

# Chapter 2

## Jamming Suppression

### 2.1 Introduction

The vulnerabilities of GNSS systems expose them to impacts from various intentional and unintentional interferences. Of these various interferences, jamming is the most common type, which mainly includes narrowband jamming and wideband jamming. Even though GNSS adopts spread spectrum technology so that it has certain capability of interference mitigation, this capability is constrained by spreading gain. Usually the power of jamming is much larger than the power of the GNSS signal. Under these conditions, the jamming power, even after de-spreading, is still much larger than the signal power, and consequently a receiver cannot acquire GNSS signal.

Techniques used for GNSS jamming mitigation can be roughly divided into three categories: temporal domain and frequency domain filtering techniques, spatial domain filtering techniques, and space-time domain joint filtering techniques. Temporal domain and frequency domain filtering techniques [1–7] have the advantages of low cost and simplicity. But when there are multiple narrowband jammings or one single wideband jamming (temporal domain wideband signal), these techniques cannot deliver effective jamming suppression performance. Space filtering technique is a commonly used jamming mitigation approach. It can effectively suppress narrowband and wideband jammings (array wideband signals) using adaptive antenna arrays. Existing space filtering techniques include power minimization algorithm (also known as power inversion algorithm) [8–11], Capon algorithm [12, 13], and blind adaptive beamforming method using GNSS signal characteristic (the known spread spectrum code characteristic of periodic repetition) [14, 15]. However, only space filtering technique consumes one degree of freedom to suppress one narrowband jamming. To suppress wideband jamming, the number of antenna array elements needs to be increased. This increases the cost of GNSS receivers and also is difficult to implement in the case of restricted antenna aperture (e.g. airborne and missile-borne conformal arrays). Space Time Adaptive

Processing (STAP), which was originally designed for Airborne Early Warning (AEW) radar to suppress ground clutter [16, 17], can be used to solve the above problem. This technique, under the premise of not increasing the number of array elements, can increase the degree of freedom of an adaptive filtering system by using multiple taps.

Due to the limitations of temporal domain and frequency domain techniques, this chapter only briefly introduces the temporal domain and frequency domain filtering techniques, and highlights the spatial domain and space-time filtering techniques. First, we introduce the general model for array signal processing, then we introduce the conventional spatial domain adaptive filtering technique. Secondly we study two methods that use array signal processing gain to improve GNSS signal's carrier-to-noise ratio (CNR) [18]. Based on this foundation, we list an implementation scheme for a type of digital multi-beam jamming mitigation receiver and its experiment results. Finally, we study the space-time adaptive filtering algorithm and the corresponding reduced rank algorithm and equalization algorithm.

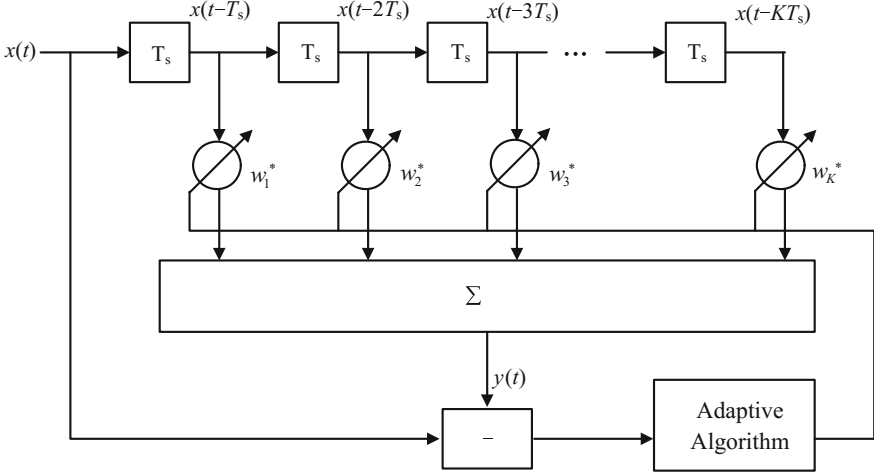
## 2.2 Temporal Domain and Frequency Domain Adaptive Filtering

Temporal domain and frequency domain filtering methods were the earliest GNSS jamming mitigation algorithms. The temporal domain filtering method suppresses the jamming signal by designing adaptive temporal domain filtering. Frequency domain filtering method first converts the signal into frequency domain for processing, then a threshold can be set, and all spectrum values larger than the threshold are set to be zero to suppress jamming signals. Compared with the temporal domain filtering algorithm, the frequency domain processing is easier to be implemented, so it is a commonly used jamming suppression technique at the present time.

### 2.2.1 Temporal Domain Filtering

When  $Q$  temporal domain narrowband jamming sources exist, the model of the received signals by a GNSS receiver is

$$x(t) = \sum_{l=1}^L s_l(t) + \sum_{q=1}^Q j_q(t) + e(t) \quad (2.1)$$



**Fig. 2.1** Block diagram of an adaptive FIR filter

where  $s_l(t) = A_l D_l(t - \tau_l) c_l(t - \tau_l) e^{j\omega_{dl}t}$  represents the  $l$ th GNSS signal ( $l = 1, 2, \dots, L$ );  $\omega_{dl}$  is the angular frequency of the signal;  $A_l$  is the carrier amplitude;  $D_l(t)$  is the navigation data bit information;  $c_l(t)$  is the C/A code for the  $l$ th satellite signal;  $\tau_l$  is the propagation for the  $l$ th satellite signal time delay;  $j_q(t)$  is the  $q$ th jamming signal ( $q = 1, 2, \dots, Q$ );  $e(t)$  is the thermal noise of the receiver (usually it is additive Gaussian white noise with zero mean and  $\sigma_e^2$  variance). We can assume that the GNSS signal, jamming signal, and noise are not correlated with each other.

Among the signals received by the receiver, the GNSS signal and noise are temporal domain wideband signals, and the correlation between their sample values is relatively small. But the jamming signal is a narrowband signal, so its sample values have a very strong correlation. At the same time, the power of GNSS signal is much smaller than the receiver's noise power (by about 20 dB). The sum of both can be approximated as receiver thermal noise. Therefore the jamming signal can be estimated by designing an adaptive linear prediction filter, as a result the jamming signal can be subtracted from the received signal [19]. This type of filter usually adopts the non-recursive transversal structure, also known as adaptive Finite Impulse Response (FIR) filter. Figure 2.1 shows the block diagram of an adaptive FIR filter used for GNSS receivers.

In Fig. 2.1, given the number of FIR taps is  $K$ , the input signal vector of the filter can be denoted as

$$\mathbf{x}(t) = [x(t - T_s), x(t - 2T_s), \dots, x(t - KT_s)]^T \quad (2.2)$$

where  $T_s$  is the sample time;  $(\bullet)^T$  represents the matrix transpose operation. Assuming the filter's weighted vector is

$$\mathbf{w} = [w_1, w_2, \dots, w_K]^T \quad (2.3)$$

The filter's output signal is

$$y(t) = \mathbf{w}^H \mathbf{x}(t) \quad (2.4)$$

where  $(\bullet)^H$  represents the conjugate transpose operation. To suppress the jamming signal, we can select a filter's weighted vector to make the filter's output signal  $y(t)$  the estimated jamming signal. Since the GNSS signal is very weak, and the jamming signal is a strong narrowband signal, the input signal  $x(t)$  can be used as the reference signal  $d(t)$ . By using the Minimum Mean Square Error (MMSE) for filter design, the weighted vector can be determined. The cost function can be defined as

$$F(\mathbf{w}) = E\left\{|x(t) - \mathbf{w}^H \mathbf{x}(t)|^2\right\} \quad (2.5)$$

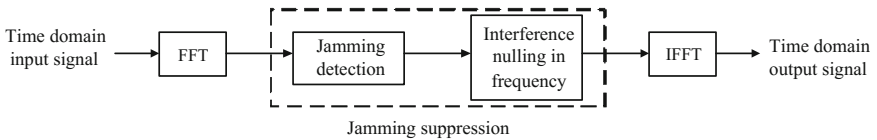
where  $E\{\bullet\}$  denotes mathematical expectation. By minimizing  $F(\mathbf{w})$ , we can derive

$$\mathbf{w}_{opt} = \mathbf{R}_x^{-1} \mathbf{r}_{xd} \quad (2.6)$$

where  $\mathbf{R}_x = E\{\mathbf{x}(t)\mathbf{x}^H(t)\}$  is the autocorrelation matrix of the filter input signal;  $\mathbf{r}_{xd} = E\{\mathbf{x}(t)d^*(t)\}$  is the cross-correlation between the filter input signal vector and the reference signal; and  $(\bullet)^*$  denotes the conjugate operation. At present time, many algorithms can be chosen to solve Eq. (2.6), and the most representative methods include Levinson-Durbin algorithm, Burg algorithm, Least Mean Square (LMS) algorithm, recursive least squares algorithm etc. The details of these algorithms can be found in Ref. [20], and we will not elaborate on them in this chapter.

### 2.2.2 Frequency Domain Filter

A simple type of frequency domain method can be adopted for narrowband jamming suppression, and its block diagram is shown in Fig. 2.2. The algorithm first performs Fast Fourier Transform (FFT) on the input signal. Then it detects the jamming signals and suppresses them by setting the corresponding spectral values in the frequency domain to zero. Finally it converts the signal back into temporal

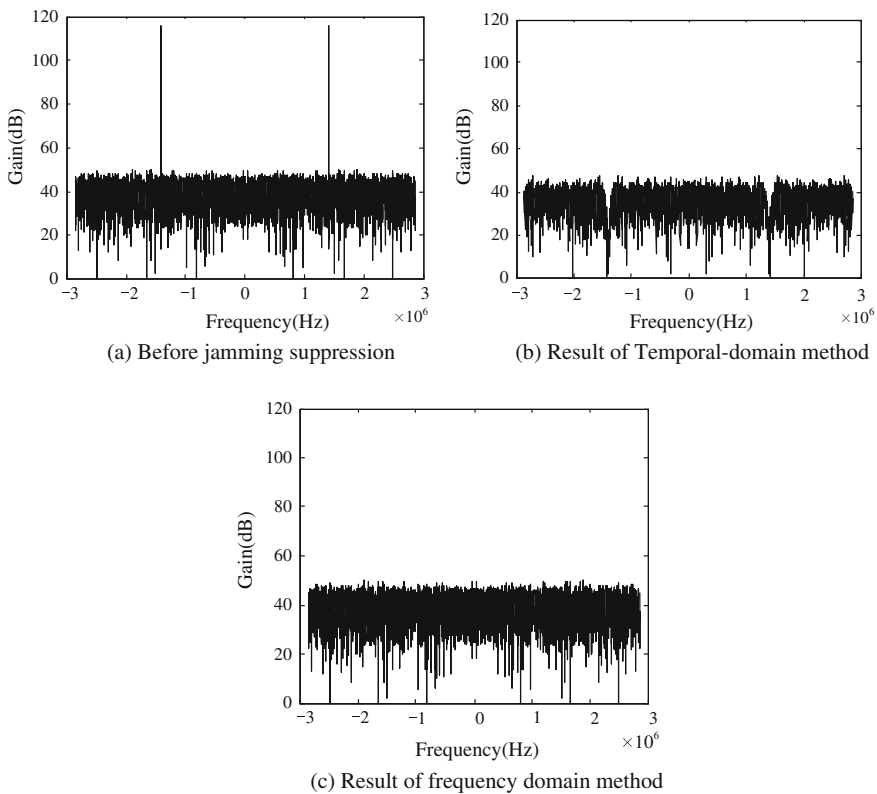


**Fig. 2.2** Frequency domain jamming suppression block diagram

domain using Inverse Fast Fourier Transform (IFFT), and sends the converted signal into a common receiver for positioning calculation [21, 22].

In Fig. 2.2, the frequency domain processing performs jamming detection first, and then sets the corresponding spectral values to zero. Using a method similar to the Constant False Alarm Rate (CFAR) method used in radar application (more details can be found in Chap. 6 of this book), a threshold can be pre-determined, and the spectral values after FFT can be compared with the threshold. If a value surpasses the threshold, it can then be regarded as a jamming signal. The jamming signal can then be suppressed by performing zero-setting on the detected spectral values in the frequency domain.

When multiple narrowband jammings exist, even if we can suppress narrowband jamming by using zero-value setting in the frequency domain, the method could cause some loss on the GNSS signal. In particular, if the jamming signal is a co-channel wideband signal related to the GNSS signal, the GNSS signal can be filtered out using this method. Therefore even though the temporal domain and frequency domain methods are easy to implement, their performances can



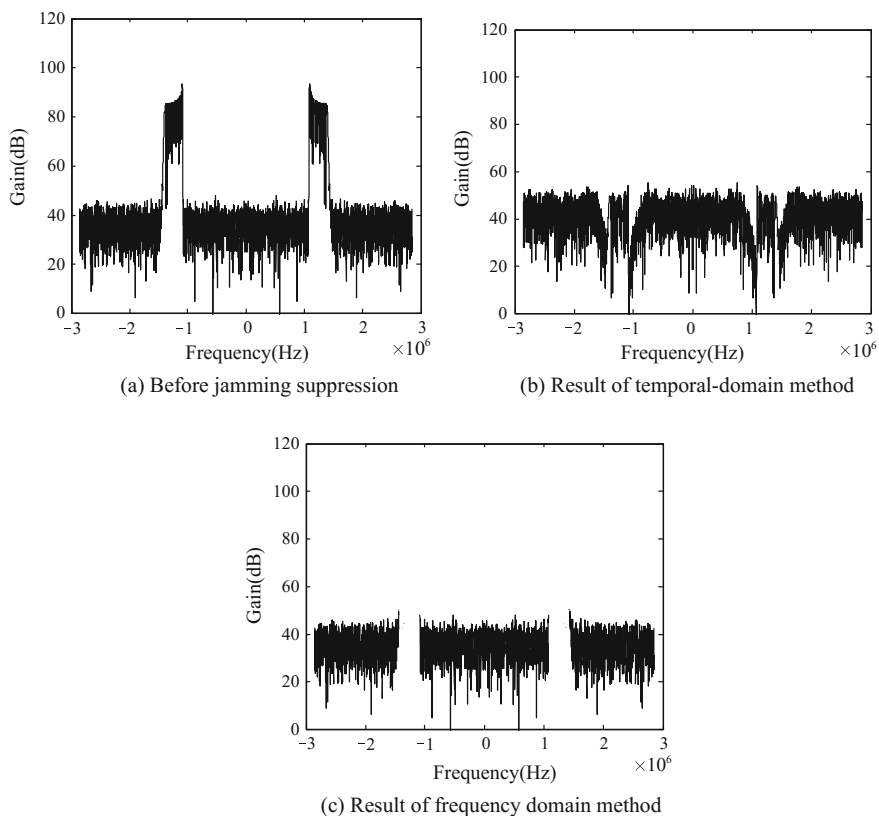
**Fig. 2.3** Comparisons of signal spectrums before and after using temporal domain and frequency domain methods (single frequency jamming scenario)

drastically degrade, or even totally lose effectiveness in the cases of multiple narrowband jammings or a wideband jamming.

When a jamming signal's frequency changes over time (e.g. a linear frequency modulated signal), a more effective method is to use time-frequency analysis tool [23] to separate the jamming signal and the GNSS signals, and then perform jamming suppression. References [24–27] studied the time-frequency domain GNSS jamming mitigation method based on Short Time Fourier Transformation, and subspace techniques etc.

### 2.2.3 Simulation Results

In this section we verify the effectiveness of temporal domain and frequency domain methods through simulations. For the first simulation, the received signals include one GPS signal and one single frequency jamming signal with 40 dB



**Fig. 2.4** Comparisons of signal spectrums before and after using temporal domain and frequency domain methods (0.3 MHz jamming bandwidth scenario)

jamming-to-noise ratio,  $-20$  dB signal-to-noise ratio and the order of temporal domain filter is 20. The received signal has a digital intermediate frequency of 4.309 MHz, with a sampling rate of 5.714 MHz. Figure 2.3 compares the temporal domain and frequency domain methods. It can be seen in Fig. 2.3 that for single frequency jamming, both methods can effectively suppress the jamming signal.

For the second simulation, the received signals include a jamming signal with an approximately 0.3 MHz bandwidth, and the other simulation conditions are the same as those for the first simulation. Figure 2.4 compares the temporal domain and frequency domain methods. It can be seen in Fig. 2.4 that when the jamming signal has wider bandwidth, the temporal domain processing cannot effectively suppress the jamming signal, and even though the frequency domain processing may suppress the jamming signal, it could cause significant satellite signal loss.

## 2.3 Conventional Spatial Domain Adaptive Filtering

Temporal domain and frequency domain methods have the advantages of low cost and simplicity. But since they lack the capability to distinguish desired signal and jamming signal in the spatial domain, they cannot cope with a large number of narrowband jammings and wideband jammings. Spatial domain methods can distinguish spatial directions of GNSS signals and jamming signals, so they have become an effective means used for GNSS jamming mitigation. Among them, the most widely used is the power minimization algorithm [9], also known as the power inversion algorithm. The algorithm uses array antennas to suppress jamming. Before we introduce the power minimization algorithm in this section, we first introduce the basic principle and methodology for adaptive array processing.

### 2.3.1 Adaptive Array Overview

Array processing mainly processes the signal in the spatial domain, including signal filtering, signal detection, and parameter estimation. It has a dual relationship with the temporal domain FIR filter. FIR filter performs discrete sequential sampling on the temporal domain signal, while array processing performs discrete parallel sampling on the spatial domain signal. Usually we can assume the space signal is a far field incident signal. The so-called far field means that the signal source is far enough from the array that the wave front arrives at the array can be approximated as a plane wave. A near field signal, on the other hand, can be regarded as a spherical wave.

For array processing, whether a signal is narrowband or wideband is determined relative to the array itself, which is different from the usual criterion of using the ratio between the signal bandwidth and the center frequency. A coherent array corresponds to a narrowband signal and a non-coherent array corresponds to a

wideband signal. If an array performs as a coherent array towards a certain signal, then the maximum time difference for the signal to arrive at various array elements should be small enough so that the signal envelopes received by various array elements are consistent. Assuming  $\tau_{\max}$  represents the longest time difference, and  $B_w$  represents the signal bandwidth, then a coherent array should satisfy the following condition:

$$\tau_{\max} \ll \frac{1}{B_w} \quad (2.7)$$

i.e. the time for the signal to go through the array is much shorter than the equivalent time width of the signal. Otherwise, it should be regarded as a non-coherent array.

By considering an arbitrary array composed of  $M$  array elements in the far field, there is a narrowband signal incidenting on the array, with an incident polar angle  $\theta$ , and an azimuth  $\phi$ , as shown in Fig. 2.5. Each array element is assumed to be isotropic, and point  $O$  is the reference point for array reception data.

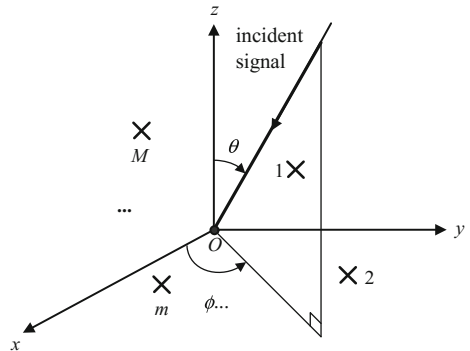
Under the assumed condition of far field narrowband, the received signal by the array can be represented as

$$\mathbf{x}(t) = \mathbf{a}(\theta, \phi)s(t) + \mathbf{e}(t) \quad (2.8)$$

where  $\mathbf{x}(t) = [x_1(t), x_2(t), \dots, x_M(t)]^T$  is an  $M \times 1$  dimension array data vector (the signal sample obtained by performing a single sampling pass on the signal is called the single snapshot data vector);  $\mathbf{e}(t)$  is a  $M \times 1$  dimension array representing the received noise vector; it can usually be assumed to follow a Gaussian distribution;  $s(t)$  is the complex envelop of the signal;  $\mathbf{a}(\theta, \phi)$  is the signal's steering vector, and

$$\mathbf{a}(\theta, \phi) = [e^{-j\mathbf{u}^T \mathbf{p}_1}, e^{-j\mathbf{u}^T \mathbf{p}_2}, \dots, e^{-j\mathbf{u}^T \mathbf{p}_M}]^T \quad (2.9)$$

**Fig. 2.5** Received signal by an arbitrary array





where

$$\mathbf{u} = \frac{2\pi}{\lambda} \begin{bmatrix} \sin \theta \cos \phi \\ \sin \theta \sin \phi \\ \cos \theta \end{bmatrix} \quad (2.10)$$

is the wave number vector.

$$\mathbf{p}_m = [x_m, y_m, z_m]^T, \quad m = 1, 2, \dots, M \quad (2.11)$$

is the three-dimensional position vector of the  $m$ th array element.

For array signal processing, a uniform linear array is a relatively common form of array structure. For convenience, unless otherwise specified, in this chapter we use a uniform linear array as the example to discuss adaptive array processing algorithm. For a uniform linear array, we usually can assume the first array element as a reference point, and then we have

$$\mathbf{a}(\theta) = \left[ 1, e^{\frac{-j2\pi d \sin \theta}{\lambda}}, \dots, e^{\frac{-j2\pi(M-1)d \sin \theta}{\lambda}} \right]^T \quad (2.12)$$

where  $\theta$  represents the angle between the incident signal and the array's normal line;  $d$  represents the distance between two array elements;  $\lambda$  represents the wavelength of the incident signal.

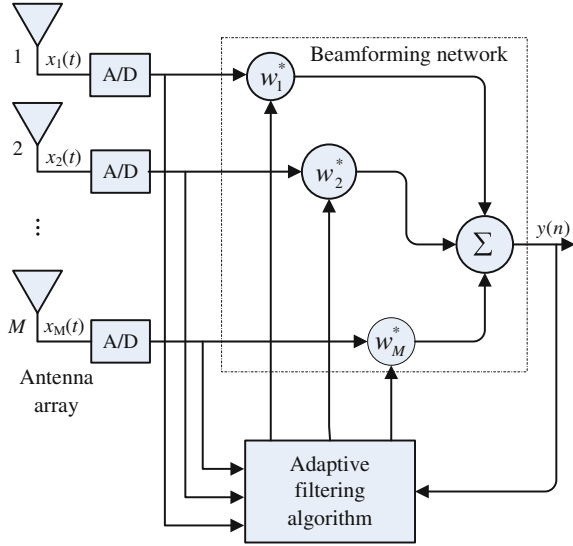
If a desired signal and  $Q$  jamming signals incident on a uniform linear array, and their arrival angles are  $\theta_0$  and  $\theta_q$  ( $q = 1, 2, \dots, Q$ ) respectively, then the array's received signal model becomes

$$\mathbf{x}(t) = \mathbf{a}(\theta_0)s_0(t) + \sum_{q=1}^Q \mathbf{a}(\theta_q)j_q(t) + \mathbf{e}(t) \quad (2.13)$$

Adaptive array processing is also known as spatial domain adaptive filtering, and it is one of the most important research directions of array signal processing (the other direction is super-resolution direction of arrival estimation). Its basic principle is shown in Fig. 2.6.

In Fig. 2.6,  $\mathbf{w} = [w_1, w_2, \dots, w_M]^T$  is an array weight vector. The weight vector performs weighted sum on the received array signals (beamforming), and the output is used as the array output signal. The key of adaptive array processing is to select a proper array weight vector to filter out the jamming signal and preserve the desired signals. Various adaptive array processing methods differ on their algorithms of calculating array weight vector. Usually we can intuitively evaluate the performance of an adaptive array processing method using array pattern. Array pattern is

**Fig. 2.6** Adaptive array processing



defined as the vector weight vector's responses to signals arrived from different angles, i.e.

$$F(\theta) = \mathbf{w}^H \mathbf{a}(\theta) \quad (2.14)$$

### 2.3.2 Statistical Optimal Beamforming

Statistical optimal beamforming, under the assumption of accurately knowing the received array signal's second order statistics, calculates an array's weight vector to adaptively form null steering towards the direction of jamming. Therefore it can guarantee the reception of desired signals. However, in actual application, an array signal's second order statistics usually cannot be obtained based on finite snapshot data, therefore an optimal weight vector cannot be obtained. The beamforming based on finite snapshot data is usually called adaptive beamforming. Statistical optimal beamforming is an analysis tool for adaptive array signal processing, and it provides a theoretical foundation for implementing adaptive beamforming.

The statistical optimal beamforming problem can be summarized as solving for an optimal weight vector based on certain criteria. Usually these criteria are Maximum Signal to Interference add Noise Ratio (MSINR), Minimum Mean Square Error (MMSE), and Minimum Noise Variance (MNV). It can be proved that these three criteria are equivalent under certain conditions [28].

### 1. Maximum SINR Criterion

Let  $\mathbf{w}$  represents the array weight array, and  $y(t)$  represents array output, then

$$y(t) = \mathbf{w}^H \mathbf{x}(t) \quad (2.15)$$

By substituting (2.13) into (2.15), we can obtain the desired signal and the residue jamming plus noise from the array output.

$$y_s(t) = \mathbf{w}^H \mathbf{a}(\theta_0) s_0(t) \quad (2.16)$$

$$y_{j+e}(t) = \mathbf{w}^H \left( \sum_{q=1}^Q \mathbf{a}(\theta_q) j_q(t) + \mathbf{e}(t) \right) \quad (2.17)$$

Then the corresponding output powers are

$$\sigma_{os}^2 = E \left\{ y_s(t) y_s^*(t) \right\} = \mathbf{w}^H \sigma_s^2 \mathbf{a}(\theta_0) \mathbf{a}^H(\theta_0) \mathbf{w} \triangleq \mathbf{w}^H \mathbf{R}_s \mathbf{w} \quad (2.18)$$

$$\sigma_{o(j+e)}^2 = E \left\{ y_{j+e}(t) y_{j+e}^*(t) \right\} = \mathbf{w}^H \left( \sum_{q=1}^Q \sigma_q^2 \mathbf{a}(\theta_q) \mathbf{a}^H(\theta_q) + \sigma_e^2 \mathbf{I} \right) \mathbf{w} \triangleq \mathbf{w}^H \mathbf{R}_{j+e} \mathbf{w} \quad (2.19)$$

In (2.18) and (2.19),  $\mathbf{R}_s$  and  $\mathbf{R}_{j+e}$  are covariance matrices for the desired signal and jamming plus noise;  $\sigma_s^2$  is the desired signal power;  $\sigma_q^2$  is the power of the  $q$ th jamming signal. The maximum SINR Criterion looks for array weight vector, to maximize the ratio between the array output signal power and the output jamming plus noise power, i.e.

$$\max_{\mathbf{w}} \frac{\mathbf{w}^H \mathbf{R}_s \mathbf{w}}{\mathbf{w}^H \mathbf{R}_{j+e} \mathbf{w}} \quad (2.20)$$

where  $\max_{\mathbf{w}}[\bullet]$  represents the selected weight value that maximizes the  $[\bullet]$ . Using the Lagrange multiplier method, we can derive that the solution for (2.20) is equivalent to the solution for the following generalized eigenvalue problem, i.e.

$$\mathbf{R}_s \mathbf{w} = \lambda \mathbf{R}_{j+e} \mathbf{w} \quad (2.21)$$

Therefore, the optimum weight vector  $\mathbf{w}_{opt}$  is the corresponding eigenvector for the maximum eigenvalue in (2.21). It can be proved that, the solution of (2.21) is equivalent to

$$\mathbf{w}_{opt} = \gamma \mathbf{R}_{j+e}^{-1} \mathbf{a}(\theta_0) \quad (2.22)$$

where  $\gamma$  is a constant factor.

## 2. Minimum Mean Square Error Criterion

Minimum mean square error criterion uses a reference signal to solve for the array weight vector, i.e. to select a weight vector to minimize the mean square error between the reference signal and the array output

$$\min_{\mathbf{w}} E\left\{|d(t) - \mathbf{w}^H \mathbf{x}(t)|^2\right\} \quad (2.23)$$

where  $d(t)$  is the reference signal. We define the covariance matrix of the array received data as  $\mathbf{R}_x = E\{\mathbf{x}(t)\mathbf{x}^H(t)\}$ , then the solution (2.23) is

$$\mathbf{w}_{opt} = \mathbf{R}_x^{-1} \mathbf{r}_{xd} \quad (2.24)$$

where  $\mathbf{r}_{xd} = E\{\mathbf{x}(t)d^*(t)\}$  is the cross-correlation between the array received signal and the reference signal. Equation (2.24) is the solution of Wiener-Hoff equation in matrix form and it is the optimal Weiner solution.

If the useful signal  $s_0(t)$  in (2.13) is used as the reference signal  $d(t)$ , since the desired signal, jamming signal and noise are not correlated to each other, then

$$\mathbf{r}_{xd} = \mu \mathbf{a}(\theta_0) \quad (2.25)$$

where  $\mu = E\{s_0(t)d^*(t)\}$ . By substituting (2.25) into (2.24), we can have

$$\mathbf{w}_{opt} = \mu \mathbf{R}_x^{-1} \mathbf{a}(\theta_0) \quad (2.26)$$

## 3. Minimum Noise Variance Criterion

Minimum noise variance criterion selects the optimal weight vector by minimizing the array output power. To avoid the case of  $\mathbf{w}_{opt} = 0$ , some corresponding constraints need to be added. A usual constraint is to guarantee that desired signals pass without distortion, i.e. the array's response on the desired signal is a constant. Therefore, the cost function for the minimum noise variance criterion is

$$\begin{aligned} \min_{\mathbf{w}} \quad & \mathbf{w}^H \mathbf{R}_x \mathbf{w} \\ \text{s.t.} \quad & \mathbf{w}^H \mathbf{a}(\theta_0) = 1 \end{aligned} \quad (2.27)$$

By using the method of Lagrange multiplier, the solution for (2.27) is

$$\mathbf{w}_{opt} = \frac{1}{\mathbf{a}^H(\theta_0) \mathbf{R}_x^{-1} \mathbf{a}(\theta_0)} \mathbf{R}_x^{-1} \mathbf{a}(\theta_0) \quad (2.28)$$

The optimal beamformer expressed by (2.28) is the Minimum Variance Distortless Response (MVDR) beamformer, also known as Capon beamformer.

To compare (2.26) and (2.28), it can be seen that the minimum mean square error criterion and the minimum noise variance criterion differ only by a constant factor. This value does not have impact on an array output's SINR, so that the minimum mean square error criterion and minimum noise variance criterion are equivalent to each other. Based on the matrix inverse lemma it can be proved that, under the conditions of accurately knowing the desired signal steering vector and covariance matrix, the minimum noise variance criterion and the maximum SINR criterion are equivalent, therefore the three statistically optimal beamforming criteria are equivalent to each other.

### 2.3.3 Power Minimization Algorithm

To compare the three criteria in Sect. 2.3.3, it can be seen that they all need to know the directions of the desired signal or a known reference signal. Therefore, if the two criteria of MSINR and MNV can be used for GNSS jamming mitigation, the directions of the GNSS signals need to be known so that the implementation is relatively complex. If the MMSE criterion is used, then the GNSS signal needs to be known.

Using the GPS system as an example, when the GPS signal and  $Q$  jamming signals incident on an  $M$ -element uniform linear array, the received signal at time  $t$  is

$$\mathbf{x}(t) = \sum_{l=1}^L \mathbf{a}(\theta_l) s_l(t) + \sum_{q=1}^Q \mathbf{a}(\theta_q) j_q(t) + \mathbf{e}(t) \quad (2.29)$$

where  $s_l(t)$  represents the  $l$ th GNSS signal ( $l = 1, 2, \dots, L$ );  $\theta_l$  is the  $l$ th GPS signal's direction of arrival;  $\mathbf{a}(\theta_l)$  is the corresponding array steering vector;  $j_q(t)$  represents the  $q$ th jamming signal ( $q = 1, 2, \dots, Q$ );  $\theta_q$  is the  $q$ th jamming signal's direction of arrival;  $\mathbf{a}(\theta_q)$  is its corresponding array steering vector;  $\mathbf{e}(t)$  represents the thermal noise vector of the receiver, and is usually an additive Gaussian white noise vector with zero mean and  $\sigma_e^2$  variance, assuming that the GNSS signal, jamming signal and noise do not correlate with each other.

Since the GNSS signal is very weak and buried in the noise (usually around 20 dB lower than the noise), we only need to consider jamming mitigation, i.e. forming a null steering on the direction of jamming without considering where the direction of the beam is pointing. This is the power minimization algorithm that has been widely applied for GNSS jamming mitigation. The algorithm calculates array weight vector by minimizing the output power, but prevents the weight vector from

having a zero solution by guaranteeing that the reference antenna's signal in the antenna array has no distorted output, i.e.

$$\begin{aligned} \min_{\mathbf{w}} \mathbf{w}^H \mathbf{R}_x \mathbf{w} \\ s.t. \mathbf{w}^H \boldsymbol{\delta}_M = 1 \end{aligned} \quad (2.30)$$

where  $\boldsymbol{\delta}_M = [1, 0, \dots, 0]^T$  is a  $M \times 1$  dimension vector. The solution for (2.30) can be obtained using the method of Lagrange multiplier

$$\mathbf{w}_{opt} = \frac{1}{\boldsymbol{\delta}_M^H \mathbf{R}_x^{-1} \boldsymbol{\delta}_M} \mathbf{R}_x^{-1} \boldsymbol{\delta}_M \quad (2.31)$$

where  $\mathbf{R}_x$  represents the theoretical array covariance matrix, in reality it usually is replaced by the sample covariance matrix defined as

$$\hat{\mathbf{R}}_x = \frac{1}{N} \sum_{n=1}^N \mathbf{x}(n) \mathbf{x}^H(n) \quad (2.32)$$

where  $N$  represents the number of snapshots.

Power minimization algorithm can adaptively form a null steering towards the jamming direction and there is no need to know the direction of the GNSS signal, and no need to know the array manifold. Therefore this algorithm belongs to the category of blind adaptive beamforming. The implementation is simple and it is compatible with common receivers. But the algorithm can only suppress jamming, and cannot aim the antenna pattern's main lobe towards the GNSS signal. Consequently, it cannot provide signal processing gain brought by the antenna array, and cannot enhance the GNSS signal's carrier-to-noise ratio. The carrier-to-noise ratio is a very important parameter for the receiver, and the receiver's positioning accuracy is highly correlated with this parameter [29]. When the carrier-to-noise ratio is low, the position accuracy is usually worse.

## 2.4 Spatial Domain Adaptive Filtering Using Periodic Repetitive Characteristic of GNSS Signals

The power minimization algorithm is easy to implement, but cannot improve the carrier-to-noise ratio of a GNSS signal. In this section, on the foundation of the GNSS signal's periodic repetitive characteristic, we discuss the algorithm that uses the signal processing gain brought by antenna arrays to improve GNSS signal carrier-to-noise ratio.

### 2.4.1 Periodic Repetitive Characteristic of GNSS Signals

The C/A code of GPS signal is a Gold code with 1.023 Mcps chip rate, with a 1 ms chip period, i.e. every period contains 1023 chips. For a civilian GPS signal, the C/A code modulates the D code, and the D code chip rate is 50 Hz. Therefore within a D code period, 20 repetitive cycles of the C/A code exist. Since a D code does not change within one period, the corresponding 20 spread spectrum codes have the same structure, i.e. showing the characteristic of periodic repetition. The periodic repetition characteristic of the C/A code for a GPS signal is a special case of signal cyclostationarity. Below we introduce the signal's cyclostationarity characteristic first.

For a random signal  $x(t)$ , if for a certain time delay  $\tau$ , the correlation value between  $x(t)$  and the obtained signal after frequency shift  $x(t)$  by  $\alpha$  is not 0, then we can say that  $x(t)$  has cyclostationarity characteristic [30], i.e.

$$r_{xx}^{\alpha}(\tau) \triangleq \frac{\langle x(t)[x(t-\tau)e^{j2\pi\alpha t}]^* \rangle_{\infty}}{\sqrt{\langle |x(t)|^2 \rangle_{\infty} \langle |x(t-\tau)e^{j2\pi\alpha t}|^2 \rangle_{\infty}}} = R_{xx}^{\alpha}(\tau)/R_{xx}(0) \neq 0 \quad (2.33)$$

where symbol  $\langle \rangle_{\infty}$  represents the average of infinite time observation domain;  $\alpha$  is defined as the cycle frequency;  $R_{xx}^{\alpha}(\tau)$  is a cycle correlation function of the signal  $x(t)$  that can be defined as  $R_{xx}^{\alpha}(\tau) = \langle x(t)[x(t-\tau)e^{j2\pi\alpha t}]^* \rangle_{\infty}$ , and  $R_{xx}(0) = \langle |x(t)|^2 \rangle_{\infty}$  is the signal power.

Similarly, if for a certain time delay  $\tau$ , the correlation value between  $x(t)$  and the signal obtained by frequency shifting its conjugate signal by  $\alpha$  is not 0, then  $x(t)$  has conjugate cyclostationary characteristic, i.e.

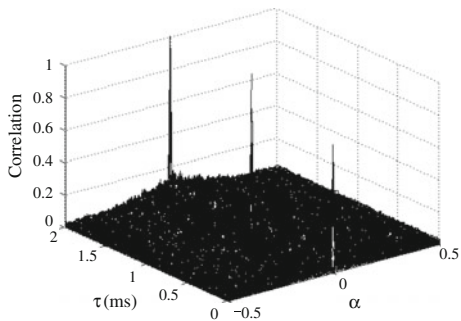
$$r_{xx^*}^{\alpha}(\tau) \triangleq \frac{\langle x(t)[x^*(t-\tau)e^{j2\pi\alpha t}]^* \rangle_{\infty}}{\sqrt{\langle |x(t)|^2 \rangle_{\infty} \langle |x^*(t-\tau)e^{j2\pi\alpha t}|^2 \rangle_{\infty}}} = R_{xx^*}^{\alpha}(\tau)/R_{xx}(0) \neq 0 \quad (2.34)$$

where  $R_{xx^*}^{\alpha}(\tau)$  is the conjugate cyclic correlation function of the signal  $x(t)$ . To simplify, we can denote  $R_{xx}^{\alpha}(\tau)$  and  $R_{xx^*}^{\alpha}(\tau)$  in a unified way:

$$R_{xu} = \begin{cases} R_{xx}^{\alpha}(\tau) & \text{when } u(t) = x(t-\tau)e^{j2\pi\alpha t} \\ R_{xx^*}^{\alpha}(\tau) & \text{when } u(t) = x^*(t-\tau)e^{j2\pi\alpha t} \end{cases} \quad (2.35)$$

Communication signals are usually assumed to be cyclostationary signals. GPS signals adopt a spread spectrum communication system that has the cyclostationary property as well. Based on what was described before, since the C/A code has periodic repetitions, we can regard the cycle frequency as  $\alpha = 0$ , and  $\tau$  is an integer

**Fig. 2.7** GPS C/A code signal cyclostationary correlation diagram



multiple of the C/A code period (1 ms). This is a special case of cyclostationary signal. Its cycle correlation function is shown in Fig. 2.7.

For array signal processing, in terms of a  $M \times 1$  dimension signal vector  $\mathbf{x}(t)$ , the cyclostationarity correlation matrix and the conjugate periodic stationary correlation matrix are defined as

$$\mathbf{R}_{xu} = \begin{cases} \mathbf{R}_{xx}^{\alpha}(\tau), & \text{when } \mathbf{u}(t) = \mathbf{x}(t-\tau)e^{j2\pi\alpha t} \\ \mathbf{R}_{xx^*}^{\alpha}(\tau), & \text{when } \mathbf{u}(t) = \mathbf{x}^*(t-\tau)e^{j2\pi\alpha t} \end{cases} \quad (2.36)$$

### 2.4.2 SCORE Algorithm

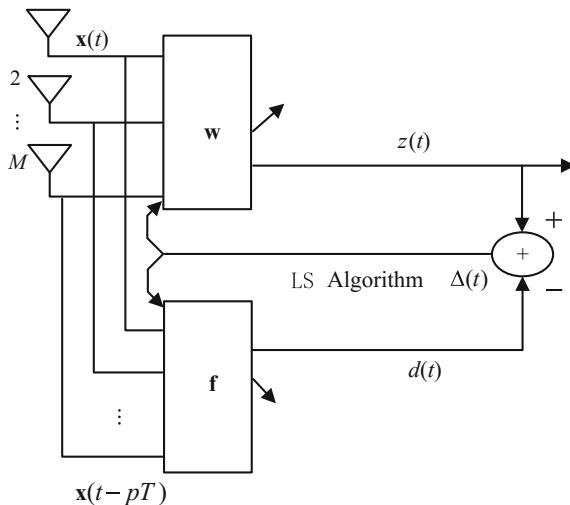
The R&D team, led by Professor Moeness at Villanova university in the US, proposed the Self-COherence Restoral (SCORE) jamming mitigation algorithm using periodic repetition characteristic of C/A codes [14, 31]. The schematic of the algorithm is shown in Fig. 2.8.

As shown in Fig. 2.8, the SCORE algorithm is composed of two beamformers. The main channel beamformer  $\mathbf{w}$  processes the data block signal  $\mathbf{x}(t)$ , while the auxiliary channel beamformer  $\mathbf{f}$  processes the reference block signal  $x(t - pT)$ . The reference block signal is the data after delaying the data block signal by  $p$  ( $1 \leq p \leq 19$ ) C/A code periods. Since a navigation data bit includes 20 C/A code periods, we can assume that the data block data and the reference data block are located inside the same navigation data bit. The structure can be shown in Fig. 2.9.

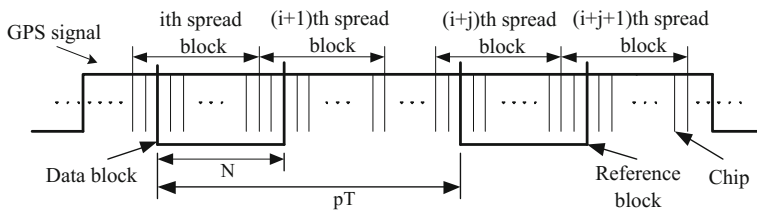
When only one GNSS signal is considered, (2.29) becomes

$$\mathbf{x}(t) = \mathbf{a}(\theta_0)s(t) + \sum_{q=1}^Q \mathbf{a}(\theta_q)j_q(t) + \mathbf{e}(t) \quad (2.37)$$





**Fig. 2.8** SCORE algorithm



**Fig. 2.9** Data block and reference block

SCORE algorithm assumes that the carrier waves have been synchronized completely (i.e. without the impacts from Doppler frequency shift), and the navigation data bit does not change (assuming  $D(t) = 1$ ), then (2.37) becomes

$$\mathbf{x}(t) = A\mathbf{a}(\theta_0)c(t - \tau) + \sum_{q=1}^Q \mathbf{a}(\theta_q)j_q(t) + \mathbf{e}(t) \quad (2.38)$$

Let the data after delaying  $p$  ( $1 \leq p \leq 19$ ) C/A code periods equals  $\mathbf{u}(t)$ , then

$$\mathbf{u}(t) = \mathbf{x}(t - pT) = A\mathbf{a}(\theta_0)c(t - \tau - pT) + \sum_{q=1}^Q \mathbf{a}(\theta_q)j_q(t - pT) + \mathbf{e}(t - pT) \quad (2.39)$$

Within a navigational data bit, the C/A code has the characteristic of periodic repetition, i.e.

$$c(t - \tau - pT) = c(t - \tau) \quad (2.40)$$

By substituting (2.40) into (2.39), we have

$$\mathbf{u}(t) = A\mathbf{a}(\theta_0)c(t - \tau) + \sum_{q=1}^Q \mathbf{a}(\theta_q)j_q(t - pT) + \mathbf{e}(t - pT) \quad (2.41)$$

In Fig. 2.8, the respective output signals for the main and auxiliary channel beamformers are:

$$z(t) = \mathbf{w}^H \mathbf{x}(t) \quad (2.42)$$

$$d(t) = \mathbf{f}^H \mathbf{u}(t) \quad (2.43)$$

where  $\mathbf{w}$  and  $\mathbf{f}$  represent the weight vectors for the main channel and auxiliary channel respectively. We can define

$$R_{zd} = E\{z(t)d^*(t)\} = \mathbf{w}^H E\{\mathbf{x}(t)\mathbf{u}^H(t)\} \mathbf{f} \quad (2.44)$$

$$R_{zz} = E\{z(t)z^*(t)\} = \mathbf{w}^H E\{\mathbf{x}(t)\mathbf{x}^H(t)\} \mathbf{w} \quad (2.45)$$

$$R_{dd} = E\{d(t)d^*(t)\} = \mathbf{f}^H E\{\mathbf{u}(t)\mathbf{u}^H(t)\} \mathbf{f} \quad (2.46)$$

Since the GNSS signal, jamming signal, and noise are independent from each other, then the received data's covariance matrix is

$$\mathbf{R}_x = E\{\mathbf{x}(t)\mathbf{x}^H(t)\} = \mathbf{R}_s + \mathbf{R}_j + \mathbf{R}_e \quad (2.47)$$

where  $\mathbf{R}_s$ ,  $\mathbf{R}_j$  and  $\mathbf{R}_e$  represent the covariance matrices for GNSS signal, jamming signal, and noise respectively. Similarly, the covariance matrix for the reference signal is

$$\mathbf{R}_u = E\{\mathbf{u}(t)\mathbf{u}^H(t)\} = \mathbf{R}_x \quad (2.48)$$

Since jamming signals and noises have no periodic repetition characteristic, the cross-covariance matrix between the received data and the reference data is

$$\mathbf{R}_{xu} = E\{\mathbf{x}(t)\mathbf{u}^H(t)\} = \mathbf{R}_s = \sigma_s^2 \mathbf{a}(\theta_0) \mathbf{a}^H(\theta_0) \quad (2.49)$$

where  $\sigma_s^2$  represents the GNSS signal's power. Let  $\Delta(t)$  represents the difference between the main channel's output signal and the reference signal, i.e.

$$\Delta(t) = z(t) - d(t) \quad (2.50)$$

By fixing  $\mathbf{w}$ ,  $E\{|\Delta(t)|^2\}$  can be minimized in the sense of least squares

$$\mathbf{f}_{LS} = \mathbf{R}_u^{-1} \mathbf{r}_{uz} \quad (2.51)$$

where

$$\mathbf{r}_{uz} = E\{\mathbf{u}(t)z^*(t)\} = \mathbf{R}_{xu}^H \mathbf{w} \quad (2.52)$$

Similarly, by fixing  $\mathbf{f}$ , we have

$$\mathbf{w}_{LS} = \mathbf{R}_x^{-1} \mathbf{R}_{xu}^H \mathbf{f} \quad (2.53)$$

SCORE algorithm maximizes the cross-correlation between the main channel's output signal and auxiliary channel's output signal in order to select the weight vector  $\mathbf{w}$ , then we have the following cost function

$$\max J(\mathbf{w}, \mathbf{f}) = \frac{|R_{zd}|^2}{R_{zz}R_{dd}} = \frac{|\mathbf{w}^H \mathbf{R}_{xu} \mathbf{f}|^2}{[\mathbf{w}^H \mathbf{R}_x \mathbf{w}][\mathbf{f}^H \mathbf{R}_u \mathbf{f}]} \quad (2.54)$$

By using  $\mathbf{f}_{LS}$ ,  $\mathbf{w}_{LS}$  to substitute  $\mathbf{f}$  and  $\mathbf{w}$  in (2.54), we can have

$$J(\mathbf{w}, \mathbf{f}_{LS}) = \frac{\mathbf{w}^H \mathbf{R}_{xu} \mathbf{R}_x^{-1} \mathbf{R}_{xu}^H \mathbf{w}}{\mathbf{w}^H \mathbf{R}_x \mathbf{w}} \quad (2.55)$$

$$J(\mathbf{f}, \mathbf{w}_{LS}) = \frac{\mathbf{f}^H \mathbf{R}_{xu} \mathbf{R}_x^{-1} \mathbf{R}_{xu}^H \mathbf{f}}{\mathbf{f}^H \mathbf{R}_u \mathbf{f}} \quad (2.56)$$

It can be proved that, to maximize (2.55) and (2.56) is equivalent to solve the corresponding eigenvectors for the following maximum generalized eigenvalue problem, i.e.

$$\mathbf{R}_x \mathbf{w} = \lambda \mathbf{R}_{xu} \mathbf{R}_x^{-1} \mathbf{R}_{xu}^H \mathbf{w} \quad (2.57)$$

$$\mathbf{R}_x \mathbf{f} = \kappa \mathbf{R}_{xu} \mathbf{R}_x^{-1} \mathbf{R}_{xu}^H \mathbf{f} \quad (2.58)$$

where  $\lambda$  and  $\kappa$  are eigenvalues.

By substituting (2.49) into (2.57), we can have

$$\mathbf{R}_x \mathbf{w} = \gamma \mathbf{R}_x \mathbf{w} \quad (2.59)$$

where  $\gamma$  is the eigenvalue. It can be known from (2.59) that the solution of (2.57) converges to the maximum signal-to-noise ratio criterion. Therefore the SCORE algorithm can make the array pattern's main beam point towards the direction of GNSS signal. It can be known from the derivation process of the SCORE algorithm that SCORE does not need to know any priori information on the array. Therefore, it is a blind adaptive beamforming technology. Nevertheless, SCORE algorithm only considers the scenario of one GNSS signal. Actually, a receiver can perform positioning calculation by receiving at least 4 or more GNSS signals. When  $L$  GNSS signals are received, (2.49) becomes

$$\mathbf{R}_{xu} = \sum_{l=1}^L \sigma_{sl}^2 \mathbf{a}(\theta_l) \mathbf{a}^H(\theta_l) \quad (2.60)$$

where  $\sigma_{sl}^2$  represents the power of the  $l$  ( $l = 1, 2, \dots, L$ )th GNSS signal;  $\mathbf{a}(\theta_l)$  represents the steering vector of the  $l$ th GNSS signal. Then, (2.57) has larger generalized eigenvalues. Consequently, the eigenvector corresponding to the largest generalized eigenvalue cannot ensure that the mainlobe of the pattern points to every GNSS signal's direction. Therefore it cannot guarantee that the receiver can capture all GNSS signals.

### 2.4.3 Single-Channel Single-Delay Crosscorrelation Algorithm

For the convenience of narration, (2.29) is re-written as below:

$$\mathbf{x}(t) = \sum_{l=1}^L \mathbf{a}(\theta_l) s_l(t) + \sum_{q=1}^Q \mathbf{a}(\theta_q) j_q(t) + \mathbf{e}(t) \quad (2.61)$$

Since the period of a navigation message is much larger than the period of C/A code, within the weight vector computation time we can assume that navigation message does not change. For the convenience of narration, we set  $D_l(t) = 1$ . The single channel single delay cross-correlation algorithm [18] takes advantage of the periodic repetitive characteristic of spread spectrum code to estimate the GNSS signal direction, then performs beamforming, thereby increases a GNSS signal's carrier-to-noise ratio.

#### 1. Jamming Mitigation Based on Subspace Projection Technique

When jamming exists, it is hard to estimate a GNSS signal's direction only relying on the periodic repetition characteristic of the spread spectrum code. Therefore the jamming signal must be suppressed first. Since the GNSS signal is

very weak, the array's covariance matrix is determined mainly by jamming and noise, i.e.

$$\mathbf{R}_x \approx \mathbf{R}_j + \mathbf{R}_e \quad (2.62)$$

Since jamming and noise are not correlated with each other, then we have

$$\mathbf{R}_x \approx \sum_{q=1}^Q \sigma_q^2 \mathbf{a}(\theta_q) \mathbf{a}^H(\theta_q) + \sigma_e^2 \mathbf{I}_{M \times M} \quad (2.63)$$

where  $\mathbf{I}_{M \times M}$  is an  $M \times M$  order unit array. To perform eigenvalue decomposition on  $\mathbf{R}_x$ , we can have

$$\mathbf{R}_x = \sum_{m=1}^M \lambda_m \mathbf{u}_m \mathbf{u}_m^H \quad (2.64)$$

where  $\lambda_m$  and  $\mathbf{u}_m (m = 1, 2, \dots, M)$  are  $\mathbf{R}_x$ 's eigenvalues and the corresponding eigenvectors, and the  $\lambda_m$  are in descending order. Assuming that all array element noises are independent from each other and have equal power, we can have

$$\lambda_1 \geq \lambda_2 \geq \dots \geq \lambda_Q \gg \lambda_{Q+1} = \dots = \lambda_M = \sigma_e^2 \quad (2.65)$$

Then (2.64) turns into

$$\mathbf{R}_x = \sum_{m=1}^Q (\lambda_m - \sigma_e^2) \mathbf{u}_m \mathbf{u}_m^H + \sigma_e^2 \mathbf{I}_{M \times M} \quad (2.66)$$

By comparing (2.63) and (2.66), we know that the eigenvectors corresponding to the array covariance matrix's  $Q$  largest eigenvalues  $\mathbf{u}_m (m = 1, 2, \dots, Q)$  can be expanded as the jamming subspace. So the jamming subspace is

$$\mathbf{U}_j = \text{span}\{\mathbf{u}_1, \mathbf{u}_2, \dots, \mathbf{u}_Q\} \quad (2.67)$$

Then its orthogonal complement space is

$$\mathbf{U}_j^\perp = \mathbf{I} - \mathbf{U}_j \mathbf{U}_j^H \quad (2.68)$$

By projecting the data received by the array into this space, the jamming signal can be mitigated. The data after projection becomes

$$\mathbf{y}(t) = \mathbf{U}_j^\perp \mathbf{x}(t) = \mathbf{U}_j^\perp \sum_{l=1}^L \mathbf{a}(\theta_l) s_l(t) + \mathbf{U}_j^\perp \mathbf{e}(t) \quad (2.69)$$

where  $\mathbf{y}(t) = [y_1(t), y_2(t), \dots, y_M(t)]^T$  represents the output data on the array channels after projection. Since GPS adopts CDMA spread spectrum, it has certain jamming mitigation capabilities (determined by spread spectrum gain). Therefore when we need to adopt the specialized spatial domain adaptive jamming mitigation algorithm to suppress jamming, we usually assume that the jamming power is much higher than the receiver's thermal noise power. When the noise and the signal are not correlated, the received signal's covariance matrix inversion is proportional to the jamming subspace's orthogonal complement space [32]. The detailed derivation procedure is described as below.

Based on (2.64), we have

$$\mathbf{R}_x^{-1} = \sum_{m=1}^M \frac{1}{\lambda_m} \mathbf{u}_m \mathbf{u}_m^H \quad (2.70)$$

Equation (2.70) can be derived from (2.65)

$$\begin{aligned} \mathbf{R}_x^{-1} &= \sum_{m=1}^Q \frac{1}{\lambda_m} \mathbf{u}_m \mathbf{u}_m^H + \frac{1}{\sigma_e^2} \sum_{m=Q+1}^M \mathbf{u}_m \mathbf{u}_m^H \\ &= \frac{1}{\sigma_e^2} \left( \mathbf{I} - \sum_{m=1}^Q \frac{\lambda_m - \sigma_e^2}{\lambda_m} \mathbf{u}_m \mathbf{u}_m^H \right) \end{aligned} \quad (2.71)$$

Then we have

$$\lim_{\sigma_e^2 \rightarrow 0} \sigma_e^2 \mathbf{R}_x^{-1} = \mathbf{U}_J^\perp \quad (2.72)$$

Consequently, we can use a receiver signal's covariance matrix inversion  $\mathbf{R}_x^{-1}$  to replace jamming subspace's orthogonal complement space. Then there is no need to perform eigenvalue decomposition, therefore it can reduce the computation complexity. Equation (2.69) turns into

$$\mathbf{y}(t) = \sum_{l=1}^L \bar{\mathbf{a}}(\theta_l) A_l c_l(t - \tau - T) e^{-j\omega_d(t-T)} + \bar{\mathbf{e}}(t) \quad (2.73)$$

where

$$\bar{\mathbf{a}}(\theta_l) = \mathbf{R}_x^{-1} \mathbf{a}(\theta_l) \quad (2.74)$$

$$\bar{\mathbf{e}}(t) = \mathbf{R}_x^{-1} \mathbf{e}(t) \quad (2.75)$$

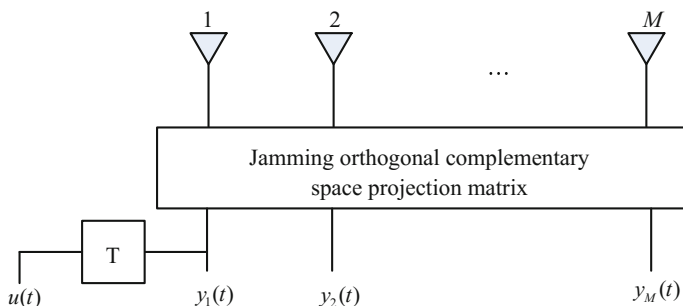
## 2. GNSS Signal Direction Estimation Using Periodic Repetition Characteristic of Spreading Code

It can be known from (2.73) that the GNSS signal is still buried by the noise after projection. If a navigation receiver acquires and tracks the projected data directly, it is equivalent to the power minimization algorithm. Therefore, to improve the carrier-to-noise ratio of a GNSS signal and increase positioning precision, the signal processing gain provided by antenna arrays must be fully leveraged. An effective solution is to produce multiple antenna beams, and make every beam point to a GNSS signal. This forms a null steering towards the jamming directions. The key for such an approach is to first estimate the direction of arrival of every GNSS signal, and then directly use the traditional high resolution DOA (Direction of Arrival) estimation algorithm [33, 34] (e.g. Multiple Signal Classification (MUSIC) and Estimation of Signal Parameters via Rotational Invariance Technique (ESPRIT)). The approach faces two problems: ① The number of satellites could be greater than the number of array elements; ② The GNSS signal is too weak. These two problems result in the difficulty to acquire GNSS signal subspace using the above DOA estimation algorithm. Consequently the characteristic of the GNSS signal must be fully considered to study novel GNSS signal DOA estimation algorithm.

Within the period of a data bit, the C/A code signal repeats itself periodically 20 times, which is also known as the periodic repetition characteristic. Therefore, for the  $l$ th GNSS signal, we have

$$c_l(t - \tau_l - pT) = c_l(t - \tau_l) \quad 1 \leq p \leq 19 \quad (2.76)$$

where  $p$  is the delayed number of periods, obtained by delaying the projected signal from the first antenna channel (usually serves as the reference antenna) with one C/A code period, as shown in Fig. 2.10.



**Fig. 2.10** The diagram of reference signal for single channel single delay cross correlation algorithm

Assuming that the delayed signal and the projected signal are located in the same data bit and can be expressed as  $u(t)$ , then we have

$$u(t) = y_1(t - T) = \sum_{l=1}^L \bar{\mathbf{a}}_1(\theta_l) A_l c_l(t - \tau_l - T) e^{-j\omega_d(t-T)} + \bar{\mathbf{e}}_1(t - T) \quad (2.77)$$

where  $\bar{\mathbf{a}}_1(\theta_l)$  is the first element of  $\bar{\mathbf{a}}(\theta_l)$ ;  $\bar{\mathbf{e}}_1(t)$  is the projected noise data of the first array channel. By making the  $p$  in (2.76) as 1 and substitute it into (2.77), we have

$$u(t) = \sum_{l=1}^L \bar{A}_l c_l(t - \tau_l) e^{-j\omega_d(t-T)} + \bar{\mathbf{e}}_1(t - T) \quad (2.78)$$

where  $\bar{A}_l = \bar{\mathbf{a}}_1(\theta_l) A_l$ . Since different C/A codes are independent from each other, the cross correlation between the projected signal  $\mathbf{y}(t)$  and the delayed signal  $u(t)$  is

$$\mathbf{r} = \sum_{l=1}^L \beta_l \bar{\mathbf{a}}(\theta_l) \quad (2.79)$$

where  $\beta_l = \sigma_{s_l}^2 e^{-j\omega_d T} \bar{\mathbf{a}}_1^*(\theta_l)$  is an unknown complex number. To make

$$\boldsymbol{\beta} = [\beta_1, \beta_2, \dots, \beta_L]^T \quad (2.80)$$

$$\boldsymbol{\theta} = [\theta_1, \theta_2, \dots, \theta_L]^T \quad (2.81)$$

$$\mathbf{A} = [\bar{\mathbf{a}}(\theta_1), \bar{\mathbf{a}}(\theta_2), \dots, \bar{\mathbf{a}}(\theta_L)] \quad (2.82)$$

we could re-write (2.79) in the matrix form below

$$\mathbf{r} = \mathbf{A}\boldsymbol{\beta} \quad (2.83)$$

By using sample cross correlation vector  $\hat{\mathbf{r}}$  to replace the  $\mathbf{r}$  in (2.83), and minimizing the following cost function  $g$  to estimate  $\boldsymbol{\beta}$  and  $\boldsymbol{\theta}$ .

$$g(\{\theta_l, \beta_l\}_{l=1}^L) = \|\hat{\mathbf{r}} - \mathbf{A}\boldsymbol{\beta}\|_2^2 \quad (2.84)$$

We expand (2.84) to have

$$\begin{aligned} g(\{\theta_l, \beta_l\}_{l=1}^L) &= (\hat{\mathbf{r}} - \mathbf{A}\boldsymbol{\beta})^H (\hat{\mathbf{r}} - \mathbf{A}\boldsymbol{\beta}) \\ &= \left[ \boldsymbol{\beta} - (\mathbf{A}^H \mathbf{A})^{-1} \mathbf{A}^H \hat{\mathbf{r}} \right]^H (\mathbf{A}^H \mathbf{A}) \left[ \boldsymbol{\beta} - (\mathbf{A}^H \mathbf{A})^{-1} \mathbf{A}^H \hat{\mathbf{r}} \right] \\ &\quad + \hat{\mathbf{r}}^H \hat{\mathbf{r}} - \hat{\mathbf{r}}^H \mathbf{A} (\mathbf{A}^H \mathbf{A})^{-1} \mathbf{A}^H \hat{\mathbf{r}} \end{aligned} \quad (2.85)$$



It can be known from (2.85), to minimize (2.85) is equivalent to make the first term of (2.85) zero and maximizing its last term, i.e.

$$\hat{\boldsymbol{\theta}} = \arg \max_{\boldsymbol{\theta}} \left[ \hat{\mathbf{r}}^H \mathbf{A} (\mathbf{A}^H \mathbf{A})^{-1} \mathbf{A}^H \hat{\mathbf{r}} \right] \quad (2.86)$$

Correspondingly we have

$$\hat{\boldsymbol{\beta}} = (\mathbf{A}^H \mathbf{A})^{-1} \mathbf{A}^H \hat{\mathbf{r}}|_{\boldsymbol{\theta}=\hat{\boldsymbol{\theta}}} \quad (2.87)$$

The cost function in (2.86) has complex multi-peak shape, so it is hard to solve directly. The usual method to solve the problem is to use the loop optimization algorithm, e.g. CLEAN algorithm [35]. The CLEAN algorithm originated from solving astronomical image processing problems in radio astronomy. Since then, it has been widely applied in the fields of microwave imaging, image processing and spectral estimation [36–38]. Based on the characteristic of CLEAN algorithm (computational simplicity), it is used to solve Nonlinear Least Squares (NLS) problems in (2.86). Before the CLEAN algorithm is given, the following preparatory works are performed:

Given  $\left\{ \hat{\theta}_l, \hat{\beta}_l \right\}_{l=1, l \neq k}^L$  is known or has been estimated, we define

$$\hat{\mathbf{r}}_k = \hat{\mathbf{r}} - \sum_{\substack{l=1 \\ l \neq k}}^L \hat{\beta}_l \hat{\mathbf{a}}(\hat{\theta}_l) \quad (2.88)$$

We also define

$$g_k(\theta_k, \beta_k) = \|\hat{\mathbf{r}}_k - \beta_k \bar{\mathbf{a}}(\theta_k)\|_2^2 \quad (2.89)$$

By minimizing  $g_k(\theta_k, \beta_k)$ , we can obtain

$$\hat{\theta}_k = \arg \max_{\theta_k} \left[ \frac{|\bar{\mathbf{a}}^H(\theta_k) \hat{\mathbf{r}}_k|^2}{\bar{\mathbf{a}}^H(\theta_k) \bar{\mathbf{a}}(\theta_k)} \right] \quad (2.90)$$

$$\hat{\beta}_k = \frac{\hat{\mathbf{a}}^H(\hat{\theta}_k) \hat{\mathbf{r}}_k}{\hat{\mathbf{a}}^H(\hat{\theta}_k) \hat{\mathbf{a}}(\hat{\theta}_k)} \quad (2.91)$$

Below we list the computation steps of the CLEAN algorithm

- (1) Assuming  $L = 1$ , let  $\hat{\mathbf{r}}_1 = \hat{\mathbf{r}}$ , (2.90) and (2.91) can be used to estimate  $\{\hat{\theta}_1, \hat{\beta}_1\}$ .
- (2) Assuming  $L = 2$ ,  $\hat{\mathbf{r}}_2$  can be estimated from (2.88) using  $\{\hat{\theta}_1, \hat{\beta}_1\}$ , then  $\hat{\mathbf{r}}_2$  can be used to estimate  $\{\hat{\theta}_2, \hat{\beta}_2\}$  from (2.90) and (2.91).
- (3) Assuming  $L = 3$ ,  $\{\hat{\theta}_l, \hat{\beta}_l\}_{l=1}^2$  can be used to estimate  $\hat{\mathbf{r}}_3$  from (2.88), then  $\hat{\mathbf{r}}_3$  is used to estimate  $\{\hat{\theta}_3, \hat{\beta}_3\}$  from (2.90) and (2.91).

The above steps are repeated, until  $L$  equals to the pre-determined number of satellites. Then the signal parameter estimates for all the satellites and  $\{\hat{\theta}_l, \hat{\beta}_l\}_{l=1}^L$  can be obtained.

In addition to the CLEAN algorithm, the RELAX algorithm proposed by Li et al. [39–41] is also a type of loop optimization algorithm (also known as a method based on decoupled parameter estimation theory). The RELAX algorithm, compared to the CLEAN algorithm, is more accurate and has higher resolution, but its computation is more complex [42]. If an array antenna receives two GNSS signals with close direction of arrivals, the RELAX algorithm can accurately distinguish the two GNSS signals while the CLEAN algorithm cannot distinguish between the two signals, and it can only treat the two signals as one signal for processing. Nevertheless, this is not a problem for GNSS, because under such a scenario, the pattern's mainlobe obtained using the estimated direction of arrivals of the GNSS signals using the CLEAN algorithm can point to the two GNSS signals simultaneously, thus it still can improve the carrier-to-noise ratio of the two GNSS signals.

### 3. Beamforming

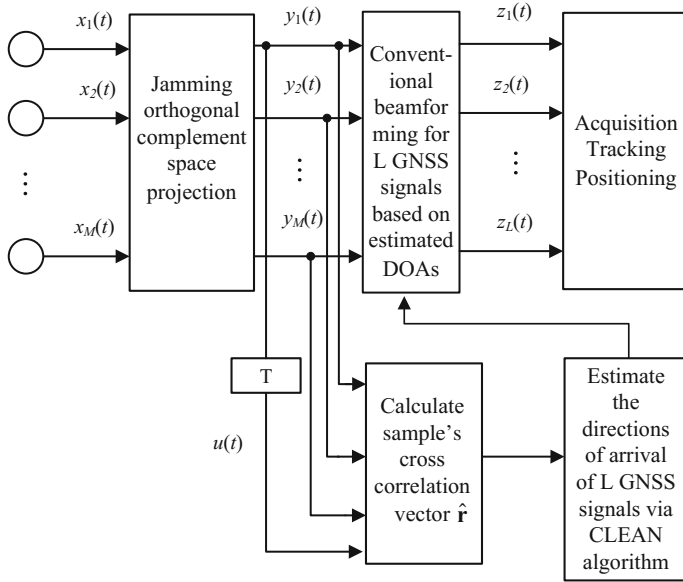
After estimating directions of arrivals for all GNSS signals, these estimated values can be used to perform beamforming. The array weight vector for the  $l$ th GNSS signal is

$$\mathbf{w}_l = \mathbf{R}_x^{-1} \mathbf{a}(\hat{\theta}_l) \quad (2.92)$$

Since the new algorithm uses the array's single channel single delay data to estimate GNSS signal direction of arrival, the algorithm is named single channel single delay cross correlation algorithm. The output signal of the  $l$ th beam is

$$z_l(t) = \mathbf{w}_l^H \mathbf{x}(t) \quad (2.93)$$

The block diagram of the single channel single delay cross correlation algorithm is shown in Fig. 2.11.



**Fig. 2.11** Block diagram of the single channel single delay cross correlation algorithm

#### 4. Cross Correlation Vector Estimation

The single channel single delay cross correlation algorithm needs to calculate the sample cross correlation vector  $\hat{\mathbf{r}}$  between the projected data and the data after periodic delay. The respective definitions of the  $M \times N$  order sample data matrix after projection and the  $1 \times N$  order periodic delay sample data vector are

$$\mathbf{Y}_N = [\mathbf{y}(n), \mathbf{y}(n-1), \dots, \mathbf{y}(n-(N-1))] \quad (2.94)$$

$$\mathbf{u}_N = [u(n), u(n-1), \dots, u(n-(N-1))] \quad (2.95)$$

Then we have

$$\hat{\mathbf{r}} = \frac{1}{N} \mathbf{Y}_N \mathbf{u}_N^H \quad (2.96)$$

It can be known from (2.86) that the cross correlation vector estimate  $\hat{\mathbf{r}}$  determines the performance of the single channel single delay algorithm. Based on the assumptions in this chapter,  $\mathbf{y}(t)$  and  $u(t)$  are located in the same navigation message data bit, and consequently  $\hat{\mathbf{r}}$  is an effective estimation. Nevertheless, since the data sample is obtained through random sampling, there is no guarantee that  $\mathbf{y}(t)$  and  $u(t)$  are located in the same data bit. When both of them are located in

neighboring data bits, then how should we estimate  $\hat{\mathbf{r}}$ ? Towards this question, Ref. [43] proposed a solution by defining the following events:

$H_1$ :  $\mathbf{y}(t)$  and  $u(t)$  are located in the same data bit.

$H_2$ :  $\mathbf{y}(t)$  and  $u(t)$  are located in the neighboring data bits.

$H_{21}$ :  $\mathbf{y}(t)$  and  $u(t)$  are located in the neighboring data bits with the same navigation data bit

$H_{22}$ :  $\mathbf{y}(t)$  and  $u(t)$  are located in the neighboring data bits with the opposite navigation data bit.

The probabilities for the above events to happen are:

$$\text{prob}\{H_1\} = 1 - \frac{p}{20} \quad (2.97)$$

$$\text{prob}\{H_2\} = \frac{p}{20} \quad (2.98)$$

$$\text{prob}\{H_{21}\} = \frac{p}{40} \quad (2.99)$$

$$\text{prob}\{H_{22}\} = \frac{p}{40} \quad (2.100)$$

Therefore we have

$$\begin{aligned} \mathbf{r}_{yu} &= E\{y(t)u^*(t)|H_1\}\text{prob}\{H_1\} + E\{y(t)u^*(t)|H_2\}\text{prob}\{H_2\} \\ &= E\{y(t)u^*(t)|H_1\}\text{prob}\{H_1\} + E\{y(t)u^*(t)|H_{21}\}\text{prob}\{H_{21}\} \\ &\quad + E\{y(t)u^*(t)|H_{22}\}\text{prob}\{H_{22}\} \end{aligned} \quad (2.101)$$

Since

$$E\{y(t)u^*(t)|H_1\} = E\{y(t)u^*(t)|H_{21}\} = \mathbf{r} \quad (2.102)$$

$$E\{y(t)u^*(t)|H_{22}\} = -\mathbf{r} \quad (2.103)$$

where  $\mathbf{r} = \mathbf{A}\boldsymbol{\beta}$  represents the cross correlation between the projected GNSS signal and the periodically delayed signal [see (2.83)]. By substituting (2.97)–(2.100), (2.102) and (2.103) into (2.101), we can have

$$\mathbf{r}_{yu} = \left(1 - \frac{p}{20}\right)\mathbf{r} \quad (2.104)$$

For the single channel single delay cross correlation vector method, we have  $p = 1$ , then

$$\mathbf{r}_{yu} = \frac{19}{20} \mathbf{r} \approx \mathbf{r} \quad (2.105)$$

It can be seen from the above discussion that, when the impact of the navigation data bit is considered, the cross correlation between  $\mathbf{y}(t)$  and  $u(t)$  is close to  $\mathbf{r}$ . Thus (2.96) can be used as the estimate of  $\mathbf{r}$ . In addition, to fully take advantage of the periodic repetition characteristic of the C/A code, the  $G$  projected data block, and periodic repetition data block can be selected, and the cross correlation vector's estimate becomes

$$\hat{\mathbf{r}}_G = \frac{1}{G} \sum_{g=1}^G \frac{\mathbf{Y}_N(g) \mathbf{u}_N^H(g)}{N} \quad (2.106)$$

It can be proved that [31]

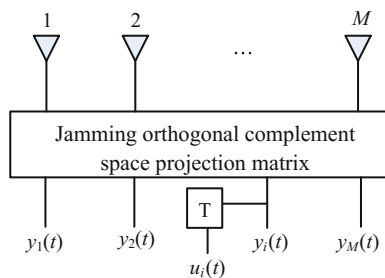
$$E\{\hat{\mathbf{r}}_G\} = \mathbf{r}_{yu} \quad (2.107)$$

i.e.  $\hat{\mathbf{r}}_G$  is  $\mathbf{r}_{yu}$ 's unbiased estimate.

#### 2.4.4 Multiple Channel Single Delay Cross Correlation Algorithm

Since the single channel single delay cross correlation algorithm estimates the GNSS signal's direction of arrival using the cross correlation between the projected data and the reference array element's single delay data after subspace projection, it does not fully take advantage of single delay data of the other array elements. When the array element fails, the algorithm might fail as well. Consequently, let  $u_i(t)$  ( $i = 1, 2, \dots, M$ ) represent the signal obtained by delaying the projected data  $\mathbf{y}(t)$ 's  $i$ th antenna channel by one C/A code period, as shown in Fig. 2.12.

**Fig. 2.12** The diagram of reference signal for multiple channels single delay cross correlation algorithm



Then we have

$$u_i(t) = y_i(t - T) = \sum_{l=1}^L \bar{\mathbf{a}}_i(\theta_l) A_l c_l(t - \tau - T) e^{-j\omega_{dl}(t-T)} + \bar{\mathbf{e}}_i(t - T) \quad (2.108)$$

where  $\bar{\mathbf{a}}_i(\theta_l)$  is  $\bar{\mathbf{a}}(\theta_l)$ 's  $i$ th elements;  $\bar{\mathbf{e}}_i(t)$  represents the noise data of the  $i$ th array channel after projection. By setting the  $p$  in (2.76) as 1, and substituting it into (2.108), we have

$$u_i(t) = \sum_{l=1}^L \bar{\mathbf{a}}_i(\theta_l) A_l c_l(t - \tau) e^{-j\omega_{dl}(t-T)} + \bar{\mathbf{e}}_i(t - T) \quad (2.109)$$

Due to independence among different C/A codes, the cross correlation between the projected signal  $\mathbf{y}(t)$  and the delayed signal  $u_i(t)$  is

$$\mathbf{r}_i = \sum_{l=1}^L \tilde{\beta}_l \bar{\mathbf{a}}_i^*(\theta_l) \bar{\mathbf{a}}(\theta_l) \quad (2.110)$$

where  $\tilde{\beta}_l = \sigma_{s_l}^2 e^{-j\omega_{dl}T}$ . By constructing a new vector  $\tilde{\mathbf{r}}$

$$\tilde{\mathbf{r}} = [\mathbf{r}_1^T, \mathbf{r}_2^T, \dots, \mathbf{r}_M^T]^T \quad (2.111)$$

By substituting (2.110) into (2.111) and after simplification, we can have

$$\tilde{\mathbf{r}} = \sum_{l=1}^L \tilde{\beta}_l \bar{\mathbf{a}}^*(\theta_l) \otimes \bar{\mathbf{a}}(\theta_l) \quad (2.112)$$

where  $\otimes$  denotes Kronecker product. Let

$$\tilde{\mathbf{a}}(\theta_l) = \bar{\mathbf{a}}^*(\theta_l) \otimes \bar{\mathbf{a}}(\theta_l) \quad (2.113)$$

Equation (2.112) becomes

$$\tilde{\mathbf{r}} = \sum_{l=1}^L \tilde{\beta}_l \tilde{\mathbf{a}}(\theta_l) \quad (2.114)$$

Let  $\tilde{\mathbf{A}} = [\tilde{\mathbf{a}}(\theta_1) \quad \tilde{\mathbf{a}}(\theta_2) \quad \dots \quad \tilde{\mathbf{a}}(\theta_L)]$ , (2.113) can be written in matrix form:

$$\tilde{\mathbf{r}} = \tilde{\mathbf{A}} \tilde{\boldsymbol{\beta}} \quad (2.115)$$

By substituting the  $\tilde{\mathbf{r}}$  in (2.115) with the sample cross correlation vector  $\hat{\tilde{\mathbf{r}}}$ , and minimizing the following cost function  $f$  to estimate  $\hat{\tilde{\boldsymbol{\beta}}}$  and  $\hat{\boldsymbol{\theta}}$ .

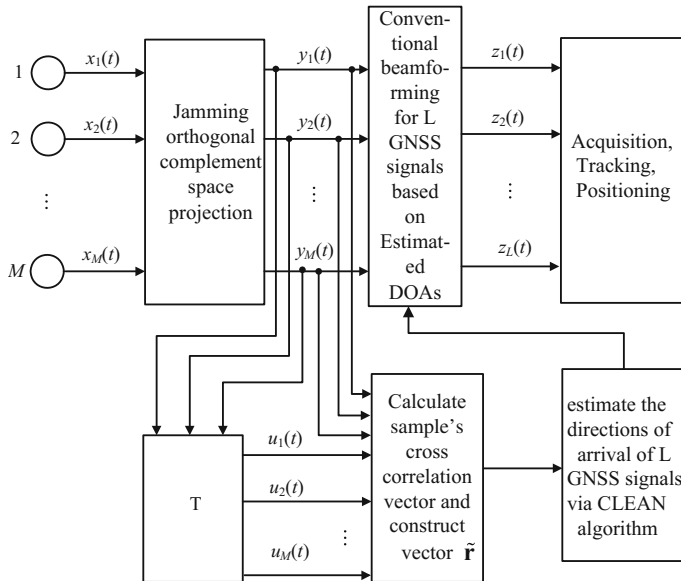
$$f\left(\left\{\theta_l, \tilde{\beta}_l\right\}_{l=1}^L\right)=\left\|\hat{\tilde{\mathbf{r}}}-\tilde{\mathbf{A}}\tilde{\boldsymbol{\beta}}\right\|_2^2 \quad (2.116)$$

Equations (2.116) and (2.84) have the same structure, so they can be solved using the CLEAN algorithm described in the Sect. 2.4.3 in this chapter. Here we list an estimation formula when there is one single GNSS signal, i.e.

$$\hat{\theta}_1=\arg \max _{\theta}\left[\frac{\left|\tilde{\mathbf{a}}^H(\theta) \hat{\tilde{\mathbf{r}}}\right|^2}{\tilde{\mathbf{a}}^H(\theta) \tilde{\mathbf{a}}(\theta)}\right] \quad (2.117)$$

$$\hat{\tilde{\beta}}_1=\frac{\tilde{\mathbf{a}}^H\left(\hat{\theta}_1\right) \hat{\tilde{\mathbf{r}}}}{\tilde{\mathbf{a}}^H\left(\hat{\theta}_1\right) \tilde{\mathbf{a}}\left(\hat{\theta}_1\right)} \quad (2.118)$$

For parameter estimations for other GNSS signals, we can use similar approaches like the CLEAN algorithm, to convert it to a series of single GNSS signal parameter estimation problems. The block diagram of multiple channel single delay cross correlation algorithm is shown in Fig. 2.13.



**Fig. 2.13** Block diagram of multiple channels single delay cross correlation algorithm

In this section, we discuss two types of adaptive spatial domain filtering algorithm: single channel single delay algorithm and multiple channel single delay algorithm based on GNSS signal's periodic repetition characteristic. It can also be expanded to single channel multiple delays cross correlation algorithm, but two-dimensional search is needed and the computational load is heavy. Compared with the single channel single delay algorithm, multiple channel single delay algorithm takes full advantage of the data from various channels in the array, so it performs better and is more robust. For example, it is not sensitive to a fault in the channel data.

### 2.4.5 Simulation Results

In this section, we use simulations to verify the effectiveness of the algorithms. For simulations, the antenna array has a 10-element uniform linear array with an interval of a half wavelength. 4 GPS GNSS signals, PRN1, PRN2, PRN3, and PRN20 incident on the array from the directions of  $0^\circ$ ,  $-55^\circ$ ,  $-20^\circ$  and  $30^\circ$ , and one jamming signal incidents on the array from the direction of  $50^\circ$ . The jamming-to-noise ratio is 40 dB, and the signal-to-noise ratio is  $-20$  dB. The digital IF frequency for the received signal by the array is 4.309 MHz, and the sampling rate is 5.714 MHz.

#### 1. Adaptive Array Pattern Comparisons

Figure 2.14a–d are the adaptive array patterns for power minimization algorithm, SCORE algorithm, single channel single delay cross correlation algorithm, and multiple channel single delay cross correlation algorithm. The vertical lines in various subfigures represent the incoming GNSS signal directions, and the vertical dashed lines represent the incoming direction of the jamming signal. By comparing Fig. 2.14a–d, we can see that all these methods can suppress jamming. But the array pattern generated by the SCORE algorithm only forms main lobes on the directions of the PRN1 and PRN3 GNSS signals, and for the PRN2 and PRN20 GNSS signals it even has a suppressive effect. The power minimization method cannot provide gains on any satellite. The two new algorithms discussed in this chapter can generate multiple beams, and can ensure that every beam's mainlobe always aims towards the direction of one GNSS signal, so the array pattern with high gain can be obtained.

#### 2. Comparisons of Acquisition Results

Figure 2.15 shows the acquisition results after applying the SCORE algorithm for jamming mitigation. In Fig. 2.15, the x-coordinate represents the GPS satellite's PRN number, and the y-coordinate represents the normalized acquisition factor, which is defined as the ratio between the maximum and the second maximum normalized correlation peaks in the receiver acquisition module. Figure 2.15 shows



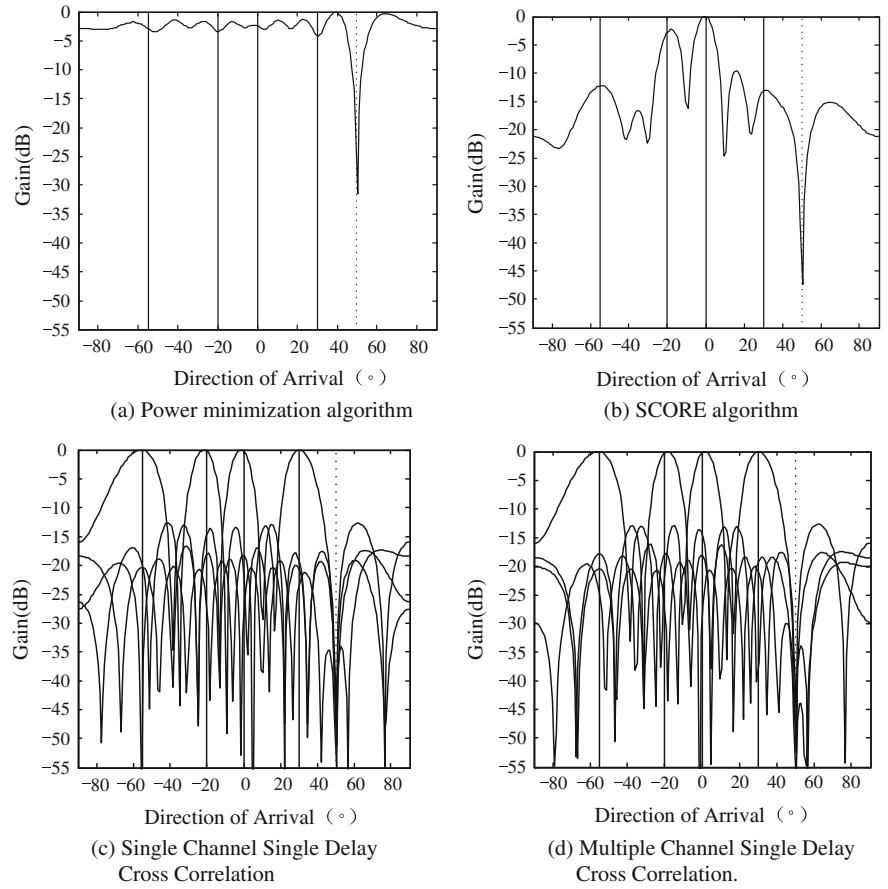
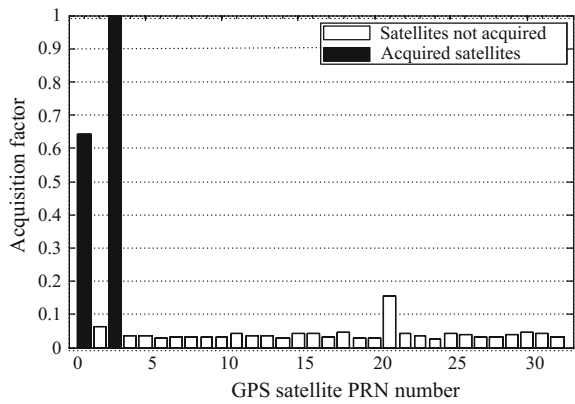


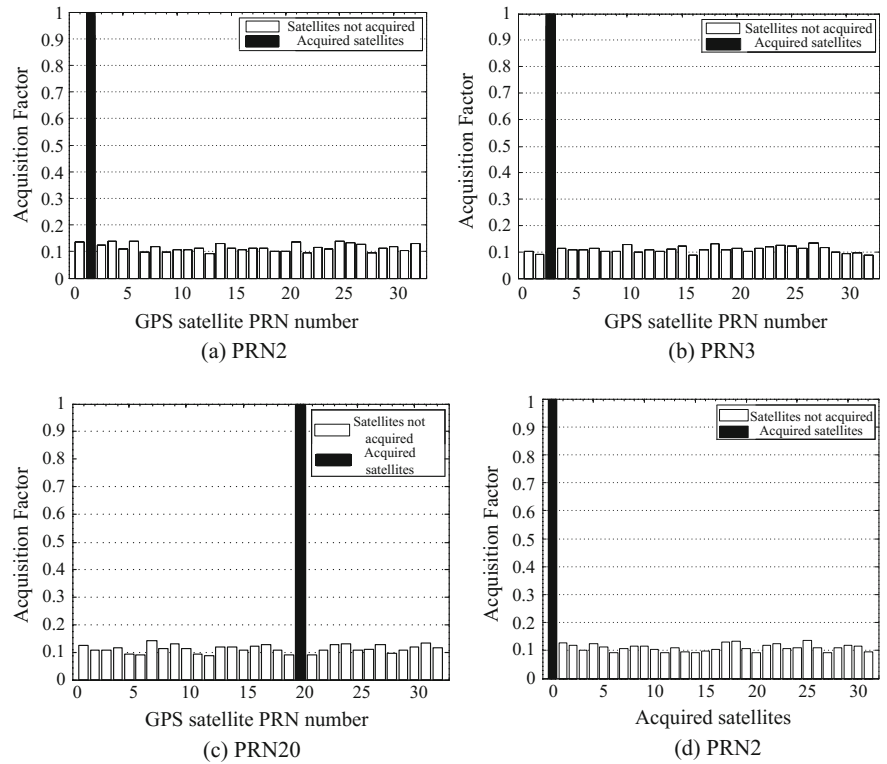
Fig. 2.14 Comparisons on adaptive array patterns obtained by different methods

Fig. 2.15 Acquisition results of SCORE algorithm



that the algorithm only acquires two GNSS signals (PRN1 and PRN3). This is because the pattern formed based on this method cannot accurately aim towards every GNSS signal, and it can even degrade some GNSS signals (PRN2 and PRN20). As a result the receiver cannot acquire all GNSS signals.

Figure 2.16 shows the acquisition results using the single channel single delay cross correlation algorithm for jamming mitigation. The multiple channel single delay cross correlation algorithm’s simulation results are similar to the single channel single delay cross correlation algorithm, and as a result we cannot draw any redundant graphs for that algorithm. Figure 2.16 and the plots for all the other acquisition have the same coordinate meanings as the Fig. 2.15. Therefore the descriptions are not repeated. It can be seen in Fig. 2.16 that every beam generated by the new algorithm points to a GNSS signal and as a result the receiver can acquire GNSS signals. Since the new algorithm does not use the same weight vector to generate multiple beams, the receiver has to acquire the output of every beam separately, thus we have four plots of acquisition results. It can be seen that the new algorithm is not directly compatible with regular receivers. A receiver adopting the new algorithm needs to perform acquisition, tracking and solve parameters such as



**Fig. 2.16** Single channel single delay cross correlation algorithm’s acquisition results

pseudo-range on every individual beam, then combine these information to perform positioning calculation. Its procedures to perform acquisition, tracking, pseudo-range calculation, and positioning calculation are the same as regular receivers, and can be implemented using software receivers.

Figure 2.17 compares the acquisition results on the same GNSS signals using the single channel single delay cross correlation algorithm and the power minimization algorithm. Among the subfigures, Fig. 2.17a corresponds to the power minimization algorithm; Fig. 2.17b corresponds to the single channel single delay cross correlation algorithm; Fig. 2.17c corresponds to the multiple channels single delay cross correlation algorithm. It can be seen in Fig. 2.17, after jamming mitigation, even though the power minimization algorithm can acquire the GNSS signal, the sidelobe level of the normalized cross correlation coefficient is rather large. These two algorithms take advantage of the GNSS signal’s periodic repetition

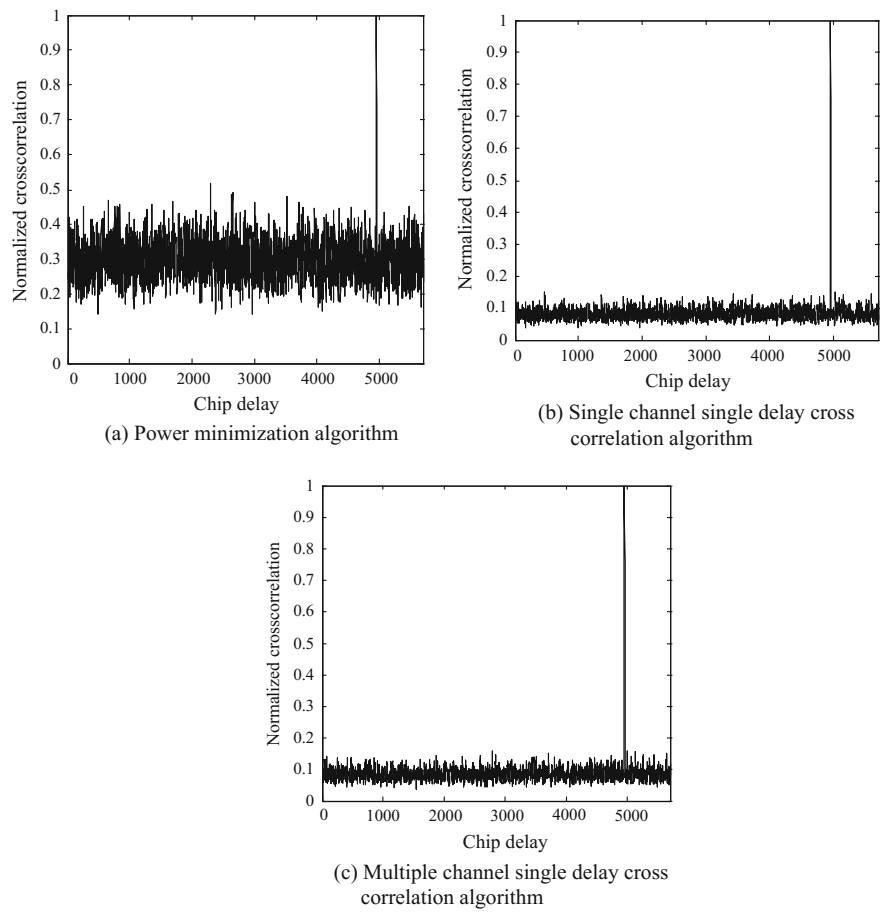
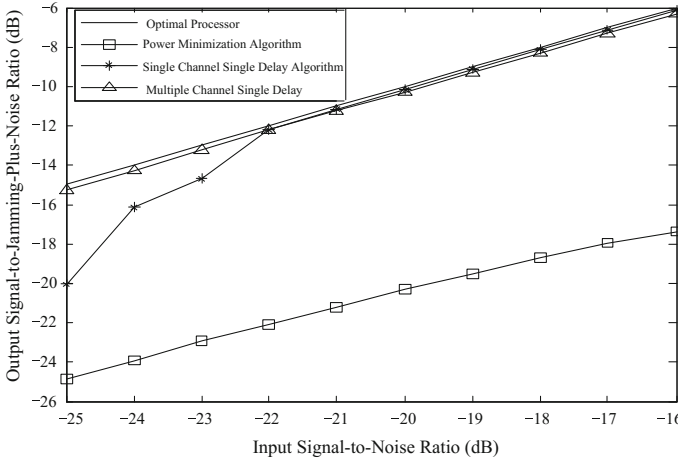


Fig. 2.17 Comparisons of acquisition results



**Fig. 2.18** Comparisons of output signal-jamming-plus-noise curves for different algorithms

characteristic to aim the antenna mainlobe towards the direction of the GNSS signal, and consequently its acquisition result is much better than that of the power minimization algorithm.

### 3. Comparisons of output signal-jamming-plus-noise ratio

Figure 2.18 compares the output signal-jamming-plus-noise ratio curves of the single channel single delay cross correlation algorithm and the power minimization algorithm. Among them, the solid line is the curve for the optimal processor, the “\*” line is the curve for the single channel single delay cross correlation algorithm, the “□” line is the curve for the power minimization algorithm, and the “Δ” line is the curve for the multiple channel single delay cross correlation algorithm. It can be seen in Figure 2.18 that since the power minimization algorithm has no beam pointing direction, it cannot improve output signal-jamming-plus-noise ratio, i.e. it cannot improve a GNSS signal’s carrier-to-noise ratio. But the single channel single delay cross correlation algorithm can increase a GNSS signal’s carrier-to-noise ratio. And as the input signal-to-noise ratio increases, the output signal-jamming-plus-noise ratio can approach the theoretical optimum value.

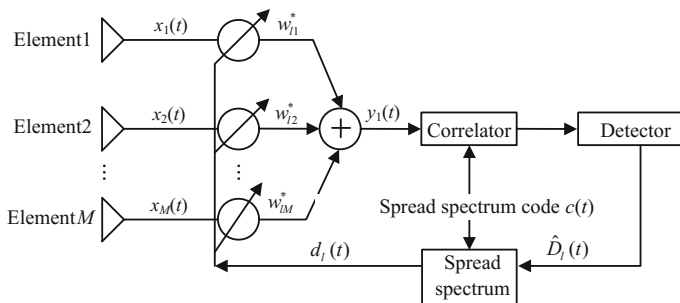
## 2.5 Spatial Domain Adaptive Filtering Using Known Spreading Code Information

The adaptive jamming mitigation algorithm using a GNSS signal’s periodic repetition characteristic can generate multiple beams aiming towards the GNSS signals. But array manifold information is needed to estimate the directions of the GNSS signals. If the array has errors such as amplitude error, phase error, or array position

error, the algorithm performance degrades. In essence, the adaptive jamming mitigation algorithm based on the GNSS signal's periodic repetition characteristic is equivalent to the MMSE criterion described in Sect. 2.3.3. In this section, from the perspective of MMSE criterion, we discuss the algorithm for GNSS signal reconstruction using the de-spread re-spread technique when given known spread spectrum code, and performing beamforming using the reconstructed signal as the reference signal. The algorithm does not need to know the array manifold and the directions of the GNSS signals, so it is very robust. This type of algorithm originated from the blind adaptive jamming mitigation algorithm for the CDMA system, and the most typical variant of the algorithm is the least-squares de-spread re-spread multi-target array [43–46].

### 2.5.1 Least-Squares De-spread Re-spread Multi-target Array

The least-squares de-spread re-spread multi-target array using spread spectrum information [47] is a blind jamming mitigation algorithm aiming towards CDMA systems. This method uses the spread spectrum codes for different users in the CDMA system to obtain the weight vector of a multi-target beamformer, thereby forming multiple beams. To be more specific, the correlations between different users' received signal and spread spectrum code are calculated, to estimate every user's bit information. Then, the spread spectrum code is used again to perform re-spread on the estimated bit information, and the signal after the re-spread is used as the reference signal for the adaptive beamformer to refresh the weight vector. Since GPS signals use CDMA modulation as well, the de-spread and re-spread techniques can be used for GPS jamming mitigation. But compared with other types of CDMA signals, GPS signals are very weak, so the correlator might fail when jamming exists. Consequently, this characteristic has to be fully considered when the de-spread re-spread algorithm is applied. The block diagram for least-squares de-spread re-spread algorithm is shown in Fig. 2.19.



**Fig. 2.19** Block diagram for least-squares de-spread re-spread algorithm

In Fig. 2.19,  $y_l(t)$  denotes the array output of the  $l$ th GNSS signal;  $\mathbf{w}_l = [w_{l1}, w_{l2}, \dots, w_{lM}]^T$  is the array weight vector for the  $l$ th GNSS signal;  $d_l(t)$  is the re-spread signal of the  $l$ th GNSS signal. The Least Squares De-spread Re-spread (LS-DR) determines the optimum weight vector  $\mathbf{w}_l$  by minimizing the cost function in (2.119)

$$F(\mathbf{w}_l) = \sum_{n=1}^N |y_l(n) - d_l(n)|^2 = \sum_{n=1}^N |\mathbf{w}_l^H \mathbf{x}(n) - d_l(n)|^2 \quad (2.119)$$

where  $N$  is the number of samples, and the number of samples within a C/A code period is usually selected. The problem of minimizing (2.119) can be solved using the generalized Gauss method [44]. And the cost function can be expressed as

$$F(\mathbf{w}_l) = \sum_{n=1}^N g_n^2(\mathbf{w}_l) \quad (2.120)$$

where  $g_n(\mathbf{w}_l) = |\mathbf{w}_l^H \mathbf{x}(n) - d_l(n)|$ , and its gradient is

$$\nabla(g_n(\mathbf{w}_l)) = \mathbf{x}(n) \frac{v_l^*(n)}{|v_l(n)|} \quad (2.121)$$

where  $v_l(n) = \mathbf{w}_l^H \mathbf{x}(n) - d_l(n)$ . Based on the Gauss method, the weight vector's updating formula is

$$\mathbf{w}_l(k+1) = \mathbf{w}_l(k) - [\mathbf{G}(\mathbf{w}_l(k))\mathbf{G}^H(\mathbf{w}_l(k))]^{-1} \mathbf{G}(\mathbf{w}_l(k))\mathbf{g}(\mathbf{w}_l(k)) \quad (2.122)$$

where

$$\begin{aligned} \mathbf{g}(\mathbf{w}_l) &= [g_1(\mathbf{w}_l), g_1(\mathbf{w}_l), \dots, g_N(\mathbf{w}_l)]^T \\ &= [|v_1(n)|, |v_1(n)|, \dots, |v_N(n)|]^T \end{aligned} \quad (2.123)$$

$$\mathbf{G}(\mathbf{w}_l) = [\nabla(g_1(\mathbf{w}_l)), \nabla(g_2(\mathbf{w}_l)), \dots, \nabla(g_N(\mathbf{w}_l))] \quad (2.124)$$

By substituting (2.121) into (2.124), we can obtain

$$\mathbf{G}(\mathbf{w}_l) = \left[ \mathbf{x}(1) \frac{v_l^*(1)}{|v_l(1)|}, \mathbf{x}(2) \frac{v_l^*(2)}{|v_l(2)|}, \dots, \mathbf{x}(N) \frac{v_l^*(N)}{|v_l(N)|} \right] = \mathbf{X}\mathbf{V}_l \quad (2.125)$$

where

$$\mathbf{X} = [\mathbf{x}(1), \mathbf{x}(2), \dots, \mathbf{x}(N)] \quad (2.126)$$

$$\mathbf{V}_l = \text{diag} \left( \left[ \frac{v_l^*(1)}{|v_l(1)|}, \frac{v_l^*(2)}{|v_l(2)|}, \dots, \frac{v_l^*(N)}{|v_l(N)|} \right] \right) \quad (2.127)$$

Based on (2.125)–(2.127), we can obtain

$$\mathbf{G}(\mathbf{w}_l)\mathbf{G}^H(\mathbf{w}_l) = \mathbf{X}\mathbf{V}_l\mathbf{V}_l^H\mathbf{X}^H = \mathbf{X}\mathbf{X}^H \quad (2.128)$$

From (2.123) and (2.125), the following can be obtained

$$\mathbf{G}(\mathbf{w}_l)\mathbf{g}(\mathbf{w}_l) = \mathbf{X}\mathbf{V}_l \begin{bmatrix} |v_l(1)| \\ \vdots \\ |v_l(N)| \end{bmatrix} = \mathbf{X} \begin{bmatrix} v_l^*(1) \\ \vdots \\ v_l^*(N) \end{bmatrix} = \mathbf{X}(\mathbf{X}^H\mathbf{w}_l - \mathbf{d}_l) \quad (2.129)$$

where

$$\mathbf{d}_l = [d_l(1), \quad d_l(2), \quad \dots, \quad d_l(N)]^H \quad (2.130)$$

By substituting (2.128) and (2.129) into (2.122), then

$$\begin{aligned} \mathbf{w}_l(k+1) &= \mathbf{w}_l(k) - [\mathbf{X}(k)\mathbf{X}^H(k)]^{-1}\mathbf{X}(k)[\mathbf{X}^H(k)\mathbf{w}_l(k) - \mathbf{d}_l(k)] \\ &= [\mathbf{X}(k)\mathbf{X}^H(k)]^{-1}\mathbf{X}(k)\mathbf{d}_l(k) \end{aligned} \quad (2.131)$$

where

$$\mathbf{X}(k) = [\mathbf{x}(1+kN), \quad \mathbf{x}(2+kN), \quad \dots, \quad \mathbf{x}(N+kN)]^T \quad (2.132)$$

$$\mathbf{d}_l(k) = \hat{D}_l(k)[c_l(1+kN - \hat{\tau}_l), \quad c_l(2+kN - \hat{\tau}_l), \quad \dots, \quad c_l(N+kN - \hat{\tau}_l)]^T \quad (2.133)$$

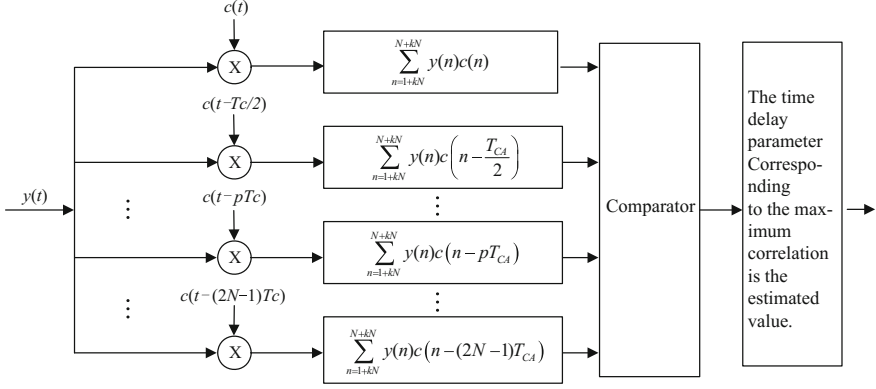
where  $\hat{D}_l(k)$  is the navigation message's estimate of the  $l$ th GNSS signal, i.e.

$$\hat{D}_l(k) = \text{sgn} \left\{ \text{Re} \left[ \sum_{k=1+kN}^{N+kN} y_l(k)c_l(k - \hat{\tau}) \right] \right\} \quad (2.134)$$

where  $\text{sgn}$  is the signal function, then we have

$$\mathbf{y}_l(k) = [y_l(1+kN), \quad y_l(1+kN), \quad \dots, \quad y_l(N+kN)]^T = [\mathbf{w}_l^H(k)\mathbf{X}(k)]^T \quad (2.135)$$

Based on the derivation steps of the above least squares de-spread re-spread algorithm, it can be seen that, to obtain  $\hat{D}_l(k)$ , we need to estimate the  $l$ th GNSS signal's  $\hat{\tau}$ . For a spread spectrum system, the estimation of this parameter can be regarded as a synchronization problem, and usually a correlator structure [35] is used to achieve synchronization and obtain the estimate on the  $\tau$ , denoted by  $\hat{\tau}$ . During the synchronization process, the locally generated spread spectrum code



**Fig. 2.20** Block diagram of estimating time delay parameters using correlators

$c_l(t)$  gradually delays by a half chip interval. Correlations are calculated between every delayed spread spectrum code sequence and the array output data  $y_l(t)$ . Then the outputs from every correlator are selected, and the maximum output is selected. At this point we can consider that the spread spectrum code on the branch corresponds to the maximum output and the transmitted signal achieves the coarse synchronization. The time delay of the corresponding spread spectrum chip is  $\hat{\tau}$ . The detailed block diagram of the correlator is shown in Fig. 2.20.

The least squares de-spread re-spread algorithm for the  $l$ th GNSS signal can be summarized as below:

- (1) initialize weight vector  $\mathbf{w}_l(0)$ .
- (2) calculate array output vector  $\mathbf{y}_l(k)$  using (2.135).
- (3) use the correlator shown in the Fig. 2.20 to estimate  $\hat{\tau}$ , and then use (2.134) to obtain the estimated navigation message  $\hat{D}_l(k)$ .
- (4) use (2.133) to re-spread the navigation message, and obtain the reference signal  $d_l(k)$ .
- (5) use (2.131) to update the weight vector  $\mathbf{w}_l(k+1)$ .
- (6) repeat step (2)–step (5), until  $F(\mathbf{w}_l(k+1))$  is smaller than the preset threshold, and the algorithm converges.

### 2.5.2 New De-spread Re-spread Jamming Mitigation Algorithm

The least squares de-spread re-spread algorithm described in Sect. 2.5.1 needs to have a suitable initial weight vector. The reason is that for a GNSS, when jamming exists, the correlators shown in Fig. 2.20 cannot estimate  $\hat{\tau}$ , so the GNSS signal cannot be reconstructed. In addition, when this algorithm re-spreads the signal, it



only considers the time delay of the C/A code, but not the Doppler frequency. This greatly reduces the performance of the algorithm. Consequently we need to improve this algorithm.

It is well known that if no jamming exists, the acquisition and tracking modules of the GNSS can accurately obtain GNSS signal parameters such as chip delay, Doppler frequency, and navigation messages, i.e. it can de-spread the GNSS signal. To fully take advantage of the acquisition and tracking modules of the GNSS receiver, the jamming suppression has to be performed first. In this section we use array covariance matrix's inverse  $\hat{\mathbf{R}}_x^{-1}$  to replace jamming orthogonal complement space (see Sect. 2.4.3 for details). By rewriting the signal after projection for (2.73), the signal becomes

$$\mathbf{y}(t) = \sum_{l=1}^L \bar{\mathbf{a}}(\theta_l) s_l(t) + \bar{\mathbf{e}}(t) \quad (2.136)$$

After removing the jamming, the receiver can acquire and track the satellite signal. Assuming that the time delay, Doppler frequency and navigation message for the  $l$ th satellite signal are  $\hat{\tau}_l$ ,  $\hat{\omega}_{dl}$  and  $\hat{D}_l(t - \hat{\tau}_l)$  respectively, we can then obtain the reference signal after the re-spread as

$$d_l(t) = \hat{D}_l(t - \hat{\tau}_l) c_l(t - \hat{\tau}_l) e^{j\hat{\omega}_{dl}t} \quad (2.137)$$

Then, the cross-correlation between that signal and the projected array output is

$$\mathbf{r}_{yd} = E\{\mathbf{y}(t)d_l^*(t)\} \quad (2.138)$$

Since the C/A codes for GNSS signals are orthogonal to each other, and the GNSS signal and the noise are independent to each other, then we can substitute (2.136) into (2.138) to obtain

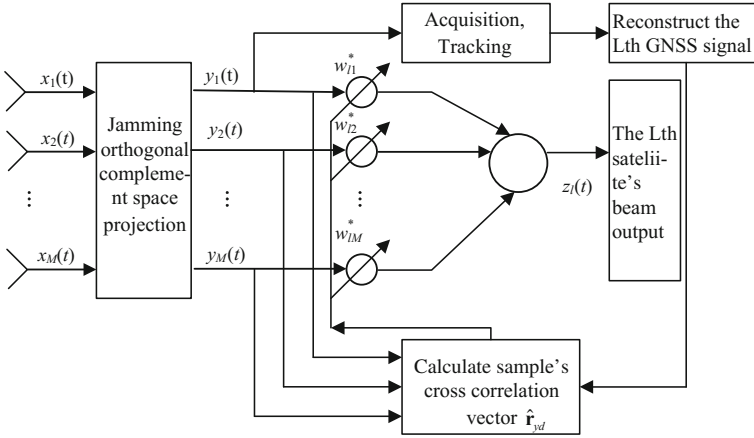
$$\mathbf{r}_{yd} = A_l \bar{\mathbf{a}}(\theta_l) \quad (2.139)$$

Obviously,  $\mathbf{r}_{yd}$  is proportional to the projection direction vector  $\bar{\mathbf{a}}(\theta_l)$  of the  $l$ th GNSS signal. Therefore  $\mathbf{r}_{yd}$  can be used to enhance the  $l$ th GPS signal, then the array weight vector for signal after the projection is

$$\mathbf{w}_l = \mathbf{r}_{yd} \quad (2.140)$$

For actual weight vector calculation, the sample cross correlation vector  $\hat{\mathbf{r}}_{yd}$  is usually used to substitute the theoretical cross correlation vector  $\mathbf{r}_{yd}$ , i.e.

$$\hat{\mathbf{r}}_{yd} = \sum_{n=1}^N \mathbf{y}(n) d_l^*(n) \quad (2.141)$$



**Fig. 2.21** Block diagram of the new de-spread re-spread algorithm

Since GNSS signals are weak, to fully take advantage of the GNSS signal's spread spectrum gain, the number of samples  $N$  is at least more than the number of samples within one C/A code period. The new de-spread and re-spread algorithm's block diagram is shown in Fig. 2.21.

As shown in Fig. 2.21, the new de-spread re-spread algorithm is a two-stage jamming mitigation processor. The stage one processor projects the signal received by the antenna array towards the jamming orthogonal complement space in order to eliminate the jamming signal; and the stage two processor acquires and tracks the output signal of the first antenna after projection and reconstructs the  $l$ th GPS signal. Then the sample cross correlation  $\hat{\mathbf{r}}_{yd}$  between that signal and the signal after the projection is calculated, and the vector is used as the weight vector to enhance the  $l$ th GPS signal. Thereby, the array's overall weight vector is

$$\mathbf{w}_{lopt} = \hat{\mathbf{R}}_x^{-1} \hat{\mathbf{r}}_{yd} \quad (2.142)$$

Based on Fig. 2.21 and the derivation steps of the new algorithm, we know that compared with the least squares de-spread and re-spread algorithm, the new algorithm can accurately obtain GNSS signal information such as time delay, Doppler frequency and navigation messages. There is no need for iterative calculation, so the implementation is simple.

In Sects. 2.3–2.5, we discussed jamming mitigation algorithms. To facilitate readers to read and compare, Table 2.1 compares and summarizes the applicabilities of various algorithms mentioned before.

**Table 2.1** Comparisons of applicabilities for various jamming mitigation algorithms

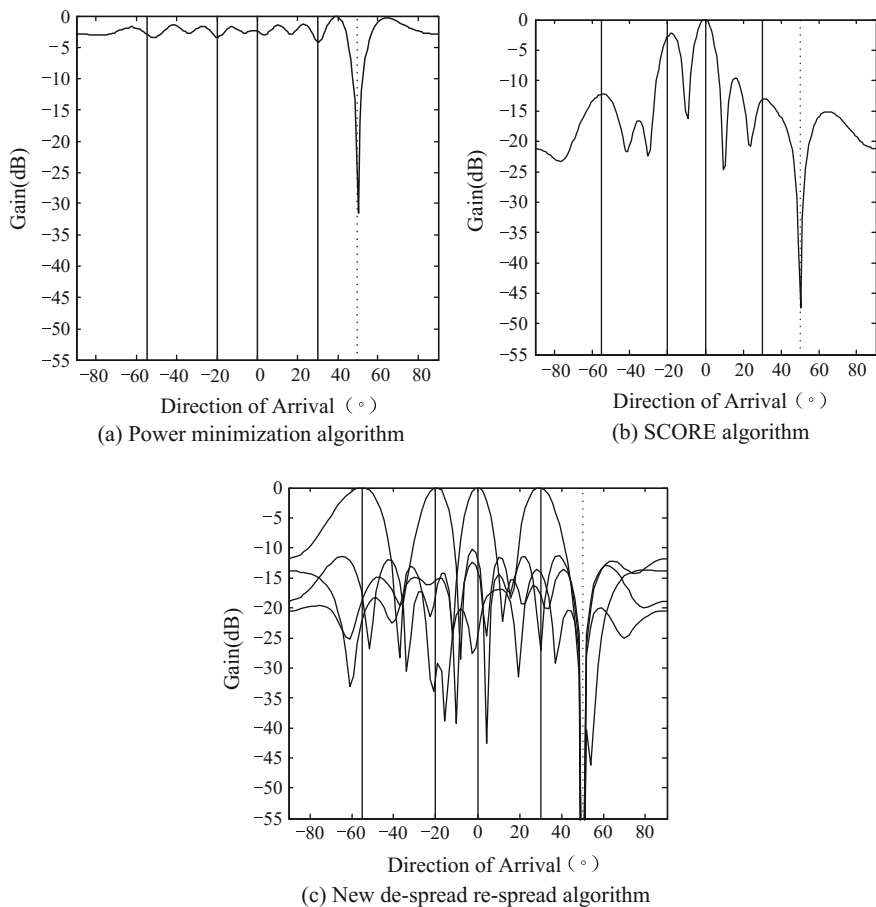
Applicability	Algorithm			
	Power minimization algorithm	Capon algorithm	Adaptive filtering algorithm using periodic repetition characteristic	Adaptive filtering algorithm using known spread spectrum code information
Provide signal processing gain	No	Yes	Yes	Yes
Need to know GNSS signal's direction of arrival	Not required	Required	Not required	Not required
Need to know array manifold	Not required	Required	Required	Not required
Array structure	Random array	Random array	Random array	Random array

2.5.3 Simulation Results

For simulation, the antenna array has a 10-element uniform linear array with an interval of half a wavelength. 4 GPS GNSS signals, PRN1, PRN2, PRN3 and PRN20 incident on the array from the directions of  $0^\circ$ ,  $-55^\circ$ ,  $-20^\circ$  and  $30^\circ$ , and one jamming signal incidents on the array from the direction of  $50^\circ$ . The jamming-to-noise ratio is 40 dB, and the signal-to-noise ratio is  $-20$  dB. The digital IF frequency for the received signal by the array is 4.309 MHz, and the sampling rate is 5.714 MHz.

1. Comparisons of Adaptive Array Patterns

Figure 2.22 lists the adaptive array patterns obtained using three methods. Among them, Fig. 2.22a is the adaptive array pattern for the power minimization algorithm. Fig. 2.22b is for the SCORE algorithm, and Fig. 2.22c is for the new de-spread re-spread algorithm. The vertical solid lines in the subfigures represent the directions of the GNSS signals; the vertical dotted lines represent the directions of the jamming signals. By comparing Fig. 2.22a–c, we know that all three methods can suppress jamming. The SCORE algorithm generates the mainlobes towards the directions of the two GNSS signals (PRN1 and PRN3), and at the same time it attenuates the other two GNSS signals (PRN2 and PRN20). The power minimization method cannot provide gain on any satellite. But the new de-spread re-spread algorithm can generate multiple beams, and every beam's mainlobe aims



**Fig. 2.22** Comparisons of adaptive array patterns for different algorithms

towards the GNSS signal, therefore the GNSS receivers can track signals from all satellites.

## 2. Comparisons of Acquisition Results

Figure 2.23 shows the acquisition results of using the SCORE algorithm for jamming mitigation. It can be seen in the figure that, since the SCORE algorithm attenuates the two GNSS signals (PRN2 and PRN20), the receiver only acquires two GNSS signals (PRN1 and PRN3).

Figure 2.24 shows the acquisition results using the new de-spread re-spread algorithm. It can be seen in Fig. 2.24 that every beam generated by the new algorithm aims towards a GNSS signal, thereby the receiver can acquire every GNSS signal.

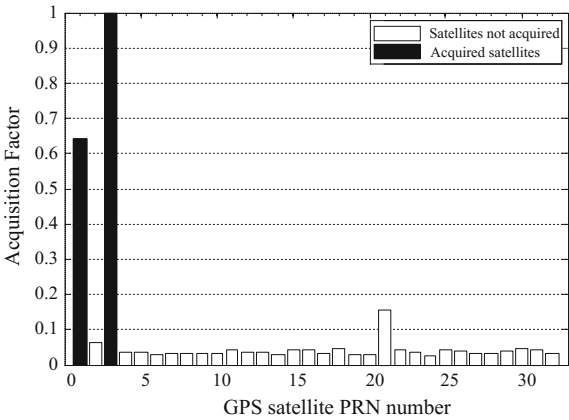


Fig. 2.23 Acquisition results of SCORE algorithm

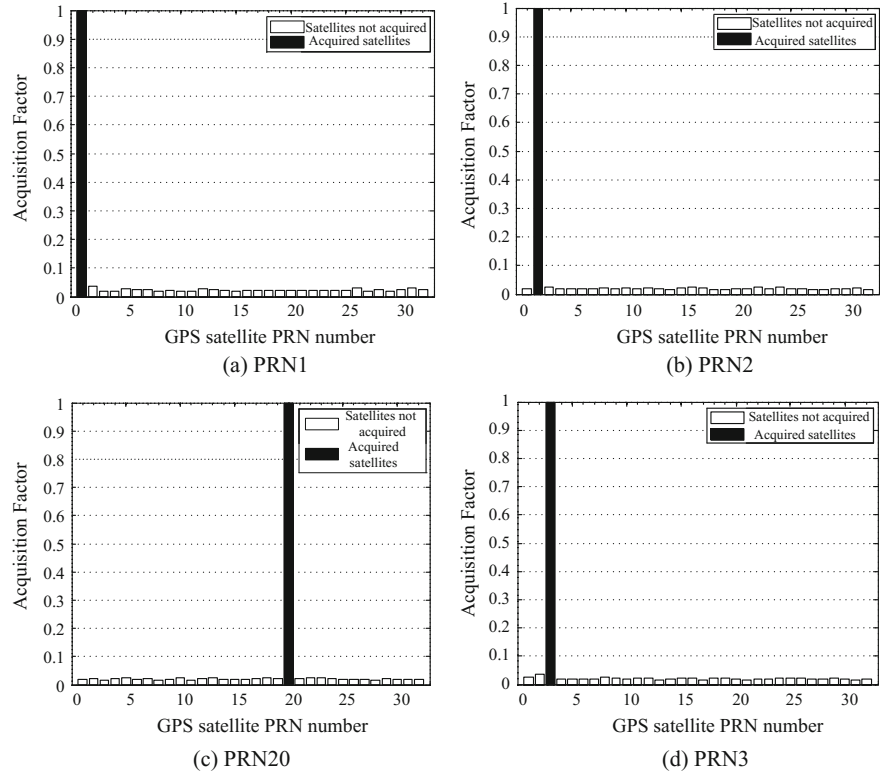
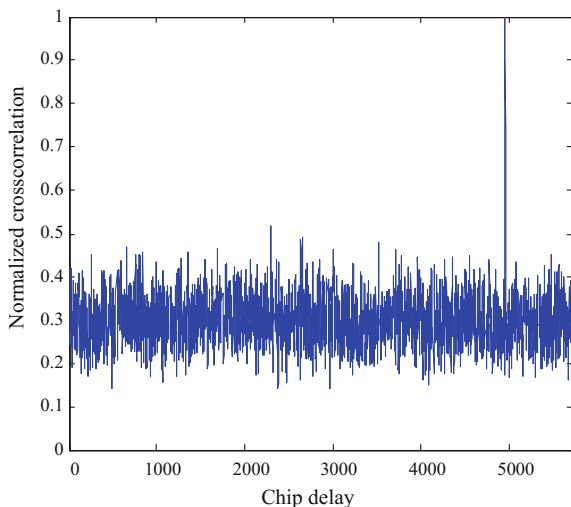
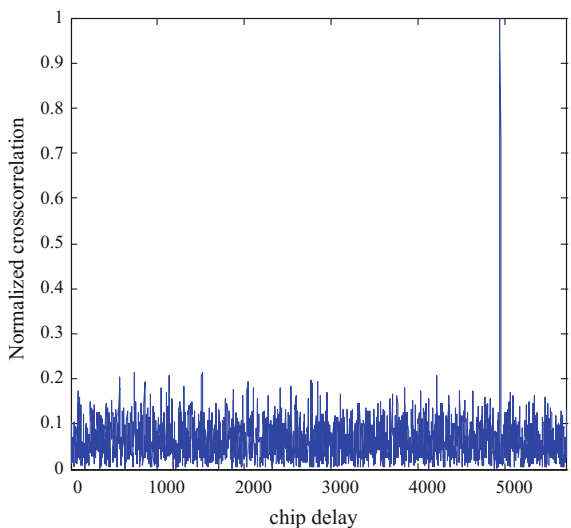


Fig. 2.24 Acquisition results of the new de-spread re-spread algorithm

**Fig. 2.25** Comparisons of acquisition results



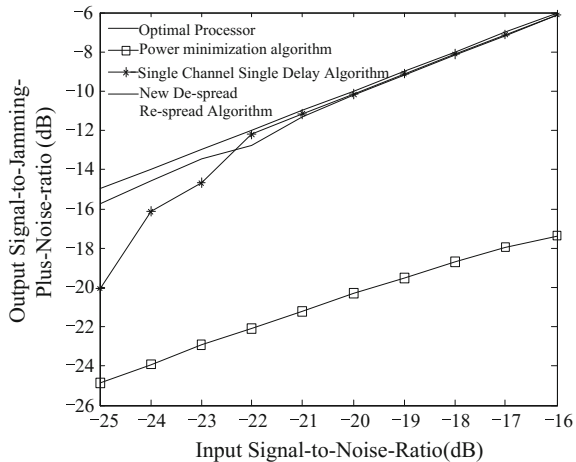
(a) Power minimization algorithm



(b) New de-spread re-spread algorithm

Figure 2.25 compares the acquisition results on the same GNSS signal using the new de-spread and re-spread algorithm and the power minimization algorithm. Among them, Fig. 2.25a is the result of the power minimization algorithm; Fig. 2.25b is the result for the new de-spread and re-spread algorithm. By comparing Fig. 2.25a, b, it can be seen that, after jamming suppression, the power minimization algorithm can acquire the GNSS signal, but can not provide the signal processing gain brought by the antenna array, thereby the normalized cross correlation number's sidelobe level is high. But the new de-spread and re-spread

**Fig. 2.26** Comparisons of signal-to-jamming-plus-noise ratio curves for different algorithms



algorithm can make the antenna mainlobe aim towards the direction of the GNSS signal, so that its acquisition result is better than that of the power minimization algorithm,

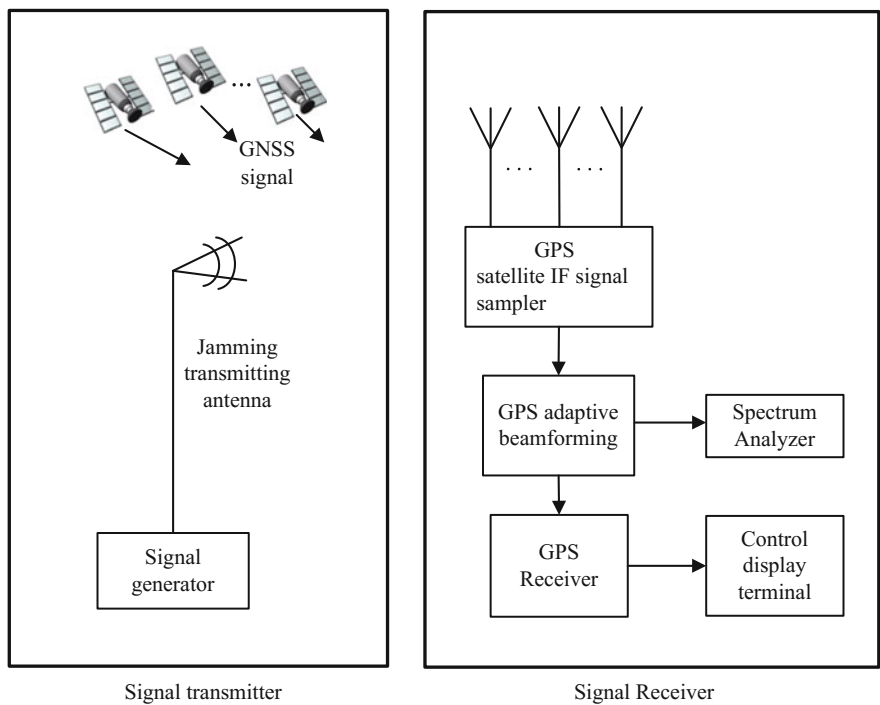
### 3. Comparisons of Output Signal-jamming-plus-noise Ratio

Figure 2.26 compares the output signal-jamming-plus-noise ratios for different algorithms. Among them, the solid line corresponds to the optimal processor; the dotted line corresponds to the new de-spread, re-spread algorithm; the “\*” line corresponds to the single channel single delay cross correlation algorithm, and the “□” line corresponds to the power minimization algorithm. It can be seen in Fig. 2.26 that since the power minimization algorithm has no beam directions, it cannot improve a GNSS signal’s carrier-to-noise ratio. Both the new de-spread re-spread algorithm and the single channel single delay cross correlation algorithm can improve the carrier-to-noise ratio of the GNSS signal.

## 2.5.4 Results on Hardware Platform Experiments

### 1. Brief Introduction on Hardware Platform Experiments

The new de-spread re-spread algorithm has the characteristics of high gain and high robustness. We have developed a digital multi-beam jamming mitigation real time processing system based on the algorithm. Based on the block diagram in the Fig. 2.21, the system can mainly be implemented using two main modules: the jamming mitigation module and the beamforming module. Figure 2.27 shows the experimental block diagram. In the Fig. 2.27, the experimental system is composed of two main parts: the signal transmitter and the receiver. The jamming signal in the transmitter is produced by a signal generator, and the GNSS signal refers to the

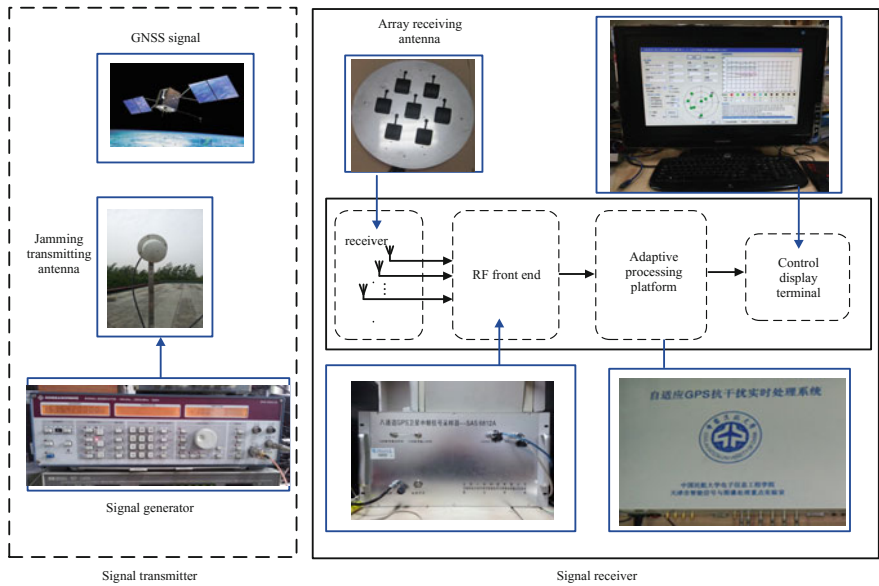


**Fig. 2.27** Experimental block diagram for beam jamming mitigation real time processing system

signal transmitted by the actual GPS satellite. The receiver receives the signal using an array antenna first, then the received signal is converted to a digital IF signal using a down-converter and A/D sampling quantilization, and finally the digital IF signal is fed into the adaptive processing platform, After jamming suppression, GPS receiver acquisition, tracking and positioning, the results are transmitted to a personal computer control terminal for displaying the results of tracking and positioning. The effects of signal jamming can be observed by monitoring the frequency spectrum change before and after the jamming mitigation.

Figure 2.28 shows the experimental scene diagram for the digital multi-beam jamming mitigation real time processing system test. The left subfigure is the signal transmitter, with the top-left subfigure representing the GNSS signal and the bottom-left subfigure representing the signal source that generate the jamming signal to which a jamming transmitting antenna is connected. The right subfigure is the signal receiver, and in the order of processing there is a 7-element array antenna, IF signal sampler including an active downconverter, adaptive processing platform and computer display terminal. The sampling rate is 5.714286 MHz, and the intermediate frequency (IF) is 4.309 MHz.



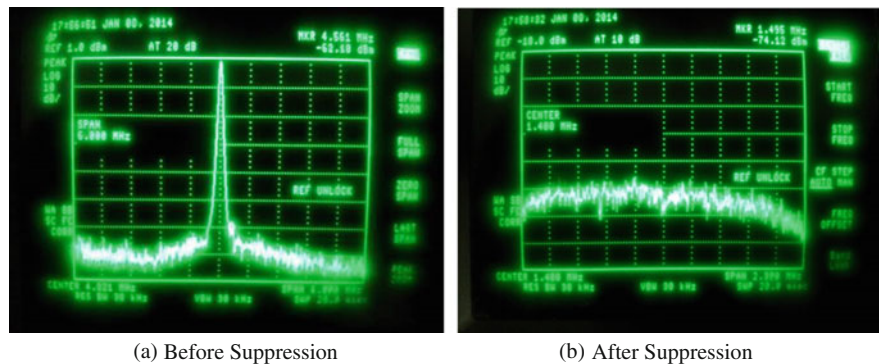


**Fig. 2.28** Experimental scene diagrams for digital multi-beam jamming mitigation real time processing system test

2. Hardware Platform Testing Results

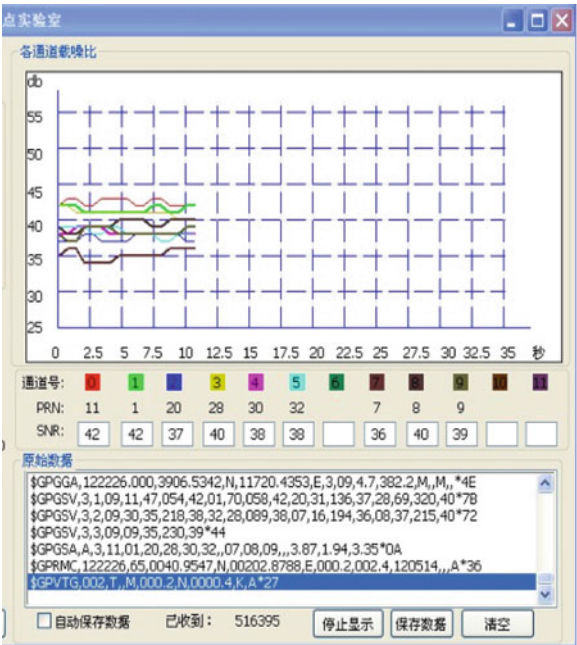
Figure 2.29 shows the IF signal frequency spectrum observed on the spectrum analyzer before and after the jamming suppression under the experiment conditions described in Fig. 2.27. The jamming signal produced by the signal generator has a frequency of 1575.42 MHz, and a jamming transmitting power of  $-10$  dBm (jamming-to-noise ratio is 70 dB).

It can be seen in Fig. 2.29, after adaptive system processing, the peak of the signal spectrum is removed, so the jamming signal is effectively suppressed.

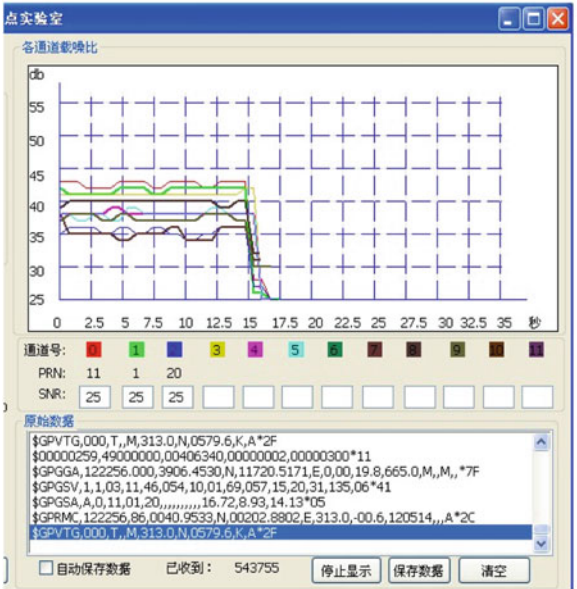


**Fig. 2.29** Signal frequency spectrum comparisons before and after jamming suppression

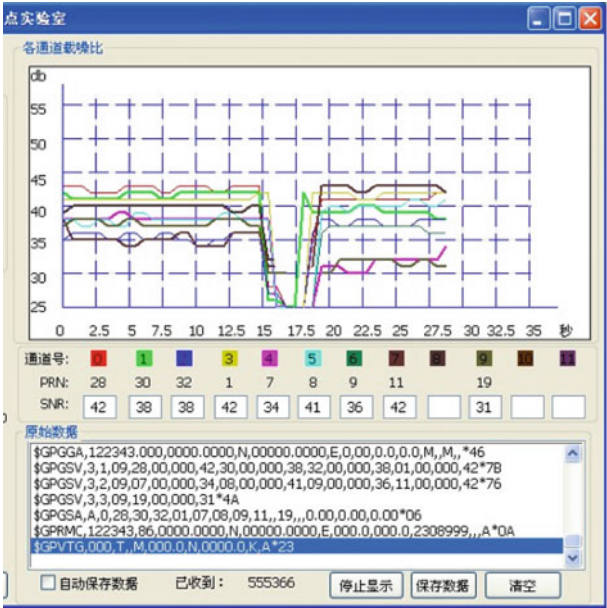
**Fig. 2.30** Tracking results obtained from the GPS receiver display terminal before and after adaptive beamforming processing



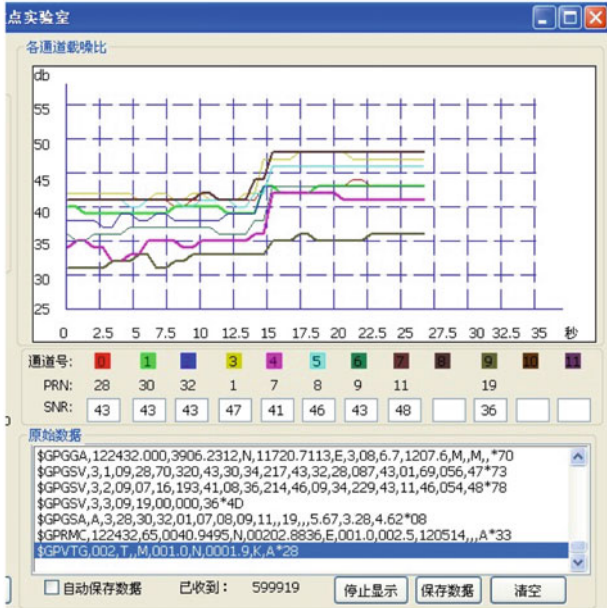
(a) No Jamming



(b) With jamming



(c) After jamming suppression.



(d) After the beamforming is added.

Fig. 2.30 (continued)



with that of after the jamming mitigation due to the signal processing gain brought by the antenna array. This is because the GPS receiver module has acquisition function towards the low signal-to-noise visible satellites. In addition, at different moments, the visible satellites can change, so that after jamming mitigation, the system can track the PRN20 satellite. The experiment results show that while the system implemented suppresses the jamming signal, it forms a higher beam gain towards the direction of the satellites, so that effectively improves the signal's carrier-to-noise ratio.

## 2.6 Space Time Adaptive Filtering

Space-time Adaptive Processing (STAP) was proposed by Brennan et al. [48] in 1973 to solve the ground clutter suppression problem that shows a space-time two-dimensional coupled distribution for airborne radar. It essentially expands the one-dimensional spatial domain filtering technique to time and space two-dimensional domain, to form Time-Space two-dimensional processing structure. Reed et al. [49], proposed the Sample Matrix Inverse (SMI) algorithm to compute adaptive weight. After this, to solve the problem of large computational load of two-dimensional optimal processing, Klemm proposed the auxiliary channel (ACR) [42] algorithm, Hong W. et al. proposed the joint domain localized processing (JDL) algorithm [50] and the  $\sum - \Delta$  STAP algorithm [51], and Pao Zheng et al. proposed the method of "time first, space next" adaptive cascade processing method and partial joint adaptive processing method (abbreviated as two-dimensional Capon method and 3DT method) [47]. All these algorithms are computation-efficient. In addition, to promote the applicability of the STAP, scholars globally have studied problems such as reduced ranking processing and STAP under non-uniform environment.

For airborne and missile-borne receiver applications, the array antenna apertures are usually restricted. When there are various types of jamming (e.g. narrow band and wideband jamming, dispersion multipath interference etc.), only spatial domain processing cannot provide enough degrees of freedom to have good jamming suppression performance. Consequently, Fante et al. applied the STAP technique on GPS jamming mitigation, to greatly improve the degrees of freedom of the adaptive system. They also studied the criterion of the space-time jamming mitigation, the impact from multipath jamming, and the GPS signal distortion caused by STAP [16, 52–54] etc.

### 2.6.1 Space-Time Processing Data Model

Considering a uniform linear array composed of  $M$  array elements, and each array element has  $K$  taps with a tap delay of  $T$ . The STAP structure is shown in Fig. 2.31.

It can be seen on every array element channel that a FIR filter is constructed using delays at all levels that can be used in the temporal domain to remove jamming. From the perspectives of the nodes with the same time delays, different array elements constitute adaptive filter in the spatial domain, so it can identify spatial jamming sources, and suppress jamming by forming spatial domain nulls. Consequently, space-time processing has the ability to eliminate the jamming in the space-time domain.

We assume that after down-conversion, the received baseband signal  $x(t)$  at the reference point at time  $t$  is represented as

$$x(t) = s(t) + \sum_{q=1}^Q j_q(t) + e(t) \quad (2.143)$$

where  $s(t)$  denotes the satellite baseband signal;  $j_q(t)$  denotes the baseband signal of the  $q$ th jamming signal;  $e(t)$  is the additive Gaussian White noise. The data received by the array at time  $t$  can be written in the form of a  $MK \times 1$  dimensional vector, i.e.

$$\mathbf{x}(t) = [x_{11}(t), x_{12}(t), \dots, x_{1K}(t), \dots, x_{M1}(t), \dots, x_{MK}(t)]^T \quad (2.144)$$

where  $x_{mk}(t)$  is the received signal at time  $t$  for the  $m$ th array's  $k$ th delay unit, and it can be represented as

$$x_{mk}(t) = s_{mk}(t) + \sum_{q=1}^Q j_{qmk}(t) + e_{mk}(t) \quad (2.145)$$

For the GNSS signal in (2.145), the first array element of the uniform linear array is used as the reference point. By referring to the definition in (2.12), the GNSS signal received by the  $m$ th array's  $k$ th delay unit at time  $t$  can be written as

$$s_{mk}(t) = s(t - (k-1)T) e^{-j \frac{2\pi(m-1)d \sin \theta}{\lambda}} \quad (2.146)$$

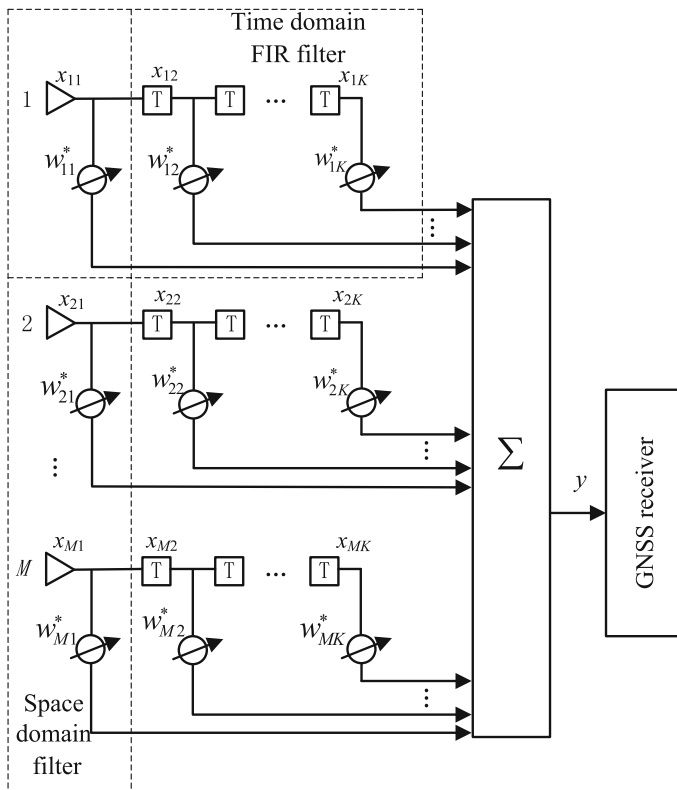
where  $\theta$  denotes the angle of arrival of the satellite signal, and  $d$  denotes interval between array elements, and  $\lambda$  denotes the wavelength of the GNSS signal.

The GNSS signal received by the antenna array can be written at time  $t$  in the form of a  $MK \times 1$  dimensional vector, i.e.

$$\bar{\mathbf{s}}(t) = [s_{11}(t), s_{12}(t), \dots, s_{1K}(t), \dots, s_{M1}(t), \dots, s_{MK}(t)]^T = \mathbf{A}\mathbf{s}(t) \quad (2.147)$$

where  $\mathbf{s}(t) = [s(t), s(t-T), \dots, s(t-(K-1)T)]^T$  is the delayed GNSS signal at the  $K$ th level, and  $\mathbf{A} = \mathbf{I}_{K \times K} \otimes \mathbf{a}(\theta)$ ,  $\mathbf{a}(\theta)$  is the spatial domain steering vector (reference (2.12)), and  $\otimes$  denotes the Kronecker product.

In Fig. 2.31,  $\{w_{mk}\} (m = 1, 2, \dots, M; k = 1, 2, \dots, K)$  is the space-time two-dimensional weight vector, and  $w_{mk}$  is the weight for the  $k$ th delay unit of



**Fig. 2.31** STAP block diagram

the  $m$ th array element. The weight vector is written in the form of  $MK \times 1$  dimensional vector, i.e.

$$\mathbf{w} = [w_{11}, w_{12}, \dots, w_{1K}, \dots, w_{M1}, \dots, w_{MK}]^T \quad (2.148)$$

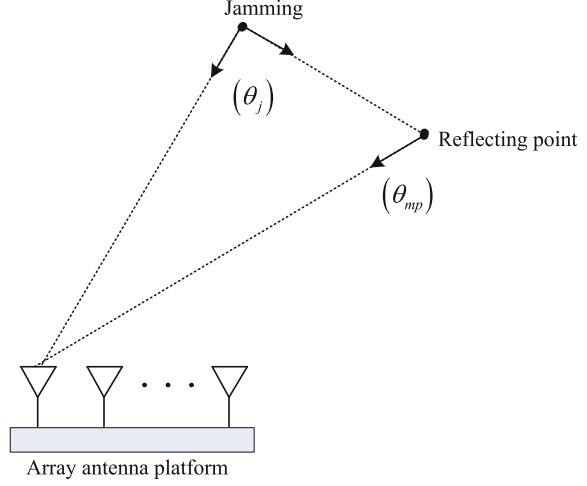
Then, the output the space-time processor is

$$y(t) = \mathbf{w}^H \mathbf{x}(t) \quad (2.149)$$

For the jamming component in (2.145), if multipath interference is not considered, its model is very similar to the model of the GNSS signal. If the jamming signal and the jamming signal's multipath signal incident on the array antenna simultaneously, as shown in Fig. 2.32, the received jamming signal becomes a combined jamming signal after the directly arrived wave and the reflected wave interfere with each other. This scenario happens more often when a GNSS receiver antenna is installed on a mobile platform. Using an airplane example, due to the reflections of the fuselage and the wings, the reflection waves of the jamming



**Fig. 2.32** Multipath jamming



among different antennas are only partially correlated, i.e. the array generates the dispersion phenomenon, also known as dispersion multipath jamming [55]. Below we use airborne dispersion multipath jamming as an example, to illustrate the necessity of the STAP processing.

For the  $q$ th jamming signal  $j_q(t)$ , if the array antenna receives another even larger reflected jamming signal relative to the LOS jamming signal, then the jamming signal received by the  $m$ th antenna becomes

$$J_{qm}(t) = j_{qm}(t) + \alpha j_{qm}(t - \tau_{mp}) \quad (2.150)$$

where the time delay of the multipath jamming relative to the LOS jamming is  $\tau_{mp}$ ; the multipath attenuation coefficient is  $\alpha$ . The jamming signal received by the array can be written in the form of a  $M \times 1$  dimension vector as  $\mathbf{J}_q(t) = [J_{q1}(t), J_{q2}(t), \dots, J_{qM}(t)]^T$ . Then the array covariance matrix for the jamming is

$$\begin{aligned} \mathbf{R}_j &= E\left\{\mathbf{J}_q(t)\mathbf{J}_q^H(t)\right\} \\ &= \sigma_j^2 \begin{bmatrix} \mathbf{a}(\theta_j) & \mathbf{a}(\theta_{mp}) \end{bmatrix} \begin{bmatrix} 1 & \alpha r^*(\tau_{mp}) \\ \alpha^* r(\tau_{mp}) & |\alpha|^2 \end{bmatrix} \begin{bmatrix} \mathbf{a}(\theta_j) & \mathbf{a}(\theta_{mp}) \end{bmatrix}^H \end{aligned} \quad (2.151)$$

where  $\sigma_j^2$  is the power of jamming signal;  $\theta_j$  is the direction of LOS jamming;  $\theta_{mp}$  is the direction of multipath multipath;  $\mathbf{a}(\theta_j)$  is the array steering vector of LOS jamming;  $\mathbf{a}(\theta_{mp})$  is the array steering vector of the multipath jamming;  $r(\tau_{mp})$  is the correlation function between the LOS jamming and the multipath jamming.

Under the above given assumptions, if the correlation function between the LOS jamming and the multipath jamming  $r(\tau_{mp})$  is not equal to 1, then the rank of the



jamming covariance matrix is 2, and the array shows dispersion phenomenon. For this same reason, we can derive that, if one LOS jamming and  $P$  multipath jamming incident on the array simultaneously, since the received signals among different antennas are not correlated anymore, the rank of the jamming covariance matrix is larger than 1, and at maximum it can reach  $P + 1$ . Under the condition, if only spatial domain array process is used, multiple degrees of freedom are consumed to resist one jamming source. For  $M$ -element antenna array spatial domain processing, at maximum  $M - 1$  nulls can be formed. When dispersion jamming exists, the reduction of degrees of freedom can degrade the performance of only spatial domain jamming suppression. STAP can greatly improve the degrees of freedom for adaptive processing, and furthermore it can improve performance by resisting dispersion jamming.

When wideband jammings exist, similar to the case above, the signals received by various array elements are not correlated with each other anymore. Therefore, STAP is also needed to improve the degrees of freedom for adaptive processing.

### 2.6.2 Space-Time Power Minimization Algorithm

The process used in space-time adaptive jamming mitigation uses different space-time algorithms to derive weight vectors for the array, so that the array output can be rendered immune from the impacts of jamming. This optimum criterion is the theoretical foundation in determining the space-time processing optimal weight vector. Different optimum weight vectors satisfying different requirements can be derived based on different criteria. Among them, the space-time power minimization algorithm [8] is widely applied on many GNSS receivers based on array antennas because it does not need any a priori information and the implementation is simple.

The power minimization algorithm, if the GNSS signal at the receiver is rather weak (around 20 dB lower than the noise level) and the jamming signal is very strong (often much larger than the noise level), determines the weight vector by minimizing the array output signal power. The method requires the weight of the first tap to be 1, and then a weight vector is selected to minimize the output power, i.e.

$$\begin{aligned} \min_w \mathbf{w}^H \mathbf{R}_x \mathbf{w} \\ \text{s.t. } \mathbf{w}^H \boldsymbol{\delta}_{MK} = 1 \end{aligned} \quad (2.152)$$

where  $\mathbf{R}_x = E\{\mathbf{x}(t)\mathbf{x}^H(t)\}$  is the theoretical covariance matrix for received signals by the array. In actual computation, it is usually substituted using the sample covariance matrix  $\hat{\mathbf{R}}_x = \frac{1}{N} \sum_{n=1}^N \mathbf{x}(n)\mathbf{x}^H(n)$ ;  $\boldsymbol{\delta}_{MK} = [1, 0, \dots, 0]^T$  is a  $MK \times 1$  dimension vector. Based on the Lagrange principle, the weight vector can be derived as

$$\mathbf{w}_{opt} = \frac{\mathbf{R}_x^{-1} \delta_{MK}}{\delta_{MK}^H \mathbf{R}_x^{-1} \delta_{MK}} \quad (2.153)$$

Even though the power minimization method does not need any a priori information and can be easily implemented, it cannot provide beam gain brought by array processing. So it cannot maximize the output signal to jamming-plus-noise ratio.

### 2.6.3 Space-Time De-spread Re-spread Algorithm

As described in Sect. 2.5.2, the blind adaptive algorithm based on de-spread re-spread algorithm takes advantage of the fact that the GNSS signal's spread spectrum code is known, so that it can simultaneously achieve array signal processing gain and jamming suppression with no need to know the directions of GNSS signals beforehand. Thereby, to expand the de-spread re-spread technique to the space-time domain, we can obtain space-time de-spread re-spread jamming suppression algorithm, and its block diagram is shown in Fig. 2.33.

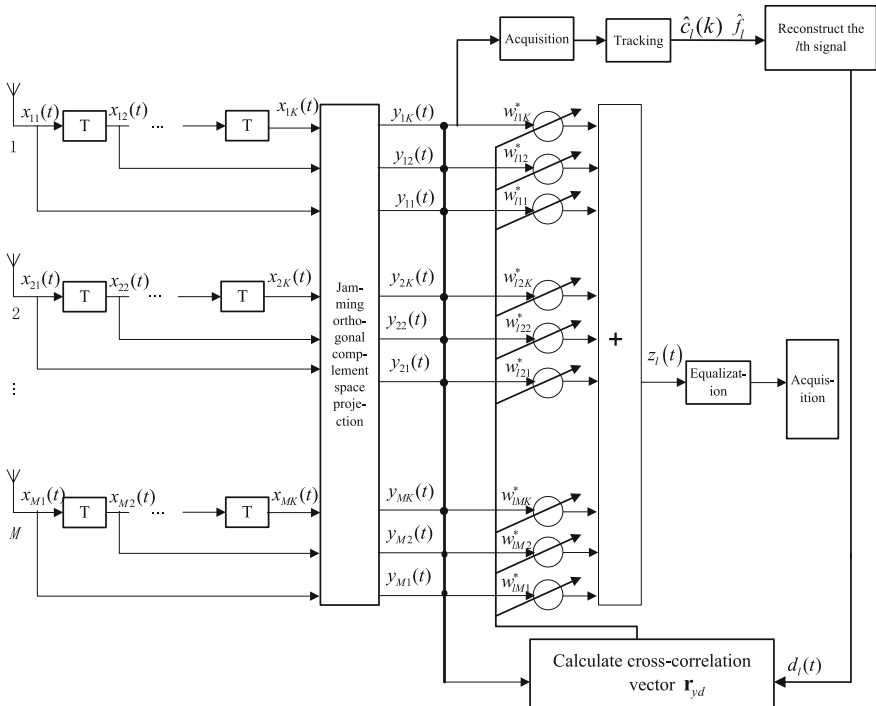


Fig. 2.33 Block diagram of space-time de-spread re-spread algorithm

First, we project the space-time data towards the inverse matrix of the covariance matrix, i.e.

$$\mathbf{y}(t) = \mathbf{R}_x^{-1} \mathbf{x}(t) \quad (2.154)$$

Then to perform acquisition and tracking on any one signal in (2.154), we can obtain the PRN code  $c_l(t - \hat{\tau}_l)$ , frequency  $\hat{\omega}_l$  and navigation message  $\hat{D}_l(t - \hat{\tau}_l)$  synchronized with the  $l$ th GNSS signal. From this information the  $l$ th GNSS signal can be reconstructed, i.e.

$$d_l(t) = \hat{D}_l(t) c_l(t - \hat{\tau}_l) e^{j\hat{\omega}_l(t - \hat{\tau}_l)} \quad (2.155)$$

We can reconstruct GNSS signals received from all  $L$  satellites or those satellites satisfying the position requirements using this method.

By performing correlation between reconstructed  $l$ th GNSS signal and the projected space-time data, the derived cross-correlation is

$$\hat{\mathbf{r}}_{yd} = \sum_{n=1}^N \mathbf{y}(n) d_l^*(n) \quad (2.156)$$

Based on this, similar to (2.142), we can obtain the space-time de-spread re-spread weight vector

$$\mathbf{w}_{lopt} = \mathbf{R}_x^{-1} \hat{\mathbf{r}}_{yd} \quad (2.157)$$

By repeating the steps described in (2.155)–(2.157), we can obtain the corresponding beams for  $L$  GNSS signals, and every beam points to one GNSS satellite.

#### 2.6.4 Reduced Rank Space-Time Adaptive Filtering Algorithm

Good jamming suppression effects can be achieved by applying STAP technique on GNSS receivers. But, STAP processing involves the inverse of two-dimensional high order matrix. Not only does it require a large amount of computation, it also has a higher demand on the number of snapshots. Therefore, it is very important to perform reduced rank simplification on STAP [56]. It not only can greatly reduce the amount of computation, it also reduces the speed of convergence. In this section, we introduce some commonly used reduced rank STAP jamming suppression algorithms presently, including Principal Components (PC) [56], Cross Spectral Metric (CSM) [57], Multistage Nested Wiener Filter (MWF) [58–60], and Multistage Iterative Reduced Rank (MIRR).

Before introducing these commonly used reduced rank algorithms, we give the unified structure of reduced rank space-time adaptive jamming suppression algorithm based on the Wiener filter structure shown in Fig. 2.34.

Weiner-Hopf equation's solution is

$$\mathbf{w}_0 = \mathbf{R}_x^{-1} \mathbf{r}_{xd_0} \triangleq (E\{\mathbf{x}\mathbf{x}^H\})^{-1} E\{\mathbf{x}d_0^*\} \quad (2.158)$$

where  $d_0$  denotes the desired signal;  $\mathbf{x}$  is the two-dimension snapshot vector corresponding to the observed signal received by space-time processor;  $\mathbf{w}_0$  is the weight vector of space-time processor;  $\hat{d}_0 = \mathbf{w}_0^H \mathbf{x}$  is the estimate on the desired signal;  $\mathbf{R}_x = E\{\mathbf{x}\mathbf{x}^H\}$  is the covariance matrix corresponding to the observed signal;  $\mathbf{r}_{xd_0} = E\{\mathbf{x}d_0^*\}$  is the cross-correlation vector between the observed signal and the desired signal. The minimum mean square error of the output corresponding to (2.158) is

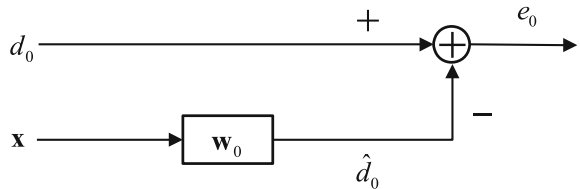
$$MMSE = \sigma_{d_0}^2 - \mathbf{r}_{xd_0}^H \mathbf{R}_x^{-1} \mathbf{r}_{xd_0} \quad (2.159)$$

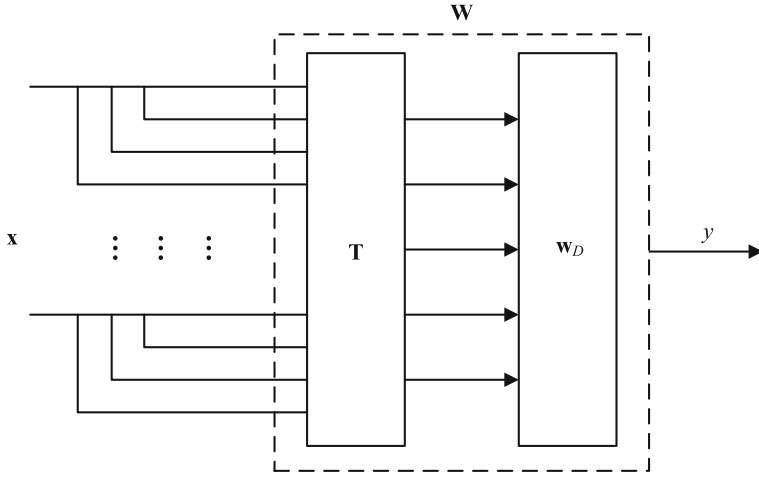
where  $\sigma_{d_0}^2 = E\{d_0 d_0^*\}$  is the average power of the desired signal. It can be seen in (2.158) that the weight vector  $\mathbf{w}_0$  makes  $\hat{d}_0$  approach the signal component of the desired signal  $d_0$  that is correlated with the observed signal, i.e.  $\hat{d}_0$  is the projection component of the observed signal  $\mathbf{x}$  on the direction of cross-correlation  $\mathbf{r}_{xd_0}$ .

To solve for the classic Wiener filter's optimum weight vector, an inverse operation on the covariance matrix of the observed signal received by space-time processor is needed. The amount of computation is very large and the speed of convergence is slow (i.e. very high requirement on the number of snapshots). Thus further illustrates the necessity to perform reduced rank processing. By performing reduced rank processing, a smaller subspace can be created to approximate observation space that can greatly reduce the computation complexity and achieve faster speed of convergence. Theoretically, when the dimension of the reduced rank is reduced down to the space dimension of the jamming signal, the impacts on the processor's jamming suppression performance are not significant.

The reduced rank STAP jamming suppression algorithm can be expressed in a unified fashion: by using a  $MK \times D$  dimension reduced rank transformation matrix  $\mathbf{T} = [\mathbf{t}_1, \mathbf{t}_2, \dots, \mathbf{t}_D]$  (where  $D < MK$  represents the dimension after reducing the rank of the input observation signal), so that the dimension of the observed signal is reduced from  $MK$  dimension to  $D$  dimension, and the  $D$ -dimension adaptive

**Fig. 2.34** Block diagram of a classical Wiener filter





**Fig. 2.35** Unified structure for reduced rank space-time adaptive jamming suppression algorithm

processing can be done in this reduced rank space. A  $D \times 1$  dimension adaptive weight vector  $\mathbf{w}_D$  can then be obtained, as shown in Fig. 2.35.

After performing  $\mathbf{T}$  transformation reduced rank processing on the observed signal, the new input observed signal vector obtained is

$$\mathbf{x}_D = \mathbf{T}^H \mathbf{x} \quad (2.160)$$

The covariance matrix corresponding to the observed data after rank reduction is

$$\mathbf{R}_D = \mathbf{T}^H \mathbf{R}_x \mathbf{T} \quad (2.161)$$

It can be seen that the dimension of covariance matrix reduces from  $MK \times MK$  to  $D \times D$ . Thereby, to solve the space-time adaptive weight vector after rank reduction  $\mathbf{w}_D$ , the amount of computation needed for covariance matrix inverse operation can be reduced significantly. The overall weight vector of the space-time adaptive processor is

$$\mathbf{w} = \mathbf{T} \mathbf{w}_D \quad (2.162)$$

The key for reduced rank processing is to select the proper reduced rank transformation matrix  $\mathbf{T}$ , so that as the rank of the processor reduces, we still can achieve performance close to that of a full-dimensional optimum processor.

### 1. Principal Component Method

The main idea of the principal component method (PC) [56] is to perform eigenvalue decomposition on the observed data's covariance matrix  $\mathbf{R}_x$ , to obtain eigenvalue  $\lambda_i$  and the corresponding eigenvector  $\mathbf{v}_i$ , i.e.

$$\mathbf{R}_x = \mathbf{V} \mathbf{\Lambda} \mathbf{V}^H = \sum_{i=1}^{MK} \lambda_i \mathbf{v}_i \mathbf{v}_i^H \quad (2.163)$$

where the diagonal element of the diagonal matrix  $\mathbf{\Lambda}$  is  $\mathbf{R}_x$ 's eigenvalue;  $\mathbf{V}$  is the matrix composed of column vectors of  $\mathbf{R}_x$ 's eigenvector.

The GNSS signal reaching the receiver is much weaker than the noise level, while the jamming signal is fairly strong and is way above the noise level, thereby the space expanded from the eigenvector corresponding to the  $\mathbf{R}_x$ 's large eigenvalues can be regarded as the jamming subspace, while the space expanded from the eigenvector corresponding to  $\mathbf{R}_x$ 's small eigenvalues can be regarded as the noise subspace. By arranging  $\mathbf{R}_x$ 's eigenvalues in descending order (i.e.  $\lambda_1 \geq \lambda_2 \geq \dots \geq \lambda_{MK}$ ), the eigenvectors  $\mathbf{v}_i$  corresponding to the first  $D$  largest eigenvalues  $\lambda_i$  can be used as the reduced rank matrix, i.e.

$$\mathbf{T} = \mathbf{V}_D = [\mathbf{v}_1, \mathbf{v}_2, \dots, \mathbf{v}_D] \quad (2.164)$$

Now, the observed data's covariance matrix  $\mathbf{R}_x$  can be approximated using the following formula

$$\mathbf{R}_x \approx \mathbf{R}_{PC} = \mathbf{V}_D \mathbf{\Lambda}_D \mathbf{V}_D^H = \sum_{i=1}^D \lambda_i \mathbf{v}_i \mathbf{v}_i^H \quad (2.165)$$

where the diagonal matrix  $\mathbf{\Lambda}_D = \text{diag}\{\lambda_1, \lambda_2, \dots, \lambda_D\}$ , and is a  $D \times D$  dimensional matrix; the reduced rank eigenvector  $\mathbf{V}_D$  is a  $MK \times D$  dimension matrix, and the reduced rank dimension  $D \leq MK - 1$ . If  $D$  is bigger than the dimension of the jamming subspace, the jamming information can be preserved. Then, the principal component method's weight vector is

$$\mathbf{w}_{PC} = \mathbf{R}_{PC}^{-1} \mathbf{r}_{xd0} = \sum_{i=1}^D \frac{1}{\lambda_i} \mathbf{v}_i \mathbf{v}_i^H \mathbf{r}_{xd0} \quad (2.166)$$

If all jamming components that need to be suppressed are all contained in the reduced rank matrix  $\mathbf{T}$ , the principal component method has very good reduced rank performance. But the PC method approximates the auto-correlation matrix based on the eigenvalues' amplitudes only. It does not consider the space-time two-dimension steering vector of the desired signal. Consequently, the space compression is not complete. When there is characteristic spectrum expansion and subspace leakage, the performance of the PC method's compression usually experiences sharp degradation. Aiming towards the above questions, we introduce the cross spectral metric to improve the principal component method.

## 2. Cross Spectral Metric

The main idea of Cross spectral metric (CSM) [57] is to use the amplitude of cross spectral energy as the criterion to select reduced rank matrix. By substituting the observed data covariance matrix's eigenvector decomposition expansion formula (2.163) into the Wiener filter output's minimum mean square error expression (2.159), we can obtain

$$\begin{aligned} MMSE &= \sigma_{d_0}^2 - \mathbf{r}_{xd_0}^H \mathbf{R}_x^{-1} \mathbf{r}_{xd_0} = \sigma_{d_0}^2 - \mathbf{r}_{xd_0}^H \sum_{i=1}^{MK} \frac{\mathbf{v}_i \mathbf{v}_i^H}{\lambda_i} \mathbf{r}_{xd_0} \\ &= \sigma_{d_0}^2 - \sum_{i=1}^{MK} \frac{|\mathbf{v}_i^H \mathbf{r}_{xd_0}|^2}{\lambda_i} \triangleq \sigma_{d_0}^2 - \sum_{i=1}^{MK} \frac{\rho_i^2}{\lambda_i} \end{aligned} \quad (2.167)$$

where  $\rho_i = |\mathbf{v}_i^H \mathbf{r}_{xd_0}|$  denotes the space correlation between the  $i$ th eigenvector and the desired signal's space-time two-dimension steering vector. By defining  $\rho_i^2 / \lambda_i$  as cross spectral energy, we maximize the overall cross spectral energy by minimizing the output MMSE in formula (2.167). Therefore, the cross spectral metric method selects the eigenvectors  $\tilde{\mathbf{v}}_i$  corresponding to the first  $D$  number of eigenvalues  $\tilde{\lambda}_i$  that maximize cross spectrum energy as the reduced rank matrix, i.e.

$$\mathbf{T} = \tilde{\mathbf{V}}_D = [\tilde{\mathbf{v}}_1, \tilde{\mathbf{v}}_2, \dots, \tilde{\mathbf{v}}_D] \quad (2.168)$$

Then, the observed data's covariance matrix  $\mathbf{R}_x$  can be approximated as

$$\mathbf{R}_x \approx \mathbf{R}_{CSM} = \tilde{\mathbf{V}}_D \tilde{\Lambda}_D \tilde{\mathbf{V}}_D^H = \sum_{i=1}^D \tilde{\lambda}_i \tilde{\mathbf{v}}_i \tilde{\mathbf{v}}_i^H \quad (2.169)$$

Then, we can obtain the cross spectral metric method's weight vector

$$\mathbf{w}_{CSM} = \mathbf{R}_{CSM}^{-1} \mathbf{r}_{xd_0} = \sum_{i=1}^D \frac{1}{\tilde{\lambda}_i} \tilde{\mathbf{v}}_i \tilde{\mathbf{v}}_i^H \mathbf{r}_{xd_0} \quad (2.170)$$

The data compression effect of the cross spectrum measure method is better than that of the principal component method. But, both the cross spectrum measure method and the principal component method need to perform eigenvalue decomposition on the observed data's covariance matrix to obtain reduced rank subspace, so that the amount of computation required is still large. To some degree, this constraint limits the application of these methods in actual engineering practices. The multistage nested wiener filter method introduced below can avoid matrix inverse and eigenvalue decomposition so it can reduce the computation complexity for the above two problems.

### 3. Multistage Nested Wiener Filter

The multistage Nested Wiener Filter (MWF) [58] is a multistage equivalent realization of the Wiener filter. It uses a series of orthogonal projection to perform multistage decomposition on observed input signal vector, then performs multistage scalar Wiener filtering to synthesize the output error signal of the Wiener filter. Compared with the PC and CSM methods that are based on feature decomposition, the MWF method avoids complex operations such as matrix inverse and feature decomposition.

Before we introduce multistage Wiener filter, we must first give the following theorems [38].

**Theorem 2.1** *Using full rank matrix  $\mathbf{T} \in \mathbb{C}^{MK \times MK}$ , we perform pre-filtering on observed data  $\mathbf{x}_0$  (originally  $\mathbf{x}$ ) to obtain new observed data  $\mathbf{z}_1 = \mathbf{T}\mathbf{x}_0$ . The Wiener filter weight vector  $\mathbf{w}_{z_1}$  and estimated signal  $\hat{d}_0 = \mathbf{w}_{z_1}^H \mathbf{z}_1$  are also derived based on this data, and the MMSE derived is the same as that of no prefiltering.*

Below we give proof on Theorem 2.1, to derive the Wiener filtering performance on full rank matrix pre-filtering. Figure 2.36 shows the block diagram of using full rank matrix  $\mathbf{T} \in \mathbb{C}^{MK \times MK}$  to perform pre-filtering processing on observed data  $\mathbf{x}_0$ .

New observed data after pre-filtering can be written as

$$\mathbf{z}_1 = \mathbf{T}\mathbf{x}_0 \quad (2.171)$$

The corresponding new mean square error is

$$MSE_{z_1} = \sigma_{d_0}^2 - \mathbf{w}_{z_1}^H \mathbf{r}_{z_1 d_0} - \mathbf{w}_{z_1} \mathbf{r}_{z_1 d_0}^H + \mathbf{w}_{z_1}^H \mathbf{R}_{z_1} \mathbf{w}_{z_1} \quad (2.172)$$

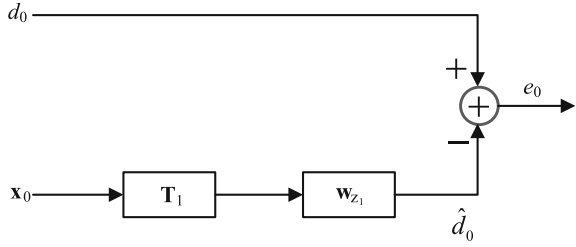
where  $\mathbf{R}_{z_1} = E\{\mathbf{z}_1 \mathbf{z}_1^H\} = E\{\mathbf{T}\mathbf{x}_0 \mathbf{x}_0^H \mathbf{T}^H\} = \mathbf{T}\mathbf{R}_{x_0} \mathbf{T}^H$  is the covariance matrix for the new observed data;  $\mathbf{r}_{z_1 d_0} = E\{\mathbf{z}_1 d_0^*\} = E\{\mathbf{T}\mathbf{x}_0 d_0^*\} = \mathbf{T}\mathbf{r}_{x_0 d_0}$  is the cross-correlation between new observed data and expected data. The derived Wiener filtering weight vector  $\mathbf{w}_{z_1}$  can be represented as

$$\begin{aligned} \mathbf{w}_{z_1} &= \mathbf{R}_{z_1}^{-1} \mathbf{r}_{z_1 d_0} \\ &= [\mathbf{T}^{-1}]^H \mathbf{R}_{x_0}^{-1} \mathbf{T}^{-1} \mathbf{T} \mathbf{r}_{x_0 d_0} = [\mathbf{T}^{-1}]^H \mathbf{R}_{x_0}^{-1} \mathbf{r}_{x_0 d_0} \\ &= [\mathbf{T}^{-1}]^H \mathbf{w}_{x_0} \end{aligned} \quad (2.173)$$

The new desired signal estimated based on new observed data is

$$\hat{d}_0 = \mathbf{w}_{z_1}^H \mathbf{z}_1 = \mathbf{w}_{x_0}^H \mathbf{T}^{-1} \mathbf{T} \mathbf{x}_0 = \mathbf{w}_{x_0}^H \mathbf{x}_0 \quad (2.174)$$



**Fig. 2.36** Wiener filter including pre-filtering

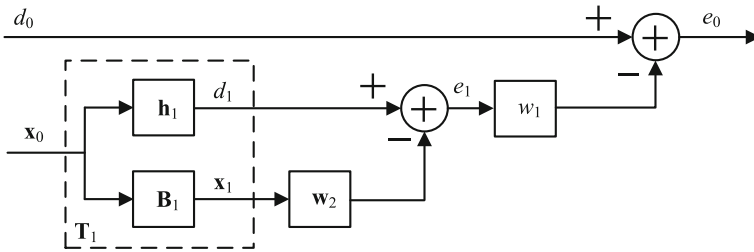
It can be seen that the estimated results are the same as those before pre-filtering. Now the new MMSE can be recalculated. Under the minimum mean square error criterion, we can derive using (2.172)

$$\begin{aligned}
 MMSE_{z_1} &= \sigma_{d_0}^2 - \mathbf{w}_{z_1}^H \mathbf{R}_{z_1} \mathbf{w}_{z_1} \\
 &= \sigma_{d_0}^2 - \mathbf{w}_{x_0}^H \mathbf{T}^{-1} \mathbf{T} \mathbf{R}_{x_0} \mathbf{T}^H [\mathbf{T}^{-1}]^H \mathbf{w}_{x_0} \\
 &= \sigma_{d_0}^2 - \mathbf{w}_{x_0}^H \mathbf{R}_{x_0} \mathbf{w}_{x_0} \\
 &= MMSE_{x_0}
 \end{aligned} \tag{2.175}$$

It can be seen that the output MMSE is the same as before pre-filtering. Based on Theorem 2.1, we can use the full rank pre-filtering matrix

$$\mathbf{T}_1 = \begin{bmatrix} \mathbf{h}_1^H \\ \mathbf{B}_1 \end{bmatrix} \in \mathbb{C}^{MK \times MK} \tag{2.176}$$

where  $\mathbf{h}_1 = \mathbf{r}_{x_0 d_0} / \|\mathbf{r}_{x_0 d_0}\| \in \mathbb{C}^{MK \times 1}$  is a normalized cross-correlation vector;  $\mathbf{B}_1 = \text{null}(\mathbf{h}_1) \in \mathbb{C}^{(MK-1) \times MK}$  is a  $(MK-1) \times MK$  dimension matrix orthogonal to  $\mathbf{h}_1$ , and is regarded as the blocking matrix, and its expanded space is  $\mathbf{h}_1$ 's null space, i.e.  $\mathbf{B}_1 \mathbf{h}_1 = 0$ . Then, Fig. 2.37 shows the structure of a Wiener filter. Where  $d_1 = \mathbf{h}_1^H \mathbf{x}_0$  is  $\mathbf{x}_0$ 's projection on the projection direction of the cross-correlation vector  $\mathbf{r}_{x_0 d_0}$ ;  $\mathbf{h}_1$  ensures that  $d_1$  and  $d_0$  have the largest correlation;  $\mathbf{x}_1 = \mathbf{B}_1 \mathbf{x}_0$  is a  $MK-1$

**Fig. 2.37** Two-stage Wiener filter structure

dimension vector, and it corresponds to the projection  $\mathbf{x}_0$ 's projection on its  $MK - 1$  dimension subspace orthogonal to cross-correlation vector  $\mathbf{r}_{x_0 d_0}$ , not containing any information in  $d_0$ .

The new observed data after pre-filtering the observed data  $\mathbf{x}_0$  can be represented as

$$\mathbf{z}_1 = \mathbf{T}_1 \mathbf{x}_0 = \begin{bmatrix} \mathbf{h}_1^H \mathbf{x}_0 \\ \mathbf{B}_1 \mathbf{x}_0 \end{bmatrix} = \begin{bmatrix} d_1 \\ \mathbf{x}_1 \end{bmatrix} \quad (2.177)$$

Its corresponding covariance matrix is

$$\mathbf{R}_{z_1} = \begin{bmatrix} \sigma_{d_1}^2 & \mathbf{r}_{x_1 d_1}^H \\ \mathbf{r}_{x_1 d_1} & \mathbf{R}_{x_1} \end{bmatrix} \quad (2.178)$$

where

$$\begin{aligned} \sigma_{d_1}^2 &= E\{d_1 d_1^H\} = \mathbf{h}_1^H \mathbf{R}_{x_0} \mathbf{h}_1 \\ \mathbf{r}_{x_1 d_1} &= E\{\mathbf{x}_1 d_1^*\} = \mathbf{B}_1 \mathbf{R}_{x_0} \mathbf{h}_1 \\ \mathbf{R}_{x_1} &= E\{\mathbf{x}_1 \mathbf{x}_1^H\} = \mathbf{B}_1 \mathbf{R}_{x_0} \mathbf{B}_1^H \end{aligned} \quad (2.179)$$

The  $\mathbf{R}_{x_0}$  in the above formula is the original  $\mathbf{R}_x$ . The cross-correlation vector between the new observed data and the desired data is

$$\mathbf{r}_{z_1 d_0} = E\{\mathbf{z}_1 d_0^*\} = \mathbf{T}_1 \mathbf{r}_{x_0 d_0} = \begin{bmatrix} \|\mathbf{r}_{x_0 d_0}\| \\ \mathbf{0}_{(MK-1) \times 1} \end{bmatrix} \quad (2.180)$$

Therefore, the Wiener filtering weight vector corresponds to the data in (2.177) is

$$\mathbf{w}_{z_1} = \mathbf{R}_{z_1}^{-1} \mathbf{r}_{z_1 d_0} \quad (2.181)$$

By applying the block Hermite matrix inverse formula, we can derive

$$\mathbf{R}_{z_1}^{-1} = \xi_1^{-1} \begin{bmatrix} 1 & -\mathbf{r}_{x_1 d_1}^H \mathbf{R}_{x_1}^{-1} \\ -\mathbf{R}_{x_1}^{-1} \mathbf{r}_{x_1 d_1} & \mathbf{R}_{x_1}^{-1} (\xi_1 \mathbf{I} + \mathbf{r}_{x_1 d_1} \mathbf{r}_{x_1 d_1}^H \mathbf{R}_{x_1}^{-1}) \end{bmatrix} \quad (2.182)$$

where  $\xi_1 = E\{|e_1|^2\} = \sigma_{d_1}^2 - \mathbf{r}_{x_1 d_1}^H \mathbf{R}_{x_1}^{-1} \mathbf{r}_{x_1 d_1}$ . By substituting (2.182) into (2.181), we can derive the Wiener filtering weight vector after pre-filtering [59]

$$\begin{aligned}
\mathbf{w}_{z_1} &= \zeta_1^{-1} \begin{bmatrix} 1 & -\mathbf{r}_{x_1 d_1}^H \mathbf{R}_{x_1}^{-1} \\ -\mathbf{R}_{x_1}^{-1} \mathbf{r}_{x_1 d_1} & \mathbf{R}_{x_1}^{-1} (\zeta_1 \mathbf{I} + \mathbf{r}_{x_1 d_1} \mathbf{r}_{x_1 d_1}^H \mathbf{R}_{x_1}^{-1}) \end{bmatrix} \begin{bmatrix} \|\mathbf{r}_{x_0 d_0}\| \\ \mathbf{0}_{(MK-1) \times 1} \end{bmatrix} \\
&= \zeta_1^{-1} \|\mathbf{r}_{x_0 d_0}\| \begin{bmatrix} 1 \\ -\mathbf{R}_{x_1}^{-1} \mathbf{r}_{x_1 d_1} \end{bmatrix} \\
&= w_1 \begin{bmatrix} 1 \\ -\mathbf{w}_2 \end{bmatrix}
\end{aligned} \tag{2.183}$$

It can be seen that, an  $MK$  dimension Wiener weight vector can be decomposed into a form of one scalar and an  $MK - 1$  dimension weight vector. Among them, the scalar  $w_1 = \zeta_1^{-1} \|\mathbf{r}_{x_0 d_0}\|$ , the  $MK - 1$  dimension weight vector  $\mathbf{w}_2 = \mathbf{R}_{x_1}^{-1} \mathbf{r}_{x_1 d_1}$  is the Wiener solution of the estimated scalar  $d_1$  based on  $MK - 1$  dimension data  $\mathbf{x}_1$ .

Using the same method we can decompose the Wiener filter stage by stage, as shown in Fig. 2.38, until it can be decomposed to the  $(MK - 1)$ th stage. Then, we can simplify the matrix inverse as the reciprocal of multiple scalars.  $\mathbf{h}_r$  of every decomposition is the cross-correlation between the previous upper branch's desired signal and the lower branch's observed data  $\mathbf{x}_{r-1}$ , in order to preserve the information from the previous step as much as possible. The block matrix of every decomposition  $\mathbf{B}_r$  can ensure orthogonalities among ever reduced rank components. After Multistage decomposition, we can still obtain the same output MMSE as the original Wiener filtering.

Actually, there is no need to completely decompose the filter. Only  $D$ -step decompositions ( $D \leq MK - 1$ ) can obtain almost all useful information. Then, the observed data  $\mathbf{x}_D$  contains almost no jamming components, and its covariance matrix tends to whiten, so that decomposition can stop [60]. As shown in Fig. 2.38, a multistage Wiener filter is composed of an analysis filter and a synthetic filter. The analysis filter uses an orthogonal projection transformation to perform decomposition, and the synthetic filter uses a set of iterative Wiener filters for synthesis, and its input is the output of the analysis filter. The detailed steps of the original multistage Wiener reduced rank algorithm's iterative process are listed in Table 2.2. In the table,  $\mathbf{B}_{ptran} = \mathbf{B}_p^H$ .

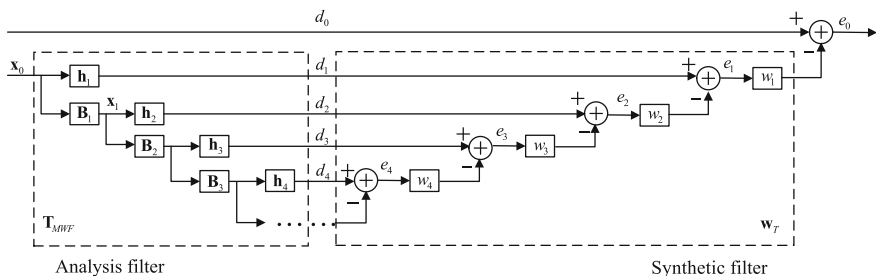


Fig. 2.38 Multistage Wiener filter structure

**Table 2.2** Detailed steps of original multistage Wiener filtering algorithm

Analysis filter
Initialization: $\mathbf{R}_x = \mathbf{R}_{x_0} = E\{\mathbf{x}_0 \mathbf{x}_0^H\}$ , $\mathbf{r}_{xd} = \mathbf{r}_{x_0 d_0} = E\{\mathbf{x}_0 d_0^H\}$ , $\mathbf{B}_p = \mathbf{I}_{MK-1}$ , $\mathbf{B}_{ptran} = \mathbf{I}_{MK-1}$
Iterative computation: for $i = 1, 2, \dots, D$ $\delta_i = \sqrt{\mathbf{r}_{xd}^H \mathbf{r}_{xd}}$ , $\mathbf{h}_i = \mathbf{r}_{xd} / \delta_i$ , $\sigma_i^2 = \mathbf{h}_i^H \mathbf{R}_x \mathbf{h}_i$ , $\mathbf{B}_i = \text{null}(\mathbf{h}_i)$ , $\mathbf{t}_i = \mathbf{B}_i^H \mathbf{h}_i$ , $\mathbf{B}_{ptran} = \mathbf{B}_p^H$ , $\mathbf{B}_p = \mathbf{B}_i \mathbf{B}_p$ , $\mathbf{r}_{xd} = \mathbf{B}_p \mathbf{R}_{x_0} \mathbf{B}_{ptran} \mathbf{h}_i$ , $\mathbf{R}_x = \mathbf{B}_p \mathbf{R}_{x_0} \mathbf{B}_p^H$
Synthetic filter
Initialization: $\xi_D = \sigma_D^2$ , $w_D = \delta_D / \xi_D$
Iterative computation: for $i = D-1, D-2, \dots, 2, 1$ $\xi_i = \sigma_i^2 - \delta_{i+1}^2 / \xi_{i+1}$ , $w_i = \delta_i^2 / \xi_i$
Weight computation
Initialization: $\mathbf{w}_r = 0$ , $w_p = 1$
Iterative computation: for $i = 1, 2, \dots, D$ $w_p = -w_p w_i^*$ , $\mathbf{w}_r = \mathbf{w}_r + \mathbf{t}_i w_p$
When iteration completes, we can obtain: $\mathbf{w}_{MWF} = \mathbf{w}_r$

Based on the above iterative solution process for the original multistage Wiener filter, we can implement the reduced rank filter for the input observed signal. Below we give the explicit expression of the method, i.e. (2.184)–(2.186). After completing the decomposition, the equivalent reduced rank matrix can be expressed as

$$\mathbf{T}_{MWF} = [\mathbf{t}_1, \mathbf{t}_2, \dots, \mathbf{t}_D] = \left[ \mathbf{h}_1, \mathbf{B}_1^H \mathbf{h}_2, \dots, \left( \prod_{i=1}^{D-1} \mathbf{B}_i^H \right) \mathbf{h}_D \right] \quad (2.184)$$

$[\mathbf{t}_1, \mathbf{t}_2, \dots, \mathbf{t}_D]$  is the basis vector of the reduced rank subspace. The equivalent weight vector after rank reduction can be expressed as

$$\mathbf{w}_D = \left[ w_1^*, -w_1^* w_2^*, \dots, (-1)^{D-1} \prod_{i=1}^{D-1} w_i^* \right]^T = \left( \mathbf{T}_{MWF}^H \mathbf{R}_{x_0} \mathbf{T}_{MWF} \right)^{-1} \mathbf{T}_{MWF}^H \mathbf{r}_{x_0 d_0} \quad (2.185)$$

The overall weight matrix can be denoted as

$$\mathbf{w}_{MWF} = \mathbf{T}_{MWF} \mathbf{w}_D \quad (2.186)$$

Different block matrices can enable different implementations for multistage Wiener filter. Goldstein, Reed and Scharf, who originally proposed the multistage Wiener filter, provided a method to compute a block matrix in the appendix of

Ref. [58]. The implementation method is known as GRS-MWF. The vector is represented as

$$\mathbf{h}_i = [h_{i1}, h_{i2}, \dots, h_{i(MK-i)}] \quad (2.187)$$

Then GRS-MWF's block matrix is

$$\mathbf{B}_i = \begin{bmatrix} \frac{1}{h_{i1}} & \frac{-1}{h_{i2}} & 0 & \cdots & 0 & 0 & 0 \\ 0 & \frac{1}{h_{i2}} & \frac{-1}{h_{i3}} & \cdots & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & \frac{1}{h_{i(MK-i-2)}} & \frac{-1}{h_{i(MK-i-1)}} & 0 \\ 0 & 0 & 0 & \cdots & 0 & \frac{1}{h_{i(MK-i-1)}} & \frac{-1}{h_{i(MK-i)}} \end{bmatrix} \quad (2.188)$$

It can be seen that  $\mathbf{B}_i$  is a  $(MK - i - 1) \times (MK - i)$  rectangular array. Since  $\mathbf{x}_i = \mathbf{B}_i \mathbf{x}_{i-1}$ , we know that the dimension of  $\mathbf{x}_i$  decrements by 1 as  $i$  increases, and that it is very beneficial for reducing computation load and storage requirement. By truncating the multistage Wiener filter after the  $D$ th step ( $D < MK - 1$ ), we can obtain a  $D$ -order reduced rank multistage filter, and this is also known as a multistage Wiener filter with rank  $D$ .

Multistage Wiener filter can also be implemented using the correlation subtraction structure called CSA-MWF, as shown in Fig. 2.39. The structure is equivalent to a block matrix as

$$\mathbf{B}_i = \mathbf{I}_i - \frac{\mathbf{h}_i \mathbf{h}_i^H}{\|\mathbf{h}_i\|^2} \quad (2.189)$$

Furthermore the analysis filter in Fig. 2.37 can also be simplified. As shown in Fig. 2.39, the CSA-MWF structure is based on a data field, so it does not need to explicitly compute the input observation signal's covariance matrix and block matrix. Therefore, the amount of computation can be greatly reduced compared with the GRS-MWF structure.

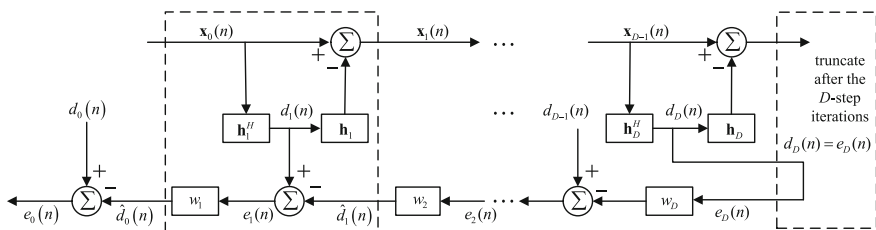


Fig. 2.39 Data field CSA-MWF implementation structure

**Table 2.3** Detailed steps for data field CSA-MWF algorithm

Synthetic filter
Initialization: $d_0(n) = \delta_{MK}^H \mathbf{x}(n)$ , $\mathbf{x}_0(n) = \mathbf{x}(n) - \delta_{MK} d_0(n)$
Iterative computation: for $i = 1, 2, \dots, D$ $\mathbf{h}_i = E\{\mathbf{x}_{i-1}(n)d_{i-1}^*(n)\} / \ E\{\mathbf{x}_{i-1}(n)d_{i-1}^*(n)\}\ $ , $d_i(n) = \mathbf{h}_i^H \mathbf{x}_{i-1}(n)$ , $\mathbf{x}_i(n) = \mathbf{x}_{i-1}(n) - \mathbf{h}_i(n)d_i(n)$
Synthetic filter
Initialization: $e_D(n) = d_D(n)$
Synthetic filter
Iterative computation: for $i = D, D-1, \dots, 2, 1$ $w_i = E\{e_i(n)d_{i-1}^*(n)\} / E\{\ e_i(n)\ ^2\}$ , $e_{i-1}(n) = d_{i-1}(n) - w_i^* e_i(n)$
Weight calculation
Initialization: $w_p = 1$
Iterative computation: for $i = 1, 2, \dots, D$ $w_p = -w_p w_i^*$ , $w_i = w_p$

The detailed steps for the data field CSA-MWF algorithm iterative process are shown in Table 2.3.

Below we give the explicit expression for the method, i.e. (2.190)–(2.192). After completing the decomposition, the equivalent reduced rank matrix can be expressed as

$$\mathbf{T}_{MWF} = [\delta_{MK}, \mathbf{h}_1, \dots, \mathbf{h}_D] \quad (2.190)$$

After rank reduction, the equivalent weight vector can be expressed as

$$\mathbf{w}_D = [1, w_1, \dots, w_D]^T \quad (2.191)$$

The overall weight vector can be denoted as

$$\mathbf{w}_{MWF} = \mathbf{T}_{MWF} \mathbf{w}_D \quad (2.192)$$

#### 4. Multistage Iterative Reduced Rank Method

It can be discovered from the Multistage Nested Wiener Filter introduced above that these methods use iterative computation to avoid the processes of covariance matrix inverse and feature decomposition so the amount of computation can be reduced. Among them, the data field's CSA-MWF implementation structure improves the computational complexity on the analysis filter by not calculating the covariance matrix, as well as reduces every iteration's computation complexity by calculating cross correlations among vectors to derive new desired signal and

observed data. This method then iterates backwards in the synthetic filter to determine the weight vector. Even though it is a scalar Wiener filter structure, but they nest with each other so that they can not be realized using parallel computation, thereby the overall real time performance of the algorithm suffers. The synthetic filter structure can be improved by making it execute one iteration for every added step, e.g. compute the weight vector in parallel. This parallel multistage iterative structure after improving both the analysis filter and synthetic filter is known as the Multistage Iterative Reduced Rank method.

From (2.185) and (2.186), the overall weight vector can be represented as

$$\mathbf{w}_{MWF} = \mathbf{T} \left( \mathbf{T}^H \mathbf{R}_{\mathbf{x}_0} \mathbf{T} \right)^{-1} \mathbf{T}^H \mathbf{r}_{x_0 d_0} \quad (2.193)$$

where we can simplify so that  $\mathbf{T} = \mathbf{T}_{MWF}$ . For the multistage Wiener filter structure shown in Fig. 2.36 with a rank of  $D$ , the  $i$ th desired signal after analysis filter decomposition can be written as

$$d_i = \left( \prod_{j=1}^{i-1} \mathbf{B}_j^H \right) \mathbf{h}_i \mathbf{x}_0 = \mathbf{t}_i^H \mathbf{x}_0 \quad (2.194)$$

By writing various stages of desired signals into vector form representation, we have

$$\mathbf{d}^{(D)} = [d_1, d_2, \dots, d_D]^T = \left[ \mathbf{T}^{(D)} \right]^H \mathbf{x}_0 \quad (2.195)$$

By substituting (2.195) into (2.193), we can obtain the overall weight vector as

$$\mathbf{w}_{MWF} = \mathbf{T}^{(D)} \left[ \mathbf{R}_d^{(D)} \right]^{-1} \mathbf{r}_{dd_0} \quad (2.196)$$

where  $\mathbf{R}_d^{(D)} = E \{ \mathbf{d}^{(D)} [\mathbf{d}^{(D)}]^H \} = [\mathbf{T}^{(D)}]^H \mathbf{R}_{\mathbf{x}_0} \mathbf{T}^{(D)}$  is a transformation covariance matrix, and we derive

$$\begin{aligned} \mathbf{R}_d^{(D)}(i, i) &= E \{ d_i d_i^* \} = \sigma_{d_i}^2 \\ \mathbf{R}_d^{(D)}(i, i-1) &= E \{ d_i d_{i-1}^* \} = E \{ \mathbf{h}_i^H \mathbf{x}_{i-1} d_{i-1}^* \} = \mathbf{h}_i^H \mathbf{r}_{\mathbf{x}_{i-1} d_{i-1}} = \delta_i \\ \mathbf{R}_d^{(D)}(i-1, i) &= E \{ d_{i-1} d_i^* \} = E \{ d_{i-1} \mathbf{x}_{i-1}^H \mathbf{h}_i \} = \mathbf{r}_{\mathbf{x}_{i-1} d_{i-1}}^H \mathbf{h}_i = \delta_i, \quad i = 2, 3, \dots, D \end{aligned} \quad (2.197)$$

Therefore,  $\mathbf{R}_d^{(D)}$  is a real number matrix in the form of three diagonal matrices

$$\mathbf{R}_d^{(D)} = \begin{bmatrix} \sigma_{d_1}^2 & \delta_2 & 0 & \cdots & 0 & 0 \\ \delta_2 & \sigma_{d_2}^2 & \delta_3 & \cdots & 0 & 0 \\ 0 & \delta_3 & \sigma_{d_3}^2 & \ddots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \delta_{D-1} & 0 \\ 0 & 0 & 0 & \delta_{D-1} & \sigma_{d_{D-1}}^2 & \delta_D \\ 0 & 0 & 0 & 0 & \delta_D & \sigma_{d_D}^2 \end{bmatrix} = \begin{bmatrix} \mathbf{R}_d^{(D-1)} & \mathbf{0} \\ \mathbf{0} & \delta_D & \sigma_{d_D}^2 \end{bmatrix} \quad (2.198)$$

Also in (2.196)

$$\mathbf{r}_{dd_0} = E\left\{\mathbf{d}^{(D)}d_0^*\right\} = E\left\{\left[\mathbf{T}^{(D)}\right]^H \mathbf{x}_0 d_0^*\right\} = \left[\mathbf{T}^{(D)}\right]^H \mathbf{r}_{x_0 d_0} = \left[\mathbf{T}^{(D)}\right]^H \mathbf{h}_1 \delta_1 = \begin{bmatrix} \delta_1 \\ \mathbf{0} \end{bmatrix} \quad (2.199)$$

It can be known from (2.199) that the weight vector in (2.196) only needs to be used in the first row of  $[\mathbf{R}_d^{(D)}]^{-1}$ , and multiply it by  $\delta_1$ . By defining

$$\mathbf{C}^{(D)} = \begin{bmatrix} \mathbf{C}_1^{(D)} & \mathbf{C}_2^{(D)} & \cdots & \mathbf{C}_D^{(D)} \end{bmatrix} = \left[\mathbf{R}_d^{(D)}\right]^{-1} \quad (2.200)$$

and further expanding (2.200) based on the block Hermite matrix inversion formula, we can obtain

$$\mathbf{C}^{(D)} = \left[\mathbf{R}_d^{(D)}\right]^{-1} = \begin{bmatrix} \mathbf{C}^{(D-1)} & \mathbf{0} \\ \mathbf{0} & 0 \end{bmatrix} + \beta_D^{-1} \mathbf{b}^{(D)} \left[\mathbf{b}^{(D)}\right]^H \quad (2.201)$$

$$\beta_D = \sigma_{d_D}^2 - \delta_D^2 \mathbf{C}_{D-1,D-1}^{(D-1)} \quad (2.202)$$

$$\mathbf{b}^{(D)} = \begin{bmatrix} -\delta_D \mathbf{C}_{D-1}^{(D-1)} \\ 1 \end{bmatrix} \quad (2.203)$$

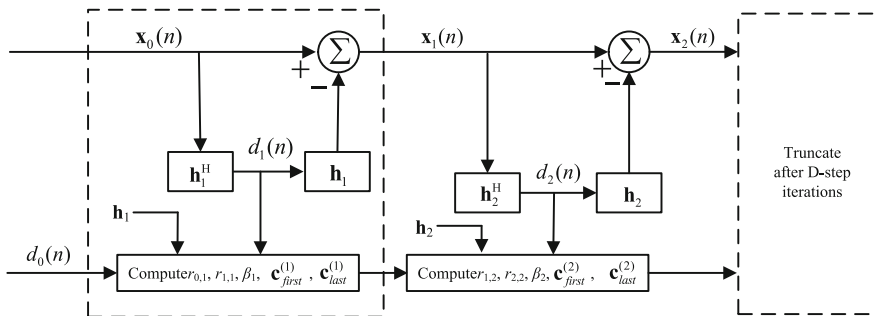
where  $\mathbf{C}_{i,j}^{(D)}$  denotes the  $i$ th element of the  $j$ th column  $\mathbf{C}_j^{(D)}$ . We only need the first row  $\mathbf{C}_1^{(D)}$  of  $\left[\mathbf{R}_d^{(D)}\right]^{-1}$  to compute the weight vector in (2.196).

$$\mathbf{C}_1^{(D)} = \begin{bmatrix} \mathbf{C}_1^{(D-1)} \\ 0 \end{bmatrix} + \beta_D^{-1} \mathbf{C}_{1,D-1}^{(D-1)} \begin{bmatrix} \delta_D^2 \mathbf{C}_{D-1}^{(D-1)} \\ -\delta_D \end{bmatrix} \quad (2.204)$$

where  $\mathbf{C}_D^{(D)} = \beta_D^{-1} \begin{bmatrix} -\delta_D \mathbf{C}_{D-1}^{(D-1)} \\ 1 \end{bmatrix}$ ,  $\mathbf{C}_{D,D}^{(D)} = \beta_D^{-1}$ ,  $\beta_D = \sigma_{d_D}^2 - \delta_D^2 \beta_{D-1}^{-1}$ . Since  $\mathbf{C}_1^{(D)}$  can be solved iteratively based on (2.204), the overall weight vector is

$$\mathbf{w}_{MWF} = \delta_1 \mathbf{T}^{(D)} \mathbf{C}_1^{(D)} \quad (2.205)$$





**Fig. 2.40** Implementation structure of multi-step iteration

From (2.205), we extrapolate that for every incremental stage of the filter, only one iteration needs to be added, so that the recursive updating for backward iteration can be avoided. If the analysis filter adopts the CSA-MWF implementation structure, and at the same time the synthetic filter adopts the backward iterative algorithm, the constructed multistage iterative implementation structure is shown in Fig. 2.40.

Detailed steps of multi-step iterative algorithm are listed in Table 2.4.

We can summarize the various reduced rank methods introduced above: When the jamming subspace components are all contained in the reduced rank matrix  $\mathbf{T}$ , the principal component method has very good reduced rank performance; When there exist a characteristic spectrum expansion and subspace leakage, the cross spectral metric method has better data compression effect; The multistage nested

**Table 2.4** Detailed steps of the multi-step iteration algorithm

Initialization
$d_0(n) = \delta_{MK}^H \mathbf{x}(n)$ , $\mathbf{x}_0(n) = \mathbf{x}(n) - \delta_{MK} d_0(n)$ , $\mathbf{h}_1 = E\{\mathbf{x}_0(n)d_0^*(n)\} / \ E\{\mathbf{x}_0(n)d_0^*(n)\}\ $ , $d_1(n) = \mathbf{h}_1^H \mathbf{x}_0(n)$ , $\mathbf{x}_1(n) = \mathbf{x}_0(n) - \mathbf{h}_1(n)d_1(n)$ , $r_{0,1} = 0$ , $r_{1,1} = E\{d_1(n)d_1^*(n)\}$ , $\mathbf{c}_{\text{first}}^{(1)} = r_{1,1}^{-1}$ , $\mathbf{c}_{\text{last}}^{(1)} = r_{1,1}^{-1}$
Iterative computation
for $i = 2 : MK - 1$ $\mathbf{h}_i = E\{\mathbf{x}_{i-1}(n)d_{i-1}^*(n)\} / \ E\{\mathbf{x}_{i-1}(n)d_{i-1}^*(n)\}\ $ , $d_i(n) = \mathbf{h}_i^H \mathbf{x}_{i-1}(n)$ , $\mathbf{x}_i(n) = \mathbf{x}_{i-1}(n) - \mathbf{h}_i(n)d_i(n)$ , $r_{i,i} = E\{d_i(n)d_i^*(n)\}$ , $r_{i-1,i} = E\{d_{i-1}(n)d_i^*(n)\}$ , $\beta_i = r_{i,i} -  r_{i-1,i} ^2 \mathbf{c}_{\text{last},i-1}^{(i-1)}$ , $\mathbf{c}_{\text{first}}^{(i)} = \begin{bmatrix} \mathbf{c}_{\text{first}}^{(i-1)} \\ 0 \end{bmatrix} + \beta_i^{-1} \mathbf{c}_{\text{last},1}^{(i-1)*} \begin{bmatrix}  r_{i-1,i} ^2 \mathbf{c}_{\text{last}}^{(i-1)} \\ -r_{i-1,i}^* \end{bmatrix}$ , $\mathbf{c}_{\text{last}}^{(i)} = \beta_i^{-1} \begin{bmatrix} -r_{i-1,i} \mathbf{c}_{\text{last}}^{(i-1)} \\ 1 \end{bmatrix}$ , $MSE^{(i)} = \sigma_{d_0}^2 - \ \mathbf{r}_{\mathbf{x}_0 d_0}\ _2^2 \mathbf{c}_{\text{first},1}^{(i)}$
If $MSE^{(D)} < MSE_{\text{threshold}}$ , Then $D = i$ , jump out
Weight calculation
$\mathbf{T}_D = [\mathbf{t}_1, \mathbf{t}_2, \dots, \mathbf{t}_D] = [\mathbf{h}_1, \dots, \mathbf{h}_D]$ , $\mathbf{w}_{MWF} = \ \mathbf{r}_{\mathbf{x}_0 d_0}\ _2 \mathbf{T}_D \mathbf{c}_{\text{first}}^{(D)}$

Wiener filter method avoids performing matrix inverse and eigenvalue decomposition; The multistage Wiener filter with correlated subtractive structure does not need to explicitly compute observed data's covariance matrix, and does not need to compute the block matrix, therefore it can further reduce dynamic range and amount of computation. The multi-step iterative reduced rank method achieves parallel computation of the weight vector, so the real-time performance of the reduced rank algorithm improves significantly.

### 2.6.5 Impacts on GNSS Signal by Space-Time Processing and Equalization Technique

Since a space-time processing method performs jamming suppression in the joint space-time domain, frequency responses over the whole processing bandwidth are not consistent, inevitably leading to distortions on GNSS signals, as a result cross correlation functions between array outputs and local signals have phenomena such as mainlobe broadening, cross correlation peak shift etc. Those phenomena have impacts on GPS signal acquisition, pseudo-range measurement, and determination of user positions [61]. This section analyzes the impacts of space-time processing on GNSS signals, and introduces some existing main STAP compensation techniques, including weight constraint [61], temporal-domain filter [61], least squares inverse filtering and equalization method based on homomorphic filtering principle [62].

Array output signal after the space-time adaptive jamming mitigation is

$$y(n) = \mathbf{w}^H \mathbf{x}(n) = \mathbf{w}^H \mathbf{A} \mathbf{s}(n) + \mathbf{w}^H \mathbf{j}(n) + \mathbf{w}^H \mathbf{e}(n) \quad (2.206)$$

where  $\mathbf{x}(n)$  denotes the space-time two-dimension snapshot;  $\mathbf{w}$  denotes the adaptive weight vector corresponding to the STAP algorithm;  $\mathbf{A} = \mathbf{I}_{K \times K} \otimes \mathbf{a}(\theta)$ ,  $\mathbf{a}(\theta) = \left[ 1, e^{-j\frac{2\pi d \sin \theta}{\lambda}}, \dots, e^{-j\frac{2\pi(M-1)d \sin \theta}{\lambda}} \right]^T$  is related to the actual direction of arrival  $\theta$  for GNSS signals;  $\mathbf{s}(n)$  is the  $K$ -stage delayed GNSS signal;  $\mathbf{j}(n)$  and  $\mathbf{e}(n)$ , respectively, are the jamming and noise components of space-time two dimensional snapshot.

To suppress and eliminate the jamming, it is equivalent to

$$\mathbf{w}^H \mathbf{j}(n) \approx 0 \quad (2.207)$$

Then the output of the array is

$$y(n) \approx \mathbf{w}^H \mathbf{A} \mathbf{s}(n) + \mathbf{w}^H \mathbf{e}(n) = \mathbf{w}^H \mathbf{A} \mathbf{s}(n) + \bar{e}(n) \quad (2.208)$$

where  $\bar{e}(n) = \mathbf{w}^H \mathbf{e}(n)$  represents noise output.

After the STAP processing, uniform linear array response coefficients for different time delays can be represented using a  $K \times 1$  dimension vector  $\mathbf{h} = (\mathbf{w}^H \mathbf{A})^T$ , i.e.

$$\mathbf{h} = [h(0), h(1), \dots, h(K-1)]^T \quad (2.209)$$

where

$$h(k-1) = \sum_{m=1}^M w_{mk} e^{-j \frac{2\pi(M-1)d \sin \theta}{\lambda}} \quad (2.210)$$

By transforming (2.210) to frequency domain, the GNSS signal's frequency response can be obtained using space-time processing

$$\begin{aligned} H(\omega) &= \sum_{k=1}^K h(k-1) e^{-j\omega(k-1)T} \\ &= \sum_{k=1}^K \sum_{m=1}^M w_{mk} e^{-j \frac{2\pi(M-1)d \sin \theta}{\lambda} - j\omega(k-1)T} \end{aligned} \quad (2.211)$$

If it is assumed that the frequency response of a satellite transmitting signal  $s(t)$  is  $S(\omega)$ , then the frequency response for the output after the STAP processing is  $S(\omega)H(\omega)$ . Therefore, we can obtain the output signal after the STAP for the GNSS signal, and it has the following format

$$y(n) = \frac{1}{2\pi} \int_{-\pi}^{\pi} S(\omega) H(\omega) e^{j\omega n} d\omega \quad (2.212)$$

It can be known from distortion-free transmission conditions  $H(\omega) = K e^{-j\omega t_0}$  (where  $K$  and  $t_0$  are constants) that when the STAP's frequency response to the GNSS signal  $H(\omega)$  does not satisfy the above conditions, the GNSS signal has distortion.

To compensate for the impacts on the GNSS signal by the STAP process, scholars all over the world have performed related research studies. Among them, the weight constraint method proposed by professor Hatke from MIT [61] performs equalization on all distortions of the GNSS signal, to make their impacts on the distortions the same. This method adds an orthogonal constraint condition to solve new weights for different jamming mitigation algorithms. The MITRE Corporation in the US proposed an equalization algorithm based on a temporal-domain filter [61] to make different GNSS beams' STAP responses consistent, i.e. let every GNSS signal's beam output convolute with STAP responses of other GNSS signals using an added temporal-domain filter. The University of Chinese National Defense proposed the method of adding an inverse filter. The method, based on the principle

of deterministic least squares inverse filter, solves for inverse filter coefficients for every GNSS signal, and adds array outputs, to recover the desired GPS signal. Since linear filter has difficulty separating the product signal or convolution signal, the China Civil Aviation University proposed an equalization algorithm based on homomorphic filtering [62]. The method transforms a convolution type combined signal to an additive signal, then designs a filter to separate signals in it, so that responses of STAP at different delay taps are consistent. Below we introduce the basic principles of these methods one by one.

### 1. Weighted Constraint Equalization Algorithm

Weighted Constraint Equalization Algorithm [61] directly adds constraint conditions on jamming mitigation weights of the STAP process, and re-solves for new weights. From (2.211) we can obtain the response of the space-time processing on the GNSS signal as:  $\mathbf{h} = [h(0), h(1), h(2), \dots, h(K-1)]^T$ ,  $\mathbf{h} = (\mathbf{w}^H \mathbf{A})^T$  is a  $K \times 1$  dimension vector. The weighted constraint method, in terms of a specific satellite, means that the group delay of the constraint  $\mathbf{h}$  is a fixed value. And the constraint can be achieved using the following orthogonal constraint condition

$$\mathbf{h}^T \partial \mathbf{c} = 0 \quad (2.213)$$

where

$$\partial \mathbf{c} = \begin{bmatrix} \partial R_s(\Delta T(D-1)) \\ \vdots \\ \partial R_s(\Delta T(D-K)) \end{bmatrix} / \partial \tau \quad (2.214)$$

$D\Delta T$  is the fixed time delay;  $R_s(\tau)$  is the auto-correlation of the GNSS signal;  $\partial(\bullet)$  represents the derivative operation.

This method, for different space-time adaptive jamming mitigation processing algorithms, needs to have additional orthogonal constraint conditions. For example, by adding a constraint condition for the Capon jamming mitigation algorithm  $\mathbf{w}^H \mathbf{A} \partial \mathbf{c} = 0$ , the solution of weights can be achieved using the formula below

$$\begin{aligned} \min \quad & \mathbf{w}^H \mathbf{R}_x \mathbf{w} \\ \text{s.t.} \quad & \begin{cases} \mathbf{A}^H \mathbf{w} = \boldsymbol{\beta}_{K \times 1} \\ \mathbf{w}^H \mathbf{A} \partial \mathbf{c} = 0 \end{cases} \end{aligned} \quad (2.215)$$

where  $\boldsymbol{\beta}_{K \times 1}$  is a constant vector. Let  $\mathbf{B} = [\mathbf{A}, \mathbf{A} \partial \mathbf{c}]$ ,  $\mathbf{C} = [\boldsymbol{\beta}^H, 0]$ . By writing (2.215) in the form of augmented matrix, weights can be recalculated as

$$\begin{aligned} \min \quad & \mathbf{w}^H \mathbf{R}_x \mathbf{w} \\ \text{s.t.} \quad & \mathbf{w}^H \mathbf{B} = \mathbf{C} \end{aligned} \quad (2.216)$$

Using the Lagrange method to solve weights, the new weights can be obtained as

$$\mathbf{w} = \mathbf{R}_x^{-1} \mathbf{B} (\mathbf{B}^H \mathbf{R}_x^{-1} \mathbf{B})^{-1} \mathbf{C}^H \quad (2.217)$$

## 2. Temporal domain filter equalization algorithm

Temporal domain filter equalization algorithm [61] makes different GNSS' STAP processing responses consistent by adding temporal domain filters. The goal of this method is to convolute one GNSS signal's beam output with other GNSS signals' STAP responses. Thereby, the minimum order of filters for this equalization algorithm is  $3K-2$ .

Assuming signals from four GNSS satellites incident onto the array, the output signals after array jamming mitigation for every satellite are  $y_1, y_2, y_3$  and  $y_4$  respectively, and the space-time responses on the four satellites are  $\mathbf{h}_1, \mathbf{h}_2, \mathbf{h}_3$  and  $\mathbf{h}_4$  respectively. To make the responses of STAP processes the same for all four GNSS signals, we perform the computation below, to obtain four new output signals  $y_{1\Delta}(n), y_{2\Delta}(n), y_{3\Delta}(n)$  and  $y_{4\Delta}(n)$ .

$$y_{1\Delta}(n) = y_1(n) * \mathbf{h}_2 * \mathbf{h}_3 * \mathbf{h}_4 = \{\mathbf{w}_1^H \mathbf{x}(n)\} * \mathbf{h}_2 * \mathbf{h}_3 * \mathbf{h}_4 \quad (2.218)$$

$$y_{2\Delta}(n) = y_2(n) * \mathbf{h}_1 * \mathbf{h}_3 * \mathbf{h}_4 = \{\mathbf{w}_2^H \mathbf{x}(n)\} * \mathbf{h}_1 * \mathbf{h}_3 * \mathbf{h}_4 \quad (2.219)$$

$$y_{3\Delta}(n) = y_3(n) * \mathbf{h}_1 * \mathbf{h}_2 * \mathbf{h}_4 = \{\mathbf{w}_3^H \mathbf{x}(n)\} * \mathbf{h}_1 * \mathbf{h}_2 * \mathbf{h}_4 \quad (2.220)$$

$$y_{4\Delta}(n) = y_4(n) * \mathbf{h}_1 * \mathbf{h}_2 * \mathbf{h}_3 = \{\mathbf{w}_4^H \mathbf{x}(n)\} * \mathbf{h}_1 * \mathbf{h}_2 * \mathbf{h}_3 \quad (2.221)$$

where  $*$  denotes the convolution operation.

After using this method of adding temporal domain filter, we can make the impacts of STAP processes the same for different GNSS signals. The four output signals can have correlation processing with different local spread spectrum codes, and the obtained cross-correlations are consistent. But this method can narrow the signal spread spectrum. For example, the frequency spectrum of the first GNSS becomes

$$Y_l(\omega) = H_1(\omega)S(\omega)H_2(\omega)H_3(\omega)H_4(\omega) \quad (2.222)$$

This also leads to the expansion of correlation function.

## 3. Least Squares Inverse Filter Equalization Algorithm

The equalization compensation method based on deterministic least squares inverse filter solves an inverse filter

$$\mathbf{g} = [g(0), g(1), \dots, g(K)]^T \quad (2.223)$$

which minimizes the error energy  $V(\mathbf{g}) = \|\delta - \mathbf{h} * \mathbf{g}\|$ ,  $\delta$  denotes an unit impulse function. To solve this inverse filter, we need to solve the derivative of  $V(\mathbf{g})$  related to  $\mathbf{g}$ , and make that derivative as zero. Based on this, we can derive

$$\sum_{k=0}^K g(k)r(k-l) = q(l), \quad l = 0, 1, \dots, K \quad (2.224)$$

where

$$\begin{aligned} r(k-l) &= \sum_{i=-\infty}^{+\infty} h(i-l)h^*(i-k) \\ &= \sum_{i=0}^{+\infty} h(i)h^*(i-(l-k)), \quad l = 0, 1, \dots, K \end{aligned} \quad (2.225)$$

$$\begin{aligned} q(l) &= \sum_{k=-\infty}^{+\infty} \delta(k)h(k-l) = h(-l), \quad l = 0, 1, \dots, K \\ &= \begin{cases} h(0) & l = 0 \\ 0 & l > 0 \end{cases} \end{aligned} \quad (2.226)$$

In matrix form, it is

$$\begin{bmatrix} r(0) & r(1) & \dots & r(K) \\ r(-1) & r(0) & \dots & r(K-1) \\ \vdots & \vdots & & \vdots \\ r(-K) & r(-(K-1)) & \dots & r(0) \end{bmatrix} \begin{bmatrix} g(0) \\ g(1) \\ \vdots \\ g(K) \end{bmatrix} = \begin{bmatrix} h(0) \\ 0 \\ \vdots \\ 0 \end{bmatrix} \quad (2.227)$$

Then it becomes convenient to solve for the inverse filter's coefficients  $\mathbf{g}$ .

#### 4. Equalization Algorithm Based on Homomorphic Filtering

Homomorphic filtering [63] can be used to separate or process non-additive combinational signals (e.g. product-type signal and convolution-type signal). The basic approach of the method is to perform Fourier transform or Z-transform and logarithmic transformation on mixed signals, so that the product-type signal and convolution-type signal can be converted to additive signals. After performing signal separation using linear systems, the signals can be converted back.

It can be known from (2.208) that the array output after the STAP jamming mitigation processing can be represented as

$$\begin{aligned}
y(n) &\approx \mathbf{w}^H \mathbf{A} \mathbf{s}(n) + \bar{e}(n) = \mathbf{h}^T \mathbf{s}(n) + \bar{e}(n) \\
&= [h(0), h(1), \dots, h(K-1)] \begin{bmatrix} s(n) \\ s(n-1) \\ \vdots \\ s(n-(K-1)) \end{bmatrix} + \bar{e}(n) \\
&= \sum_{k=0}^{K-1} h(k)s(n-k) + \bar{e}(n)
\end{aligned} \tag{2.228}$$

It can be seen that unlike the expected receiving GNSS signal  $s(n)$ , array output is equivalent to the process of convolving the array response and the signal, i.e.

$$y(n) = h * s(n) + \bar{e}(n) \tag{2.229}$$

It can be seen in (2.229) that the impact on the signal by the STAP process can be seen as a convolution distortion. Therefore, the correlation function between the derived array output signal and the local spread spectrum code has distortion, leading to errors in positioning results.

Below this method of homomorphic filtering can be used to isolate the GNSS signals. First, Fourier transformations are performed on both sides of (2.229) to convert the convolution distortion to a product-type distortion.

$$\begin{aligned}
Y(\omega) &= H(\omega)S(\omega) + \bar{E}(\omega) \\
&= H(\omega)[S(\omega) + E_1(\omega)]
\end{aligned} \tag{2.230}$$

where  $\bar{E}(\omega) = H(\omega)E_1(\omega)$ . Then let  $S_1(\omega) = S(\omega) + E_1(\omega)$ , then (2.230) can be written as

$$Y(\omega) = H(\omega)S_1(\omega) \tag{2.231}$$

After taking the logarithm, the product-type distortion can be converted to an additive distortion, and we have

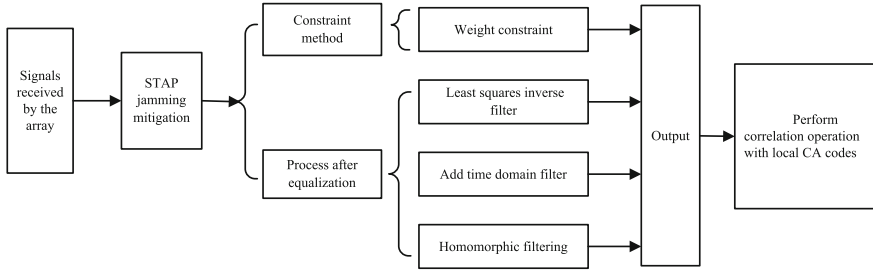
$$\ln(Y(\omega)) = \ln(H(\omega)S_1(\omega)) = \ln(H(\omega)) + \ln(S_1(\omega)) \tag{2.232}$$

We then subtract the additive distortion after the transformation

$$\ln(S_1(\omega)) = \ln(Y(\omega)) - \ln(H(\omega)) \tag{2.233}$$

By taking exponential operations on both sides of (2.23), we can obtain

$$S_1(\omega) = \exp[\ln(Y(\omega)) - \ln(H(\omega))] \tag{2.234}$$



**Fig. 2.41** STAP jamming mitigation and equalization compensation process

It can be known from (2.230) that  $S_1(\omega)$  is the GNSS signal frequency response under the influence of noise. By performing inverse Fourier transformation on (2.234), the obtained signal is  $s_1(n) = s(n) + e_1(n)$ , we then can recover the desired GNSS signal, but the obtained signal still contains the noise component  $e_1(n)$ .

The four methods described above compensate for STAP process, and all need to use the GNSS signal steering vector. Among them, the weight constraint method needs to re-add orthogonal constraint conditions for different jamming mitigation algorithms to solve for the new weights, but the added computation is mainly due to matrix multiplication, so it adds relatively small loads. Every satellite beam output for the temporal domain filter equalization algorithm needs to convolute other GNSS signals' response coefficients with different time delays, to let the correlation function have certain degrees of expansion and time delay, so that the added computation is for multiple convolution operations. When it is used to solve the inverse filter coefficients for every GNSS signal, the least squares inverse filter algorithm needs to have the covariance matrix of a GNSS signal's STAP response, so it increases computational load. Equalization algorithm based on homomorphic filtering uses the homomorphic filtering to separate out the GNSS signal, so that it is compatible with algorithms for estimating GNSS signal's steering vector such as the space-time de-spread re-spread method. The additional amount of computation is mainly for fast Fourier transformation, logarithmic operation, and exponential operation, so the amount of computation is still relatively small. Figure 2.41 shows the STAP jamming mitigation and equalization compensation process.

### 2.6.6 Simulation Results

Below we perform simulations to compare the performances of various algorithms described in this section. For simulations, we use an equidistant linear array with the number of array elements being  $M = 4$ , and the interval between array elements as half a wavelength. The signals received by the array can be down-converted to the intermediate frequency 4.309 MHz, and the sampling rate is 5.714 MHz. After

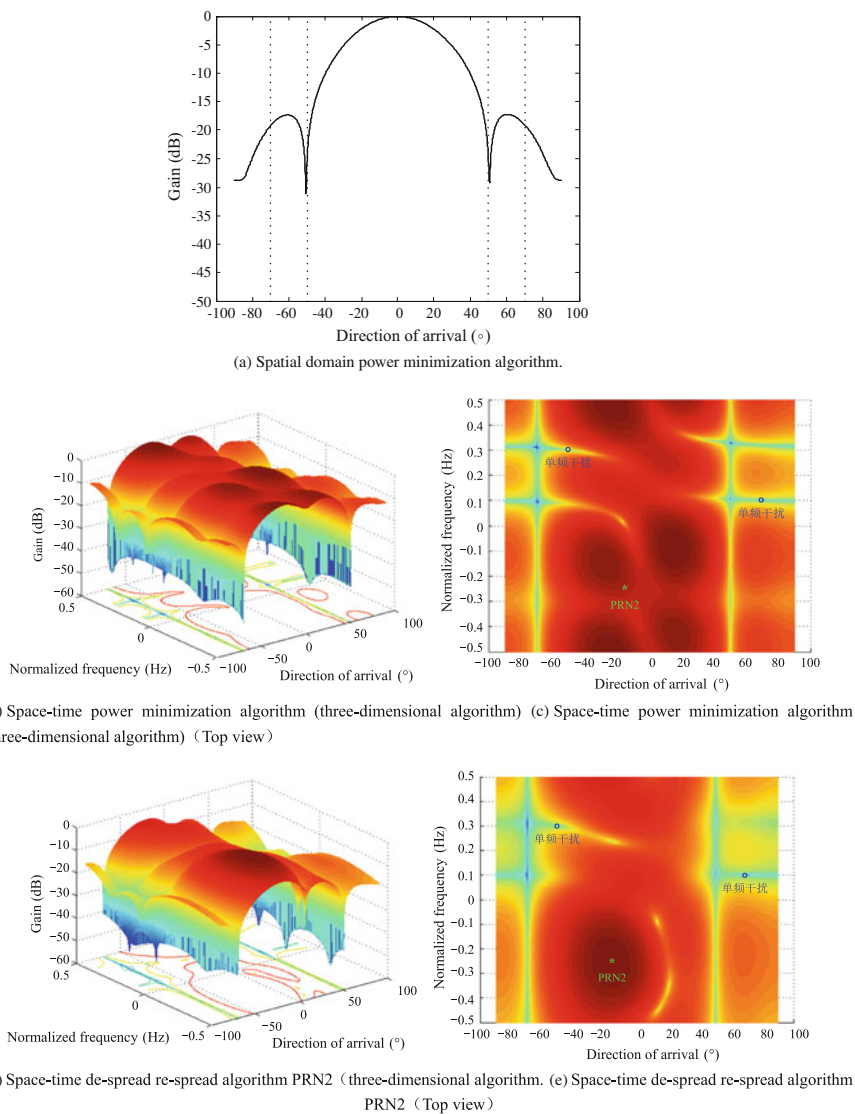


every array element, three time taps are connected to form a space-time processing structure, and every time tap has a one-sample point delay. We set the following simulation environment: five satellites PRN2, PRN6, PRN14, PRN20 and PRN25, and their signals incident onto the antenna array with angles of  $-15^\circ$ ,  $20^\circ$ ,  $-5^\circ$ ,  $5^\circ$ , and  $15^\circ$ . The signal-to-noise ratio is  $-20$  dB. Two single-frequency jamming signals with frequencies of 2.5948 MHz and 3.7376 MHz, incident onto the antenna array with angles of  $-50^\circ$  and  $70^\circ$ , and the jamming-to-noise ratio is 40 dB. Two wideband jamming signals incident on the antenna array with angles of  $-70^\circ$  and  $50^\circ$ , and the jamming-to-noise ratio is 40 dB.

### 1. Simulation Result Comparisons Between Spatial Domain Processing and Space-Time Processing

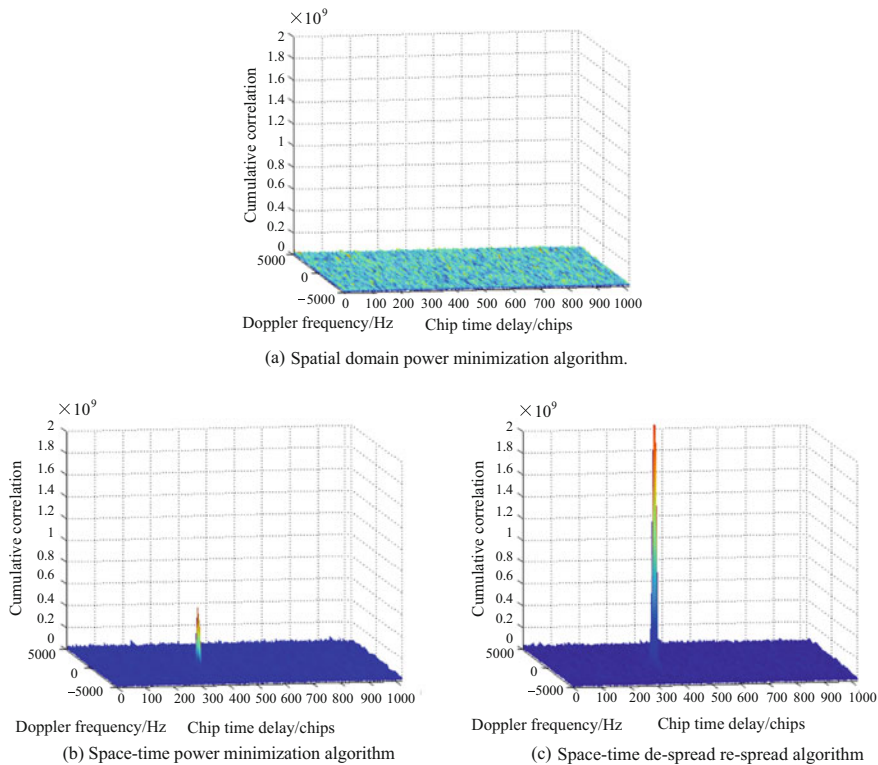
When both wideband jamming and single frequency jamming exist, the simulation results of spatial domain jamming mitigation and space-time jamming mitigation are compared in Fig. 2.42. Spatial domain processing uses the spatial domain power minimization algorithm, and space-time processing can use the space-time power minimization algorithm and space-time de-spread and re-spread algorithm described in this chapter. Figure 2.42a shows the antenna pattern of the spatial domain processing, where the dotted position represents the direction of the jamming signal. It can be seen in the figure that not enough nulls are formed towards the jamming directions for the spatial domain processing due to the constraints on the degrees of freedom. Figure 2.42b–e show the space-time domain processing's space-time two-dimension response graphs and the corresponding top-view graphs. In these graphs, we mark the GNSS signal (PRN2) and single-frequency jamming's space-time two-dimension coordinates on the top view graph. The nulls of wideband jamming aim towards the directions of  $-70^\circ$  and  $50^\circ$ . It can be seen from the subfigures that the space-time power minimization algorithm and the space-time de-spread re-spread algorithm can form nulls towards the jamming directions since they have more degrees of freedom. The space-time de-spread re-spread algorithm can use more a priori information, so that the obtained space-time two-dimension response graph has its mainlobe aiming towards the GNSS signal's space-time two-dimension coordinates, to achieve array signal processing gain.

Figure 2.43 further explains the performances of spatial domain jamming mitigation and space-time jamming mitigation processes by acquiring PRN2. In the figure, the  $x$ -axis denotes the chip delay, the  $y$ -axis denotes the Doppler frequency, and the  $z$ -axis denotes the accumulated correlation. Figure 2.43a is the acquisition result after spatial domain processing. It can be seen that since the spatial domain processing can not effectively eliminate jamming impacts, the correlation output does not have obvious peaks, so that it can not acquire the GNSS. Figure 2.43b, c show acquisition results using the space-time power minimization algorithm and the space-time de-spread re-spread algorithm. It can be seen that both subfigures have obvious peaks, so they show that the GNSS signal PRN2 can be successfully



**Fig. 2.42** Comparisons of adaptive patterns

acquired. The jamming mitigation performance of space-time processing is better than that of spatial domain processing. By comparing Fig. 2.43b, c further, we find that the space-time de-spread re-spread algorithm can provide signal processing gains, so that its correlation output peaks of acquiring GNSS are greater. This is beneficial for the follow-up GNSS signal tracking.



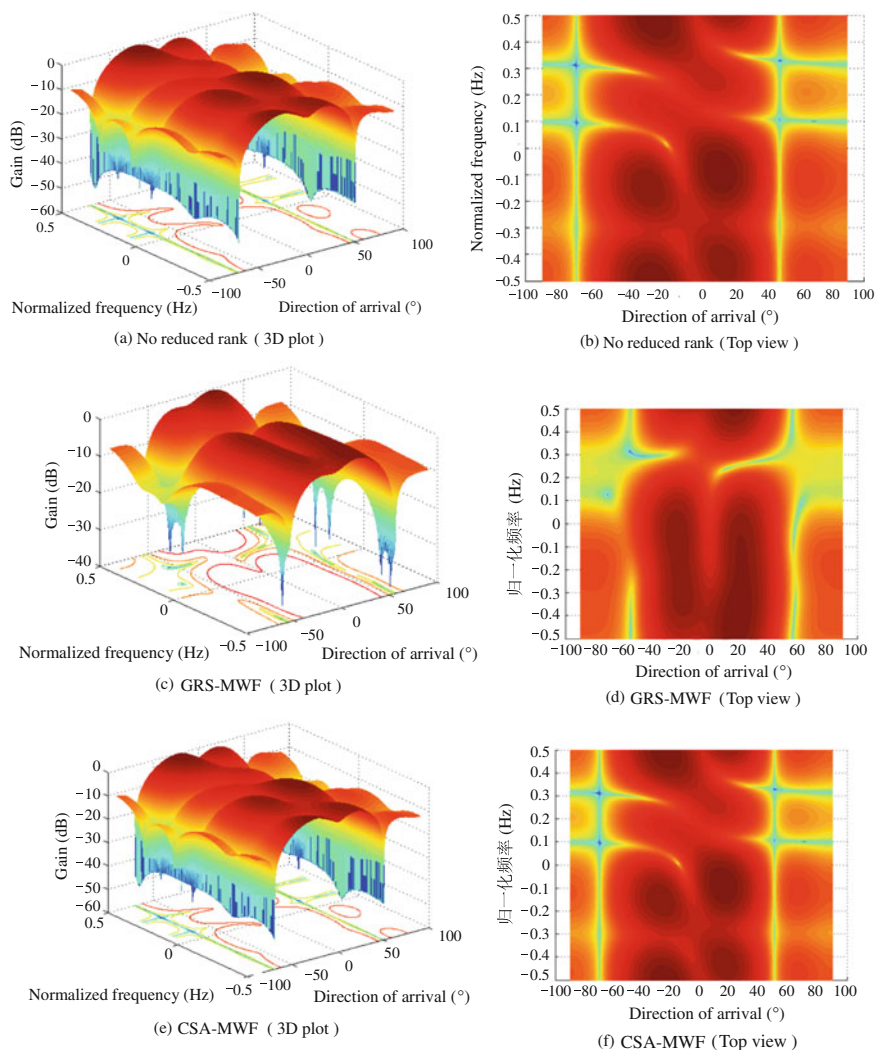
**Fig. 2.43** Comparisons on PRN2's acquisition results after jamming mitigation processing

## 2. Simulation Result Comparisons of Several Reduced Rank STAP Algorithms

Below we discuss and compare the rank reduction performances of applying several rank reduction algorithms on the space-time power minimization algorithm. Figure 2.44 shows the simulation results. For every set of subfigures, the left side is the space-time two-dimension response, where the  $x$ -axis denotes the direction of arrival; the  $y$ -axis denotes the normalization frequency; and the  $z$ -axis denotes the space-time processing gain; the right side shows the corresponding top view. To make it convenient to compare, we redraw Fig. 2.42b, which is the result before rank reduction, as Fig. 2.44a, b. Due to simulation conditions, the dimension of jamming subspace is 10, so the number of dimensions for several rank reduction algorithms in the simulations are all set as  $D = 10$ . From the space-time two-dimension responses and the corresponding top views, it can be seen that various rank reduction methods can all form deep nulls towards the jamming.

## 3. Simulation Result Comparisons of Several STAP Equalization Algorithms

Figures 2.45 and 2.46 list comparative simulation results for several different STAP equalization algorithms. In Fig. 2.45, the solid line denotes the normalized



**Fig. 2.44** Comparisons of STAP algorithm adaptive response diagrams for different reduced ranks

cross correlation between the array output signal without equalization processing and the local PRN codes; the “○” line denotes the normalized cross correlation results after the STAP jamming mitigation and the additional homomorphic filtering processing; the “\*” line corresponds to the least squares inverse filtering method; and the “◇” line corresponds to the weight constraint method. It can be seen that after the STAP jamming mitigation the peak values of correlation functions have some offsets and broadening, and this results in the measured PRN code’s initial

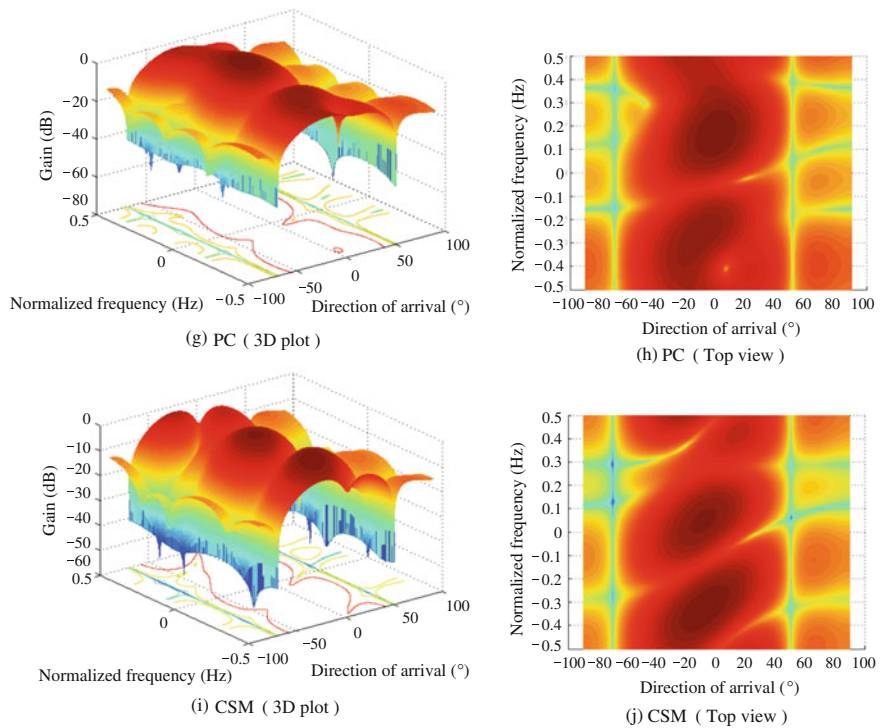
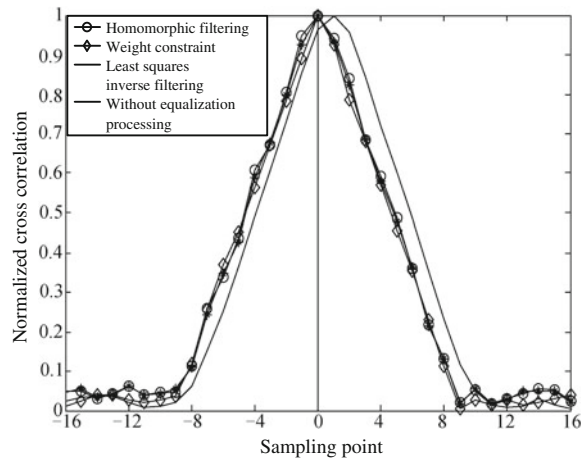
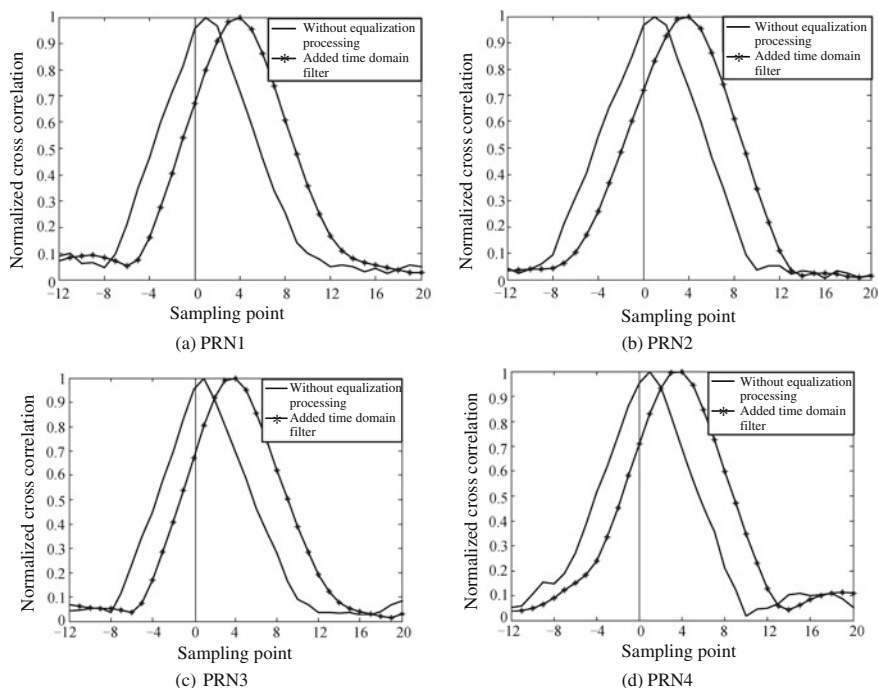


Fig. 2.44 (continued)

Fig. 2.45 Comparison of normalized cross-correlations





**Fig. 2.46** Different satellites' cross correlation graphs with added temporal domain filtering

position errors, thereby acquisition and tracking are impacted. Several equalization methods can all correct this type of distortion.

Since the added temporal domain filter method is different from other methods, its results are listed in Fig. 2.46. When 4 GNSS signals incident on the array, it can be seen from Fig. 2.46 that, for every satellite, when there is no added temporal domain filter, peak offsets of cross correlation are different. When the temporal domain filter is added, even though the correlation function has certain broadening, the delays towards every GNSS signals are consistent, i.e. the offset amount is the same, so it achieves the objective of equalization.

## 2.7 Summary

In this chapter, we discuss jamming suppression techniques from the perspectives of temporal domain, spatial domain, and space-time domain. Temporal domain processing is low cost and easy to implement, so it is suitable for suppressing single frequency jamming with a low number of jamming sources. Spatial domain processing is a fairly mature jamming suppression technique, so it is suitable in environments of multiple narrowband jamming sources, and it has many

corresponding adaptive filtering algorithms. In this chapter, we also analyze in detail various spatial domain algorithms' features and applicabilities. We proposed a high-gain and high-robustness jamming mitigation algorithm for hardware implementation. The space-time process represents a developing trend of jamming mitigation, and can be used under various complex jamming environments. But the amount of computation needed is large, and it is easier to trigger signal distortion. Therefore we need to consider rank reduction processing and signal distortion compensation techniques.

## References

1. Upton DM, Upadhyay TN, Marchese J. Commercial-off-the-shelf (COTS) GPS interference canceller and test results. In: ION Navigation conference. 1998. p. 319–25.
2. Dimos G, Upadhyay T, Jenkins T. Low-cost solution to narrowband GPS interference problem. *Process IEEE Natl Conf Aerosp Electron*. 1995;1:45–153.
3. Rifkin R, Vaccaro JJ. Comparisons of narrowband adaptive filter technologies for GPS. MITRE technique report. 2000. p. 125–31.
4. Milstein LB. Interference rejection technique in spread spectrum communications. *Process IEEE*. 1988;76(6):657–71.
5. Ma W, Mao W, Chang F. Design of adaptive all-pass based notch filter for narrowband anti-jamming GPS system. In: *Proceedings of 2005 international symposium on intelligent signal processing and communication system*, Hong Kong. 2003. p. 305–8.
6. Soderstrand MA, Johnson TG, Strandberg RH, et al. Suppression of multiple narrow-band interference using real-time adaptive notch filters. *IEEE Trans Circuits and Syst II: Analog Digit Signal Process*. 1997;44(3):217–25.
7. Landry RJ, Calmettes V, Bousquet M. Impact of interference on a generic GPS receiver and assessment of mitigation techniques. In: *IEEE 5th international symposium on spread spectrum techniques and applications*. 1998. vol 1. p. 87–91.
8. Zoltowski MD, Haber F. Advanced adaptive null steering concepts for GPS. In: *Processing MILCOM conference, universal communications*, San Diego, CA. 1995. vol 3. p. 1214–8.
9. Gecan A, Zoltowski MD. Power minimization technique for GPS null steering antennas. In: *Institute of Navigation conference*, Palm Springs, CA. 1995. vol 12, no 1, p. 13–5.
10. Wang WY, Du QR, Wu RB et al. Interference suppression with flat gain constraint for satellite navigation systems. *IET Radar Sonar Navig*. 2015;9(7):852–6.
11. Moelker DJ, Pol EV, Yeheskel BN. Adaptive antenna arrays for interference cancellation in GPS and GLONASS receivers. In: *Proceeding of the IEEE position location and navigation symposium*. 1996. p. 191–8.
12. Jay RS. Interference mitigation approaches for the global positioning system. *Lincoln Lab J*. 2003;14(2):168–80.
13. Kaplan ED, Hegarty CJ. *Understanding GPS principles and applications*, 2nd ed. Artech House; 2006.
14. Sun W, Amin MG. A self-coherence anti-jamming GPS receiver. *IEEE Trans Signal Process*. 2005;53(10):3910–5.
15. Li P, Lu D, Wu R, et al. Adaptive anti-jamming algorithm based on the characteristics of the GPS signal. In: *Proceedings of 2008 international symposium on intelligent signal processing and communication systems*. 2008. p. 181–4.
16. Fante RL, Vaccaro JJ. Wideband cancellation of interference in a GPS receiver array. *IEEE Trans Aerosp Electron Syst*. 2000;36(2):549–64.

17. Myrick WL, Goldstein JS, Zoltowski MD. Anti-jam space-time preprocessor for GPS based on multistage nested wiener filter. *IEEE Mil Commun (Atlantic NJ)*. 1999;1:675–81.
18. Lu D, Wu R. Global positioning system anti-jamming algorithm based on period repetitive CLEAN. *IET Radar Sonar Navig*. 2013;7(2):164–9.
19. Poor HV, Rusch LA. Narrowband interference suppression in spread spectrum CDMA. *IEEE Pers Commun Mag*. 1994;3:14–27.
20. Haykin S. *Adaptive filter theory*. 4th ed. New Jersey: Prentice Hall; 2002.
21. Young JA, Lehnert J, Lehnert S. Analysis of DFT-based frequency excision algorithm for direct sequence spread spectrum communications. *IEEE Trans Commun*. 1998;46:1076–87.
22. Capozza PT, Holland BJ, Hopkinson TM, et al. A single-chip narrow-band frequency-domain excisor for a global positioning system (GPS) receiver. *IEEE J Solid-State Circuits*. 2000;35(3):51–5.
23. Hlawatsch F, Boudreaux-Bartels GF. Linear and quadratic time-frequency signal representations. *IEEE Signal Process*. 1992;9(2):21–67.
24. Badke B, Spanias AS. Partial band interference excision for GPS using frequency-domain exponents. In: *Proceedings of the 2002 IEEE international conference on acoustic, speech, and signal processing*. 2002. vol 4. p. 3936–9.
25. Ouyang X, Amin MG. Short-time fourier transform receiver for nonstationary interference excision in direct sequence spread spectrum communications. *IEEE Trans Signal Process*. 1988;184–213.
26. Zhao L, Amin MG, Lindsey AR. Subspace projection techniques for anti-FM jamming GPS receivers. *IEEE SSAP*; 2000. p. 529–33.
27. Zhang Y, Amin MG, Lindsey AR. Anti-jamming GPS receivers based on bilinear signal distributions. In: *Proceedings of the MILCOM conference, universal Communications*. 2001. vol 2. p. 1070–4.
28. Carlson BD. Covariance matrix estimation errors and diagonal loading in adaptive arrays. *IEEE Trans Aerosp Electron Syst*. 1988;24(4):397–401.
29. Misra, Pratap. *Global positioning system: signals, measurements, and performance*. Ganga-Jamuna Press; 2006.
30. Agee BG, Schell SV, Gardner WA. Spectral self-coherence restoral: a new approach to blind adaptive signal extraction using antenna arrays. *Proc IEEE*. 1990;78(4):753–67.
31. Amin MG, Sun W. A novel interference suppression scheme for global navigation satellite systems using antenna array. *IEEE J Sel Areas Commun*. 2005;23(5):999–1012.
32. Gershman AB, Nickel U, Bohme JF. Adaptive beamforming algorithms with robustness against jammer motion. *IEEE Trans Signal Process*. 1997;45(7):1878–85.
33. Shmidt RO. Multiple emitter location and signal parameter estimation. *IEEE Trans Antennas Propag*. 1986;34(3):276–80.
34. Roy R, Paulraj A, Kailath T. ESPRIT—a subspace rotation approach to estimation of parameter of cissoids in noise. *IEEE Trans Acoust Speech Signal Process*. 1986;34:1340–2.
35. Pickholtz RL, Schilling DL, Milstein LB. Theory of spread-spectrum communications: a tutorial. *IEEE Trans Commun*. 1982;30(5):855–84.
36. Hogbom JA. Aperture synthesis with a non-regular distribution of interferometer baselines. *Astron Astrophys Suppl Ser*. 1974;15:417–26.
37. Gough PT. A fast spectral estimation algorithm based on the FFT. *IEEE Trans Signal Process*. 1994;42(6):1317–22.
38. Tsao J, Steinberg BD. Reduction of sidelobe and speckle artifacts in microwave imaging: the CLEAN technique. *IEEE Trans Antennas Propag*. 1988;36(4):543–56.
39. Li J, Stoica P. Efficient mixed-spectrum estimation with applications to target feature extraction. *IEEE Trans Signal Process*. 1996;44(2):281–95.
40. Li J, Zheng D, Stoica P. Angle and waveform estimation via RELAX. *IEEE Trans Aerosp Electron Syst*. 1997;33(3):1077–87.
41. Liu Z, Li J. Implementation of the RELAX algorithm. *IEEE Trans Aerosp Electron Syst*. 1998;34(2):657–64.



42. Klemm R. Adaptive airborne MTI: an auxiliary channel approach. *IEE Proc Commun Radar Signal Process.* 1987;134(3):269–76.
43. Rong Z. Simulations of adaptive array algorithm for CDMA system. M.S. thesis. Blacksburg: Virginia Tech; 1996.
44. Rong Z, Rappaport TS. Simulation of multitarget adaptive algorithms for wireless CDMA systems. In: *Proceeding of IEEE vehicular technology conference.* 1997. p. 1–5.
45. Thanh VD, Hung NL. New adaptive beamforming algorithms for smart antennas in DS-CDMA mobile communication systems. In: *2004 3th international conference on computational electromagnetics and its applications proceedings.* 2004. p. 165–8.
46. Du Z, Gong P, Wu W. Block based RLS de-spread re-spread multitarget array: algorithm and performance. In: *Proceeding of IEEE vehicular technology conference.* 2001. p. 190–3.
47. Wu R, Bao Z. Array pattern distortion and remedies in space-time adaptive processing for airborne radar. *IEEE Trans Antennas Propag.* 1998;46(7):963–70.
48. Brennan LE, Mallett JD, Reed IS. Adaptive arrays in airborne MTI radar. *IEEE Trans Antennas Propag.* 1976;24(5):607–15.
49. Reed IS, Mallett JD, Brennan LE. Rapid convergences rate in adaptive arrays. *IEEE Trans Aerosp Electron Syst.* 1974;10(6):853–63.
50. Wang H, Cai LJ. On adaptive spatial-temporal processing for airborne surveillance radar systems. *IEEE Trans Aerosp Electron Syst.* 1994;30(3):660–9.
51. Wang H. An overview of space-time adaptive processing for airborne radars. In: *CIE international conference of radar.* 1996. p. 789–94.
52. Fante RL, Vaccaro JJ. Cancellation of jammers and jammer multipath in a GPS receiver. *IEEE Aerosp Electron Syst Mag.* 1998;13(11):25–8.
53. Fante RL, Vaccaro JJ. Evaluation of adaptive space-time-polarization cancellation of broadband interference. In: *IEEE position location and navigation symposium.* 2002. p. 1–3.
54. David S. Navigation accuracy and interference reject for GPS adaptive antenna arrays. Doctor degree thesis. Stanford University; 2007.
55. Fante RL, Torres JA. Cancellation of diffuse jammer multipath by airborne adaptive radar. *IEEE Trans Aerosp Electron Syst.* 1995;31(2):805–20.
56. Tufts DW, Kumaresan R, Kirshteins I. Data adaptive signal estimation by signal value decomposition of a data matrix. *Proc IEEE.* 1982;70(6):684–5.
57. Goldstein JS, Reed IS. Reduced-rank adaptive filtering. *IEEE Trans Signal Process.* 1997;45(2):492–6.
58. Goldstein JS, Reed IS, Nehorai A. A multistage representation of the wiener filter based on orthogonal projections. *IEEE Trans Inf Theory.* 1998;44(7):2943–59.
59. Honig ML, Xiao WM. Performance of reduced-rank linear interference suppression. *IEEE Trans Inf Theory.* 2001;47(5):1928–46.
60. Ricks DC, Goldstein JS. Efficient architectures for implementing adaptive algorithms. In: *Proceedings of the 2000 antenna applications symposium.* USA: Monticello. p. 29–41.
61. Hatke GF. Adaptive array processing for wideband nulling in GPS systems. *Signals, systems & computers.* In: *Conference record of the thirty-second Asilomar conference.* 1998. p. 1332–6.
62. Wu R, Xu R, Lu D, et al. STAP equalization technique based on homomorphic filtering in GPS. In: *2010 IEEE international symposium on phased array systems and technology.* 2010. p. 841–5.
63. Vaseghi SV. *Advanced digital signal processing and noise reduction.* 3rd ed. USA: Wiley; 2006. p. 27–269.

Adaptive Interference Mitigation in GNSS

Wu, R.; Wang, W.; Lu, D.; Wang, L.; Jia, Q.

2018, XIX, 274 p. 136 illus., Hardcover

ISBN: 978-981-10-5570-6