA, and subsequently obtaining a decoupled DOA estimation problem where the estimated steering vectors are projected onto the model based steering vectors [60].

Maximum Likelihood estimation for the joint CM signals estimation and DOA estimation is the optimal way. For the trivial case of a single source it had been shown to be equivalent to L_1 beamforming, which is more robust than the classical L_2 beamformer [61]. For more than a single source a computationally attractive approach has been suggested in [62, 63].

The Cramer Rao Bound (CRB) gives a lower bound on the variance of any unbiased estimator, and is an important measure for the efficiency of estimators. It has been widely used for estimating the performance bound of DOA estimation. The additional information brought by the CM assumption has been computed in [60, 64]. In this section we present some of these results: the Cramer Rao lower bound, a description of the Maximum Likelihood algorithm, and simulation results presenting the various methods and the robustness to mis-modeling.

6.8.1 Data model

For the purpose of this section, we have to extend the previously used data model $\mathbf{x}(t) = \mathbf{A}\mathbf{s}(t) + \mathbf{n}(t)$ to include a parametrically known channel matrix $\mathbf{A}$. Since now we cannot place the unknown source amplitudes into $\mathbf{A}$, we also have to introduce a gain factor $\mathbf{B}$, which leads to

$$\mathbf{x}(t) = \mathbf{A}(\boldsymbol{\theta})\mathbf{B}\mathbf{s}(t) + \mathbf{n}(t) \tag{6.50}$$

where

$\mathbf{A}(\boldsymbol{\theta}) = [\mathbf{a}(\theta_1), \cdots, \mathbf{a}(\theta_d)]$, where $\mathbf{a}(\theta)$ is the array response vector for a signal from direction θ, and $\boldsymbol{\theta} = [\theta_1, \cdots, \theta_d]$ is the DOA vector of the sources (for simplicity, we assume 1 parameter per source),

$\mathbf{B} = \text{diag}(\boldsymbol{\beta})$ is the gain matrix, with parameters $\boldsymbol{\beta} = [\beta_1, \cdots, \beta_d]^{\mathrm{T}}$, where $\beta_i \in \mathbb{R}^+$ is the amplitude of the i-th signal as received by the array.

As usual in DOA estimation, we require that the array manifold satisfies the uniqueness condition, i.e., every collection of M vectors on the manifold is linearly independent.

As before, we assume that all sources have constant modulus. Unequal source powers are absorbed in the gain matrix $\mathbf{B}$. Phase offsets of the sources after demodulation are part of the $s_i(t)$. Thus we can write $s_i(t) = e^{j\phi_i(t)}$, where $\phi_i(t)$ is the unknown phase modulation for source i, and we define $\boldsymbol{\phi}(t) = [\phi_1(t), \cdots, \phi_d(t)]^{\mathrm{T}}$ as the phase vector for all sources at time t. We further assume that the noise is Gaussian and spatially white with covariance matrix $\mathbf{R}_{nn} = \sigma^2\mathbf{I} = \nu\mathbf{I}$, where ν is the known noise variance as received on a single antenna.

6.8.2 Likelihood function

Based on the model and assuming N received sample vectors collected in a matrix $\mathbf{X}$, we can derive the likelihood function. For deterministic CM signals in white Gaussian noise the likelihood function is given by

$$L(\mathbf{X}|\boldsymbol{\Phi}, \boldsymbol{\theta}, \boldsymbol{\beta}) = \frac{1}{(2\pi)^N (\frac{\nu}{2})^{pN}} \exp\left\{ -\frac{1}{\nu} \sum_{k=1}^{N} \mathbf{e}^{\mathrm{H}}(k)\mathbf{e}(k) \right\}, \tag{6.51}$$

where

$$\mathbf{e}(k) = \mathbf{x}(k) - \mathbf{A}\mathbf{B}\mathbf{s}(k)\,, \tag{6.52}$$

and

$$\boldsymbol{\Phi} = [\boldsymbol{\phi}(1), \ldots, \boldsymbol{\phi}(N)]\;. \tag{6.53}$$

Let $\mathcal{L}(\mathbf{X}|\boldsymbol{\Phi}, \boldsymbol{\theta}) = \log L(\mathbf{X}|\boldsymbol{\Phi}, \boldsymbol{\theta}, \boldsymbol{\beta})$. After omitting constants we obtain the log-likelihood function

$$\mathcal{L}(\mathbf{X}|\boldsymbol{\Phi}, \boldsymbol{\theta}, \boldsymbol{\beta}) = -\frac{1}{\nu} \sum_{k=1}^{N} \|\mathbf{e}(k)\|^2\,. \tag{6.54}$$

6.8.3 Cramer-Rao Bound

The Cramer Rao Bound (CRB) is a lower bound on the estimation variance of any unbiased estimator. Its derivation from the log-likelihood function follows along standard lines [65]. Indeed, the CRB is given by the main diagonal of the inverse of the Fisher Information Matrix (FIM). In turn, the FIM specifies the "sensitivity" of the log-likelihood function (regarded as cost function) to changes in the parameters,

$$\mathbf{F}_N = \mathrm{E}\left\{ \frac{\partial \mathcal{L}}{\partial \boldsymbol{\rho}} \cdot (\frac{\partial \mathcal{L}}{\partial \boldsymbol{\rho}})^{\mathrm{T}} \right\}$$

where $\boldsymbol{\rho}$ is a vector which collects all parameters,

$$\boldsymbol{\rho} = [\boldsymbol{\phi}(1)^{\mathrm{T}}, \cdots, \boldsymbol{\phi}(N)^{\mathrm{T}},\ \boldsymbol{\theta}^{\mathrm{T}},\ \boldsymbol{\beta}^{\mathrm{T}}]^{\mathrm{T}} .$$

For the case at hand, this can be worked out in closed form as follows [60]. Partition the FIM as

$$\mathbf{F}_N = \begin{bmatrix} \mathbf{F}_{11} & \mathbf{F}_{12} \\ \mathbf{F}_{21} & \mathbf{F}_{22} \end{bmatrix} \qquad (6.55)$$

where the partitioning follows the partitioning of $\boldsymbol{\rho}$ into vec($\boldsymbol{\Phi}$) followed by $[\boldsymbol{\theta}; \boldsymbol{\beta}]$. Then

$$\mathbf{F}_{11} = \begin{bmatrix} \mathbf{H}_1 & & \mathbf{0} \\ & \ddots & \\ \mathbf{0} & & \mathbf{H}_N \end{bmatrix}, \quad \mathbf{F}_{12} = \begin{bmatrix} \boldsymbol{\Delta}_1^{\mathrm{T}} & \mathbf{E}_1^{\mathrm{T}} \\ \vdots & \vdots \\ \boldsymbol{\Delta}_N^{\mathrm{T}} & \mathbf{E}_N^{\mathrm{T}} \end{bmatrix} \qquad (6.56)$$

$$\mathbf{F}_{21} = \begin{bmatrix} \boldsymbol{\Delta}_1 & ,\cdots, & \boldsymbol{\Delta}_N \\ \mathbf{E}_1 & ,\cdots, & \mathbf{E}_N \end{bmatrix}, \quad \mathbf{F}_{22} = \begin{bmatrix} \boldsymbol{\Gamma} & \boldsymbol{\Lambda}^{\mathrm{T}} \\ \boldsymbol{\Lambda} & \boldsymbol{\Upsilon} \end{bmatrix}, \qquad (6.57)$$

where

$$\begin{array}{lcll}
\mathbf{H}_k & := & \mathrm{E}\frac{\partial \mathcal{L}}{\partial \boldsymbol{\phi}(k)}(\frac{\partial \mathcal{L}}{\partial \boldsymbol{\phi}(k)})^{\mathrm{T}} & = & \frac{2}{\nu}\mathrm{Re}(\mathbf{S}_k^{\mathrm{H}}\mathbf{B}^{\mathrm{H}}\mathbf{A}^{\mathrm{H}}\mathbf{A}\mathbf{B}\mathbf{S}_k) \\
\boldsymbol{\Delta}_k & := & \mathrm{E}\frac{\partial \mathcal{L}}{\partial \boldsymbol{\theta}}(\frac{\partial \mathcal{L}}{\partial \boldsymbol{\phi}(k)})^{\mathrm{T}} & = & -\frac{2}{\nu}\mathrm{Im}(\mathbf{S}_k^{\mathrm{H}}\mathbf{B}^{\mathrm{H}}\mathbf{D}^{\mathrm{H}}\mathbf{A}\mathbf{B}\mathbf{S}_k) \\
\mathbf{E}_k & := & \mathrm{E}\frac{\partial \mathcal{L}}{\partial \boldsymbol{\beta}}(\frac{\partial \mathcal{L}}{\partial \boldsymbol{\phi}(k)})^{\mathrm{T}} & = & -\frac{2}{\nu}\mathrm{Im}(\mathbf{S}_k^{\mathrm{H}}\mathbf{A}^{\mathrm{H}}\mathbf{A}\mathbf{B}\mathbf{S}_k) \\
\boldsymbol{\Gamma} & := & \mathrm{E}\frac{\partial \mathcal{L}}{\partial \boldsymbol{\theta}}(\frac{\partial \mathcal{L}}{\partial \boldsymbol{\theta}})^{\mathrm{T}} & = & \frac{2}{\nu}\sum_{k=1}^{N}\mathrm{Re}(\mathbf{S}_k^{\mathrm{H}}\mathbf{B}^{\mathrm{H}}\mathbf{D}^{\mathrm{H}}\mathbf{D}\mathbf{B}\mathbf{S}_k) \\
\boldsymbol{\Lambda} & := & \mathrm{E}\frac{\partial \mathcal{L}}{\partial \boldsymbol{\beta}}(\frac{\partial \mathcal{L}}{\partial \boldsymbol{\theta}})^{\mathrm{T}} & = & \frac{2}{\nu}\sum_{k=1}^{N}\mathrm{Re}(\mathbf{S}_k^{\mathrm{H}}\mathbf{A}^{\mathrm{H}}\mathbf{D}\mathbf{B}\mathbf{S}_k) \\
\boldsymbol{\Upsilon} & := & \mathrm{E}\frac{\partial \mathcal{L}}{\partial \boldsymbol{\beta}}(\frac{\partial \mathcal{L}}{\partial \boldsymbol{\beta}})^{\mathrm{T}} & = & \frac{2}{\nu}\sum_{k=1}^{N}\mathrm{Re}(\mathbf{S}_k^{\mathrm{H}}\mathbf{A}^{\mathrm{H}}\mathbf{A}\mathbf{S}_k)
\end{array}$$

and

$$\mathbf{S}_k = \mathrm{diag}(\mathbf{s}(k)), \qquad \mathbf{D} = \left[\frac{d\mathbf{a}}{d\theta}(\theta_1), \cdots, \frac{d\mathbf{a}}{d\theta}(\theta_d)\right] .$$

To give closed-form expressions for the inverse of the FIM, we use the following general result for block-partitioned matrices (which can be easily derived by inverting an LDU factorization):

$$\begin{bmatrix} \mathbf{F}_{11} & \mathbf{F}_{12} \\ \mathbf{F}_{21} & \mathbf{F}_{22} \end{bmatrix}^{-1} = \begin{bmatrix} \mathbf{I} & -\mathbf{F}_{11}^{-1}\mathbf{F}_{12} \\ \mathbf{0} & \mathbf{I} \end{bmatrix} \begin{bmatrix} \mathbf{F}_{11}^{-1} & \\ & (\mathbf{F}_{22} - \mathbf{F}_{21}\mathbf{F}_{11}^{-1}\mathbf{F}_{12})^{-1} \end{bmatrix} \begin{bmatrix} \mathbf{I} & \mathbf{0} \\ -\mathbf{F}_{21}\mathbf{F}_{11}^{-1} & \mathbf{I} \end{bmatrix}$$

Thus let

$$\begin{bmatrix} \Xi_{11} & \Xi_{12} \\ \Xi_{21} & \Xi_{22} \end{bmatrix} := \mathbf{F}_{21}\mathbf{F}_{11}^{-1}\mathbf{F}_{12} = \begin{bmatrix} \sum_{k=1}^{N} \boldsymbol{\Delta}_k \mathbf{H}_k^{-1} \boldsymbol{\Delta}_k^{\mathrm{T}} & \sum_{k=1}^{N} \boldsymbol{\Delta}_k \mathbf{H}_k^{-1} \mathbf{E}_k^{\mathrm{T}} \\ \sum_{k=1}^{N} \mathbf{E}_k \mathbf{H}_k^{-1} \boldsymbol{\Delta}_k^{\mathrm{T}} & \sum_{k=1}^{N} \mathbf{E}_k \mathbf{H}_k^{-1} \mathbf{E}_k^{\mathrm{T}} \end{bmatrix}$$

and

$$\boldsymbol{\Psi} := \mathbf{F}_{22} - \mathbf{F}_{21}\mathbf{F}_{11}^{-1}\mathbf{F}_{12} = \begin{bmatrix} \boldsymbol{\Gamma} & \boldsymbol{\Lambda}^{\mathrm{T}} \\ \boldsymbol{\Lambda} & \boldsymbol{\Upsilon} \end{bmatrix} - \begin{bmatrix} \Xi_{11} & \Xi_{12} \\ \Xi_{21} & \Xi_{22} \end{bmatrix} \tag{6.58}$$

so that

$$\begin{aligned} (\mathbf{F}_N^{-1})_{11} &= \begin{bmatrix} \mathbf{H}_1^{-1} & & \mathbf{0} \\ & \ddots & \\ \mathbf{0} & & \mathbf{H}_N^{-1} \end{bmatrix} + \\ &\begin{bmatrix} \mathbf{H}_1^{-1}\boldsymbol{\Delta}_1^{\mathrm{T}} & \mathbf{H}_1^{-1}\mathbf{E}_1^{\mathrm{T}} \\ \vdots & \vdots \\ \mathbf{H}_N^{-1}\boldsymbol{\Delta}_N^{\mathrm{T}} & \mathbf{H}_N^{-1}\mathbf{E}_N^{\mathrm{T}} \end{bmatrix} \boldsymbol{\Psi}^{-1} \begin{bmatrix} \boldsymbol{\Delta}_1\mathbf{H}_1^{-1}, \cdots, \boldsymbol{\Delta}_N\mathbf{H}_N^{-1} \\ \mathbf{E}_1\mathbf{H}_1^{-1}, \cdots, \mathbf{E}_N\mathbf{H}_N^{-1} \end{bmatrix} \end{aligned} \tag{6.59}$$

$$(\mathbf{F}_N^{-1})_{12} = -\begin{bmatrix} \mathbf{H}_1^{-1}\boldsymbol{\Delta}_1^{\mathrm{T}} & \mathbf{H}_1^{-1}\mathbf{E}_1^{\mathrm{T}} \\ \vdots & \vdots \\ \mathbf{H}_N^{-1}\boldsymbol{\Delta}_N^{\mathrm{T}} & \mathbf{H}_N^{-1}\mathbf{E}_N^{\mathrm{T}} \end{bmatrix} \boldsymbol{\Psi}^{-1} \tag{6.60}$$

$$(\mathbf{F}_N^{-1})_{21} = -\boldsymbol{\Psi}^{-1} \begin{bmatrix} \boldsymbol{\Delta}_1\mathbf{H}_1^{-1} & ,\cdots, & \boldsymbol{\Delta}_N\mathbf{H}_N^{-1} \\ \mathbf{E}_1\mathbf{H}_1^{-1} & ,\cdots, & \mathbf{E}_N\mathbf{H}_N^{-1} \end{bmatrix} \tag{6.61}$$

$$(\mathbf{F}_N^{-1})_{22} = \boldsymbol{\Psi}^{-1} = \left(\begin{bmatrix} \boldsymbol{\Gamma} & \boldsymbol{\Lambda}^{\mathrm{T}} \\ \boldsymbol{\Lambda} & \boldsymbol{\Upsilon} \end{bmatrix} - \begin{bmatrix} \Xi_{11} & \Xi_{12} \\ \Xi_{21} & \Xi_{22} \end{bmatrix} \right)^{-1} . \tag{6.62}$$

We assumed that the $\mathbf{H}_k$ are invertible, which would follow from the independence condition on the array manifold and the independence of the sources.

The CRB on the parameters is given by the diagonal elements of $\mathbf{F}_N^{-1}$. Using the partitioned matrix inversion formula again on (6.62), the CRB for DOAs and amplitudes follows as

$$\begin{aligned} \mathrm{CRB}_N(\boldsymbol{\theta}) &= \mathrm{diag}(\boldsymbol{\Psi}^{-1})_{11} \\ &= \mathrm{diag}\left[(\boldsymbol{\Gamma} - \Xi_{11}) - (\boldsymbol{\Lambda}^{\mathrm{T}} - \Xi_{12})(\boldsymbol{\Upsilon} - \Xi_{22})^{-1}(\boldsymbol{\Lambda} - \Xi_{21})\right]^{-1} \end{aligned} \tag{6.63}$$

and

$$\begin{aligned} \mathrm{CRB}_N(\boldsymbol{\beta}) &= \mathrm{diag}(\boldsymbol{\Psi}^{-1})_{22} \\ &= \mathrm{diag}\left[(\boldsymbol{\Upsilon} - \Xi_{22}) - (\boldsymbol{\Lambda} - \Xi_{21})(\boldsymbol{\Gamma} - \Xi_{11})^{-1}(\boldsymbol{\Lambda}^{\mathrm{T}} - \Xi_{12})\right]^{-1} . \end{aligned} \tag{6.64}$$

Similarly, the bound on the estimation variance of the signal phases follows as

$$\mathrm{CRB}_N(\phi(k)) = \mathrm{diag}\left\{\mathbf{H}_k^{-1}\left[\mathbf{I}+[\mathbf{\Delta}_k^{\mathrm{T}} \quad \mathbf{E}_k^{\mathrm{T}}]\mathbf{\Psi}^{-1}\begin{bmatrix}\mathbf{\Delta}_k\\ \mathbf{E}_k\end{bmatrix}\mathbf{H}_k^{-1}\right]\right\}. \qquad (6.65)$$

Note that the number of samples and the quality of DOA estimation affects the bound only through the matrix $\mathbf{\Psi}^{-1}$.

6.8.4 CM-DOA algorithm based on ACMA initialization

To estimate the parameters of the model, we have to minimize the negative log-likelihood function (6.54), which is a Least Squares problem:

$$\min \ \| \mathbf{X} - \mathbf{A}(\boldsymbol{\theta})\mathbf{B}(\boldsymbol{\beta})\mathbf{S}(\mathbf{\Phi}) \|_F^2 .$$

In spite of the simple appearance, it cannot be solved in closed form. A simple technique in such cases is to revert to Alternating Least Squares types of algorithms (as in some of the preceding sections): estimate $\mathbf{S}(\mathbf{\Phi})$, then estimate $\mathbf{A}(\boldsymbol{\theta})$, etc. Based on this idea, the following ad-hoc technique gives surprisingly good results:

1. Blindly estimate a matrix $\hat{\mathbf{A}}$, using the CM assumption on $\mathbf{S}$. This step can be done by ACMA;
2. Estimate the directions which best fit the matrix $\hat{\mathbf{A}}$.

The second step can be carried out for each column of $\hat{\mathbf{A}}$ separately: let $\hat{\mathbf{A}} = [\hat{\mathbf{a}}_1, \ldots, \hat{\mathbf{a}}_d]$ be the estimate of the mixing matrix, then solve for each source i

$$\hat{\theta}_i = \arg\min_{\theta,\beta} \| \hat{\mathbf{a}}_i - \mathbf{a}(\theta)\beta \|^2$$

This problem can be decoupled. The optimal value for β is $\hat{\beta} = \mathbf{a}(\theta)^{\dagger}\hat{\mathbf{a}}_i$, and after eliminating β, the estimate for θ_i is given by

$$\hat{\theta}_i = \arg\min_{\theta} \| \left(\mathbf{I} - \mathbf{a}(\theta)\mathbf{a}(\theta)^{\dagger}\right) \hat{\mathbf{a}}_i \|$$

which can be converted into

$$\hat{\theta}_i = \arg\max_{\theta} \frac{|\hat{\mathbf{a}}_i^{\mathrm{H}}\mathbf{a}(\theta)|}{\|\mathbf{a}(\theta)\|}. \qquad (6.66)$$

Equation (6.66) simply describes a maximization of the projection of the estimated vector onto the array manifold. The computational complexity of this method is not very large compared to other DOA estimation methods, and can be ignored in view of the complexity of the first step.

Various other suboptimal methods for combining existing DOA and CM estimators have been proposed. E.g., in [66] the ESPRIT algorithm for DOA estimation is suboptimally combined with ACMA. The problem in such an approach is the choice of weighting of the two properties: without a good weighting, the solution may be worse than the best single method by itself, and finding the proper weighting is a hard problem. Often, the simple two-step approach gives equally good results. Another advantage of the simple CM-DOA method is that it is applicable to arbitrary array configurations: it does not use the special array structure required by ESPRIT.

6.8.5 Deterministic maximum likelihood techniques

In general, Maximum Likelihood techniques are more complex but are expected to give better results. A deterministic ML approach for DOA estimation of CM signals was derived in [63]. It is based on the Least Squares formulation of the log likelihood in equation (6.54):

$$\hat{\boldsymbol{\rho}} = \arg\min_{\boldsymbol{\rho}} \sum_{k=1}^{N} \|\mathbf{e}(k)\|^2 \tag{6.67}$$

The Newton scoring method (where we replace the Hessian by its expected value for better numerical performance) to find the minimum is the iteration

$$\boldsymbol{\rho}^{(n+1)} = \boldsymbol{\rho}^{(n)} - \lambda \mathbf{F}_N^{-1}(\boldsymbol{\rho}^{(n)}) \nabla \mathcal{L}(\boldsymbol{\rho}^{(n)})$$

where $\lambda \leq 1$ is a suitable step size, $\mathbf{F}_N$ is the FIM, which is the expected value of the Hessian, and $\nabla \mathcal{L}(\boldsymbol{\rho}^{(n)})$ is the gradient, with components $\nabla_{\boldsymbol{\phi}(k)}\mathcal{L}$, $\nabla_{\boldsymbol{\theta}}\mathcal{L}$, and $\nabla_{\boldsymbol{\beta}}\mathcal{L}$. These are given by

$$\nabla_{\boldsymbol{\phi}(k)}\mathcal{L} = \frac{\partial \mathcal{L}}{\partial \boldsymbol{\phi}(k)} = \frac{2}{\nu} \mathrm{Im}\left(\mathbf{S}(k)^{\mathrm{H}}\mathbf{B}^{\mathrm{H}}\mathbf{A}^{\mathrm{H}}\mathbf{e}(k)\right). \tag{6.68}$$

$$\nabla_{\boldsymbol{\theta}}\mathcal{L} = \frac{\partial \mathcal{L}}{\partial \boldsymbol{\theta}} = \frac{2}{\nu} \sum_{k=1}^{N} \mathrm{Re}\left(\mathbf{S}^{\mathrm{H}}(k)\mathbf{B}^{\mathrm{H}}\mathbf{D}^{\mathrm{H}}\mathbf{e}(k)\right) \tag{6.69}$$

and

$$\nabla_{\boldsymbol{\beta}}\mathcal{L} = \frac{\partial \mathcal{L}}{\partial \boldsymbol{\beta}} = \frac{2}{\nu} \sum_{k=1}^{N} \mathrm{Re}\left(\mathbf{S}^{\mathrm{H}}(k)\mathbf{A}^{\mathrm{H}}\mathbf{e}(k)\right). \tag{6.70}$$

The Newton update direction $\mathbf{v}$ is given by $\mathbf{v} = \mathbf{F}_N^{-1}\nabla\mathcal{L}$. This is a $d(N+2) \times 1$ vector function of the parameters. Explicit expressions for the components of

```
1. Find an initial estimate $\boldsymbol{\rho}_0$ for the parameters using the suboptimal algo-
   rithm in section 6.8.4.
2. For $n = 0, 1, \cdots$ until convergence do
      a. Estimate the Newton direction $\mathbf{v}$ using (6.71)
      b. Compute $\lambda$ by $\lambda = \arg\min_\mu -\mathcal{L}(\boldsymbol{\rho}^{(n)} - \mu\mathbf{v})$
      c. Update the parameters using $\boldsymbol{\rho}_{n+1} = \boldsymbol{\rho}^{(n)} - \lambda\mathbf{v}$
   end
```

Fig. 6.18 Deterministic ML DOA estimation and source separation for constant modulus sources.

$\mathbf{v}$ are given by:

$$\begin{aligned}
(\mathbf{v})_{\boldsymbol{\phi}(k)} = \; & \mathbf{H}_k^{-1}\nabla_{\boldsymbol{\phi}(k)}\mathcal{L} + \\
& \begin{bmatrix} \mathbf{H}_k^{-1}\boldsymbol{\Delta}_k^{\mathrm{T}} & \mathbf{H}_k^{-1}\mathbf{E}_k^{\mathrm{T}} \end{bmatrix} \boldsymbol{\Psi}^{-1} \begin{bmatrix} \sum_{i=1}^N \boldsymbol{\Delta}_i\mathbf{H}_i^{-1}\nabla_{\boldsymbol{\phi}(i)}\mathcal{L} - \nabla_{\boldsymbol{\theta}}\mathcal{L} \\ \sum_{i=1}^N \mathbf{E}_i\mathbf{H}_i^{-1}\nabla_{\boldsymbol{\phi}(i)}\mathcal{L} - \nabla_{\boldsymbol{\beta}}\mathcal{L} \end{bmatrix} \\
(\mathbf{v})_{\boldsymbol{\theta},\boldsymbol{\beta}} = \; & \boldsymbol{\Psi}^{-1} \begin{bmatrix} \nabla_{\boldsymbol{\theta}}\mathcal{L} - \sum_{i=1}^N \boldsymbol{\Delta}_i\mathbf{H}_i^{-1}\nabla_{\boldsymbol{\phi}(i)}\mathcal{L} \\ \nabla_{\boldsymbol{\beta}}\mathcal{L} - \sum_{i=1}^N \mathbf{E}_i\mathbf{H}_i^{-1}\nabla_{\boldsymbol{\phi}(i)}\mathcal{L} \end{bmatrix}
\end{aligned} \tag{6.71}$$

where $(\mathbf{v})_{\boldsymbol{\phi}(k)}$ are the components related to the phase parameters, and $(\mathbf{v})_{\boldsymbol{\theta},\boldsymbol{\beta}}$ are the components related to the DOAs and signal power parameters. Note that the matrices $\mathbf{H}_k$ are of size $d \times d$ and therefore their inversion is simple, and that there is no dependence of $\mathbf{v}$ on the noise variance ν.

We thus arrive at the algorithm in figure 6.18 [63]. The step size parameter λ in the algorithm is selected such that it minimizes the likelihood function, and can be obtained by a standard one-dimensional optimization method (cf. [67]), using the good initialization $\lambda_0 = 1$ which is optimal for the quadratic approximation of the likelihood.

The computational complexity of an update step of the scoring algorithm can be estimated as $O\left((d^3 + Md)N\right)$ [63], and can be reduced by optimizing the order of operations in the computation. This puts the algorithm in the class of moderate complexity algorithms: harder than eigenspace methods or polynomial rooting algorithms, but lower than multidimensional search methods. Since it can be used with an arbitrary array geometry, it is appealing in many cases where specific assumptions on the array geometry are not valid.

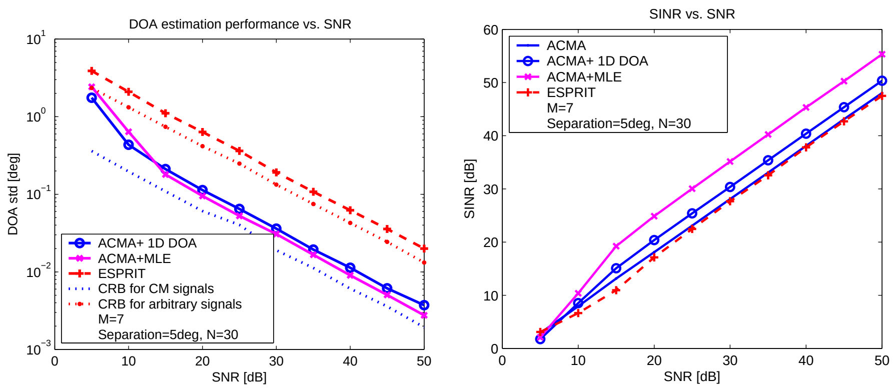


Fig. 6.19 First experiment: (a) DOA estimation accuracy vs. SNR, (b) SINR vs. SNR.

6.8.6 Simulation results

We finish the section by some simulations demonstrating the efficiency of the CM-DOA estimation methods. We have used an $M = 7$-element Uniform Linear Array (ULA), and $d = 3$ sources with varying angle separations.

The first experiment tests the performance as function of the input signal to noise ratio (SNR). The three sources had equal power and were located at $-5°, 0°, 5°$, the number of samples was $N = 30$, and the SNR was varied from 5 to 50 dB. Figure 6.19 shows the results for various techniques: ACMA followed by a 1-dimensional DOA search as in section 6.8.4, the same technique followed by the ML estimation as described in figure 6.18, and (in dashed lines) the ESPRIT algorithm [68] which uses the ULA structure. As seen in figure 6.19, the use of the CM property gives an order of magnitude improvement in the DOA estimate. In terms of output Signal to Interference and Noise Ratio (SINR), ACMA and ESPRIT are about equal, but the MLE is 3–5 dB better. The number of iterations required for the MLE was about 5 for large SNRs, and 15–20 for smaller SNRs (10 dB and below).

A second experiment tests the performance as a function of the angle separation between the sources, in a case with near-far problems. We have used $N = 40$ samples. The central source is fixed at 0° while the two other sources are located at $-\Delta°, \Delta°$, where Δ is changed from 4° to 30°. The Signal to Interference Ratio (SIR) of the central source is at -20 dB below the strong sources, and the SNR for the weak source is 20 dB. This tests the near-far robustness of the method. Figure 6.20 presents the standard deviation of the DOA estimate and the output SINR for the weak source.

For small separations, the CM-initialized techniques lead to better DOA estimates than ESPRIT. However, for large separations the ESPRIT algorithm

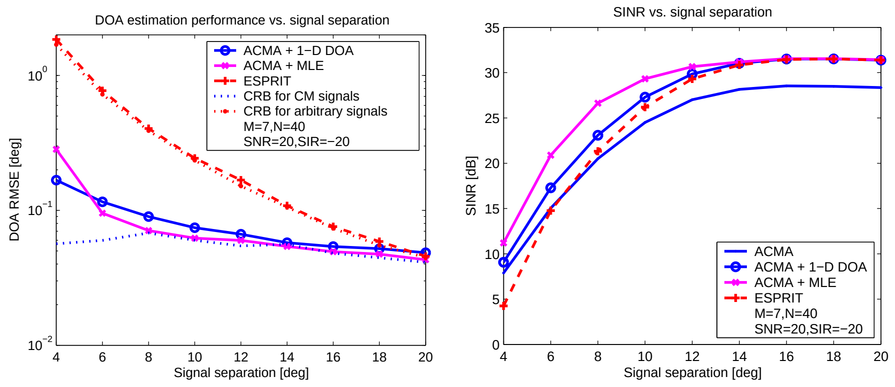


Fig. 6.20 Second experiment: (a) DOA estimation accuracy vs. separation, (b) SINR vs. separation.

tends to have better output SINR performance than ACMA. The MLE outperforms the other techniques, with a significant 3–5 dB advantage at small separations, which demonstrates the importance of exploiting both the array structure and the CM property.

6.9 CONCLUDING REMARKS

In this chapter, we studied algorithms for the blind separation of multiple constant modulus signals. The challenge was to find the complete set of *all* beamformers (one for each impinging signal). For a small batch of samples, ACMA is currently the only algorithm that can reliably do this. For a moving window of samples (sample-adaptive techniques), only algorithms which adaptively prewhiten the data and recondition the orthogonality of the beamformers are reliable. One example is the MUK algorithm. We have derived an adaptive implementation of ACMA which was shown to be more reliable than MUK in a rapidly time-varying scenario, at a similar computational complexity. The robustness and performance of this algorithm in more general cases (e.g., varying number of sources) still needs to be established.

If the aim is to estimate the directions of arrival of the sources, algorithms which also exploit the constant modulus property have been shown to give a significant performance improvement at reasonable computational costs over algorithms which only consider the array manifold (e.g., the ESPRIT algorithm). This is in particular true at small angular separations of the sources.

Acknowledgment

This work was supported in part by the Dutch Min. Econ. Affairs under TSIT 1025 "Beyond-3G", and by NWO-STW under the VICI programme (DTC.5893).

References

1. J.R. Treichler and B.G. Agee, “A new approach to multipath correction of constant modulus signals,” *IEEE Trans. Acoust., Speech, Signal Processing*, vol. 31, pp. 459–471, April 1983.

2. R. Gooch and J. Lundell, “The CM array: An adaptive beamformer for constant modulus signals,” in *IEEE Intl. Conf. Acoustics, Speech, Signal Processing (ICASSP)*, (Tokyo), pp. 2523–2526, 1986.

3. J.R. Treichler and M.G. Larimore, “New processing techniques based on constant modulus adaptive algorithm,” *IEEE Trans. Acoust., Speech, Signal Processing*, vol. 33, pp. 420–431, April 1985.

4. A.J. van der Veen and A. Paulraj, “An analytical constant modulus algorithm,” *IEEE Trans. Signal Processing*, vol. 44, pp. 1136–1155, May 1996.

5. J.F. Cardoso and A. Souloumiac, “Blind beamforming for non-Gaussian signals,” *IEE Proc. F (Radar and Signal Processing)*, vol. 140, pp. 362–370, December 1993.

6. C. Papadias, “Globally convergent blind source separation based on a multiuser kurtosis maximization criterion,” *IEEE Trans. Signal Processing*, vol. 48, pp. 3508–3519, December 2000.

7. B. Widrow and S.D. Stearns, *Adaptive Signal Processing.* Englewood Cliffs, NJ: Prentice-Hall, 1985.

8. M.G. Larimore and J.R. Treichler, “Convergence behavior of the constant modulus algorithm,” in *IEEE Intl. Conf. Acoustics, Speech, Signal Processing (ICASSP)*, pp. 13–16 vol.1, 1983.

9. Y. Sato, “A method of self-recovering equalization for multilevel amplitude-modulation systems,” *IEEE Trans. Communications*, vol. 23, pp. 679–682, June 1975.

10. D.N. Godard, “Self-recovering equalization and carrier tracking in two-dimensional data communication systems,” *IEEE Trans. Communications*, vol. 28, pp. 1867–1875, November 1980.

11. S. Haykin, ed., *Blind Deconvolution.* Englewood Cliffs: Prentice Hall, 1994.

12. R. Johnson, P. Schniter, T.J. Endres, J.D. Behm, D.R. Brown, and R.A. Casas, "Blind equalization using the constant modulus criterion: a review," *Proceedings of the IEEE*, vol. 86, pp. 1927–1950, October 1998.

13. J.R. Treichler, M.G. Larimore, and J.C. Harp, "Practical blind demodulators for high-order QAM signals," *Proceedings of the IEEE*, vol. 86, pp. 1907–1926, October 1998.

14. K. Hilal and P. Duhamel, "A convergence study of the constant modulus algorithm leading to a normalized-CMA and a block-normalized-CMA," in *Signal Processing VI - Theories and Applications. Proceedings of EUSIPCO-92, Sixth European Signal Processing Conference* (J. Vandewalle, R. Boite, M. Moonen, and A. Oosterlinck, eds.), (Brussels), pp. 135–138 vol.1, Elsevier, 1992.

15. S. Haykin, *Adaptive Filter Theory.* Prentice-Hall, 1991.

16. R. Pickholtz and K. Elbarbary, "The recursive constant modulus algorithm: A new approach for real-time array processing," in *27-th Asilomar Conf. Signals, Syst. Comp.*, pp. 627–632, IEEE, 1993.

17. A.V. Keerthi, A. Mathur, and J.J. Shynk, "Misadjustment and tracking analysis of the constant modulus array," *IEEE Trans. Signal Processing*, vol. 46, pp. 51–58, January 1998.

18. A. Mathur, A.V. Keerthi, J.J. Shynk, and R.P. Gooch, "Convergence properties of the multistage constant modulus array for correlated sources," *IEEE Trans. Signal Processing*, vol. 45, pp. 280–286, January 1997.

19. J.J. Shynk and R.P. Gooch, "The constant modulus array for cochannel signal copy and direction finding," *IEEE Trans. Signal Processing*, vol. 44, pp. 652–660, March 1996.

20. T. Nguyen and Z. Ding, "Blind CMA beamforming for narrowband signals with multipath arrivals," *Int. J. Adaptive Control and Signal Processing*, vol. 12, pp. 157–172, March 1998.

21. C.B. Papadias and A.J. Paulraj, "A constant modulus algorithm for multiuser signal separation in presence of delay spread using antenna arrays," *IEEE Signal Processing Letters*, vol. 4, pp. 178–181, June 1997.

22. O. Shalvi and E. Weinstein, "New criteria for blind deconvolution of nonminimum phase systems (channels)," *IEEE Trans. Information Theory*, vol. 36, pp. 312–321, March 1990.

23. G. Golub and C.F. Van Loan, *Matrix Computations.* The Johns Hopkins University Press, 1989.

24. S.C. Douglas, "Numerically-robust adaptive subspace tracking using householder transformations," in *First IEEE Sensor Array and Multichannel Signal Processing Workshop*, (Boston, MA), pp. 499–503, March 2000.

25. B.G. Agee, "The least-squares CMA: A new technique for rapid correction of constant modulus signals," in *IEEE Intl. Conf. Acoustics, Speech, Signal Processing (ICASSP)*, (Tokyo), pp. 953–956, 1986.

26. R.W. Gerchberg and W.O. Saxton, "A practical algorithm for the determination of phase from image and diffraction plane pictures," *Optik*, vol. 35, pp. 237–246, 1972.

27. Y. Wang, Y.C. Pati, Y.M. Cho, A. Paulraj, and T. Kailath, "A matrix factorization approach to signal copy of constant modulus signals arriving at an antenna array," in *Proc. 28-th Conf. on Informat. Sciences and Systems*, (Princeton, NJ), March 1994.

28. A.J. van der Veen, "Algebraic constant modulus algorithms," in *Signal Processing Advances in Wireless and Mobile Communications* (G. Giannakis e.a., ed.), vol. 2, ch. 3, Prentice Hall, 2000.

29. A.J. van der Veen, "Asymptotic properties of the algebraic constant modulus algorithm," *IEEE Trans. Signal Processing*, vol. 49, pp. 1796–1807, August 2001.

30. A. Leshem, N. Petrochilos, and A.J. van der Veen, "Finite sample identifiability of multiple constant modulus sources," *IEEE Trans. Information Theory*, vol. 49, pp. 2314–2319, September 2003.

31. A.J. van der Veen, "Statistical performance analysis of the algebraic constant modulus algorithm," *IEEE Trans. Signal Processing*, vol. 50, pp. 3083–3097, December 2002.

32. A.J. van der Veen, "Analytical method for blind binary signal separation," *IEEE Trans. Signal Processing*, vol. 45, pp. 1078–1082, April 1997.

33. Ming Gu and Lang Tong, "Geometrical characterizations of constant modulus receivers," *IEEE Trans. Signal Processing*, vol. 47, pp. 2745–2756, October 1999.

34. H.H. Zeng, Lang Tong, and C.R. Johnson, "Relationships between the constant modulus and Wiener receivers," *IEEE Trans. Inform. Theory*, vol. 44, pp. 1523–1538, July 1998.

35. H.H. Zeng, Lang Tong, and C.R. Johnson, "An analysis of constant modulus receivers," *IEEE Trans. Signal Processing*, vol. 47, pp. 2990–2999, November 1999.

36. A. Belouchrani, K. Abed-Meraim, J.-F. Cardoso, and E. Moulines, "A blind source separation technique using second-order statistics," *IEEE Trans. Signal Processing*, vol. 45, pp. 434–444, February 1997.

37. P. Binding, "Simultaneous diagonalization of several Hermitian matrices," *SIAM J. Matrix Anal. Appl.*, vol. 4, no. 11, pp. 531–536, 1990.

38. A. Bunse-Gerstner, R. Byers, and V. Mehrmann, "Numerical methods for simultaneous diagonalization," *SIAM J. Matrix Anal. Appl.*, vol. 4, pp. 927–949, 1993.

39. J.-F. Cardoso and A. Souloumiac, "Jacobi angles for simultaneous diagonalization," *SIAM J. Matrix Anal. Appl.*, vol. 17, no. 1, pp. 161–164, 1996.

40. M.T. Chu, "A continuous Jacobi-like approach to the simultaneous reduction of real matrices," *Lin. Alg. Appl.*, vol. 147, pp. 75–96, 1991.

41. B.D. Flury and B.E. Neuenschwander, "Simultaneous diagonalization algorithms with applications in multivariate statistics," in *Approximation and Computation* (R.V.M. Zahar, ed.), pp. 179–205, Basel: Birkhäuser, 1995.

42. M. Haardt and J.A. Nossek, "Simultaneous Schur decomposition of several nonsymmetric matrices to achieve automatic pairing in multidimensional harmonic retrieveal problems," *IEEE Trans. Signal Processing*, vol. 46, pp. 161–169, January 1998.

43. L. De Lathauwer, B. De Moor, and J. Vandewalle, "Independent component analysis based on higher-order statistics only," in *Proc. IEEE SP Workshop on Stat. Signal Array Proc.*, (Corfu, Greece), pp. 356–359, 1996.

44. N.D. Sidiropoulos, G.B. Giannakis, and R. Bro, "Parallel factor analysis in sensor array processing," *IEEE Trans. Signal Processing*, vol. 48, pp. 2377–2388, August 2000.

45. M. Wax and J. Sheinvald, "A least-squares approach to joint diagonalization," *IEEE Signal Processing Letters*, vol. 4, pp. 52–53, February 1997.

46. A. Yeredor, "Non-orthogonal joint diagonalization in the least-squares sense with application in blind source separation," *IEEE Trans. Signal Processing*, vol. 50, July 2002.

47. B. Yang, "Projection approximation subspace tracking," *IEEE Trans. Signal Processing*, vol. 43, pp. 95–107, January 1995.

48. P. Comon and G.H. Golub, "Tracking a few extreme singular values and vectors in signal processing," *Proc. IEEE*, vol. 78, pp. 1327–1343, August 1990.

49. S.C. Douglas, "Combined subspace tracking, prewhitening, and contrast optimization for noisy blind signal separation," in *Proc. 2nd int. workshop Indept. Component Anal. Source Sep.*, (Helsinki, Finland)), pp. 579–584, June 2000.

50. S.C. Douglas, "Numerically-robust $O(N^2)$ RLS algorithms using least-squares prewhitening," in *IEEE Intl. Conf. Acoustics, Speech, Signal Processing (ICASSP)*, (Istanbul, Turkey), pp. 412–415, June 2000.

51. S. Attallah and K. Abed-Meraim, "Fast algorithms for subspace tracking," *IEEE Signal Processing Letters*, vol. 8, pp. 203–206, July 2001.

52. K. Abed-Meraim, S. Attallah, A. Chkeif, and Y. Hua, "Orthogonal Oja algorithm," *IEEE Signal Processing Letters*, vol. 7, pp. 116–119, May 2000.

53. S.C. Douglas, S.-Y. Kung, and S. Amari, "A self-stabilized minor subspace rule," *IEEE Signal Processing Letters*, vol. 5, pp. 328–330, December 1998.

54. S.C. Douglas and X. Sun, "A projection approximation minor subspace tracking algorithm," in *9th IEEE DSP Workshop*, (Hunt, TX), October 2000.

55. E. Oja, "Principal components, minor components, and linear neural networks," *Neural Networks*, vol. 5, pp. 927–935, November 1992.

56. H. Krim and M. Viberg, "Two decades of array signal processing research: The parametric approach," *IEEE Signal Processing Magazine*, vol. 13, pp. 67–94, July 1996.

57. H.L. van Trees, *Optimum Array Processing (part IV of Detection, Estimation, and Modulation Theory)*. New York: Wiley Interscience, 2002.

58. A.V. Keerthi, A. Mathur, and J.J. Shynk, "Direction-finding performance of the multistage CMA array," in *28-th Asilomar Conf. Signals, Syst. Comp.*, pp. 847–852 vol.2, IEEE, 1994.

59. J.J. Shynk, A.V. Keerthi, and A. Mathur, "Steady state analysis of the multistage constant modulus array," *IEEE Trans. Signal Processing*, vol. 44, pp. 948–962, April 1996.

60. A. Leshem and A.J. van der Veen, "Direction-of-arival estimation for constant modulus signals," *IEEE Trans. Signal Processing*, vol. 47, November 1999.

61. P. Stoica and O. Besson, "Maximum likelihood DOA estimation for constant-modulus signal," *Electronics Letters*, vol. 36, pp. 849–851, April 2000.

62. A. Leshem, "Maximum likelihood separation of phase modulated signals," in *IEEE Intl. Conf. Acoustics, Speech, Signal Processing (ICASSP)*, March 1999.

63. A. Leshem, "Maximum likelihood estimation of constant modulus signals," *IEEE Trans. Signal Processing*, vol. 48, pp. 2948–2952, October 2000.

64. B.M. Sadler, R.J. Kozick, and T. Moore, "Bounds on bearing and symbol estimation with side information," *IEEE Trans. Signal Processing*, vol. 49, pp. 822–834, April 2001.

65. P. Stoica and A. Nehorai, "MUSIC, maximum likelihood, and Cramer-Rao bound," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. 37, pp. 720–743, May 1989.

66. A.J. van der Veen, "Blind source separation based on combined direction finding and constant modulus properties," in *IEEE SP workshop on Stat. Signal Array processing*, (Portland, OR), September 1998.

67. P. Gill, W. Murray, and M.H. Wright, *Practical Optimization.* Academic Press, 1981.

68. R. Roy, A. Paulraj, and T. Kailath, "ESPRIT—A subspace rotation approach to estimation of parameters of cisoids in noise," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. 34, pp. 1340–1342, October 1986.