Partially Constrained Adaptive Beamforming

By

Erik Hornberger

Submitted to the Department of Electrical Engineering and Computer Science and the
Graduate Faculty of the University of Kansas
in partial fulfillment of the requirements for the degree of
Master of Electrical Engineering

Shannon Blunt, Chairperson

Committee members

James Stiles

Erik Perrins

Date defended: July 15, 2015

The Thesis Committee for Erik Hornberger certifies
that this is the approved version of the following thesis :

Partially Constrained Adaptive Beamforming

Shannon Blunt, Chairperson

Date approved: July 15, 2015

Abstract

The ReIterative Super-Resolution (RISR) was developed based on an iterative implementation of the Minimum Mean Squared Error (MMSE) estimator. Here, a novel approach to direction of arrival estimation, partially constrained beamforming is introduced by building from existing work on the RISR algorithm. First, RISR is rederived with the addition of a unity gain constraint, with the result denoted as Gain Constrained RISR (GC-RISR), though this formulation exhibits some loss in resolution. However, by taking advantage of the similar structure of RISR and GC-RISR, they can be combined using a geometric weighting term α to form a partially constrained version of RISR, which we denote as PC-RISR.

Simulations are used to characterize PC-RISR's performance, where it is shown that the geometric weighting term can be used to control the speed of convergence. It is also demonstrated that this weighting term enables increased super-resolution capability compared to RISR, improves robustness to low sample support for super-resolving signals with low SNR, and the ability to detect signals with an SNR as low as -10dB given higher sample support.

Acknowledgements

I would like to thank the Madison & Lila Self Graduate Fellowship for providing the funding that allowed me to dedicate my time to research, and for providing invaluable professional training that has enriched my time at the University of Kansas.

I would also like to thank my advisor Dr. Shannon Blunt for giving me the freedom to pursue whatever topics piqued my interest, and for being a mentor to me in both in and out of the lab.

Contents

List of Figures

List of Tables

Chapter 1

Direction of Arrival (DOA) Estimation

This chapter includes a brief introduction to the field of Direction of Arrival Estimation and establishes several figures of merit that will be used throughout to compare the performance of various algorithms.

1.1 Overview of DOA

Direction Of Arrival (DOA) estimation refers to the problem of discerning the direction from which some field or wave impinges on an array of sensors. It has a wide variety of applications in everything from seismology to speech processing. In medical imaging technologies such as magnetoencephalography (MEG) and electroencephalography (EEG), sensors placed around the skull capture electric and magnetic waves emitted from the brain and work backwards to an image. In radar applications, large arrays are used to locate and differentiate objects, such as aircraft or weather systems. DOA estimation can also be applied to matter waves, such as sound waves, using an array of microphones. For example, audio engineers can employ DOA estimation as a means to achieve proper equalization or isolate echoes. In fact, there are as many potential applications of DOA estimation as there are varieties of sensors and waves, and in some cases DOA estimation can actually be equivalent to frequency analysis, which opens up another whole domain of applications [14, pp.6-12] [19].

1.2 Figures of Merit

When discussing DOA estimation there are number of metrics used to compare the performance of different techniques and an assortment of conditions that may impose limitations on some estimation methods. Perhaps the most common figure of merit is the *resolution* a given algorithm provides paired with some particular array. Resolution, in the most general sense, describes the minimum angular separation between two signals such that they can still be separated as distinct signals. Unfortunately, this is a rather ambiguous measure because there is no generally acceptable rule for determining if two signals are separate. Consider Figure 1.1, in which two spatial power estimates are shown. In the right plot, it is difficult to say for certain if there is one signal or two because there is a null, but it is not very deep. In the left, the null is sufficiently deep, but it is difficult to determine if the left lobe is the mainlobe of a weaker signal, or just a sidelobe of the signal on the right.

A common convention to facilitate comparison of different beamforming methods is to decide that closely spaced signals are separable if there is a 3 dB null between them. This convention has its limitations, particularly when peaks are sharper than those shown in the previous example, but while imperfect it is a generally accepted basis by which to evaluate the performance of DOA estimation algorithms.

Another commonly used figure of merit is *dynamic range*. Signals are most easily separated when they have the same strength, though, this is often not the case. What if one signal is much smaller than the other? Will it still be possible to discern the smaller signal, or will it get buried in the sidelobes of the larger (see Figure 1.1)? Dynamic range measures the maximum signal power difference at which signals can successfully separated.

Stationarity, *sample support* and *convergence speed* are all closely related as well. If the received signal does not exhibit adequate stationarity it can be difficult to obtain enough samples to realize a good estimate, thus suggesting the need for operation at low sample support. Of course, most DOA estimation methods require a large sample support, with their performance often evaluated in the asymptotic sample support regime [14]. The vast majority of DOA estimation

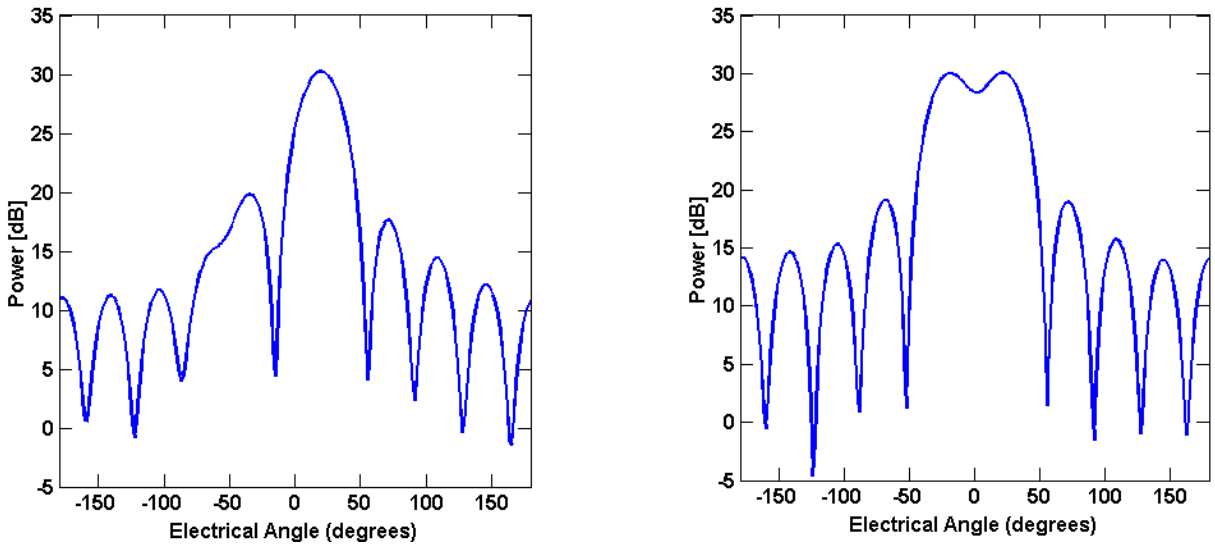

Figure 1.1: Two spatial power estimates that demonstrate the ambiguity involved with deciding if signals are separable. Insufficient null depth (right) and difficulty distinguishing main lobes from sidelobes (left) are two common problems.

techniques require the estimation of sample covariance matrix (SCM)

Finally, the *array manifolds* must be taken into account. The array manifold (explained in detail in Chapter 2) is a function of the array's geometry that characterizes its response to a signal from any direction. Virtually all DOA estimation techniques can be used with uniform linear arrays (ULA), so ULA's are often used to evaluate the performance of beamformers because they facilitate comparison. However, there are many applications, such as in the case of conformal arrays, when algorithms that can be used with arbitrary or unknown array manifolds have a distinct advantage.

Chapter 2

Phased Arrays and Beamforming

2.1 Introduction to Phased Arrays

DOA estimation necessarily involves being able to point the antenna or array in an arbitrary direction. Naturally, it is possible to perform DOA estimation by physically rotating the whole array over a range of angles while recording the received power — that is the approach used with parabolic dishes. However, using a phased array has notable advantages, and is becoming a more and more common solution. Unlike a parabolic dish, which is typically just one antenna, an array is composed of multiple elements, sometimes as many as thousands. If the received signal changes slowly with respect for the amount of time it takes the signal to travel across the array, then because there is some separation between elements there will also be some time difference between elements. Given some basic information about the impinging wave and the geometry of the antennas' configuration, it is possible to determine the direction from which the wavefront originated. Figure 2.1 illustrates how the wavefront of a signal reaches the elements of a phased array at different times. Considering Figure 2.1, one can imagine how changing the direction of arrival would likewise change the time it takes the wave the travel between elements.

Figure 2.1 depicts a *uniform linear array* (ULA). In an ideal ULA, all of the antenna elements are identical, equally spaced, and isotropic, meaning the gain in any direction around an individual

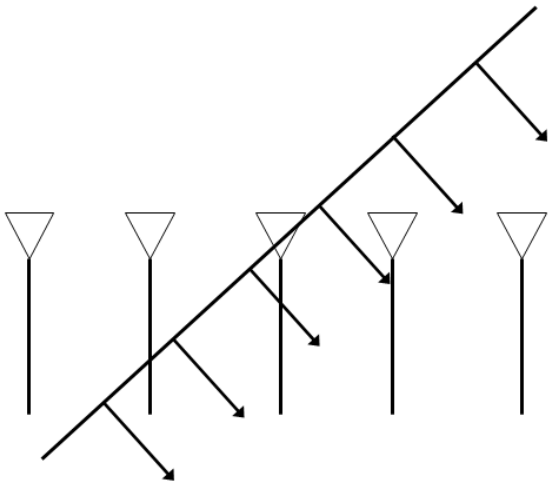

Figure 2.1: A wave impinging on a ULA. The time it takes the wave to travel between elements is a function the angle it originates from.

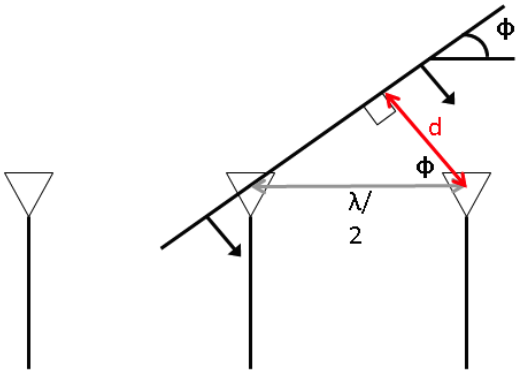

Figure 2.2: Simple trigonometry can be used to find the distance the wave travels between elements.

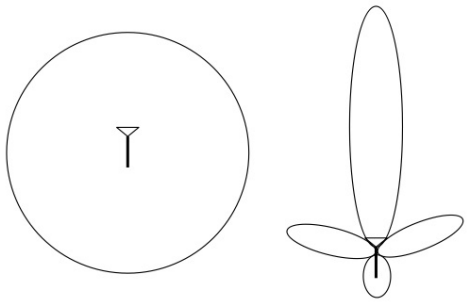

Figure 2.3: An isotropic antenna (left) and a non-isotropic antenna (right).

element is the same [14, p.37]. Furthermore, the spacing between each element is optimally half of a wavelength to maximize resolution without incurring grating lobes. Real, non-ideal ULA's have similar, nearly equally spaced, nearly isotropic elements. Fig 2.3 depicts the beampattern of an isotropic antenna element.

There are two more important assumptions that are often implicit with ULAs. The far field assumption, which states that incident fields originated sufficiently far away that they can be approximated by plane waves, and the narrow band assumption. Simply stated, invoking the narrow band assumption means assuming that the signal changes slowly with respect to the length of time it takes the signal to travel across the array. The narrowband assumption allows us to treat the time difference between elements as a phase difference, and greatly simplifies the problem of DOA estimation. If the narrowband assumption is not met, the array geometry can still be used to back out the direction of arrival, but it becomes considerably more complicated. Stated mathematically, the narrowband assumption can be expressed as

$$BT \ll 1, \tag{2.1}$$

where B is the incident signal's bandwidth and T is the time it takes the signal to travel across the array [14, p. 34].

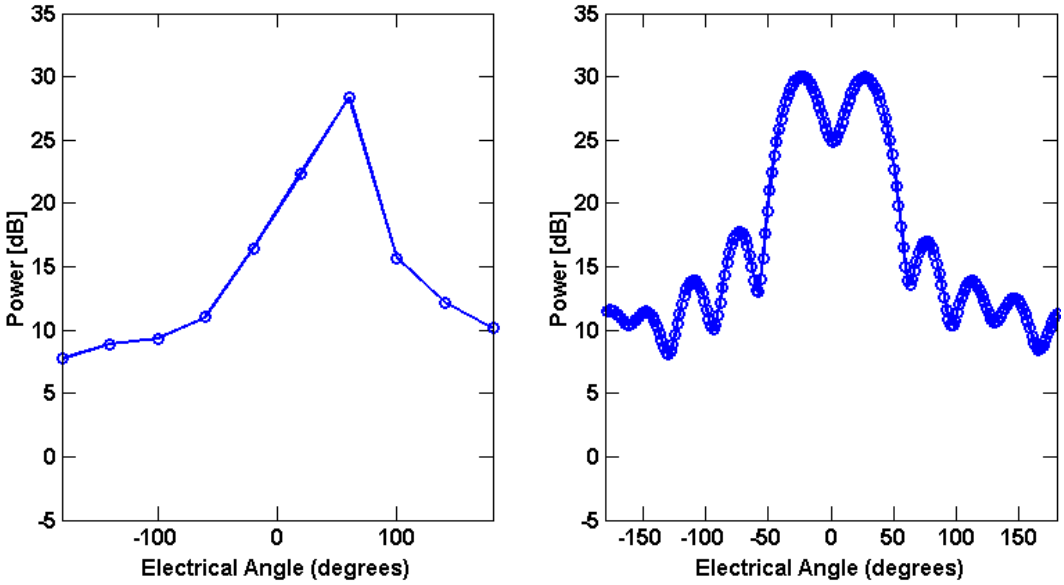

Figure 2.4: The same spatial power spectrum estimated with an $N \times N$ steering vector matrix (left) and an $N \times 20N$ steering vector matrix (right). Sufficient sampling of the array manifold is necessary to see fine features.

2.2 Array Manifolds and Steering Vectors

The *array manifold* is the response across the array from the continuum of directions from which an incident signal could arrive. It can be approximated using steering vectors, each of which is a column vector corresponding to one specific incident angle. A steering vector is associated with a specific angle and contains one entry for each antenna describing the phase difference between elements for a signal arriving from that angle. By collecting steering vectors for many incident angles together to form a *steering vector matrix*, it is possible to created a spatially sampled version of the array manifold which can be used to estimate the DOAs. Steering vector matrices need to be sufficiently sampled in spatial angle (with respect to the spatial resolution dictated by the number of antenna elements) so that there are enough points to distinguish between adjacent peaks. Figure 2.4 shows why sampling is necessary for accurate DOA estimation.

Consider the simplest case, in which a wave impinges on a ULA from the boresite direction, the angle perpendicular to the array, here denoted as 0 degrees. In this case, (invoking the far field assumption) the phase will be the same across all array elements, yielding the steering vector

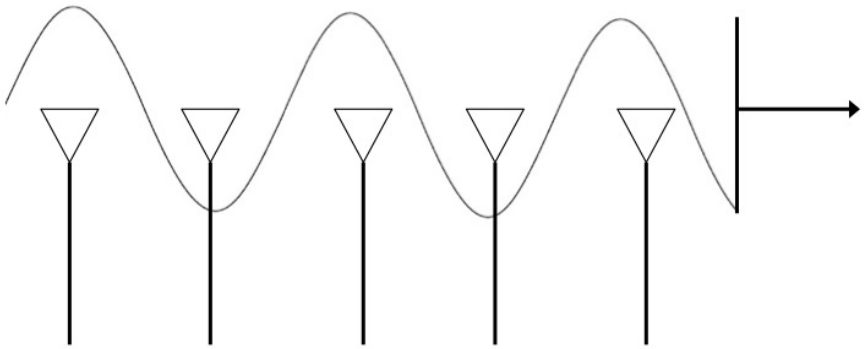

Figure 2.5: A wavefront impinging across a phased array. Note the phase difference at each element.

$$\mathbf{s}(0^\circ) = \begin{bmatrix} 1 & 1 & \cdots & 1 \end{bmatrix}^\mathsf{T}, \tag{2.2}$$

where $^\mathsf{T}$ is the transpose operator. Next, consider the endfire scenario, in which the wave comes from the side, as shown in Figure 2.5. Because the elements are spaced a half wavelength apart, the phase changes by 180° between each element. Accordingly, if there are N antenna elements, the steering vector would be

$$\mathbf{s}(90^\circ) = \begin{bmatrix} e^{j(0)\pi} & e^{j(1)\pi} & \cdots & e^{j(N-1)\pi} \end{bmatrix}^\mathsf{T}. \tag{2.3}$$

The geometry of the array can be used to find the steering vector for any spatial angle ϕ. Using Figure 2.2 and some simple trigonometry, it is easy to see that the wave will travel a distance of $d = (\lambda/2)\sin\phi$ between each element. The associated phase shift θ is then $(d/\lambda)(2\pi)$, or

$$\theta = \pi \sin\phi, \tag{2.4}$$

which is often referred to as the *electrical angle*. The electrical angle and the spatial angle are directly related, but the electrical angle subsumes the wavelength and allows ULAs to be treated

more generally. Accordingly, we can describe any general steering vector for a ULA as

$$\mathbf{s}(\theta) = \begin{bmatrix} e^{j(0)\theta} & e^{j(1)\theta} & \dots & e^{j(N-1)\theta} \end{bmatrix}^{\mathsf{T}}. \tag{2.5}$$

This particular description of steering vectors, in which the entries' exponents progress linearly, is called the Vandermonde form [14, pp. 37].

For an N element array, each length N steering vector, as a function of discretized electrical angle, is collection into the steering vector matrix. For M discrete angles, this matrix is expressed as

$$\mathbf{S} = \begin{bmatrix} \mathbf{s}(0) & \mathbf{s}(\theta) & \mathbf{s}(2\theta) & \cdots & \mathbf{s}((M-1)\theta) \end{bmatrix}. \tag{2.6}$$

2.3 Standard (Non-Adaptive) Beamforming

Suppose a signal is incident on the array at some angle θ. What is received at the array is the original signal multiplied by the associated steering vector, plus a vector of noise $\mathbf{v}(\ell)$

$$\mathbf{y}(\ell) = x(\ell)\mathbf{s}(\theta) + \mathbf{v}(\ell). \tag{2.7}$$

Pre-multiplying the received signal from (2.7) by $\mathbf{s}^{\mathrm{H}}(\theta)$, which is the standard (non-adaptive) beamformer for electrical angle θ, then yields

$$\begin{aligned} \mathbf{s}^{\mathrm{H}}(\theta)\mathbf{y}(\ell) &= \mathbf{s}^{\mathrm{H}}(\theta)\mathbf{s}(\theta)x(\ell) + \mathbf{s}^{\mathrm{H}}(\theta)\mathbf{v}(\ell) \\ &= Nx(\ell) + \mathbf{s}^{\mathrm{H}}(\theta)\mathbf{v}(\ell), \end{aligned} \tag{2.8}$$

where $^{\mathrm{H}}$ is the Hermitian operator, or complex conjugate transpose. The imaginary components of the first term all sum to zero, resulting in coherent integration of the signal, while the noise, assuming it has random phase, would be largely unaffected.

For a signal from direction θ_o, $\mathbf{s}(\theta)^{\mathrm{H}}\mathbf{y}(\ell)$ is maximized when $\theta = \theta_o$, and drops off as θ

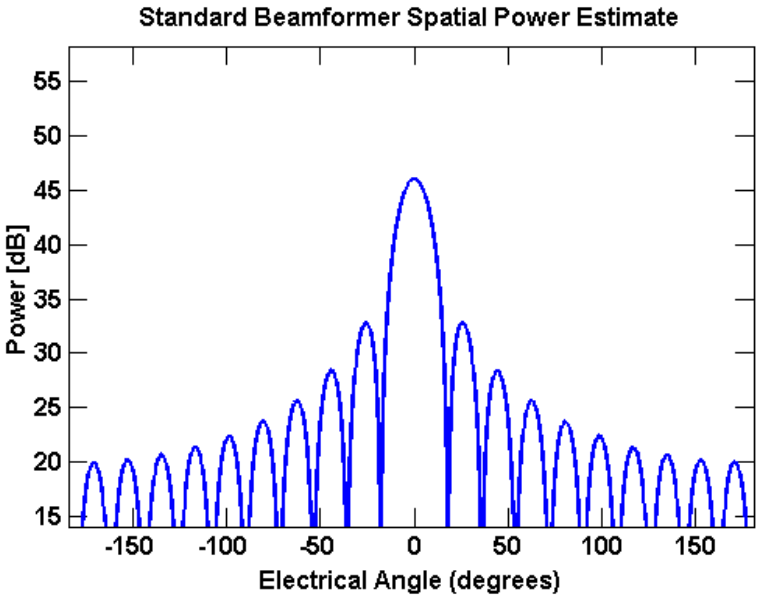

Figure 2.6: Spatial power estimate using the standard beamformer of single source at 0°.

becomes dissimilar to θ_o. Using the Hermitian of the steering vector matrix to coherently integrate up the signal is the principle behind *matched filtering*. Each of the M vectors in $\mathbf{S}$ is applied to the received signal and the result is a spatial spectrum that shows how correlated each of the steering vectors is to the received signal. The result is called the *spatial complex amplitude estimate*

$$\mathbf{p}(\theta, \ell) = \mathbf{S}^{\mathrm{H}}\mathbf{y}(\ell). \tag{2.9}$$

Figure 2.6 shows an example of what a matched filter power spectrum looks like for a single source at 0°. Note that in addition to a fairly wide main lobe, there are also many sidelobes that contain much less power, but are still significant. One can imagine that if several signals were spaced closely, it would be difficult to determine their number.

Matched filtering, also known as the *standard beamformer* is the most basic type of beamforming.

When the standard beamformer is employed, each lobe will have a fixed beamwidth equal to $360^\circ/N$ in electrical angle. When two signals are separated by an angle of at least one beamwidth, i.e. when their mainlobes no longer overlap, they are generally separable using the standard beam-

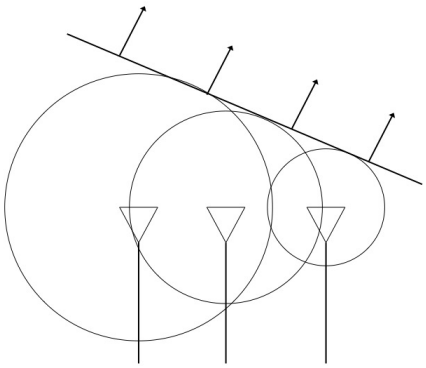

Figure 2.7: Using phase delays to steer a wavefront away from the array at an angle

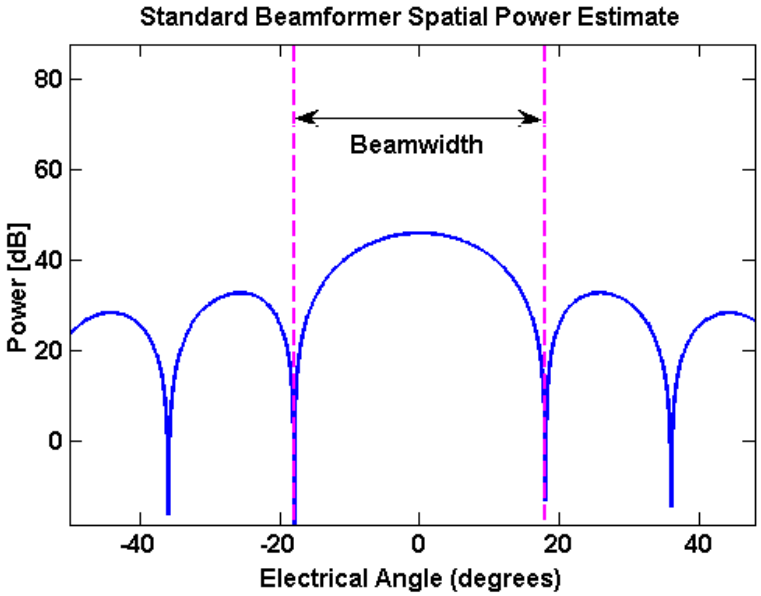

Figure 2.8: The beamwidth of the standard beamformer is determined solely by the number of antenna elements. Here $N = 10$ elements corresponds to a beamwidth of $36°$.

former (though there are many factors such as signal correlation and SNR that can make signals inseparable even when their mainlobes do not overlap). $360°/N$ is known as the **Rayleigh resolution** or the nominal resolution. If a method that can separate signals that are closer than the Rayleigh resolution it is considered to achieve super-resolution [14, pp. 48].

Chapter 3

Super-resolution and Adaptive Signal Processing

3.1 Definition of Super-resolution and Applications

As shown in Chapter 2, the standard beamformer has limited resolution, which can be especially problematic when the number of antenna elements at one's disposal is small. Recall that the Rayleigh Resolution for N antenna elements in a ULA is

$$\text{Rayleigh Resolution} = \frac{360^\circ}{N} = \frac{2\pi}{N}. \tag{3.1}$$

Depending on the degree of resolution one hopes to achieve many antenna elements my be necessary. Furthermore, if signals are correlated, or if they have different power levels, then even at the Rayleigh resolution separability cannot be guaranteed. The standard beamformer does not suffice for some applications that require high resolution, which leads to the development of super-resolution techniques that can separate signals spaced closer than the Rayleigh resolution.

Super-resolution is a blanket term for any algorithm or formulation that results in an estimate providing more resolution than the standard beamformer does. Algorithms can be characterized by the degree of resolution they achieve relative to the Rayleigh resolution. For example, if an

algorithm can separate signals at half the nominal resolution, it is said to have a super-resolution factor of two. Unfortunately, even in terms of super-resolution factor, it can be difficult to compare super-resolution techniques fairly because, as discussed in Chapter 1, there are many factors such as resilience to signal correlation and supported array structures that cannot be easily compared. Some algorithms require ULAs be used, but others can be used with any known array. Finally, some perform better in low SNR regimes than others do. The net effect is that it is difficult to design a fair test for comparing super-resolution the context of a specific application.

Super-resolution algorithms are applicable in most settings that sensor arrays are used, such as radar, cellular communications systems, medical imaging, speech processing, audio engineering, and astronomical studies. The only real limitation to where super-resolution techniques can be used is computing power. Some super-resolution DOA techniques require significant computational complexity, which means some may not be an option if power is limited, memory is limited, or if computation needs to be performed in real time.

3.2 Introduction to Adaptive Signal Processing

The reason that many super-resolution techniques require so much computing power is because they all rely on *adaptive signal processing* techniques. Recall that the standard beamformer in Chapter 2 was based solely on the geometry of the array. No statistical information about the received signal is considered. In adaptive signal processing, the received signal is used in addition to the array geometry to provide an improved estimate of DOA, resulting in super-resolution. It is termed adaptive because the resulting filter is a function of the received signal. For standard beamforming, little computation was needed aside from applying the matched filter, while in adaptive beamforming the received signal must constantly be analyzed and updates must be made any time the statistical properties of the signal change. As a result, a higher computational cost is incurred, though resolution is enhanced. It should be noted that there is tremendous variety among adaptive signal processing techniques, and that some are remarkably computationally efficient, while others

cannot be run in real time even on modern super computers. Broadly speaking, techniques that involve finding the expectation of a random variable tend to have high computational cost than those that only operate on one snapshot of data at a time.

3.2.1 Degrees of Freedom

An important concept in adaptive signal processing is degrees of freedom. Degrees of freedom allow adaptive algorithms to place nulls or amplify signals in a certain directions and they have a large influence on how well DOA estimation can be performed. The number of available degrees of freedom is dictated by the number of independent data channels (e.g. number of antenna elements).

For an array with $N = 10$ elements, there are likewise 10 degrees of freedom. One degree of freedom is used to focus on a specific direction of interest, which leaves 9 degrees of freedom to to suppress other signals so that the direction of interest can be estimated accurately. Degrees of freedom are also used up when constraints are applied. In the next section we will apply a gain constraint to a DOA estimation algorithm, resulting is a loss of one degree of freedom. In these ways, degrees of freedom are helpful for understanding how array size and the number of signals present factor into the problem of DOA estimation.

3.3 Example: Capon Beamformer

The Capon beamformer, also known as the Minimum Variance Distortionless Response (MVDR) beamformer, is a common adaptive beamformers and a good introduction to how the statistical properties of the received signal can be used to improve upon the standard beamformer [14, pp.440-442]. Consider estimating the complex magnitude in a direction θ at time index ℓ as

$$\hat{x}(\ell, \theta) = \mathbf{w}^{\mathrm{H}}(\theta)\mathbf{y}(\ell), \tag{3.2}$$

where $\mathbf{y}(\ell)$ is the received signal and $\mathbf{w}(\theta)$ is a spatial filter for the direction θ. The received signal model is

$$\mathbf{y}(\ell) = \sum_{k=1}^{K} x_k(\ell)\mathbf{s}(\theta_k) + \mathbf{n}(\ell), \tag{3.3}$$

where there are K incident signals and $\mathbf{n}(\ell)$ is a vector of additive noise.

Recall from Figure 2.6 that the standard beamformer produced many sidelobes. To suppress sidelobes, one can minimize the squared magnitude of $\hat{x}(\ell,\theta)$, which can be written as the cost function

$$\mathrm{J} = \mathrm{E}[\left|\hat{x}(\ell,\theta)\right|^2]. \tag{3.4}$$

This cost function is desirable because it is quadratic, which means that it only has one minimum, so by using the gradient to search for extrema one is guaranteed to find the global minima. To avoid the trivial solution of $\mathbf{w} = \mathbf{0}$ a constraint is needed. If $\mathbf{w}^{\mathrm{H}}(\theta)\mathbf{s}(\theta) = g$ is enforced, then by combining (3.2) and (3.3) it is possible to show that

$$\hat{x}(\ell,\theta) = \sum_{k=1}^{K} g(x_k) + \mathbf{w}(\theta_k)^{\mathrm{H}}\mathbf{n}(\ell) \tag{3.5}$$

Since it is simply an arbitrary scaling factor any value of g is valid, with $g = 1$ yielding the minimum variance distortionless response (MVDR) formulation. The constraint then becomes

$$\mathbf{w}^{\mathrm{H}}(\theta)\mathbf{s}(\theta) = 1. \tag{3.6}$$

With the cost function and constraint defined, a Lagrange multiplier must be used to apply the constraint to the cost function, resulting in an unconstrained function for optimization. Adding a gain constraint reduces the available degrees of freedom by one, but is necessary to avoid the trivial solution. With the constraint incorporated, the cost function becomes

$$\mathrm{J}_{\mathrm{const}} = \mathrm{E}[\left|\hat{x}(\ell,\theta)\right|^2] + \Re[\lambda^*(\mathbf{w}^{\mathrm{H}}(\theta)\mathbf{s}(\theta) - 1)]. \tag{3.7}$$

The first term can be simplified some by omitting the indices and rewriting it as

$$\begin{aligned} \mathrm{E}[|\hat{x}|^2] =& \mathrm{E}[(\mathbf{w}^{\mathrm{H}}\mathbf{y})^{\mathrm{H}}(\mathbf{w}^{\mathrm{H}}\mathbf{y})] \\ =& \mathrm{E}[(\mathbf{w}^{\mathrm{H}}\mathbf{y})(\mathbf{x}^{\mathrm{H}}\mathbf{w})] \\ =& \mathbf{w}^{\mathrm{H}}\mathrm{E}[\mathbf{y}\mathbf{y}^{\mathrm{H}}]\mathbf{w} \\ =& \mathbf{w}^{\mathrm{H}}\mathbf{R}\mathbf{w}, \end{aligned} \tag{3.8}$$

where $\mathbf{R}$ is the covariance matrix of the received signal. The cost function can then be rewritten as

$$\mathrm{J}_{\mathrm{const}} = \mathbf{w}^{\mathrm{H}}\mathbf{R}\mathbf{w} + \mathbb{R}[\lambda^*(\mathbf{w}^{\mathrm{H}}\mathbf{s} - 1)]. \tag{3.9}$$

With the constraint incorporated directly into the cost function, the optimal filter can now be found by taking the gradient of the cost function with respect to $\mathbf{w}^{\mathrm{H}}$ and solving the resulting dual problem. Taking the gradient results in

$$\bigtriangledown \mathrm{J}_{\mathrm{const}} = \mathbf{R}\mathbf{w} + \lambda^*\mathbf{s}(\theta) \equiv 0. \tag{3.10}$$

From this result it is clear that

$$\mathbf{w} = -\lambda^*\mathbf{R}^{-1}\mathbf{s}, \tag{3.11}$$

but λ^* still needs to be specified using the constraint, which can be accomplished by substituting (3.11) into (3.6) and solving for λ^* as follows:

$$\begin{aligned} [\mathbf{w}^{\mathrm{H}}\mathbf{s}]^* &= [1]^* \\ \mathbf{s}^{\mathrm{H}}\mathbf{w} &= 1 \\ -\lambda^*\mathbf{s}^{\mathrm{H}}\mathbf{R}^{-1}\mathbf{s} &= 1 \\ \lambda^* &= \frac{-1}{\mathbf{s}^{\mathrm{H}}\mathbf{R}^{-1}\mathbf{s}} \end{aligned} \tag{3.12}$$

Finally, λ can be inserted back into the Capon beamformer filter of (3.11) to obtain

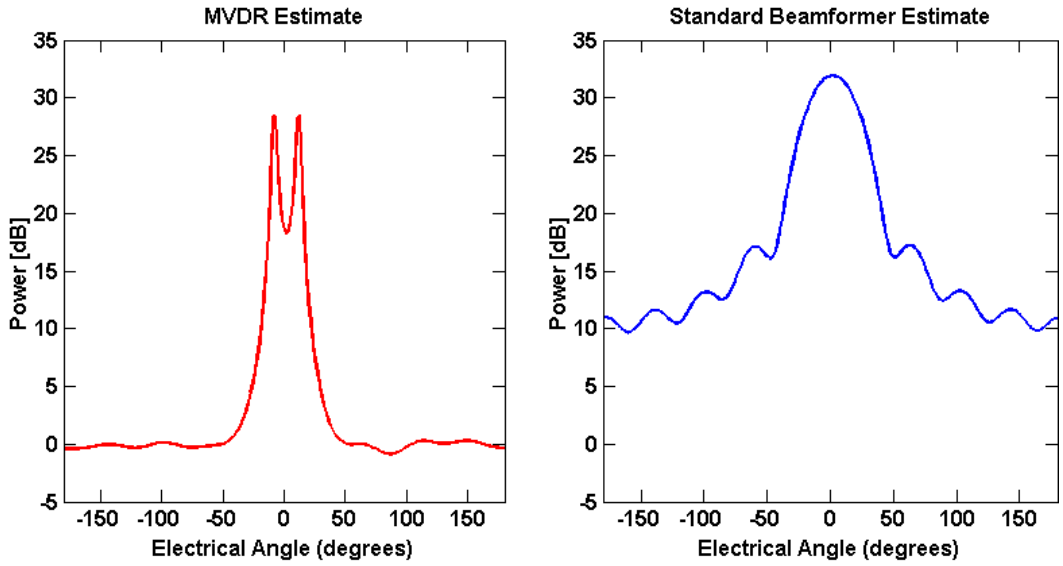

Figure 3.1: Comparison of the spatial power distribution estimate of two 30 dB signals at $\pm 10^\circ$ for the Capon beamformer (left) and standard beamformer (right). The Capon beamformer achieves super-resolution because it successfully separates signals 20° apart, much less than the Rayleigh resolution of 36°.

$$\mathbf{w} = \frac{\mathbf{R}^{-1}\mathbf{s}}{\mathbf{s}^{\mathrm{H}}\mathbf{R}^{-1}\mathbf{s}}. \tag{3.13}$$

One can determine the spatial power spectrum for this filter. It was shown in (3.8) that the power $\left|\hat{x}(n)\right|^2$ is equal to $\mathbf{w}^{\mathrm{H}}\mathbf{R}\mathbf{w}$. If the solution for $\mathbf{w}$ is inserted here, the result is

$$\begin{aligned} \mathrm{p}_{\mathrm{capon}}(\ell,\theta) &= \left(\frac{\mathbf{R}^{-1}\mathbf{s}(\theta)}{\mathbf{s}^{\mathrm{H}}(\theta)\mathbf{R}^{-1}\mathbf{s}(\theta)}\right)\mathbf{R}\left(\frac{\mathbf{R}^{-1}\mathbf{s}(\theta)}{\mathbf{s}^{\mathrm{H}}(\theta)\mathbf{R}^{-1}\mathbf{s}(\theta)}\right) \\ &= \frac{1}{\mathbf{s}^{\mathrm{H}}(\theta)\mathbf{R}^{-1}\mathbf{s}(\theta)}. \end{aligned} \tag{3.14}$$

Figure 3.1 is a comparison of the same signal's spatial power distribution estimated with both the standard beamformer and the Capon beamformer. Clearly, the Capon beamformer is superior to the standard beamformer in this setting. It performs better because it is using the sample covariance matrix $\mathbf{R}$ to leverage information from the data that simple beamforming does not take into account. Thus the Capon beamformer's performance is dependent on the accuracy of the estimate of $\mathbf{R}$. When there are sufficient samples to form an SCM, then the performance will be nearly optimal,

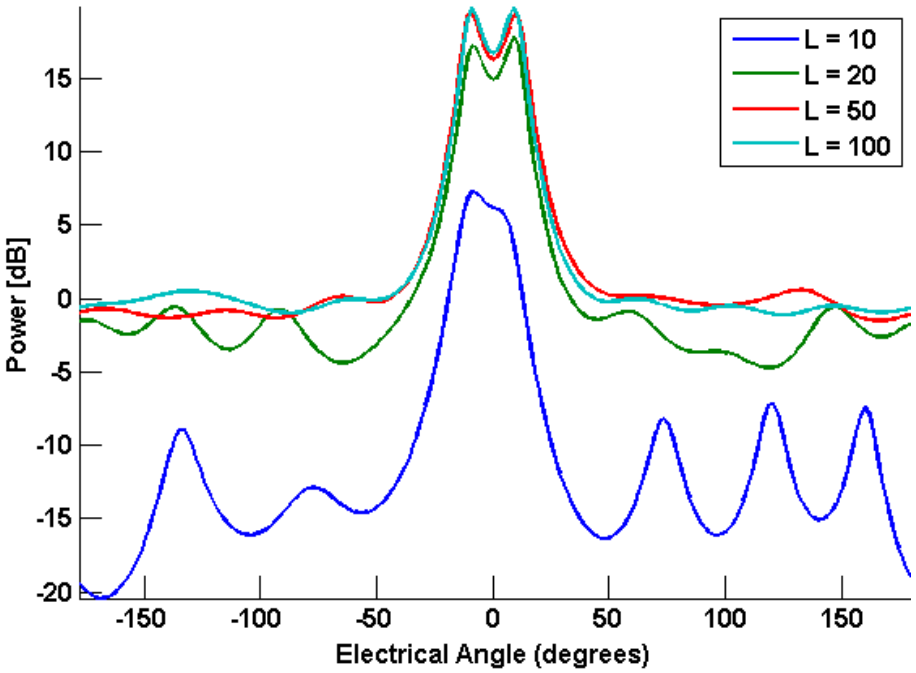

Figure 3.2: Two 20 dB signals are at $\pm 10^{\circ}$. The Capon beamformer's is sensative to sample support.

but in sample starved environments, like when the signal does not remain stationary for very long, its utility will be limited. Figure 3.2 shows how sample support impacts the estimated power spectrum.

Many adaptive beamforming techniques employ a sample covariance matrix like the Capon beamformer because it is an effective way to exploit extra information contained in the data. However, as can be seen in Figure 3.2, having limited sample support affects the spatial power distribution estimate. Such approaches have limited utility in low sample support environments, but in the next chapter a more robust method is introduced that can achieve super-resolution using as few as a single spatial snapshot.

3.4 Multiple Signal Classification (MUSIC)

The Multiple Signal Classification (MUSIC) algorithm is perhaps one of the most popular super-resolution algorithms. Its implementation is relatively straightforward and it has excellent perfor-

mance. In fact, its ease of implementation and performance have helped it to become a benchmark against which other super-resolution algorithms are compared. Here we introduce MUSIC in its simplest form, and later we discuss some improvements to MUSIC [14, p.1158].

MUSIC's models signals as

$$\mathbf{y}(\ell) = \mathbf{x}(\ell) + \mathbf{n}(\ell) \tag{3.15}$$

where $\mathbf{y}(\ell)$ is the $M \times 1$ received signal vector at time index ℓ, $\mathbf{x}(\ell)$ is a superposition of $P < M$ signals and their steering vectors, and $\mathbf{n}(\ell)$ is additive noise. The vector $\mathbf{x}(\ell)$ can be described by

$$\mathbf{x}(\ell) = \sum_{i=1}^{P} a_i(\ell)\mathbf{s}(\theta_i) \tag{3.16}$$

where a_i are the P signals' amplitudes and θ_i is the angle associated with a_i. If we assume that the signal and noise are independent then the received signal sample covariance matrix (SCM) can be written as

$$\mathbf{R}_{yy} = \mathbf{R}_{xx} + \mathbf{R}_{nn}. \tag{3.17}$$

Assuming white noise, $\mathbf{R}_{nn} = \sigma_n^2\mathbf{I}$. Similarly, $\mathbf{R}_{xx}$ can be expressed as $\mathbf{R}_{xx} = \sum_{i=1}^{P} \mathrm{E}[|a_i|^2]\mathbf{s}(\theta_i)\mathbf{s}^{\mathrm{H}}(\theta_i)$. Note that while $\mathbf{R}_{yy}$ is full rank because of the noise, $\mathbf{R}_{xx}$ is only rank P. Then taking the eigendecomposition of the right side

$$\mathbf{R}_{yy} = \mathbf{V}(\Lambda_x + \Lambda_n)\mathbf{V}^{\mathrm{H}}, \tag{3.18}$$

where $\mathbf{V}$ is a unitary eigenvector matrix and Λ_x and Λ_n are diagonal eigenvalue matrices for $\mathbf{R}_x x$ and $\mathbf{R}_y y$ respectively. Deconstructing each term further reveals that

$$\mathbf{R}_{yy} = \sum_{i=1}^{P} \lambda_{x,i}\mathbf{v}_i\mathbf{v}_i^{\mathrm{H}} + \sum_{i=1}^{M} \sigma_n^2\mathbf{v}_i\mathbf{v}_i^{\mathrm{H}}. \tag{3.19}$$

Taking care to keep track of the summation indices, (3.19) can be modified further to show that

$$\mathbf{R}_{yy} = \sum_{i=1}^{P} (\lambda_{x,i} + \sigma_n^2)\mathbf{v}_i\mathbf{v}_i^{\mathrm{H}} + \sum_{i=P+1}^{M} \sigma_n^2 \mathbf{v}_i\mathbf{v}_i^{\mathrm{H}}. \tag{3.20}$$

Thus $\mathbf{R}_{yy}$ can be expressed as the sum of two *orthogonal* subspaces. The first is called the signal+noise subspace and the term is called the noise-only subspace. These subspaces are guaranteed to be orthogonal to one another because the columns of a unitary matrix are orthogonal. Furthermore, it can also be shown using (3.16) - (3.20) that the steering vectors associated with the incident signal span the same subspace as the eigenvectors corresponding to the signal+noise subspace.

MUSIC then employs a clever trick to locate signals. If we take the inner product of an eigenvector from the noise subspace with a steering vector, the result will be zero if steering vector belongs to the signals subspace and nonzero if it belongs to the noise subspace. It follows then, that if we can determine which eigenvalues belong to the noise subspace, then we can determine the steering vectors that belong to the signal subspace. It is often possible to distinguish between the noise subspace and signal subspace by taking an eigendecomposition and observing the eigenvalues' distribution. As can be seen in Figure 3.3, when eigenvalues are arranged in order of magnitude it is sometimes very easy to determine which eigenvalues belong to which subspace. However, there are also conditions (such as low SNR) in which it is difficult to determine precisely where the signal subspace ends and the noise subspace begins. Popular methods include the Akaike Information Criterion and Bayesian Information Criterion [20] but neither will be discussed in depth here.

Once the order of each subspace has been determined, form

$$Q(\phi) = \sum_{k=P+1}^{M} \left| \mathbf{s}^{\mathrm{H}}(\phi)\mathbf{v}_k \right|^2. \tag{3.21}$$

Because of orthogonality, $Q(\phi)$ will have a very small value anywhere that a signal is present, and a large value anywhere that a signal does not exist. Finally, A pseudo-spectrum estimate can then be obtained by inverting $Q(\phi)$. Figure 3.4 shows a MUSIC pseudo-spectrum estimate of two 20

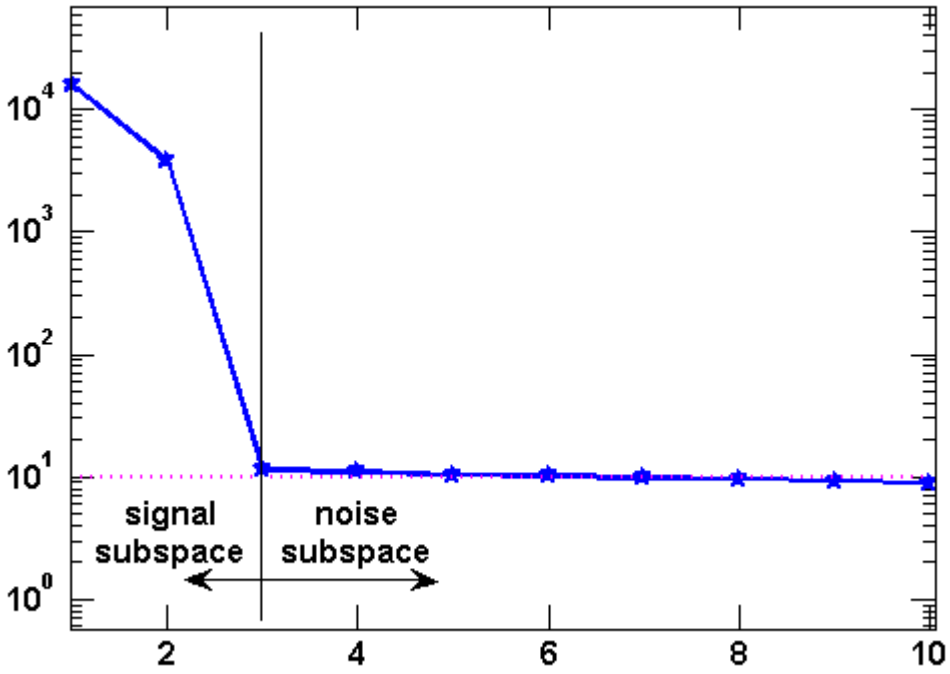

Figure 3.3: The signal subspace and noise subspace can be determined by arranging the eigenvectors of the received signal covariance matrix in descending order. In this instance, the signal subspace has an order of 2, and the noise subspace has an order of 8.

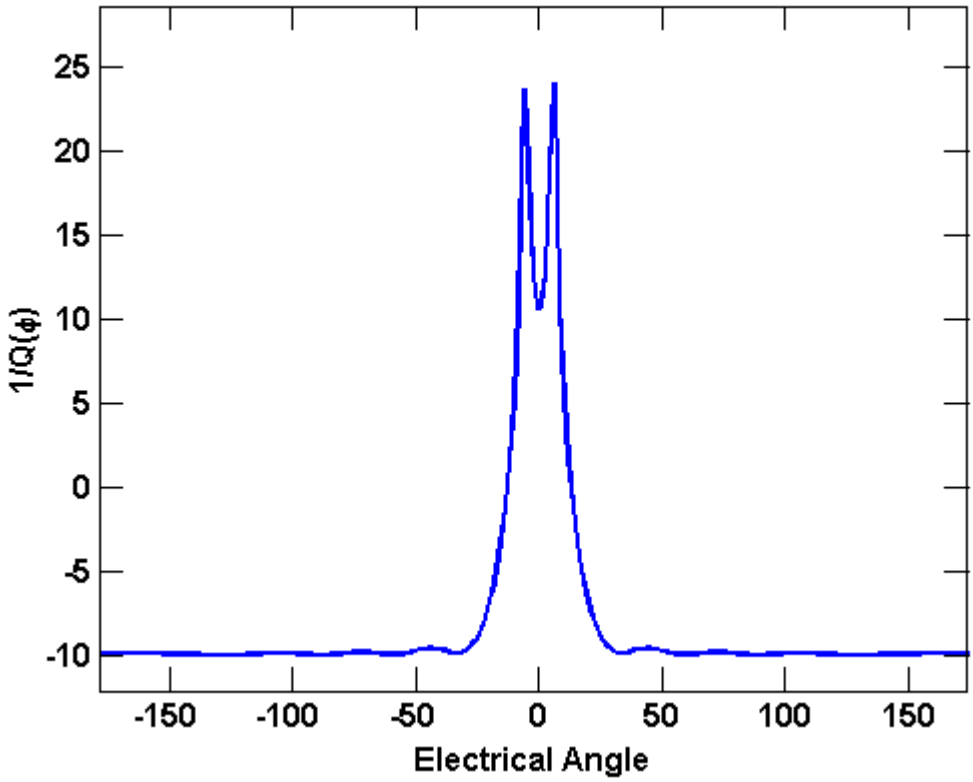

Figure 3.4: MUSIC pseudo-spectrum estimate of two 20 dB signals at $\pm 9^\circ$ (1/4 nominal resolution). Note that MUSIC does not provide an estimate of signal power.

dB signals at $\pm 9^\circ$ (1/4 of Rayleigh resolution). The reason that MUSIC's estimate is referred to as a pseudo-spectrum instead of a spectrum like the previous Capon beamformer's is because MUSIC does *not* provide an estimate of the signals' powers, but rather just an estimate of their DOA. If an estimate of the power is desired, a separate power estimation technique must be applied after estimating the DOA with MUSIC.

3.5 Other Versions of MUSIC

The version of MUSIC introduced here is among the more basic versions of MUSIC. Many other versions of MUSIC have been developed that increase its precision and robustness, including cyclic MUSIC and root-MUSIC. As its name implies, root-MUSIC uses root finding to locate sources. Unlike MUSIC, it does not produce a spectrum, but instead outputs a vector containing DOA

estimates. Root-MUSIC is considerably more robust than MUSIC and does not require any peak detection logic, which makes it a popular choice. Estimation of Signal Parameters via Rotational Invariance Techniques (ESPRIT) [22] is another eigenspace DOA alogorithm similar to MUSIC that itself has several versions [14, pp.1170].

Chapter 4

Reiterative Super-resolution Algorithm

4.1 Motivation for RISR

The next beamformer that will be treated, ReIterative Super-Resolution (RISR), is an algorithm. Unlike the Capon beamformer which is a one step optimization, RISR is a comprised of a set of calculations that have to be repeated several times before a solution is reached.

There are a number of characteristics that make RISR unique, but it is perhaps most noteworthy for its robustness. RISR can be implemented with any known array manifold, takes unknown array modeling errors into account, and can operate on as few as just one snapshot of data, which allows it to be used on signals that do not exhibit prolonged stationarity.

RISR's derivation includes two unique features that give it its robustness. First, instead of employing a sample covariance matrix like that used in the Capon beamformer, RISR uses a spatial covariance matrix formed from the correlation between antennas elements. Because correlation is calculated between elements instead of between time samples, RISR can operate on as few as one snapshot.

Second, its formulation includes a calibration error covariance matrix that helps account for imprecision in the antenna elements' gain and phase. Typically, the calibration error covariance matrix can be determined from the manufacturing tolerances of the array itself, so there is no need

to estimate it from the data [30] [29].

4.2 Data Model

The data model that RISR employs is

$$\mathbf{y}(\ell) = (\mathbf{S}\mathbf{x}(\ell)) \odot \mathbf{z} + \mathbf{v}(\ell), \tag{4.1}$$

where $\mathbf{y}(\ell)$ is an $N \times 1$ vector of received signal values at time index ℓ, with one entry corresponding to each antenna element. $\mathbf{S}$ is an $N \times M$ matrix of steering vectors, and $\mathbf{x}(\ell)$ is an $M \times 1$ spatial complex magnitude vector, in which each entry contains the true signal from the m^{th} direction. $\mathbf{z}$ is a column vector of complex values that incorporates the unknown modeling errors and $\odot$ represents element wise multiplication, otherwise known as the Hadamard product. Finally, $\mathbf{v}(\ell)$ is additive noise, which is often modeled as additive white Gaussian noise but could have any distribution. Note that neither $\mathbf{S}$ or $\mathbf{z}$ are functions of the time index ℓ. They are the same for all snapshots, because the array geometry is assumed not to change as a function of time.

Each entry in $\mathbf{z}$ is a complex number that describes the gain and phase error of a particular antenna element and can be mathematically specified as

$$z_n = [1 + \Delta_{a,n}] e^{j\Delta_{\phi,n}} \tag{4.2}$$

where $\Delta_{a,n}$ and $\Delta_{\phi,n}$ are some amplitude and phase deviation, both with arbitrary distributions. Assuming that all of the antennas are from the same distribution, z_m will have some variance σ_z^2. The deviation of each element is presumed to be independent, so the calibration error can be modeled as noise by making the approximation

$$\begin{aligned} \mathbf{y}(\ell) &= ((\mathbf{S}\mathbf{x}(\ell)) \odot \mathbf{z} + \mathbf{v}(\ell) \\ &\approx \mathbf{S}\mathbf{x}(\ell) + \mathbf{v}(\ell) + \mathbf{v}_z. \end{aligned} \tag{4.3}$$

In the coming sections, it will be necessary to find the covariance matrix for $\mathbf{v}_z$, so we will do that here in advance. Working from (4.3), it is simple to show that

$$\begin{aligned} \mathbf{Sx}(\ell) \odot \mathbf{z} &= \mathbf{Sx}(\ell) + \mathbf{v}_z \\ \mathbf{v}_z &= \mathbf{Sx}(\ell) \odot \mathbf{z} - \mathbf{Sx}(\ell) \\ \mathbf{v}_z &= \mathbf{Sx}(\ell) \odot (\mathbf{z} - \mathbf{1}). \end{aligned} \tag{4.4}$$

From the definition of a covariance matrix, the covariance matrix for $\mathbf{v}_z$ will be

$$\begin{aligned} \mathbf{R}_z &= \mathrm{E}[\mathbf{v}_z \mathbf{v}_z^{\mathrm{H}}] \\ &= \mathrm{E}[(\mathbf{z} - \mathbf{1}) \odot (\mathbf{Sx}(\ell))(\mathbf{Sx}(\ell))^{\mathrm{H}} \odot (\mathbf{z} - \mathbf{1})^{\mathrm{H}}] \\ &= \mathrm{E}[\tilde{\mathbf{Z}}(\mathbf{Sx}(\ell))(\mathbf{Sx}(\ell))^{\mathrm{H}} \tilde{\mathbf{Z}}^{\mathrm{H}}] \end{aligned} \tag{4.5}$$

where

$$\tilde{\mathbf{Z}} = \begin{bmatrix} z_0 - 1 & 0 & \cdots & 0 \\ 0 & z_1 - 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & z_{N-1} - 1 \end{bmatrix}. \tag{4.6}$$

Using the two assumptions that modeling errors are uncorrelated to each other and to the source signal, (4.5) will simplify to

$$\mathbf{R}_z = \sigma_z^2 \mathbf{I}_{N \times N} \odot (\mathbf{SPS}^{\mathrm{H}}). \tag{4.7}$$

4.3 Derivation of RISR

Using the data model developed in the previous section (see (4.3)), RISR attempts to find some $N \times M$ filter bank $\mathbf{W}$ that will estimate $\mathbf{x}(\ell)$ from the snapshot $\mathbf{y}(\ell)$. Denoting the estimate of $\mathbf{x}(\ell)$ as $\hat{\mathbf{x}}(\ell)$ and omitting the time index,

$$\hat{\mathbf{x}} = \mathbf{W}^{\mathrm{H}}\mathbf{y}. \tag{4.8}$$

RISR finds $\mathbf{W}$ that minimizes the mean squared error between $\mathbf{x}$ and $\hat{\mathbf{x}}$. Using expectation to approximate the mean, the cost function can be written as

$$\mathrm{J} = \mathrm{E}\left[|\mathbf{x} - \hat{\mathbf{x}}|^2\right]. \tag{4.9}$$

Inserting the previous definition of $\hat{\mathbf{x}}$ from (4.8), the cost function becomes

$$\begin{aligned} J &= \mathrm{E}\left[\left|\mathbf{x} - \mathbf{W}^{\mathrm{H}}\mathbf{y}\right|^2\right] \\ &= \mathrm{E}\left[(\mathbf{x} - \mathbf{W}^{\mathrm{H}}\mathbf{y})(\mathbf{x} - \mathbf{W}^{\mathrm{H}}\mathbf{y})^{\mathrm{H}}\right] \\ &= \mathrm{E}\left[(\mathbf{x} - \mathbf{W}^{\mathrm{H}}\mathbf{y})(\mathbf{x}^{H} - \mathbf{y}^{\mathrm{H}}\mathbf{W})\right]. \end{aligned} \tag{4.10}$$

The next step is to perform optimization by minimizing the cost function. The gradient is taken with respect to $\mathbf{W}^{\mathrm{H}}$ and the result set equal to zero

$$\bigtriangledown J = \mathrm{E}\left[-\mathbf{y}\mathbf{x}^{\mathrm{H}}\right] + \mathrm{E}\left[\mathbf{y}\mathbf{y}^{\mathrm{H}}\right]\mathbf{W} \equiv 0. \tag{4.11}$$

Note that because the cost function is quadratic, the gradient is guaranteed to locate the global minimum. Recalling the data model given in (4.3), substitute in for $\mathbf{y}$. Assuming that $\mathbf{v}$ and $\mathbf{v}_z$ are both zero mean causes some of the expectations to drop out, resulting in

$$\bigtriangledown J = -\mathbf{S}\mathrm{E}\left[\mathbf{x}\mathbf{x}^{\mathrm{H}}\right] + \mathrm{E}\left[(\mathbf{S}\mathbf{x} + \mathbf{v} + \mathbf{v}_z)(\mathbf{x}^{\mathrm{H}}\mathbf{S}^{\mathrm{H}} + \mathbf{v}^{\mathrm{H}} + \mathbf{v}_z^{\mathrm{H}})\right]\mathbf{W} = \mathbf{0}. \tag{4.12}$$

From here, define $\mathbf{P}$ as the spatial power distribution matrix, an $M \times M$ diagonal matrix containing $\mathrm{E}[x_m^2]$ as its entries. To emphasize that it is a diagnoal matrix, it can be written as $\mathbf{P} = \mathrm{E}[\mathbf{x}\mathbf{x}^{\mathrm{H}}] \odot \mathbf{I}_{M\times M}$. Using this new notation,

$$\bigtriangledown J = -\mathbf{S}\mathbf{P} + \left(\mathbf{S}\mathbf{P}\mathbf{S}^{\mathrm{H}} + \mathrm{E}[\mathbf{v}\mathbf{v}^{\mathrm{H}} + \mathbf{v}_z\mathbf{v}_z^{\mathrm{H}}]\right)\mathbf{W} = 0. \tag{4.13}$$

To simplify notation further, denote the covariance matrix of $\mathbf{v}$ and $\mathbf{v}_z$ as $\mathbf{R}$ and $\mathbf{R_z}$, respectively.

$$-\mathbf{SP} + [\mathbf{SPS}^{\mathrm{H}} + \mathbf{R} + \mathbf{R_z}]\mathbf{W} = 0. \tag{4.14}$$

Finally, solve for $\mathbf{W}$ to get the final filter solution

$$\mathbf{W} = (\mathbf{SPS}^{\mathrm{H}} + \mathbf{R} + \mathbf{R_z})^{-1}\mathbf{SP}. \tag{4.15}$$

4.3.1 Reiteration

An astute observer will realize that an estimate of the spatial power distribution $\mathbf{x}$ is needed in order to calculate $\mathbf{W}$. However, the reason for finding $\mathbf{W}$ is so that $\mathbf{x}$ can be estimated. How can this be? It is because RISR is an iterative solution in which $\mathbf{W}$ and $\mathbf{x}$ are used in turn to estimate one another. The steps for this iterative process are will be explained here.

In order to calculate the filter bank $\mathbf{W}$, an initial estimate of the spatial power distribution matrix $\mathbf{P}_0$ is necessary, and to calculate $\mathbf{P}_0$, some initial estimate $\hat{\mathbf{x}}_0$ is needed. To get the initial estimate of $\hat{\mathbf{x}}$, the standard beamformer introduced in Chapter 2 is used. Find the initial estimate of the spatial complex magnitude as

$$\hat{\mathbf{x}}_0 = \mathbf{S}^{\mathrm{H}}\mathbf{y}. \tag{4.16}$$

With this result, $\mathbf{P}_0$ can be calculated, and then the first solution for the filter bank, $\mathbf{W}_1$.

Next the data model in (4.3) can be used to return back to the beginning and obtain a new estimate of the spatial complex magnitude distribution,

$$\hat{\mathbf{x}}_1 = \mathbf{W}_1^{\mathrm{H}}\mathbf{y}. \tag{4.17}$$

From this point, iteration can be continued until either (1) the change in $\hat{\mathbf{x}}_i$ from one iteration to the next becomes smaller than some predetermined threshold, or (2) a chosen number of iterations

is completed. The basic pattern is

1. Use $\hat{\mathbf{x}}_{i-1}$ to find $\mathbf{P}_{i-1}$.
2. Find $\mathbf{W}_i$ with $\mathbf{P}_{i-1}$.
3. Obtain $\hat{\mathbf{x}}_i$ by using $\mathbf{W}_i$
4. Repeat

4.3.2 Incorporating Noncoherent Integration

The derivation given above is for a single spatial snapshot, however it is also possible to use multiple snapshots to boost RISR's performance even further, provided that the samples are taken over a period where the signal is stationary. The data model in (4.8) must be updated to incorporate L time samples. The new data model is

$$\hat{\mathbf{X}} = \bar{\mathbf{W}}^{\mathrm{H}}\mathbf{Y}. \tag{4.18}$$

In the equation above, $\hat{\mathbf{X}} = [\hat{\mathbf{x}}(\ell) \;\; \hat{\mathbf{x}}(\ell+1) \;\; \cdots \;\; \hat{\mathbf{x}}(\ell+L-1)]$ is an $M \times L$ matrix that holds an estimate of the spatial complex magnitude distribution for each time sample. $\mathbf{Y}$ is a similarly constructed $N \times L$ matrix containing the received signal at each of the N antennas at each of the L time samples. Accordingly, an average spatial power distribution can now be calculated as

$$\bar{\mathbf{P}} = [(1/L)\sum_{\tau=0}^{L-1} \hat{\mathbf{x}}(\ell+\tau)\hat{\mathbf{x}}^{\mathrm{H}}(\ell+\tau)] \odot \mathbf{I}_{M\times M}. \tag{4.19}$$

By inserting these new definitions of $\hat{\mathbf{X}}$, $\mathbf{Y}$, and $\hat{\mathbf{P}}$ into the algorithm developed in the previous section, RISR will be modified to use multiple snapshots to estimate the average filter bank $\hat{\mathbf{W}}$, resulting in improved DOA estimates.

Both RISR and SCM based approaches like the Capon beamformer perform better with more snapshots, but the way in which those snapshots are used is subtly different. For SCM approaches,

providing many samples allows for truer approximation of the SCM matrix, which results in a better DOA esimate. In RISR's case, more samples allows for noncoherent integration of the actual estimates themselves, which helps to suppress noise and improve performance.

4.3.3 Guide to Implementation

Implementation of the RISR algorithm is summarized below. Initialization is performed once L spatial snapshots have been collected and arranged in a matrix $\mathbf{Y}$.

1. Initialize:

   - Find initial estimate of complex amplitude distribution:

$$\hat{\mathbf{X}}_0 = \mathbf{S}^{\mathrm{H}}\mathbf{Y}$$

   - Find initial estimate of average spatial power distribution:

$$\bar{\mathbf{P}}_0 = [(1/L)\sum_{\tau=0}^{L-1} \hat{\mathbf{x}}_0(\ell+\tau)\hat{\mathbf{x}}_0^{\mathrm{H}}(\ell+\tau)] \odot \mathbf{I}_{M\times M}$$

   - By assumption or measurement, obtain noise only covariance matrix $\mathbf{R}$
   - Based on array tolerances, obtain the model error variance σ_z^2

2. Iterate: for $i = 1, 2, \ldots$

   - Calculate the RISR filter bank:

$$\bar{\mathbf{W}}_i = [\mathbf{S}\bar{\mathbf{P}}_{i-1}\mathbf{S}^{\mathrm{H}} + \sigma_z^2\mathbf{I}_{M\times M} \odot (\mathbf{S}\bar{\mathbf{P}}_{i-1}\mathbf{S}^{\mathrm{H}})]^{-1}\mathbf{S}\bar{\mathbf{P}}_{i-1}$$

   - Update estimate of spatial complex amplitudes:

$$\hat{\mathbf{X}}_i = \bar{\mathbf{W}}_i^{\mathrm{H}}\mathbf{Y}$$

- Update estimate of spatial power distribution:

$$\bar{\mathbf{P}}_i = [(1/L)\sum_{\tau=0}^{L-1} \hat{\mathbf{x}}_i(\ell+\tau)\hat{\mathbf{x}}_i^{\mathrm{H}}(\ell+\tau)] \odot \mathbf{I}_{M\times M}$$

3. Once iteration is completed, the RISR spatial magnitude distribution can be obtained as

$$x_m = \sqrt{\bar{\mathbf{P}}_{mm}}.$$

4. The number of sources and their magnitude can be estimated from the peaks in the RISR spectrum.

4.4 Performance Characteristics

In this section we will outline the basic performance of RISR, with particular attention given to convergence characteristics, resolution, and dynamic range, all with respect to SNR. How RISR's behavior is affected by sample support and calibration errors will be discussed, but not in exhaustive detail, as it is well documented elsewhere and not of significant importance in the development of PC-RISR in the next chapter [29] [30].

4.4.1 Convergence

As outlined in the previous section, RISR requires an initial estimate of the spatial power spectrum before iteration can begin. A matched filter is used to provide this estimate, so initially the RISR spectrum begins from the matched filter beamformer result. At each iteration, RISR attempts to suppress any part of the spectrum that appears not to be a signal, which results in the formation of deep nulls and sharp peaks. Typically RISR will reach convergence in roughly 10 iterations, but it may be slightly slower or faster depending on how many signals are present and how closely they are spaced. Generally, more signals spaced nearby require more iterations to separate than fewer signals placed far apart.

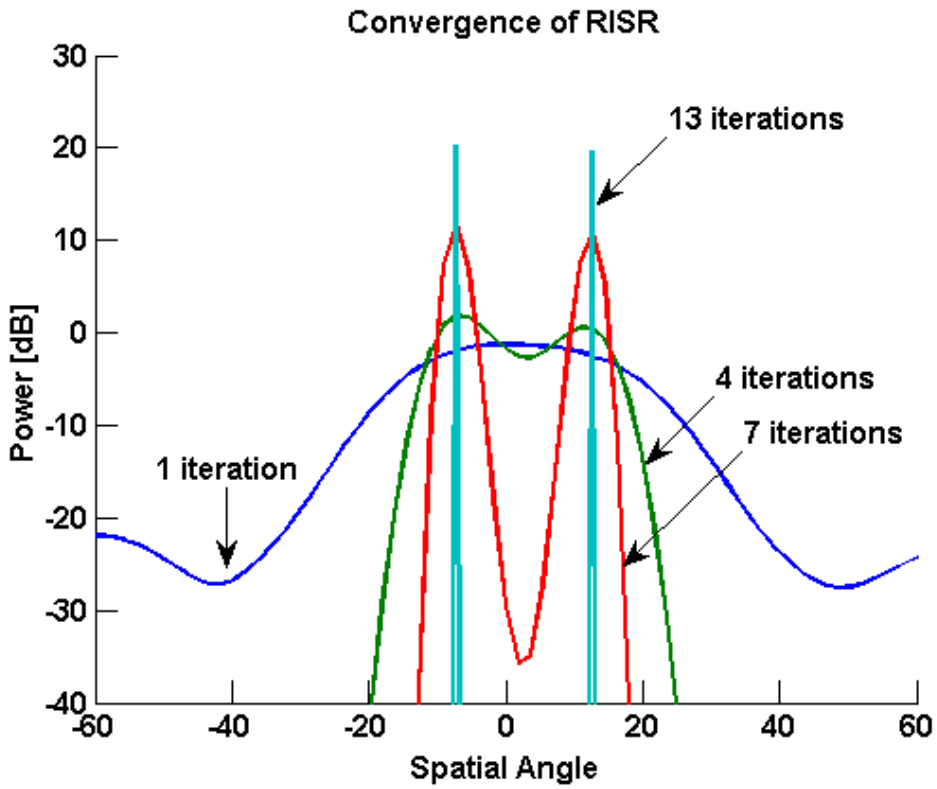

Figure 4.1: The convergence of RISR illustrated at several steps along the way. Two 20 dB signals are incident at $\pm 10^{\circ}$.

Figure 4.1 shows how RISR's estimate converges through each iteration. Note that RISR is not iterating over time samples. It is iterating back upon its own estimate in a recursive fashion. This is an important distinction because it is one of the factors that enables RISR to operate on very low sample support.

4.4.2 Super-Resolution

Resolution is of primary importance to all super-resolution algorithms, as it is a measure of their ability to separate closely spaced signals. As discussed in the introduction, codifying when two signals are separable is difficult because it can be affected by SNR, signal correlation, number of signals, array manifold, and other factors, making comparing any two methods in a fair manner difficult.

With that said, RISR operates well only at relatively high SNR values on the order of 20 dB

to 30 dB, so comparisons will be made in that regime. Root-MUSIC is a common benchmark, so RISR's results will be compared to root-MUSIC's to provide context for comparison.

Results in this section are obtained using Monte Carlo simulations in which two uncorrelated signals are incident on the array, with the first on the bore site and the second a predetermined positive angular distance away from the bore site. The number of antennas and snapshots used are $N = 10$ and $L = 10$, respectively, a spatial sampling factor of 20 ($M = 20N$) is used, and σ_z^2 is chosen to be $1.5 \times 10^{-}3$, corresponding roughly to 1% gain and phase errors in the array, and additive noise is modeled as complex additive white Gaussian noise. After running RISR for 10 iterations all data below the noise floor (0 dB) is thrown out (set to 0 dB) and peak detection is performed. If precisely two peaks are found and both lie within half of a beamwidth $(180^\circ/N)$ of their true direction, that trial is deemed a successful separation. Note that no null depth requirement is enforced because RISR's spectrum will only contain lobes a single point wide once full convergence is reached. For each permutation, 500 trials are run and the results compiled to produce the following plots.

Root-MUSIC is implemented using forward backward averaging and the Bayesian Information Criterion (BIC) is used to find an estimate of the model order[20][1][16]. Since root-MUSIC does not produce a spectrum like RISR does, but rather just a vector of estimated direction, we just check to ensure that both angles fall within a half a beamwidth of their true directions.

Figure 4.2 shows the probability that RISR successfully separates two signals as a function of how (angularly) far apart they are. The result is that RISR performs about as well as root-MUSIC at high SNR, but not as well at low SNR. Note that both RISR and root-MUSIC manage to separate signals spaced about 5° apart. The Rayleigh resolution is 36°, so this corresponds to a super-resolution factor of roughly 7. However, when the SNR gets down to about 15 dB, both root-MUSIC and RISR degrade. It seems that RISR will not be useful when the SNR is much below 20 dB, but Root-MUSIC is still viable, though an appreciable amount of its ability to super-resolve is lost.

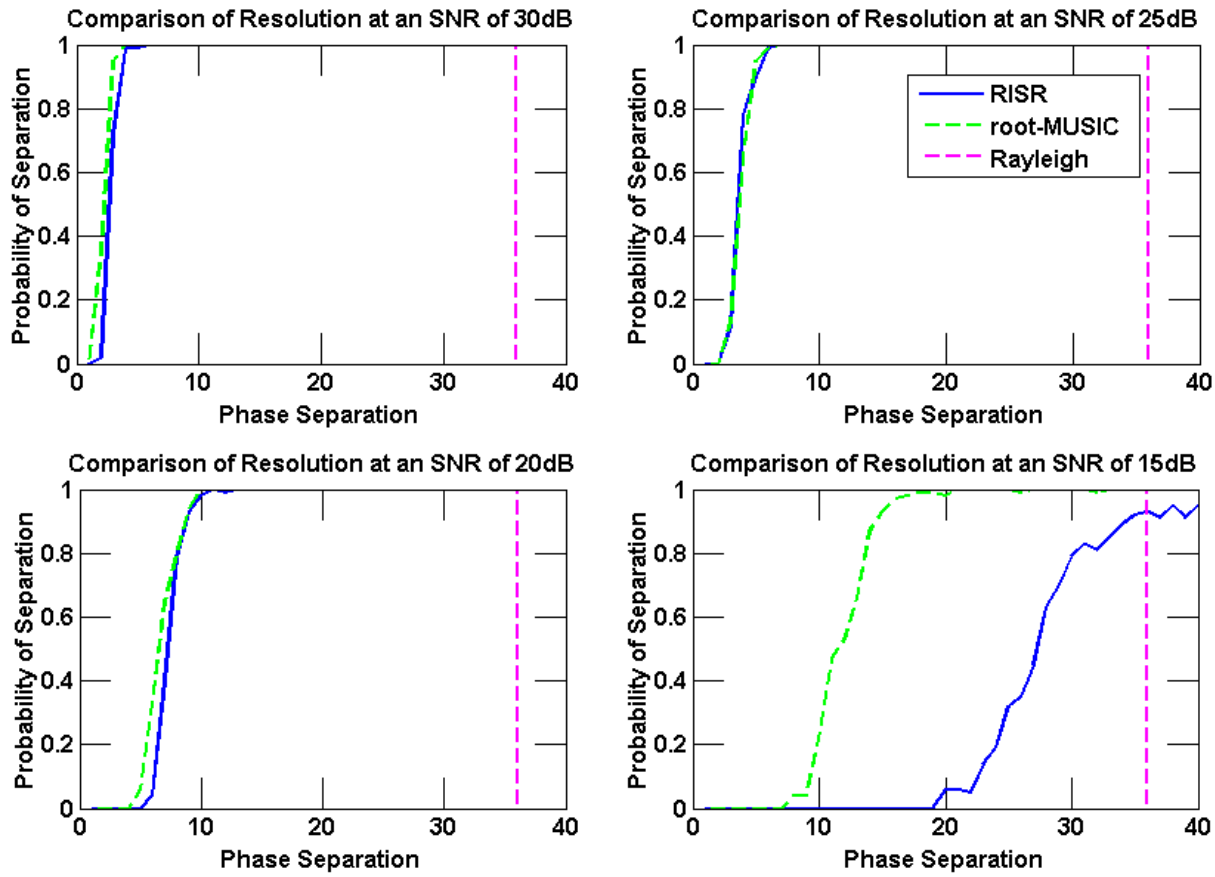

Figure 4.2: Comparison of resolution for RISR and root-MUSIC at various SNR shown with the Rayleigh resolution. Signals have random phase and are uncorrelated, $N = 10$ antennas and $L = 10$ snapshots are used.

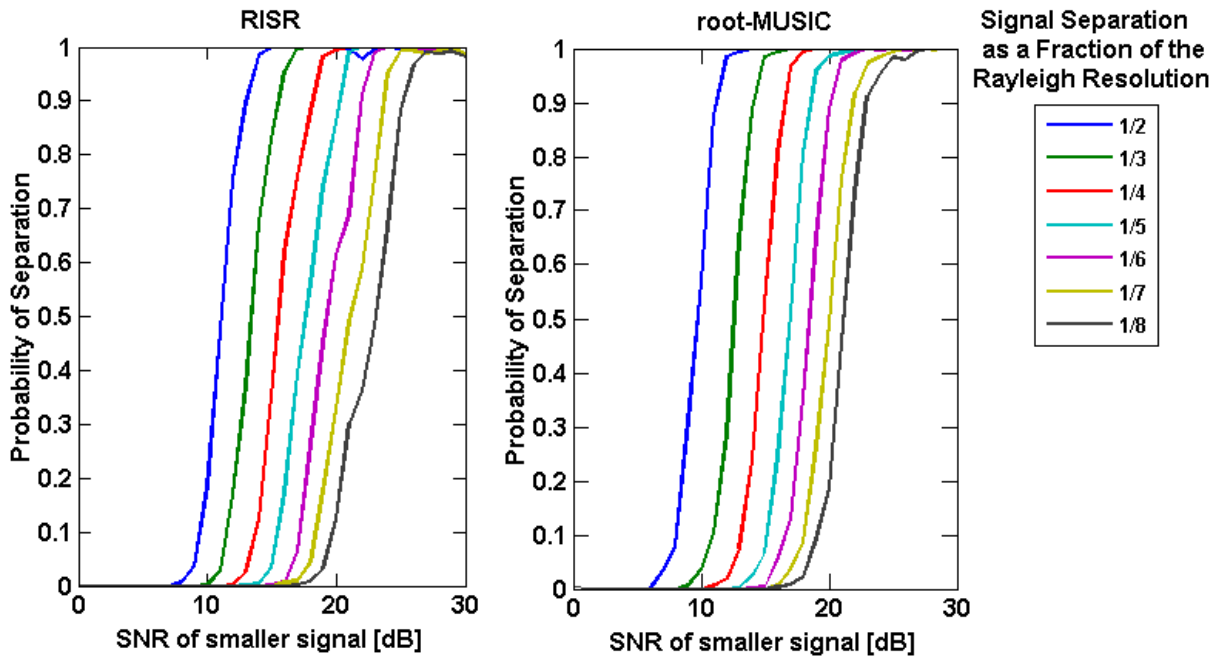

Figure 4.3: A comparison of RISR's and root-MUSIC's dynamic range for several separation values. The larger of two signals is held at 30 dB, $N = 10$ antenna elements, $L = 10$ snapshots, and signals are uncorrelated. Performance is comparable for RISR and root-MUSIC.

4.4.3 Dynamic Range

When examining super-resolution factor, both signals were specified to have the same power. In this section, we hold one of the signals constant while varying the second in order to characterize RISR's dynamic range.

In the results given in Figure 4.3 two sources are incident on the array. The SNR of the first signal is 30 dB and the SNR of the second signal is varied to find the limit at which the two signals can be successfully separated. Signals are simulated 1000 times at each of various degrees of separation in order to collect reliable statistics. The simulations used to obtain Figure 4.4 are identical except that the first signal is fixed at 20 dB instead of 30 dB.

What we find is that RISR's dynamic range is comparable to root-MUSIC at high SNR, but it degrades when the SNR is lowered. Root-MUSIC's performance degrades more slowly, with each of the curves leaning over further to the right as SNR drops, whereas RISR remains almost as steep as at high SNR, but all the traces shift to the right. The reason that RISR's performance

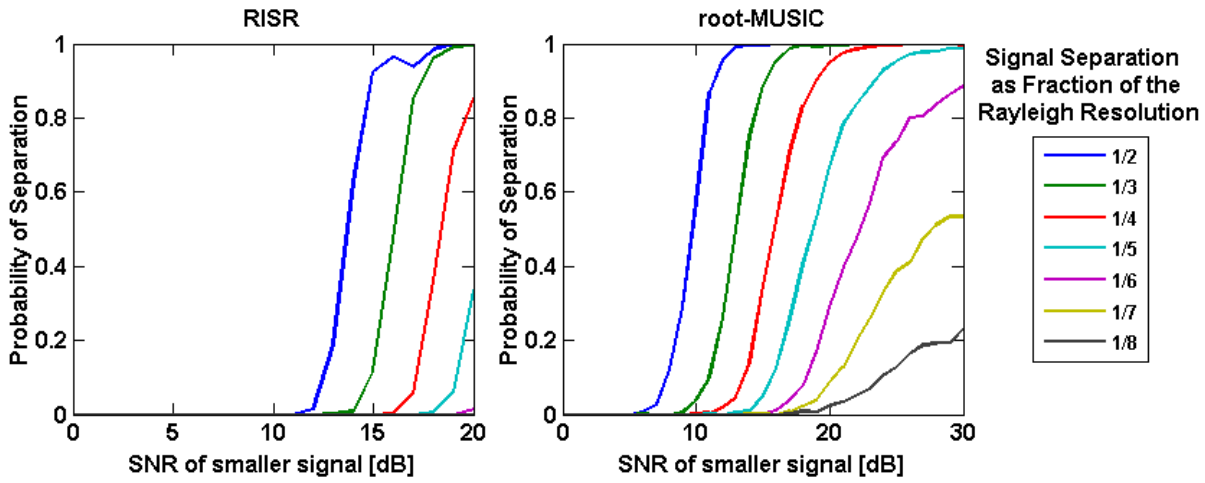

Figure 4.4: A comparison of RISR's and root-MUSIC's dynamic range for several separation values. The larger two signals is held at 20 dB, $N = 10$ antenna elements, $L = 10$ snapshots, and signals are uncorrelated. Performance is no longer comparable.

is affected more than root-MUSIC's when the SNR is reduced is because, as shown in Figure 4.2, RISR cannot perform at as low an SNR as root-MUSIC can.

4.4.4 Correlated Signals

Correlated signals, such as those that result from multipath, are commonly encountered in practice, so understanding how correlation affects DOA estimation is important. RISR produces slightly different results for highly correlated signals, but is indeed quite robust. A performance limit can be found by examining the worst case scenario - perfect correlation. As shown in Figure 4.5, RISR is barely affected by signals that are correlated and out of phase, but some resolution is lost when signals are correlated and in phase. Realistically, signals will never be perfectly correlated, nor are they likely ever to be perfectly in phase even when highly correlated, so Fig 4.5 likely represents a lower limit on performance.

4.4.5 Sample Support

One of the defining characteristics of RISR is that it requires only one spatial snapshot. Most DOA estimation techniques require estimating a sample covariance matrix, which requires at least

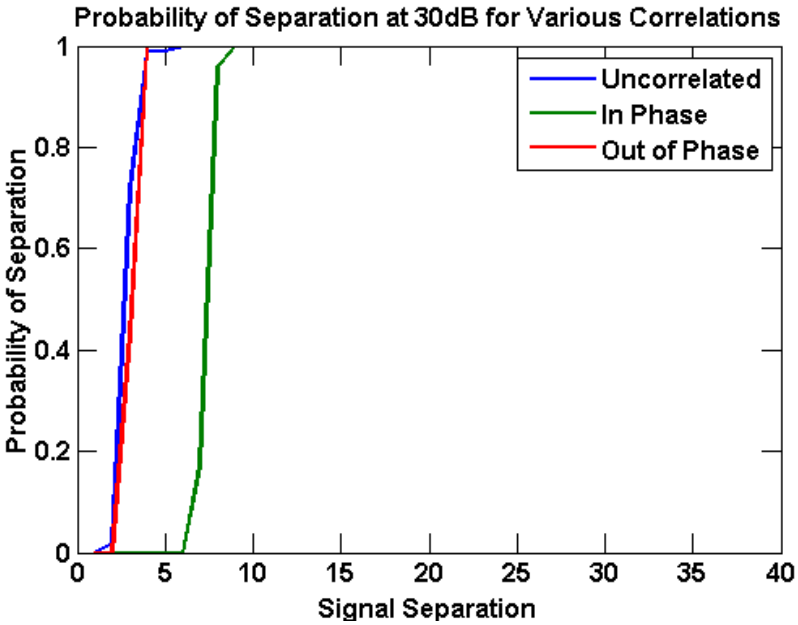

Figure 4.5: The probability of separation as a function of signal separation for the uncorrelated, correlated and in phase, and correlated and out of phase scenarios. SNR is fixed at 30 dB.

as many samples as antenna elements, and to obtain a good estimate many more samples are needed. In contrast, RISR can provide a good estimate with relatively few snapshots. Figure 4.6 is a single anecdotal result showing the convergence of RISR for various degrees of sample support. Figure 4.7 shows how sample support affects the ability of RISR to separate signals at fixed SNR as a function of their separation distance, and Fig 4.8 shows how sample support affects the probability of separation as function of SNR with signal separation fixed at a fraction of the Rayleigh Resolution.

What we can learn from these figures is that there is significant improvement for the first few snapshots beyond one, but there is very little left to be gained beyond about N snapshots. Furthermore, the performance difference is exaggerated at lower SNRs.

4.4.6 Calibration Error

RISR's robustness to calibration error will only be covered briefly as it is documented elsewhere [30]. RISR is robust to calibration error because of the way the calibration error variance term σ_z^2 is incorporated into the signal model given in 4.3. Calibration error can be especially troublesome

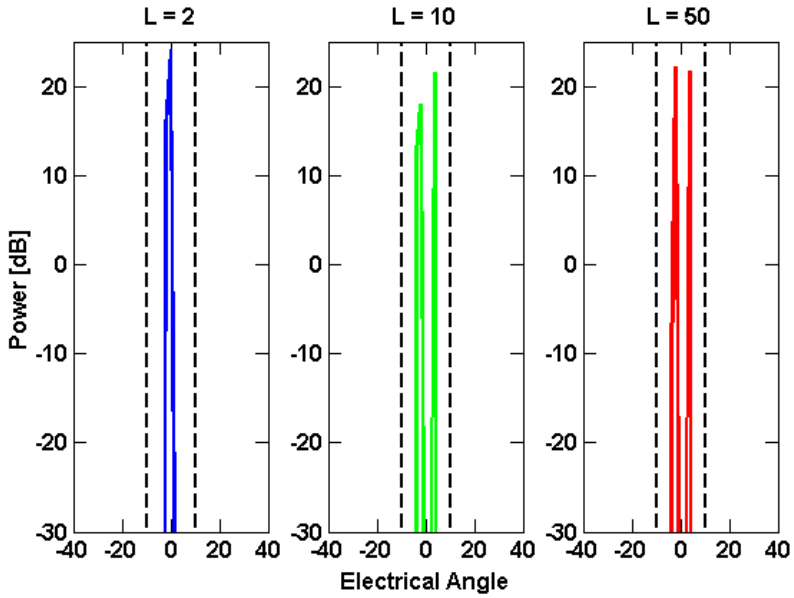

Figure 4.6: An anecdotal case showing how the estimate RISR produces can be affected by changes in sample support. 20 dB uncorrelated signals are present at $\pm 10^{\circ}$, $N = 10$ elements and $L = 10$ snapshots are used.

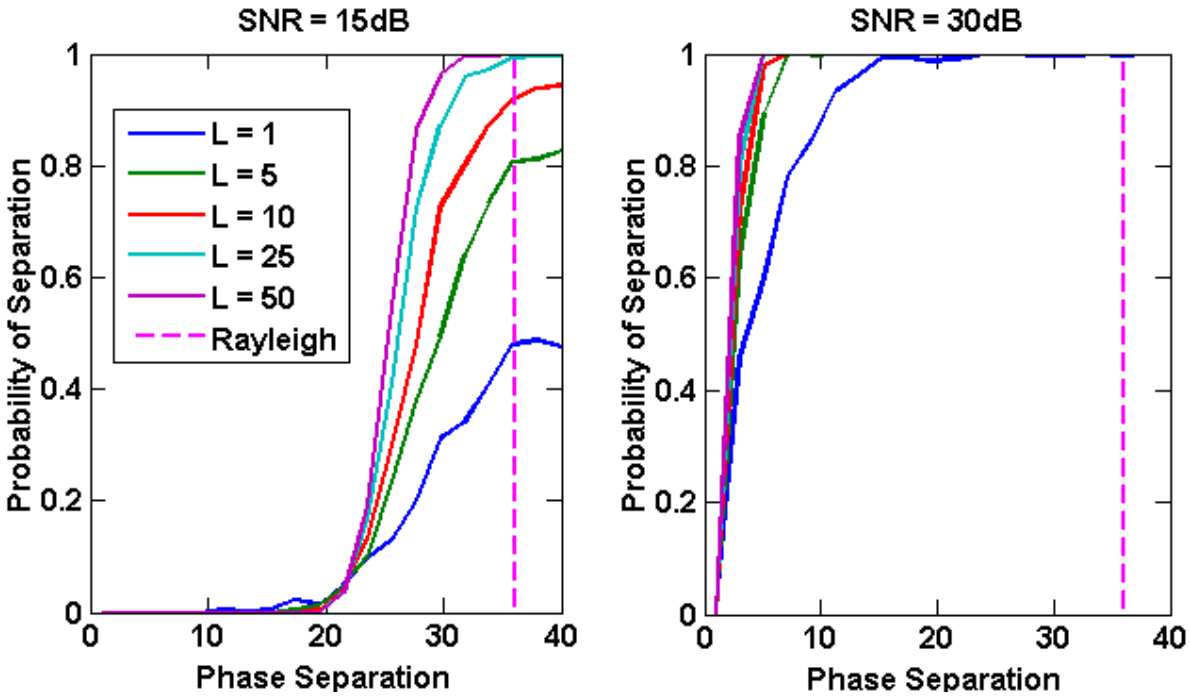

Figure 4.7: RISR's probability of separation as a function of signal separation, with SNR fixed at 15 dB (left) and 30 dB (right) for various degrees of sample support L.

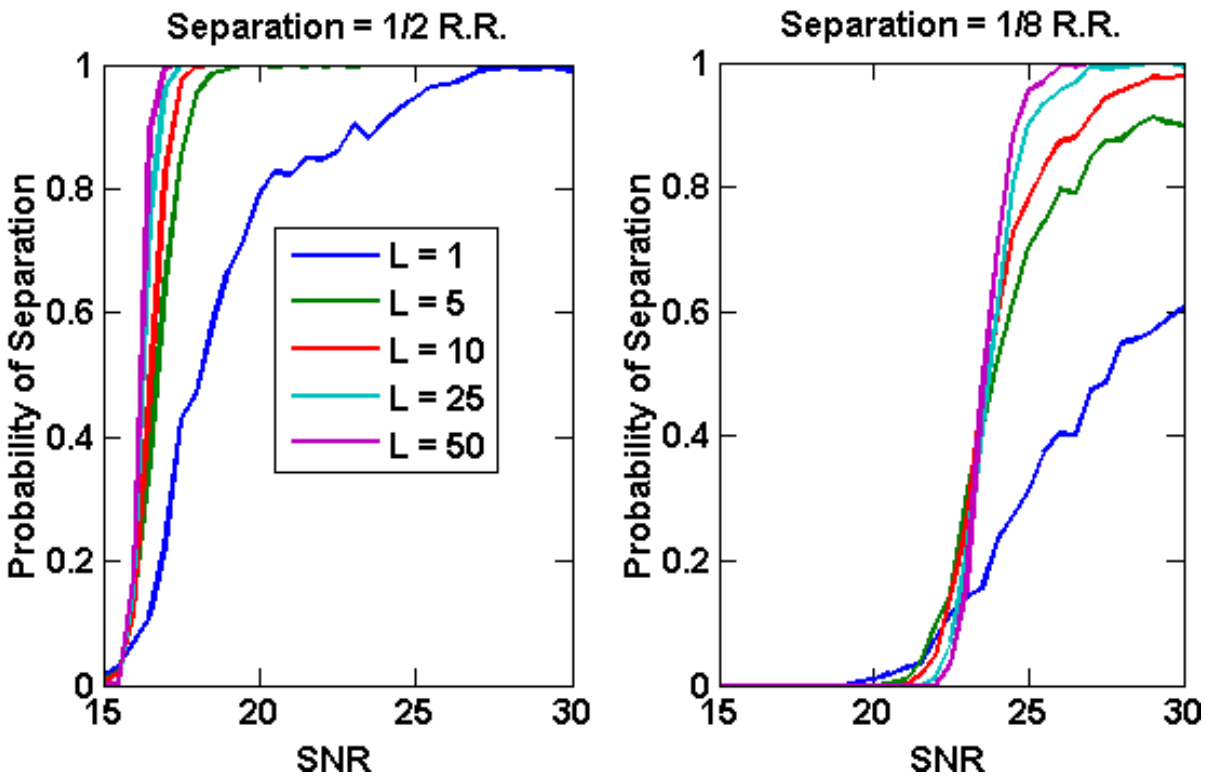

Figure 4.8: RISR's probability of separation as a function of SNR, with signal separation fixed at 1/2 (left) and 1/8 (right) of the Rayleigh resolution.

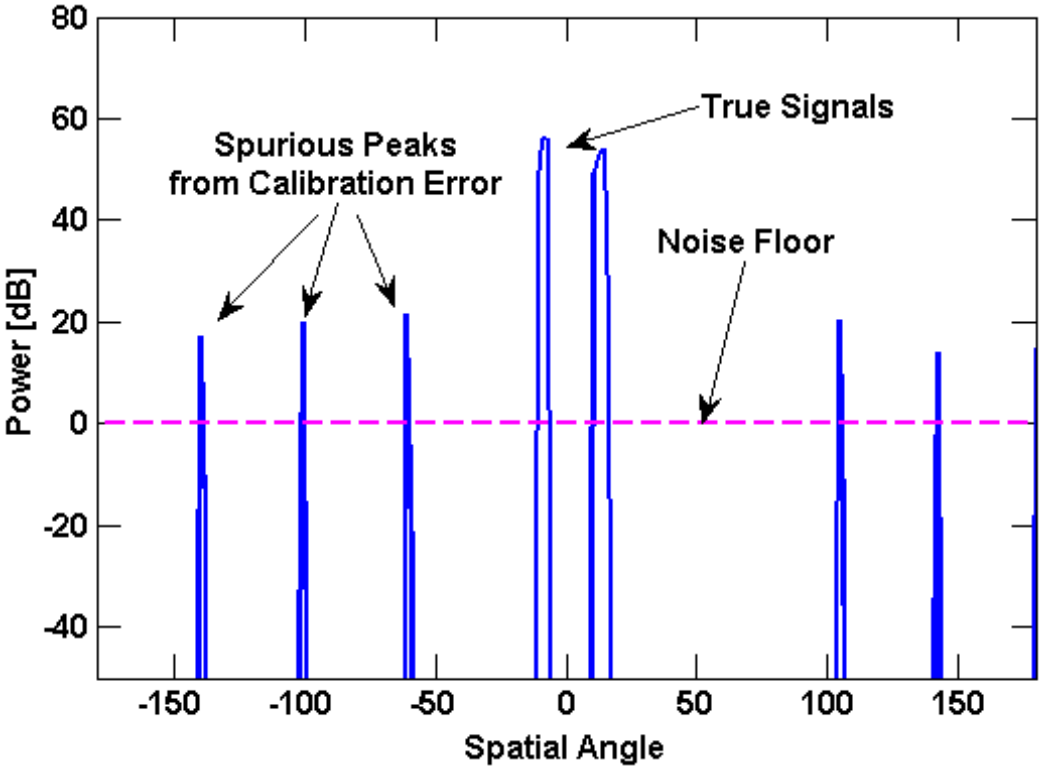

Figure 4.9: Spurious peaks caused by calibration error at a high SNR of 50dB.

because unlike noise which has an additive effect, calibration error has a multiplicative effect that can result in spurious peaks. This multiplicative effect can be seen in the signal model in (4.3). Typically having a high SNR can only improve the performance of a system, but if calibration errors are present, they actually become more disruptive as SNR increases because of their multiplicative effect. Figure 4.9 shows an example of the spurious peaks that result from calibration error.

Chapter 5 will be focused primarily on the low SNR regime where calibration errors do not have as large of an effect. For that reason, we deem it reasonable to omit a detailed treatment of calibration error's effects on RISR.

4.4.7 Computational Cost

The computational cost of RISR for one iteration is $O(MN^2)$ for $L < 2N$ and $O(MNL)$ otherwise, where N is the number of antennas, M is the discretization in angle relative to the number of

antenna elements, typically a multiple of N on the order of 20, and L is the number of snapshots used [30].

For comparison, the computational cost of MUSIC is $O(LN^2)$, which is noticeably less than that of RISR. Furthermore, RISR requires multiple iterations, so the actual computational cost will be about 10 times higher than the value given above. We have shown that it is possible to implement RISR in real time for modest array size, though large arrays remain limited to offline processing.

Chapter 5

Partially Constrained RISR (PC-RISR)

5.1 Genesis of PC-RISR

In this chapter the Partially Constrained Reiterative Super-resolution (PC-RISR) algorithm is derived and characterized. As its name implies, PC-RISR is a modification of the original RISR. By adding a constraint and a variable α to control the degree to which that constraint is enforced, PC-RISR can be made even more robust and than its forerunner.

The building blocks of PC-RISR are the original RISR algorithm, a gain constrained version of RISR (GC-RISR), and a geometric weighting factor α. Before constructing PC-RISR we will derive GC-RISR and briefly examine its performance. We find that GC-RISR successfully eliminates spurious peaks, but results in some loss of super-resolution, producing a spatial spectrum with wider rounded peaks.

If allowed to run for enough iterations (usually on the order of 20) RISR will reduce all sources to a single point, and drive the entire remainder of the spectrum to the lowest value supported by the processor being used. This can be desirable in some applications, such as radar where objects in the far field can be closely approximated as point targets. However there are also applications where sources should not be modeled as points, such as in medical imaging where sources are distributed volumetrically.

PC-RISR exhibits two operating regimes, one in which all sources are resolved to points as in RISR, and one in which sources are modeled with some width. PC-RISR was originally conceived as an attempt to model distributed sources by combining GC-RISR's spectral shape with RISR's super-resolution to create an algorithm with an adjustable degree of resolution, so early focus was on the latter regime, however it slowly became clear during the process of characterizing PC-RISR that its real utility lies in modeling point sources. PC-RISR operating in the point solution regime has been found to have super-resolution capability in excess of RISR and the ability to operate at lower SNR values with lower sample support.

5.2 Derivation of Gain-Constrained RISR

A gain constraint can be applied to a reiterative minimum mean squared error (RMMSE) based technique known as adaptive pulse compression (APC) in order to improve its performance in the presence of calibration errors [28][33]. The same gain constraint can be applied to RISR since it is also based on RMMSE. In RISR, the calibration errors translate into the spurious peaks shown in Figure 4.9. RISR already has built in robustness to model errors because it incorporates a model error noise term, but a gain constraint can be added for increased robustness.

The data model used in GC-RISR is identical to RISR's, given in Equation 4.3. The sole difference in the derivation is that the cost function used in GC-RISR includes a Lagrange multiplier used to enforce a unit gain constraint. To find the GC-RISR filter, minimize the cost function

$$J = \mathrm{E}\left[|x_m - \hat{x}_m|^2\right] \tag{5.1}$$

subject to the constraint

$$\mathbf{w}_m^{\mathrm{H}} \mathbf{s}_m = 1. \tag{5.2}$$

where $\mathbf{w}_m$ and $\mathbf{s}_m$ are the filter and steering vector for the m^{th} direction. Using a Lagrange multiplier

to incorporate the constraint direction into the cost function gives

$$\mathrm{J} = \mathrm{E}\left[|x_m - \hat{x}_m|^2\right] + \lambda^*(\mathbf{w}_m^{\mathrm{H}}\mathbf{s}_m - 1). \tag{5.3}$$

As before, take the gradient with respect to $\mathbf{w}_m^{\mathrm{H}}$ and set the result equal to zero. Because the cost function is quadratic, the global minimum is the only extrema. The gradient of the first term, $\mathrm{E}\left[|x_m - \hat{x}_m|^2\right]$, will lead to the same result given in (4.14). The gradient of the second term will simplify to $\lambda^*\mathbf{s}_m$. The result is then

$$\bigtriangledown \mathrm{J} = -p_m\mathbf{s}_m + [\mathbf{S}\mathbf{P}\mathbf{S}^{\mathrm{H}} + \mathbf{R} + \mathbf{R}_{\mathbf{z}}]\mathbf{w}_m + \lambda^*\mathbf{s}_m \equiv \mathbf{0}. \tag{5.4}$$

Solving this equation for $\mathbf{w}_m$ yields a result similar to the filter for RISR

$$\mathbf{w}_m = (\mathbf{S}\mathbf{P}\mathbf{S}^{\mathrm{H}} + \mathbf{R} + \mathbf{R}_{\mathbf{z}})^{-1}(p_m\mathbf{s}_m - \lambda^*\mathbf{s}_m). \tag{5.5}$$

From here, the original constraint given in Equation 5.2 must be used to solve the dual for λ^* so that $\mathbf{w}_m$ can be fully specified. Multiplying both sides of (5.5) by $\mathbf{s}_m^{\mathrm{H}}$ gives

$$\mathbf{s}_m^{\mathrm{H}}\mathbf{w}_m = p_m\mathbf{s}_m^{\mathrm{H}}(\mathbf{S}\mathbf{P}\mathbf{S}^{\mathrm{H}} + \mathbf{R} + \mathbf{R}_{\mathbf{z}})^{-1}\mathbf{s}_m - \lambda^*\mathbf{s}_m^{\mathrm{H}}(\mathbf{S}\mathbf{P}\mathbf{S}^{\mathrm{H}} + \mathbf{R} + \mathbf{R}_{\mathbf{z}})^{-1}\mathbf{s}_m. \tag{5.6}$$

Solving for λ^* then yields

$$\lambda^* = p_m - \frac{1}{\mathbf{s}_m^{\mathrm{H}}(\mathbf{S}\mathbf{P}\mathbf{S}^{\mathrm{H}} + \mathbf{R} + \mathbf{R}_{\mathbf{z}})^{-1}\mathbf{s}_m}. \tag{5.7}$$

Finally, insert (5.7) into (5.5) results in the m^{th} gain constrained RISR (GC-RISR) filter

$$\mathbf{w}_{\mathrm{GC},m} = \frac{(\mathbf{S}\mathbf{P}\mathbf{S}^{\mathrm{H}} + \mathbf{R} + \mathbf{R}_{\mathbf{z}})^{-1}}{\mathbf{s}_m^{\mathrm{H}}(\mathbf{S}\mathbf{P}\mathbf{S}^{\mathrm{H}} + \mathbf{R} + \mathbf{R}_{\mathbf{z}})^{-1}\mathbf{s}_m}\mathbf{s}_m. \tag{5.8}$$

This solution is quite similar to the result for unconstrained RISR. The difference is the normalizing factor in the denominator. Apart from the filter, implementation of GC-RISR is otherwise identical

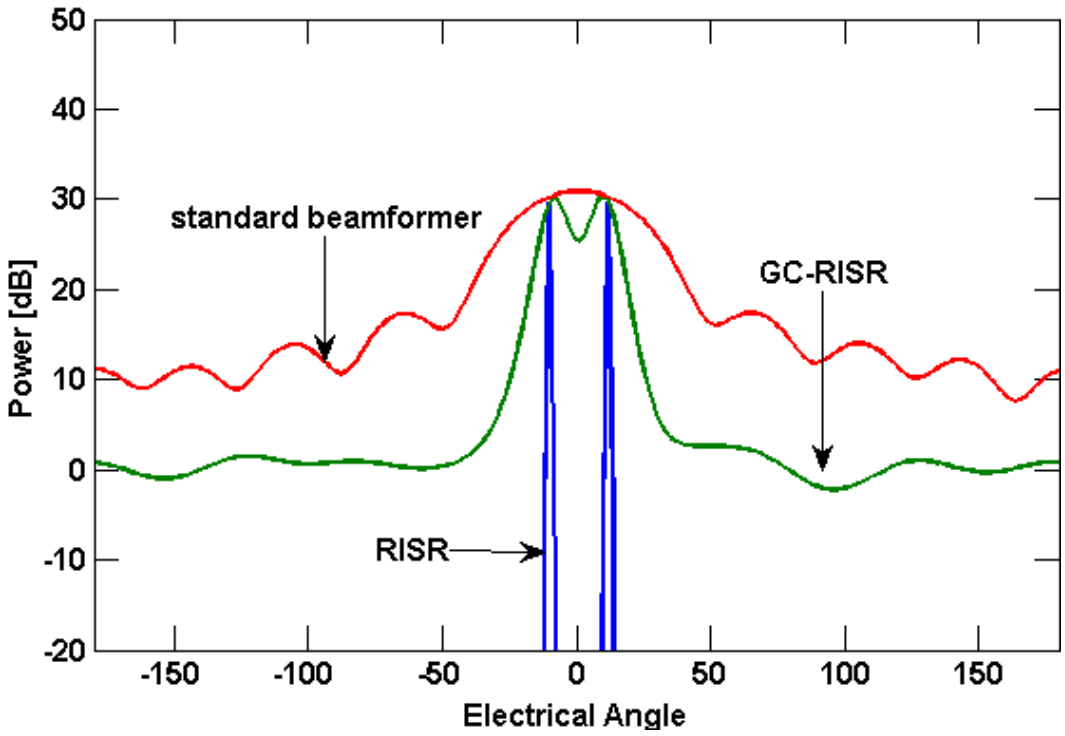

Figure 5.1: Comparison of the estimate produced by RISR, GC-RISR, and the standard beamformer. GC-RISR has more resolution than the standard beamformer, but less than RISR. Two 30 dB signals are present at $\pm 10^{\circ}$.

to that of RISR.

Figure 5.1 shows how an estimate produced with GC-RISR compares to one produced with RISR and the standard beamformer. The most important difference is that the super-resolution factor is reduced compared to RISR, but is still super-resolved. Another feature worth noting is that GC-RISR naturally levels off at the noise floor, while RISR does not. A consequence is that some detectors that work well for RISR may not be appropriate for GC-RISR.

5.3 Formulation of PC-RISR

Gain Constrained RISR was successful in adding robustness against model mismatch, but it also resulted in sacrificing some of RISR's super-resolution ability. Partially constrained RISR (PC-RISR) is a compromise that lies somewhere between RISR and GC-RISR, and it is made possible by how similar the filter construction for RISR and GC-RISR are. To simplify notation, denote

$\mathbf{D} = (\mathbf{SPS}^{\mathrm{H}} + \mathbf{R} + \mathbf{R_z})^{-1}$. The two filters then become

$$\mathbf{w}_{\mathrm{GC},m} = \frac{1}{\mathbf{s}_m^{\mathrm{H}}\mathbf{D}^{-1}\mathbf{s}_m}\mathbf{D}^{-1}\mathbf{s}_m \tag{5.9}$$

$$\mathbf{w}_{\mathrm{RISR},m} = p_m\mathbf{D}^{-1}\mathbf{s}_m \tag{5.10}$$

Note that both filters contain $\mathbf{D}^{-1}$ term, which means that the cost of computing (5.10) is insignificant after computing (5.9). It is then possible to formulate the PC-RISR filter by combining the RISR and GC-RISR filters geometrically using a weighting factor α.

$$\mathbf{w}_{\mathrm{PC},m} = \left[\left(\frac{1}{\mathbf{s}_m^{\mathrm{H}}\mathbf{D}^{-1}\mathbf{s}_m}\right)^{\alpha}(p_m)^{1-\alpha}\right]\mathbf{D}^{-1}\mathbf{s}_m. \tag{5.11}$$

The weighting factor can be chosen to bias the PC-RISR filter toward either the constrained or unconstrained solution. It can have any value between 0 and 1, with 0 giving the same result as RISR and 1 being identical to GC-RISR. In fact, it is possible to think of RISR and GC-RISR as special cases of PC-RISR.

5.4 Guide to Implementing PC-RISR

Implementation of the PC-RISR algorithm is summarized below. With the exception of the filter bank, it is identical to the that of RISR and GC-RISR given previously. Initialization is performed once L spatial snapshots have been collected and arranged in a matrix $\mathbf{Y}$.

1. Initialize:

   - Find initial estimate of complex amplitude distribution:

$$\hat{\mathbf{X}}_0 = \mathbf{S}^{\mathrm{H}}\mathbf{Y}$$

- Find initial estimate of average spatial power distribution:

$$\bar{\mathbf{P}}_0 = [(1/L)\sum_{\tau=0}^{L-1} \hat{\mathbf{x}}_0(l+\tau)\hat{\mathbf{x}}_0^{\mathrm{H}}(l+\tau)] \odot \mathbf{I}_{M\times M}$$

- By assumption or measurement, obtain noise only covariance matrix $\mathbf{R}$
- Based on array tolerances, obtain the model error variance σ_z^2
- Based on sample support and computational resources, choose $0 \leq \alpha \leq 1$

2. Iterate: for $i = 1, 2, ...$

- Calculate the PC-RISR filter bank $\bar{\mathbf{W}}_i$ whose columns M columns are:

$$\bar{\mathbf{w}}_{\mathrm{PC},m,i} = \left[\left(\frac{1}{\mathbf{s}_m^{\mathrm{H}}\mathbf{D}^{-1}\mathbf{s}_m}\right)^{\alpha}(p_m)^{1-\alpha}\right]\mathbf{D}^{-1}\mathbf{s}_m$$

- Update estimate of spatial complex amplitudes:

$$\hat{\mathbf{X}}_i = \bar{\mathbf{W}}_i^{\mathrm{H}}\mathbf{Y}$$

- Update estimate of spatial power distribution:

$$\bar{\mathbf{P}}_i = [(1/L)\sum_{\tau=0}^{L-1} \hat{\mathbf{x}}_i(l+\tau)\hat{\mathbf{x}}_i^{\mathrm{H}}(l+\tau)] \odot \mathbf{I}_{M\times M}$$

3. Once iteration is completed, the PC-RISR spatial magnitude distribution can be obtained as

$$x_m = \sqrt{\bar{\mathbf{P}}_{mm}}.$$

4. The number of sources and their magnitude can be estimated from the peaks in the RISR spectrum.

5.5 Performance Characteristics

In this section the performance of PC-RISR will be characterized via simulations and the results compared to RISR, GC-RISR, and root-MUSIC. Special attention is given to convergence because it is more complicated than RISR and GC-RISR due to the incorporation of the geometric weighting term α. It will be shown in our examination of PC-RISR's resolution that it has good performance at low SNR, so the focus in subsequent simulations will be on the low to middle SNR regimes where PC-RISR is expected to have the most utility. As a direct result of the ability to operate at low SNR, we will show that PC-RISR also has improved dynamic range. In addition, PC-RISR's performance will be analyzed for the cases of correlated signals and varied sample support. Finally, computational cost will be discussed.

5.5.1 Convergence

The way in which PC-RISR converges is perhaps the most important difference between it and previous versions of RISR, including GC-RISR. PC-RISR contains a geometric average weighting factor α that allows for PC-RISR's convergence to be controlled, creating a new dimension which PC-RISR can be tuned in. Here we outline the effect α has on convergence speed.

First, we examine the convergence speed of PC-RISR as a function of α. The results in Figure 5.2 were obtained by generating simulated received data according to (4.3) and then running PC-RISR on that data using various values for α and increasing the number of iterations until each converged. To ensure all versions of PC-RISR would be able to detect the signals, they were set at to an SNR of 30 dB (after coherent integration) and placed at $\pm 10^{\circ}$. Estimates were considered to be converged when the mean squared error of subsequent spatial power estimates fell below 0.001. This process is repeated 10 times to find an average. The number of iterations at which PC-RISR converges is very consistent for fixed α and signal separation, so even just 10 repetitions is sufficient to generate reliable statistics. A ULA with $N = 10$ antenna elements is used with $L = 10$ snapshots, magnitude and phase calibration errors of 1% are included, and σ_z^2 is chosen to

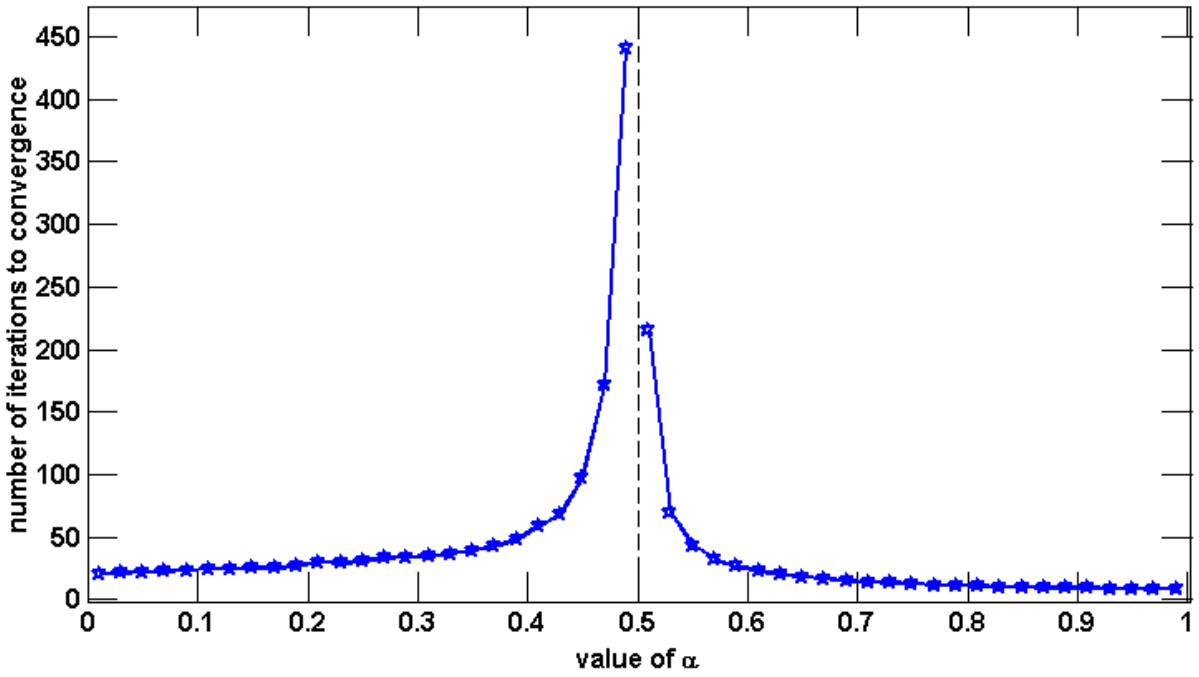

Figure 5.2: Convergence speed of PC-RISR as function of α.

be 1.5×10^{-3}.

As can been seen in Figure 5.2, the number of iterations required initially increases slowly and nearly linearly with α, however as α approaches 0.4 the relationship quickly becomes exponential. At first glance, Figure 5.2 may appear to be symmetric, but it is not. The right half is steeper near $\alpha = 0.5$ and quickly levels off as α increases. The plot is shown piece wise because convergence could not be achieved at $\alpha = 0.5$. It also needs to be stated that convergence speed will be affected by SNR, the number of signals, and their spacing, so Figure 5.2 should be considered a rough guideline.

The relationship between alpha and convergence speed is born out of the geometric mean formed in (5.11). When alpha is close to either 0 or 1, RISR or GC-RISR quickly dominates the other and a solution is reached in just a few iterations. Conversely, when α is close to 0.5, it takes many more iterations to converge.

PC-RISR's performance can be divided into two regimes based on α. The first regime, the point solution regime, corresponds to $\alpha < 0.5$ and will always reach a point solution if allowed to converge completely. The second regime, corresponding to $\alpha > 0.5$, will still be super-resolved

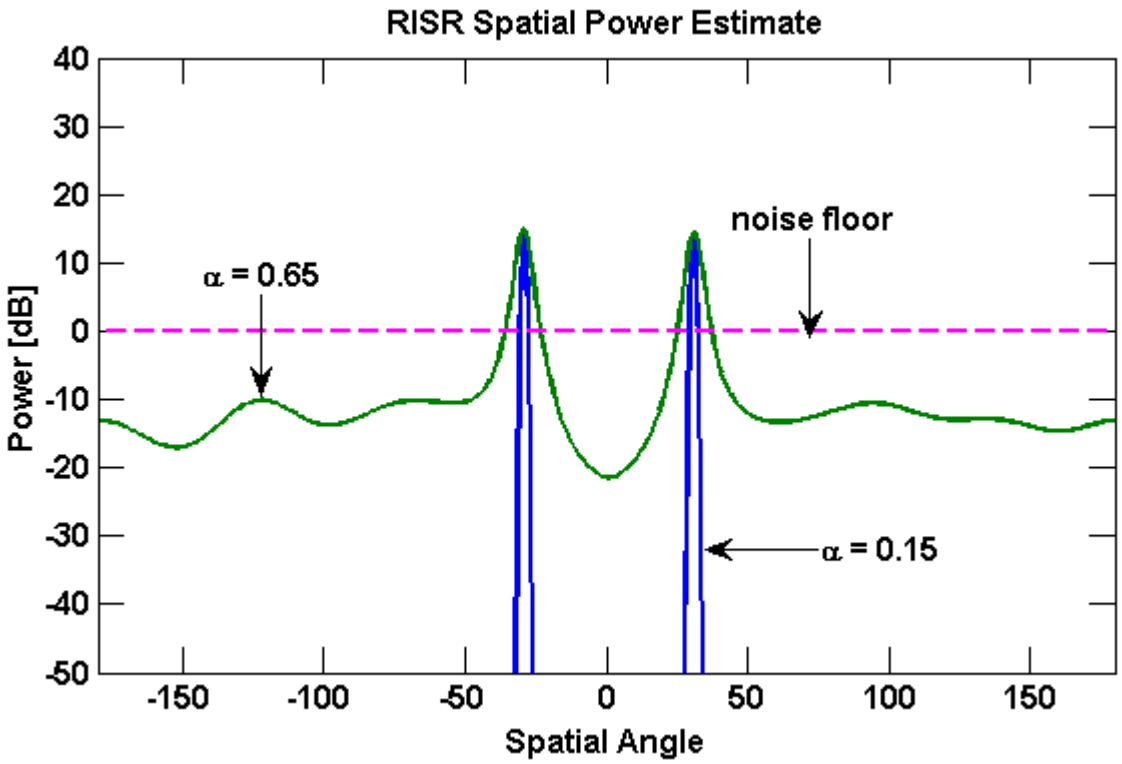

Figure 5.3: An illustration of PC-RISR's two regimes. If $\alpha < 0.5$ a point solution is reached, if $\alpha > 0.5$ peaks are rounded and a natural noise floor exists

but will not converge to a point solution, and the spatial power estimate will in fact resemble that of GC-RISR, albeit with sharper peaks and deeper nulls. Figure 5.3 shows an example of how the same signal processed with α in each regime produces significantly different results.

Because each regime is so different, they are suited to different applications and require separate processes to ascertain the number and nature of the incident signals. As such, they merit separate treatment. The general behavior of PC-RISR for several ranges of α are examined here and each regime is discussed in detail in the following sections.

5.5.2 Point Solution Regime

At each iteration, PC-RISR with $\alpha < 0.5$ tries to suppress the portions of the spatial spectrum that it does not identify as a signal. Given enough iterations, all points except the signals will be suppressed and the fully converged PC-RISR spectrum in the point solution regime will resemble

Point Solution Regime				
	$0 < \alpha < 0.2$	$0.2 < \alpha < 0.3$	$0.3 < \alpha < 0.4$	$0.4 < \alpha < 0.5$
Required Iterations	10 - 15	15 - 25	25 - 50	50 - 300
Utility	Suitable for mid to high SNRs	Suitable for mid SNRs	Suitable for low SNRs	Limit of low SNR performance
Behavior	Low SNR signals may be suppressed. Appreciable improvement over RISR for small cost	Super-resolution of signals with an SNR on the order of 15dB Comparable performance to root-MUSIC	Super-resolution of signals with an SNR on the order of 10dB Performs well even with low sample support	Allows resolving of closely spaced signals at extremely low SNR High sample support required to prevent spurious peaks

Table 5.1: Summary of the utility and behavior of α in the point solution regime

Constraint Dominant Regime			
	$0.5 < \alpha < 5.2$	$5.2 < \alpha < 0.55$	$0.55 < \alpha < 1.0$
Required Iterations	300 - 50	50 - 15	15 - 5
Utility	Suitable for low SNR DOA estimation	Suitable for modeling small distributed sources	Suitable for modeling small distributed sources
Behavior	Sharp peaks and deep nulls. Similar to point solution regime, but less sharp. Underestimates signal power	Number of iterations to convergence decreases quickly. Natural floor develops below noise floor. Underestimates signal power	Natural floor rises toward noise floor with increasing α, and few iterations are required. Tendency to underestimate signal power decreases with larger α.

Table 5.2: Summary of the utility and behavior of α in the constraint dominant regime

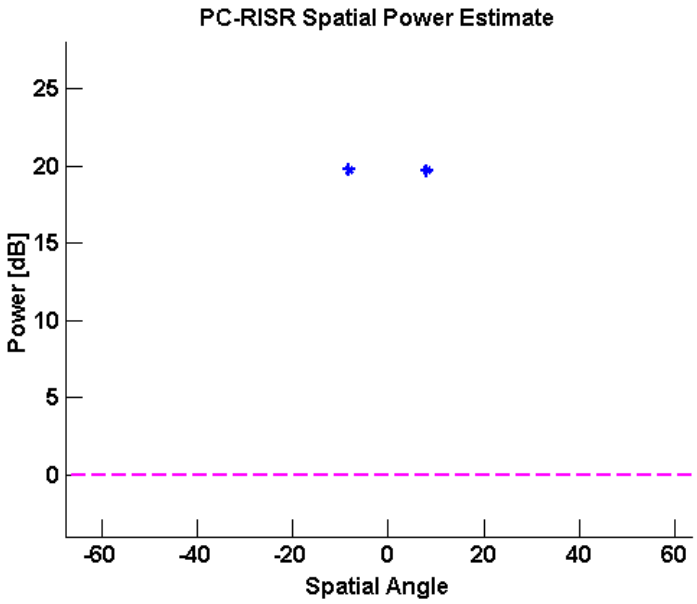

Figure 5.4: PC-RISR in the point solution regime suppresses all parts of the spatial spectrum not identified as a signal, leaving only points.

Figure 5.4. In fact, it could even be argued that the true output of PC-RISR's point solution regime is not a spatial spectrum estimate, but a list of points comprised of estimated DOA's and associated power estimates. In this respsect, PC-RISR bears similarity to root-MUSIC, which produces a vector of DOA estimates (without power estimates). With that said, we will treat the result of PC-RISR as a spatial power spectrum and all results presented (including the previous results) are halted just before reaching a true point solution so that the estimate retains some shape.

It should also be noted that in the point solution regime, PC-RISR often does not require as many iterations as Figure 5.2 suggests. If signals are spaced far apart (but still closer than the Rayleigh resolution), SNR is high, or sample support is abundant, PC-RISR can often provide a good estimate without fully converging. Figure 5.5 shows a case in which PC-RISR's estimate after 200 iterations provides no more information than it did after 100. There are many cases in which fewer iteration are needed than suggested by Figure 5.2.

At high SNR (greather than about 15 dB), one finds that all versions of PC-RISR in the point source regime converge to a similar solution. However if SNR is reduced, versions with a lower value of α will detect only one signal instead of two, and if SNR is lowered even further they will

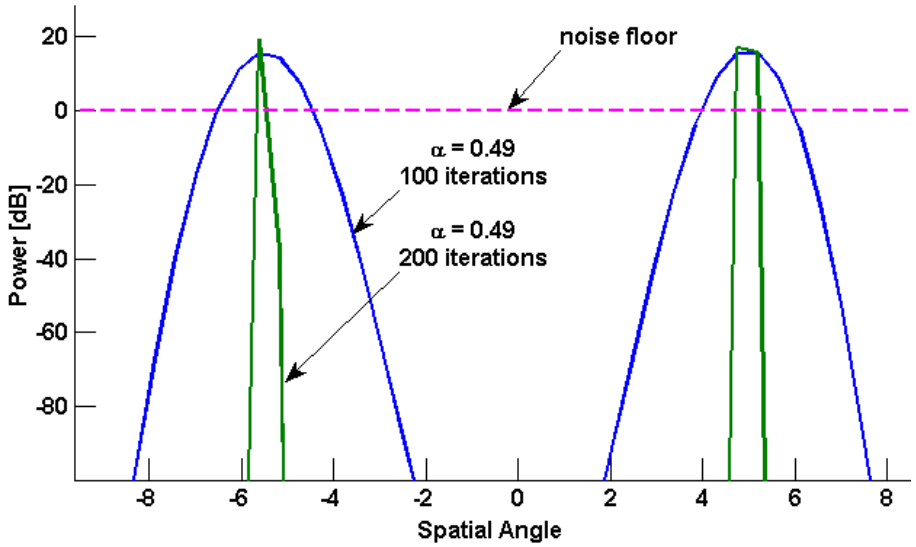

Figure 5.5: PC-RISR often provides a good estimate even before fully converging. In this simulation two 20 dB signals were presnt at 5° and -5°.

suppress the signal. This is demonstrated in Figure 5.6.

On the other hand, for larger α the possibility of misinterpreting noise as a signal and amplifying it increases. The probability of this phenomenon in inversely related to sample support. As an example, for $N = L$ spurious peaks becomes a concern for $\alpha > 0.4$ and result in false detections that limit the performance of $\alpha > 0.4$. Figure 5.7 shows an example of the false peaks that can occur when using α greater than 0.4 with low sample support. When sample support is plentiful, noncoherent integration suppresses spurious peaks, resulting in the ability to use all values of α without the concern of spurious peaks.

5.5.2.1 Limit of Low SNR Performance

The most obvious application of PC-RISR operating in the point solution regime is in low SNR super-resolved DOA estimation. Raising α slows down the convergence of PC-RISR, which helps it to detect signals that RISR misses. The simulation results given below characterize the ability of PC-RISR to resolve signals at some fixed separation as a function of SNR, with the purpose of probing the lower limit of PC-RISR's ability to super-resolve. The question of how much super-resolution PC-RISR can provide will be discussed in the next section.

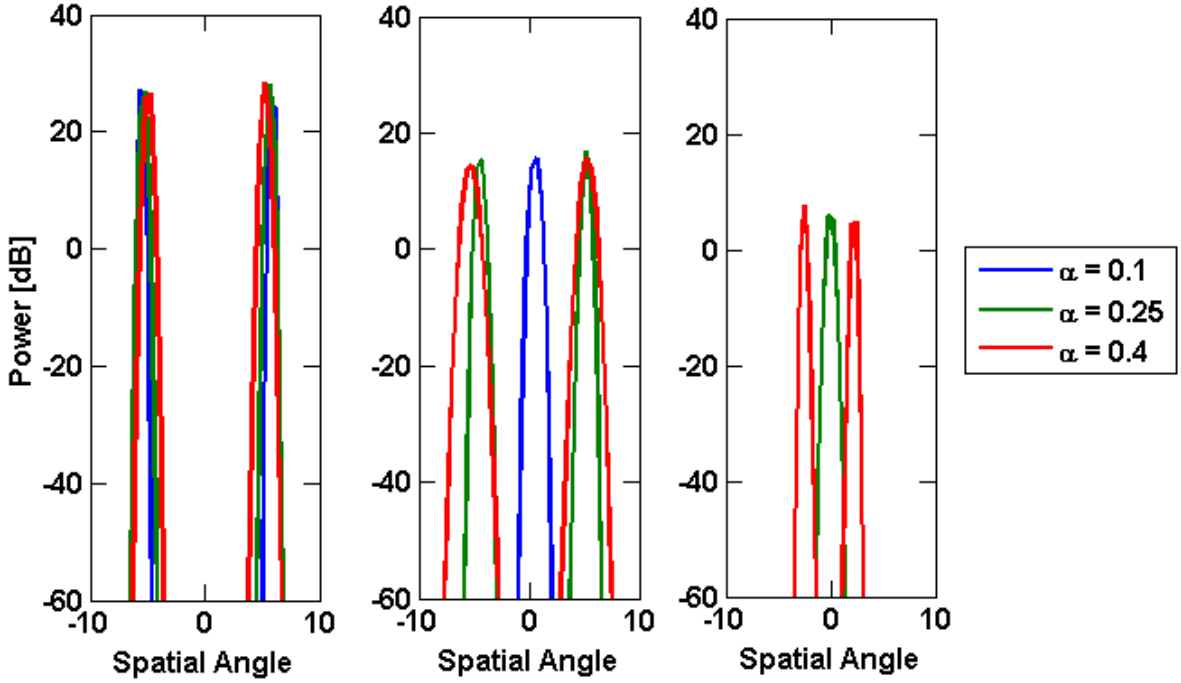

Figure 5.6: At low SNR lower values of α lose the ability to resolve both signals and can even fail to detect a signal at all. Two uncorrelated signals at $\pm 5^\circ$ at an SNR of 30 dB (left) and 15 dB (middle), and 10 dB (right) with $N = 10$ and $L = 10$.

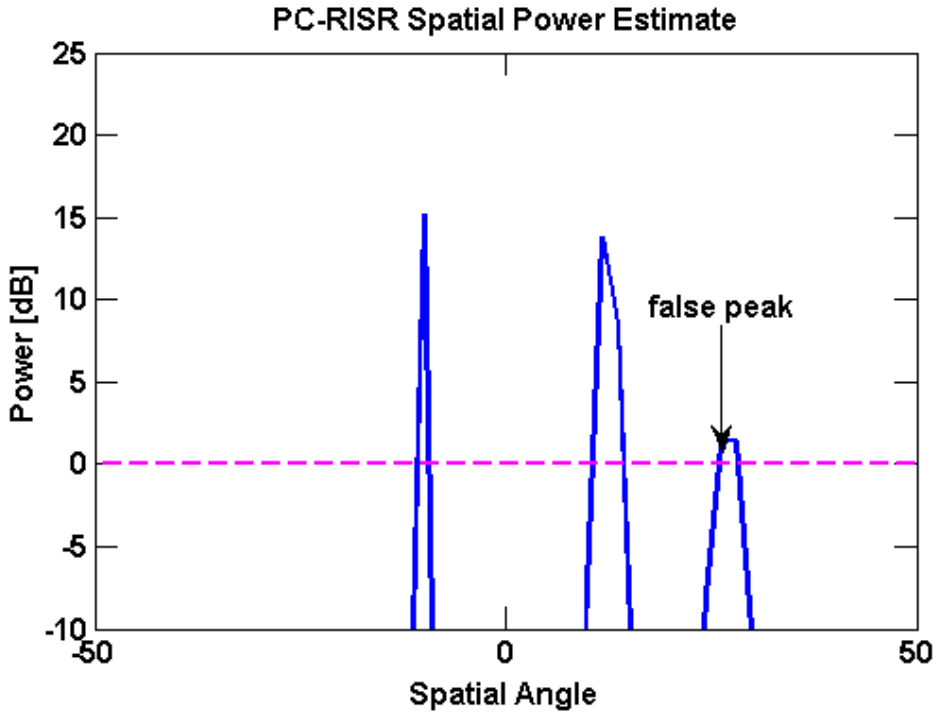

Figure 5.7: 800 iterations are run at an SNR of 15 dB with $\alpha = 0.499$ to illustrate how high values of α can result in spurious peaks above the noise floor. Signals are at $\pm 12^\circ$ and $N = L = 10$.

The following Monte Carlo results are obtained by performing 500 trials at each permutation of SNR and α, with signal separation fixed. A ULA with $N = 10$ antenna elements and $L = 10$ time samples were employed, signals are random and uncorrelated, and σ_z^2 is chosen to be 1.5×10^{-3}. Two signals of equal power are incident on the array, the first at $\phi_1 = 0^\circ$ (boresite) and the second at some fixed angular distance ϕ_2, which is some fraction of the Rayleigh resolution. For all values of α 30 iterations are used, unless specified otherwise. Results from root-MUSIC are also included as a benchmark. Root-MUSIC was implemented using forward-backward averaging to effectively double its sample support, so it is essentially operating on twice as many samples as PC-RISR [16]. Spatial smoothing was not used with root-MUSIC because signals are uncorrelated [34].

The detector used is rather simple. After iteration completes (which may not be the same as full convergence for large α) all data below the noise floor (0 dB) is thrown away (set to 0 dB) and peak detection is performed. If exactly two peaks are found within half a beamwidth of their true locations, the signals are deemed separated. Otherwise that trial is deemed a failure.

Figure 5.8 shows the probability of separation as a function of SNR for signal separations of 1/5, 1/2, and 9/10 the Rayleigh resolution. Here, a probability of separation of 0.9 will be considered separable, though 0.5 is also commonly used as a basis for comparison. Using this criterion and examining the plots given in Figure 5.8, we see that signals spaced further apart can be resolved at a lower SNR than closely spaced signals. For signals at 1/5 of the Rayleigh resolution, an SNR of about 15 dB is required. Half the Rayleigh resolution only requires an SNR of about 7 dB, and for 9/10 of the Rayleigh resolution, which is barely super-resolved, an SNR of only 5 dB is enough to separate signals with 90% probability. The reason that $\alpha = 0.45$ for 300 iterations never reaches a probability of separation of one is because $L = 10$ is not sufficient sample support to suppress spurious peaks for large α.

5.5.2.2 Super-Resolution Factor

In the last section we examined the lowest at which PC-RISR can super-resolve signals. In this section, we examine how much super-resolution PC-RISR can provide at a given SNR. We charac-

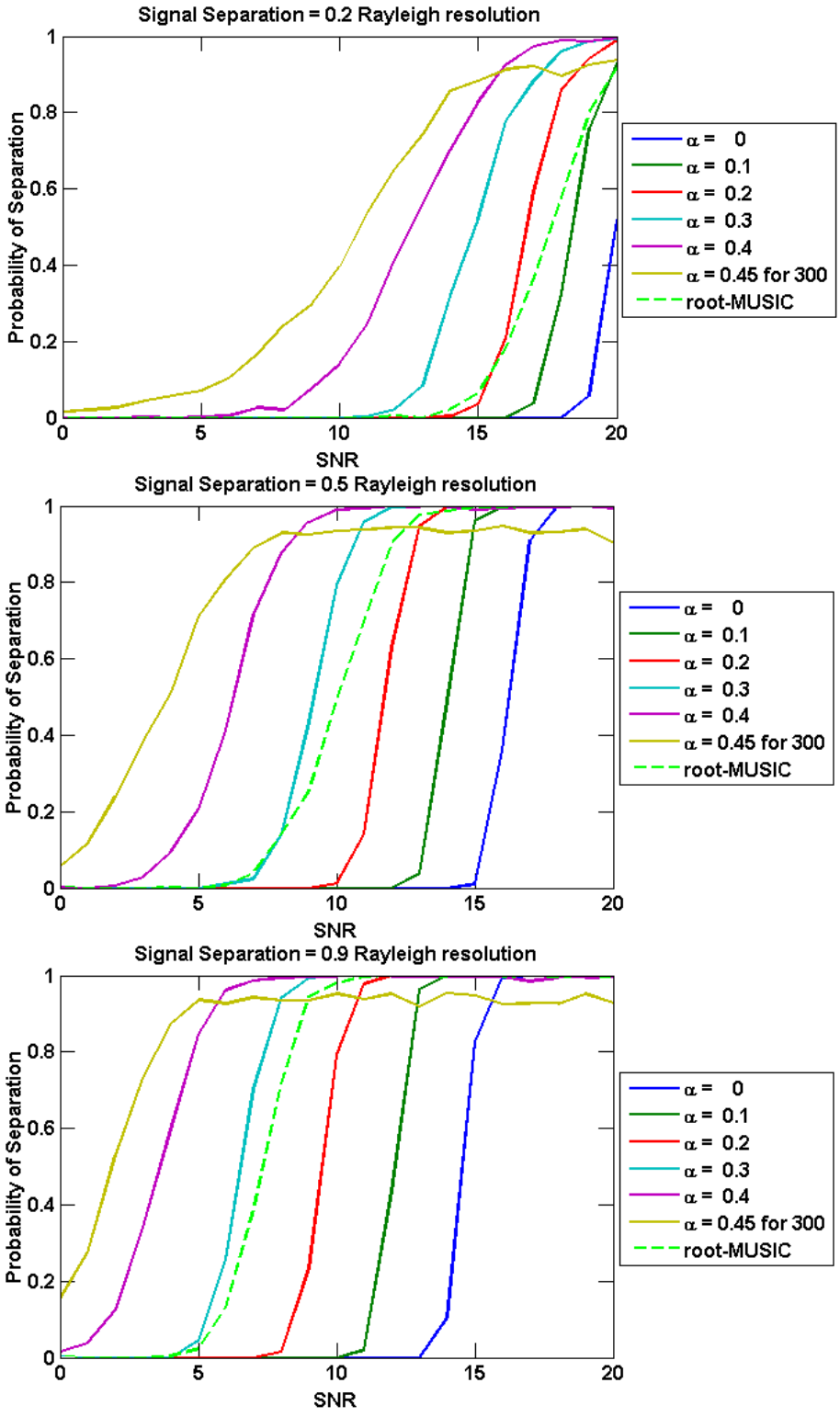

Figure 5.8: Probability of separation as a function of SNR with signal separation fixed, for several values of α in the point solution regime. The number of antenna elements is $N = 10$ and the number of snapshots is $L = 10$.

terize the super-resolution factor that PC-RISR achieves by using Monte Carlo simulations where the SNR is fixed and signal separation is allowed to vary. We find that, as with all super-resolution algorithms, the degree of super-resolution that can be achieved tends downward with SNR, so results will be given at several SNR values. The results here are procured in the same manner as outlined in the previous section, using the same values, with the sole difference being that SNR is fixed while separation is varied.

What we observe from the results in Figure 5.9 is that at high SNR, while $\alpha = 0.4$ gives the best results, all versions of PC-RISR give comparable estimates. Here, we determine the super-resolution factor with respect to a 90% chance of separation. Accordingly, PC-RISR's maximum super-resolution factor achieved at high SNR is about 7.2, which is appreciably better than root-MUSIC's 5.1 and RISR's 4.5.

The reason that $\alpha = 0.45$ fails to reach a 100% probability of separation in Figure 5.9 is because it occasionally fails to suppress spurious peaks above the noise floor, resulting in a failed separation. Increasing sample support would solve this problem, and with enough samples support optimal performance would be obtained using α close to 0.5. However, with $L = 10$, the values of α that can be used without risking spurious peaks are limited.

When the SNR is 15 dB as in Figure 5.10, the performance of unconstrained RISR significantly degrades. Root-MUSIC still functions well, but with limited sample support 15 dB is close to its limit. Notice that $\alpha = 0.3$ exhibits similar performance to root-MUSIC. There are a variety of factors involved, but one will often find that PC-RISR with α set between 0.2 and 0.3 results in similar performance to root-MUSIC. With the SNR reduced, PC-RISR ($\alpha = 0.4$), RISR, and root-MUSIC's super-resolution factors are 5.1, 2.9, and 1, respectively. PC-RISR provides the same degree of super-resolution at an SNR of 15 dB that root-MUSIC did at 20 dB.

In Figure 5.11 the SNR is lowered even further to 10 dB, which goes beyond unconstrained RISR's ability to detect a signal at all, so it no longer appears in the figure. Root-MUSIC and PC-RISR's performance is degraded somewhat, and $\alpha = 0.3$ continues to be a rough approximation of Root-MUSIC's performance. At this SNR all of the low α versions of PC-RISR either fail to

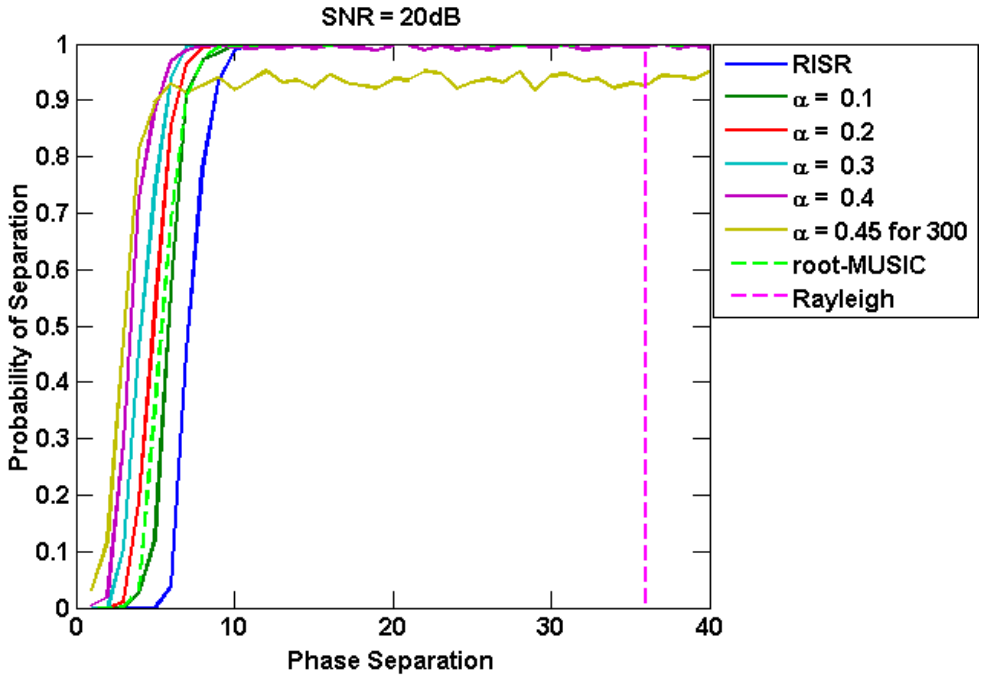

Figure 5.9: Probability of separation as a function of the two signals' phase difference. False peaks breaching the noise floor cause $\alpha = 0.45$ not to reach 100% chance of separation.

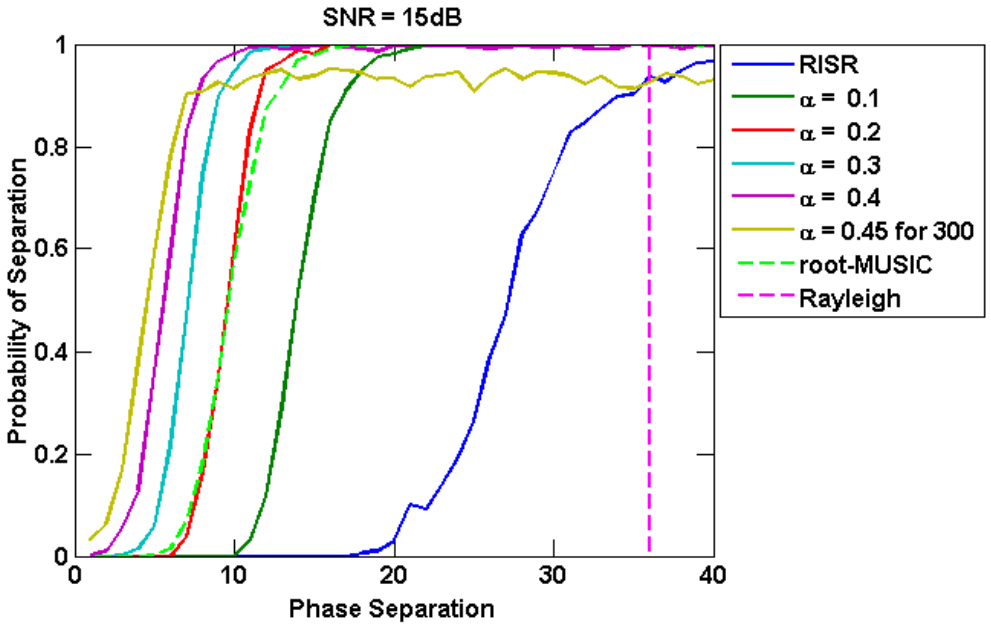

Figure 5.10: Probability of separation as a function of the two signals phase difference. PC-RISR with α set to 0.3 performs similarly to root-MUSIC.

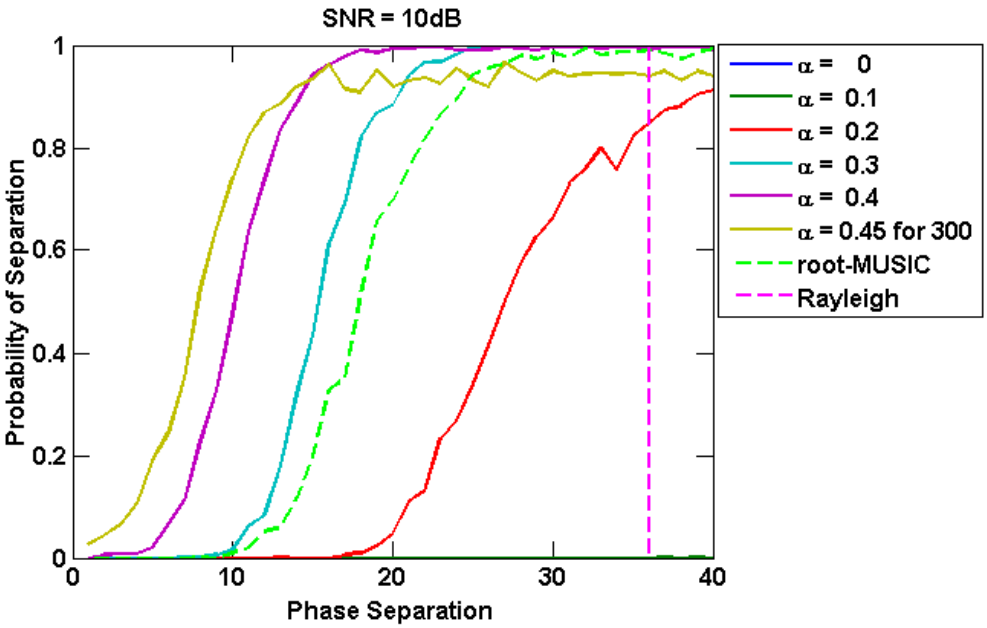

Figure 5.11: Probability of separation as a function of the two signals phase difference. At an SNR of 10 dB RISR is no longer able to detect a signal, but PC-RISR can separate them.

detect the signal at all or cannot provide any super-resolution. PC-RISR's super-resolution factor remains at about 3, and root-MUSIC's is reduced to 1.5.

If the SNR is pushed lower still, root-MUSIC fails to detect the signals, as do mid α range versions of PC-RISR, but high α exhibits some very interesting behavior. Previously, $\alpha = 0.45$ was better than $\alpha = 0.4$, although not by very much, but now it is significantly better. While high values of α require more sample support and carry a high computational cost because they require more iterations (see Figure 5.2), they are also considerably more robust to low SNR. Using $\alpha = 0.4$ results in a super-resolution factor of 1 (i.e. no super resolution), but $\alpha = 0.45$ manages 1.5.

Pushing the SNR down to 3 dB, PC-RISR finally loses its ability to super-resolve. As will be demonstrated later, PC-RISR's performance can be pushed even further by increasing sample support to unlock values of α close to 0.5.

Having characterized PC-RISR thus far, we felt that the simplistic detector we employed did not do justice to PC-RISR for $\alpha > 0.4$. While the simple solution would be to increase sample support, more could also be done with the detector, so we will present one final set of results that employ a different detector. The detector used for the following results is in some sense the

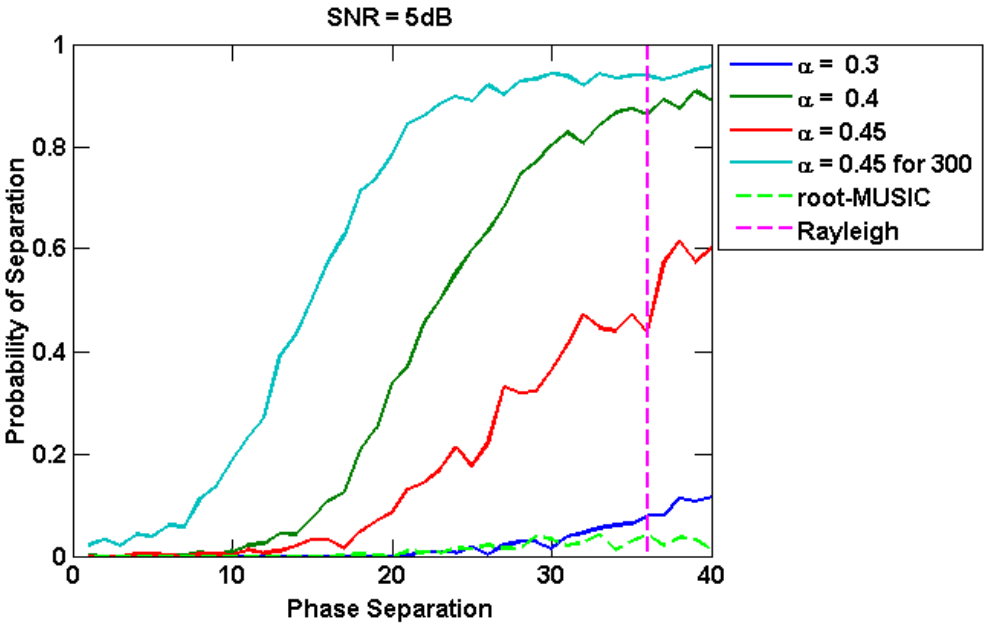

Figure 5.12: Probability of separation as a function of the two signals phase difference. PC-RISR with $\alpha = 0.45$ has not converged after 30 iterations.

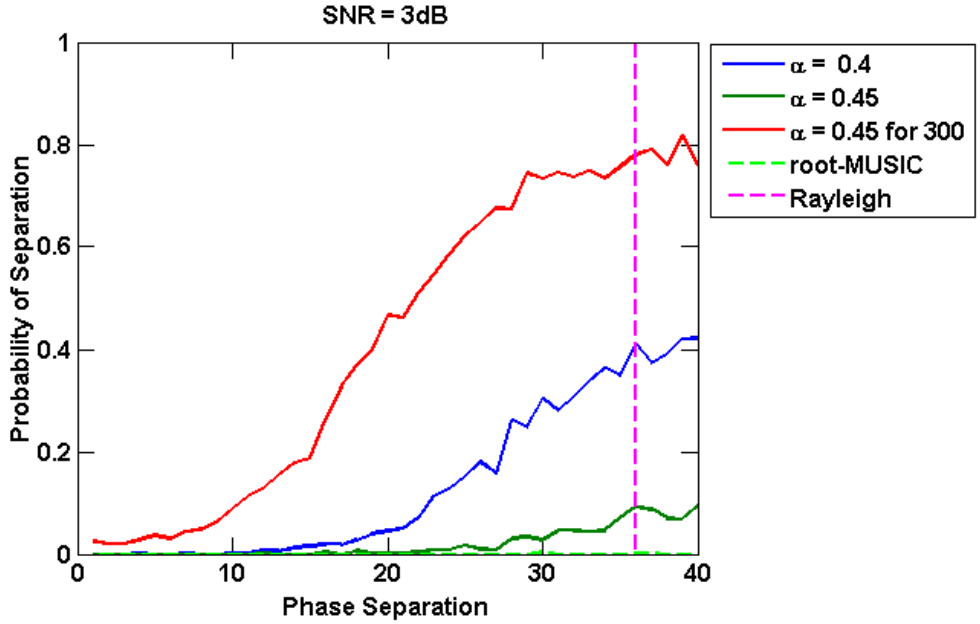

Figure 5.13: Probability of separation as a function of the two signals phase difference. At 3 dB we are approaching PC-RISR's limit of usefullness at this level of sample support.

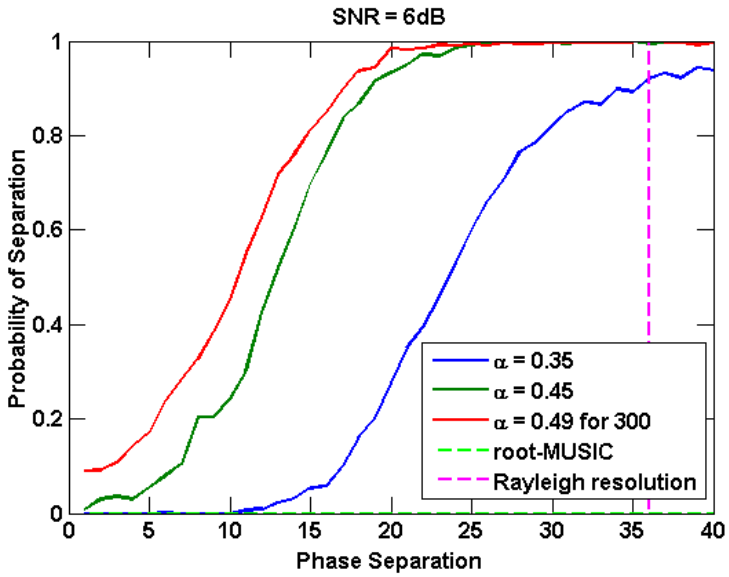

Figure 5.14: Using a more lenient detector, PC-RISR is able to separate signals with 100% probability down to an SNR of 6 dB

opposite of the detector used thus far. It merely checks to see if the two signals were properly separated and found within some tolerance of where they are known to be, and pays no heed to spurious peaks at all. As an example, the detector we have been using would label the estimate in Figure 5.7 a failed separation because if the third peak were to be interpreted as a signal, it would be an error. The detector used for Figure 5.15 and 5.14 would label it a successful separation because PC-RISR did indeed succeed in separating the signals presented to it. In that sense, the results in Figure 5.15 and 5.14 correspond to the limit of PC-RISR to actually separate the two signals, with no thought given to being able to realistically differentiate them from false peaks, and must not be conflated as achievable. With an optimum receiver the actual performance will most likely lie between the results given for the strict detector in Figure 5.9 through 5.13, and the those presented here for the lenient detector.

5.5.2.3 Dynamic Range

In the two previous sections PC-RISR's super-resolution capability was explored both as a function of SNR and signal separation, only the two signal scenario in which both signals have equal power was examined. In this section we examine PC-RISR's dynamic range by testing how PC-RISR

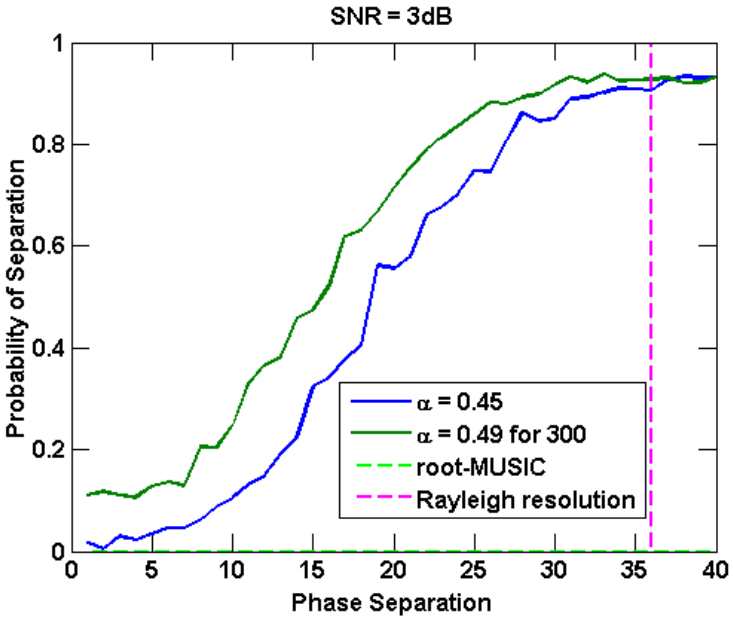

Figure 5.15: The theoretical super-resolution limit of PC-RISR is at an SNR of 3 dB.

performs on nearby signals with disparate powers.

To obtain the results in Figure 5.16 Monte Carlo simulations were performed in which two signals were presented for separation, the first at $\phi_1 = 0^\circ$ on the boresite and the second at some fixed positive angular distance ϕ_2 , which is some fraction of the Rayleigh resolution. The first signal has its power fixed at 20 dB (after applying array gain and coherent integration to subsume array size) and the second signal's SNR is varied between 0 dB and 20 dB. We chose $\alpha = 0.4$ and used 30 iterations. For each permutation 500 trials are run and the results are averaged to obtain the statistics shown. The results from root-MUSIC are also provided as a benchmark.

Comparing the two plots, it is clear that PC-RISR provides superior dynamic range. For signals spaced at 1/3 of the Rayleigh resolution PC-RISR is able to discern signals even when one is an entire order of magnitude larger than the other. The traces corresponding to closer signal spacing also reveal something interesting. Root-MUSIC cannot separate signals spaced at 1/8, 1/7, or 1/6 of the Rayleight resolution even when they have equal power (the best case scenario), but PC-RISR can.

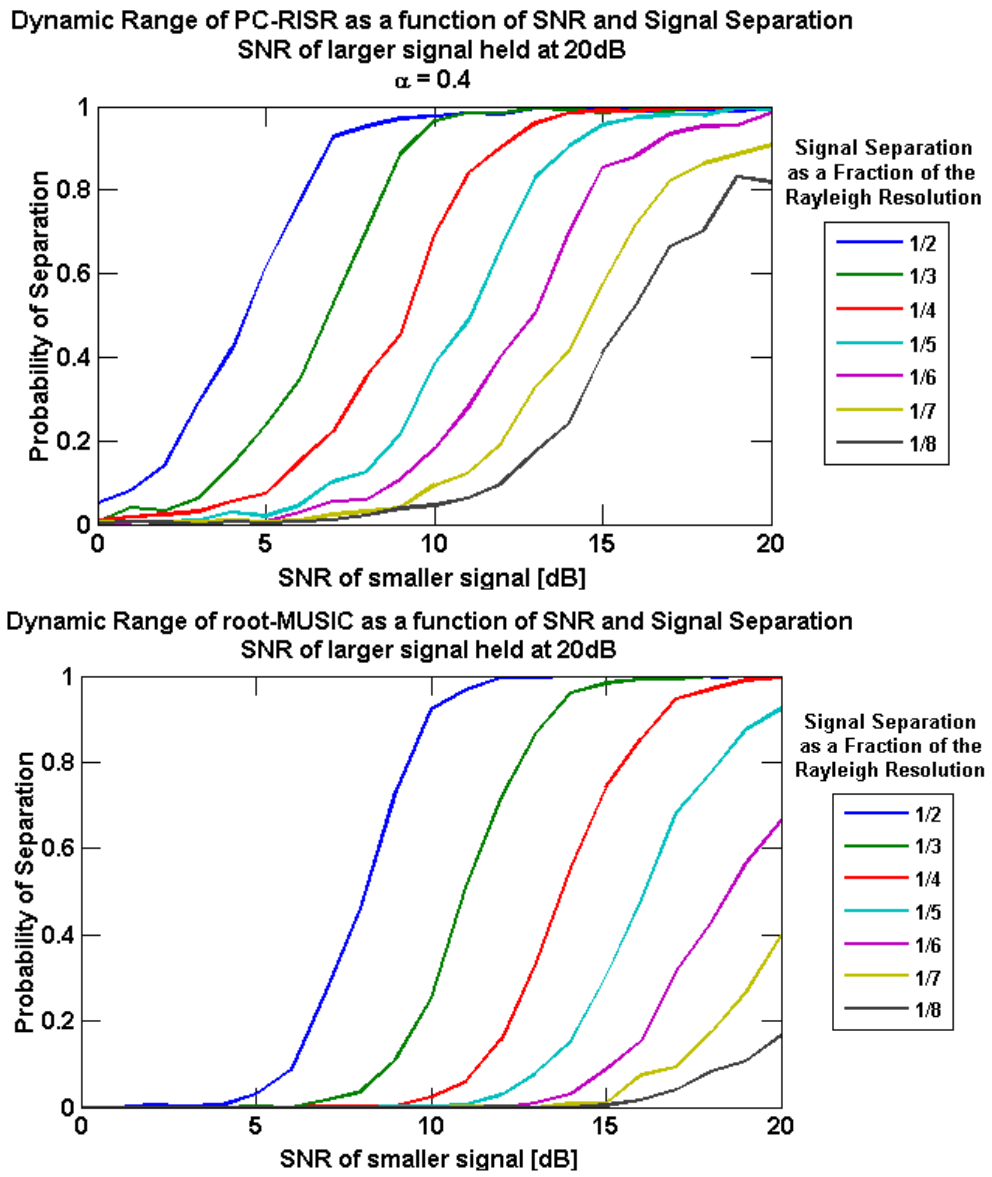

Figure 5.16: Comparison of PC-RISR and root-MUSIC's dynamic range. The first signal is held at 20 dB while the second signal's power is varied.

5.5.2.4 Correlated Signals

Signal correlation is a common problem because of the ubiquity of multipath propagation. In this section we examine how PC-RISR performs when the incident signals are correlated. Two primary cases are treated, the fully correlated-in-phase case and the fully correlated-out-of-phase case. Because these two cases are fully correlated, we can safely assume that partially correlated signals will lead to performance somewhere between what was found for the uncorrelated case in previous sections and the two correlated cases treated here.

Simulated signals are correlated in time. That is, the phase at each time index $\ell + 1$ is equal to the phase at ℓ plus some fixed phase offset. A fixed offset of 180° and 0° is used to simulate signals that are out-of-phase and in-phase, respectively.

Using a correlated received signal model, each of the previous simulations is repeated for both the correlated-in-phase and correlate-out-of phase cases. Here, root-MUSIC is implemented using both forward-backward averaging *and* spatial smoothing with two subarrays [16]. Without spatial smoothing root-MUSIC cannot be used on correlated signals, but using spatial smoothing has a side effect of lowering root-MUSIC's resolution due to the need for subarraying [34]. While PC-RISR's performance is affected somewhat by signal correlation, no extra measures have to be taken. As a result, the performance difference between PC-RISR and root-MUSIC is slightly larger than in the uncorrelated case because root-MUSIC's performance degrades as a side effect of spatial smoothing while PC-RISR's performance is mostly unaffected.

Observing Figure 5.17, it appears as though PC-RISR and root-MUSIC cannot separate signals that are in-phase, but in the context of Figure 5.18 and Figure 5.19 it is clear that PC-RISR and root-MUSIC both have lower resolution for in-phase signals than for uncorrelated or out-of-phase signals.

The next set of results in Figure 5.20 - Figure 5.23 include the correlated in-phase and out-of-phase cases of the Monte Carlo simulations used to produce Figure 5.8. They show the probability of separation as a function of signal separation instead of SNR, which helps discern how much super-resolution can be achieved at a given SNR.

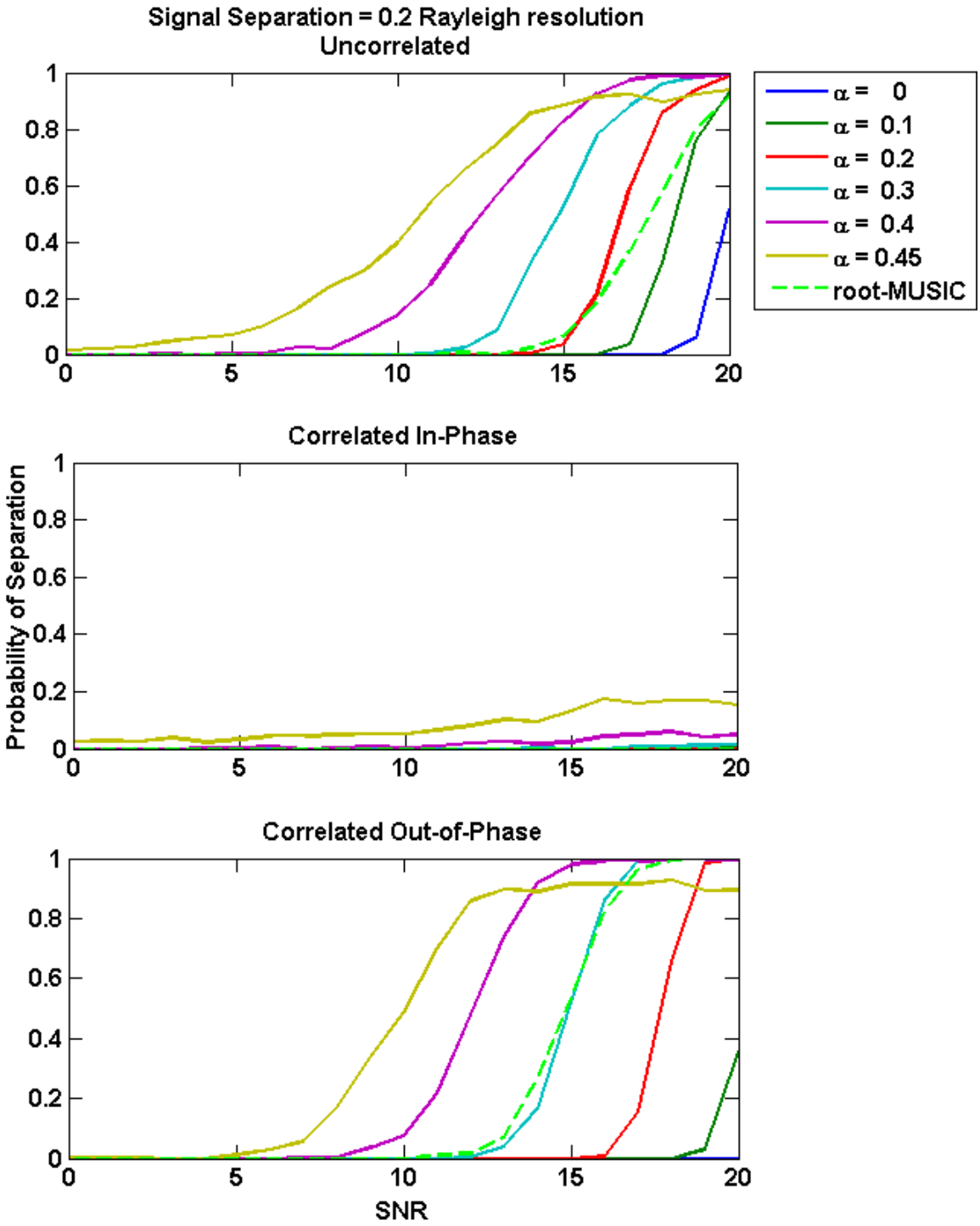

Figure 5.17: Comparison of the probability of separation as a function of SNR for the uncorrelated, correlated in-phase, and correlated out-of-phase cases. Signal separation is fixed at 1/5 of the Rayleigh resolution, $N = 10$ antenna elements, and $L = 10$ snapshots.

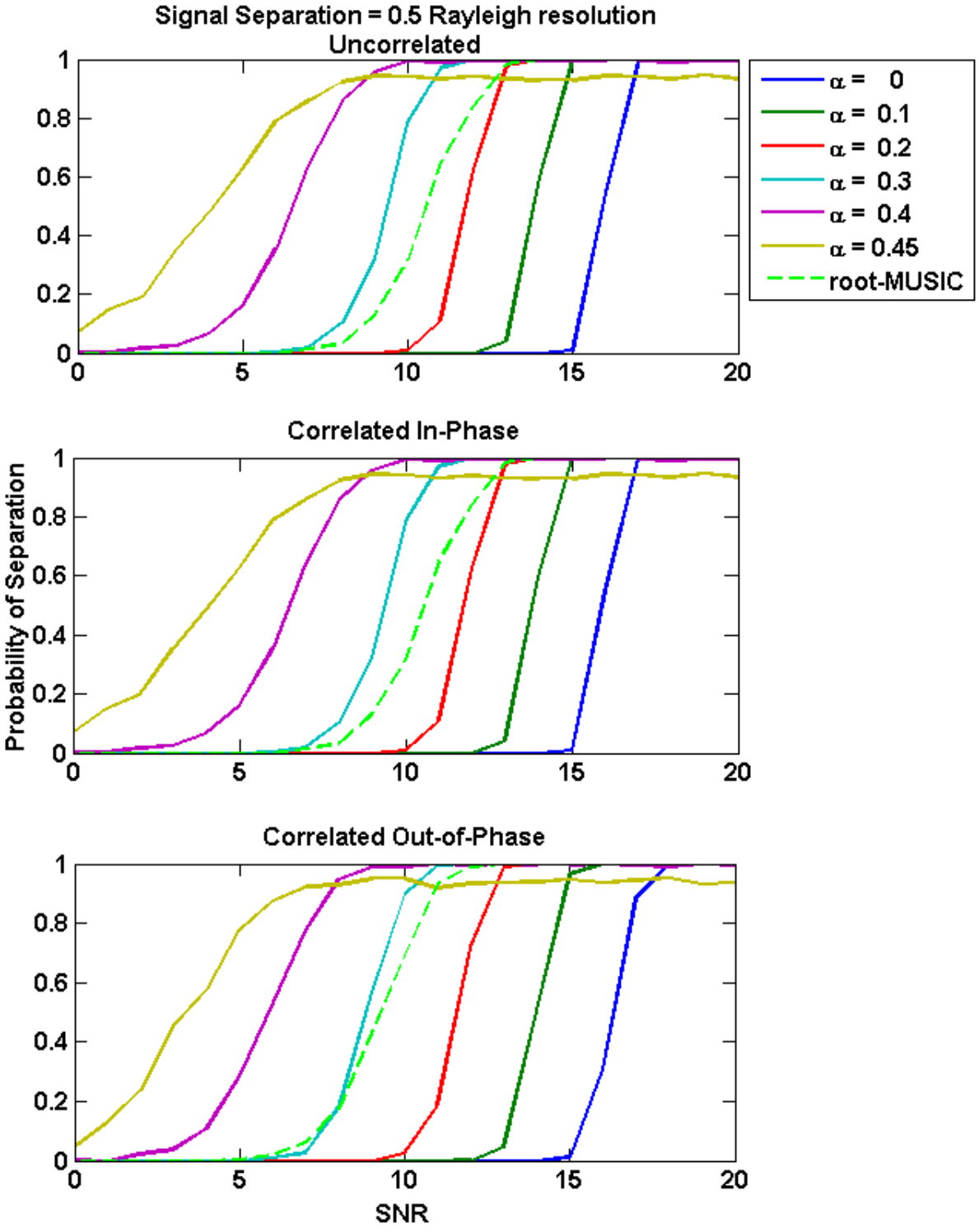

Figure 5.18: Comparison of the probability of separation as a function of SNR for the uncorrelated, correlated in-phase, and correlated out-of-phase cases. Signal separation is fixed at 1/2 of the Rayleigh resolution, $N = 10$ antenna elements, and $L = 10$ snapshots.

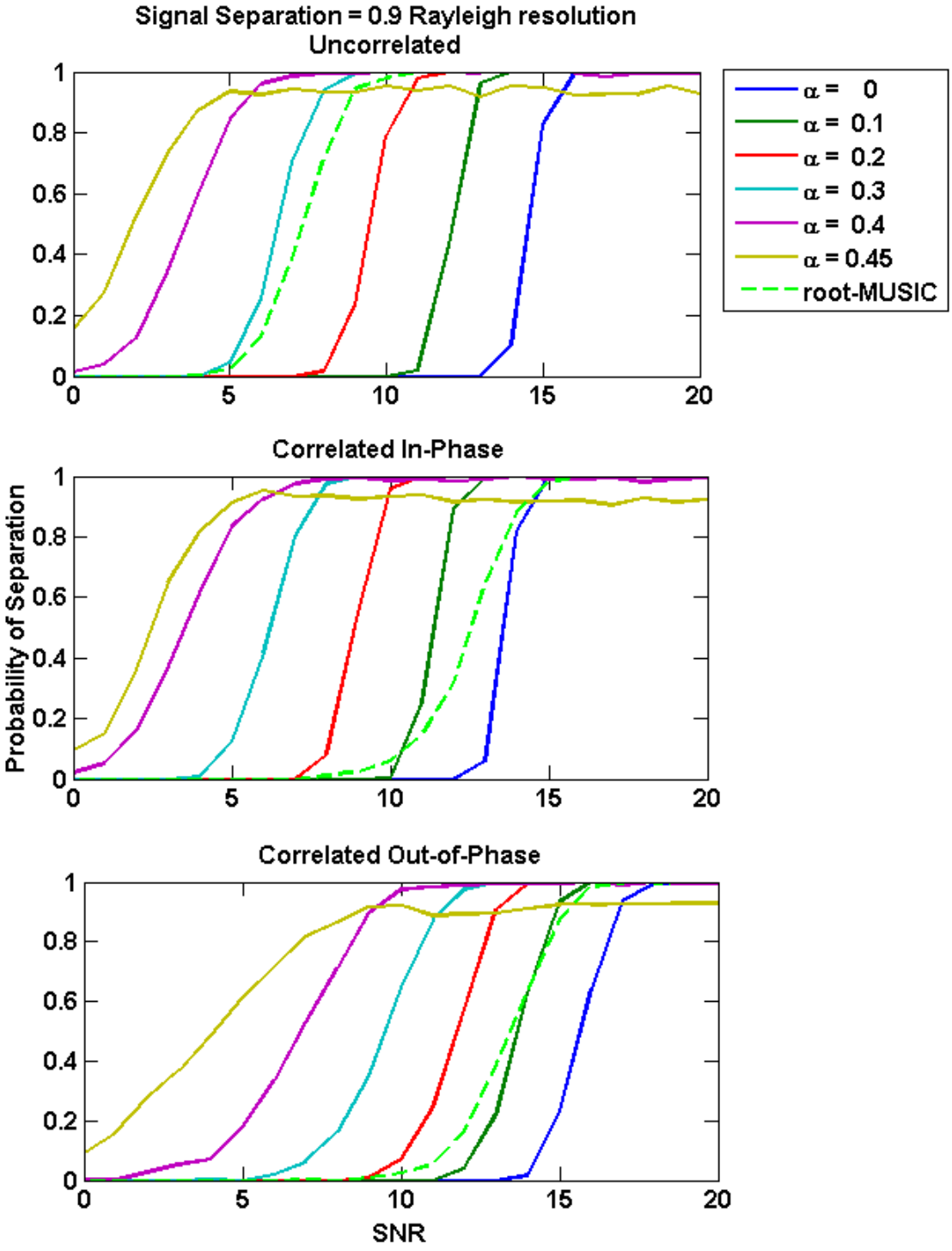

Figure 5.19: Comparison of the probability of separation as a function of SNR for the uncorrelated, correlated in-phase, and correlated out-of-phase cases. Signal separation is fixed at 9/10 of the Rayleigh resolution, $N = 10$ antenna elements, and $L = 10$ snapshots.

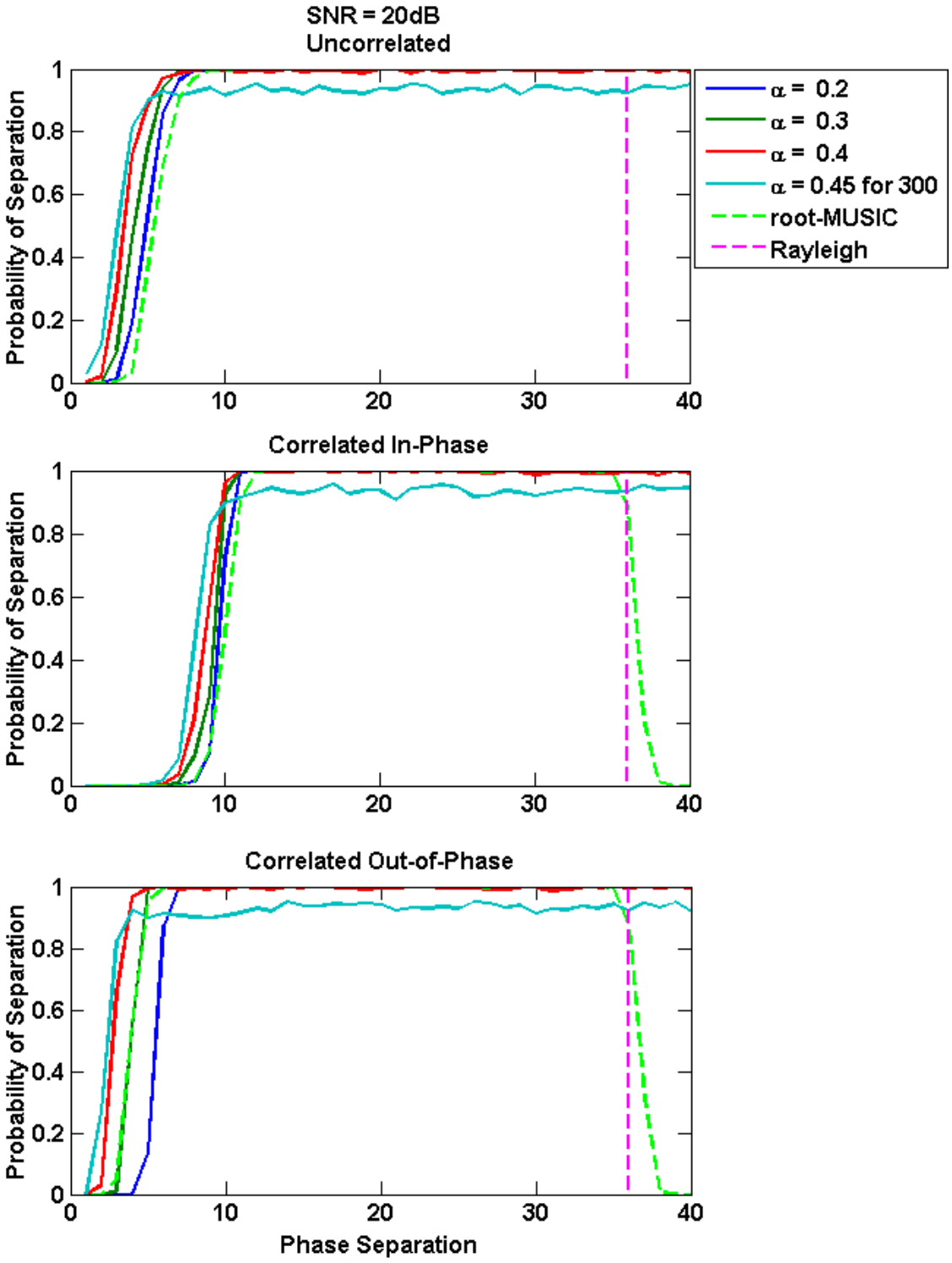

Figure 5.20: Probability of separation as a function of signal separation for the uncorrelated, in-phase, and out-of-phase cases. SNR is fixed at 20 dB, $N = 10$ antenna elements, and $L = 10$ snapshots.

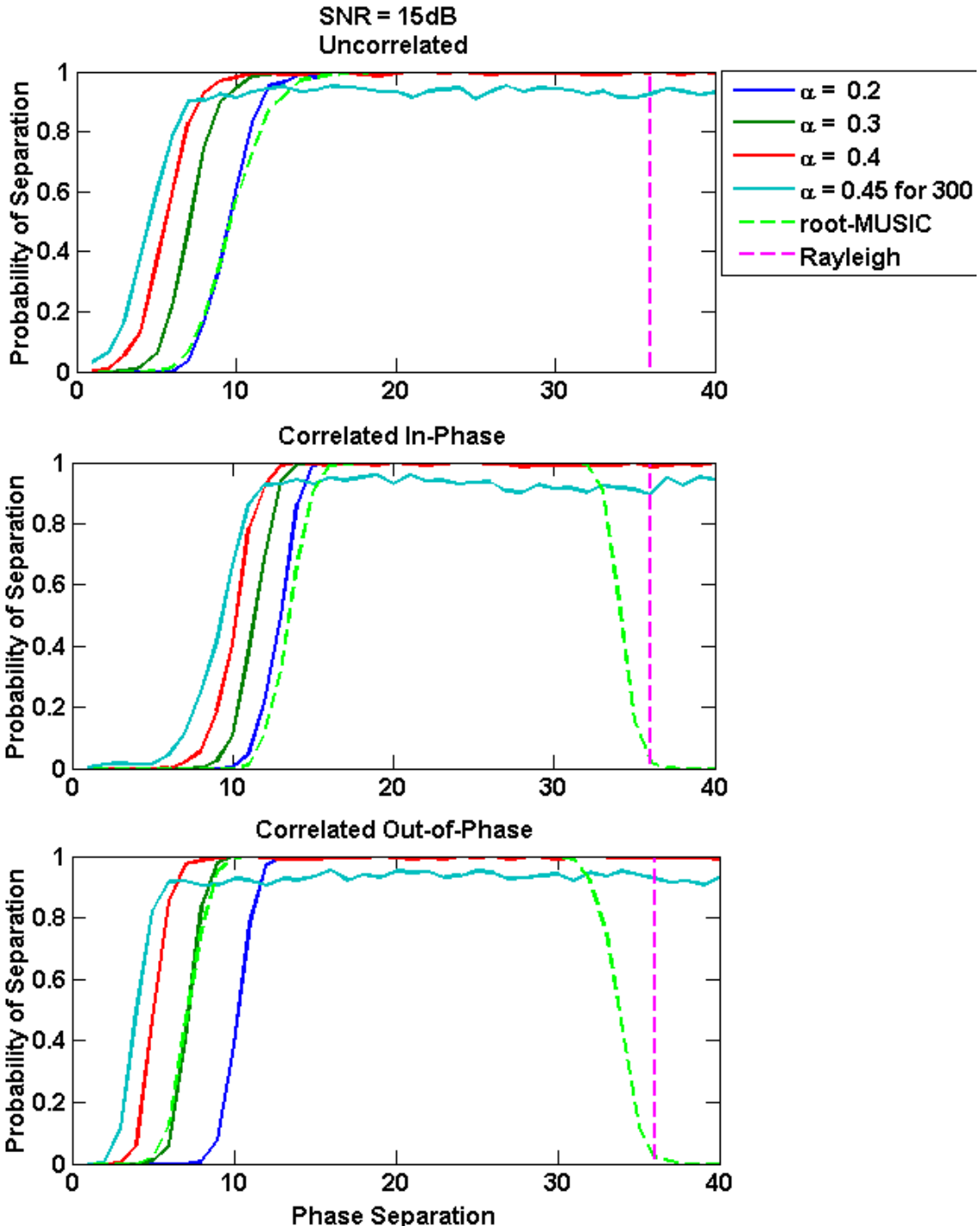

Figure 5.21: Probability of separation as a function of signal separation for the uncorrelated, in-phase, and out-of-phase cases. SNR is fixed at 15 dB, $N = 10$ antenna elements, and $L = 10$ snapshots.

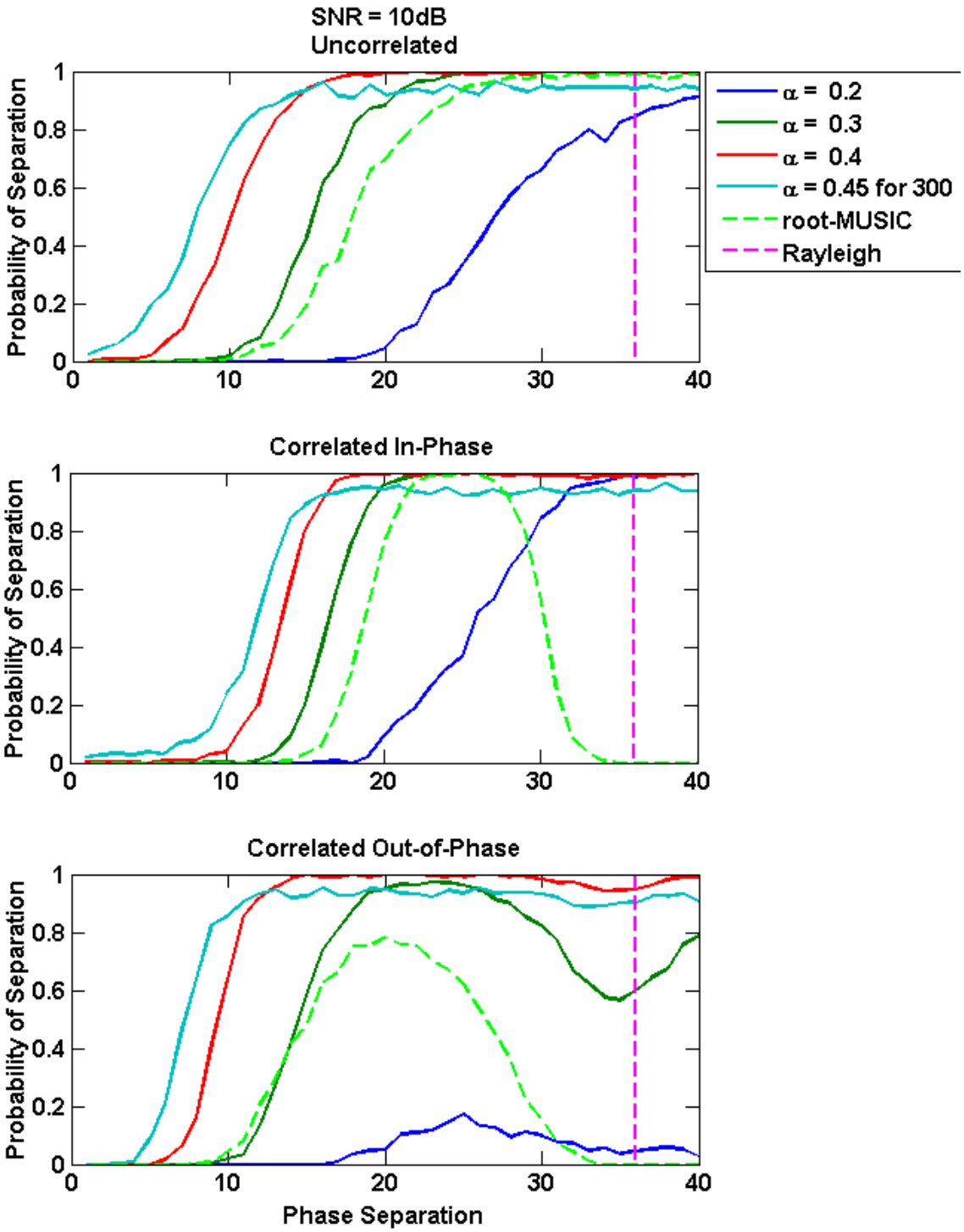

Figure 5.22: Probability of separation as a function of signal separation for the uncorrelated, in-phase, and out-of-phase cases. SNR is fixed at 10 dB, $N = 10$ antenna elements, and $L = 10$ snapshots.

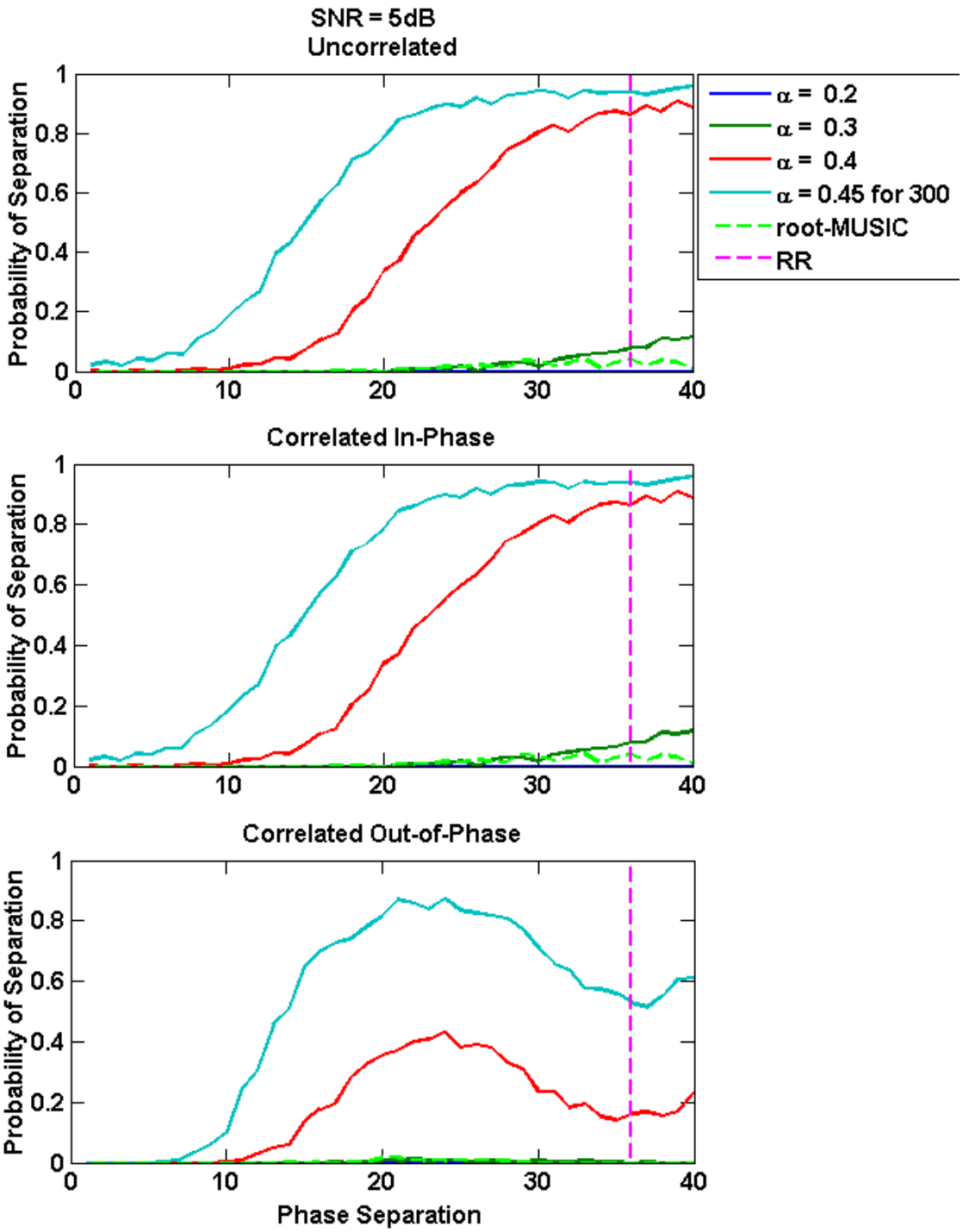

Figure 5.23: Probability of separation as a function of signal separation for the uncorrelated, in-phase, and out-of-phase cases. SNR is fixed at 5 dB, $N = 10$ antenna elements, and $L = 10$ snapshots.

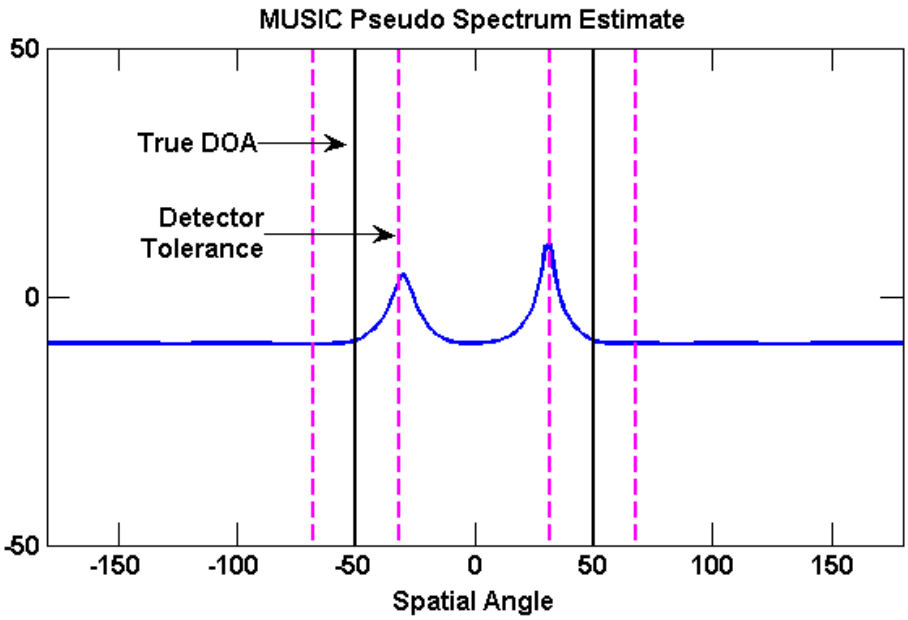

Figure 5.24: MUSIC has bias toward placing signals closer than they actually are, which can result in failed detection when they are far apart.

These results, particularly Figure 5.20, confirm that PC-RISR and root-MUSIC both suffer in terms of super-resolution factor when signals are in-phase, backing up our interpretation of Figure 5.17. We also observe that root-MUSIC actually performs better on out-of-phase signals than it does on uncorrelated signals, while PC-RISR's best performance is for the uncorrelated case. Finally, Figure 5.22 and Figure 5.23 show that PC-RISR becomes more susceptible to signal correlation at lower SNR, with it being particularly affected by the out-of-phase case.

Root-MUSIC's sudden drop off at high separation in Figures 5.20 - 5.22 can be explained by a bias toward locating sources close together. This bias is stronger for signals that are further apart or correlated, and the bias can become strong enough to pull root-MUSIC's estimate outside of the half beamwidth specified by the detector, resulting in failed detection. This tendency explains root-MUSIC's behavior in these figures. Root-MUSIC does not produce a spectrum, but it exhibits the same behavior as MUSIC does, so we use MUSIC to illustrate this bias in Figure 5.24. Generally, if sources are so far apart that MUSIC's bias becomes problematic, they can be separated with a simple matched filter beamformer. In practice MUSIC's bias is not as problematic as these simulation results might suggest.

Lastly, we will consider the effects signal correlation has on PC-RISR's dynamic range. As found in previous simulations, correlated signals (especially in-phase signals) reduce PC-RISR's maximum degree of super-resolution, but do not significantly affect its performance at greater separations. In particular, observe that at a separation of 1/8 the Rayleigh resolution, dynamic range is adversely affected by signal correlation, but in contrast, at 1/2 beamwidth separation PC-RISR is largely unaffected by signal correlation.

5.5.2.5 Sample Support

One of the merits of PC-RISR is that it can operate on as few as just one spatial snapshot. However, providing PC-RISR with more snapshots can greatly improve its performance in two ways. For fixed α, increased sample support moderately increases performance, but with increased sample support it is also possible to use higher values of α, which improves performance even further. Thus far, all simulations have used the same number of snapshots as antennas, but here we vary sample support to examine how it influences the limit of super-resolution.

The results in Figure 5.26 are obtained by performing Monte Carlo simulations using $N = 10$ with the previous specifications for 500 trials at each permutation, where the SNR (after coherent integration) was fixed at 10 dB and signal separation at half the Rayleigh resolution while the number of snapshots was allowed to vary. Unlike in previous simulations, instead of using 30 iterations for all values of α, we ran enough iterations for convergence based on Figure 5.2.

In Figure 5.26 $\alpha = 0.1$ and $\alpha = 0.2$ never manage to separate the signals because 10 dB is below their ability to operate. Root-MUSIC works well with enough sample support, but 10 dB is below where root-MUSIC is typically applied. Figure 5.26 is meant to show how robust PC-RISR is to low sample support, and that it can perform well at low SNR even without many samples. Figure 5.27 is the same as Figure 5.26, but with signal separation lowered to one-fifth of the Rayleigh resolution and the SNR raised to 20 dB to put all of the algorithms on equal footing.

In these simulations the previously described detector which ignores data below the noise floor is used. In a later section we will consider data below the noise floor, which enables operation at

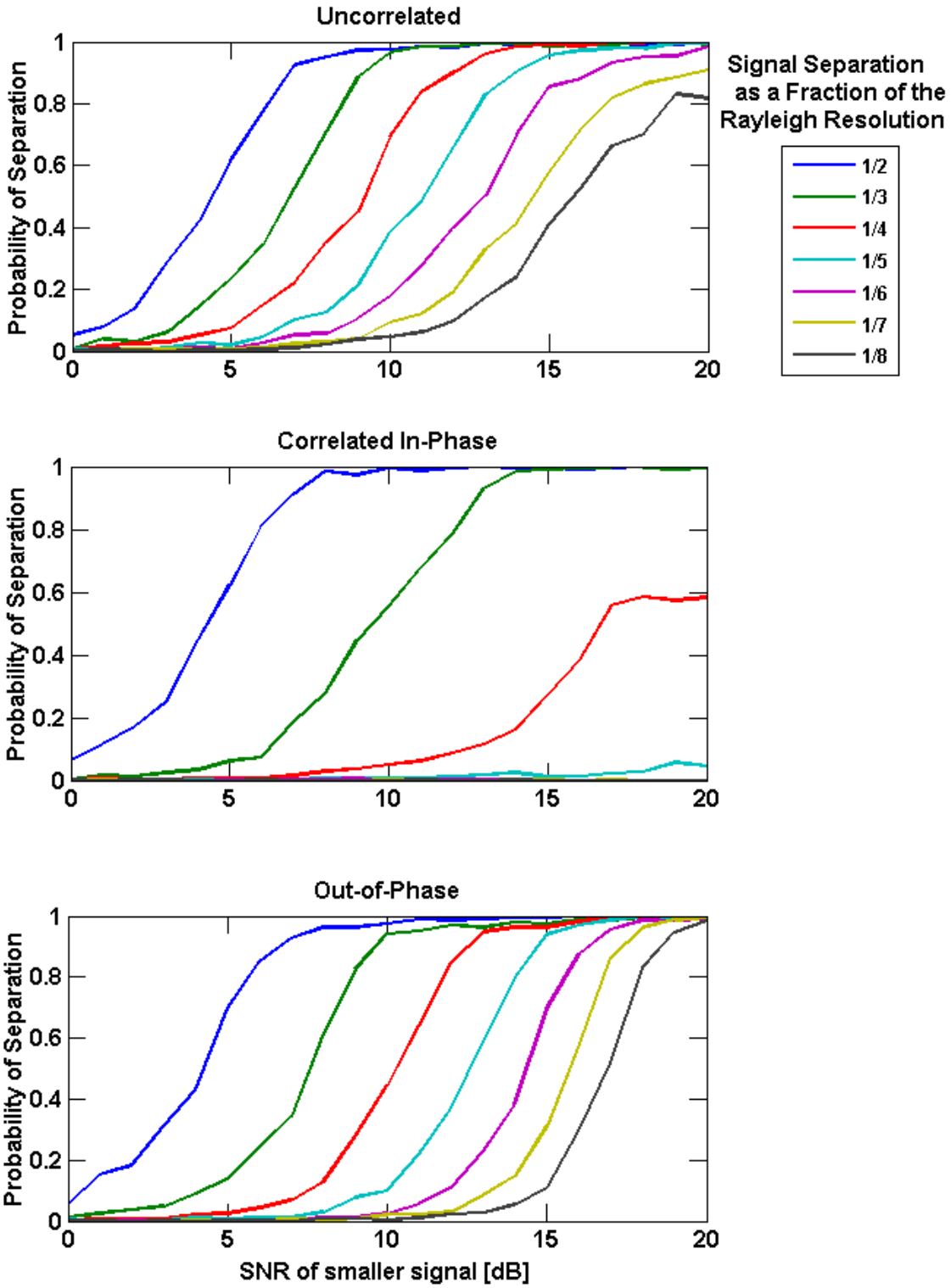

Figure 5.25: Dynamic range of PC-RISR for the uncorrelated, in-phase, and out-of-phase cases. The larger signal is held at 20 dB, $\alpha = 0.4$, $N = 10$ antenna elements, $L = 10$ snapshots, and 30 iterations are used.

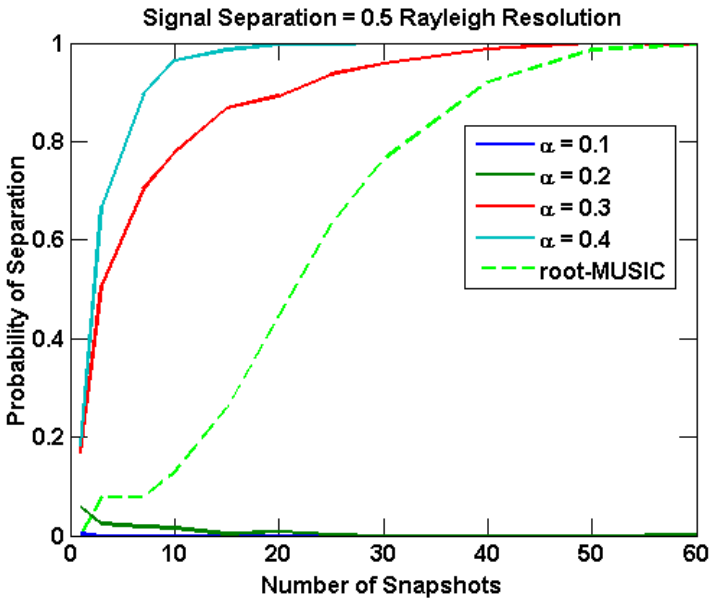

Figure 5.26: Probability of separation as a function of sample support for an SNR of 10 dB with signals separated by half of the Rayleigh resolution. An $N = 10$ element array is used.

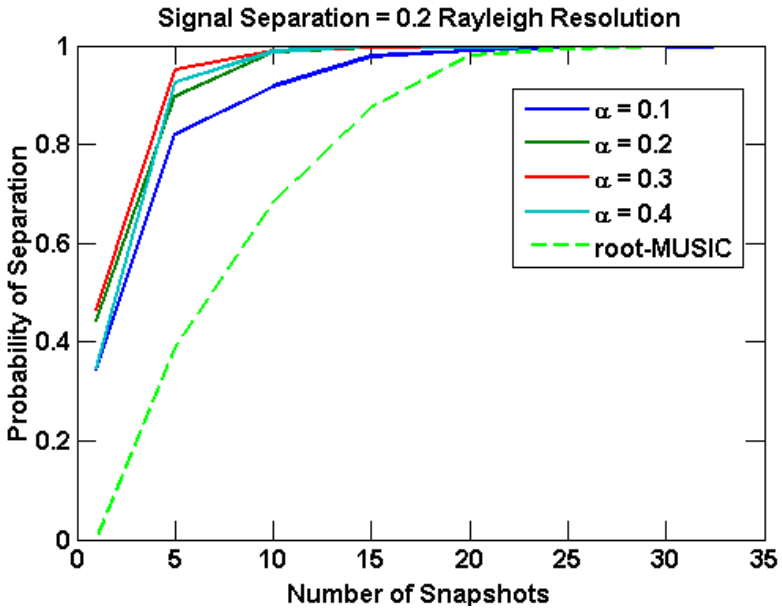

Figure 5.27: Probability of separation as a function of sample support for an SNR of 20 dB with signals separated by half of the Rayleigh resolution. An $N = 10$ element array is used.

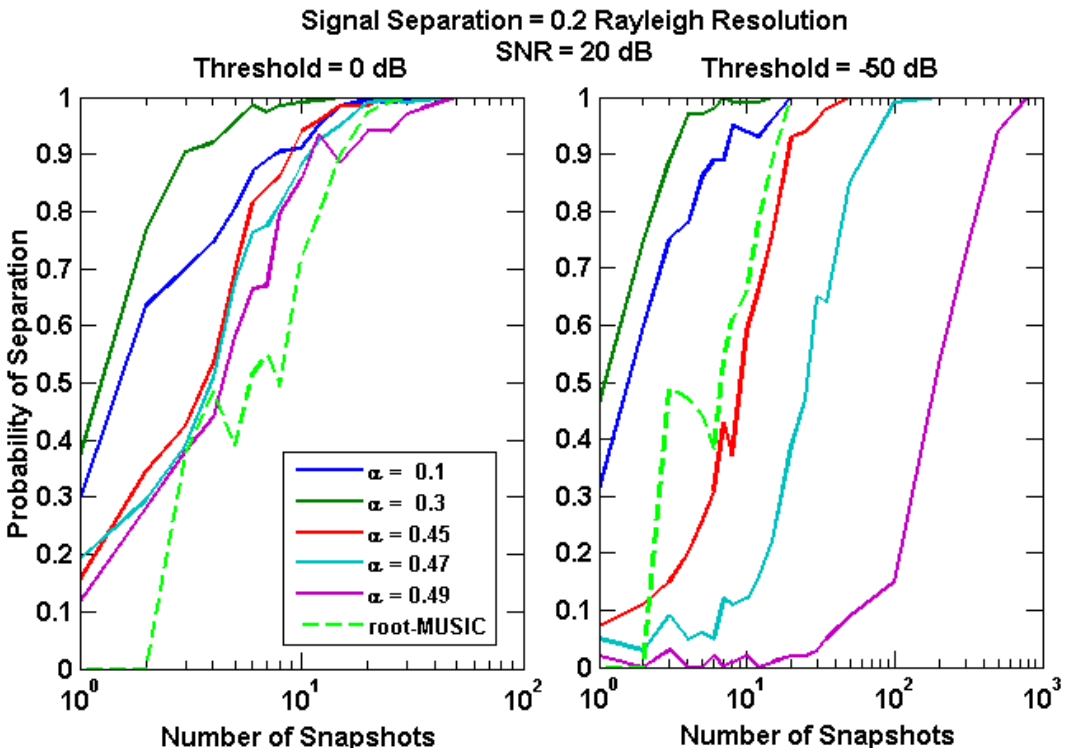

Figure 5.28: The number of snapshots needed for various values of α in order to separate signals at 0.2 the Rayleigh resolution for an SNR of 20 dB with the detector threshold at 0 dB (left) and -50 dB (right).

even lower SNR values, but also drastically changes the effects of sample support. In preparation for that, the next set of results explores how the importance of sample support changes when the detectors threshold is lowered from 0 to -50 dB.

Perhaps the most important question that needs to be asked in this context is, for a given value of α, how many snapshots are needed? Or asked in another way, with a fixed number of snapshots, what is the highest value of α that can be used without incurring spurious peaks? This question is answered in Figure 5.28 through Figure 5.30, which show the probability of being able to separate two signals as a function of sample support with the threshold at 0 dB as used in previous simulations, and at -50 dB to enable super resolution at even lower SNR values. Note that the results for root-MUSIC are the same in both cases because root-MUSIC does not use the threshold.

It is likely that in practice a different detector would be used or the threshold would be set at

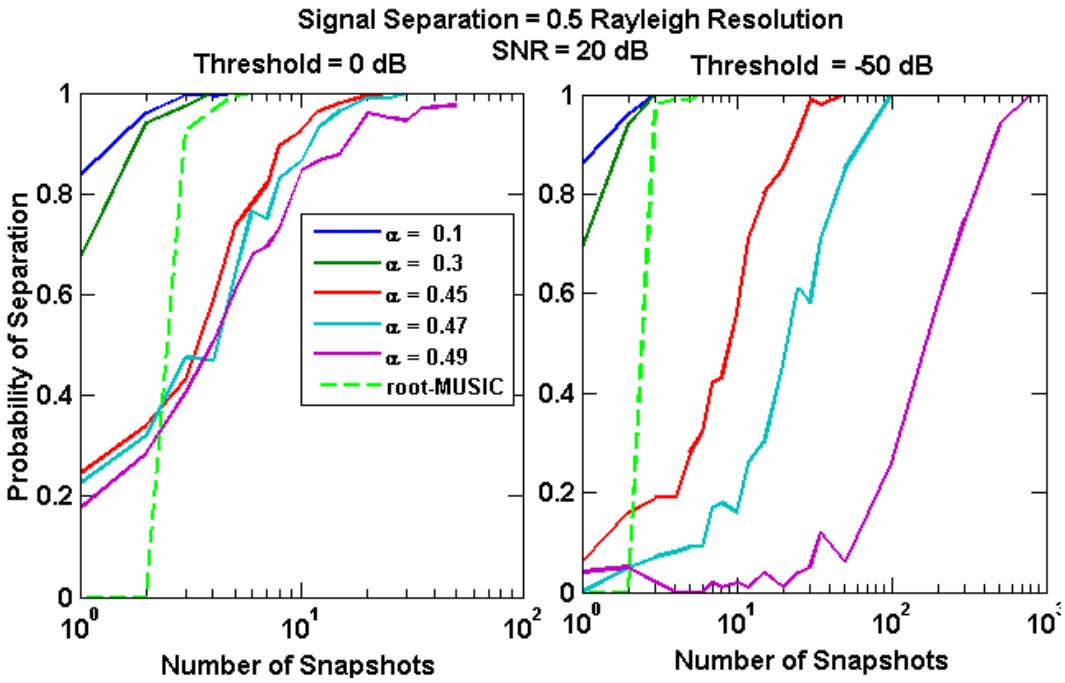

Figure 5.29: The number of snapshots needed for various values of α in order to separate signals at 0.5 the Rayleigh resolution for an SNR of 20 dB with the detector threshold at 0 dB (left) and -50 dB (right).

a different value. These results are for two rather extreme cases, so the performance of an actual system will almost certainly fall somewhere between those given here.

5.5.3 Constraint Dominant Regime

Having largely characterized the point solution regime of PC-RISR, we will now treat the constraint dominant regime. Values of α above 0.5 will never reach a point solution no matter how many iterations are run. Rather, they result in a spatial spectrum estimate that looks like that of GC-RISRt stretched downward. Figure 5.31 shows a series of estimates produced with values of α in the constraint dominant regime.

The original motivation for PC-RISR was actually not for super-resolution, but for modeling distributed sources — PC-RISR's low SNR super-resolution capability, or the point solution regime, was a later discovery. First GC-RISR was successful in increasing robustness to calibration error, but it sacrifices some of RISR's super-resolution. We wanted to build a version of RISR

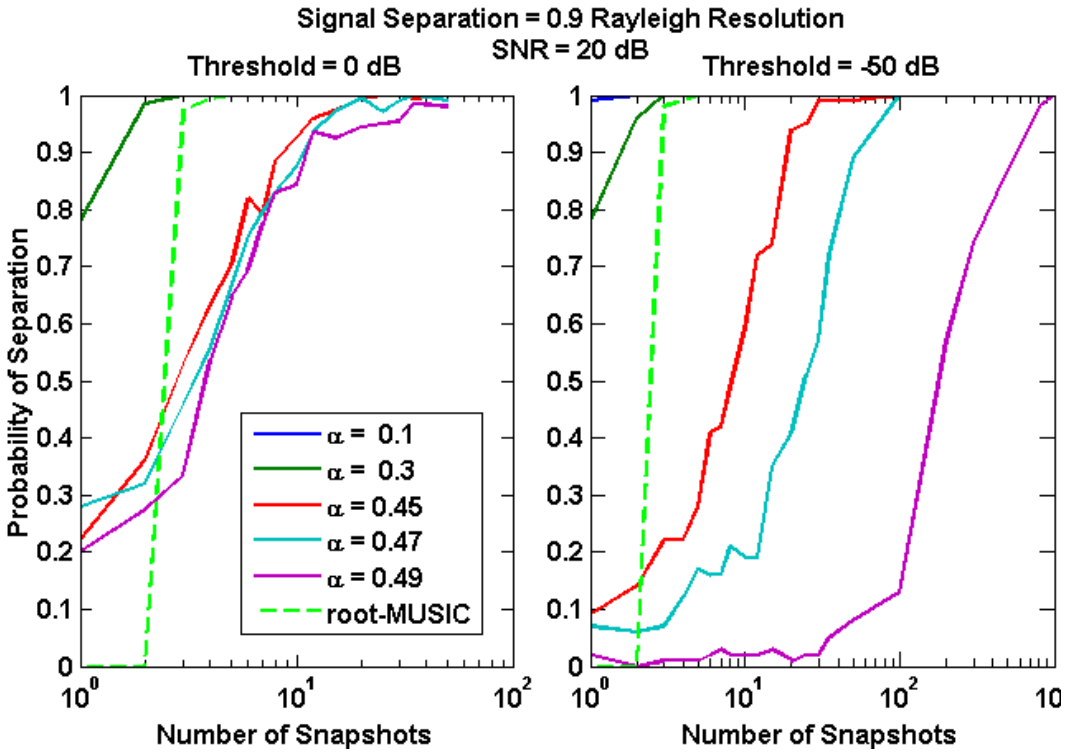

Figure 5.30: The number of snapshots needed for various values of α in order to separate signals at 0.9 the Rayleigh resolution for an SNR of 20 dB with the detector threshold at 0 dB (left) and -50 dB (right).

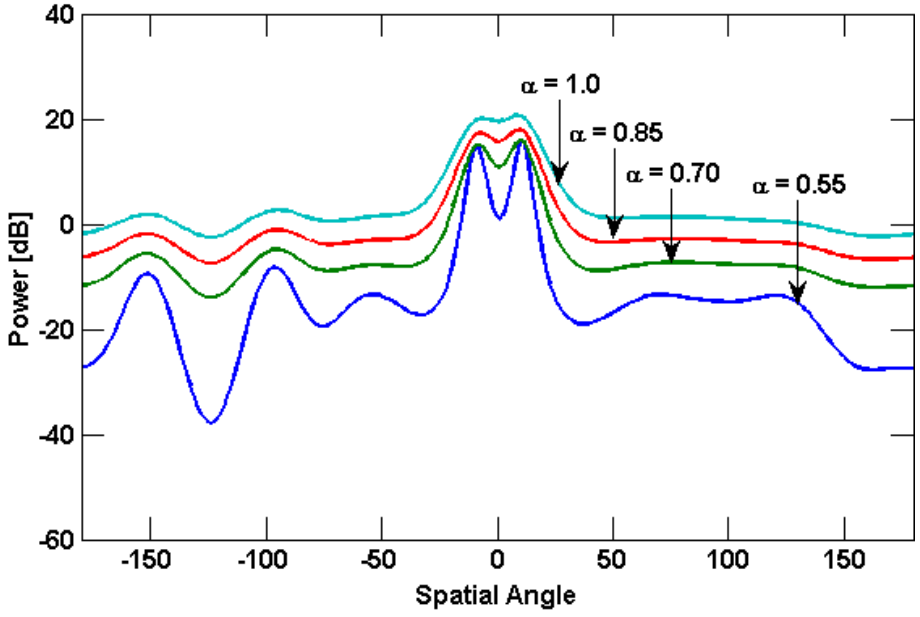

Figure 5.31: The spatial power estimate for various α in the constraint dominant regime. Note that lower values of α tend to underestimate the signal's power.

that would allow for tuning of the super-resolution factor, which is what the constraint dominant regime of PC-RISR accomplishes. To date, we know of no other algorithms that have a tunable degree of super-resolution.

One consequence of being the first algorithm to provide tunable resolution is that there are no pre-existing benchmarks or measures by which to evaluate PC-RISR in the constraint dominant regime, so we develop our own performance measures. We borrow some intuition from digital filter design because an ideal filter in some sense closely mimics what we would expect of perfect distributed source detection. We desire a flat response where the source exists (akin to the passband of a filter), and the sharpest rolloff and the greatest possible suppression of sidelobes away from the main lobe (analagous to a filter's stopband). In filter design, 3 dB ripple is common a figure for maximum passband ripple, and in detection settings a 3 dB null between peaks is often a criteria for ruling that two separate signals are present. Based on this, distributed sources are presented to PC-RISR for estimation and it is tuned to achieve maximum rolloff and sidelobe suppression such that the estimate is does not drop more than 3 dB from its maximum anywhere within in the extent of the distributed source.

Figure 5.32 shows an example where one estimate (in red) passes the test, and two (in blue) do not. For this example, we conclude that $\alpha = 1.0$ is the best of the three because it provides the sharpest rolloff and lowest noise floor without identifying more than one source or modeling the source as being narrower than it actually is.

Using this criteria, we design a Monte Carlo simulation similar to those used previously. Distributed sources are modeled as many (200) closely spaced point sources all having the same random phase, and with the total signal power divided equally among all the contingent sources. The SNR was fixed at 20 dB, a value high enough that detecting the signal would not be a concern, and a spatial oversampling factor of $M = 200$ was used to guarantee sufficient granularity for checking the 3 dB criterion near the edge of the sourse. The same 10 iterations were run for all values of α, since fewer iterations are need in the constraint dominant regime. As before, 500 trials were run for each permutation of various values of α and the width of the distributed source. The results,

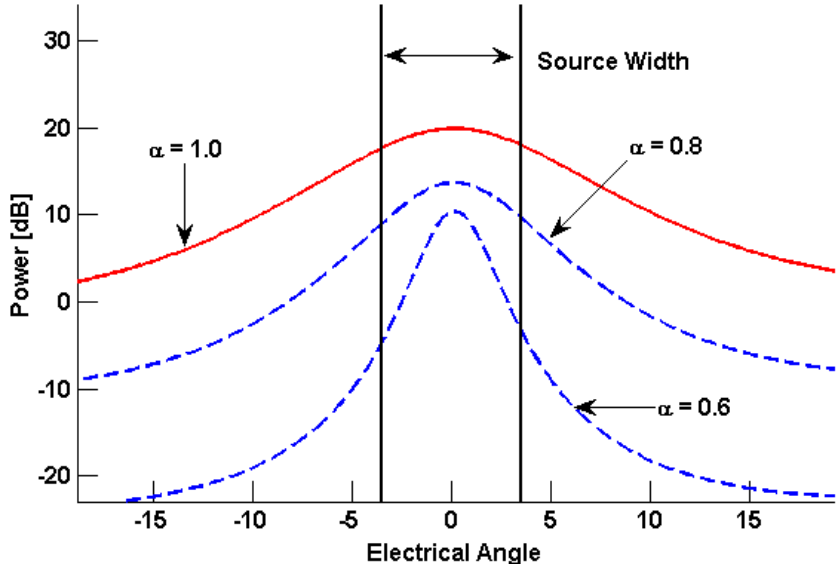

Figure 5.32: An example demonstrating the figure of merit used for PC-RISR in the constraint dominant regime. The red trace is deemed successful because the estimate does not fluctuate by more than 3dB within the source width. Blue traces are classified as failures.

given in Figure 5.33 are arranged to show the probability of modeling the distributed source in a way that meets our criteria as a function of α. The goal is to find the value of α that should be chosen to achieve a desired probability of correct modeling for a given source size.

In order to minimize sidelobes and push the floor as low as possible, one should choose the lowest value of α that achieves a suitable probability of proper modeling. Some general trends that we can observe from Figure 5.33 are that as SNR rises a larger value of α is necessary to keep peaks as wide as the source. As a consequence, PC-RISR can correctly model wider distributed sources at lower SNR. At an SNR of 10 dB the limit is about 3/10 of the Rayleigh resolution. For an SNR of 15 dB and 20 dB it is 0.2 and 0.15 times the Rayleigh resolution, respectively.

For an $N = 10$ element array and a signal with an SNR of 10 dB, that translates to modeling sources with a a maximum angular width of about 12° in electrical angle. Whether or not this is sufficient will depend on the application. In radar for example, it is unlikely that any target in the far field will be that large. On the other hand, in an application such as brain imaging where the far field assumption may not be valid and the sensors are very close to the sources, distributed sources larger than PC-RISR can model may be common. In such cases, the standard beamformer is a

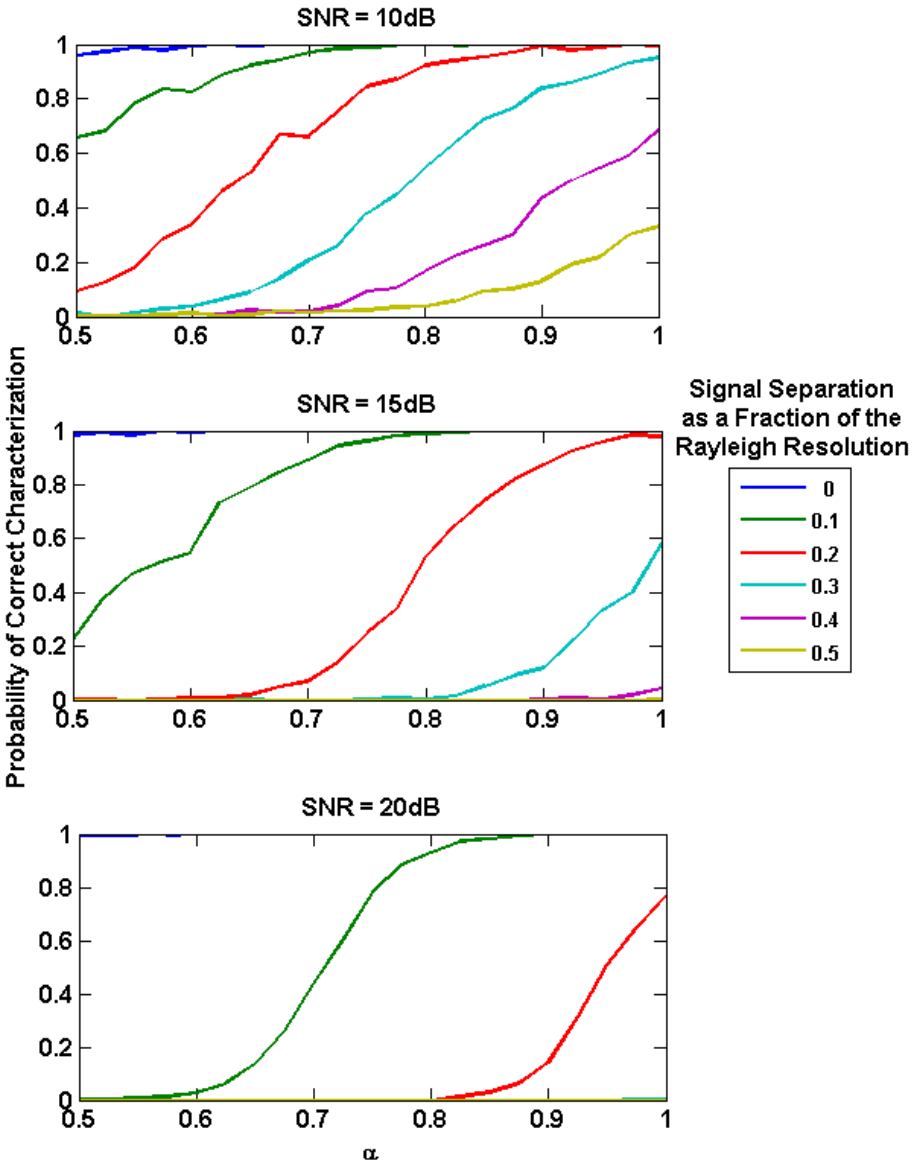

Figure 5.33: The probability of correctly modeling a distributed source as a function of α for several source widths

reasonable alternative, though its rolloff will not be as sharp.

5.5.4 Array Size

PC-RISR is a non-linear combination of two non-linear algorithms, so it is indeed very reasonable to question whether PC-RISR's performance scales linearly. Ideally, it would be possible to generate statistics for many large and small array sizes, but the high computational cost of PC-RISR makes this intractable, so instead we simply present anecdotal evidence. The following results come from Monte Carlo trials similar to those used in previous sections, however in stead of 10, $N = 50$ antennas and $L = 50$ snapshots are used, and to lower the cost of computation, only 200 trials are run for each permutation instead of 500.

In Figure 5.34 we can see that with all else held constant, PC-RISR achieves a greater level of super-resolution for 50 antennas than it does for 10. For $\alpha = 0.40$ and using the a probability of 0.9 as a reference point, using 50 elements results in a super-resolution factor of 3.6 while 10 elements only achieves about 2.8. By contrast, root-MUSIC holds steady at about 4.4. The discrepancy in PC-RISR's super-resolution factor is an effect of its non-linearity. Without the ability to thoroughly simulate a large number of antenna elements, we cannot tell if PC-RISR's performance will continue to improve with the addition of elements, but we can conclusively say that 50 antennas performs better than 10 even *after* normalization. It is likely that PC-RISR's super-resolution factor improves as a function of the number of elements N at least up to 50, and potentially beyond. Because of its non-linearity, it is possible that the margin by which PC-RISR outperforms root-MUSIC will continue to grow as more elements are added, however that has not been verified, and the high computational cost of PC-RISR makes using it with large arrays less feasible than root-MUSIC.

5.5.5 Computational Cost

The computational cost of PC-RISR is closely related to that of RISR given in [30] and outlined again in Chapter 4. Again referring back to the formulation of PC-RISR in 5.11, observe that both

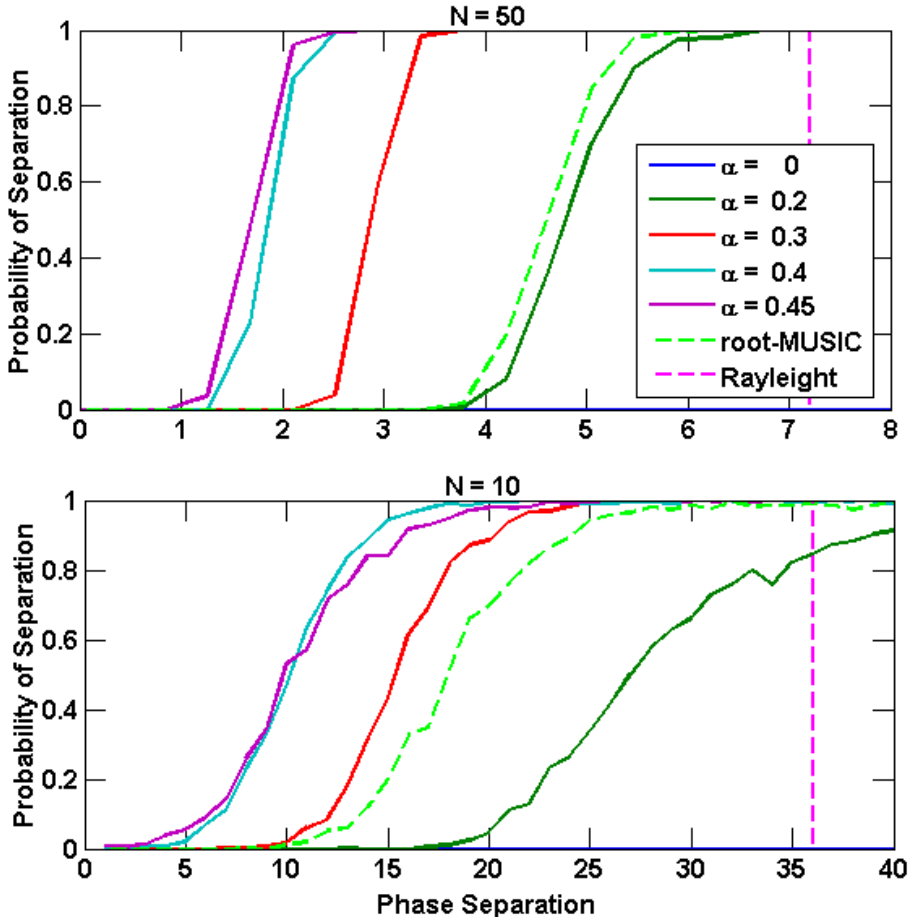

Figure 5.34: Probability of separation as a function of spatial separation at an SNR of 5 dB for a 50 element array using 50 snapshots.

the component from RISR and the one from GC-RISR contain the same matrix inverse, which means the cost of computing PC-RISR's filter is only trivially more than computing just RISR's. In fact, the only noteworthy difference is the two matrix multiplies required for the GC-RISR term. The exponents are applied to real scalars, and the combining the two RISR and GC-RISR terms is simply a multiplication of two real valued scalars.

$$\mathbf{w}_{PC,m} = \left[\left(\frac{1}{\mathbf{s}_m^{\mathrm{H}} D^{-1} \mathbf{s}_m} \right)^{\alpha} (p_m)^{1-\alpha} \right] D^{-1} \mathbf{s}_m$$

As a result, the computation cost of PC-RISR for one iteration is of the same order as RISR, $O(MN^2)$ for $L < 2N$ and $O(MNL)$ otherwise, where N is the number of antennas, M is the discretization of the spatial angle relative to N, and L is the number of snapshots used. Typically M is a multiple of N on the order of 20 [30].

It is important to keep in mind that while the cost of a single iterations may be the same, PC-RISR typically requires more iterations than RISR depending on the value of α chosen, though for large values of α PC-RISR converges faster than RISR. In most cases the choice of α will be dictated by a combination of the lowest expected SNR, signal separation, and sample support.

5.6 DOA Estimation for SNR < 0 dB

The detector used in simulations up to this point has been a very simple one. We check for peaks above the noise floor, and if there are only two peaks and both are within half of a bandwidth of their known location, we rule a successful separation. This detector operates under the assumption that no information below the noise floor can be considered reliable, but PC-RISR's excellent performance *near* the noise floor begs the question, "Can it work below the noise floor?"

As it turns out, with a different detector (one that does not ignore data below the noise floor) and enough sample support, PC-RISR is capable of operating at *negative* SNR values, however doing so requires a large number of samples and many iterations, resulting in hefty computational costs that make real time implementation less feasible than for the previous detector.

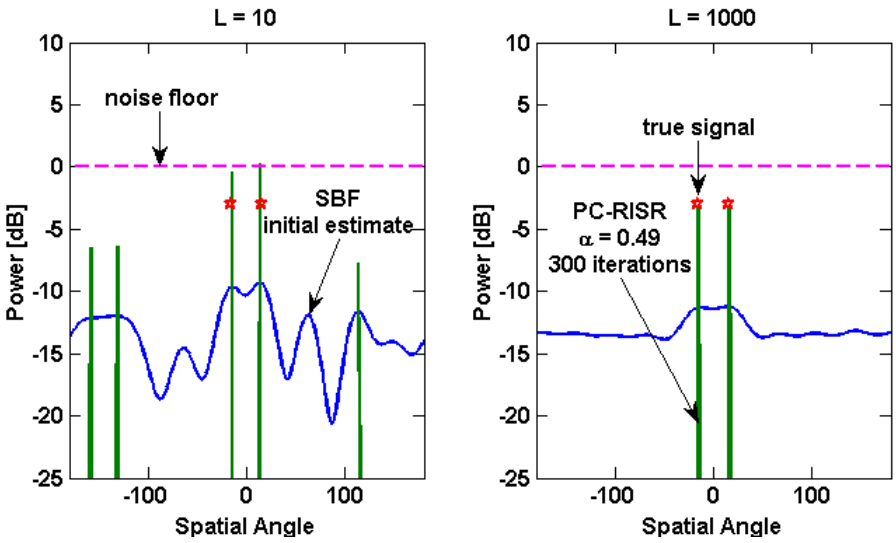

Figure 5.35: The initial standard beamformer estimate and the resulting PC-RISR estimate that results from it. Signals are at $\pm 15^\circ$ with -3 dB SNR.

The reason DOA estimation below the noise floor is possible is because of the way PC-RISR non-coherently integrates time samples. The noise, which is random in both magnitude and phase, when summed over many samples tends to drop because of cancellation, while the signal is constantly present, so it does not get averaged out. The same effect can be observed even with a standard beamformer, and indeed, that is PC-RISR's initial estimate. Figure 5.35 demonstrates how the initial estimate provided for PC-RISR changes as more samples are used, and how this affects the reliability of the final estimate.

Figure 5.36 illustrates an important concept - that the estimate above the noise floor is robust to low sample support, but the estimate below the noise floor is highly dependent on the number of snapshots available. As a result, PC-RISR with the 0 dB threshold detector is extremely robust to low sample support, but performance drops off at single digit SNR values because of the nature of the detector. On the other hand, if the detector's threshold is set lower or removed all together, then suddenly sample support becomes imperative for avoiding false detections. Observe that in Figure 5.36, both the left and right plots would have properly detected the single peak at 0° with the previous detector, but without the threshold the left plot would result in many spurious peaks. Figure 5.35 shows a single trial example of how high sample support can be used to make data

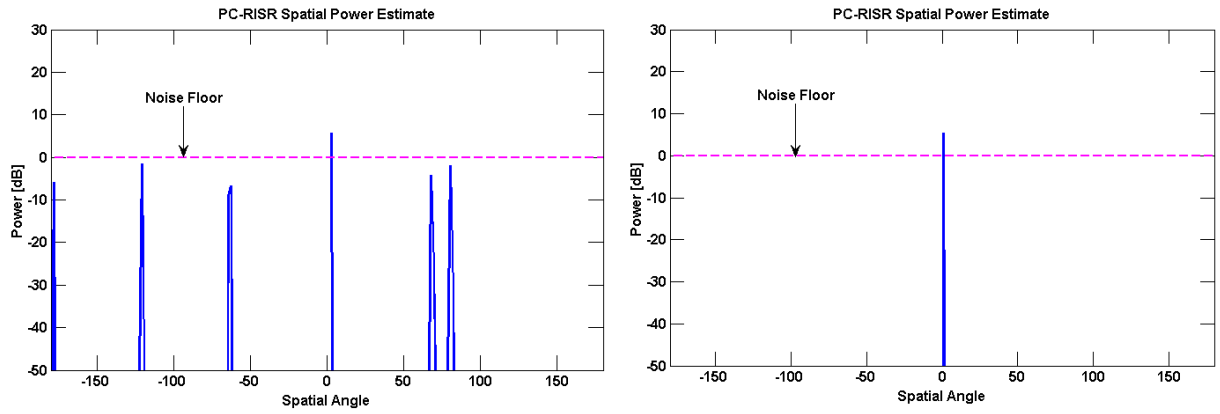

Figure 5.36: Sample support has a large effect on the estimate below the noise floor, but not as strong an influence on the estimate above the noise floor. An estimate with L = 5 (left) snapshots is contrasted with a estimate produced with L = 50 snapshots (right) of a single 10 dB source at 0° and $\alpha = 0.45$.

below the noise floor reliable enough to perform super-resolved signal separation.

Again, we employ Monte Carlo trials as a means of characterizing the performance of PC-RISR when the detector's threshold is adjusted. In the following results, the detector was modified such that instead of ignoring all data below 0 dB, it now only ignores data below -50 dB. Similarly, instead of using just $L = 10$ snapshots, $L = 1000$ were used to suppress spurious peaks via non-coherent integration. All other parameters are the same as those used in previous Monte Carlo simulations.

Figure 5.37 and Figure 5.38 show the probability of separation as a function of SNR, with signal separation fixed at 0.5 and 0.9 times the Rayleigh resolution, respectively. Figure 5.39 shows the probability of separation as a function of signal separation with SNR fixed at -5 dB.

Next we return to the important question of how many samples are required for a certain value of *alpha*, but this time examine specifically the low SNR regime. Results were compiled from 200 trials per permutation while varying the number of snapshots and fixing the SNR at 0 dB and signal separation at 1/5, 1/2, and 9/10 of the Rayleigh resolution. The same trial was repeated with both the 0 dB threshold detector and the -50 dB threshold detector to show how the choice of detector influences the degree to which sample support matters. It is also important to notice that for Figures 5.28 - 5.30, the SNR is large, which makes using a 0 dB threshold distinctly advantageous.

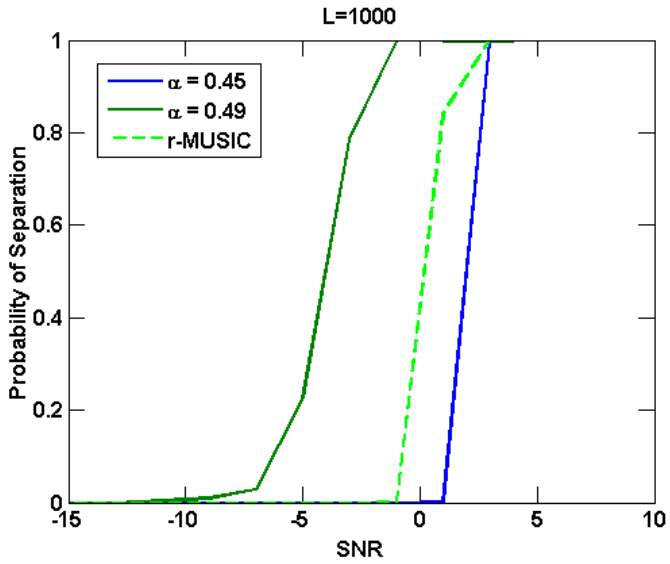

Figure 5.37: Probability of separation of two signals separated by 0.5 Rayleigh resolution. $N = 10$, $L = 1000$, 300 iterations for all values of α.

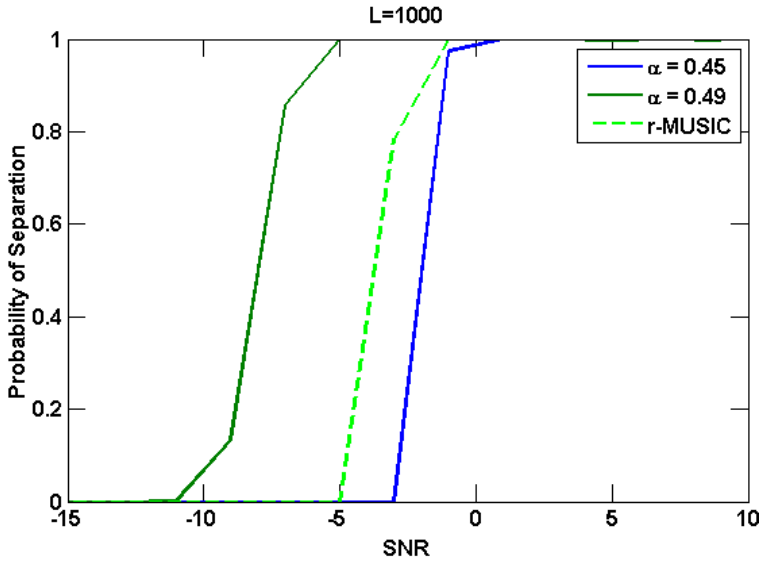

Figure 5.38: Probability of separation of two signals separated by 0.9 Rayleigh resolution. $N = 10$, $L = 1000$, 300 iterations for all values of α.

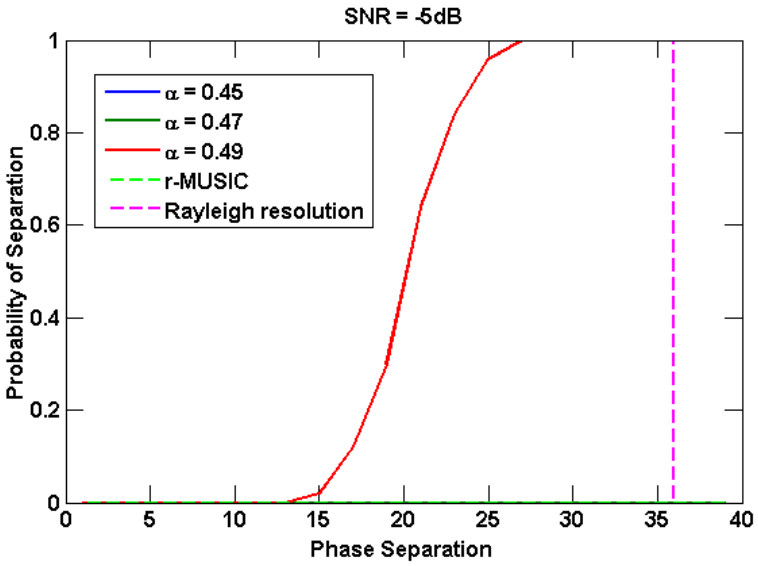

Figure 5.39: Probability of separation as a function of signal separation. SNR $= -5$dB, $N = 10$, $L = 1000$, 300 iterations for all values of α.

However, in figure 5.40 - 5.42 the SNR is 0 dB, which means the most of the points in the estimate will be ignored with high probability, so a lower threshold has to be used even though it will result in requiring more sample support. In practice the SNR of the incident signal will generally be unknown, so the threshold should be set low unless weak signals are of no interest or some prior knowledge about the SNR exists. Likewise detectors used in practice may use a value between 0 and -50 dB, or may not employ a floor at all, so these results should be interpreted as a loose bounds on performance.

What we find is that for the 0 dB threshold detector is that at high SNR with closely spaced signals, PC-RISR outperforms root-MUSIC for all values of α. As signals are spread further apart, values of α greater than 0.4 require almost the same number of samples to suppress spurious peaks as before, while low values of α and root-MUSIC can get by on fewer samples than previously. For signals that are just barely super-resolved, root-MUSIC performs better in terms of sample support than PC-RISR with high α, but PC-RISR with low α is the best.

If the SNR is reduced to 0 dB, the 0 dB threshold detector can no longer be used, but the -50 dB threshold can be. We find that root-MUSIC requires signals to be at nearly the Rayleigh resolution in order to separate them, and even then it requires almost 1000 samples. By contrast, PC-RISR

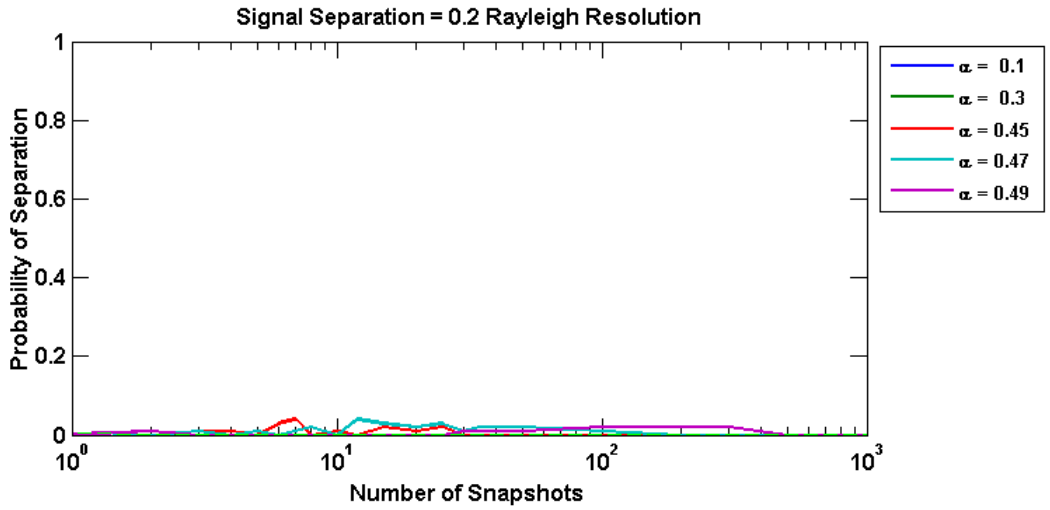

Figure 5.40: The number of snapshots needed for various values of α in order to separate signals at 0.2 the Rayleigh resolution for an SNR of 0 dB with the detector threshold at -50 dB.

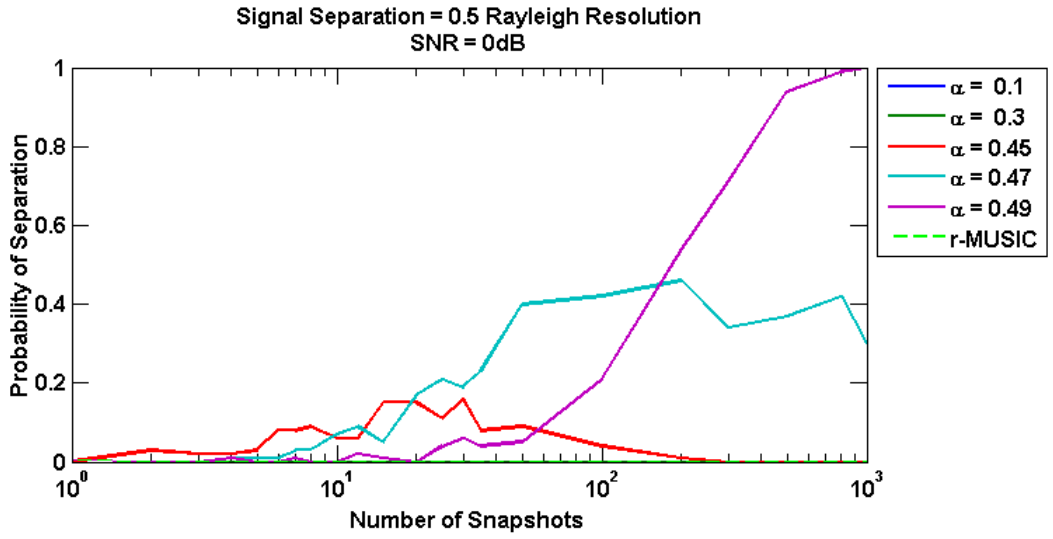

Figure 5.41: The number of snapshots needed for various values of α in order to separate signals at 0.5 the Rayleigh resolution for an SNR of 0 dB with the detector threshold at -50 dB.

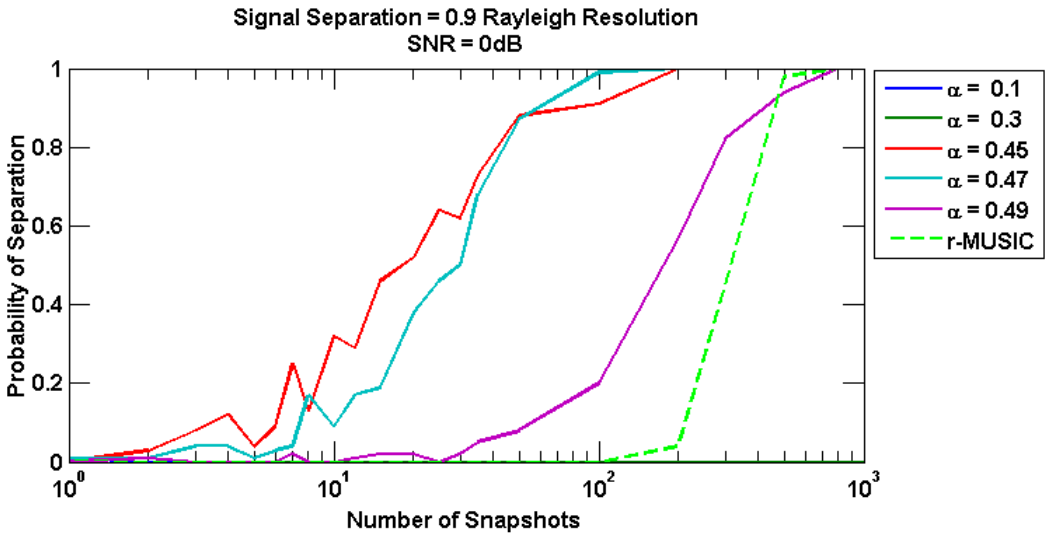

Figure 5.42: The number of snapshots needed for various values of α in order to separate signals at 0.9 the Rayleigh resolution for an SNR of 0 dB with throw away thrhreshold at -50 dB.

can separate the same signals with 10 times fewer samples, and with the same number of samples it can separate signals that are almost twice as close together.

Chapter 6

Applications of PC-RISR to Detection

PC-RISR's distinguishing feature is its ability to operate at low SNR, which begs the question of detection. How small a signal can be detected, if we are not concerned with having to be able to separate it from another signal?

It was shown in Chapter 5 (see Figure 5.26), that PC-RISR is robust to low sample support as long as interest is only in signals above the noise floor. However, if a detector is employed that does not ignore data below the noise floor, PC-RISR's performance becomes highly dependant on L. This behavior can be explained by the nature in which sample support affects PC-RISR's estimate. As demonstrated in Figure 6.1, the estimate below the noise floor is much more sensitive to sample support than is the estimate above the noise floor. As a result, the single source detection case in which we enforce that no false peaks appear within 50 dB of the noise floor becomes highly sensitive to sample support, whereas if data below the noisefloor is ignored, PC-RISR achieves nearly optimal performance with relatively few snapshots.

The spurious peaks that can occur when using PC-RISR are statistically related to the signal SNR, α, and L. When the SNR is small, α is large, or L is limited, spurious peaks have a higher probability of occurring. Conversely, they can be mitigated by reducing α or collecting more samples. It is important to understand the nature by which spurious peaks are suppressed. Improving one or both of these parameters will not result in the spurious peaks becoming lower, but rather in

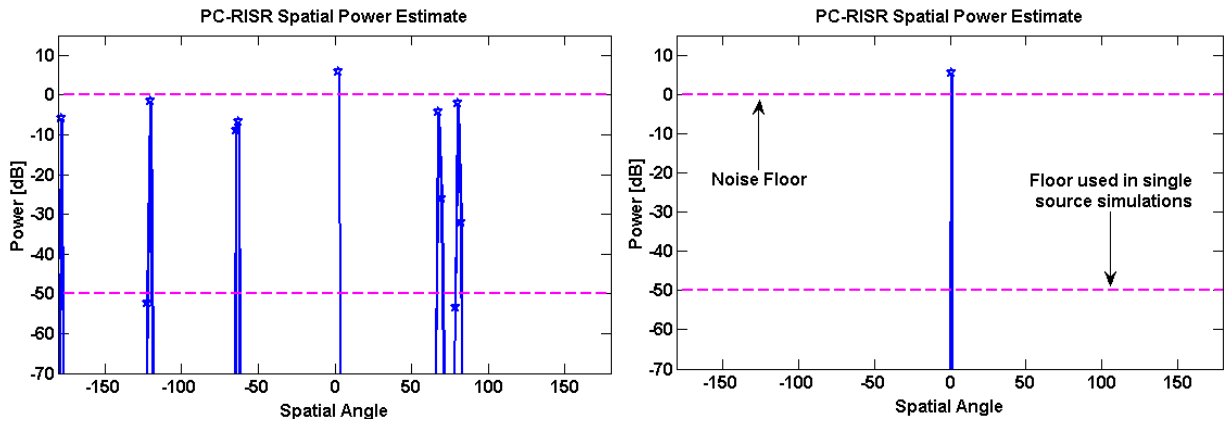

Figure 6.1: Sample support has a large effect on the estimate below the noise floor, but not as strong an influence on the estimate above the noisefloor. An estimate with L = 5 (left) snapshots is contrasted with a estimate produced with L = 50 snapshots (right)

making them less likely to occur. As such the power of the *average spurious peak* is not greatly affected by these three parameters.

Another Monte Carlo simulation is performed, but this time only one signal on the boresite (at 0°) is incident on the array, and different detection logic is employed. In this case we desire to be able to observe data below the noise floor, so the detector threshold is moved down to -50dB. To be considered a successful detection, PC-RISR must produce only one peak above -50dB, and it must be within half of a bandwidth (18° for $N = 10$) of the true DOA. The signal's SNR was varied between -25 dB and 5 dB and the number of snapshots was varied between 1 and 1000. Gain and phase model mismatch of errors of 1% were included and PC-RISR's model mismatch variance term was set to 1.5×10^{-3}. As stated in previous results, for low SNR, low and mid values of α result in low SNR signals being suppressed, so for this simulation only values of α greater than 0.4 give meaningful results. Based on Figure 5.2, a ample number of iterations to allow convergence was chosen for each value of α. For each permutation 500 iterations are performed and the results compiled to produce Figure 6.2.

These results are surprising because they show that with sufficient sample support PC-RISR is capable of detecting signals not only very close to the noise floor, but several dB below it. In the plot corresponding to $L = 300$, the trace for $\alpha = 0.49$ begins early but never reaches 100% probability of detection because sample support is still insufficient to mitigate the spurious peaks

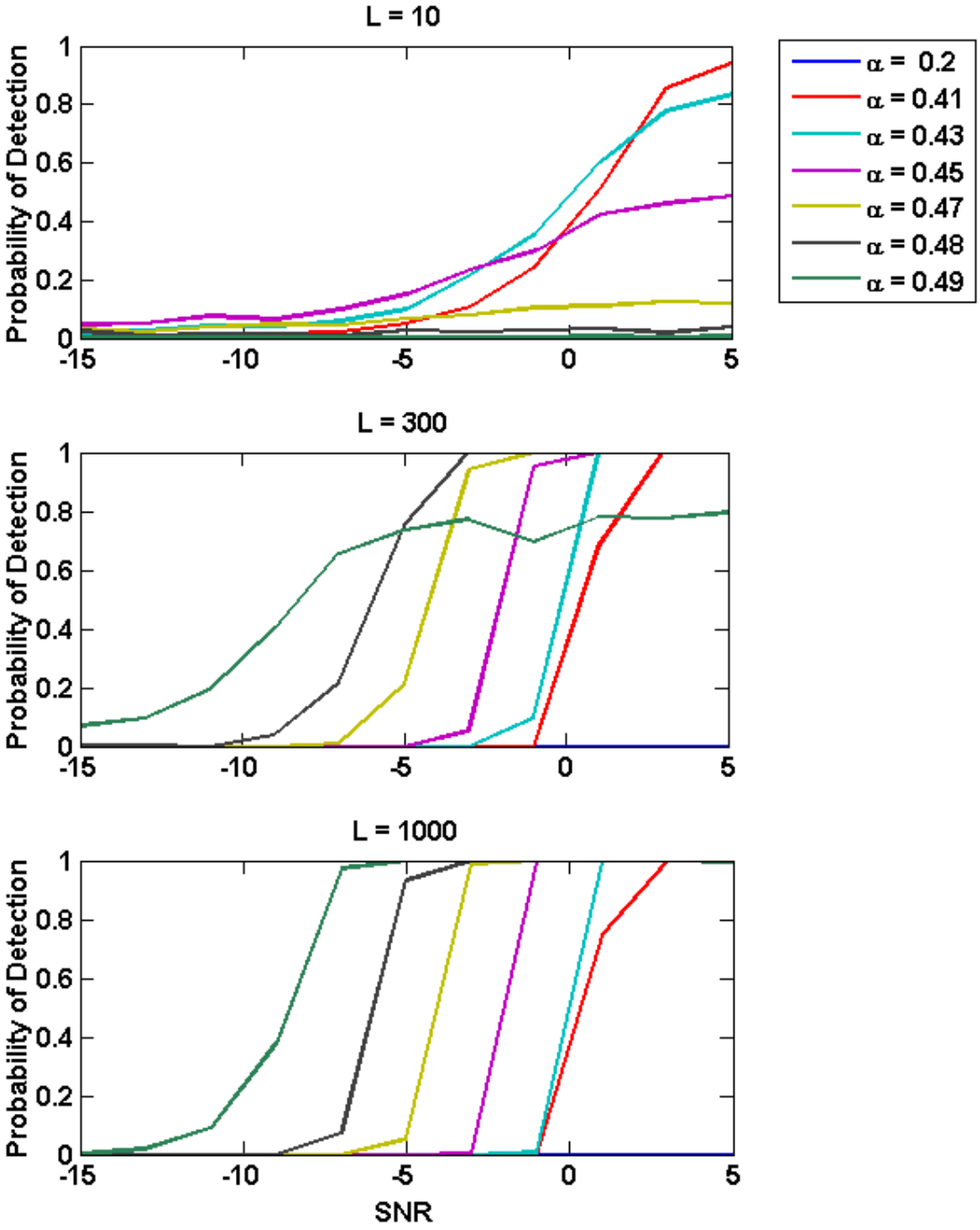

Figure 6.2: The probability of detection as a function of SNR for several values of α and L.

associated with high values of α.

To gain further insight into the nature of the detection problem, another Monte Carlo simulation was created that tabulates the probability of each kind of detection outcome, as described below:

1. The true signal is detected (false peaks may or may not be present);
2. False peak(s) are detected (true peak may or may not have been detected);
3. Both true signal and false peaks are detected (may or may not be ruled failed detection depending on the detector used);
4. Only the true signal is detected (successful detection);
5. Only false peak(s) are detected;
6. No signals detected at all.

Here, we deem a trial a successful detection when solely the true signal was detected. It may be possible to design a detector that can distinguish between false peaks and the true signal in some cases, but we did not consider that possibility in this simulation. At each permutation in the simulation 500 trials were run, and during each peak detection was performed following the completion of PC-RISR. The true signal was presented at 0°, so if a peak was found outside of half a bandwidth from the boresite, or if it was more than 10 dB less than the true signal, it was automatically ruled a false peak. The first peak to fall within a half bandwith of the boresite is ruled the true signal, and any subsequent peaks with half a bandwidth are marked false peaks. After each trial, the detection outcome is tabulated and recorded, and a the end of the 500 trials, before proceeding to the next permutation, the probability of each of the six outcomes listed above is calculated, as well as the average number of false peaks that occurred. The results are presented in Figures 6.3 - 6.8.

One important conclusion that can be drawn from Figure 6.8 is that the number of spurious peaks PC-RISR produces is a function only of L and α — it is independent of the SNR. Spurious

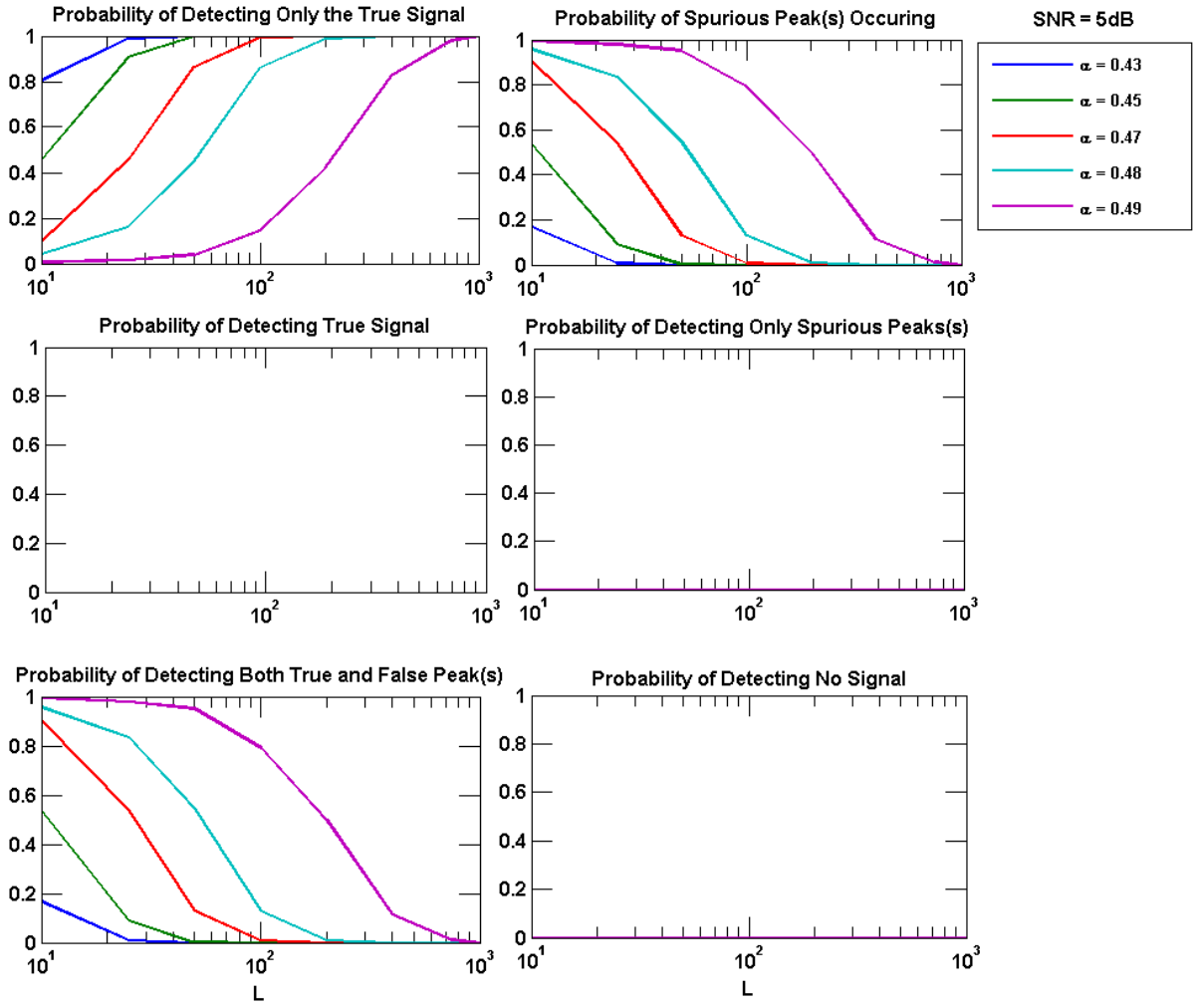

Figure 6.3: The probability of each detection outcome at an SNR of 5dB

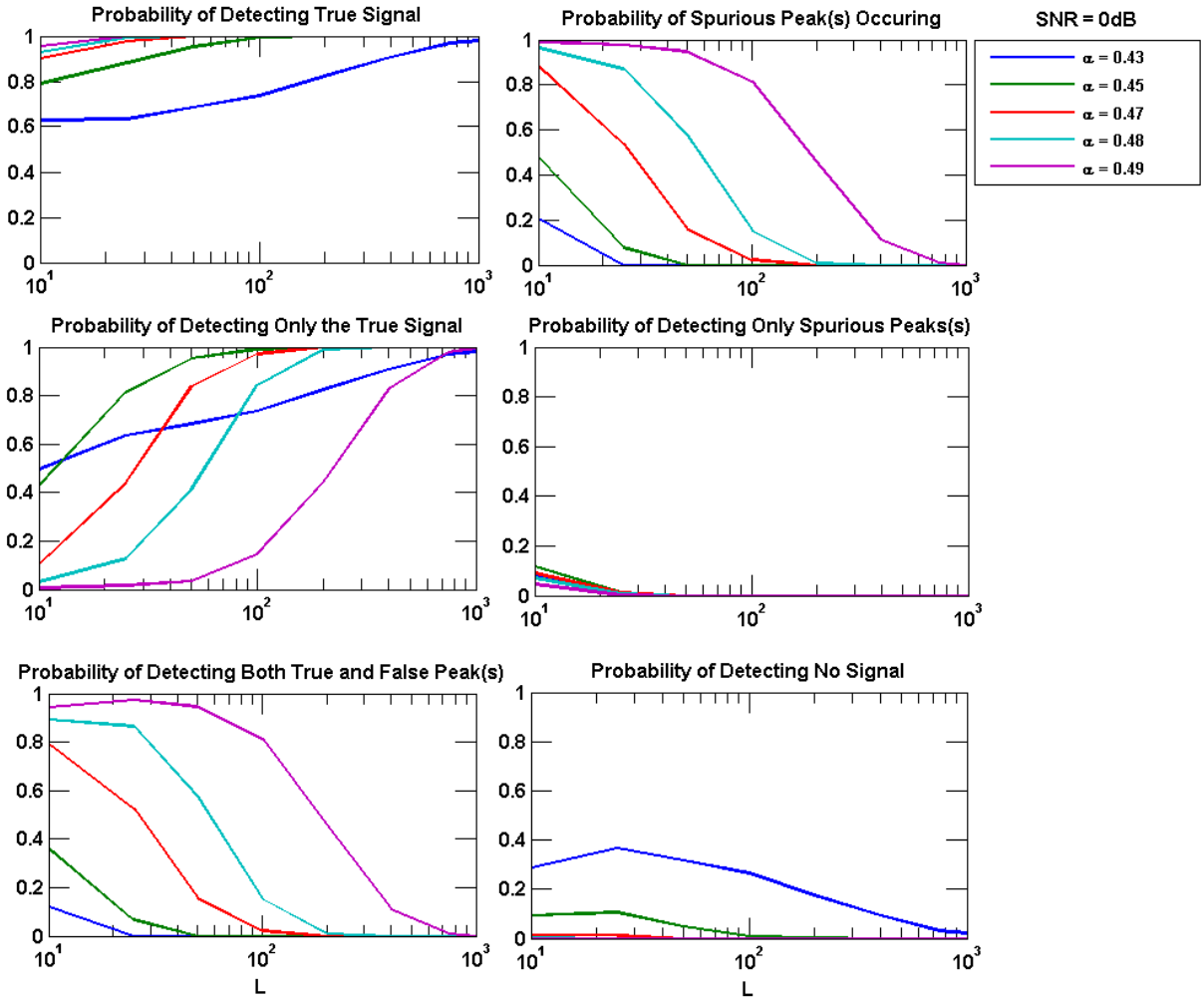

Figure 6.4: The probability of each detection outcome at an SNR of 0dB

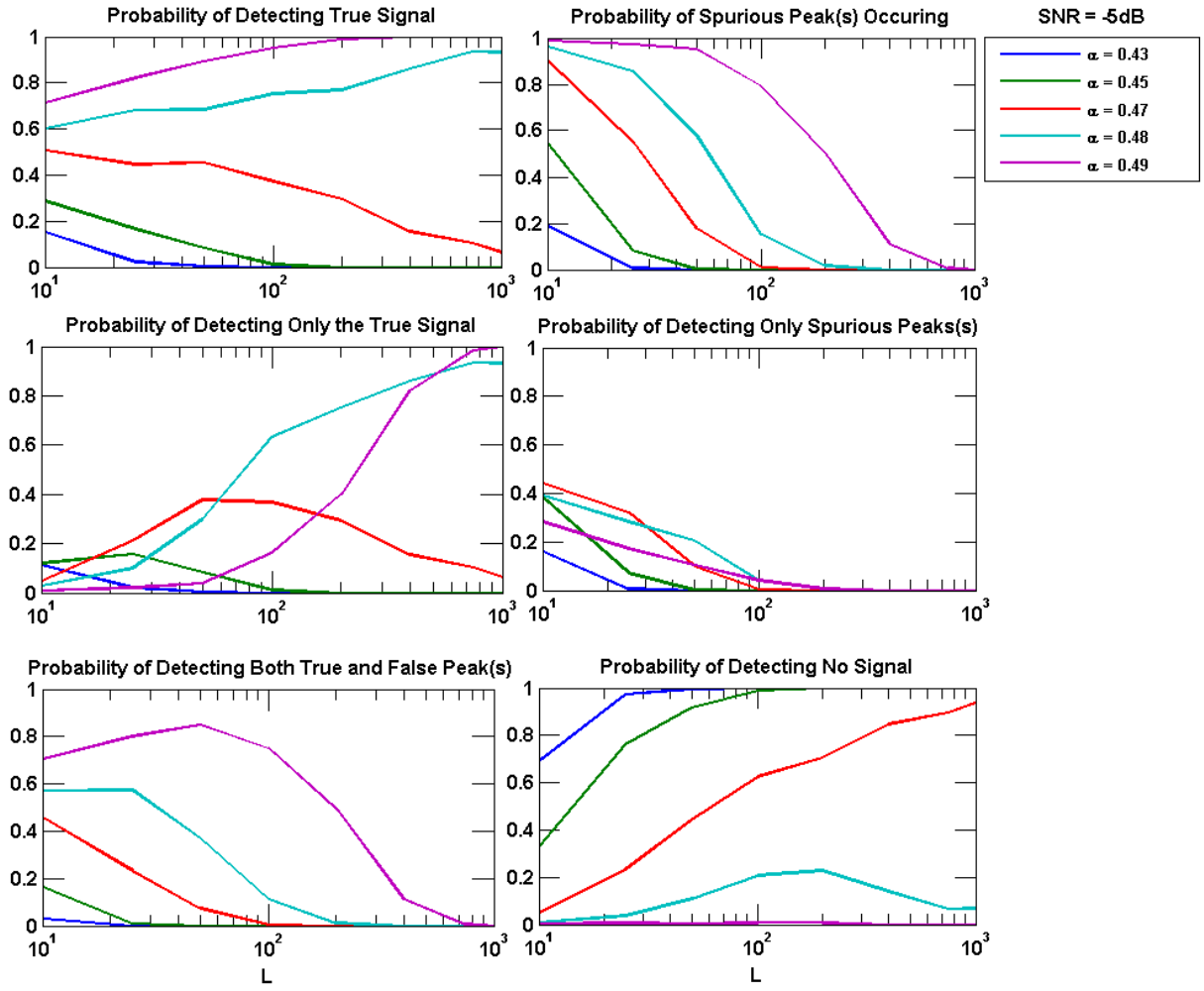

Figure 6.5: The probability of each detection outcome at an SNR of -5dB

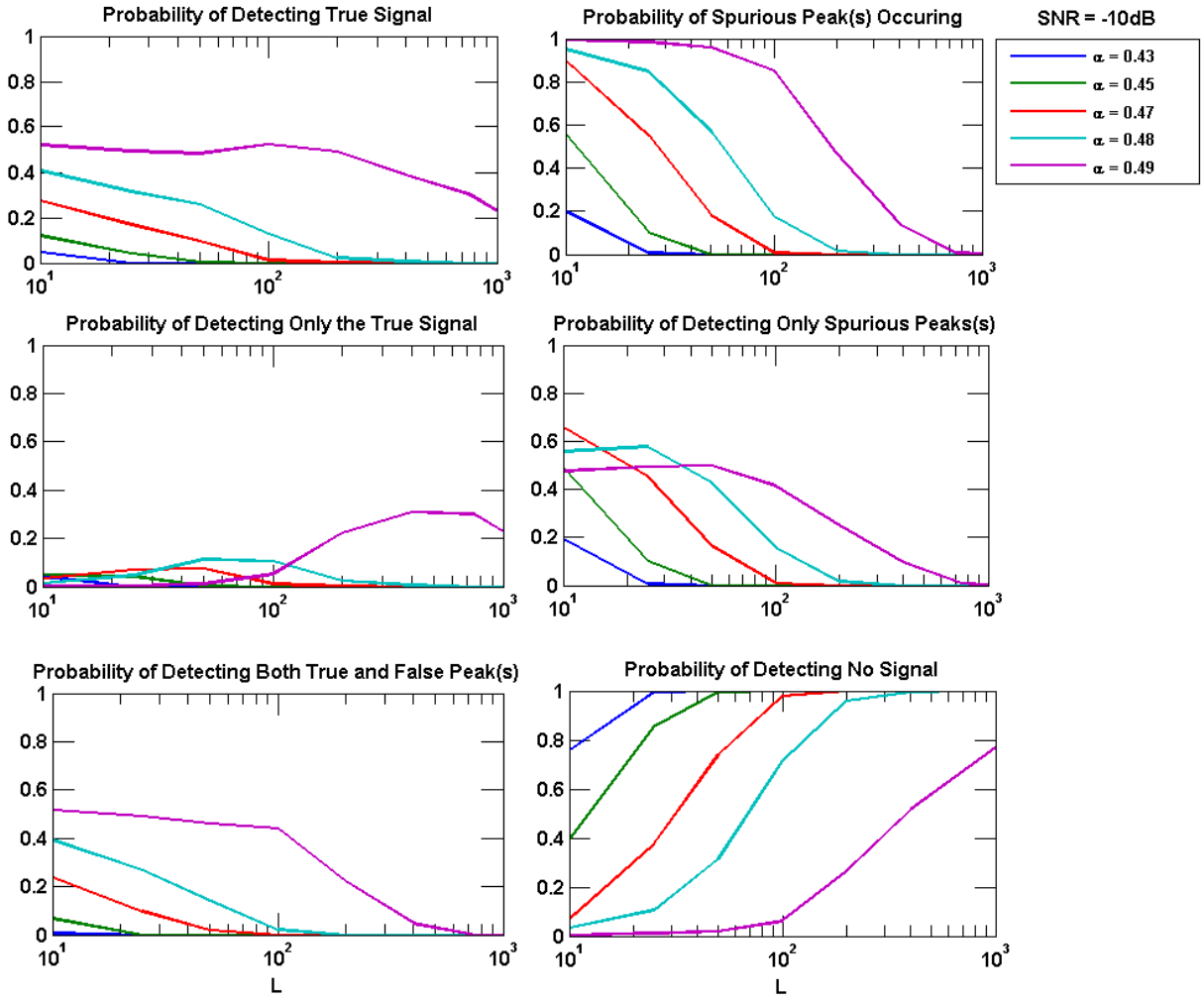

Figure 6.6: The probability of each detection outcome at an SNR of -10dB

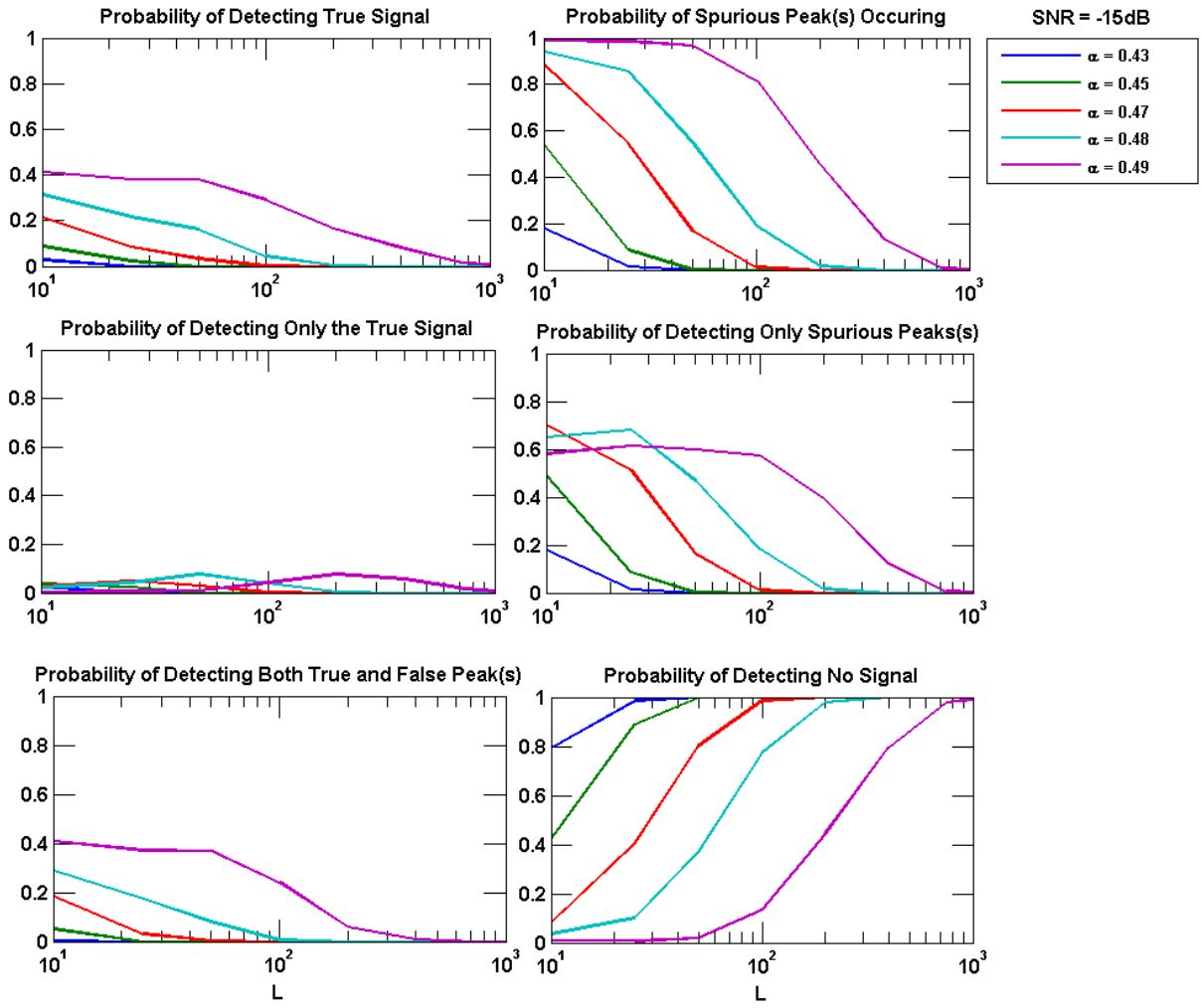

Figure 6.7: The probability of each detection outcome at an SNR of -15dB

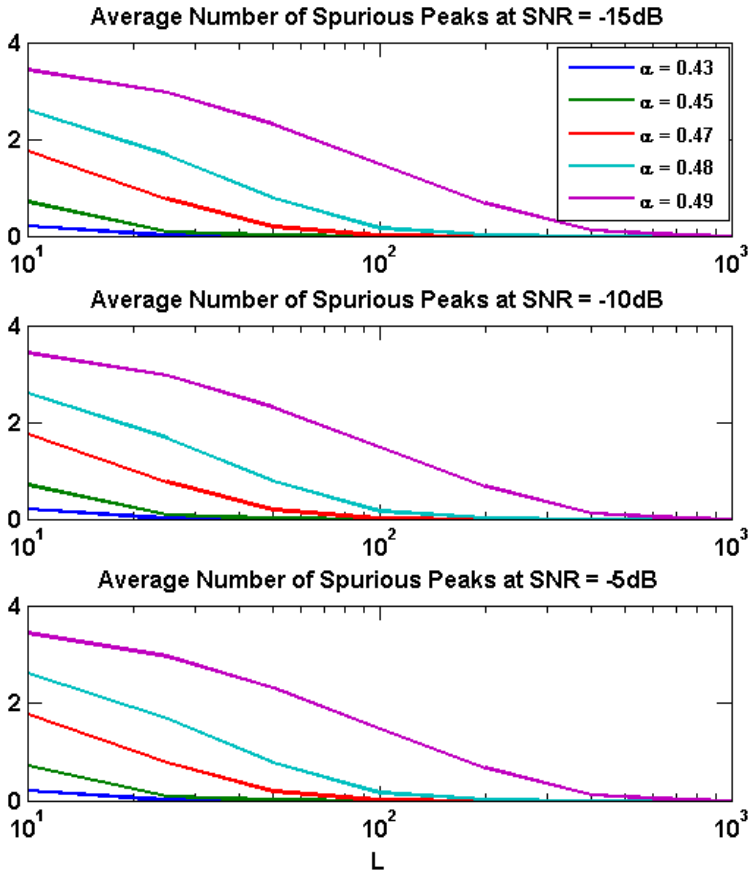

Figure 6.8: The average number of spurious peaks produced by PC-RISR as function of sample support L and geometric weighting α. The average number of spurious is independent of SNR.

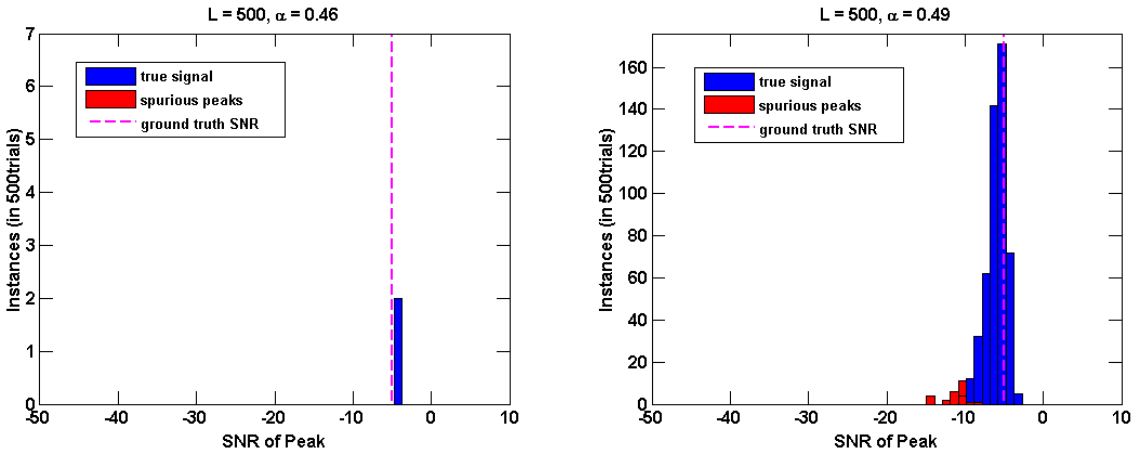

Figure 6.9: Histogram of true signal and spurious peak values over 500 trials at -5dB. $\alpha = 0.46$ never results in a false alarm, but has many missed detections.

peaks are the result of noise and calibration error. Calibration error has a multiplicative effect, so high SNR spurious peaks are associated with calibration error, and at low SNR they are associated with noise.

Another good way to examine these results is by employing a histogram to approximate statistical distribution. In the following simulations, one signal was presented on the bore sight for detection, and using the same logic as previously, each peak was ruled either the true signal or a false peak and recorded, and the results used to create the histograms shown in Figure 6.9 - 6.12. Note that for each trial there can be at most one true signal, but there may be several false peaks.

Observe from the results in Figure 6.9 - 6.12 that regardless of signal SNR, choice of α, and number of samples, spurious peaks tend to have about the same power. Furthermore, the number of spurious peaks drops off with increasing number of snapshots, and finally, lowering alpha reduces the number of false peaks detected, but also lowers the probability of detecting the true peak.

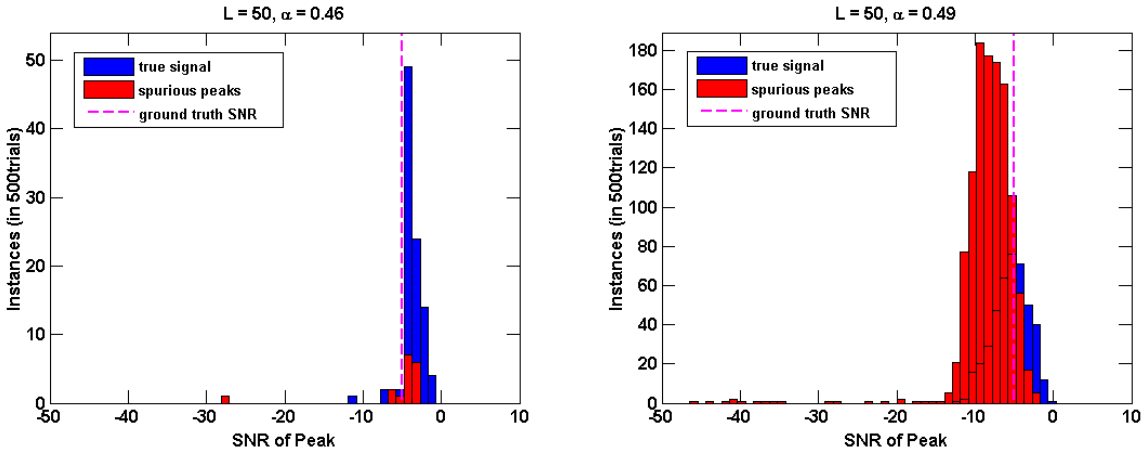

Figure 6.10: Histogram of true signal and spurious peak values over 500 trials at -5dB. There is insufficient sample support to use $\alpha = 0.49$.

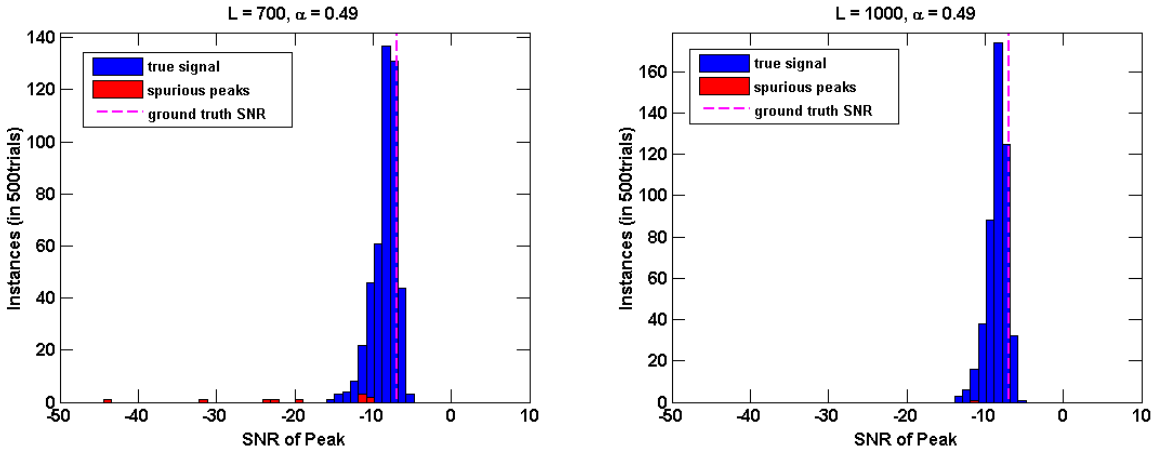

Figure 6.11: Histogram of true signal and spurious peak values over 500 trials at -7dB.

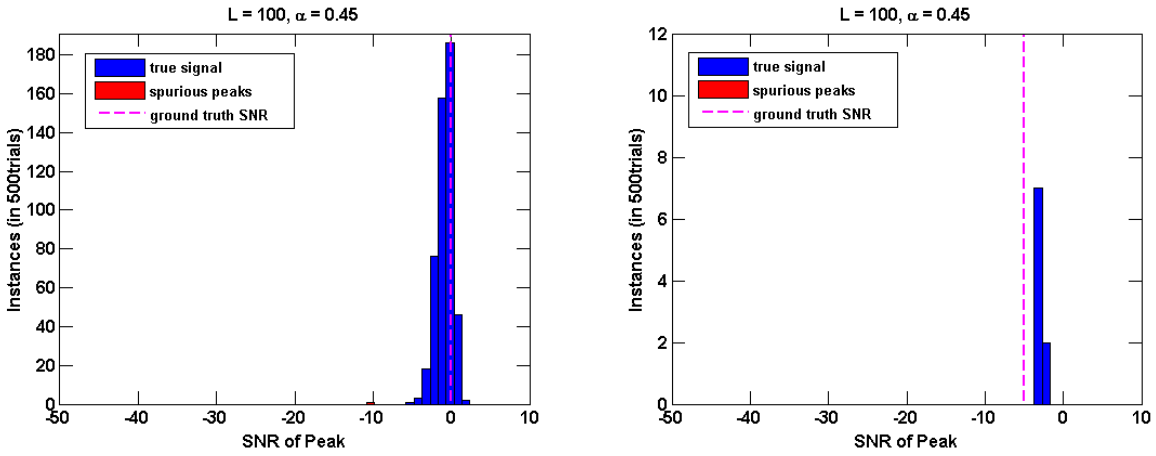

Figure 6.12: Histogram of true signal and spurious peak values over 500 trials at 0 dB (left) and -5 dB (right). No false peaks are detected because α is not too high, but it also misses more peaks as SNR is reduced.

Chapter 7

Conclusions and Future Work

Using a variety of simulations to analyze PC-RISR from many different angles, it was shown that there exists regimes in which it is superior to RISR and MUSIC in every respect except for computational cost, though with a modest array size and limited sample support PC-RISR can still be implemented in real time. In addition to providing superior super-resolution at high SNR, it was shown that it can perform super-resolved DOA estimation at single digit SNR values even with limited sample support. Furthermore, with sufficient sample support, it can function even at negative SNR values. PC-RISR was shown to have extremely good dynamic range and applications in detection were briefly explored.

The use of PC-RISR in the point solution regime ($\alpha < 0.5$) to perform super-resolved DOA estimation and detection was given through treatment, and the potential of using the constraint dominant regime ($\alpha > 0.5$) was explored to model angularly distributed sources. While the point solution regime showed promising results, the constraint dominant regime's tendency to underestimate signal powers is not well understood, nor has a method for compensation yet been discovered, leaving its usefulness an open question. Further insight into the constraint dominant regime is necessary, but the highly non-linear nature of PC-RISR makes mathematical analysis difficult.

PC-RISR's primary drawback is it's computational cost. One of the only variables not analyzed in detail here is the array size N. As mentioned above, using large N results in taking the inverse of

a sizable matrix, which is a computational burden that prohibits the kind of thorough Monte Carlo simulations used to characterize the influence of other variables. Partial simulations to date with modest array sizes show that performance tends to increase slightly with the number of antennas, however the high computation cost of PC-RISR makes performing simulations with larger arrays challenging.

The inclusion of array modeling errors in PC-RISR's signal model yields quite high fidelity simulations, but PC-RISR still needs to be verified on real data. Simulations presented here all assumed a uniform linear array and additive white Gaussian noise. How PC-RISR's performance is affected by other varieties of manifolds and colored noise has yet to be simulated or tested.

References

[1] A.J. Barabel. Improving the resolution performance of eigenstructure- based direction-finding algorithms. *IEEE Intl. Conf. Acoustics, Speech, and Signal Processing*, pages 336–339, apr 1983.

[2] S. Blunt, W. Dower, and K. Gerlach. Hybrid adaptive receive processing for multistatic radar. In *Computational Advances in Multi-Sensor Adaptive Processing, 2007. CAMPSAP 2007. 2nd IEEE International Workshop on*, pages 5–8, Dec 2007.

[3] S. Blunt, W. Dower, and K. Gerlach. Hybrid interference suppression for multistatic radar. *Radar, Sonar Navigation, IET*, 2(5):232–333, October 2008.

[4] S. Blunt and K. Gerlach. A generalized formulation for adaptive pulse compression of multistatic radar. In *Sensor Array and Multichannel Processing, 2006. Fourth IEEE Workshop on*, pages 349–353, July 2006.

[5] S. Blunt and K. Gerlach. Multistatic adaptive pulse compression. *Aerospace and Electronic Systems, IEEE Transactions on*, 42(3):891–903, July 2006.

[6] S. Blunt, K. Gerlach, and E. Mokole. Pulse compression eclipsing-repair. In *Radar Conference, 2008. RADAR '08. IEEE*, pages 1–5, May 2008.

[7] S. Blunt and T. Higgins. Achieving real-time efficiency for adaptive radar pulse compression. In *Radar Conference, 2007 IEEE*, pages 116–121, April 2007.

[8] S. Blunt and T. Higgins. Dimensionality reduction techniques for efficient adaptive radar pulse compression. *IEEE Trans. Aerospace & Electronics Systems*, 46(1):349–362, jan 2010.

[9] S. Blunt, A. Shackelford, K. Gerlach, and K. Smith. Doppler compensation amp; single pulse imaging using adaptive pulse compression. *Aerospace and Electronic Systems, IEEE Transactions on*, 45(2):647–659, April 2009.

[10] S. Blunt, K. Smith, and K. Gerlach. Doppler-compensated adaptive pulse compression. In *Radar, 2006 IEEE Conference on*, pages 6 pp.–, April 2006.

[11] K. Gerlach and S. Blunt. Radar pulse compression repair. *Aerospace and Electronic Systems, IEEE Transactions on*, 43(3):1188–1195, July 2007.

[12] K. Gerlach, A. Shackelford, and S. Blunt. A novel approach to shared-spectrum multistatic radar. In *Radar, 2006 IEEE Conference on*, pages 7 pp.–, April 2006.

[13] K. Gerlach, A. Shackelford, and S. Blunt. Combined multistatic adaptive pulse compression and adaptive beamforming for shared-spectrum radar. *Selected Topics in Signal Processing, IEEE Journal of*, 1(1):137–146, June 2007.

[14] H. Van Trees. *Optimum Array Processing*. New York: Wiley, 2002.

[15] D. Henke, P. McCormick, S. D. Blunt, and T. Higgins. Practical aspects of optimal mismatch filtering and adaptive pulse compression for fm waveforms. In *Radar Conference (RadarCon), 2015 IEEE*, pages 1149–1155, May 2015.

[16] J.E. Evans, J.R. Johnson, and D.F. Sun. Application of advanced signal processing techniques to angle of arrival estimation in atc navigation and surveillance systems. Technical Report Report 582, MIT Lincoln Lab, jun 1982.

[17] K. Gerlach, A. Shackelford, and S.D. Blunt. Combined adaptive pulse compression and adaptive beamforming for multistatic shared-spectrum radar. In *2nd International Waveform Diversity and Design Conference*, Lihue, Hawaii, jan 2006.

[18] K. Gerlach, A. Shackelford, and S.D. Blunt. Single pulse imaging. In *2nd International Waveform Diversity and Design Conference*, Lihue, Hawaii, jan 2006.

[19] M. Popescu, S.D. Blunt, and T. Chan. Magnetoencephalography source localization using the source affine image reconstruction (saffire) algorithm. *IEEE Trans. Biomedical Engineering*, 42(2):572–584, jul 2010.

[20] P. Chen, T.J. Wu, and J. Yang. A comparative study of model selection criteria for the number of signals. *IET Radar, Sonar & Navigation*, 2(3):180 – 188, 2008.

[21] P. McCormick, J. Jakabosky, S.D. Blunt, C. Allen, and B. Himed. Joint polarization/waveform design and adaptive receive processing. *IEEE International Radar Conference*, may 2015.

[22] R. Roy and T. Kailath. Esprit-estimation of signal parameters via rotational invariance techniques. *Acoustics, Speech and Signal Processing, IEEE Transactions on*, 37(7):984–995, Jul 1989.

[23] S.D. Blunt and K. Gerlach. A novel pulse compression scheme based on minimum mean-square error reiteration. In *IEEE International Conference on Radar*, Adelaide, Australia, sep 2003.

[24] S.D. Blunt and K. Gerlach. Adaptive repair of pulse-compressed radar waveforms: Seeing the trees despite the forest. In *1st International Waveform Diversity and Design Conf.*, Edinburgh, Scotland, nov 2004.

[25] S.D. Blunt and K. Gerlach. Joint adaptive pulse compression to enable multi-static radar. In *1st International Waveform Diversity and Design Conf.*, Edinburgh, Scotland, nov 2004.

[26] S.D. Blunt and K. Gerlach. Adaptive pulse compression radar repair processing. In *IEEE International Radar Conf.*, Arlington, VA, may 2005.

[27] S.D. Blunt and K. Gerlach. Aspects of multistatic adaptive pulse compression. In *IEEE International Rada Conf.*, Arlington, VA, may 2005.

[28] S.D. Blunt and K. Gerlach. Adaptive pulse compression via mmse estimation. *IEEE Trans. Aerospace & Electronic Systems*, 42(2):572–584, jul 2006.

[29] S.D. Blunt, T. Chan, and K. Gerlach. A new framework for direction-of- arrival estimation. *IEEE Sensor Array & Multichannel Processing Workshop*, jul 2008.

[30] S.D. Blunt, T. Chan, and K. Gerlach. Robust doa estimation: the reiterative super resolution (risr) algorithm. *IEEE Trans. Aerospace & Electronic Systems*, 47(1):332–346, jan 2011.

[31] A. Shackelford, J. de Graaf, S. Talapatra, S. Blunt, and K. Gerlach. Adaptive pulse compression: Preliminary experimental measurements. In *Radar Conference, 2007 IEEE*, pages 234–237, April 2007.

[32] A. Shackelford, J. de Graaf, S. Talapatra, K. Gerlach, and S. Blunt. Shared-spectrum multistatic radar: Preliminary experimental results. In *Waveform Diversity and Design Conference, 2007. International*, pages 443–447, June 2007.

[33] T. Higgins, S.D. Blunt, and K. Gerlach. Gain-constrained adaptive pulse compression via an mvdr framework. *IEEE Radar Conf.*, may 2009.

[34] T. Shan, M. Wax, and T. Kailath. On spatial smoothing for direction-of- arrival estimation of coherent signals. *IEEE Trans. Acoustics, Speech, and Signal Processing*, ASSP-33(4):806–844, aug 1985.

[35] T. Yardibi, J. Li, and P. Stoica. Nonparametric and sparse signal representations in array processing via iterative adaptive approaches. In *Signals, Systems and Computers, 2008 42nd Asilomar Conference on*, pages 278–282, Oct 2008.

[36] T. Yardibi, J. Li, P. Stoica, M. Xue, and A. Baggeroer. Source localization and sensing: A

nonparametric iterative adaptive approach based on weighted least squares. *Aerospace and Electronic Systems, IEEE Transactions on*, 46(1):425–443, Jan 2010.