Project Report
LSP-270

Beamforming with Distributed Arrays: FY19 RF Systems Line-Supported Program

K.W. Forsythe
L. Collins
H. Naumer

07 January 2020

Lincoln Laboratory
MASSACHUSETTS INSTITUTE OF TECHNOLOGY
LEXINGTON, MASSACHUSETTS

This report is the result of studies performed at Lincoln Laboratory, a federally funded research and development center operated by Massachusetts Institute of Technology. This material is based upon work supported by the Under Secretary of Defense for Research and Engineering and the United States Air Force under Air Force Contract No. FA8702-15-D-0001. Any opinions, findings, conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the Under Secretary of Defense for Research and Engineering, or the United States Air Force.

Massachusetts Institute of Technology
Lincoln Laboratory

Beamforming with Distributed Arrays: FY19 RF Systems Line-Supported Program

K. W. Forsythe
Group 62

L. Collins
H. Naumer
Formerly Group 62, Summer Students

Project Report LSP-270

07 January 2020

Lexington Massachusetts

This page intentionally left blank.

ABSTRACT

Spatial beamforming using distributed arrays of RF sensors is treated. Unlike the observations from traditional RF antenna arrays, the distributed array's data can be subjected to widely varying time and frequency shifts among sensors and signals. These shifts require compensation upon reception in order to perform spatial filtering. To perform beamforming with a distributed array, the complex-valued observations from the sensors are shifted in time and frequency, weighted, and summed to form a beamformer output that is designed to mitigate interference and enhance signal energy. The appropriate time-frequency shifts required for good beamforming are studied here using several different methodologies.

This page intentionally left blank.

TABLE OF CONTENTS

TABLE OF CONTENTS
(Continued)

Page

LIST OF FIGURES

LIST OF FIGURES
(Continued)

LIST OF FIGURES

(Continued)

LIST OF FIGURES
(Continued)

LIST OF FIGURES
(Continued)

LIST OF FIGURES

(Continued)

LIST OF FIGURES
(Continued)

LIST OF FIGURES
(Continued)

1. INTRODUCTION AND EXECUTIVE SUMMARY

1.1 PROBLEM SETTING AND CHALLENGES

We treat spatial beamforming using distributed arrays of RF sensors. Unlike the observations from traditional RF antenna arrays, the distributed array's data can be subjected to widely varying time and frequency shifts among sensors and signals. These shifts require compensation upon reception in order to perform spatial filtering. To perform beamforming with a distributed array, the complex-valued observations from the sensors are shifted in time and frequency, weighted, and summed to form a beamformer output that is designed to mitigate interference and enhance signal energy. The appropriate time-frequency shifts required for good beamforming are studied here using several different methodologies.

Adaptive beamforming for wideband signals suffers from a lack of null depth due to signal decorrelations between sensors and source and/or sensor dynamics. Signal decorrelations occur when the sensors are separated by spatial intervals that result in differential delays in the signal arrival times that are comparable to the reciprocal bandwidth of the signal. For closely spaced sensors, decorrelation occurs with large signal bandwidths. For distributed arrays such as UAVs, decorrelation can occur even with small system bandwidths due to the large separations of sensors.

In addition to decorrelation caused by wide signal bandwidths and/or large sensor separations, source or sensor motion can lead to decorrelation over time that significantly degrades null depth. For linear motion models, the dominant short-term effect of motion is differential Doppler between sensors for each signal. This Doppler can be well-modeled as a frequency shift of the signal modulation, as long as the fractional bandwidth of the RF signal is small. When the sensors are closely spaced and moving together, the differential Doppler is small. From the viewpoint of beamforming, the common Doppler shift can be attributed exclusively to the source and, thus, has only an incidental effect on beamforming (e.g., a demodulator in the loop must account for a Doppler shift).

When differential Doppler is not a problem, signal processing that handles large delay spreads can take several forms. One architecture, space-time adaptive processing (STAP, [War98]) Fig. 1, utilizes tapped delay lines to create correlations between signals at different sensors with different delays. These correlations can be used to suppress interference and maintain gain, or even equalize a desired signal. Taps are often spaced at one-half sample spacings, given Nyquist sampling, to allow for interpolation. An alternative architecture Fig. 2 processes the signals through a filter bank that cuts them into subbands that are narrow enough to avoid significant sensor-to-sensor decorrelation. This architecture has better scaling properties to large differential delay spreads than does the STAP architecture. However, channelized architectures require reconstruction of signals from the beamformed subband outputs.

With distributed arrays, differential Dopplers can be significant and different for each signal. Neither the STAP nor the channelized architectures are able to cope with large differential delay and differential Doppler spreads. It is possible to elaborate on the STAP architecture of Fig. 1 as shown in Fig. 3. Each tap is associated with a time delay and a time-domain modulation, which is typically a frequency shift, although the modulation can also be more general. For example, derivative-based

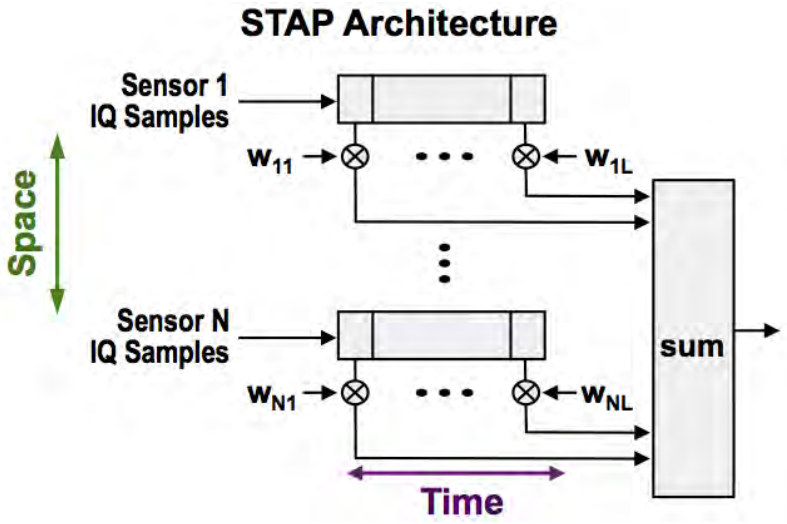


Figure 1. For space-time adaptive processing (STAP), the beamformer combines scaled, delayed outputs from all sensors. Each tap at each antenna element is associated with a particular delay. Complex-scaled versions of the tap outputs are added to mitigate interference and equalize the desired signal.

updates (DBU) utilize a linear ramp. With frequency shifting as the time-domain weight behavior, this signal processing architecture is called space-time-frequency adaptive processing (STFAP).

A more practical approach to beamforming can involve a combination of the channelized and STFAP architectures. More precisely, a filter bank is used to divide the band into subbands.

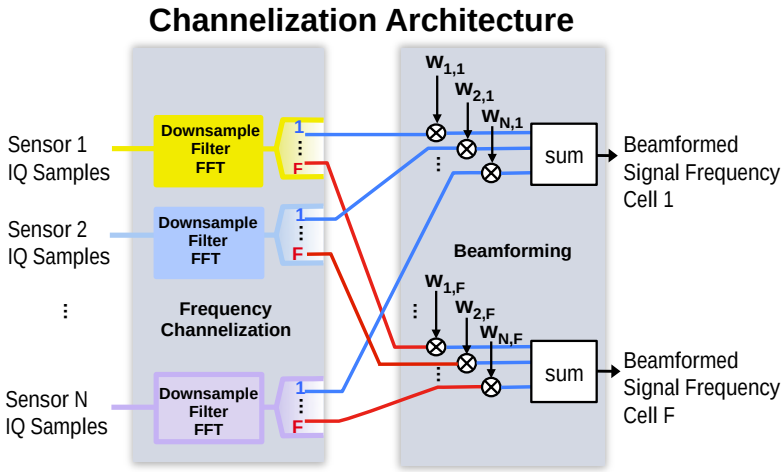


Figure 2. As an alternative to the tapped delay line architecture of Fig. 1, the signals can be passed through a filter bank, which cuts them into narrow subbands. The subband bandwidth is chosen so that signals do not decorrelate between sensors significantly within the subband. Beamforming occurs within each subband. The full signal must be stitched together after beamforming.

In each subband, a STFAP architecture is used for beamforming in the subband. Subbanding is effective in handling delay spreads, but, as the subband size decreases, the sensitivity to differential Doppler increases within the subband. For example, as the subband size decreases, the separation of Nyquist samples increases, allowing the Doppler shifts to have a large effect on smaller numbers of samples. By combining the channelized and STFAP architectures, there is an opportunity for a better beamformer. It is also possible to elaborate on the combined architectures by allowing beamforming to take into account correlations between nearby subbands. This topic will not be considered here.

In Fig. 4, we treat an example of beamforming using two sensors and a dense array of delay-Doppler taps (STFAP) applied to a subband. There is only a single interferer. The only requirement is that the number of samples available in the subband of bandwidth B over an interval of time T be larger than the number of unknown complex tap coefficients (the number of delay-Doppler taps). All delay-Doppler taps are spaced by $\frac{1}{2B} \times \frac{1}{2T}$ to support interpolation. It is assumed that the taps span the delay and Doppler spreads. This is sufficient for this two-sensor problem, but is inadequate, in general. The results show a significant range of (large) delay-Doppler spread products that do not have enough sample support to estimate the unknowns.

The previous example, in effect, uses the unknowns to locate the signal's differential delay and frequency shift. If these parameters were known (at least approximately) ahead of time, fewer taps could be placed appropriately and there would be enough data to estimate the tap coefficients. Such solutions are called sparse tap solutions as opposed to the dense taps solutions treated in Fig. 4.

The bulk of this report is focused on sparse tap placement algorithms and associated aspects of performance assessment. The tap placement for beamforming is based on knowledge of the differential delays and differential Dopplers of the sources. Such information can be obtained by channel probes in cooperative systems and by geolocation techniques otherwise. Since geolocation can take advantage of distributed beamforming, there is a bit of a chicken and egg story here. However, it is possible to geolocate in interference without waveform knowledge.

1.2 NULLING SOLUTIONS

The main product of all tap placement methodologies is the sparse pattern of delay and Doppler taps that can be used to cancel interference. The tap coefficients are estimated from the data by forming covariance estimates of the environment.

Tap placements can be complicated and highly nonintuitive when there is more than a single interferer. It is clear that a single interferer can be aligned among sensors using a single delay-Doppler tap at each sensor. The interferer can then be cancelled in multiple ways, using, for example, different sensor pairs. With more than a single interferer, it is not clear how to find tap patterns that align all interferers in a manner that allows cancellation with a linear combination of the tap outputs. This is not, in fact, difficult to show when only delay spreads or only Doppler spreads are present. The combination of delay and Doppler spreads complicates tap placement.

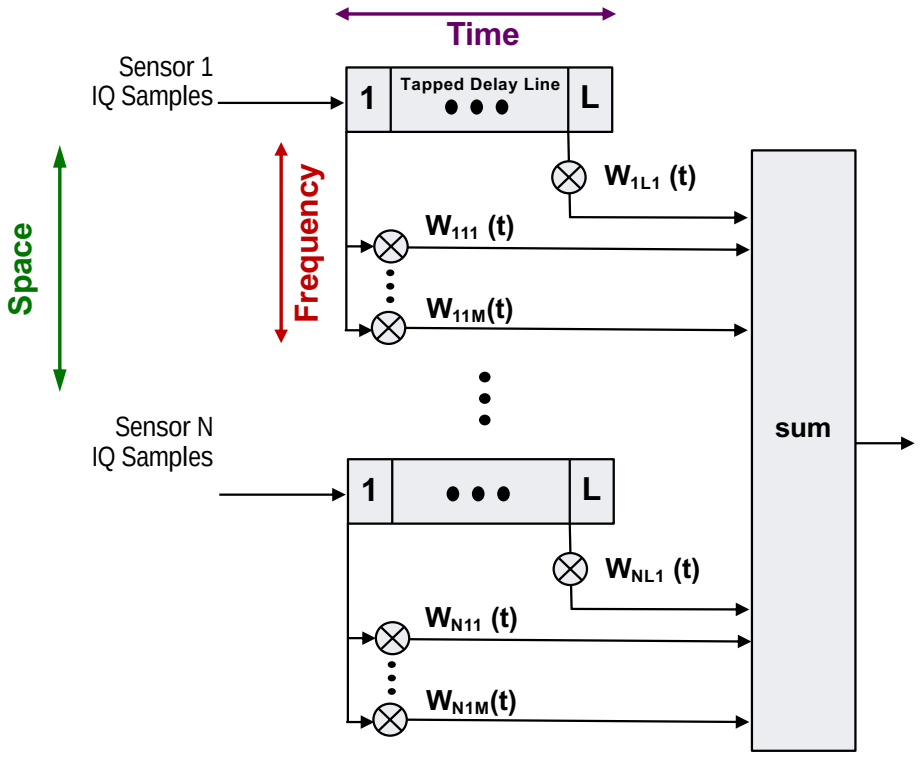


Figure 3. The beamformer combines scaled, delayed, and frequency shifted outputs from all sensors. Each tap at each antenna element is associated with a particular delay and, in the case of dense taps, with multiple frequency shifts. The diagram illustrates the frequency shifts as time-dependent weights. This model also allows other time-domain weight behaviors. The tap outputs are added to mitigate interference and, as required, equalize the desired signal.

We investigate several methods for placing delay-Doppler taps. One method utilizes mutual information to place taps in a greedy manner. This is not an optimal procedure, but one that works well in practice. As somewhat of a digression, we also illustrate the application of this approach to the design of narrowband antenna arrays.

The other methods we consider for tap placement all rely on nulling, also called zero-forcing here. Taps are placed to support nulling of all interference signals. One approach taken here utilizes compressive sensing to find sparse solutions of an underdetermined nulling equation. Another approach formulates zero-forcing as a problem in noncommutative polynomials (i.e., Ore rings) that is addressed by using the algebraic machinery of noncommutative Groebner bases and syzygies. The last approach, which is just sketched here, solves a nulling equation using Gaussian elimination over fractions based on Ore polynomials, clearing denominators at each step using noncommutative common multiple constructions. All approaches have their merits and offer different types of results. Thus the main contribution of this report is a toolbox of techniques for tap placement.

Along with the description of tap placement algorithms, we develop a set of performance measures. Mutual information is required for one tap placement procedure. Based on this we develop an effective SNR that has more intuitive appeal since it can be used to express an SNR loss. We also develop a capacity expression for certain channels with delay and Doppler spreads.

As an example of the performance benefits of sparse delay-Doppler taps, Fig. 5 illustrates the SNR loss achieved by space-time adaptive processing (STAP), derivative-based update (DBU) STAP, and space-time-frequency adaptive processing (STFAP) in a scenario that involves large delay and Doppler spreads with two interferers. Update intervals must be short if only delay taps, themselves sparse, are employed. Short update intervals do not provide, necessarily, enough data samples to determine tap coefficients. The benefits of sparse-tap STFAP solutions is the availability

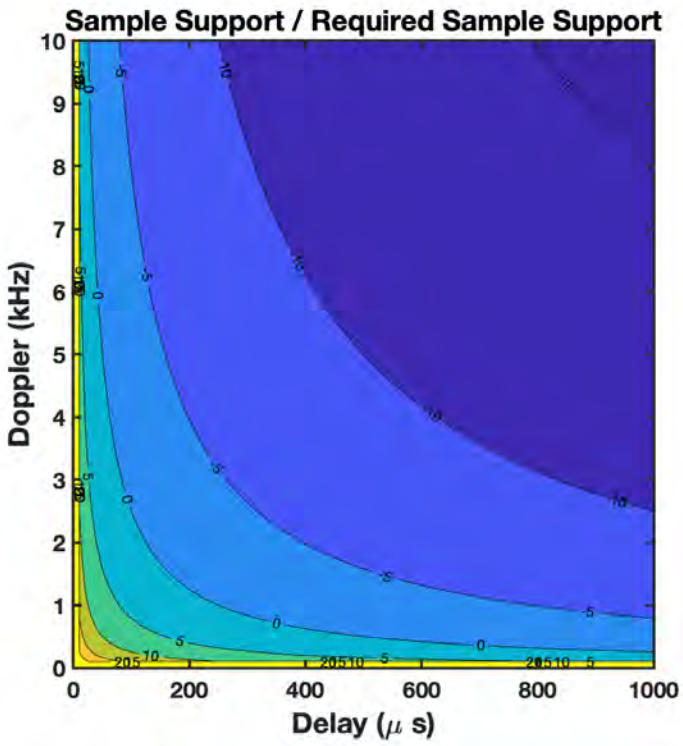


Figure 4. We can plot the sample support (i.e., the number of samples TB observed in the subband of bandwidth B over observation time T) divided by the number of unknowns for a dense delay-Doppler STFAP approach. The latter number is taken to be the number of taps in the delay spread, sampled at twice Nyquist, times the number of taps in the Doppler spread, sampled at twice the frequency resolution cell $1/T$. Of course, this assumes that taps need not be placed beyond the boundaries formed by delay and Doppler spreads. This is not always the case, as we will see when there is more than one signal nulled, but it is the smallest number of dense taps spaced thusly that can align the signal in delay and Doppler for the two-sensor case. We can see in the figure that the sample support is inadequate for a large range of combined delay and Doppler spreads no matter what subband bandwidth is used for array processing.

of ample data for estimating tap weights. This is achieved, in the case of large delay and Doppler spreads, by reducing the number of taps required for nulling to a level that does not scale with the delay and Doppler spreads.

Next, we discuss some previous work and present somewhat more detail on the tap placement methodologies.

Previous work using mutual information for antenna array design includes [SJW17] and [SJW19], where submodularity is demonstrated and used to provide performance guarantees. Submodularity is an analog of convexity for functions defined on finite sets. In this case, the function is a type of mutual information among the members of the set (e.g., antenna elements). The benefit of submodularity is that a greedy approach to optimization is guaranteed to lie above a large fraction of the global optimum. For our applications, submodularity does not hold, but still can motivate the same design procedure and leads, as we show, to reasonable levels of performance as shown in Section 3.

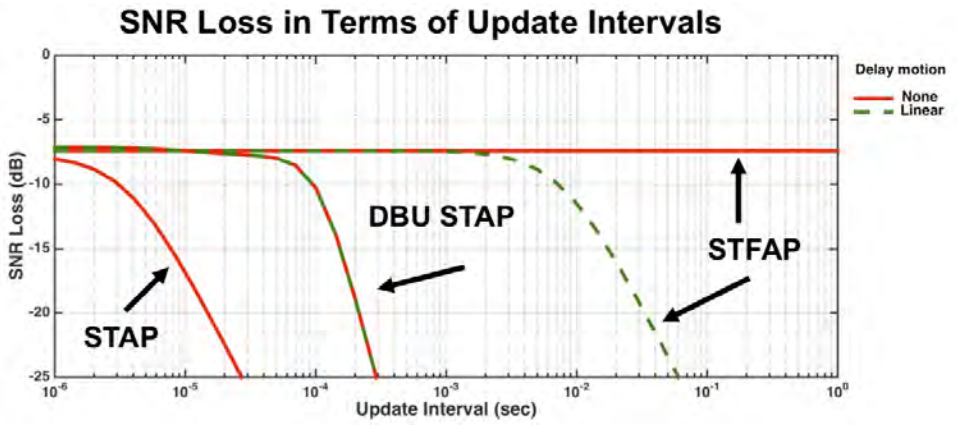


Figure 5. SNR loss is shown for several sparse-tap algorithms. The scenario consistes of two interferers and three sensors. The differential delays spreads are about 1 ms and the differential Doppler is about 1 kHz. This places the delay-Doppler product outside the region that can be handled with dense tap placements. Signal bandwidth is 1 MHz. Although space-time array procesing (STAP) appears to offer good performance with rapid weight updates, not enough samples are available to determine the STAP weights and maintain good performance. With delay-only taps, the performance of STAP weights can be extended by allowing weights dynamics, such as the linear time-varying weights shown for derivative-based update (DBU) STAP. In this case, each time-delayed tap output is weighted additionally with a linearly, time-varying ramp, doubling the number of taps. The useful update intervals increase by much more than a factor of 2 to improve sample support for estimating tap weights. Performance is best when the time variation of the weights matches the time variations in the signals, which is primarily due to Doppler shifts. Thus, space-time-frequency array processing (STFAP) weights offer the largest update intervals. Each tap is associated with a particular delay and Doppler shift. In other words, each delay tap has a time-varying weight that varies as a complex exponential. Performance is stable over all update intervals shown given the Doppler model of signal dynamics. The Doppler model is a consequence of a linear motion model for sources and sensors. This model will eventually require replacement of taps as differential delays change.

Algebraic approaches to nulling have been treated for different applications and under different names in the computer science community. The terminology syzygies used there refers to nulling weights in our context. The machinery for computing these weights utilizes Groebner basis concepts and algorithms, as well as faster, more sophisticated techniques still under development. In [BGTV03], a very thorough treatment of the Groebner basis approach is presented in the generality (noncommutative polynomials) needed here for applications to scenarios with large delays and Doppler spreads. We provide a self-contained treatment of the special scenario in which delay spread (or, dually, Doppler spread) is solely significant. This is used to introduce the concepts basic to the noncommutative Groebner basis approach, which is sketched. These algebraic techniques provide not just examples of nulling weights, but a method for finding all weights that null. In addition, these techniques provide a means for equalizing a desired signal that would otherwise be distorted by a nulling weight. Again, more than a single solution is provided. The algebraic approach provides the means of generating all nulling solutions that equalize a desired signal.

Compressive sensing (CS) has beeen used for channel estimation problems in the past [BWHZ10]. We apply CS techniques [NT09] to find sparse solutions for nulling equations that are built from the known (approximately) differential delay and Doppler shifts of the cochannel signals. CS solutions, unlike the algebraic solutions, are designed to be sparse. This can be both good and bad. The sparsity is good in the sense that complexity is reduced for the beamformer. Sparsity can be bad since gain and, more significantly, the ability to equalize the desired signal can be adversely effected. These are not necessarily serious problems with CS approaches, since it is possible to use algebraic techniques for equalization and CS for nulling.

The most obvious approach to solving nulling equations is Gaussian elimination, discussed in Section 4.4. Gaussian elimination is applied to the channel matrix, which characterizes propagation to the sensors. The same procedure can be applied to delay-Doppler problems, where the channel matrix has delay-Doppler operator entries. Unlike the narrowband case, we encounter operator polynomial expressions that, if we follow Gaussian elimination precisely, must be inverted. To surmount this barrier, we use the solution of common multiple problems (e.g., Section 4.2) to clear denominators, in effect, at each stage in the process of elimination, yielding fraction-free triangular matrices with delay-Doppler polynomials as entries.

Nulling of interference is a constraint placed on adaptive beamforming weights which must also provide an output that contains a desired signal. For the fundamental problem of narrowband nulling, the number of signals that can be handled, nulling interferers and providing nonzero gain on the desired signal, equals the number of sensors. This is a consequence of linear algebra. Placing nulls forces linear constraints on the potential beamforming vectors. With one less interferer than sensors, the beamformer weight is completely determined (typically) up to a complex scalar. Thus the signal-to-interference-plus-noise ratio (SINR) is completely determined for the desired signal.

The situation is more complex when delay and Doppler spreads are at play. Again, it is possible to null one less interferer than the number of sensors, but the beamformer no longer has only gain on the desired signal. It also distorts the desired signal with delayed and Doppler shifted replicas. When there are two fewer interferers than sensors, more than one nulling weight can be found. These weights are unrelated in a sense explained later. With only delay spreads (or only

Doppler spreads), there are at least two nulling weights, whose outputs can typically be combined to equalize the desired signal. In other words, we can find a nulling weight that also produces an undistorted replica of the desired signal. Thus, with one more sensor than the number of signals, it is possible to null interference and equalize the desired signal (typically).

With the combination of significant delay and Doppler spreads, three unrelated nulling weights can be combined to find a weight that equalizes. Thus, when there are two more sensors than signals, it is possible to null all interferers while equalizing the desired signal (again, typically).

Although equalization is desirable for legacy signals, it is not necessary to achieve good spectral efficiency utilizing sparse delay-Doppler taps. In fact, equalization tends to fill-in otherwise sparse tap placements, significantly reducing sparsity. Levels of sparsity are reduced to the extent that CS tap placement, which requires the existence of sparse solutions, has difficulty finding equalized solutions.

For modems with adaptive coding matched to the channel characteristics, multiple nulling weights can be used to provide both interference suppression and gain on the desired signal. With strong interference and large, unrelated delay and Doppler spreads, the principal loss to channel capacity due to interference is reflected (for weak signals and approximately) by the loss in sensor degrees of freedom from the number of sensors to the number of sensors minus the number of interferers.

1.3 MERITS OF DIFFERENT TAP PLACEMENT METHODOLOGIES

It is worth making some comments on the strengths and weaknesses of the various tap placement techniques.

It is a challenge for all of the techniques to null large numbers of interferers. Although we don't have good estimates of the number of taps required for arbitrary scenarios, we do have an upper bound on the number of taps required in the delay-only case, namely $n_{\text{sig}}!$, the factorial of the total number of signals. For large delay-Doppler spreads the number appears to be significantly higher. These large numbers are indicative of the complex, combinatorial alignments required to simultaneously null multiple signals.

The CS technique appears to have the best scaling properties but also offers the least in terms of performance guarantees. It does not handle equalization, unless that happens to have a sparse solution. It does not provide more than a single nulling weight. There are remedies for all of these deficiencies, but the remedies are ad hoc.

Tap placement based on mutual information uses a relevant performance metric. It does not require nulling and thus, in principle, should offer better performance. However, greedy placement based on mutual information has no performance guarantees in the presence of interference. Placing nulls on interference may be suboptimal, but in a more predictable way.

Algebraic approaches to tap placement offer not just one or a few nulling weights, but a generating set for *all* nulling weights. This approach also provides the ability to equalize the desired signal. Tap placement is slow with the Buchberger algorithms we implemented, but should be much

faster using more recent approaches (e.g., noncommutative version of Faugere's F4 algorithm) implemented in a few symbolic algebra packages (e.g., Magma Computational Algebra System).

Fraction-free Gaussian elimination (GE) as formulated here requires the repeated solution of noncommutative common multiple problems. The latter can be solved using algebraic approaches (syzygies) or CS approaches (common multiple problems are solutions to linear systems of equations). Gaussian elimination, used for different subarrays, can provide a useful set of nulling weights. It remains to understand the efficiency and performance of GE approaches.

Other approaches to fraction-free Gaussian elimination, in the commutative case, are described in [LS95]. For the noncommutative case, some recent work is described in [LS12], where a number of other references can be found. More generally, in [GGRW05] we find a discussion of the solution of noncommutative linear systems using a theory of quasideterminants and a type of Cramer's rule. These approaches are not pursued here, but could provide interesting alternatives. It's worth mentioning that there are some other array processing applications of noncommutative linear algebra (e.g., [BM04]).

1.4 SECTION WALK-THROUGH

The definitions of delay and Doppler operators and their basic properties are treated in Section 2. Included in these properties are covariance matrices, channel capacities, and effective SNRs, all derived under the assumption of Gaussian signal models. Channel capacities are derived under an additional assumption of cyclostationarity. The calculations presented here are essential for performance assessment and can also be used as a tool for tap placement. As formulated, the propagation channel is assumed to be line-of-sight, with simple delay and Doppler shifts between each source and sensor. More general channel models can be handled using straightforward extensions of the calculations presented. It is worth noting that all of the tap placement algorithms considered here are formulated for this same propagation model, but each placement technique can be extended to handle more general models.

The remainder of this report addresses tap placement procedures. In general terms, tap placement is treated by finding a good adaptive beamformer and observing the delay and Doppler taps it requires. When channel dynamics are well-modeled by delay and Doppler, the specific gains of the propagation channel are not required for tap placement. Thus, it is possible to place taps based only on delay and Doppler shifts. When additional information is available, involving a specific channel, this information can be used by some of the tap placement algorithms, namely the algorithm described in Section 3.

Section 3 discusses an information theoretic criterion that has been used for sensor placement [SJW17], [SJW19]. The utility of this criterion is based on a property called submodularity that is discussed in this section. Submodularity provides guarantees on the optimality of its solutions to sensor placement. However, submodularity does not hold for the tap placement applications that interest us. Nevertheless, in some situations, submodularity has approximate validity. We give some examples of sensor and tap placement using this criterion.

The approach of Section 3 relies on channel capacity or effective adapted SINRs in order to place taps. The remainder of this report focuses on techniques that use another criterion. These techniques null interfering sources. This is commonly called zero-forcing. Section 4 treats zero-forcing in terms of solving linear systems. These systems involve matrices and vectors with operator entries, which complicates their solution. The entries can be interpreted as polynomials in two variables, representing delay and Doppler. However, these variables do not commute with each other. The solution that we desire for adaptive beamforming should involve a weight vector whose components are themselves delay-Doppler polynomials. In Section 4, we illustrate several procedures for arriving at such a solution. Along the way, we provide some examples where a complete solution can be written down. More generally, we illustrate the problems that are encountered in finding a solution. These problems often involve common multiple concepts for delay-Doppler polynomials. Given a solution to these common multiple problems, a modified Gaussian elimination procedure is sketched in Section 4.4 . This procedure finds adaptive beamformers of the type we require. The solution to the beamforming problem also provides the solution to tap placement. The reader can skip Section 4.2.2 on a first reading since it deals largely with an existence proof of common multiples in a very general setting and is not required for the rest of this report.

Section 5 considers an algebraic approach to zero-forcing. Delay-Doppler polynomial algebras can be studied using tools developed for computer algebra systems. After a simpler introduction in Section 5.1 to the algebraic viewpoint in the delay-only case, we provide a very breezy introduction to the methods used for the much more complicated case of delay-Doppler polynomials, which is treated in much more detail in [BGTV03]. These techniques provide not just a method for finding zero-forcing solutions and tap placement, but also provide a set of generators for all zero-forcing solutions. Algebraic techniques also provide a mechanism for equalizing the desired signal in addition to nulling interferers. These methods also provide insight into the construction of sparse (minimal tap) solutions. One should be aware that a minimal zero-forcing solution is not necessarily the most desired solution. Minimizing the number of taps often means minimizing the number of sensors employed for beamforming, throwing away gain on the desired signal.

Finally, in Section 6, we apply compressive sensing techniques to adaptive beamforming and tap placement. Unlike the algebraic, bottom-up solutions to zero-forcing, which create small generator solutions from which one can build all solutions, compressive sensing is a top-down approach which proceeds from an underdetermined set of zero-forcing equations based on a large number of densely placed taps. Compressive sensing is used to find the sparsest (numerically) solution to these underdetermined equations.

2. DELAY-DOPPLER OPERATOR MODEL OF BEAMFORMING

The delays and Doppler shifts of signals arriving at a distributed array can be represented by the application of delay and Doppler operators to a fixed reference signal for each independent source. The algebraic relations between delay and Doppler operators show that these operators do not commute with each other, leading to additional complexities in signal processing.

The receiver architecture that we address primarily is shown in Fig. 3. The taps in this figure represent sensor outputs that are subject to delay and Doppler shifts that involve the application of delay and Doppler operators at each tap. These operators are monomials in delay and Doppler in a sense that will become clear below. The weighted combination of taps in each sensor represents a delay-Doppler polynomial applied to the sensor output. In this sense, the architecture of Fig. 3 can be interpreted as beamforming with beamforming weights that have delay-Doppler polynomial operators as entries in the weights.

The machinery of delay-Doppler operators is developed in this section. Statistics of the sensor outputs are calculated in terms of delay-Doppler operators for stationary, Gaussian signal models. These statistics are use to evaluate channel capacities analytically for certain structured (e.g., cyclostationary) propagation channels. Some useful SNR expressions are derived and will be used in later sections for performance evaluation and, in one case, for receiver design in the sense of tap placement.

Tap placement is the topic of the bulk of this report. It refers, in the context of Fig. 3, to the taps (monomial terms) that are used (have nonzero coefficients) in the combining (applying delay-Doppler polynomials) of sensor outputs. Receiver complexity is impacted significantly by the number of taps required. Furthermore, since signal dynamics require the update of beamformer weights, there must be sufficient samples in the update intervals to determine the tap coefficients (monomial coefficients) from the data. In some cases, this is possible only the taps are sparsely located with respect to a sampling grid in time and frequency.

2.1 DELAY DOPPLER OPERATORS

Assume that the time-varying delay $\delta(t)$ has a linear approximation $\delta(t) \approx \delta_0 + \delta_1 t$. Define the Fourier transform of the signal $s(t)$ by $\hat{s}(\omega) \stackrel{\text{def}}{=} \int_{-\infty}^{\infty} s(t) e^{-i\omega t}\, dt$. Then the Fourier transform of $s(t - \delta(t))$ can be approximated by

$$\begin{aligned} \int_{-\infty}^{\infty} s(t-\delta(t)) e^{-i\omega t}\, dt &\approx \int_{-\infty}^{\infty} s((1-\delta_1)t - \delta_0) e^{-i\omega t}\, dt \\ &= e^{-i\omega \frac{\delta_0}{1-\delta_1}} \int_{-\infty}^{\infty} s(t) e^{-i\frac{\omega t}{1-\delta_1}} \frac{dt}{1-\delta_1} = e^{-i\omega \frac{\delta_0}{1-\delta_1}} \hat{s}\left(\frac{\omega}{1-\delta_1}\right). \end{aligned} \tag{1}$$

We have $\delta_1 = v/c$, where v is the component of radial velocity and c is the speed of light. Denote by $s(t)$ a complex signal of bandwidth B centered about the frequency ω_c. Since $v/c \ll 1$, $|\omega - \omega_c| \leq \pi B$, and since $|\frac{\omega_c}{2\pi B}| \gg 1$ for typical RF signals, we can make the approximation $\frac{\omega}{1-\delta_1} \approx \omega + \rho$,

where $\rho \stackrel{\text{def}}{=} \omega_c v/c$ defines the Doppler shift. We can put this together with Equ. (1) to get

$$\int_{-\infty}^{\infty} s(t-\delta(t))e^{-i\omega t}\,dt \approx e^{-i(\omega+\rho)\delta_0}\hat{s}(\omega+\rho). \tag{2}$$

The delay Δ^δ and Doppler D^ρ operators can be defined as

$$\begin{aligned}(\Delta^\delta s)(t) &\stackrel{\text{def}}{=} s(t-\delta)\\ (D^\rho s)(t) &\stackrel{\text{def}}{=} e^{-i\rho t}s(t).\end{aligned} \tag{3}$$

Then, in the spectral domain, these operators become

$$\begin{aligned}(\widehat{\Delta^\delta s})(\omega) &= \hat{s}(\omega)e^{-i\delta\omega}\\ (\widehat{D^\rho s})(\omega) &= \hat{s}(\omega+\rho).\end{aligned} \tag{4}$$

Note that the operators have the commutation relations

$$D^\rho\Delta^\delta = e^{-i\rho\delta}\Delta^\delta D^\rho. \tag{5}$$

The failure of these operators to commute with each other is at the heart of the difficulty of beamforming challenged by the combination of large delay and Doppler shifts.

2.2 TAP STATISTICS

2.2.1 Signal Models and Statistics

We employ the spectral representation of stationary random processes (a very general treatment with references to more typical variants is contained in [KM71]) with finite first and second moments. In an informal sense, the representation allows us to write a stationary process $s(t)$ as

$$s(t) = \int_{-\infty}^{\infty} \hat{s}(\omega)e^{i\omega t}\,\frac{dN(\omega)}{2\pi}, \tag{6}$$

where the i.i.d. increments $dN(\omega)$ have the covariance structure

$$\mathrm{E}\left[\frac{dN(\omega_1)}{2\pi}\overline{\frac{dN(\omega_2)}{2\pi}}\right] = \frac{\delta(\omega_1-\omega_2)d\omega}{2\pi}. \tag{7}$$

Note that we have chosen to restrict ourselves to stationary processes without discrete components by using the random measure $dN(\omega)$. In the context of this process model, the delay and Doppler operators take the form

$$(\Delta^\delta s)(t) \stackrel{\text{def}}{=} \int \hat{s}(\omega)e^{-i\delta\omega}e^{i\omega t}\,\frac{dN(\omega)}{2\pi} \tag{8}$$

and

$$(D^\rho s)(t) \stackrel{\text{def}}{=} \int \hat{s}(\omega+\rho)e^{i\omega t}\,\frac{dN(\omega+\rho)}{2\pi}. \tag{9}$$

When the limits in the integral sign are left out as in the above two definitions, we imply integration over $(-\infty, \infty)$. We need to evaluate the behavior of these operators with respect to the inner product

$$\langle s_1, s_2 \rangle \stackrel{\text{def}}{=} \int_{-T/2}^{T/2} \mathrm{E}[s_1(t)\overline{s}_2(t)]\, dt. \tag{10}$$

There are a number of relations that we need. The proofs are highly redundant, so we illustrate the reasoning with a few examples and then state the needed results. We start with

$$\begin{aligned}
\langle D^\rho \Delta^\delta s_1, s_2 \rangle &= \int_{-T/2}^{T/2} \mathrm{E}\left[\int \hat{s}_1(\omega_1 + \rho) e^{-i\delta(\omega_1+\rho)} e^{i\omega_1 t} \frac{dN(\omega_1+\rho)}{2\pi} \overline{\int \hat{s}_2(\omega_2) e^{i\omega_2 t} \frac{dN(\omega_2)}{2\pi}} \right] dt \\
&= \left(\int_{-T/2}^{T/2} e^{-i\rho t}\, dt \right) \left(\int \hat{s}_1(\omega) \overline{\hat{s}}_2(\omega) e^{-i\delta\omega}\, d\omega/2\pi \right) \\
&= T\mathrm{sinc}(\rho T/2) \left(\int \hat{s}_1(\omega) \overline{\hat{s}}_2(\omega) e^{-i\delta\omega}\, d\omega/2\pi \right).
\end{aligned} \tag{11}$$

Next, we evaluate

$$\begin{aligned}
\langle \Delta^\delta D^\rho s_1, s_2 \rangle &= \int_{-T/2}^{T/2} \mathrm{E}\left[\int \hat{s}_1(\omega_1 + \rho) e^{-i\delta\omega_1} e^{i\omega_1 t} \frac{dN(\omega_1+\rho)}{2\pi} \overline{\int \hat{s}_2(\omega_2) e^{i\omega_2 t} \frac{dN(\omega_2)}{2\pi}} \right] dt \\
&= e^{i\rho\delta} \left(\int_{-T/2}^{T/2} e^{-i\rho t}\, dt \right) \left(\int \hat{s}_1(\omega) \overline{\hat{s}}_2(\omega) e^{-i\delta\omega}\, d\omega/2\pi \right) \\
&= e^{i\rho\delta} T\mathrm{sinc}(\rho T/2) \left(\int \hat{s}_1(\omega) \overline{\hat{s}}_2(\omega) e^{-i\delta\omega}\, d\omega/2\pi \right) \\
&= e^{i\rho\delta} \langle D^\rho \Delta^\delta s_1, s_2 \rangle.
\end{aligned} \tag{12}$$

Note that the results are consistent with the commutation relations of Δ^δ and D^ρ, as they should be. There are a few more relations we need that are derived in an almost identical manner. They are

$$\begin{aligned}
\langle D^\rho \Delta^\delta s_1, \Delta^{\delta'} s_2 \rangle &= \langle D^\rho \Delta^{\delta-\delta'} s_1, s_2 \rangle \\
\langle D^\rho s_1, D^{\rho'} s_2 \rangle &= \langle D^{\rho-\rho'} s_1, s_2 \rangle.
\end{aligned} \tag{13}$$

Finally, note that from Equ. (11),

$$\langle D^\rho \Delta^\delta s, s \rangle = T\mathrm{sinc}(\rho T/2) \left(\int |\hat{s}(\omega)|^2 e^{-i\delta\omega}\, d\omega/2\pi \right) = TB\mathrm{sinc}(\rho T/2)\mathrm{sinc}(\pi B\delta), \tag{14}$$

where the last equality results from the assumption that $|\hat{s}(\omega)| = \chi_{[-\pi B, \pi B]}(\omega)$ is the rectangular function over the baseband support of a signal of bandwidth B.

The signal received at the j^{th} element of the array has the form $v_j D^{\rho_j} \Delta^{\delta_j} s(t)$, where $s(t)$ has the stochastic model treated above. The complex scalar v_j represents the effects of delay at the RF center frequency. The element output can be subject to a further delay and doppler shift, resulting

in the output $\mathcal{Z}_{j,\rho,\delta} = v_j D^\rho \Delta^\delta D^{\rho_j} \Delta^{\delta_j} s(t)$. We emphasize that this represents the output of the (ρ, δ) tap from the j^{th} element. Using the commutation relations, we have

$$\langle \mathcal{Z}_{j,\rho,\delta}, s\rangle = v_j e^{i\delta\rho_j} \langle D^{\rho+\rho_j} \Delta^{\delta+\delta_j} s, s\rangle = v_j e^{i\delta\rho_j} TB\text{sinc}((\rho+\rho_j)T/2)\text{sinc}(\pi B(\delta+\delta_j)). \tag{15}$$

The latter expression provides a matched filter output for a given element and delay-Doppler tap combination, averaged over the modeled class of bandwidth-limited and time-limited stationary waveforms. The covariance matrix of the element-tap outputs can be expressed in the form

$$\begin{aligned}
\langle \mathcal{Z}_{j,\rho,\delta}, \mathcal{Z}_{k,\rho',\delta'}\rangle &= v_j \overline{v}_k \langle D^\rho \Delta^\delta D^{\rho_j} \Delta^{\delta_j} s, D^{\rho'} \Delta^{\delta'} D^{\rho_k} \Delta^{\delta_k} s\rangle \\
&= v_j \overline{v}_k e^{i\delta\rho_j} e^{-i\delta'\rho_k} \langle D^{(\rho+\rho_j)-(\rho'+\rho_k)} \Delta^{(\delta+\delta_j)-(\delta'+\delta_k)} s, s\rangle \\
&= v_j \overline{v}_k e^{i\delta\rho_j} e^{-i\delta'\rho_k} \cdot TB \cdot \text{sinc}[T\frac{(\rho+\rho_j)-(\rho'+\rho_k)}{2}] \\
&\cdot \quad \text{sinc}[\pi B((\delta+\delta_j)-(\delta'+\delta_k))],
\end{aligned} \tag{16}$$

using first the commutation relations, then Equ. (13), and finally Equ. (14), for the last equality.

2.2.2 Receiver Subbanding and Large Doppler Shifts

The relations given above assume that the receiver is open to a bandwidth that includes the signal and all of its Doppler shifts. This is a reasonable assumption when the Doppler shifts are small, but may not be physically plausible with large Doppler shifts or in the case that only a subbanded portion of the signal is treated, by applying a bandpass filter to the environment before employing delay-Doppler taps in the signal processing. Thus, we need an elaboration on the calculation given above that incorporates bandpass filtering at each sensor.

First, note that the delay-Doppler shift of a signal $s(t)$ can be written in the form

$$(D^\rho \Delta^\delta s)(t) = e^{-i\rho\delta} \int \hat{s}(\omega+\rho) e^{-i\omega\delta} e^{i\omega t} \frac{dN(\omega+\rho)}{2\pi}. \tag{17}$$

Thus, we have

$$(D^\rho \Delta^\delta D^{\rho_j} \Delta^{\delta_j} s)(t) = e^{-i\rho\delta} e^{-i\rho_j\delta_j} \int \hat{s}(\omega+\rho+\rho_j) e^{-i\omega\delta} e^{-i(\omega+\rho)\delta_j} e^{i\omega t} \frac{dN(\omega+\rho+\rho_j)}{2\pi}. \tag{18}$$

This allows us to rewrite the covariance calculation previously given in Equ. (85) in terms of

$$\begin{aligned}
\langle D^\rho \Delta^\delta D^{\rho_j} \Delta^{\delta_j} s, D^{\rho'} \Delta^{\delta'} D^{\rho_k} \Delta^{\delta_k} s\rangle &= \int_{-T/2}^{T/2} \text{E}\left[(D^\rho \Delta^\delta D^{\rho_j} \Delta^{\delta_j} s)(t) \overline{(D^\rho \Delta^\delta D^{\rho_j} \Delta^{\delta_j} s)}(t)\right] dt \\
&= Te^{i\rho_j\delta} e^{-i\rho_k\delta'} \text{sinc}[((\rho+\rho_j)-(\rho'+\rho_k))T/2] \cdot \int |\hat{s}(\omega)|^2 e^{-i\omega[(\delta+\delta_j)-(\delta'+\delta_k)]}\, d\omega/2\pi,
\end{aligned} \tag{19}$$

after a straightforward calculation that is a more elaborate version of the the calculation used in Equ. (79), for example.

To incorporate bandpass filtering in this result, note first that

$$(h \star (D^\rho \Delta^\delta s))(t) = e^{-i\rho\delta} \int \hat{h}(\omega) \hat{s}(\omega+\rho) e^{-i\omega\delta} e^{i\omega t} \frac{dN(\omega+\rho)}{2\pi}, \tag{20}$$

where $h(t)$ denotes the impulse response of the bandpass filter and $\star$ denotes convolution. The result we need is expressed by

$$\begin{aligned}
&\langle (D^{\rho}\Delta^{\delta})(h \star (D^{\rho_j}\Delta^{\delta_j} s)), (D^{\rho'}\Delta^{\delta'})(h \star (D^{\rho_k}\Delta^{\delta_k} s))\rangle \\
&= Te^{i\rho_j\delta}e^{-i\rho_k\delta'}\mathrm{sinc}[((\rho+\rho_j)-(\rho'+\rho_k))T/2] \\
&\cdot \int \hat{h}(\omega-\rho_j)\overline{\hat{h}}(\omega-\rho_k)|\hat{s}(\omega)|^2 e^{-i\omega[(\delta+\delta_j)-(\delta'+\delta_k)]}\, d\omega/2\pi.
\end{aligned} \tag{21}$$

Again, the calculation is straightforward. To the extent the transfer function $\hat{h}(\omega)$ is bandlimited, one can see that large differential Doppler shifts (i.e., large $|\rho_j - \rho_k|$) can reduce the size of the integral factor in Equ. (21) significantly, resulting in signal decorrelation between sensors. This is a new behavior (compare Equ. (19)), introduced by the bandpass filtering, that becomes important with large Doppler shifts.

We want to evaluate the integral in Equ. (21) for the specific $\hat{h}(\omega) \stackrel{\mathrm{def}}{=} (2\pi)^{-1/4}e^{-\frac{\omega^2}{16\pi^2B^2}}$, which has RMS bandwidth B. We choose $\hat{s}(\omega) \equiv 1$ to represent a wideband signal, a portion of which we are processing. Note that $\int |\hat{h}(\omega)|^2|\hat{s}(\omega)|^2\, d\omega/2\pi = B$. Thus, with the noise assumed to have unity spectral density, the covariance scales with the signal-to-noise ratio. We have, by a short calculation,

$$\begin{aligned}
&\int \hat{h}(\omega-\rho_j)\overline{\hat{h}}(\omega-\rho_k)|\hat{s}(\omega)|^2 e^{-i\omega[(\delta+\delta_j)-(\delta'+\delta_k)]}\, d\omega/2\pi \\
&= B \cdot e^{-2\pi^2B^2[(\delta+\delta_j)-(\delta'+\delta_k)]^2} \cdot e^{-\frac{(\rho_j-\rho_k)^2}{32\pi^2B^2}} \cdot e^{-i\left(\frac{\rho_j+\rho_k}{2}\right)[(\delta+\delta_j)-(\delta'+\delta_k)]}.
\end{aligned} \tag{22}$$

Putting things together, Equ. (21) becomes

$$\begin{aligned}
&\langle (D^{\rho}\Delta^{\delta})(h \star (D^{\rho_j}\Delta^{\delta_j} s)), (D^{\rho'}\Delta^{\delta'})(h \star (D^{\rho_k}\Delta^{\delta_k} s))\rangle \\
&= TBe^{i\rho_j\delta}e^{-i\rho_k\delta'}\mathrm{sinc}[((\rho+\rho_j)-(\rho'+\rho_k))T/2] \cdot e^{-2\pi^2B^2[(\delta+\delta_j)-(\delta'+\delta_k)]^2} \\
&\cdot e^{-\frac{(\rho_j-\rho_k)^2}{32\pi^2B^2}} \cdot e^{-i\left(\frac{\rho_j+\rho_k}{2}\right)[(\delta+\delta_j)-(\delta'+\delta_k)]}.
\end{aligned} \tag{23}$$

As a point of comparison, a signal with flat ensemble spectral distribution over the band $[-B, B]$ has RMS bandwidth $B/\sqrt{12}$. The Gaussian filter shape is chosen for ease in computations. Filters with flatter passbands and narrower transition bands are likely to be more interesting in applications.

2.2.3 Derivative-Based Updates (DBU)

The DBU version of beamforming algorithms take the vector observations $\mathbf{z}(t)$, whether directly from the array outputs or out of taps that introduce delay and Doppler shifts, and form the extended data

$$\mathbf{y}(t) \stackrel{\mathrm{def}}{=} \begin{pmatrix} \mathbf{z}(t) \\ t\mathbf{z}(t) \end{pmatrix}. \tag{24}$$

All performance calculations are based on

$$\int_{-T/2}^{T/2} \mathrm{E}[\mathbf{y}(t)\mathbf{y}^{\dagger}(t)]\, dt = \int_{-T/2}^{T/2} \begin{pmatrix} \mathrm{E}[\mathbf{z}(t)\mathbf{z}^{\dagger}(t)] & t\mathrm{E}[\mathbf{z}(t)\mathbf{z}^{\dagger}(t)] \\ t\mathrm{E}[\mathbf{z}(t)\mathbf{z}^{\dagger}(t)] & t^2\mathrm{E}[\mathbf{z}(t)\mathbf{z}^{\dagger}(t)] \end{pmatrix} dt \tag{25}$$

and

$$\int_{-T/2}^{T/2} \mathrm{E}[\mathbf{y}(t)\overline{s}(t)]\,dt = \int_{-T/2}^{T/2} \begin{pmatrix} \mathrm{E}[\mathbf{z}(t)\overline{s}(t)] \\ t\mathrm{E}[\mathbf{z}(t)\overline{s}(t)] \end{pmatrix} dt. \tag{26}$$

These statistical expressions for the DBU versions of beamforming algorithms can be evaluated by taking derivatives of the expressions that have already been derived. For example, define

$$\langle \tilde{s}_1, s_2 \rangle_{f(t)} = \int_{-T/2}^{T/2} f(t)\mathrm{E}[s_1(t)\overline{s}_2(t)]\,dt. \tag{27}$$

Thus, from Equ. (81),

$$\begin{aligned} \langle D^\rho \Delta^\delta s_1, s_2 \rangle_t &= \left(\int_{-T/2}^{T/2} te^{-i\rho t}\,dt \right) \left(\int \hat{s}_1(\omega)\overline{\hat{s}}_2(\omega)e^{-i\delta\omega}\,d\omega/2\pi \right) \\ &= i\frac{T^2}{2}\mathrm{sinc}'(\rho T/2) \left(\int \hat{s}_1(\omega)\overline{\hat{s}}_2(\omega)e^{-i\delta\omega}\,d\omega/2\pi \right), \end{aligned} \tag{28}$$

where $\mathrm{sinc}'(x) \stackrel{\mathrm{def}}{=} \frac{\partial}{\partial x}\mathrm{sinc}(x)$. In particular,

$$\langle D^\rho \Delta^\delta s, s \rangle_t = TB \cdot i\frac{T}{2} \cdot \mathrm{sinc}'(\rho T/2)\mathrm{sinc}(\pi B\delta). \tag{29}$$

In a similar manner,

$$\langle s_1, s_2 \rangle_{t^2} = \int_{-T/2}^{T/2} t^2 \mathrm{E}[s_1(t)\overline{s}_2(t)]\,dt, \tag{30}$$

so that (see Equ. (85))

$$\begin{aligned} &\langle D^\rho \Delta^\delta s, D^{\rho'} \Delta^{\delta'} s \rangle_{t^2} \\ &\qquad = \langle D^{\rho-\rho'} \Delta^{\delta-\delta'} s, s \rangle_{t^2} &(31)\\ &\qquad = -TB \cdot \frac{T^2}{4} \cdot \mathrm{sinc}''[T\frac{\rho-\rho'}{2}] \cdot \mathrm{sinc}[\pi B(\delta-\delta')], &(32) \end{aligned}$$

where $\mathrm{sinc}''(x) \stackrel{\mathrm{def}}{=} \frac{\partial^2}{\partial x^2}\mathrm{sinc}(x)$. The equations Equ. (29) and Equ. (32) in addition to the previous equations Equ. (85) and Equ. (83), allow us to compute the statistics required to characterize the performance of the best STFAP or DBU STFAP beamformer over the ensemble of signals and noise, as we will see next.

2.2.4 Architecture

Operator expressions such as $D^{j\rho}\Delta^{k\delta}$ can be implemented by forming tapped delay lines whose outputs are first delayed in time by $k\delta$ and then shifted in frequency by $j\rho$. It is typical to assume that the basic delay δ is related to the spacing of samples at the Nyquist or, potentially, higher rates. The basic frequency spacing ρ is related to the time extent over which the beamformer is trained and applied. If T_s denotes the duration of the interval within which the weights are applied, then $\rho = T_s^{-1}$, or smaller. When taps are spaced systematically in delay and Doppler, the basic

receiver architecture is represented by Fig. 3. The special case without frequency shifts, called space-time adaptive processing (STAP), is treated in many places (see [War98], for example). In the more general case, we use the terminology space-time-frequency adaptive processing (STFAP).

The architecture of Fig. 3 models the application of delay-Doppler operators to the data. Each delay has its output shifted in frequency, resulting in a delay-Doppler tap. Just as the delays shifts consist of all multiples of a basic delay δ up to a maximum, the Doppler shifts consist of all multiples of a basic Doppler shift ρ up to a maximum. Each delay shift and Doppler shift pair is implemented. We refer to this as a dense (delay-Doppler) tap pattern. It is possible to use a subset of all possible pairs, forming sparse delay-Doppler tap patterns that considerably reduce the number of unknowns used in beamforming. This reduction in unknowns has both performance and, potentially, complexity benefits, especially in the cases of large differential delays and Dopplers.

It is desirable to minimize the number of delay-Doppler taps used for array processing. One way to accomplish this, for scenarios with insignificant Doppler spreads, is to apply a filter bank to the received signal and perform array processing (interference suppression) in the subbands created. If the subband bandwidth is sufficiently small, no delay taps will be required to obtain large null depths. The complexity is subsumed by the filter-bank architecture, which can be efficiently implemented using fast transforms on parallel hardware. This approach is problematic when the Doppler spreads are significant. Small subbands are more sensitive to Doppler shifts as the shift becomes a significant fraction of the subband bandwidth. Thus, when delay spreads are large, suggesting small subbands and Doppler spreads are large, decorrelating signals over the subbands, the choice of subband size becomes an important consideration in regard to null depth.

2.2.5 Performance Examples

One way of designing the signal processing for a distributed array subject to delay and Doppler spreads involves examining null depth. The covariances just described can be used to quantify the residual signal energy left after attempted nulling. For the particular case of a two-sensor problem, the null depth is conveniently quantified by the condition number of the covariance, which is a 2×2 matrix in that situation. Null depth is shown in Fig. 6 for just such a model. The signal is assumed to possess a bandwidth of 1 kHz and is observed over an interval of about 3.33 ms. It is assumed that beamforming occurs over this update interval, and hence, occurs at a rate that is the reciprocal of the update interval. Null depth is shown in Fig. 6 as a function of differential delay and differential Doppler for a fixed signal bandwidth and update rate. To be explicit, covariance calculations are based on Equ. (23), which is based on a Gaussian filter.

Since the signal is divided into subbands before array processing is applied, it is natural to ask for an optimally sized subband and update rate. The answer depends on the delay and Doppler spreads seen by the two sensors, so there is no simple solution. However, null depth is worst, roughly speaking, at the largest delay and Doppler spreads. Thus, it makes sense to choose subband bandwidth and update rates based on that largest delay-Doppler spreads.

One major issue must be addressed for this evaluation to make sense. The number of samples observed in the update interval within the subband bandwidth must support estimation of the unknown parameters required for signal suppression. For the case of a two-sensor problem, this

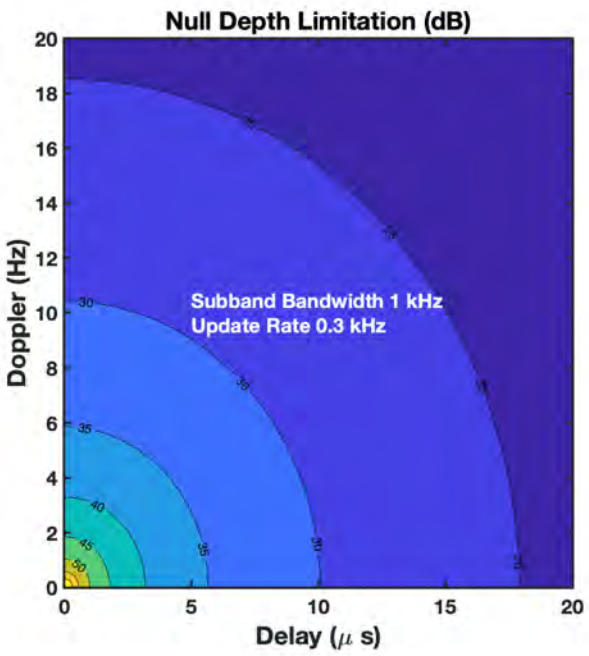


Figure 6. For a fixed signal bandwidth of 1 kHz and a fixed update rate of 300 Hz, we show the null depth of a two-sensor array as a function of the differential delay and Doppler between the sensors. Null depth decreases as either delay or Doppler spread increases and is limited to a worst-case value of about 20 dB at the extreme delay-Doppler spreads shown of 20 Hz and 20 μs.

means we need at at least two samples. We will be slightly more conservative and ask for twice as many samples as the number of unknown parameters.

In Fig. 7, we show null depth for a fixed delay spread of 20 μs and Dopper spread of 20 Hz as a function of update rate and subband bandwidth. When the sample support is less than or equal to 4, we set the null depth to 0 dB. One can see that the largest null depth occurs for bandwidth and update rate values similar to those used in Fig. 6. The largest null depths are around 20 to 25 dB.

In Fig. 8 we show the null depth of a STFAP architecture involving dense delay-Doppler taps. A 10 by 5 grid of delay-Doppler taps is used for each sensor. The delays are spaced at a four times the reciprocal RMS subband bandwidth and the Dopplers at four times the reciprocal update rate. Delay spread and Doppler spread is fixed for the two sensors at values comparable to those in the previous figures. Null depth is shown as a function of subband bandwidth and update rate. We see that performance improves until about 75 kHz subbands. Performance is best at the highest update rates allowed based on the minimal sample support required: twice the number of unknowns, which, in this case, means more than 200 samples over the update interval in the subband.

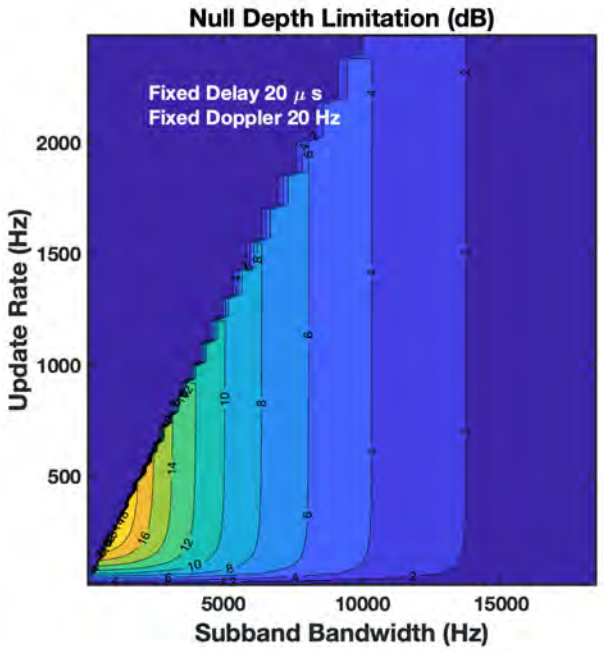


Figure 7. Since the signal is cut into subbands and weight-update intervals before array processing, it is important to consider the bandwidths and temporal extents of the processing intervals. We show null depth as a function subband bandwidth and update rate for a fixed delay spread of 20 μs and Doppler spread of 20 Hz, corresponding to the extremes of delay and Doppler spreads shown in Fig. 6. Note that the largest null depth occurs with update rate and subband bandwidth parameters close to those of Fig. 6.

Using the selected subband bandwidths of 75 kHz and update rates of 300 Hz, we can evaluate null depth as a function of delay and Doppler spreads. The results are shown in Fig. 9. Instead of the 20 to 25 dB null depths that could be achieved without the use of STFAP, null depths are increased to around 30 to 35 dB and often more with STFAP as used in the example. In both cases, the signal processing cuts the desired signal into subbands, within which the array processing is implemented. The size of the subbands yielding optimal performance differ markedly between the two approaches. In both cases, depending on the modulation and coding of the signal, it may be necessary to reconstruct the signal from its subband components.

A choice of subband bandwidth should be undertaken along with an optimization of the number of dense taps required for good performance. Since there are a number of performance measures that will be considered here, including channel capacity and effective SNR of a beamformer, as well as null depth, a choice of an array processing subband size will be entangled with the details of particular applications, including full-band signal reconstruction and channel coding. Only some tools for performing these kinds of tradeoffs are presented here.

The example chosen illustrates the tension between choosing a small bandwidth to handle delay spread and a large bandwidth to handle Doppler spread. For the chosen example, the ar-

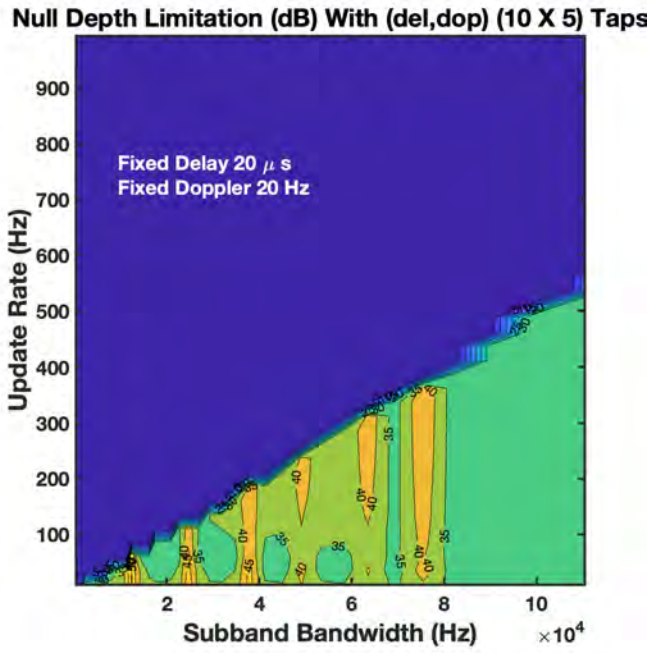


Figure 8. For a tapped-delay-line architecture with 10 delay values, spaced at one-half bandwidth, by 5 Doppler values, spaced at one-half a frequency resolution cell, null depth is shown for a two-sensor array at fixed differential delay and Doppler shifts, as a function of update rate and subband bandwidth. Note that there are a total of 50 taps per sensor since each delay and Doppler tap combination is utilized. Large sections of the plots correspond to regimes where the number of samples in the subband over the update interval is smaller than twice the number of unknown parameters that must be estimated from the data. These sections of the plots are shown with 0 dB null depth. From this plot one can deduce that a subband bandwidth of around 75 kHz and an update rate of around 300 Hz provide the best performance for the chosen fixed delays and Dopplers.

ray processing subband bandwidth is chosen to place the delay spread within a few reciprocal bandwidths. This delay spread can be easily handled by the ten delay-tap values available. The subband bandwidth is wide enough to accommodate training samples in a short enough interval to avoid Doppler sensitivity. This situation would remain roughly the same if the Doppler were increased by an order of magnitude, since the subband bandwidth still allows fast weight updates. A problem arises when both delay and Doppler spreads are large enough that the number of dense delay-Doppler taps required for good null depth is too large to be trained. It is then necessary to reduce the number of taps. In other words, delay-Doppler taps must be distributed sparsely or the training must only address a subset of the unknowns.

In Fig. 4 we see the ratio of training samples to unknowns for the subband nulling problem using a two-sensor array. The number of unknowns can be expressed by the product of the number of delay taps $n_\delta = 2\,\mathrm{Del}_{\max} B$ and the number of Doppler taps $n_D = 2\,\mathrm{Dop}_{\max} T$. We have assumed oversampling by a factor of 2 in each case to allow for interpolation for delays and Dopplers off

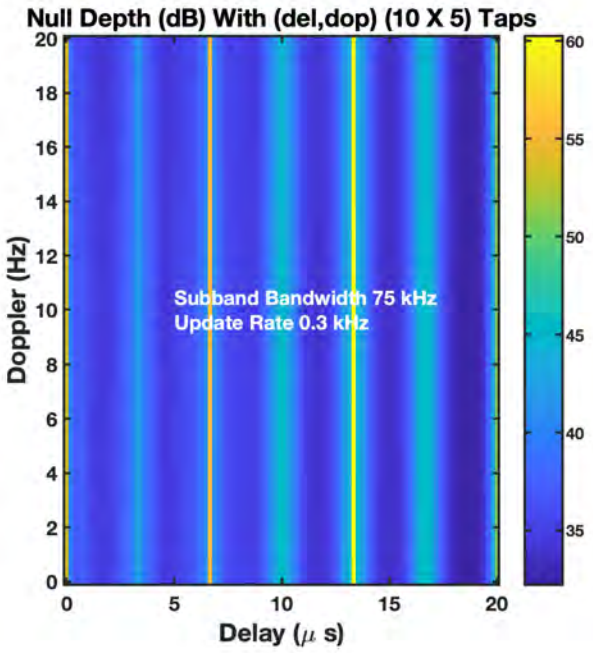


Figure 9. Having chosen the design parameters of 75 kHz for the subband and 300 Hz for the update rate, we can evaluate null depth for a range of delay and Doppler spreads. Instead of the median 25 dB null depth realized without using any delay-Doppler taps, we now achieve a median of 35 dB of null depth and often more. The plot shown has null depth limited to at most 60 dB for numerical reasons. The periodicities evident in this plot and in Fig. 8 result from delay spreads matching the lattice spacing of the tapped delay line.

lattice. Thus, the ratio of training samples to unknowns becomes $\frac{TB}{n_\delta n_D} = \frac{1}{4\mathrm{Del}_{\max}\mathrm{Dop}_{\max}}$. For a combination of large delays and Doppler spreads, the number of training samples is inadequate for estimating all unknowns using densely spaced delay-Doppler taps. For many practical applications, the sample-starved region is not approached. However, distributed arrays that are in relative motion with large velocities can present scenarios that lie in the sample-starved region. Approaches that take advantage of channel models can significantly reduce the number of delay-Doppler taps that must be trained. Several of these approaches will be treated in later sections.

2.3 BEAMFORMING, MUTUAL INFORMATION, AND EFFECTIVE SNRS

We now have enough machinery to form adaptive beams and evaluate the performance of the beamformer. The data consists of element-tap outputs that comprise delayed and Doppler shifted versions of the outputs of each antenna element. These signals have been denoted $\mathcal{Z}_{k,\rho,\delta}$, representing the output of the k^{th} antenna, delayed by δ and shifted in frequency by ρ. If this data

consists of a single emitter, correlation statistics of the form

$$\langle \mathcal{Z}_{k,\rho,\delta}, s\rangle \text{ and } \langle \mathcal{Z}_{k,\rho,\delta}, \mathcal{Z}_{k,\rho',\delta'}\rangle \tag{33}$$

have been calculated above. Here, $s \equiv s(t)$ represents one of the signals in the environment. It is assumed that different signals decorrelate in the ensemble averages. If we treat $\mathcal{Z}$ like a matrix with rows $\mathcal{Z}_{k,\rho,\delta}$, then beamforming can be expressed as

$$\mathbf{w}^\dagger \mathcal{Z} = \sum_{k,\rho,\delta} \overline{w}_{k,\rho,\delta} \mathcal{Z}_{k,\rho,\delta}. \tag{34}$$

We will use $\langle \mathcal{Z}, \mathcal{Z}\rangle$ and $\langle \mathcal{Z}, s\rangle$ to represent the matrix and vector, respectively, associated with the entries expressed in Equ. (33).

As a slight digression, consider the observations Z and signal s as the partition components X and Y in Equ. (77). We can use Equ. (78), to obtain the mutual information per sample between Z and s as expressed by

$$\log_2 \frac{|R_{ZZ}|}{|R_{ZZ} - R_{Zs}R_{ss}^{-1}R_{sZ}|} = -\log_2(1 - v^\dagger R_{ZZ}^{-1} v) = \log_2(1 + v^\dagger (R_{ZZ} - vv^\dagger)^{-1} v), \tag{35}$$

where $v \stackrel{\text{def}}{=} R_{Zs}/\sqrt{R_{ss}}$. This calculation suggests an expression like $v^\dagger(R_{ZZ} - vv^\dagger)^{-1}v$ can serve as an effective SNR for our delay-Doppler channel.

The beamforming problem we wish to solve minimizes the least-squares difference between the beamformer output and the desired signal. More precisely,

$$\begin{aligned} \min_{\mathbf{w}} \langle \mathbf{w}^\dagger \mathcal{Z} - \mathbf{s}, \mathbf{w}^\dagger \mathcal{Z} - \mathbf{s}\rangle &= \min_{w} (\mathbf{w} - \langle \mathcal{Z}, \mathcal{Z}\rangle^{-1}\langle \mathcal{Z}, \mathbf{s}\rangle)^\dagger \langle \mathcal{Z}, \mathcal{Z}\rangle (\mathbf{w} - \langle \mathcal{Z}, \mathcal{Z}\rangle^{-1}\langle \mathcal{Z}, \mathbf{s}\rangle) \\ &+ \langle \mathbf{s}, \mathbf{s}\,\rangle - \langle \mathbf{s}, \mathcal{Z}\rangle\langle \mathcal{Z}, \mathcal{Z}\rangle^{-1}\langle \mathcal{Z}, \mathbf{s}\rangle \\ &= \langle \mathbf{s}, \mathbf{s}\,\rangle - \langle \mathbf{s}, \mathcal{Z}\rangle\langle \mathcal{Z}, \mathcal{Z}\rangle^{-1}\langle \mathcal{Z}, \mathbf{s}\rangle. \end{aligned} \tag{36}$$

This gives us the beamformer weights $\mathbf{w}$ and the residual. Using the form of effective SNR just presented, we can write the residual as

$$\langle s, s\rangle \left(1 - \frac{\langle \mathbf{s}, \mathcal{Z}\rangle\langle \mathcal{Z}, \mathcal{Z}\rangle^{-1}\langle \mathcal{Z}, \mathbf{s}\rangle}{\langle \mathbf{s}, \mathbf{s}\,\rangle}\right) = \langle s, s\rangle \left(1 + v^\dagger(\langle \mathcal{Z}, \mathcal{Z}\rangle - vv^\dagger)^{-1} v\right)^{-1}, \tag{37}$$

with

$$\text{SNR}_{\text{eff}} \stackrel{\text{def}}{=} v^\dagger(\langle \mathcal{Z}, \mathcal{Z}\rangle - vv^\dagger)^{-1} v \tag{38}$$

representing the effective SNR. Here, $v \stackrel{\text{def}}{=} \langle \mathcal{Z}, s\rangle/\sqrt{\langle s, s\rangle}$. Maximizing SNR_{eff} is equivalent to minimizing the least-squares residual.

The data $\mathcal{Z}$ is summed over all signals in the environment. It is assumed that all taps contribute thermal noise, so that $\langle \mathcal{Z}, \mathcal{Z}\rangle$ has a white noise covariance (i.e., multiple of the identity) added to it. The signals are assumed to decorrelate in the ensemble average, so that the results of Section 2.1 apply to both $\langle \mathcal{Z}, \mathcal{Z}\rangle$ and to $\langle \mathcal{Z}, s\rangle$.

Although the notation suggests STFAP, we can consider processing DBU-extended (see Section 2.2.3) data using the results of Section 2.2 in Equ. (37).

Beamforming performance is shown in Fig. 10 and Fig. 11 for STAP, STFAP, and their DBU variants. The examples treat a five element array with three co-channel signals, two of which are strong interferers while the third is the desired signal received at 30 dB below each interferer. The performance relevant to the desired signal is shown in terms of the duration of the weights. Longer weight durations are more subject to dynamics, resulting in degraded performance. Performance is measured by the effective SINR given by Equ. (37). This is an ensemble level of performance that does not account for sampling effects. Hence, the good performance at low weight intervals is less achievable due to the small sample support during those short time intervals. As a point of reference, the example waveform has 25 kHz bandwidth and thus provides 250 samples at .01 s, which should be an adequate number of samples for determining the beamformer weights. For these figures, the desired signal is assumed to be aligned in delay and Doppler. The delays δ_j and Dopplers ρ_j take values centered (roughly) about zero to aid matching the reference signal.

2.4 CHANNEL CAPACITIES

Let us assume that the signal $s(t)$ is wide-sense stationary, complex Gaussian, with zero-mean and autocovariance $\mathrm{E}[s(t_1)\overline{s}(t_2)] = R(t_1 - t_2)$. Let $\alpha = (\rho, \delta)$ represent the pair of Doppler and delay shifts. Then we have the cross-covariance

$$\begin{aligned} \mathcal{R}_{\alpha\alpha'}(t_1, t_2) = \mathrm{E}[(D^\rho \Delta^\delta s)(t_1)\overline{(D^{\rho'} \Delta^{\delta'} s)}(t_2)] &= \mathrm{E}[e^{-i\rho t_1} s(t_1 - \delta) e^{i\rho' t_2} \overline{s}(t_2 - \delta')] \\ &= e^{-i\rho t_1} e^{i\rho' t_2} R(t_1 - t_2 + \delta' - \delta). \end{aligned} \tag{39}$$

Assume, in addition, that the frequencies ρ lie on a lattice, in the sense that they have a common period T_p such that $\rho T_p, \rho' T_p \in 2\pi\mathbb{Z}$. With this assumption, the cross-covariance becomes cyclostationary in the sense that $\mathcal{R}_{\alpha\alpha'}(t_1 + T_p, t_2 + T_p) = \mathcal{R}_{\alpha\alpha'}(t_1, t_2)$. Choose sampling times $0 \le t_1 < t_2 < \ldots < t_n \le T_p$ and form the vector $\mathbf{s}_\alpha(k) \stackrel{\mathrm{def}}{=} ((D^\rho \Delta^\delta s)(t_1 + kT_P), \ldots, (D^\rho \Delta^\delta s)(t_n + kT_p))^T$, which has cross-covariance matrix components

$$\mathrm{E}[\mathbf{s}_\alpha(j)\mathbf{s}^\dagger_{\alpha'}(k)]. \tag{40}$$

that depend only on the the difference $j - k$.

We are interested in channel capacity when each sensor receives a signal $s(t)$ delayed by δ and Doppler shifted by ρ. Assume $T_p = M/B$ for integral M and choose sample times $t_k = k/B$, for a signal of bandwidth B. Refining the previous notation slightly, let $\alpha = (\rho_\alpha, \delta_\alpha)$ denote the delay-Doppler parameters associated with the sensor α for a fixed signal. Then we can define single-signal covariance at lag $\tau = j - k$, $\mathbf{C}(\tau)$, as

$$C_{\alpha l \beta m}(j - k) \stackrel{\mathrm{def}}{=} \Big(\mathrm{E}[\mathbf{s}_\alpha(j)\mathbf{s}^\dagger_\beta(k)]\Big)_{lm}. \tag{41}$$

In this manner, we have converted a cyclostationary channel [HC12] into a wide-sense stationary MIMO channel.

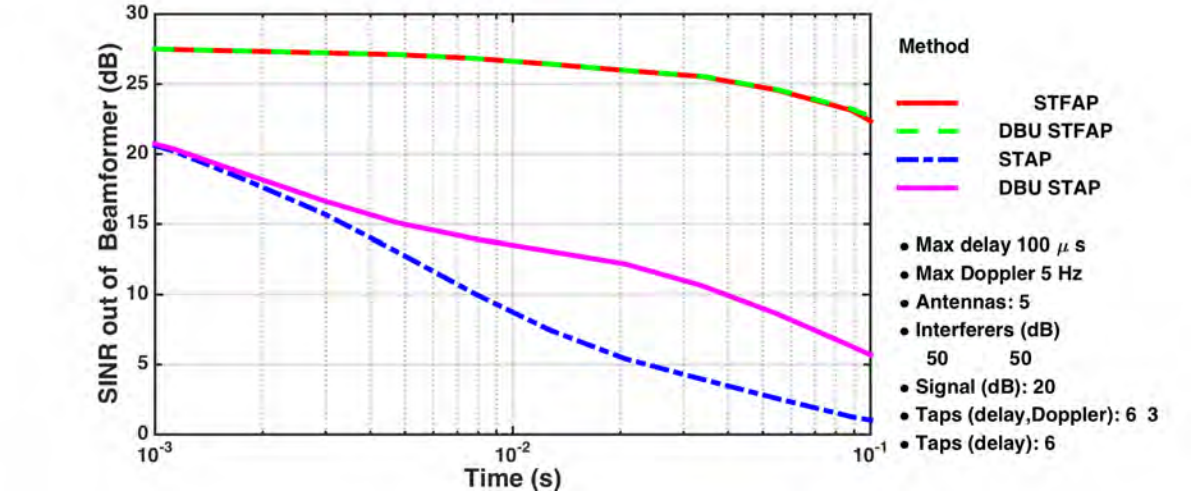


Figure 10. Shown is the median performance of a five element array employing four beamforming algorithms over an ensemble of environments that are characterized by the maximum delay and Doppler spreads allowed. The algorithms consist of STAP, using 6 taps per antenna channel, DBU STAP, STFAP, using 6×3 delay-Doppler taps per antenna channel, and DBU STFAP. The environment contains maximum delay spreads of 100 μs and Doppler spreads of 5 Hz. The scenario includes three signals, two of which are considered interferers. The interferers have element SNRs of 50 dB while the desired signal has an element SNR of 20 dB. It is assumed that the desired signal is time and Doppler aligned at all antenna elements. This is typically not true initially, but can be arranged upon synchronization. The receiver's delay taps are spaced at Nyquist sampling rates for a 25 kHz signal. Doppler taps are spaced at frequency intervals that are the reciprocal of the values on the x-axis, which specify the time duration of the weights. For signals with 25 kHz bandwidth, .01 s is equivalent to 250 samples of the signal. The effective SINR shown on the y-axis is expressed by Equ. (37), based on the residual error associated with a least-squares fit of the beamformer output to the observed data shown in Equ. (36). STFAP offers the best performance with little improvement using DBU STFAP. However, the DBU variant of STAP does offer better performance than STAP by itself. For all the curves shown here, the effective SINR over 250 samples (time duration .01) is large enough to provide good performance when demodulating a low spectral efficiency signal such as QPSK.

Putting things together, we utilize a Grenanader-Szego-Widom theorem [BDES19], [JT00] in the form

Theorem 1. *With appropriate integrability conditions,*

$$\lim_{N\to\infty} N^{-1} \log_2 |\mathbf{C}(j-k)|_{0\le j,k<N} = \int_{-\pi}^{\pi} \log_2 |\hat{\mathbf{C}}(\theta)|\, d\theta/2\pi, \tag{42}$$

where $\hat{\mathbf{C}}(\theta) \stackrel{\text{def}}{=} \sum_l \mathbf{C}(l) e^{-il\theta}$.

Proof. The cited references have more general and precise statements including assumptions. The narrow application to covariance matrices that we need meets the conditions required, provided, for example, that the Fourier series expansion for $\hat{\mathbf{C}}(\theta)$ is finite and $\hat{\mathbf{C}}(\theta) \neq 0$. A continuity argument can be used to remove the nonvanishing requirement. □

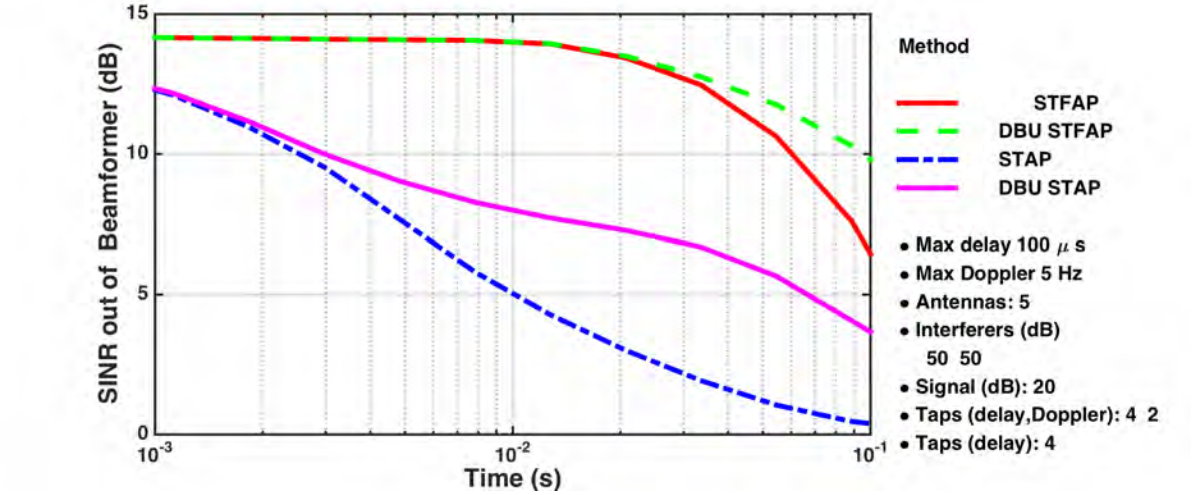


Figure 11. Reducing the delay and Doppler taps degrades performance significantly. Note that there is still substantial benefit to using more than one Doppler tap (i.e., employing some Doppler compensation). Even with these tap counts, there can still be adequate SINR for demodulation.

Corollary 1. *Let* $\mathbf{C}_I(\theta) = \mathbf{I}_n + \sum_k \mathbf{v}_k(\theta)\mathbf{v}_k^\dagger(\theta)$ *and* $\mathbf{C}_S(\theta) = \mathbf{v}(\theta)\mathbf{v}^\dagger(\theta)$. *Then we have the channel capacity*

$$\begin{aligned}\lim_{N\to\infty} N^{-1}\log_2 \frac{|\mathbf{C}_S(j-k)+\mathbf{C}_I(j-k)|}{|\mathbf{C}_I(j-k)|}\Big|_{0\le j,k<N} &= \int_{-\pi}^{\pi} \log_2 |I_n + \hat{\mathbf{C}}_I(\theta)^{-1}\hat{\mathbf{C}}_S(\theta)|\, d\theta/2\pi \quad (43)\\ &= \int_{-\pi}^{\pi} \log_2(1+\mathbf{v}^\dagger(\theta)\hat{\mathbf{C}}_I(\theta)^{-1}\mathbf{v}(\theta))\, d\theta/2\pi.\end{aligned}$$

Proof. This follows from the definition of channel capacity and the previous theorem. It's worth noting that this result expresses channel capacity in terms of the average of the capacities associated with each of the spectral, adapted SNRs $\mathbf{v}^\dagger(\theta)\hat{\mathbf{C}}_I(\theta)^{-1}\mathbf{v}(\theta)$.

With large levels of interference, $\mathbf{C}_I(\theta)$ can approximate a projection matrix onto the orthocomplement of the span of the $\mathbf{v}_k(\theta)$. If the spectral array SNRs $\|\mathbf{v}(\theta)\|^2$ are small, then the capacity is approximately $\log_2(1+\int_{-\pi}^{\pi}\mathbf{v}^\dagger(\theta)\mathbf{C}_I(\theta)^{-1}\mathbf{v}(\theta)\, d\theta/2\pi)$. If, in addition, $\mathbf{C}_I(\theta)$ is approximately a projector and behaves independently and randomly from $\mathbf{v}(\theta)$ as a function of θ, then

$$\begin{aligned}\int_{-\pi}^{\pi}\mathbf{v}^\dagger(\theta)\mathbf{C}_I(\theta)^{-1}\mathbf{v}(\theta)\, d\theta/2\pi &\approx \operatorname{tr}\left(\left(\int_{-\pi}^{\pi}\mathbf{C}_I(\theta)^{-1}\, d\theta/2\pi\right)\left(\int_{-\pi}^{\pi}\mathbf{v}(\theta)\mathbf{v}^\dagger(\theta)\, d\theta/2\pi\right)\right)\\ &\approx \frac{n-k_I}{n}\cdot \text{average ASNR}, \quad (44)\end{aligned}$$

where k_I denotes the number of interferers (i.e., summands in $\mathbf{C}_I(\theta)$). □

As another application,

Theorem 2. *Let*

$$C_{\alpha l\beta m}(j-k) = R\left(\frac{l-m}{B} - (\delta_\alpha - \delta_\beta) + (j-k)\frac{M}{B}\right) e^{-i\rho_\alpha l/B} e^{i\rho_\beta m/B} \tag{45}$$

denote the covariance matrix of a single signal with unity complex power observed at multiple sensors, indexed by α, at delay-Doppler shifts $(\rho_\alpha, \delta_\alpha)$, and at offsets l/B and m/B from the framing of length $T_p = M/B$. The Doppler shifts ρ are assumed to satisfy $\rho T_p \in 2\pi\mathbb{Z}$. The associated MIMO channel is wide-sense stationary with the expressed covariance. Letting $\mathbf{C}_S$ denote the MIMO covariance of a desired signal and $\mathbf{C}_I$ the covariance of interfering signals and noise, the spectral efficiency of the original channel can be expressed as

$$M^{-1} \int_{-\pi}^{\pi} \log_2 |\hat{\mathbf{C}}_S(\theta)\hat{\mathbf{C}}_I^{-1}(\theta) + \mathbf{I}_{Mn}| \, d\theta/2/\pi, \tag{46}$$

where $\hat{\mathbf{C}}(\theta) \stackrel{\text{def}}{=} \sum_l \mathbf{C}(l)e^{-il\theta}$, and n is the number of sensors.

Proof. This follows from standard capacity calculations for Gaussian MIMO channels along with the Szego-Widom theorem. The MIMO capacity would not divide by M. Division is required for the spectral efficiency of the original channel. □

For calculations, we use a specific autocorrelation function $R(\tau)$ that has a Gaussian shape. This leads to the result:

Theorem 3. *Given the autocorrelation $R(\tau) \stackrel{\text{def}}{=} e^{-2\pi^2\tau^2B^2}$, which has RMS bandwith B, the spectral form of the MIMO covariance for the cyclostationary delay-Doppler channel becomes*

$$\hat{C}_{\alpha l\beta m}(\theta) = e^{-i\rho_\alpha l/B} e^{i\rho_\beta m/B} e^{-2\pi^2(l-m-B(\delta_\alpha-\delta_\beta))^2} \Theta\left(2\pi i(l-m-B(\delta_\alpha-\delta_\beta))M - \frac{\theta}{2\pi}; 2\pi i M^2\right), \tag{47}$$

where $T_pB = M$ and where

$$\Theta(z;t) \stackrel{\text{def}}{=} \sum_k e^{i\pi k^2 t} e^{i2\pi kz} \tag{48}$$

is a Jacobi theta function.

Proof. Ignoring the factors associated with Doppler in Equ. (45), we have

$$\sum_k R\left(\frac{l-m}{B} - (\delta_\alpha - \delta_\beta) + k\frac{M}{B}\right) e^{-ik\theta} = \sum_k e^{-2\pi^2(r+kM)^2} e^{-ik\theta}, \tag{49}$$

with $r \stackrel{\text{def}}{=} l - m + B(\delta_\alpha - \delta_\beta)$. Write Equ. (48) as $\sum_k q^{ak^2+bk}$ with $q \stackrel{\text{def}}{=} e^{i\pi}$, $a \stackrel{\text{def}}{=} t$, and $b \stackrel{\text{def}}{=} 2z$. Now

$$\sum_k e^{-2\pi^2(r+kM)^2} e^{-ik\theta} = e^{-2\pi^2 r^2} \sum_k e^{-2\pi^2 M^2 k^2} e^{-(4\pi^2 rM + i\theta)k} = \sum_k q^{a'k^2+b'k}, \tag{50}$$

with $a' \stackrel{\text{def}}{=} i2\pi M^2$ and $b' \stackrel{\text{def}}{=} i(4\pi^2 rM + i\theta)/\pi$. Comparing a', b' with a, b and reintroducing the Doppler factors, the theorem follows. □

Of course, the results above provide little insight into the behavior of channel capacity with large delay and Doppler spreads. Such insight is attainable in the limiting cases of delay alone or Doppler alone, but a similar insight can be provided when delay and Doppler are correlated. This case has independent interest in certain applications and, sometimes, for certain signal types, so we will treat it briefly here. First, we state a result involving the eigenanalysis of a delay-Doppler operator.

Theorem 4. *Define the operator L such that $(Ls)(t) = e^{i\rho\delta/2}e^{-i\rho t}s(t-\delta)$. Define $g_\mu(t) \stackrel{\text{def}}{=} e^{-i\frac{\rho}{2\delta}(t+\mu/\rho)^2}$ for real μ. Then*

$$\begin{aligned}
(L^k s)(t) &= e^{i(k\rho)(k\delta)/2}e^{-ik\rho}s(t-k\delta), \quad k \in \mathbb{Z} \\
\int (L^j f_1)(t)\overline{f_2}(t)\,dt &= \int f_1(t)\overline{(L^{-j}f_2)}(t)\,dt \\
(Lg_\mu)(t) &= e^{i\mu}g_\mu(t) \\
\int g_{\mu_1}(t)\overline{s}_{\mu_2}(t)\,dt &= 2\pi\delta\cdot\delta_{\text{Kron}}(\mu_1-\mu_2) \\
\int \hat{f}(\mu)g_\mu(x)\,\frac{d\mu}{2\pi\delta} &= f(x) \\
\int f_1(t)\overline{f}_2(t)\,dt &= \int \hat{f}_1(\mu)\overline{\hat{f}}_2(\mu)\,d\mu/(2\pi\delta) \\
\left(\widehat{L^k f}\right)(\mu) &= e^{ik\mu}\hat{f}(\mu), \qquad (51)
\end{aligned}$$

with the 4^{th} relation interpreted in a distributional sense. Here, we define $\hat{f}(\mu) \stackrel{\text{def}}{=} \int f(t)\overline{g}_\mu(t)\,dt$.

Proof. The results follow largely from the definitions. We are cavalier about the domain of the function $f(x)$ since we don't need a precise statement of the theorem. For example, it can be assumed that f is infinitely differentiable with compact support so that $\hat{f}(\mu)$ has nice (e.g., member of Schwartz space) properties. Although the $g_\mu(t)$ may initially appear mysterious, they can be seen to be chirped waveforms with chirp slope matched to that associated with the operator L. A slightly more formal, but, as presented, not rigorous justification interprets L as $L = e^{-(\delta\partial_t + i\rho t)}$. The eigenfunctions of the differential operator $\delta\partial_t + i\rho t$ are easily evaluated to be those given in the theorem. The connection between L in the theorem and the differential operators is provided by the Lie-Trotter theorem (for finite matrices) which suggests that as $n \to \infty$, $\left(e^{-\delta\partial_t/n}e^{-i\rho t/n}\right)^n \to e^{-(\delta\partial_t + i\rho t)}$. Here, it is important to understand that $(e^{\delta\partial_t}s)(t) = \sum_{n=0}^{\infty}\frac{\partial^n s(t)}{n!}\delta^n = s(t+\delta)$ expresses a shift as a Taylor series expansion. □

For a stochastic version of Thm. 4, let $R(\tau) = \text{E}[s(t+\tau)\overline{s}(t)]$ represent the autocorrelation of the stochastic signal $s(t)$. Then, formally,

$$\text{E}[\hat{s}(\mu)\overline{\hat{s}(\nu)}] = \int\int R(t_1-t_2)\overline{g_\mu(t_1)}g_\mu(t_2)\,dt_1 dt_2 = \delta\int \hat{R}(\rho\omega)e^{-i\omega(\mu-\nu)}\,d\omega. \qquad (52)$$

Furthermore, we have

$$(L^j s)(t) = \int \hat{s}(\mu) e^{ij\mu} g_\mu(t)\, d\mu/2\pi\delta. \tag{53}$$

Thus

$$\begin{aligned} \mathrm{E}[(L^j s)(t+\tau)\overline{(L^k s)(t)}] &= 2\pi\delta \int \hat{R}(\rho\omega) \left(\int g_\mu(t+\tau) e^{-i(\omega-j)\mu}\, d\mu \right) \cdot \overline{\left(\int g_\nu(t) e^{-i(\omega-k)\nu}\, d\nu \right)}\, d\omega/2\pi \\ &= e^{i\delta\rho(j^2-k^2)/2} \int \hat{R}(\omega) e^{-i\delta\omega(j-k)} e^{-i\omega\tau}\, d\omega/2\pi, \end{aligned} \tag{54}$$

using

$$\int g_\nu(t) e^{i\omega\nu}\, d\nu = \sqrt{-i2\pi\delta\rho} e^{i\delta\rho\omega^2/2} e^{-i\omega\rho t}. \tag{55}$$

If $\hat{R}(\omega)$ is bandlimited to bandwidth B, we can define $v_j(\omega) \stackrel{\text{def}}{=} e^{i\delta\rho j^2} e^{-i\delta B j\omega}$, so that Equ. (54) becomes

$$\mathrm{E}[(L^j s)(t+k/B)\overline{(L^k s)(t)}] = B \int_{-\pi}^{\pi} \hat{R}(B\omega) \mathbf{v}(\omega) \mathbf{v}^\dagger(\omega) e^{-i\omega k}\, d\omega/2\pi. \tag{56}$$

If the array receives a signal $r_j(t) \stackrel{\text{def}}{=} (L^j s)(t)$ at the j^{th} sensor, then the space-time covariance of the signal is expressed through Equ. (56). We can apply Cor. 1 to realize channel capacities built from spectral matrices of the form $\sum_k \mathbf{v}_k(\omega)\mathbf{v}_k^\dagger(\omega)$. We can summarize this more concretely.

Theorem 5. *Let L be defined as in Thm. 4. Assume that the k^{th} sensor has input $\sum_j (L^{k_j} s_j)(t)$, where $\{s_j(t)\}$ are stationary signals with autocorrelation matrices proportional to $\{R_j(\tau)\}$. Define the k^{th} component of $\mathbf{v}_j(\omega)$ as $B\hat{R}_j(-B\omega)e^{i\delta\rho k_j^2} e^{i\delta B k_j \omega}$. If $\mathbf{v}(\omega) \equiv \mathbf{v}_1$, then defining $\mathbf{C}_I(\omega) = \sum_{j\neq 1} \mathbf{v}_j(\omega)\mathbf{v}_j^\dagger(\omega) + \mathbf{I}_n$ and using the notation of Cor. 1, the spectral efficiency of the Nyquist sampled channel becomes*

$$\begin{aligned} &\int_{-\pi}^{\pi} \log_2(1 + \mathbf{v}^\dagger(\omega)\hat{\mathbf{C}}_I(\omega)^{-1}\mathbf{v}(\omega))\, d\omega/2\pi \\ &\quad\approx \log_2(e)\, \mathrm{tr}\left(\left(\int_{-\pi}^{\pi} \mathbf{C}_I(\omega)^{-1}\, d\omega/2\pi \right) \cdot \left(\int_{-\pi}^{\pi} \mathbf{v}(\omega)\mathbf{v}^\dagger(\omega)\, d\omega/2\pi \right) \right) \\ &\quad\approx \log_2(e) \frac{n-k_I}{n} \cdot \textit{average ASNR}, \end{aligned} \tag{57}$$

where the first approximation assumes low ASNR and decorrelation of the $\mathbf{C}_I^{-1}(\omega)$ and $\mathbf{v}(\omega)\mathbf{v}^\dagger(\omega)$. The second approximation assumes that the $\mathbf{C}_I^{-1}(\omega)$ behave like random projectors onto the ortho-complements of the span of the interferers $\{\mathbf{v}_k(\omega)\}_{k>1}$.

Proof. The results follow from Cor. 1 and its approximations as well as Equ. (56). Note that the noise floor is represented by $\mathbf{I}_n$. Thus $\hat{R}_{\text{noise}}(\omega) \equiv 1/B$ expresses the normalized noise spectral density over $[-\pi B, \pi B]$. The $\hat{R}(\omega)$ for the signals is defined relative to this noise floor. The decorrelating and random behaviors are typical with distributed arrays that have large delay-Doppler shifts. Delay is large when it spreads over multiple inverse bandwidths and Doppler is

large when frequency spreads have multiple cycles over the intervals used for applying beamformer weights. □

Although the final approximation in the theorem holds when delay and Doppler shifts are correlated, as expressed through the delay-Doppler operator L defined in Thm. 4, it appears to hold empirically for unrelated, but still large, delay and Doppler shifts.

This page intentionally left blank.

3. MUTUAL INFORMATION AND TAP SELECTION

The choice of element locations to facilitate adaptive beamforming is typically a combinatorial optimization problem that is prohibitive for all but the smallest arrays. There are many figures of merit that can be used to design arrays including aperture, Cramer-Rao bounds, and sidelobe levels. Some of these measures address the direction-finding capabilities of the arrays while others also apply to beamforming. Of particular interest here is the performance measure involving the mutual information between a signal and the observations seen at the array elements. In the absence of any interference, this mutual information has a property, called submodularity, that guarantees that a greedy optimization procedure will achieve a level of performance that is close to the optimum. See [SJW17] and [SJW19]. Unfortunately, in the presence of interference, submodularity no longer holds. As examples show, however, it is still possible to obtain good array designs using greedy approaches and mutual information. We discuss some criteria for submodularity and examine their applicability to the design of antenna arrays. We also provide examples of arrays designed using greedy optimization of mutual information.

Finally, since a main topic of this report is the layout of delay-Doppler taps for the signal processing architecture of Fig. 3, we treat an example of tap placement using greedy optimization of the criterion expressed by Equ. (38), which was motivated based on mutual information.

3.1 DEFINITION OF SUBMODULARITY AND ITS UTILITY

We consider functions defined on subsets of a universe Ω. For example, $X \subseteq \Omega$ can represent the element locations of an antenna array while $f(X)$ represents the mutual information between the array's observations and a particular signal. Optimizing $f(X)$ is typically difficult since the optimization involves a combinatorial search. It is possible to employ a greedy search in case the function f possesses a property called submodularity that is related to convexity. Given X, Y, Z disjoint, we say f is submodular if

$$f(X \cup Y \cup Z) - f(Y \cup Z) \leq f(X \cup Z) - f(Z). \tag{58}$$

The functions we consider are also monotonic in their arguments. In other words, $X \subseteq Y$ implies $f(X) \leq f(Y)$. In the following, monotonicity is assumed.

We present a simple argument that shows the importance of submodularity and monotonicity. Stronger results are known.

Theorem 6. *Form the successive greedy estimates* $x_k \stackrel{\text{def}}{=} \max_x f(x \cup \{\cup_{j<k} x_j\})$ *for singleton sets* $x_k \subseteq \Omega$. *Define* $X \stackrel{\text{def}}{=} \cup_{k=1}^{m} x_k$. *If* $Y \subseteq \Omega$ *maximizes* f *over all sets* Y *with card*$(Y) = m$, *then* $f(X) \geq f(Y)/2$.

Proof. Define, and note for $k < m$,

$$r_m \stackrel{\text{def}}{=} f(\cup_{i=1}^{m} x_i) - f(\cup_{i=1}^{m-1} x_i) \leq f(x_m \cup \{\cup_{i=1}^{k} x_i\}) - f(\cup_{i=1}^{k} x_i) \leq f(\cup_{i=1}^{k+1} x_i) - f(\cup_{i=1}^{k} x_i) = r_k. \tag{59}$$

Letting $f(\emptyset) = 0$, we have $f(X) = \sum_{i=1}^{m} r_i \geq m r_m$. Assuming for the moment that $X \cap Y = \emptyset$, observe

$$f(Y \cup X) - f(X) = \sum_{i=1}^{m} [f(\{\cup_{j=1}^{i} y_j\} \cup X) - f(\{\cup_{j=1}^{i-1} y_j\} \cup X) \leq \sum_{i=1}^{m} f(y_i \cup \{\cup_{j=1}^{m-1} x_j\}) - f(\{\cup_{j=1}^{m-1} x_j\}) \leq m r_m. \tag{60}$$

Putting things together,

$$f(Y) - f(X) \leq f(Y \cup X) - f(X) \leq m r_m \leq f(X) \tag{61}$$

and hence, $f(Y) \leq 2f(X)$, as required. To handle the case when $X \cap Y = X_0$, write

$$f(Y) - f(X) \leq f(Y \cup X) - f(X) = f(Y_1 \cup X_1 \cup X_0) - f(X_1 \cup X_0) \leq f(Y_1 \cup X_1) - f(X_1) \leq f(X_1) \leq f(X), \tag{62}$$

where $Y = Y_1 \cup X_0$, $X = X_1 \cup X_0$, with disjoint subsets. The reasoning used to obtain Equ. (62) is the same used for Equ. (61), but applied to $Y_1 \cup X_1$ and X_1. □

3.2 CONDITIONS FOR SUBMODULARITY

Theorem 7. *The mutual information between the random variables S and the collection of random variables X drawn from the universe Ω, $f(X) \stackrel{\text{def}}{=} I(X; S)$ for $X \subseteq \Omega$, is submodular if and only if*

$$I(X; Y|Z) \geq I(X; Y|Z, S) \tag{63}$$

where $X, Y, Z \subseteq \Omega$ are disjoint.

Proof. Submodularity amounts to showing

$$I(X, Y, Z; S) - I(Y, Z; S) \leq I(X, Z; S) - I(Z; S). \tag{64}$$

Expanding the definitions,

$$I(X, Z; S) - I(Z; S) = H(X, Z) - H(X, Z|S) - (H(Z) - H(Z|S)) \tag{65}$$

and

$$I(X, Y, Z; S) - I(Y, Z; S) = H(X, Y, Z) - H(X, Y, Z|S) - (H(Y, Z) - H(Y, Z|S)). \tag{66}$$

Subtracting Equ. (65) from Equ. (66), we get

$$(H(X, Y, Z) - H(Y, Z)) - (H(X, Y, Z|S) - H(Y, Z|S)) - (H(X, Z) - H(Z)) + (H(X, Z|S) - H(Z|S)). \tag{67}$$

Using the information theory chain rule (i.e., $H(A, B) = H(A) + H(B|A)$), we can write this as

$$\begin{aligned} H(X|Y, Z) - H(X|Y, Z, S) - (H(X|Z) - H(X|Z, S)) &= (H(X|Z, S) - H(X|Y, Z, S)) \\ &- (H(X|Z) - H(X|Y, Z)) \\ &= I(X; Y|Z, S) - I(X; Y|Z). \end{aligned} \tag{68}$$

□

In some cases, it is easy to show submodularity. For example:

Theorem 8. *Assuming X, Y, Z are independent when conditioned on S, $I(X;S)$ is submodular.*

Proof. In this case $I(X;Y|Z,S) = 0$. Then Thm. 7 implies submodularity. □

However, it is easy to construct cases where Thm. 7 fails.

Theorem 9. *If either of the random variables X or Y are independent of S, then*

$$I(X;Y|S) \geq I(X;Y). \tag{69}$$

Proof. Employing the standardly abused notation (e.g., $p(s)$ and $p(y)$ refer to probability density functions associated with the random variables S and Y, respectively),

$$\begin{aligned} I(X;Y) = H(X) - H(X|Y) = \int\int \left(\int p(y|s)p(s)\, ds \right) \\ \cdot \quad \left[\log \frac{\int p(x|y,s)p(s)\, ds}{\int p(x|s)p(s)\, ds} \right] \cdot \left(\int p(x|y,s)p(s)\, ds \right) dxdy, \end{aligned} \tag{70}$$

which follows by computing the inner integrals explicitly (e.g., $\int p(y|s)p(s)\, ds = p(y)$). By the log-sum inequality, Equ. (70) is less than or equal to

$$\int \left(\int p(y|s)p(s)\, ds \right) p(s)p(x|y,s) \log \frac{p(x|y,s)}{p(x|s)}\, dsdxdy = \int p(y)p(s)p(x|y,s) \log \frac{p(x|y,s)}{p(x|s)}\, dsdxdy. \tag{71}$$

In turn, Equ. (71) becomes, when $p(y|s) = p(y)$,

$$\int p(y|s)p(s)p(x|y,s) \log \frac{p(x|y,s)}{p(x|s)}\, dsdxdy = H(X|S) - H(X|Y,S) = I(X;Y|S). \tag{72}$$

So far, the result has been demonstrated when Y is independent of S. The symmetry of the mutual information expressions shows that the same result holds when it is X that is independent of S. □

A simple example of a situation when independence implies increased mutual information under conditioning is given by the additive model $Y = X + N$, where S and X are independent, but S shares information with N. Conditioning on S increases the mutual information as can be seen from

$$I(X;Y) = H(X) - H(Y|X) = H(X) - H(N) \tag{73}$$

$$I(X;Y|S) = H(X) - H(N|S) \geq H(X) - H(N), \tag{74}$$

since conditioning only reduces entropy.

3.3 GAUSSIAN SIGNAL MODEL

Based on the discussion of Section 3, it may seem that submodularity can fail for the mutual information between a signal and the random variables representing observations of data. This is indeed the case. However, the failure of perfect submodularity as characterized by Thm. 7 is mitigated by the approximate submodularity exhibited in some interesting problem regimes. To proceed, we examine mutual information and the criterion of Thm. 7 using a Gaussian signal model.

3.3.1 Capacity Expressions

Let the $n \times l$ matrix $\mathbf{Z}$ consist of complex, circular, zero-mean, Gaussian observations from different sensors. Each row of $\mathbf{Z}$ is a time series of a sensor output. It is assumed that the columns of $\mathbf{Z}$ are i.i.d., with covariance matrix $\mathbf{R}$ between the entries in a column. Then the distribution of $\mathbf{Z}$ can be written

$$p(\mathbf{Z}|\mathbf{R}) = \pi^{-nl}|\mathbf{R}|^{-l}e^{-\mathrm{tr}\mathbf{R}^{-1}\mathbf{Z}\mathbf{Z}^{\dagger}}. \tag{75}$$

Then

$$H(Z) = -\mathrm{E}[\log p(\mathbf{Z}|\mathbf{R})] = ln\log(\pi e) + l\log|\mathbf{R}|. \tag{76}$$

Consider jointly distributed, zero-mean, complex, circular Gaussian random vectors X, Y with covariance matrix

$$\mathbf{R} = \begin{pmatrix} \mathbf{R}_{XX} & \mathbf{R}_{XY} \\ \mathbf{R}_{YX} & \mathbf{R}_{YY} \end{pmatrix}. \tag{77}$$

Then

$$\begin{aligned} I(X;Y) = H(X) + H(Y) - H(X,Y) &= lm\log(\pi e) + l\log|\mathbf{R}_{XX}| + l(n-m)\log(\pi e) \\ &+ l\log|\mathbf{R}_{YY}| - ln\log(\pi e) - l\log|\mathbf{R}| \\ &= -l\log\frac{|\mathbf{R}_{XX} - \mathbf{R}_{XY}\mathbf{R}_{YY}^{-1}\mathbf{R}_{YX}|}{|\mathbf{R}_{XX}|} \end{aligned} \tag{78}$$

Given an additional Gaussian random vector $\mathbf{Z}$, write

$$\begin{pmatrix} \mathbf{X} \\ \mathbf{Y}' \end{pmatrix} = \begin{pmatrix} \mathbf{X} \\ \mathbf{Y} \\ \mathbf{Z} \end{pmatrix} \tag{79}$$

with a covariance partitioned analogously to that of Equ. (77). Then we can consider the conditional covariance of X given Z and denote it by $\mathbf{R}^Z$. In particular, $\mathbf{R}^Z_{XX} = \mathbf{R}_{XX} - \mathbf{R}_{XZ}\mathbf{R}_{ZZ}^{-1}\mathbf{R}_{ZX}$. Since they both represent the inverted component $(\mathbf{R}^{XX})^{-1}$ of the inverse of the full covariance matrix of X, Y, Z, we have

$$\mathbf{R}^Z_{XX} - \mathbf{R}^Z_{XY}(\mathbf{R}^Z_{YY})^{-1}\mathbf{R}^Z_{YX} = \mathbf{R}_{XX} - \mathbf{R}_{XY'}\mathbf{R}^{-1}_{Y'Y'}\mathbf{R}_{Y'X}. \tag{80}$$

Thus, we have

$$I(X;Y|Z) \leq I(X;Y,Z) \tag{81}$$

using Equ. (78), Equ. (80), and the form of $\mathbf{R}^Z_{XX}$.

3.3.2 Signal Models

To demonstrate submodularity for the a Gaussian signal model, we investigate $I(X;Y|Z) - I(X;Y|Z,S)$. But first, we describe the relevant additive noise model. Let $X = \sum_k s_k + n$ denote a random variable consisting of the sum of independent zero-mean, complex Gaussians of complex variance $\mathrm{E}[|s_j|^2] = p_k$. Given m summands s_j, we denote by $\mathbf{V}$ the matrix of wavefronts (in columns of $\mathbf{V}$) for each of the signals s_j. Let $\mathbf{Y} = \mathbf{V}\mathbf{s} + \mathbf{N}$ denote the random vector of observed data in additive noise $\mathbf{N}$, which has i.i.d. entries that are zero-mean, complex Gaussian, unit complex variance random variables. Letting $\mathbf{p}$ denote the vector with components p_k, we find

$$\begin{aligned}
\mathbf{R}_{YX} &= \mathbf{V}\mathbf{p} \\
\mathbf{R}_{XX} &= 1 + \mathrm{tr}\mathbf{P} \\
\mathbf{R}_{YY} &= \mathbf{I} + \mathbf{V}\mathbf{P}\mathbf{V}^\dagger \\
\mathbf{R}_{XY}\mathbf{R}_{YY}^{-1}\mathbf{R}_{YX} &= \mathbf{p}^\dagger\mathbf{V}^\dagger(\mathbf{I} - \mathbf{V}(\mathbf{P}^{-1} + \mathbf{V}^\dagger\mathbf{V})^{-1}\mathbf{V}^\dagger)\mathbf{V}\mathbf{p} \\
&= \mathbf{p}^\dagger(\mathbf{V}^\dagger\mathbf{V})^{1/2}(\mathbf{I} - (\mathbf{I} + (\mathbf{V}^\dagger\mathbf{V})^{-1/2}\mathbf{P}^{-1}(\mathbf{V}^\dagger\mathbf{V})^{-1/2})^{-1})(\mathbf{V}^\dagger\mathbf{V})^{1/2}\mathbf{p} \\
&= \mathbf{p}^\dagger(\mathbf{V}^\dagger\mathbf{V})^{1/2}[(\mathbf{V}^\dagger\mathbf{V})^{-1/2}\mathbf{P}^{-1}(\mathbf{V}^\dagger\mathbf{V})^{-1/2}(\mathbf{I} + (\mathbf{V}^\dagger\mathbf{V})^{-1/2}\mathbf{P}^{-1}(\mathbf{V}^\dagger\mathbf{V})^{-1/2})^{-1}](\mathbf{V}^\dagger\mathbf{V})^{1/2}\mathbf{p} \\
&= \mathbf{p}^\dagger(\mathbf{V}^\dagger\mathbf{V})^{1/2}(\mathbf{I} + (\mathbf{V}^\dagger\mathbf{V})^{1/2}\mathbf{P}(\mathbf{V}^\dagger\mathbf{V})^{1/2})^{-1}(\mathbf{V}^\dagger\mathbf{V})^{1/2}\mathbf{p} \\
&= \mathbf{p}^\dagger\mathbf{W}(\mathbf{I} + \mathbf{W}\mathbf{P}\mathbf{W})^{-1}\mathbf{W}\mathbf{p}, \qquad (82)
\end{aligned}$$

with $\mathbf{W} \stackrel{\mathrm{def}}{=} (\mathbf{V}^\dagger\mathbf{V})^{1/2}$.

3.4 HIGH SNR APPROXIMATIONS

When $\mathbf{W}\mathbf{P}\mathbf{W} \gg I$, we have the following approximations, letting $\mathbf{A} \stackrel{\mathrm{def}}{=} \mathbf{W}\mathbf{P}\mathbf{W}$:

$$\begin{aligned}
\mathbf{p}^\dagger\mathbf{W}(\mathbf{I} + \mathbf{W}\mathbf{P}\mathbf{W})^{-1}\mathbf{W}\mathbf{p} &= \mathbf{p}^\dagger\mathbf{W}(\mathbf{I} + \mathbf{A})^{-1}\mathbf{W}\mathbf{p} \\
&= \mathbf{p}^\dagger\mathbf{W}\mathbf{A}^{-1/2}(\mathbf{I} + \mathbf{A}^{-1})^{-1}\mathbf{A}^{-1/2}\mathbf{W}\mathbf{p} \\
&\approx \mathbf{p}^\dagger\mathbf{W}\mathbf{A}^{-1}\mathbf{W}\mathbf{p} - \mathbf{p}^\dagger\mathbf{W}\mathbf{A}^{-2}\mathbf{W}\mathbf{p} \\
&= \mathbf{p}^\dagger\mathbf{P}^{-1}\mathbf{p} - \mathbf{p}^\dagger\mathbf{P}^{-1}\mathbf{W}^{-2}\mathbf{P}^{-1}\mathbf{p} \\
&= \mathrm{tr}\mathbf{P} - \mathbf{1}^T\mathbf{W}^{-2}\mathbf{1} = \mathrm{tr}\mathbf{P} - \mathbf{1}^T(\mathbf{V}^\dagger\mathbf{V})^{-1}\mathbf{1}. \qquad (83)
\end{aligned}$$

Thus,

$$\frac{|\mathbf{R}_{XX} - \mathbf{R}_{XY}\mathbf{R}_{YY}^{-1}\mathbf{R}_{YX}|}{|\mathbf{R}_{XX}|} \approx 1 - \frac{\mathrm{tr}\mathbf{P} - \mathbf{1}^T(\mathbf{V}^\dagger\mathbf{V})^{-1}\mathbf{1}}{1 + \mathrm{tr}\mathbf{P}} = \frac{1 + \mathbf{1}^T(\mathbf{V}^\dagger\mathbf{V})^{-1}\mathbf{1}}{1 + \mathrm{tr}\mathbf{P}}. \qquad (84)$$

Now, we can put together expressions to get the mutual information metrics we want. First,

$$I(X;Y,Z) = -l\log\frac{|\mathbf{R}_{XX} - \mathbf{R}_{XY'}\mathbf{R}_{Y'Y'}^{-1}\mathbf{R}_{Y'X}||}{|\mathbf{R}_{XX}|} \approx -l\log\frac{1 + \mathbf{1}^T(\mathbf{V}^\dagger\mathbf{V})^{-1}\mathbf{1}}{1 + \mathrm{tr}\mathbf{P}}, \qquad (85)$$

where we have used the general result Equ. (84) for a particular example using random vectors X, Y, Z. Similarly, we have

$$I(X;Z) = -l\log\frac{|\mathbf{R}_{XX} - \mathbf{R}_{XZ}\mathbf{R}_{ZZ}^{-1}\mathbf{R}_{ZX}||}{|\mathbf{R}_{XX}|} = -l\log\frac{|\mathbf{R}_{XX}^Z|}{|\mathbf{R}_{XX}|} \approx -l\log\frac{1 + \mathbf{1}^T(\mathbf{V}_1^\dagger\mathbf{V}_1)^{-1}\mathbf{1}}{1 + \mathrm{tr}\mathbf{P}}, \qquad (86)$$

where $\mathbf{V} = (\mathbf{V}_x^T \mathbf{V}_y^T \mathbf{V}_z^T)^T$ is the partitioning of $\mathbf{V}$ corresponding to the random vectors X, Y, Z, and $\mathbf{V}_1 = (\mathbf{V}_x^T \mathbf{V}_z^T)^T$. Using Equ. (80) as well as Equ. (85) and Equ. (86), we have

$$I(X;Y|Z) = -l \log \frac{|\mathbf{R}_{XX}^Z - \mathbf{R}_{XY}^Z (\mathbf{R}_{YY}^Z)^{-1} \mathbf{R}_{YX}^Z|}{|\mathbf{R}_{XX}^Z|} \approx -l \log \frac{1 + \mathbf{1}^T (\mathbf{V}^\dagger \mathbf{V})^{-1} \mathbf{1}}{1 + \mathbf{1}^T (\mathbf{V}_1^\dagger \mathbf{V}_1)^{-1} \mathbf{1}}. \tag{87}$$

The primary advantage of this high SNR expression is the elimination of the specific SNR parameters p_k.

Both $\mathbf{V}$ and $\mathbf{V}_1$ have m columns, corresponding to the m signals $\{s_k\}$. Consider conditioning on one of these signals, say $S \stackrel{\text{def}}{=} s_m$. The effect of conditioning is to remove the signal s_m from the data, since conditioning puts s_m in the mean of the Gaussian random variables whereas it is only the covariance that matters for mutual information. Then the matrices $\mathbf{V}'$ and $\mathbf{V}_1'$ that apply upon conditioning on S consist of the first $m-1$ columns of $\mathbf{V}$ and $\mathbf{V}_1$, respectively. Since Equ. (87) applied to $\mathbf{V}'$, $\mathbf{V}_1'$ is still valid after conditioning, we have

$$I(X;Y|Z) - I(X;Y|Z,S) \approx -l \log \frac{1 + \mathbf{1}^T (\mathbf{V}^\dagger \mathbf{V})^{-1} \mathbf{1}}{1 + \mathbf{1}^T (\mathbf{V}'^\dagger \mathbf{V}')^{-1} \mathbf{1}} + l \log \frac{1 + \mathbf{1}^T (\mathbf{V}_1^\dagger \mathbf{V}_1)^{-1} \mathbf{1}}{1 + \mathbf{1}^T (\mathbf{V}'^\dagger_1 \mathbf{V}_1')^{-1} \mathbf{1}}. \tag{88}$$

It is easy to see that the LHS of Equ. (88) is nonnegative if $m = 1$, which is a fact we already knew from the conditional independence of X, Y, and Z from S that holds when $m = 1$.

The expression Equ. (85) holds when $n \geq m$, where $\mathbf{V}$ has dimensions $n \times m$. More generally, we have the relation

$$I(X;Y,Z) \approx \begin{cases} -l \log \frac{1+\mathbf{1}^T (\mathbf{V}^\dagger \mathbf{V})^{-1} \mathbf{1}}{1+\mathrm{tr}\mathbf{P}} & n \geq m \\ -l \log \frac{1+\mathbf{1}^T [\mathbf{P} - \mathbf{P}\mathbf{P}_V \mathbf{P}^{-1} \mathbf{P}_V \mathbf{P}] \mathbf{1}}{1+\mathrm{tr}\mathbf{P}} & n < m. \end{cases} \tag{89}$$

This expression leads us to the conditional mutual information

$$I(X;Y|Z) \approx \begin{cases} -l \log \frac{1+\mathbf{1}^T (\mathbf{V}^\dagger \mathbf{V})^{-1} \mathbf{1}}{1+\mathbf{1}^T (\mathbf{V}_1^\dagger \mathbf{V}_1)^{-1} \mathbf{1}} & n > n_1 \geq m \\ -l \log \frac{1+\mathbf{1}^T (\mathbf{V}^\dagger \mathbf{V})^{-1} \mathbf{1}}{1+\mathbf{1}^T [\mathbf{P} - \mathbf{P}\mathbf{P}_{V_1} \mathbf{P}^{-1} \mathbf{P}_{V_1} \mathbf{P}] \mathbf{1}} & n \geq m > n_1 \\ -l \log \frac{1+\mathbf{1}^T [\mathbf{P} - \mathbf{P}\mathbf{P}_V \mathbf{P}^{-1} \mathbf{P}_V \mathbf{P}] \mathbf{1}}{1+\mathbf{1}^T [\mathbf{P} - \mathbf{P}\mathbf{P}_{V_1} \mathbf{P}^{-1} \mathbf{P}_{V_1} \mathbf{P}] \mathbf{1}} & m > n > n_1. \end{cases} \tag{90}$$

where we assume $\mathbf{V}_1$ has dimensions $n_1 \times m$. The projectors $\mathbf{P}_V$ are defined $\mathbf{P}_V \stackrel{\text{def}}{=} \mathbf{V}^\dagger (\mathbf{V}\mathbf{V}^\dagger)^{-1} \mathbf{V}$, which makes sense when $m \geq n$.

The behavior of the submodularity criterion of Thm. 7 is illustrated in Fig. 12 and Fig. 13. Although the criterion does not hold in the presence of background interference, it is often valid.

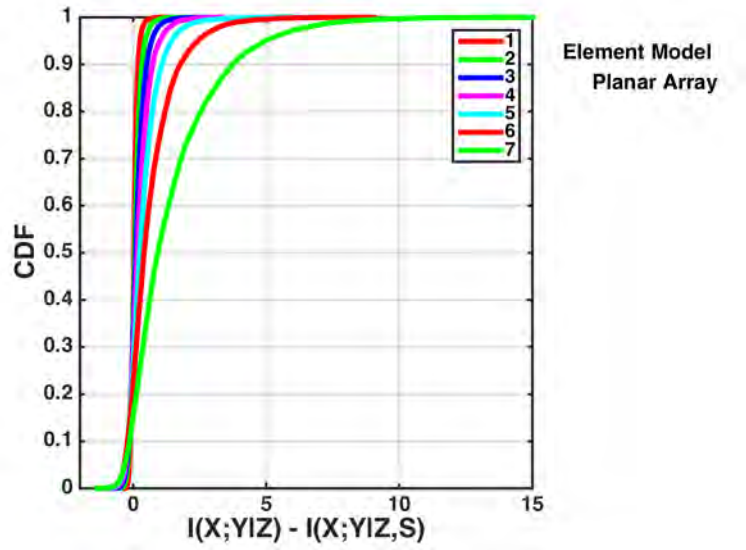


Figure 12. Shown is the CDF of the difference $I(X;Y|Z) - I(X;Y|Z,S)$ for an example where we use the approximation Equ. (88). $\mathbf{V}$ has dimensions 10×3 while $\mathbf{V}_1$ has dimensions that vary between 9×3 (curve 1) and 3×3 (curve 7). The array elements are drawn uniformly randomly from a planar box 10 wavelengths on a side. The CDF is largely, but not entirely nonnegative, indicating an approximate validity of the submodularity criterion of Thm. 7. Similar results hold for $\mathbf{V}$ and $\mathbf{V}_1$ with other dimensions.

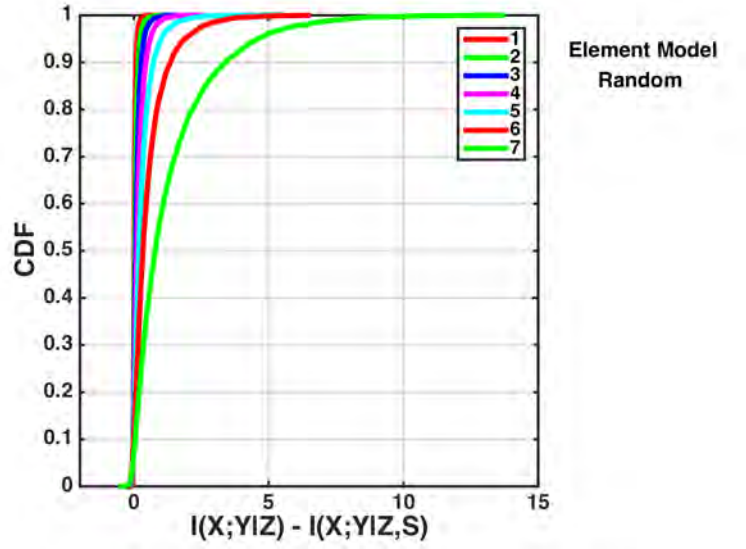


Figure 13. The submodularity criterion of Thm. 7 is examined under the approximation Equ. (88) when $\mathbf{V}$ is drawn from a random distribution with with i.i.d. components of unity magnitude and uniformly distributed phases. The CDFs are labeled according to the dimensions of $\mathbf{V}$ and $\mathbf{V}_1$ as before. Performance is very similar to that shown in Fig. 12, with even smaller negative tails.

3.5 APPLICATIONS OF GREEDY TAP SELECTION

We present some examples of array designs and tap placements that utilize mutual information.

3.5.1 Array Design

First, we present a criterion for locating antenna elements when a cochannel interferer is present. The technical discussion treats the design of linear arrays, but the extension to planar arrays is clear.

Define the response vector

$$(\mathbf{v}(u))_k = e^{i2\pi x_k u}. \tag{91}$$

The variable u designates the angle-of-arrival of a signal. All units of array element placement are in wavelengths. We assume an interference background due to a single signal with angle-of-arrival w. Then the interference background is expressed through the covariance matrix

$$\mathbf{C} \stackrel{\text{def}}{=} \mathbf{I}_n + p_I \mathbf{v}(w)\mathbf{v}^\dagger(w) \tag{92}$$

with interference SNR p_I.

The capacity of the channel for the signal u with SNR p_S in interference w is expressed by

$$\begin{aligned} \log_2\left(1 + p_S \mathbf{v}(u)^\dagger \mathbf{C}^{-1} \mathbf{v}(u)\right) &= \log_2\left(1 + np_S\left(1 - \frac{p_I}{1+np_I}\frac{|\mathbf{v}^\dagger(u)\mathbf{v}(w)|^2}{n}\right)\right) \\ &\to \log_2\left(1 + np_S\left(1 - \frac{|\sum_k e^{i2\pi x_k(w-u)}|^2}{n^2}\right)\right), \end{aligned} \tag{93}$$

where the limit is taken as $p_I \to \infty$.

If we now let u denote the angular separation between signal and interferer, then the average capacity over signal separations becomes

$$1/2 \int_{-1}^{1} \log_2\left(1 + np_S\left(1 - \frac{|\sum_k e^{i2\pi x_k u}|^2}{n^2}\right)\right) du. \tag{94}$$

In Fig. 14 we show an example of array design based on Equ. (94) when the desired signal is weak. In this case, greedily optimal arrays tend to have uniform distribution for element locations. Note that this design, as with the others shown in this section, is not claimed to be optimal, given that it is determined in a greedy fashion.

When the desired signal is stronger, the greedy mutual information approach designs sparser arrays, as show in Fig. 15 for the case of a 10 dB SNR. The array consists of subarrays with uniformly (roughly) spaced elements, but the subarrays are widely separated.

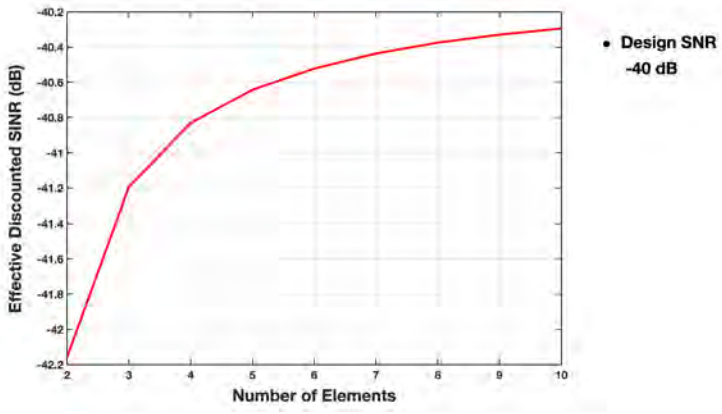
3.5.2 Tap Placement

A key application of the mutual information criterion involves the placement of delay-Doppler taps to support interference suppression. We show an example of tap placement in a scenario with three signals, two with 40 dB SNR and one with a 10 dB SNR. The delays and Doppler spreads are large and it would require around 1000 taps to cover the spreads. Of course, covering the spreads is not, in itself, enough to guarantee good performance. In Fig. 16, we examine the performance of a 129-tap solution in delay alone. The delay spread of the taps is more that enough to cover the delay spread of the signals. However, the Doppler has not been compensated and the performance is poor.

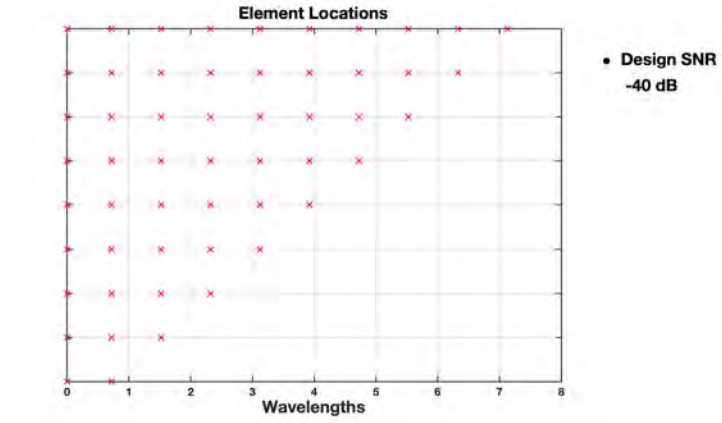
In Fig. 17 we show an intermediate form of signal processing. The tap delay line of Fig. 16 is used, but each sensor's output has a different Doppler shift applied. The Doppler shifts help performance, but it is still poor.

When a sparse tap layout is based on the greedy optimization of Equ. (38), the good performance of Fig. 18 results. In this case, only 108 taps are used. For both figures Fig. 16 and Fig. 18, there is enough data to support estimation of the beamformer weights.

Many of the approaches to tap placement that we consider in this report require estimates of the propagation channel in terms of the delays and Doppler shifts of the signals. Mutual information, or more explicitly, Equ. (38), only needs estimates of the some signal covariances at a range of delay and Doppler shifts. Assuming slow variations in the ideal tap locations, it is possible to use the data to estimate these covariances. One of the required covariances is really the cross-covariance between the data and a desired signal (for which the beamformer is designed). Estimation of the cross-covariance is possible if a portion of the desired signal is known and can be correlated with the data. We do not go into further details here, but the approach is straightforward.

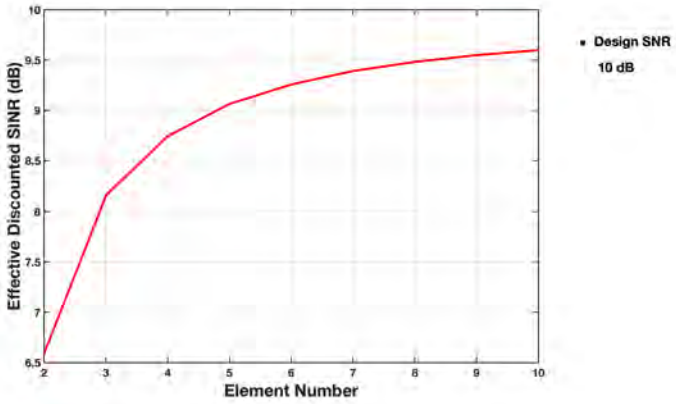


(a) As the number of sensors increases, so does the average capacity. Some of this capacity increase is due to additional gain. If we write the spectral efficiency as $\log_2(1 + n\mathrm{SNR}_{\mathrm{dis}})$*, where* n *is the number of sensors, we can plot the discounted capacity* $\mathrm{SNR}_{\mathrm{dis}}$ *as a function of the number of sensors.*

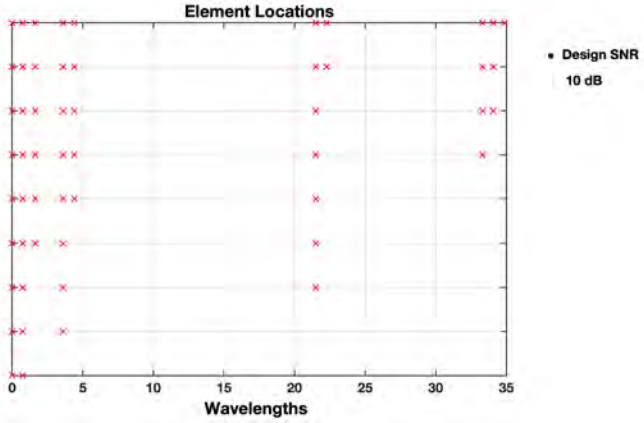


(b) The greedy search adds additional sensors one at a time as shown in the rows of the figure, from bottom to top.

Figure 14. For a weak signal (-40 dB SNR) and a strong interferer (asymptotic limit), we show the sensor locations for a one-dimensional antenna array chosen using a greedy search over the average capacity Equ. (94). The search occurs over a universe of 1000 possible tap locations spread over 40 wavelengths. The greedy solutions build nearly uniform linear arrays with an element spacing between 1/2 and 1 wavelength. The spacing is related to the spatial distribution of the interference. There is a slight variation in element spacings between the first two sensors and the rest of the adjacent pairs. This could be a numerical artifact.



(a) As the number of sensors increases, so does the average capacity. Some of this capacity increase is due to additional gain. If we write the spectral efficiency as $\log_2(1 + n\mathrm{SNR}_{\mathrm{dis}})$*, where* n *is the number of sensors, we can plot the discounted capacity* $\mathrm{SNR}_{\mathrm{dis}}$ *as a function of the number of sensors.*



(b) The greedy search adds additional sensors one at a time as shown in the rows of the figure, from bottom to top.

Figure 15. For a signal of 10 dB SNR and a strong interferer (asymptotic limit), we show the sensor locations for a one-dimensional antenna array chosen using a greedy search over the average capacity Equ. (94). The search occurs over a universe of 1000 possible tap locations spread over 40 wavelengths. The greedy solutions build uniformly spaced subarrays with wide spacings between subarrays. The smallest sensor spacings are roughly consistent with the spacings in Fig. 14 but are not identical. Again, numerical artifacts may cause the small variations in sensor spacings.

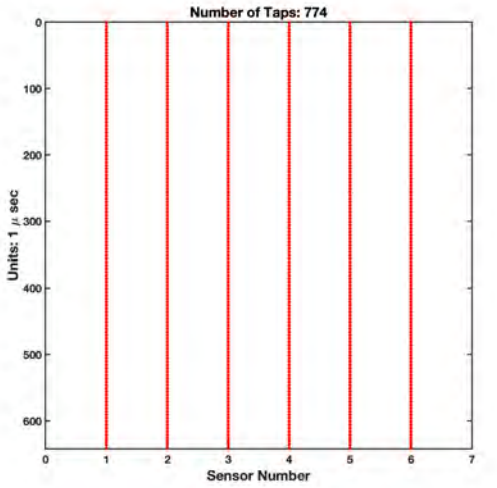
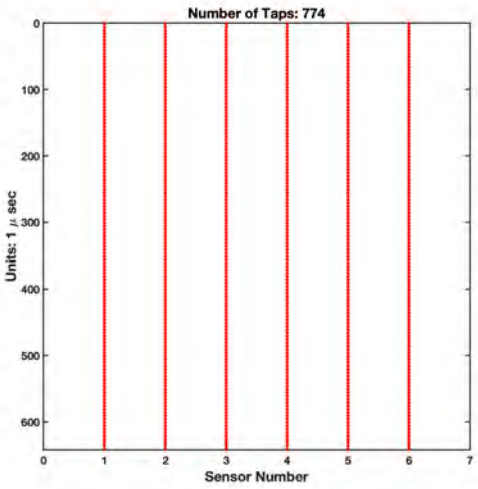
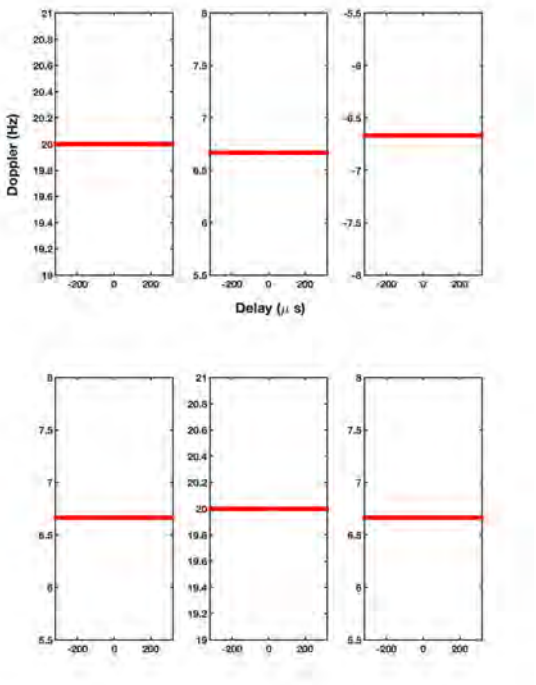


Figure 16. We consider an example scenario with 100 kHz bandwidth signals of duration 40 ms. Two of the three signals have 40 dB element SNRs while the remaining signal has a 10 dB SNR. The delay spread as seen among the six sensors is about 200 μ s and the Doppler spread is about 50 Hz for each signal. The time-bandwidth product provides about $4\,10^4$ samples while the delay spread and Doppler spread require about $6 \times 160 = 960$ taps to fully cover with dense taps spaced apart on a $\frac{1}{2B} \times \frac{1}{2Ts}$ grid. Thus, there may be enough sample support to estimate tap coeffcients, but the number of taps is large. Instead, we consider other tap placements using mutual information as the design metric. Shown here is a baseline delay-only tap placement with 129 taps, densely spaced, for each sensor. Performance is evaluated using effective SNR out of the beamformer, which is about -21 dB in this example. Even though the delay taps span over three times the signals' delay spread, delay-only taps cannot perform well.

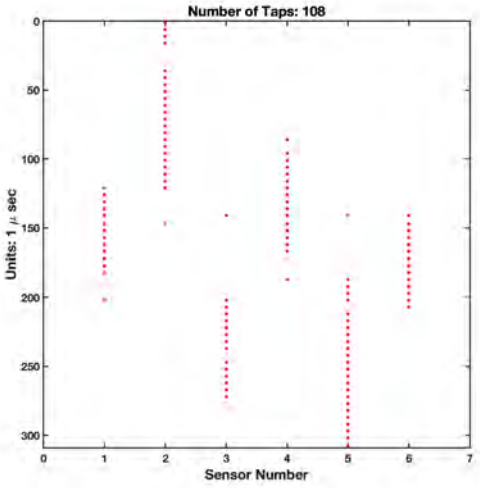


(a) The tapped delay-line remains the same as that in Fig. 16.

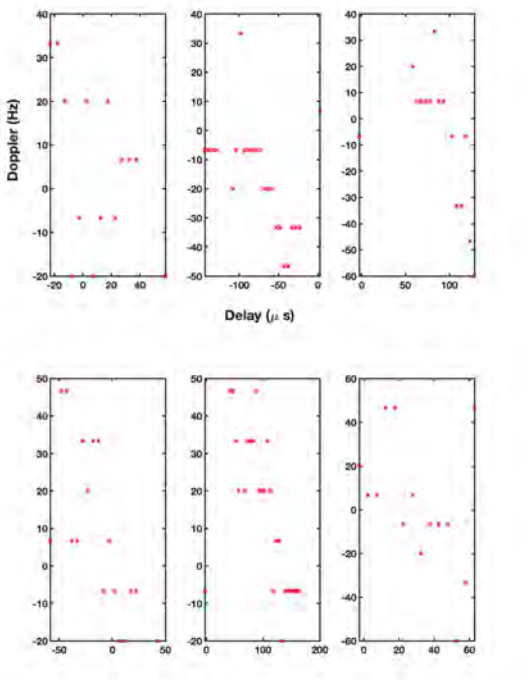


(b) Shown are the frequency shifts at each tap at for each sensor (subplot). The shifts are chosen to be the same for all taps on the same sensor.

Figure 17. In the same scenario as Fig. 16, we show the performance of the same tapped delay-line, but this time each sensor has a different frequency shift. The frequency shifts are chosen from a large range of possibilities using greedy selection based on mutual information. Performance is still poor, with an effective SINR of -14 dB.



(a) The delay components of the delay-Doppler taps are shown for each sensor.



(b) Scatter plots for the placement of taps in delay and frequency are shown for each sensor (subplot).

Figure 18. In the same scenario as Fig. 16 and Fig. 17, we show the performance of sparse delay-Doppler taps. The taps are chosen one at a time from a large range of possibilities using greedy selection based on mutual information. Performance is good now, with an effective SINR of 6 dB. The total number of taps is only 108 rather than the 960 or more that would be required for a dense grid of delay-Doppler taps.

4. SPARSE TAP SELECTION FOR ZERO-FORCING

In Fig. 4, we see that dense grids of delay-Doppler taps are limited in the extent of differential delay-Doppler products that they can handle. This is not intrinsically a problem if delay-Doppler taps are placed sparsely. The number of taps required for zero-forcing will not depend on the particular differential delay-doppler product. It will only depend on details such as number of sensors and signals. Thus, the problem of an increasing number of unknowns as the delay-Doppler spreads increases, does not occur, since the number of taps remains fixed; only their location changes. From the viewpoint of determining the tap coefficients, the number of unknowns remains fixed. Of course, the tap locations must be determined separately by other means.

In Section 4.1, we treat the special case of delay-only tap placement. It is relatively easy to describe a set of tap locations that support zero-forcing. The delays of the signals can be arbitrary. The solution presented is not the only one, nor is it a solution with the minimal number of taps. A more complete solution of the delay-only problem is treated in Section 5.

The delay-Doppler quasi-polynomials built from linear combinations $\sum c_{\rho\delta} w^\rho z^\delta$, $c_{\rho\delta} \in \mathbb{C}$, of the monomial terms $w^\rho z^\delta \equiv D^\rho \Delta^\delta$ are known as a special class of noncommutative polynomials called Ore polynomials [Ore33], at least in the case when the ρ and δ are nonnegative integers. Recall that the polynomials are noncommutative due to the relations $w^\rho z^\delta = e^{-i\rho\delta} z^\delta w^\rho$.

Zero-forcing is expressed in terms of the solution of a set of linear equations whose vector and matrix entries consist of Ore polynomials. To solve these quasi-polynomial systems, we need to form fractions of Ore polynomials in a meaningful sense. In order to accomplish this, we need to write $p(z,w)q^{-1}(z,w)$ equivalently as ${q'}^{-1}(z,w)p'(z,w)$ using the relation

$$p'(z,w)q(z,w) = q'(z,w)p(z,w). \tag{95}$$

This is just cross-multiplying the required equality, preserving the order of the factors. One can interpret Equ. (95) as finding a common multiple for the pair $p(z,w), q(z,w)$ in terms of Ore polynomials. It is known that Ore polynomials have common multiples. There are several methods for determining common multiples. Some approaches are discussed in Section 4.2.2, including a case when the ρ, δ are not necessarily integers. Section 4.2 begins with a more detailed discussion of the solution of linear equations using vectors and matrices with entries that are quasi-polynomials in delay-Doppler operators. A toy example is provided that also treats equalization of the desired signal, as well as zero-forcing for the interference.

In Section 4.3 we provide a simple example of zero-forcing for a low-dimensional, but nontrivial problem. The zero-forcing tap placements are explicit in terms of the signals' delay and Doppler spreads.

One general approach to finding the solution to a zero-forcing problem involves, in the commutative (i.e., delay or Doppler alone) case, Gaussian elimination. This can be accomplished using fractions in the delay or Doppler operators, as the case may be. With fractions of Ore polynomials, the same procedure can be employed more generally for delay-Doppler operators. However, the resulting expressions are complicated sums of products of Ore polynomials and their inverses. As shown in Section 4.2.2, it is possible to use a common multiple construction to clear denominators

along the way, resulting in zero-forcing weights with entries that are only Ore polynomials. The Gaussian elimination procedure is sketched in Section 4.4, assuming an ability to find common multiples.

4.1 ZERO-FORCING WITH DELAY-ONLY OPERATORS

Let the matrix (δ_{jk}) represent the delays of the k^{th} signal in the j^{th} channel. As above, we consider the polynomial-like delay operators $\sum_\alpha c_\alpha z^\alpha$, with $z \equiv \Delta^\alpha$. Let the $(j,k)^{th}$ entry of the matrix (a_{jk}) represent the complex amplitude of the k^{th} signal in the j^{th} channel. Let $\mathbf{V}(z)$ be a matrix operator with delay-operator entries of the form $V(z)_{jk} = a_{jk} z^{\delta_{jk}}$. Partition $\mathbf{V}(z)$ into its top $s \times s$ submatrix $\mathbf{V}_A(z)$ and a bottom $(n-s) \times s$ submatrix $\mathbf{V}_B(z)$. We have

$$\left(-\mathbf{V}_B(z)\mathbf{V}_A(z)^{-1},\ \mathbf{I}_{n-s}\right)\mathbf{V}(z) = 0. \tag{96}$$

Pulling out a determinant factor, we can write this as

$$\left(-\mathbf{V}_B(z)\,\mathrm{cofac}(\mathbf{V}_A(z)),\ |\mathbf{V}_A(z)|\,\mathbf{I}_{n-s}\right)\mathbf{V}(z) = 0 \tag{97}$$

where $\mathrm{cofac}(\mathbf{M})$ of a square matrix $\mathbf{M}$ is the cofactor matrix of the same dimensionality. Recall that the $(j,k)^{th}$ entry of $\mathrm{cofac}(\mathbf{M})$ is the product of $(-1)^{j+k}$ and the determinant of the submatrix formed from $\mathbf{M}$ by deleting the k^{th} row and j^{th} column. We have used $\mathbf{M}^{-1} = |\mathbf{M}|^{-1}\mathrm{cofac}(\mathbf{M})$ above. Since the cofactor is a matrix with delay-operator entries and the determinant is a delay-operator, it follows that the matrix

$$\left(-\mathbf{V}_B(z)\,\mathrm{cofac}(\mathbf{V}_A(z)),\ |\mathbf{V}_A(z)|\,\mathbf{I}_{n-s}\right) \tag{98}$$

has delay-operator entries and can be used to null interferers.

We can expand the determinant of $\mathbf{V}_A(z)$ as follows:

$$|\mathbf{V}_A(z)| = \sum_\pi \mathrm{sgn}\ \pi \left(\prod_j a_{j\pi(j)}\right) z^{\sum_j \delta_{j\pi(j)}} \tag{99}$$

where the sums and products involving j range over $1 \le j \le s$, and where π ranges over all permutations of $\{1, 2, \ldots, s\}$.

The $(l,k)^{th}$ entry of $-\mathbf{V}_B(z)\,\mathrm{cofac}(\mathbf{V}_A(z))$ is expressed by

$$-\sum_m a_{s+l\,m} z^{\delta_{s+l\,m}} (-1)^{k+m} \sum_\pi \mathrm{sgn}\ \pi \left(\prod_{j\neq k} \mathbf{A}^{(km)}_{j\,\pi(j)}\right) z^{\sum_{j\neq k} \Delta^{(km)}_{j\,\pi(j)}} \tag{100}$$

where the permutations π of the inner summation can be viewed as all permutations of $\{1, 2, \ldots, s\}$ subject to $\pi(k) = m$ as long as $\mathrm{sgn}\ \pi$ is computed without regard to $\pi(k)$. The matrices $\mathbf{\Delta}^{(km)}$ and $\mathbf{A}^{(km)}$ are built from (δ_{jk}) and (a_{jk}) respectively. They both consist of their top $s \times s$ block components with the k^{th} row and m^{th} column omitted.

The previous equation can be simplified somewhat. Let $\delta^{(kl)}$ denote the $s \times s$ matrix formed from the top $s \times s$ block component of δ by replacing the k^{th} row with the row vector $\delta_{s+l,\cdot}$. Similarly, let $a^{(kl)}$ be the upper $s \times s$ block of a with row k replaced by the row vector $a_{s+l,\cdot}$. By absorbing the sign factor $(-1)^{k+m}$ into the sgn function of the permutation π and using $\pi(k) = m$, we can write Equ. (100) as

$$-\sum_{\pi} \text{sgn } \pi \left(\prod a^{(kl)}_{j\,\pi(j)} \right) z^{\sum_j \delta^{(kl)}_{j\,\pi(j)}}. \tag{101}$$

To be a bit more explcit concerning the exponents of z in the delay-operators, denote

$$|d|_{\Sigma} \stackrel{\text{def}}{=} \bigcup_{\pi} \sum_{j=1}^{s} d_{j\,\pi(j)} \tag{102}$$

for any $s \times s$ matrix d. Then each of the last $(n-s)$ channels uses the taps $|\delta|_{\Sigma}$. For channel k, $1 \leq k \leq s$, the taps are given by

$$\bigcup_{1 \leq l \leq n-s} |\delta^{(kl)}|_{\Sigma}. \tag{103}$$

Since the previous unions are all disjoint given indeterminates δ_{jk}, the total number of taps over all channels is $(n-s)(s+1)!$.

There are clearly $n-s$ independent delay-operators that null the interference. There cannot be any more as a transformation to the frequency domain demonstrates.

For a given tap and signal there is one other tap on a different channel that aligns the signal at the same delay. For example, choose an element of $|\delta^{(kl)}|_{\Sigma}$ and consider the delay of signal m:

$$\delta_{km} + \sum_{r \neq k} \delta_{r\,\pi(r)} + \delta_{s+l,\,\pi(k)}. \tag{104}$$

If $\pi(k) = m$, then the same delay can be written as

$$\delta_{s+l,\,m} + \sum_{r} \delta_{r\,\pi(r)}, \tag{105}$$

in other words, as the delay of signal m in channel $s+l$ out of a unique tap in $|\delta|_{\Sigma}$. If $\pi(k) \neq m$, assume there is a permutation π' satisfying

$$\delta_{km} + \sum_{r \neq k} \delta_{r\,\pi(r)} + \delta_{s+l,\,\pi(k)} = \delta_{jm} + \sum_{r \neq j} \delta_{r\,\pi'(r)} + \delta_{s+l,\,\pi'(j)}. \tag{106}$$

If $j \neq k$, we must have $\pi'(j) = \pi(k)$, $\pi'(k) = m = \pi(j)$, and π and π' agree everywhere else. In particular, π' is uniquely determined by π.

4.2 ZERO-FORCING AS LINEAR SYSTEMS

4.2.1 Solving Linear Operator-Valued Equations

It is possible to treat beamforming problems in terms of the solution of matrix algebra problems in which the matrices have entries that are drawn, themselves, from a noncommutative algebra. For our applications, the underlying algebra of matrix entries is the algebra of delay-Doppler operators. Although the material presented here is not central to the main results, it does provide an important point of view of the beamforming problem. The treatment is concise, emphasizing the fact that beamforming solutions can be found that null interference and equalize the signal of interest. The beamforming weights have delay-Doppler operators as entries. These operators can be interpreted as tapped delay lines at each sensor that provide delays and Doppler shifts at each tap. The tap outputs are combined over all sensors outputs, resulting in the beamformed output.

Adaptive array processing involves solving linear equations of the form

$$\mathbf{\Gamma x} = \mathbf{b} \tag{107}$$

where the components of $\mathbf{\Gamma}$, $\mathbf{x}$, and $\mathbf{b}$ have values as operators. For our purposes, the operator components will be quasi-polynomials (finite number of terms) in delay and Doppler of the form

$$\sum_{jk} a_{jk} D^{j\rho} \Delta^{k\delta}. \tag{108}$$

All delays and Doppler shifts are multiples of the atomic delays and Dopplers δ and ρ, respectively. The entries are quasi-polynomials in the sense that the delay and Doppler monomials do not commute. These polynomials form a noncommutative (also called nonabelian) ring of operators that has the mathematical structure called an Ore ring [Ore33]. The ring can be extended by forming fractions of the form $q(D,\Delta)^{-1}p(D,\Delta)$ for quasi-polynomials p, q. In order for this to make sense, it is necessary to handle products of the form $q_1^{-1}p_1q_2^{-1}p_2$, which, in essence, means being able to write $p_1q_2^{-1} = q'^{-1}_2 p'_1$ for some p'_1, q'_2. This, in turn, is related to finding a (least) common multiple of q_2 and p_1, i.e., solving $q'_2p_1 = p'_1q_2$. Common multiples exist for the Ore ring based on quasi-polynomial delay-Doppler operators. The results on the Ore ring mentioned here are not proved, but can be found in the original sources as well as more recent literature.

Common multiples allow us to clear denominators from sums of fractions. For example, if we have the expression $a^{-1}c_a + b^{-1}c_b$, we can find (in our Ore ring) a', b' which provide the common multiple $b'a = m(a,b) = a'b$. Then $a^{-1}c_a + b^{-1}c_b = (b'a)^{-1}b'c_a + (a'b)^{-1}a'c_b = m(a,b)^{-1}(b'c_a + a'c_b)$, whose denominator is cleared by multiplying on the left by $m(a,b)$. This procedure can be iterated to cover more terms with the result that any element of the ring of fractions can be written in the form $q^{-1}p$ and hence, it can be made an element of the original ring of quasi-polynomials upon multiplying from the left by a quasi-polynomial.

To solve the system Equ. (107), we proceed by using Gaussian elimination on the linear system, expressing entries at each stage as fractions of the form $q^{-1}p$. This is accomplished by solving common multiple problems, as discussed above. The final solution vector has entries of the same form, whose denominators can be cleared, in the case when $b = 0$, to arrive at nulling weights

that are fraction-free. In other words, the nulling weights have delay-Doppler quasi-polynomial entries.

When equalization must be performed, as well as nulling, there are ways to combine nulling weights to form an equalized output. In the case of delay alone (or, dually, Doppler alone), polynomial operators form a conventional polynomial algebra in one variable. This algebra admits a greatest common divisor (gcd) for any two polynomials $p(x), q(x)$. Furthermore, we can find polynomials $a(x), b(x)$ such that $a(x)p(x) + b(x)q(x) = \gcd(p, q)$. For generic (chosen at random) polynomials $p(x), q(x)$, we have $\gcd(p, q) = 1$. If both $p(x)$ and $q(x)$ represent the effects of two different interference-nulling beamformers on the desired signal, then, as long as we have $\gcd(p, q) = 1$, the beamformer outputs can be combined to achieve both nulling and equalization. Combining, of course, involves multiplying each beamformer output by the delay-Doppler quasi-polynomial derived from the gcd algorithm, and then adding the results. When delay and Doppler are taken together, the Ore ring that results has a gcd algorithm. In this case, the gcd is formed by viewing the Ore polynomial as a polynomial in either delay or Doppler alone, with coefficients conventional polynomials in the remaining variable. Since the gcd algorithm creates fractions, the solution $a(x)p(x)+b(x)q(x) = 1$ has factors $a(x), b(x)$ whose coefficients are fractions (rational functions) in the remaining operator. The denominators can be cleared, but what results is a signal of interest that is equalized in the x-variable but not in the remaining variable. To handle the remaining equalization problem, we assume that we have three beamformer weights that perform nulling. These weights must, of course, be linearly independent over the Ore ring with fractions. We can use two pairs of beamformers to provide two different x-equalized solutions. Now we have arrived at the original, single operator, equalization problem, which we can solve using the polynomial gcd. An implication of this procedure is that it takes $s + 2$ sensors to null $s - 1$ interferers and equalize the one remaining signal of interest (s signals in total). If only delay or Doppler is present, it only takes $s + 1$ sensors. For conventional narrowband beamforming, only s sensors are required. Below we construct tap configurations that are somewhat less pessimistic. In a very simple example, nulling and equalization are achieved as shown in Fig. 19 and Fig. 20.

Some of the algebraic manipulations mentioned above are accomplished by solving systems of linear equations. We give a few examples next. Consider finding the common multiple of $a(x) = \sum_{k=0}^{m} a_k x^k$ and $b(x) = \sum_{k=0}^{n} b_k x^k$ where the polynomials can also have operator coefficients. For example, the x variable can represent either delay or Doppler shifts while the coefficients a_k can be polynomials in the other operator. After finding a common multiple polynomial with fractional (i.e., rational) expressions in its coefficients, we can clear the denominators of these fractions by multiplying through by a common multiple, found in the normal fashion since these coefficient fractions are elements of a commutative ring. More concretely, we want to solve

$$\underbrace{a'(x)}_{m+1 \text{ unknowns}} \qquad b(x) = \overbrace{\underbrace{b'(x)}_{n+1 \text{ unknowns}} \qquad a(x)}^{\deg m+n}, \tag{109}$$

where the coefficients are polynomials in the remaining delay or Doppler operator. The solution is obtained in terms of rational functions of the coefficients, which all commute with each other. The homogeneous system has more unknowns ($m + n + 2$) than equations ($m + n + 1$); hence it has a

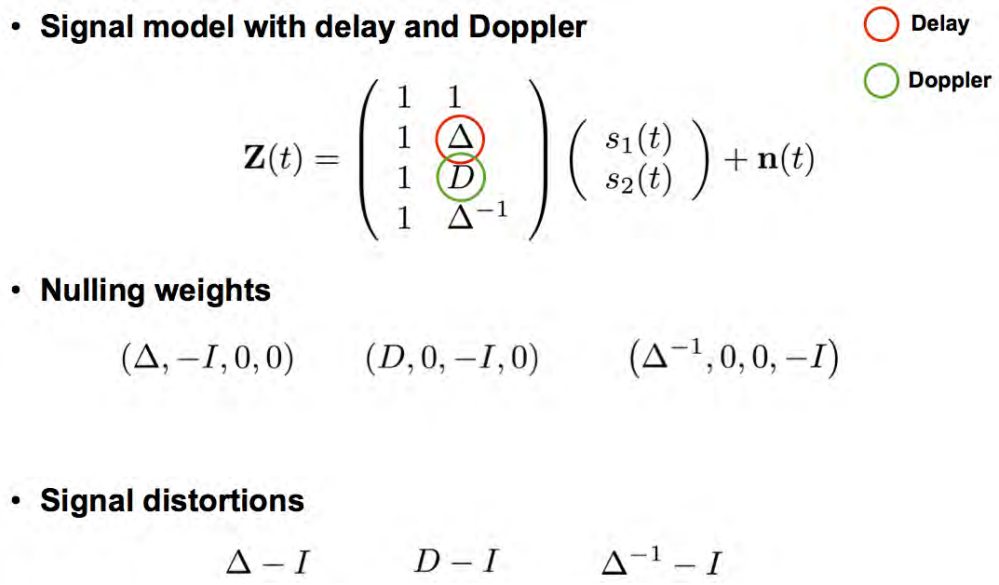


Figure 19. Nulling and equalization is illustrated for a simple delay-Doppler channel matrix. Three nulling weights provide independent beamformers that distort the signal in three different ways. The signal is time and Doppler aligned among the sensor outputs and is represented by the first column of the 4×2 channel matrix. The second column represents an unaligned interferer. Each of the nulling weights shown cancels the interferer, but the signal output from the desired signal is distorted in each case.

nontrivial solution. This solution provides a means of pushing inverses to the left and also a means of clearing denominators by multiplying on the left by a common factor. The common multiples still have fractions in the coefficients that result from solving the homogeneous system. These remaining fractions are easily cleared by multiplying through by the product of their denominators.

More generally, the quasi-polynomial expressions $a_k(x)$ of degree m_k have the common multiple $a'_k(x)a_k(x)$ upon solving $(N-1)\sum_{l=1}^{N}(m_l+1)$ equations in $\sum_{k=1}^{N}\sum_{l\neq k}(m_l+1)$ unknowns. As before, there is one more unknown than equation, providing a nontrivial solution.

4.2.2 Constructing Sparse Common Multiples

Consider the problem of finding the common multiple of 2 noncommutative polynomials $\sum_k a_k \Gamma^{\alpha_k}$ and $\sum_k b_k \Gamma^{\beta_k}$ with $\Gamma^{\alpha} \stackrel{\text{def}}{=} \Delta^{\delta} D^{\omega}$, where $\alpha \stackrel{\text{def}}{=} (\delta, \omega)$ is a 2-tuple of exponents. It is convenient to use polynomial terminology even though the exponents are not nonnegative integers (or even integers). The polynomials represent linear operators, but their algebraic properties, in particular the commutation relations $\Gamma^{\alpha}\Gamma^{\alpha'} = e^{i\omega\delta'}\Gamma^{\alpha+\alpha'}$, allow these operators to be treated as noncommutative polynomials that share some features with normal polynomials, including the existence of common multiples.

Below, it is assumed that the two given polynomials each have n monomial terms. The number of terms n is chosen to be at least 2 for some of the arguments below; otherwise the common multiple problem is trivial. The complex coefficients can be arbitrary, so polynomials

- **Commutation relations and notation**

$$\Delta D = e^{-j\omega\delta} D\Delta \qquad \overline{D} = e^{-j\omega\delta} D \text{ and } \tilde{D} = e^{j\omega\delta} D$$

- **Remove delay distortions**

$$(-(\overline{D} - I), \Delta) \begin{pmatrix} \Delta - I \\ D - I \end{pmatrix} = \overline{D} - I$$

$$\left(-(\tilde{D} - I), \Delta^{-1}\right) \begin{pmatrix} \Delta^{-1} - I \\ D - I \end{pmatrix} = \tilde{D} - I$$

- **Remove Doppler distortions**

$$e^{-j\omega\delta}(\tilde{D} - I) - e^{j\omega\delta}(\overline{D} - I) = 2j \sin(\omega\delta) \cdot I$$

Figure 20. The outputs of the nulling beamformers can be combined, using additional delay and Doppler shifts, to form an equalized copy of the desired signal. Taking two pairs formed from the three nulling beamformers, the delay distortions are removed first. Next, the output of the two pairs is combined to eliminate the Doppler distortion. What remains is a scaled version of the identity operator I acting on the desired signal. In other words, the desired signal is not distorted while the interferer is cancelled.

with different numbers of terms are handled implicitly. Some of the arguments given below assume implicitly that the complex coefficients of both polynomials are nonzero. However, this assumption is required only to establish dimensional parameters for a sytem of equations. These parameters continue to hold when the coefficients vanish. In effect, vanishing coefficients allow more economical solutions than those based on the size n.

Finding the common multiple of $\sum_k a_k \Gamma^{\alpha_k}$ and $\sum_k b_k \Gamma^{\beta_k}$ amounts to finding the solution to

$$\left(\sum_k a'_k \Gamma^{\alpha'_k}\right)\left(\sum_k b_k \Gamma^{\beta_k}\right) = \left(\sum_k b'_k \Gamma^{\beta'_k}\right)\left(\sum_k a_k \Gamma^{\alpha_k}\right). \tag{110}$$

Both the exponents $\{\alpha'_k, \beta'_k\}$ and the complex coefficients $\{a'_k, b'_k\}$ are unknown. A solution is derived by counting the number of complex unknowns $\{a'_k, b'_k\}$ and the number of equations: one for each exponent in the products given by either the LHS or equivalently the RHS of Equ. (110). Since the equations are homogeneous in the complex unknowns, we need at least one more unknown than equation to guarantee a nontrivial solution.

Of course, the exponents must be determined before the complex unknowns can be determined. To do this, elaborate on the solution in the commutative case by writing

$$\sum_k a'_k \Gamma^{\alpha'_k} \sim \left(\sum_k a_k \Gamma^{\alpha_k}\right)^{m+1} \left(\sum_k b_k \Gamma^{\beta_k}\right)^{m} \tag{111}$$

where $\sim$ indicates that the LHS and RHS expressions have common terms (terms with the same 2-tuple exponents). Similarly, we have

$$\sum_k b'_k \Gamma^{\beta'_k} \sim \left(\sum_k a_k \Gamma^{\alpha_k} \right)^m \left(\sum_k b_k \Gamma^{\beta_k} \right)^{m+1} . \tag{112}$$

We need to count the number of terms in these expressions.

Consider the number f_{mn} of terms in $(z^{l_1} + \ldots z^{l_n})^m$. We show inductively that

$$f_{mn} = \frac{(m+n-1)\cdots(m+1)}{(n-1)!} = \binom{m+n-1}{n-1} \tag{113}$$

provided there are no linear relations between the $\{l_k\}$ with integer coefficients. The following special cases are apparent:

$$\begin{aligned} f_{1n} &= n \\ f_{0n} &= 1 \end{aligned}$$

for $n \geq 1$. Also evident is $f_{m1} = 1$ for $m \geq 0$. Define

$$f_{m0} \stackrel{\text{def}}{=} \begin{cases} 0 & m \neq 0 \\ 1 & m = 0 \end{cases} . \tag{114}$$

Assuming the formula is established for f_{kn}, establish it for f_{kn+1} by observing

$$f_{mn+1} = \sum_{k=0}^{m} f_{kn} = f_{0n} + \sum_{k=1}^{m} \frac{(n+k-1)\cdots(k+1)}{(n-1)!}$$

$$= 1 + \sum_{k=1}^{m} \left[\frac{(n+k)\cdots(k+1)}{n!} - \frac{(n+k-1)\cdots k}{n!} \right] = \frac{(m+n)\cdots(m+1)}{n!}$$

proving the induction.

The RHS of Equ. (111) has $f_{m+1\,n} \cdot f_{mn}$ terms (i.e., exponents) given $\{\alpha_k, \beta_k\}$ without linear relations over the integers. Similarly, the RHS of Equ. (112) has $f_{mn} \cdot f_{m+1\,n}$ terms. Thus the total number of unknowns $\{a'_k, b'_k\}$ is $2f_{m+1\,n} \cdot f_{mn}$. The number of terms in the product Equ. (110) is expressed by $f^2_{m+1\,n}$. Each of these terms provides an equation for the complex unknowns. Now $2f_{m+1\,n} \cdot f_{mn} = \frac{2(m+1)}{m+n} f^2_{m+1\,n}$. Choosing $m = n-2$ provides equality between the number $\binom{2n-2}{n-1}^2$ of unknowns and the number of equations.

Generating at least one more equation than unknown is accomplished using a procedure that is best illustrated by an example. Let $a(\Gamma) = a_1\Gamma^{\alpha_1} + a_2\Gamma^{\alpha_2}$ and $b(\Gamma) = b_1\Gamma^{\beta_1} + b_2\Gamma^{\beta_2}$ and assume there are no integer-linear relations among the $\{\alpha_k, \beta_k\}$. Define $a'(\Gamma)$ and $b'(\Gamma)$ so that

$$\begin{aligned} a'(\Gamma) &\sim \left(1 + \Gamma^{\beta_2 - \beta_1}\right) a(\Gamma) \\ b'(\Gamma) &\sim \left(1 + \Gamma^{\beta_2 - \beta_1}\right) b(\Gamma). \end{aligned}$$

Then the relation $a'b = b'a$ involves 7 variables. Normally, there would be 8 variables, but the monomial $\Gamma^{\beta_2-\beta_1}$ causes a degeneracy between two terms, losing a variable. Similarly, there would normally be 8 equations for the unknown constants, but since $\Gamma^{\alpha_k+\beta_1}$ becomes $\Gamma^{\alpha_k+\beta_2}$ when multiplied by the monomial $\Gamma^{\beta_2-\beta_1}$, there are only 6 equations.

The same reasoning applies when $a'b = b'a$ for a' and b' as in Equ. (111) and Equ. (112) above and the number of variables and equations are equal. To see this, consider redefining a' as $(1+\Gamma^\gamma)a'$ and b' as $(1+\Gamma^\gamma)b'$. Depending on the choice of γ, the number of complex variables associated with the monomials in a' and b' is twice the number in the original a' and b' minus degeneracies, which are counted by $\#[\{\alpha'_j = \alpha'_k + \gamma\} \cup \{\beta'_j = \beta'_k + \gamma\}]$. Here, # denotes cardinality. The number of equations for the new a' and b' is similarly twice as large minus the degeneracies counted by either $\#\{\alpha'_j + \beta_k = \alpha'_l + \beta_m + \gamma\}$ or $\#\{\alpha_j + \beta'_k = \alpha_l + \beta'_m + \gamma\}$. Since the equality $\alpha'_j = \alpha'_k + \gamma$ implies $\alpha'_j + \beta_l = \alpha'_k + \beta_l + \gamma$ for all β_l, if we choose γ so that $\#\{\alpha'_j = \alpha'_k + \gamma\} = 1$ and $\#\{\beta'_j = \beta'_k + \gamma\} = 0$, then more equations are lost to degeneracies than variables, as long as $b(\Gamma)$ has at least two terms, as is assumed. It follows that there is at least one more variable than equation, guaranteeing a nontrivial solution to $a'b = b'a$.

When there are linear relations among the $\{\alpha_j\} \cup \{\beta_j\}$ with integer coefficients, some of the terms of a' or b' as defined in Equ. (111) and Equ. (112) may be degenerate. Thus the system of equations $a'b = b'a$ can have a different number of variables and equations. If $\{\alpha'_j\}$ and $\{\beta'_j\}$ represent the exponents of unique terms in a' and b' respectively, let m_j (respectively n_j) represent the corresponding degeneracies (multiplicities of each term). The number of variables lost to degeneracies becomes $\sum_j (m_j - 1) + \sum_j (n_j - 1)$, while the number of equations lost to degeneracy is at least $n \cdot \max(\sum_j (m_j - 1), \sum_j (n_j - 1)) \geq \sum_j (m_j - 1) + \sum_j (n_j - 1)$ provided $n \geq 2$, as is assumed. Thus at least as many equations are lost to degeneracies as variables. What this means is that the system $a'b = b'a$ has at least as many variables as unknowns and thus can be solved nontrivially or extended to a larger system as described above and then solved nontrivially.

Summarizing, two n-term noncommutative delay-Doppler polynomials $a(\Gamma)$ and $b(\Gamma)$ have a common multiple $a'(\Gamma)b(\Gamma) = b'(\Gamma)a(\Gamma)$ in which the polynomials a' and b' have in total at most $2\binom{2n-2}{n-1}^2 - 1$ terms.

4.3 ILLUSTRATIVE EXAMPLE OF DELAY-DOPPLER BEAMFORMING

We present a simple example of zero-forcing for a low-dimensional problem that, nontheless, has a nontrivial solution. There are three sensors and two signals to be nulled. The approach taken uses Gaussian elimination. Fractional delay-Doppler operators are handled by moving denominators to the left and eventually out. The meaning of this will become clear in the example. The machinery for moving fractional factors is the ability to compute common multiples. This can be accomplished in several different ways, some of which are treated in Section 4.2.2. We apply a simple example presented near the end of Section 4.2.2 to the zero-forcing problem treated in this section.

Recall the basic definitions of delay and Doppler operators:

$$\begin{aligned}(D^\omega s)(t) &\stackrel{\text{def}}{=} e^{-i\omega t}s(t)\\ (\Delta^\delta s)(t) &\stackrel{\text{def}}{=} s(t-\delta)\\ \alpha &\stackrel{\text{def}}{=} (\omega,\delta)\\ \Gamma^\alpha &\stackrel{\text{def}}{=} D^\omega\Delta^\delta\\ L_{jk} &\stackrel{\Delta}{=} a_{jk}\Gamma^{\alpha_{jk}}.\end{aligned}$$

The delay and Doppler operators obey the commutation relations

$$\Gamma^\alpha\Gamma^{\alpha'} = e^{-i\omega'\delta}\Gamma^{\alpha+\alpha'}. \tag{115}$$

Nulling weights are constructed by using Gaussian elimination on channel matrices. Since matrix entries are delay-Doppler operators (noncommutative polynomials of a specific kind), order is important in the calculations. An example will illustrate the procedure.

A 3×2 channel matrix can be written in terms of the monomials L_{jk} as

$$\begin{pmatrix} L_{11} & L_{12}\\ L_{21} & L_{22}\\ L_{31} & L_{32}\end{pmatrix} \tag{116}$$

Gaussian elimination results in the expression

$$\begin{pmatrix} 1 & 0 & 0\\ 0 & 1 & 0\\ 0 & -(L_{32}-L_{31}L_{11}^{-1}L_{12})(L_{22}-L_{21}L_{11}^{-1}L_{12})^{-1} & 1\end{pmatrix}\begin{pmatrix} 1 & 0 & 0\\ -L_{21}L_{11}^{-1} & 1 & 0\\ -L_{31}L_{11}^{-1} & 0 & 1\end{pmatrix}\begin{pmatrix} L_{11} & L_{12}\\ L_{21} & L_{22}\\ L_{31} & L_{32}\end{pmatrix}$$
$$= \begin{pmatrix} L_{11} & L_{12}\\ 0 & L_{22}-L_{21}L_{11}^{-1}L_{12}\\ 0 & 0\end{pmatrix}$$

Assume without loss of generality that $L_{k1}=1$ (i.e., $\alpha_{k1}\equiv 0$ and $a_{k1}\equiv 1$). The notation simplifies significantly. Define $a(\Gamma)\stackrel{\text{def}}{=}L_{32}-L_{12}$ and $b(\Gamma)\stackrel{\text{def}}{=}L_{22}-L_{12}$. Then a zero-forcing weight can be expressed as

$$\begin{pmatrix} 0 & 0 & 1\end{pmatrix}\begin{pmatrix} 1 & 0 & 0\\ 0 & 1 & 0\\ 0 & -a(\Gamma)b(\Gamma)^{-1} & 1\end{pmatrix}\begin{pmatrix} 1 & 0 & 0\\ 1 & 1 & 0\\ 1 & 0 & 1\end{pmatrix} = \begin{pmatrix} a(\Gamma)b(\Gamma)^{-1}-1 & -a(\Gamma)b(\Gamma)^{-1} & 1\end{pmatrix}.$$

By solving

$$a'(\Gamma)b(\Gamma) = b'(\Gamma)a(\Gamma), \tag{117}$$

the zero-forcing weight can be expressed alternatively as

$$b'(\Gamma)^{-1}(a'(\Gamma)-b'(\Gamma), -a'(\Gamma), b'(\Gamma)). \tag{118}$$

This form of the weight contains only delay-Doppler operators as entries, with the exception of the overall factor on the left, which is the inverse of a delay-Doppler operator. This factor plays no role in zero-forcing and will be removed. It remains to find a', b'.

The simple example illustrated near the end of Section 4.2 suggests the forms

$$\begin{aligned} a'(\Gamma) &\stackrel{\text{def}}{=} \gamma_{11}\Gamma^{\alpha_{32}} + \gamma_{21}\Gamma^{\alpha_{12}} + \gamma_{31}\Gamma^{2\alpha_{32}-\alpha_{12}} \\ b'(\Gamma) &\stackrel{\text{def}}{=} \gamma_{12}\Gamma^{\alpha_{22}} + \gamma_{22}\Gamma^{\alpha_{12}} + \gamma_{32}\Gamma^{\alpha_{32}-\alpha_{12}+\alpha_{22}} + \gamma_{42}\Gamma^{\alpha_{32}}. \end{aligned}$$

The resulting relations

$$a'(\Gamma)b(\Gamma) = b'(\Gamma)a(\Gamma), \tag{119}$$

involve 6 equations in 7 unknowns and hence have a nontrivial solution.

The number of terms in each entry of the zero-forcing weight is given by $(4, 3, 4)$. Only the number of terms in the first entry is not obvious from the definitions above. In this case, the number of terms is 5 a priori, but the system of equations (6 equations in 7 unknowns) causes one of the terms to vanish.

The exponents involved in the zero-forcing weight can be written as

$$\{\{\alpha_{32}, \alpha_{22}, 2\alpha_{32}-\alpha_{12}, \alpha_{32}-\alpha_{12}+\alpha_{22}\}, \{\alpha_{32}, \alpha_{12}, 2\alpha_{32}-\alpha_{12}\}, \{\alpha_{22}, \alpha_{12}, \alpha_{32}-\alpha_{12}+\alpha_{22}, \alpha_{32}\}\}. \tag{120}$$

For the general case, let $\alpha_{k2} \leftarrow \alpha_{k2} - \alpha_{k1}$ and $a_{k2} \leftarrow a_{k2}/a_{k1}$. Only the exponents matter since the coefficients, in practice, are estimated from the data.

4.4 DELAY-DOPPLER BEAMFORMING IN THE GENERAL CASE

A general approach toward solving the zero-forcing problem with a form of Gaussian elimination is sketched below. This approach can be used to find an interesting set of zero-forcing weights. The main technical ingredient is the construction of common multiples, as described in Section 4.2.2. When delays and Dopplers each lie on one-dimensional lattices, the matrix entries become Ore polynomials, for which other algorithms can be useful in finding common multiples (both compressive sensing Section 6 and algebraic approaches Section 5.3.3 can be used for common multiples).

A general procedure for performing Gaussian elimination and clearing denominators is provided by the following recursive updates on a matrix $\mathbf{L}^{(0)} = \mathbf{L}$ of size $m \times m-1$. Let

$$\mathbf{F}_k \stackrel{\text{def}}{=} \begin{pmatrix} \mathbf{I}_{k-1} & 0 & 0 & \dots & 0 \\ 0 & 1 & 0 & \dots & 0 \\ 0 & -L^{(k)}_{k+1k}(L^{(k)}_{kk})^{-1} & 1 & \dots & 0 \\ & \vdots & & & \\ 0 & -L^{(k)}_{mk}(L^{(k)}_{kk})^{-1} & 0 & \dots & 1 \end{pmatrix} \tag{121}$$

and

$$\mathbf{D}_k \stackrel{\text{def}}{=} \begin{pmatrix} \mathbf{I}_k & 0 \\ 0 & \operatorname{diag}\{L'^{(k)}_{kk}(r)\}_{r=k+1}^{m} \end{pmatrix} \tag{122}$$

where $L'^{(k)}_{rk} L^{(k)}_{kk} = L'^{(k)}_{kk}(r) L^{(k)}_{rk}$ provides a common multiple for $L^{(k)}_{kk}$ and $L^{(k)}_{rk}$. This relation implicitly defines $L'^{(k)}_{kk}(r)$, which should not be confused with a component of a matrix. It is simply a delay-Doppler polynomial used to solve the common multiple problem. Then

$$\mathbf{D}_k \mathbf{F}_k = \begin{pmatrix} \mathbf{I}_{k-1} & 0 & 0 & \dots & 0 \\ 0 & 1 & 0 & \dots & 0 \\ 0 & -L'^{(k)}_{k+1k} & L'^{(k)}_{kk}(k+1) & \dots & 0 \\ & \vdots & & & \\ 0 & -L'^{(k)}_{mk} & 0 & \dots & L'^{(k)}_{kk})(m) \end{pmatrix}. \tag{123}$$

Define $\mathbf{L}^{(k+1)} \stackrel{\text{def}}{=} \mathbf{D}_k \mathbf{F}_k \mathbf{L}^{(k)}$. As long as all diagonal entries are nonzero, this procedure triangularizes $\mathbf{L}$, culminating in the upper triangular matrix $\mathbf{L}^{(m-1)} = (\mathbf{D}_{m-1}\mathbf{F}_{m-1}) \cdots (\mathbf{D}_1 \mathbf{F}_1)\mathbf{L}$, which has a vanishing last row. The left kernel of $\mathbf{L}^{(m-1)}$ is spanned by $\mathbf{w} \stackrel{\text{def}}{=} (0, \dots, 0, 1)^T$. Thus

$$0 = \mathbf{w}^\dagger \mathbf{L}^{(m)} = [(\mathbf{D}_1 \mathbf{F}_1)^\dagger \cdots (\mathbf{D}_{m-1}\mathbf{F}_{m-1})^\dagger \mathbf{w}]^\dagger \mathbf{L} \tag{124}$$

where $\cdot^\dagger$ denotes a hermitian transpose. Since the matrix entries are operators, one must define $\cdot^\dagger$ on them. But $\Delta^{\delta\dagger} = \Delta^{-\delta}$ and $D^{\omega\dagger} = D^{-\omega}$. A nulling weight is thus provided by $(\mathbf{D}_1 \mathbf{F}_1)^\dagger \cdots (\mathbf{D}_{m-1}\mathbf{F}_{m-1})^\dagger \mathbf{w}$.

When we run into zero diagonal entries $L^{(k)}_{kk}$ along the way, we can use pivoting to put nonzero entries, if any exist, into the diagonal position. The algorithm is modified accordingly.

5. ALGEBRAIC ZERO-FORCING WEIGHTS

The operators that apply delay and Doppler shifts to signals can be written or approximated by polynomials in two variables. One variable represents the smallest delay shift and the other the smallest Doppler shift. The algebra formed by these polynomials has a structure that can be used to aid the formation of zero-forcing (also called nulling) weights.

In the delay-only case, which involves polynomials in the smallest delay shift and does not involve Doppler shifts in the signal processing, the polynomial algebra is familiar, but the construction of zero-forcing weights using this algebra is not as well known. We treat this special case in Section 5.1. This serves as an introduction to the more general problem of finding zero-forcing weights with delay-Doppler polynomials, which is treated in Section 5.3.

The methods utilized to handle the delay-only case are much simpler than the techniques used for delay and Doppler. The delay-only case is also easier to handle from a computational viewpoint. In both cases, computer algebra plays a significant role in the solutions for zero-forcing.

5.1 DELAY-ONLY ZERO-FORCING

The next result relies on the fact that systems of linear equations can be solved over the field of rational functions over $\mathbb{C}$ in the same manner as they could be solved over just $\mathbb{C}$. The concepts of dimension, basis, and rank hold in analogous fashion. Thus, the array processing solutions we seek can be formulated abstractly in linear algebra over the rational functions. However, we are ultimately interested in solutions with vectors and coefficient matrices having only polynomial entries, so that solutions with rational function entries are only a stepping stone. Our first result demonstrates that the concept of a basis for the null space for a channel matrix does extend to vectors and matrices with polynomial entries in certain cases.

Theorem 10. *Let*

$$N_H \stackrel{\text{def}}{=} \{\mathbf{w}(z) | \mathbf{w}(z) \cdot \mathbf{H} = 0\} \tag{125}$$

denote the linear subspace of $(\mathbb{C}[z])^n$*, vectors with polynomial entries, that corresponds to operators that null the channel matrix* $\mathbf{H} \in (\mathbb{C}[z])^{n \times k}$*. If* $\mathbf{H}^{(k)}$*, denoting the bottom* $k \times k$ *minor of* $\mathbf{H}$*, is full rank over the rational functions, then there exists a basis of* N_H *consisting of the columns of*

$$(c_1(z), \ldots, c_{n-k}(z)) = \begin{pmatrix} c_{11} & 0 & \ldots & 0 \\ c_{21} & c_{22} & \ldots & 0 \\ & & \ddots & 0 \\ c_{n-k\,1} & c_{n-k\,2} & \ldots & c_{n-k\,n-k} \\ & & \ddots & \vdots \\ c_{n1} & c_{n2} & \ldots & c_{n\,n-k} \end{pmatrix} \tag{126}$$

with $c_{jl}(z) \in \mathbb{C}[z]$*. The degree of* $c_{kk}(z)$ *is the smallest among all nulling vectors* $\mathbf{c}_k(z)$ *with zeros in the first* $k-1$ *components. Nonsingularity of the* $k \times k$ *minor* $\mathbf{H}^{(k)}$ *is a generic property of* $\mathbf{H}$*.*

Proof. Among all zero-forcing weights $w(z)$, find one with the smallest degree nonzero polynomial in the first component. An algorithm for accomplishing this will be presented later. Next, among all zero-forcing weights $w(z)$ with zero first component, find a weight with smallest degree in the second component. Proceed in this manner to create a matrix of the form Equ. (126). We claim that all zero-forcing weights can be written in the form $\sum_k p_k(z)c_k(z)$ with polynomial coefficients $p_k(z)$. To see this, note that any zero-forcing weight $w(z)$ with a nonzero first component must have a first component divisible by $c_{11}(z)$, otherwise, we could take a linear (with polynomial coefficients) combination of $c_1(z)$ and $w(z)$ that would have a nonzero, smaller degree first component. Since the linear combination still nulls, this contradicts our choice of $c_1(z)$. It follows that $c_{11}(z)$ divides the first component of $w(z)$, and hence a linear combination of $c_1(z)$ and $w(z)$ has vanishing first component. We can proceed inductively until we have a linear combination of the $c_k(z)$ and $w(z)$ for which the first $n-k$ components vanish. This results in a zero-forcing weight with at most k nonzero bottom components, which is impossible since it is assumed that $\mathbf{H}^{(k)}$ is nonsingular (rank k over the field of rational functions), unless the linear combination vanishes. Over the rational function field (i.e., with the components of $w(z)$ consisting of rational expressions), the zero-forcing weights span a $n-k$ dimensional vector space. By multiplying through by a polynomial $p(z)$ that clears denominators, we see that $p(z)w(z)$ is in the span of the $n-k$ columns of Equ. (126). Thus $w(z)$ is in the rational span of the same columns. In other words, the $n-k$ columns of Equ. (126) span the $n-k$ dimensional vector space of zero-forcing weights with rational entries. Thus these columns must be independent over the rational functions and hence also over polynomial functions. It follows that the columns of Equ. (126) are a basis as claimed. The fact that $\mathbf{H}^{(k)}$ is generically nonsingular is a consequence of the nonvanishing determinant of this minor. □

This result provides us with the existence and structure of a basis of nulling weights, but does not provide a constructive solution. In the next sections, we show how the use of GCD and LCM algorithms can provide a construction for a nulling basis. The steps proceed first by constructing a basis for nulling a single given vector of polynomials. In other words, $\mathbf{H} \in (\mathbb{C}(z))^{n\times 1}$. Next, we modify this basis to null additional vectors, one at a time.

5.1.1 GCD and Euclidean Algorithm

The first basis we derive algorithmically nulls a single vector: $\mathbf{H} = \mathbf{h}$. To express the result, we need the concept of the greatest common divisor (GCD) among the polynomials $\{a_k(z)\}_1^n$. We write

$$\gcd(a_1(z),\ldots,a_n(z)) = \left(\sum_k \hat{b}_k(z)a_k(z)\right) \text{ where } \deg \sum_k \hat{b}_k(z)a_k(z) = \min_{\{b_k(z)\}} \deg \sum_k b_k(z)a_k(z). \tag{127}$$

The GCD is determined only up to a complex scalar. It can be computed inductively by using

$$\gcd(a_1(z),\ldots,a_n(z)) = \gcd(a_1(z), \gcd(a_2(z),\ldots,a_n(z))). \tag{128}$$

At the final stage, the GCD of two polynomials is computed using the Euclidean algorithm for polynomials. Assume without loss of generality that $\deg a_2(z) \le \deg a_1(z)$.

Algorithm 1 Euclidean Algorithm

Set $k = m$, $r_k(z) = a_1(z)$, $r_{k-1}(z) = a_2(z)$, and $r_{k-2}(z) = 1$
while $r_{k-2}(z) \neq 0$ **do**
Divide to get quotient and remainder: $r_k(z) = q_{k-1}(z)r_{k-1}(z) + r_{k-2}(z)$ with $\deg r_{k-2}(z) < \deg r_{k-1}(z)$
$k \leftarrow k - 1$
end while
gcd $= (r_k(z))$

The value of m is not important since it only sets an origin for the indexing. We will choose it later for notational convenience. Note that the iterative step in the algorithm can be written

$$\begin{pmatrix} r_k \\ r_{k-1} \end{pmatrix} = \begin{pmatrix} q_{k-1} & 1 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} r_{k-1} \\ r_{k-2} \end{pmatrix}. \tag{129}$$

Taking the inverse transpose of the 2×2 matrix in the iteration, we can define an iteration

$$(\rho_k \ \sigma_{k-1}) = (\rho_{k-1} \ \sigma_{k-2}) \begin{pmatrix} 0 & 1 \\ 1 & -q_{k-1} \end{pmatrix} \tag{130}$$

that preserves the value of

$$\begin{pmatrix} \rho_k \\ \sigma_{k-1} \end{pmatrix}^T \begin{pmatrix} r_k \\ r_{k-1} \end{pmatrix}. \tag{131}$$

When k is chosen so that $r_{k-1}(z) \equiv 0$ with $r_k(z) = \gcd(a_1, a_2)$, then defining $\rho_k = 1$ and $\sigma_{k-1} = 0$ tells us that Equ. (131) is the GCD for this value of k and larger values. Thus $\gcd(a_1(z), a_2(z)) = \rho_m(z)a_1(z) + \sigma_{m-1}(z)a_2(z)$.

The GCD, as its name suggests, is the greatest common divisor of its arguments. For the two argument case, using the notation and results above, it is clear that any common divisor of r_m and r_{m-1} is a divisor of the GCD r_k. Conversely, using Equ. (129), any divisor of r_{j-1} and r_{j-2} is also a divisor of r_j. Since $(r_k \ r_{k-1}) = (r_k \ 0)$, it follows that the GCD $r_k(z)$ divides $r_m(z)$ and $r_{m-1}(z)$.

5.1.2 Least Common Multiples

It convenient to write the Euclidean algorithm in the form of Alg. 2. Assume that $r_{n+1} = 0$ and that $r_n \neq 0$ represents the GCD. Let $\theta_k \stackrel{\text{def}}{=} r_k(z)r_n^{-1}(z)$ denote the polynomial with the GCD divided out. Then the recurrence for the r_k determines a recurrence $\theta_k = q_{k+1}\theta_{k+1} + \theta_{k+2}$ for the θ_k, ending with $\theta_k = 0$ for $k \geq n$. Note that $\theta_{n-2} = q_{n-1}\theta_{n-1}$ so that, evidently, θ_{n-2} is the least common multiple (LCM) of θ_{n-2} and θ_{n-1}. Define, $\alpha_{n-2} = 1$ and $\beta_{n-2} = q_{n-1}$ so that $\alpha_{n-2}\theta_{n-2} = \beta_{n-2}\theta_{n-1}$. We would like to maintain the relationship $\alpha_l\theta_l = \beta_l\theta_{l+1}$ for $1 \leq l \leq n-2$. We do this inductively. Assume it holds for $l \geq k+1$. Since we have $\theta_k = q_{k+1}\theta_{k+1} + \theta_{k+2}$, multiply on the left by β_{k+1} to get

$$\beta_{k+1}\theta_k = \beta_{k+1}q_{k+1}\theta_{k+1} + \beta_{k+1}\theta_{k+2} = \beta_{k+1}q_{k+1}\theta_{k+1} + \alpha_{k+1}\theta_{k+1} = (\beta_{k+1}q_{k+1} + \alpha_{k+1})\theta_{k+1}. \tag{132}$$

Thus, we can define $\alpha_k \stackrel{\text{def}}{=} \beta_{k+1}$ and $\beta_k = \beta_{k+1}q_{k+1} + \alpha_{k+1}$ so that $\alpha_k\theta_k = \beta_k\theta_{k+1}$, as desired.

We say that $\gamma \mid (\alpha, \beta)$, read as γ divides α and β, iff $\alpha = \gamma\alpha'$ and $\beta = \gamma\beta'$. We have, from the induction, $\gamma \mid (\alpha_k, \beta_k)$ iff $\gamma \mid (\alpha_{k+1}, \beta_{k+1})$. We claim that $\alpha_1\theta_1 = \beta_1\theta_2$ is the least common multiple of $\theta_1$and θ_2. Suppose otherwise. Then there is a nonscalar multiple of μ such that $\beta_1\theta_2 = \gamma\mu = \alpha_1\theta_1$. Using $\gamma_1\theta_1 = \mu = \gamma_2\theta_2$, we see that $\gamma \mid (\alpha_1, \beta_1)$. But then we would have $\gamma \mid (\alpha_{n-2}, \beta_{n-2})$, which is impossible.

The least common multiple of r_1 and r_2 follows from the previous result and assumes the form $\alpha_1\theta_1 r_n = \beta_1\theta_2 r_n$. To see this, assume instead that $\alpha_1\theta_1 r_n = \gamma\mu = \beta_1\theta_2 r_n$, where μ is the least common multiple and γ is not a scalar. Then $\alpha_1\theta_1 = \gamma\mu r_n^{-1} = \beta_1\theta_2$ and μr_n^{-1} is integral not just rational, since μ is a left multiple of r_1 (or r_2), which has r_n as a right factor. This contradicts the fact that $\alpha_1\theta_1 = \beta_1\theta_2$ is the LCM of θ_1, θ_2.

Algorithm 2 Euclidean Algorithm

Set $k = 1$. Given $r_1(z)$, $r_2(z)$.
while $r_{k+1}(z) \neq 0$ **do**
Divide to get quotient and remainder: $r_k(z) = q_{k+1}(z)r_{k+1}(z) + r_{k+2}(z)$ with $\deg r_{k+2}(z) < \deg r_{k+1}(z)$
$k \leftarrow k + 1$
end while
gcd $= (r_k(z))$

Algorithm 3 Least Common Multiple

Initialize $n = 1$. Given $r_1(z)$ and $r_2(z)$.
while $r_{n+1}(z) \neq 0$ **do**
Divide to get quotient and remainder: $r_n(z) = q_{n+1}(z)r_{n+1}(z) + r_{n+2}(z)$ with $\deg r_{n+2}(z) < \deg r_{n+1}(z)$
$n \leftarrow n + 1$
end while
Initialize $\alpha_{n-1} = 1$ and $\beta_{n-1} = q_n$.
for $k = n - 2$ to $k = 1$ **do**
$\alpha_k = \beta_{k+1}$
$\beta_k = \beta_{k+1}q_{k+1} + \alpha_{k+1}$
end for
lcm $= \alpha_1 r_1 = \beta_1 r_2$

5.1.3 Nulling A Single Source

Let $\mathbf{h}(z) \in \mathbb{C}^n(z)$ be a fixed vector with polynomial entries and let $\mathbf{c}(z) \in \mathbb{C}^n(z)$ be a vector that nulls $\mathbf{h}$: $\mathbf{c}(z) \cdot \mathbf{h}(z) = 0$. Among all such nulling weights, we seek one for which deg $c_1(z)$

is as small as possible. Now, nulling is equivalent to $c_1(z)h_1(z) = -\sum_{k>1} c_k(z)h_k(z)$ which tells us that $c_1(z)h_1(z)$ equals a lowest degree expression of the form $\sum_{k>1} a_k(z)h_k(z)$ with arbitrary $a_k(z)$. An expression of the smallest degree is provided by $g(z) \stackrel{\text{def}}{=} \gcd(h_2(z), \ldots, h_n(z))$. Thus, $c_1(z)h_1(z)$ is a least common multiple of $g(z)$ and $h_1(z)$, denoted $\text{lcm}(g(z), h_1(z))$ and thus $c_1(z) = -\text{lcm}(g(z), h_1(z))/h_1(z)$ has the smallest possible first-entry degree among all nulling weights $\mathbf{c}(z)$. The remainder of the components of $\mathbf{c}(z)$ come from the GCD calculation. In the two arguments example of the GCD shown in Alg. 1, the calculation follows from Equ. (130). Following the argument given in Thm. 10, we can complete the basis of nulling weights by computing more GCDs and LCMs.

5.1.4 Nulling Multiple Sources

Let $\mathbf{H}^{(m)}(z)$ denote the $n \times m$ submatrix of $\mathbf{H}(z) \in (\mathbb{C})^{n\times k}$ from Thm. 10 consisting the the leftmost m columns. Assume that we have the matrix $\mathbf{C}^{(m)}(z) \in (\mathbb{C}(z))^{n\times n-m}$, each column of which nulls all the columns of $\mathbf{H}^{(m)}(z) \in (\mathbb{C}(z))^{n\times m}$. When $m = 1$, $\mathbf{C}^{(m)}(z)$ can be constructed by the method of Section 5.1.3. We assume that these $\mathbf{C}^{(m)}(z)$ have the properties of minimality for leading components that is described in Thm. 10. We will modify the columns of $\mathbf{C}^{(m)}(z)$ one at a time, from left to right up until the second to last column. The last column will be dropped. The final result with be a matrix $\mathbf{C}^{(m+1)}(z)$, each of whose columns null all of the columns of $\mathbf{H}^{(m+1)}(z)$.

First, note that $(\mathbf{C}^{(m)}(z))^T\mathbf{H}^{(m)}(z) = 0$ by definition. Let $\mathbf{c}_k^{(m)}(z)$ denote the k^{th} column of $\mathbf{C}^{(m)}(z)$ and let $c_{kk}^{(m)}(z)$ denote its first potentially nonzero component. Write $h_k(z) = \mathbf{c}_k^{(m)}(z) \cdot \mathbf{H}_{\cdot\, m+1}(z)$ for $1 \le k \le m$. Every nulling weight can be written as $\mathbf{w}(z) = \sum_{k=1}^{n-m} a_k(z)\mathbf{c}_k^{(m)}(z)$ by the assumed form of $\mathbf{C}^{(m)}(z)$. But $\mathbf{w}(z) \cdot \mathbf{H}_{\cdot\, m+1}(z) = a_1(z)h_1(z) + \sum_{1<k\le n-m} a_k(z)h_k(z)$. As in the argument of Section 5.1.3, we can write $a_1(z) = -\text{lcm}(h_1(z), \gcd(h_2(z), \ldots, h_{n-m}(z)))/h_1(z)$. Thus we have constructed a polynomial weight $\mathbf{c}_1^{(m+1)}(z) \stackrel{\text{def}}{=} \sum_{1\le k\le n-m} a_k(z)\mathbf{c}_k(z)$ that nulls all the columns of $\mathbf{H}^{(m+1)}(z)$ and has the smallest possible degree in the leading entry $c_{11}^{(m+1)}(z) = a_1(z)c_{11}^{(m)}(z)$. Proceeding through all the columns of $\mathbf{C}^{(m)}(z)$ except the last, we construct $\mathbf{C}^{(m+1)}(z) \in (\mathbb{C}(z))^{n\times n-m-1}$, all of whose columns null all the columns of $\mathbf{H}^{(m+1)}(z)$, and which satisfies the structure of Thm. 10. Proceeding inductively, we finally arrive at a nulling basis expressed by the columns of $\mathbf{C}^{(k)}(z) \in (\mathbb{C}(z))^{n\times k}$.

5.1.5 Reducing Weight Degrees

The zero-forcing weight $\mathbf{c}_1(z)$ can have the degrees of its top $n-k$ entries reduced by combining it with linear combinations of the remaining basis vectors in the null space. Conceptually, the leading nonzero components of these remaining vectors are used in succession to reduce the degrees of the corresponding components in $\mathbf{c}_1(z)$, replacing entries with remainders $r_k(z)$. This makes sense since the leading nonzero entries $c_{kk}(z)$ of the basis weights have low degree, and hence, so do the remainders $r_k(z)$.

This procedure typically results in a nulling weight with small polynomial degrees in its leading $n - k$ components but comparatively large degrees in the remaining k components. There is nothing optimal about this procedure. Reducing total degree across all components may seem

Algorithm 4 Degree Reduction for Zero-Forcing Weights

$\mathbf{c}^{(1)}(z) \stackrel{\text{def}}{=} \mathbf{c}_1(z)$
while $2 \leq k \leq n - j$ **do**
  Solve for $q_k(z)$ and $r_k(z)$: $c_k^{(k-1)}(z) = q_k(z)c_{kk}(z) + r_k(z)$
  $\mathbf{c}^{(k)}(z) \leftarrow \mathbf{c}^{(k-1)}(z) - q_k(z)\mathbf{c}_k(z)$
end while

a desirable goal, but this goal is often accomplished by using the smallest number of components $k+1$ that can theoretically accomodate nulling. A large number of zero components means that a significant fraction of the power in a desired (not nulled) signal is thrown away. Since nulling (i.e., zero-forcing) is not the full story of adaptive beamforming, degree minimization, by itself, is not a design criterion. Reduced degree weights offer the possibility of degree reduction while maintaining some control of gain on a desired signal.

5.1.6 Equalization

Once a nulling weight basis of the form $\mathbf{C}^{(m)}(z) = [\mathbf{c}_1(z), \ldots, \mathbf{c}_m(z)]$ has been formed, it is possible to select a vector of the form $\mathbf{w}(z) = \sum_k a_k(a)\mathbf{c}_k(z)$ such that $\mathbf{w}(z) \cdot \mathbf{v}(z)$ has the smallest degree possible given an arbitrary $\mathbf{v}(z)$ with polynomial entries. We find $\mathbf{w}(z) \cdot \mathbf{v}(z) = \gcd(\mathbf{v}(z) \cdot \mathbf{c}_1(z), \ldots, \mathbf{v}(z) \cdot \mathbf{c}_m(z))$. The $\{a_k(z)\}$ that solve $\sum_k a_k(z)(\mathbf{v}(z) \cdot \mathbf{c}_k(z)) = \gcd(\mathbf{v}(z) \cdot \mathbf{c}_1(z), \ldots, \mathbf{v}(z) \cdot \mathbf{c}_m(z))$ provide the nulling weight $\mathbf{w}(z)$ with the best equalization performance. In practice, it is often the case that $\mathbf{w}(z) \cdot \mathbf{v}(z) = 1$, resulting in perfect nulling and perfect equalization. Note that the solution to the equalization problem requires $n - m \geq 2$. Thus, the total number of signal sources that can be handled, nulling all but one signal and equalizing the remaining signal, is $n - 1$. This is one less signal than what could be handled when all response vectors have scalar, nonpolynomial entries. In this latter case, equalization involves a simple scaling of a nulling weight since $\mathbf{w} \cdot \mathbf{v}$ is always a scalar.

5.2 EXTENSION OF RESULTS TO ORE POLYNOMIALS

Let $\mathbb{C}^{(Ore)}[x, y]$ denote the ring of Ore polynomials $p(x, y) = \sum_{jk} c_{jk}x^j y^k$ with complex coefficients that satisfy the monomial multiplication rule

$$(x^j y^k)(x^l y^m) = \eta^{kl} x^{j+l} y^{k+m}. \tag{133}$$

This rule extends in a natural manner to the whole algebra $\mathbb{C}^{(Ore)}[x, y]$ of noncommutative polynomials. Recall that x and y can represent, for example, delay and Doppler operators, which obey the rule of Equ. (133). For the present, it suffices to treat the algebra abstractly, with η a complex constant.

In order to apply some of the previous techniques to the Ore ring, we need to form inverses of the polynomials in one of the variables. Specifically, we need to consider expressions of the form

$$r(x, y) = \sum_k \frac{p_k(x)}{q_k(x)} y^k \in \mathbb{C}^{(Ore)}(x)[y], \tag{134}$$

where $\mathbb{C}^{(Ore)}(x)[y]$ is implicitly defined by Equ. (134). From Equ. (133), it can be seen that $(p(x)/q(x))y^j = y^j(p(\eta^j x)/q(\eta^j x))$. Thus, the polynomials in y have coefficients from the field $\mathbb{C}(x)$ of rational functions in x. Aside from the twist in multipilcation implied by Equ. (133), these are very much like the polynomials over $\mathbb{C}$ treated in Section 5. In particular, the methods of Section 5 can be used to find a basis of zero-forcing weights which have entries that are polynomials in y with coefficients rational functions in x.

Care has been taken in the algorithms of Section 5 to handle the order of factors. The chief impacts of find zero-forcing weights with entries in $\mathbb{C}^{(Ore)}(x)[y]$ instead of $\mathbb{C}[y]$ are:

1. the division step $r_{i-1} = q_i r_i + r_{i+1}$ works with polynomials in the normal form $\sum_k \frac{p_k(x)}{q_k(x)} y^k$, with ordered factors, and
2. Equ. (133) is employed to put $q_i r_i$ and r_{i+1} into normal form .

Although we don't pursue this approach any further here, it is an entirely reasonable method for finding solutions to zero-forcing problems that are sparse in one of the delay or Doppler operators. Once solutions have been found, the rational coefficients in the remaining variable can have their denominators cleared by multiplying through by a polynomial in that remaining variable. Thus, control is exercised over only one of the variables in the problem. We next consider a more principled approach toward handling both delay and Doppler.

5.3 ZERO-FORCING USING GROEBNER BASES

5.3.1 Ideal Bases

There are alternative approaches to finding generating sets for zero-forcing weights that operate entirely within the Ore algebra $\mathbb{C}^{(Ore)}[z_1, z_2]$ introduced in Section 5.2. To understand the concepts behind these techniques and to express clearly the relevant algorithms, we need some concise notation. For monomials, we write $z_1^{\alpha_1} z_2^{\alpha_2} = z^{\boldsymbol{\alpha}}$, with $\boldsymbol{\alpha} \stackrel{\text{def}}{=} (\alpha_1, \alpha_2)$ and $\alpha_k \geq 0$. We write any $f \in \mathbb{C}^{(Ore)}[z_1, z_2]$ as $f = \sum_{\boldsymbol{\alpha}} c_{\boldsymbol{\alpha}} z^{\boldsymbol{\alpha}}$. From Equ. (133), we have $z^{\boldsymbol{\alpha}} z^{\boldsymbol{\beta}} = \eta^{\alpha_2 \beta_1} z^{\boldsymbol{\beta}} z^{\boldsymbol{\alpha}}$.

There is an obvious partial ordering defined on the ordered pairs $\boldsymbol{\alpha} = (\alpha_1, \alpha_2)$ defined by $\boldsymbol{\alpha} \leq^2 \boldsymbol{\beta}$ iff $\alpha_k \leq \beta_k$. Of course, not all pairs $\boldsymbol{\alpha}, \boldsymbol{\beta}$ can be ordered in this manner. However, in order to write $z^{\boldsymbol{\alpha}} z^{\boldsymbol{\beta}} = \text{const } z^{\boldsymbol{\gamma}}$ with $\boldsymbol{\gamma} \stackrel{\text{def}}{=} \boldsymbol{\alpha} + \boldsymbol{\beta}$, if is necessary and sufficient to have $\boldsymbol{\alpha} \leq^2 \boldsymbol{\gamma}$. In this case, we say $z^{\boldsymbol{\alpha}} \mid z^{\boldsymbol{\gamma}}$; in other words, the monomial $z^{\boldsymbol{\alpha}}$ divides the monomial $z^{\boldsymbol{\gamma}}$. Thus, the smallest (in the sense of $\leq^2$) $\boldsymbol{\gamma}$ for which $z^{\boldsymbol{\gamma}}$ is divided by both $z^{\boldsymbol{\alpha}}$ and $z^{\boldsymbol{\beta}}$ is $\boldsymbol{\gamma} \stackrel{\text{def}}{=} \boldsymbol{\alpha} \vee \boldsymbol{\beta} = (\max(\alpha_1, \beta_1), \max(\alpha_2, \beta_2))$.

We need to consider a total (all monomial pairs can be compared) ordering $\preceq$ on the exponents $\boldsymbol{\alpha}$ of $z^{\boldsymbol{\alpha}}$ that is compatible with products in the sense that $\boldsymbol{\alpha} \preceq \boldsymbol{\beta}$ if and only if $\boldsymbol{\alpha} + \boldsymbol{\gamma} \preceq \boldsymbol{\beta} + \boldsymbol{\gamma}$. Furthermore, we require $(0, 0) \preceq \boldsymbol{\alpha}$ for all exponents $\boldsymbol{\alpha}$. One example of such an order is called lexicographic (*lex*) order, where $\boldsymbol{\alpha} \preceq \boldsymbol{\beta}$ iff either $\alpha_1 < \beta_1$ or both $\alpha_1 = \beta_1$ and $\alpha_2 \leq \beta_2$. Another order is called *deglex*. In this case, $\boldsymbol{\alpha} \preceq \boldsymbol{\beta}$ iff $\sum \alpha_k < \sum \beta_k$ or both $\sum \alpha_k = \sum \beta_k$ and $\alpha_1 \leq \beta_1$. The algorithms below only require a compatible total order, but produce different results with different orders. The two orders just mentioned are of practical interest.

It's worth mentioning that the ordering terminology we use differs from [BGTV03] but seems more consistent with the literature. Furthermore, since we use only two indeterminates z_1, z_2 for all Groebner basis calculations, we do not run into issues of improved algorithm efficiency with different orderings.

It's worth keeping in mind that the two orders $\leq^2$and $\preceq$ are very different and are both used extensively. In a very rough sense, polynomial order and basis selection is based on $\preceq$ while divisibility is based on $\leq^2$.

Each element of $\mathbb{C}^{(Ore)}[z_1, z_2]$ can be written as $f = \sum_{\boldsymbol{\alpha}} c_{\boldsymbol{\alpha}} z^{\boldsymbol{\alpha}}$. Under a total, compatible monomial order, there is a largest term $c_{\boldsymbol{\alpha}_{\max}} z^{\boldsymbol{\alpha}_{\max}}$. We define $\deg(f) = (\deg_1(f), \deg_2(f)) = \boldsymbol{\alpha}_{\max}$ and $\mathrm{lc}(f) = c_{\boldsymbol{\alpha}_{\max}}$. Thus, we can write $f = \mathrm{lc}(f) z^{\deg(f)} + L.O.T.$, that is f is the sum of its top order monomial plus lower order monomial terms.

One of the central concepts we need is that of an ideal in the algebra $\mathbb{C}^{(Ore)}[z_1, z_2]$. A left ideal $I \subseteq \mathbb{C}^{(Ore)}[z_1, z_2]$ is a subset that is closed under addition and under multiplication by any element of $\mathbb{C}^{(Ore)}[z_1, z_2]$. Specifically, if $a(z), b(z) \in I$ and $c(z) \in \mathbb{C}^{(Ore)}[z_1, z_2]$, then $a(z) + b(z) \in I$ and $c(z)a(z) \in I$ (and, of course, $c(z)b(z) \in I$). We say that $\{f_i\} \in \mathbb{C}^{(Ore)}[z_1, z_2]$ generate the ideal $\mathrm{Ideal}(\{f_i\})$ iff each $f \in \mathrm{Ideal}(\{f_i\})$ can be writtten as $\sum_i a_i(z) f_i(z)$ for $a_i \in \mathbb{C}^{(Ore)}[z_1, z_2]$.

Next, we consider a division algorithm Alg. 5 that, given a finite set of divisors $f_i \in \mathbb{C}^{(Ore)}[z_1, z_2]$, reduces the degree (in the total, compatible order) of f by dividing successively by the f_i until no further division is possible. This process of reduction produces a remainder r. The ideal generated by $r \cup \{f_i\}$ is the same as the ideal generated by $f \cup \{f_i\}$, but the degree of one of the the generators has been reduced. In the division algorithm, let $\mathrm{lt}(f)$ denote the monomial (including scale factor) representing the highest order term of f.

Algorithm 5 Division Algorithm

Goal: Given the polynomial f and the finite set of generator polynomials $\{f_1, \ldots, f_s\}$ in $\mathbb{C}^{(Ore)}[z_1, z_2]$, find a remainder $r \in \mathbb{C}^{(Ore)}[z_1, z_2]$ of f and coefficient polynomials p_i such that $f - \sum_i p_i f_i = r$ and $\deg(r) \preceq \deg(f)$.
Initialize: $h \coloneqq f$, $p_i \coloneqq 0$, and $r \coloneqq 0$
while $h \neq 0$ **do**
  if some $\deg(f_i) \leq^2 \deg(h)$ **then**
    $j = \mathrm{argmin}_i \deg(f_i) \leq^2 \deg(h)$
    $c = \frac{\mathrm{lc}(h)}{\mathrm{lc}(f_i)} \eta^{-(\deg_2(h) - \deg_2(f_j))\deg_1(f_j)}$
    $h \leftarrow h - c z^{\deg(h) - \deg(f_j)} f_j$
    $p_j \leftarrow p_j + c z^{\deg(h) - \deg(f_j)}$
  else
    $r \leftarrow r + \mathrm{lt}(h)$, $\mathrm{lt}(h)$ is highest order term in h
    $h \leftarrow h - lt(h)$
  end if
end while

5.3.2 Groebner Bases

If the algorithm Alg. 5 returns $r = 0$, then $f \in \text{Ideal}(\{f_i\})$; in other words, f belongs to the ideal generated by $\{f_i\}$. However, it is not true that any $f \in \text{Ideal}(\{f_i\})$ results in a zero remainder: $r = 0$. For example, if $I = \text{Ideal}(f_1, f_2)$ with $f_1 \neq f_2$ but with the f_k having the same top monomial terms including scale factors, then $f_1 - f_2 \in I$ but $f_1 - f_2$ has a lower order top term than either f_1 or f_2, so that Alg. 5 applied to $f_1 - f_2$ results in $r = f_1 - f_2 \neq 0$. We need an enlargement of the given ideal basis in order to find a basis that determines ideal membership through the division algorithm Alg. 5.

The cancellation of top terms involving pairwise differences $f_j - f_k$ suggests that the following elaboration plays a role in enlarging the basis. First, define $\boldsymbol{\gamma} \stackrel{\text{def}}{=} (\gamma_1, \gamma_2)$ with $\gamma_k \stackrel{\text{def}}{=} \max(\deg_k(f_1), \deg_k(f_2))$. The following is the definition of an S-polynomial.

$$\text{SP}(f_1, f_2) = (\text{lc}(f_1))^{-1}\eta^{-(\gamma_2 - \deg_2(f_1))\deg_1(f_1)} z^{\boldsymbol{\gamma} - \deg(f_1)} f_1 - (\text{lc}(f_2))^{-1}\eta^{-(\gamma_2 - \deg_2(f_2))\deg_1(f_2)} z^{\boldsymbol{\gamma} - \deg(f_2)} f_2 \tag{135}$$

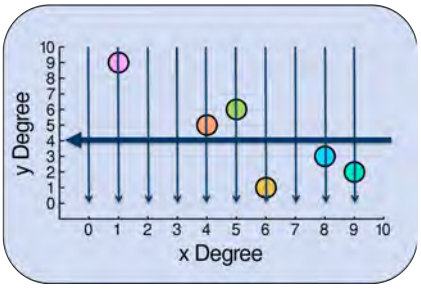
Conceptually, in bumping up the top terms of f_1, f_2 to a common degree top term and taking normalized differences to cancel the highest resulting term, we are creating new top terms in the ideal that must be reachable from a generating set that is capable of solving the ideal membership problem using Alg. 5. The following algorithm, due to Buchberger, creates a so-called Groebner basis for the ideal $\text{Ideal}(\{f_i\})$ that solves the membership problem.

Algorithm 6 Groebner Basis Algorithm

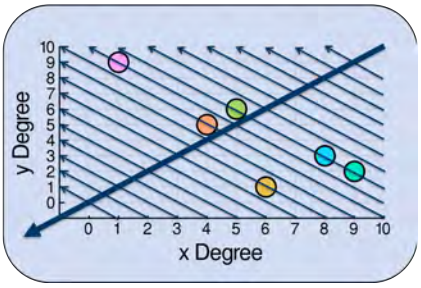
Goal: Given the finite set of nonzero polynomials $F \stackrel{\text{def}}{=} \{f_1, \ldots, f_s\} \subset \mathbb{C}^{(Ore)}[z_1, z_2]$, find a Groebner basis $\{g_1, \ldots, g_t\}$ of $\text{Ideal}(\{f_1, \ldots, f_s\})$
Initialize: $G \coloneqq F$ and $B \coloneqq \{\{f, g\} : f \neq g \in G\}$, set of all pairs from G
while $B \neq \emptyset$ **do**
  Choose any $\{f, g\} \in B$
  $B \leftarrow B \backslash \{f, g\}$, remove $\{f, g\}$ from B
  $h' = SP(f, g)$
  $h \leftarrow$ remainder of h' using G and division algorithm Alg. 5
  if $h \neq 0$ **then**
    $B \leftarrow B \cup \{\{p, h\} : p \in G\}$, add all new pairs involving h
    $G \leftarrow G \cup \{h\}$, increase set of generators
  end if
end while

Although it is not apparent, Alg. 6 terminates, yielding a finite set of generators for $\text{Ideal}(\{f_1, \ldots, f_s\})$ that includes all of the original generators. The new set, $\{g_1, \ldots, g_t\}$, solves the ideal membership problem using Alg. 5.

For illustration, two monomial orders are shown in Fig. 21.



(a) Monomial terms are ordered lexicographically, with primacy given the x-axis variable.



(b) Monomial terms are ordered by total degree, with ties broken by the order of the x-axis variable.

Figure 21. Two different monomial orders are shown, lexicographic and total degree. The color represents the sensor associated with each monomial term. Note that the terms are the same in the figures but the ordering is different.

5.3.3 Module Definitions and Relations with Zero-Forcing

We are interested in finding generators for sets of vectors with Ore polynomial entries. If the vectors are of length n, one set of vectors of interest are all vectors of length n with Ore polynomial entries, denoted $\bigoplus_1^n \mathbb{C}^{(Ore)}[z_1, z_2]$ and called the free module of rank n over $\mathbb{C}^{(Ore)}[z_1, z_2]$. These vectors are, of course, a linear space over any componentwise additions and multiplications with "scalar" Ore polynomial factors $f(z_1, z_2) \in \mathbb{C}^{(Ore)}[z_1, z_2]$. In other words, they behave just like the space of complex n-vectors, with the exception that the coefficients involved in the linear combinations are Ore polynomials, just like the vectors' entries, and these coefficients cannot be inverted in general. Furthermore, the order of coefficient multiplication matters, since the Ore polynomials are not commutative. We focus on multiplication on vectors from the left. The resulting analog of a vector space is called a module over the Ore polynomials. It differs from a traditional vector space in two significant ways. As mentioned, the coefficients are not invertible in general. Secondly, the coefficients that multiply vectors act by multiplying on the left.

More generally, we are interested in submodules of the free modules. These submodules are analogus to subspaces of a vector space and are built from generators associated with matrices.

For example, the channel matrix $\mathbf{H}$ of a communication link can be expressed as a matrix with entries drawn from $\mathbb{C}^{(Ore)}[z_1, z_2]$. Each entry of a given row and column represents the delay and Doppler shifts associated with propagation to a specific sensor (row) and arising from a specific source (column). If $\mathbf{H}$ is $n \times k$ with $k < n$, then the span of the rows of H form a submodule of the free module $\bigoplus_1^k \mathbb{C}^{(Ore)}[z_1, z_2]$. We say that the n rows of $\mathbf{H}$ generate a submodule $M_{\mathbf{H}}$ of the free module $\bigoplus_1^k \mathbb{C}^{(Ore)}[z_1, z_2]$ of rank k.

The submodules of most concern for us are those that provide zero-forcing vectors for $M_{\mathbf{H}}$. A vector $\mathbf{w} \in \bigoplus_1^n \mathbb{C}^{(Ore)}[z_1, z_2]$ is called zero-forcing iff $\mathbf{wH} = 0$. Implicitly, we assume a matrix representation of $\mathbf{w}$ as a row vector. More specifically, we want to find the submodule $M_{\mathbf{H}}^{\perp} \subseteq \bigoplus_1^n \mathbb{C}^{(Ore)}[z_1, z_2]$ of all n-vectors $\mathbf{w}$ with Ore polynomial entries that satisfy $\mathbf{wH} = 0$. Thus, the module $M_{\mathbf{H}}^{\perp}$ is the space of all zero-forcing weights. Denote by $\mathbf{H}^{\perp}$ a $m \times n$ matrix whose rows generate the module $M_{\mathbf{H}}^{\perp}$. In other words, $M_{\mathbf{H}^{\perp}} = M_{\mathbf{H}}^{\perp}$. Although $\mathbf{H}^{\perp}$ is not well-defined (it's not unique), the goal of this section is the construction of a $\mathbf{H}^{\perp}$ given $\mathbf{H}$.

5.3.4 Groebner Basis for a Module

As in the case of left ideals of the Ore ring, we are interested in a set of generators for a module that has small leading order terms in some appropriate sense related to the complexity of the zero-forcing solutions. With ideals, the multidegree $\boldsymbol{\alpha}$ is ordered compatibly with $\leq^2$ in one of several possible ways in order to formulate the multivariate division algorithm that is used with Groebner bases. For modules, we use the same approach, but augment the multidegree with a vector-component index k (the level) so that the module multidegree takes the form $(\boldsymbol{\alpha}, k)$, with $\boldsymbol{\alpha}$ representing the bidegree used above for Ore polynomials. The bidegree $\boldsymbol{\alpha}$ applies to the k^{th} vector-component of the module member. We use the ordering $\leq^{(2,n)}$ to denote the partial ordering $(\boldsymbol{\alpha}, j) \leq (\boldsymbol{\beta}, k)$ iff $j = k$ and $\boldsymbol{\alpha} \leq^2 \boldsymbol{\beta}$. Then an ordering $\preceq$ on multidegrees $(\boldsymbol{\alpha}, k)$ is compatible with $\leq^{(2,n)}$ if

$$\begin{aligned} (\boldsymbol{\alpha}, i) &\preceq (\boldsymbol{\alpha} + \boldsymbol{\beta}, i) \\ (\boldsymbol{\alpha}, i) \preceq (\boldsymbol{\beta}, j) &\rightarrow (\boldsymbol{\alpha} + \boldsymbol{\gamma}, i) \preceq (\boldsymbol{\beta} + \boldsymbol{\gamma}, j), \end{aligned} \tag{136}$$

where $\boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\gamma} \in \mathbb{N}^2$, the nonnegative 2-tuples of integers.

Write the leading term of a module member $\mathbf{h}$ as $\mathrm{lc}(\mathbf{h}) z^{\boldsymbol{\alpha}} \mathbf{e}_k$ with leading coefficient $\mathrm{lc}(\mathbf{h})$, degree $\deg(\mathbf{h}) = \boldsymbol{\alpha}$, and level $\mathrm{lev}(\mathbf{h}) = k$. The vector $\mathbf{e}_k$ is defined to have all zeros except for unity in the k^{th} entry.

Given a compatible ordering on the $\boldsymbol{\alpha}$ exponent of the Ore monomials, there are two interesting extensions to compatible orders on $(\boldsymbol{\alpha}, k)$ for modules. One, called position over term (POT), orders the level (k-component) first and then uses the order on $\boldsymbol{\alpha}$. The other, called term over position (TOP), orders by the highest order $\boldsymbol{\alpha}$ over any level k with level used to break ties. For the example presented below, we use the TOP extension of ordering.

Although we are interested in finding zero-forcing weight vectors for the module $M_{\mathbf{H}}$ generated the rows of the matrix $\mathbf{H}$, we must first find a Groebner basis for the module $M_{\mathbf{H}}$ itself. The techniques for finding this Groebner basis are completely analogous to those used to construct a Groebner basis for ideals, as presented in Alg. 6, which are based on a division algorithm utilizing

compatible orders. We simply use the order on module monomials given above. Details can be found in [BGTV03].

An illustration of a few steps in the process of computing a Groebner basis for a module is shown in Fig. 22. The final result of the basis construction is shown in Fig. 23. The steps involve the cancellation of leading order terms associated with pairs of module elements (rows of the matrix $\mathbf{H}$) using S-polynomials, and then reduction of the results using the division algorithm for modules, which is completely analogous to that for ideals ([BGTV03] has the details). Any nonzero remainders are added to the generators, as in the case with ideals.

5.3.5 Constructing Zero-Forcing Weights

Recall that a zero-forcing weight $\mathbf{w}$ for a module generated by the rows of a matrix $\mathbf{H}$ satisfies $\sum_k w_k \mathbf{h}_k = 0$, where $\mathbf{h}_k$ denotes the k^{th}row of $\mathbf{H}$.

The zero-forcing module for a Groebner module basis is defined using a variant of S-polynomials. Let the rows of the matrix $\mathbf{G}$ provide a Groebner basis for $M_{\mathbf{H}}$. Write the k^{th} row of $\mathbf{G}$ as $\mathbf{g}_k$. Then S-polynomials for zero-forcing are defined by

$$\begin{aligned}\mathrm{SP}(\mathbf{g}_j, \mathbf{g}_k) = (\mathrm{lc}(\mathbf{g}_j))^{-1}\eta^{-(\gamma_2 - \deg_2(\mathbf{g}_j))\deg_1(\mathbf{g}_j)} z^{\boldsymbol{\gamma} - \deg(\mathbf{g}_j)}\mathbf{g}_j \\ - \quad (\mathrm{lc}(\mathbf{g}_k))^{-1}\eta^{-(\gamma_2 - \deg_2(\mathbf{g}_k))\deg_1(\mathbf{g}_k)} z^{\boldsymbol{\gamma} - \deg(\mathbf{g}_k)}\mathbf{g}_k,\end{aligned} \tag{137}$$

when $\mathrm{lev}(\mathbf{g}_j) = \mathrm{lev}(\mathbf{g}_k)$ with γ defined as in Equ. (135). Since we can write

$$\mathrm{SP}(\mathbf{g}_j, \mathbf{g}_k) = \sum_l c_{jkl}\mathbf{g}_l, \tag{138}$$

with $c_{jkl} \in \mathbb{C}^{(Ore)}[z_1, z_2]$, it follows that

$$\begin{aligned}\mathbf{s}_{jk} \stackrel{\mathrm{def}}{=} (\mathrm{lc}(\mathbf{g}_j))^{-1}\eta^{-(\gamma_2 - \deg_2(\mathbf{g}_j))\deg_1(\mathbf{g}_j)} z^{\boldsymbol{\gamma} - \deg(\mathbf{g}_j)}\mathbf{e}_j \\ - \quad (\mathrm{lc}(\mathbf{g}_k))^{-1}\eta^{-(\gamma_2 - \deg_2(\mathbf{g}_k))\deg_1(\mathbf{g}_k)} z^{\boldsymbol{\gamma} - \deg(\mathbf{g}_k)}\mathbf{e}_k - \sum_l c_{jkl}\mathbf{e}_l\end{aligned} \tag{139}$$

is a zero-forcing weight for $\mathbf{G}$.

Theorem 11. *If the rows of* $\mathbf{G}$ *are a Groebner basis for the module* $M_{\mathbf{G}}$ *they generate, then the* $\mathbf{s}_{jk}$ *generate the module of zero-forcing weights for* $M_{\mathbf{G}}$*, where* j, k *range over all pairs of row indices of* $\mathbf{G}$*.*

Proof. See [BGTV03]. □

If $\mathbf{G}$ is the matrix representing a Groebner basis of $\mathbf{H}$, we can find rectangular matrices of Ore polynomials $\mathbf{Q}$ and $\mathbf{P}$ such that $\mathbf{G} = \mathbf{Q}\mathbf{H}$ and $\mathbf{H} = \mathbf{P}\mathbf{G}$. The matrix $\mathbf{P}$ can be derived by using the division algorithm while $\mathbf{Q}$ can be derived by keeping track of all the computations involved in building the Groebner basis. Note that $\mathbf{s}_{jk}\mathbf{Q}\mathbf{H} = 0$ (see Thm. 11). In other words, the $\mathbf{s}_{jk}\mathbf{Q}$ are zero-forcing weights for $\mathbf{H}$. Furthermore $(\mathbf{I} - \mathbf{P}\mathbf{Q})\mathbf{H} = 0$. In fact, we have

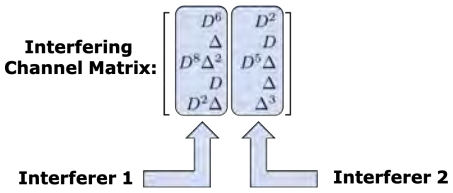
Theorem 12. *Let the rows of* $\mathbf{G}$ *be a Groebner basis for the row-space module of* $\mathbf{H}$. *Find Ore polynomial matrices* $\mathbf{Q}$ *and* $\mathbf{P}$ *such that* $\mathbf{G} = \mathbf{QH}$ *and* $\mathbf{H} = \mathbf{PG}$. *With* $\mathbf{s}_{jk}$ *defined as in Equ. (139), the zero-forcing module of* $\mathbf{H}$ *is generated by the* $\mathbf{s}_{jk}\mathbf{Q}$, *as the* j, k *range over all pairs of rows of* $\mathbf{G}$, *combined with the rows of the matrix* $\mathbf{I} - \mathbf{PQ}$.

Proof. See [BGTV03]. □

5.3.6 Equalization

We can apply generators $\mathbf{w}_i$ for the zero-forcing weights to the reponse vector $\mathbf{v}$ of a desired signal: one that has not been nulled. What results are Ore polynomials $\mathbf{w}_i \cdot \mathbf{v}$ applied to the desired signal. These polynomials represent distortion of the desired signal as the cost of nulling. All possible distortions due to zero-forcing weights are linear combinations $\sum_i c_i \mathbf{w}_i \cdot \mathbf{v}$ (with Ore coefficients c_i) of these basic distortions due to the generators. In other words, all distortion from zero-forcing weights form an equalization ideal generated by $\{\mathbf{w}_i \cdot \mathbf{v}\}$. The Groebner basis of this ideal allows us to solve, using the multivariable division algorithm Alg. 5, the ideal membership problem. In other words, we can easily test whether a particular member of $\mathbb{C}^{(Ore)}[z_1, z_2]$ belongs to the equalization ideal. The members of $\mathbb{C}^{(Ore)}[z_1, z_2]$ that do not distort the desired signal are pure monomials for the form $z^{\boldsymbol{\alpha}}$. These operators impart a simple delay and Doppler shift to the signal. If $z^{\boldsymbol{\alpha}}$ belongs to the equalization ideal, then so does $z^{\boldsymbol{\beta}}$ for any $\boldsymbol{\alpha} \leq^{(2)} \boldsymbol{\beta}$. Thus, one ad hoc procedure for finding equalization tests a $z^{\boldsymbol{\alpha}}$ with large $\boldsymbol{\alpha}$ for ideal membership.

For the case of delay-only operators treated in Section 5.1, equalization often leads to zero-forcing weights that are not sparse. This can adversely impact the complexity of adaptive beamformers. When legacy systems are important, equalization of some sort may be unavoidable. But given the opportunity to design waveforms for distributed array applications, equalization is not necessary to obtain good performance. Without equalization, channel distortions for the desired signal must be taken into account in error-correction coding.



(a) Shown is the channel matrix of a 5 element array for two interferers, whose response vectors are shown in each column. The responses have delay-Doppler monomials as entries that express the appropriate time and frequency shifts associated with signal (column) and sensor (row).

Form S-Polynomial:

Chosen Pair: $\{[\ D^6 \quad D^2\ ], [\ D^2\Delta \quad \Delta^3\ ]\}$

Match Leading Terms:
$\{[\ \boxed{\Delta D^6} \quad \Delta D^2\ ], [\ \boxed{D^6\Delta} \quad D^4\Delta^3\ ]\}$
$\{[\ \boxed{D^6\Delta} \quad \eta^4 D^2\Delta\ ], [\ \boxed{D^6\Delta} \quad D^4\Delta^3\ ]\}$

Subtract: $[\ 0 \quad D^4\Delta^3 - \eta^4 D^2\Delta\ ]$

(b) Each step in finding a Groebner basis chooses a pair of rows, multiplies by monomials to find common leading terms, and subtracts to cancel these leading terms. This is the beginning step in forming an S-polynomial (really, a module member).

Reduce S-Polynomial: $[\ 0 \quad D^4\Delta^3 - \eta^4 D^2\Delta\ ]$

Reduction Vector: $[\ \Delta \quad D\ ]$

Eliminate Leading Term:
$[\ 0 \quad D^4\Delta^3 - \eta^4 D^2\Delta\ ]$
$-\eta^3 D^3\Delta^3\ [\ \Delta \quad D\ ]$

$\boxed{[\ -\eta^3 D^3\Delta^4 \quad -\eta^4 D^2\Delta\ ]}$

(c) Next, the module generators (all the current rows) are used to reduce the result of the S-step. One step in the reduction is shown. Many others follow until zero is reached or no further reduction is possible. Any nonzero remainder is added to the Groebner module basis.

Figure 22. A Groebner basis for a module is formed. The basis and the S-polynomials can be used to form zero-forcing weights.

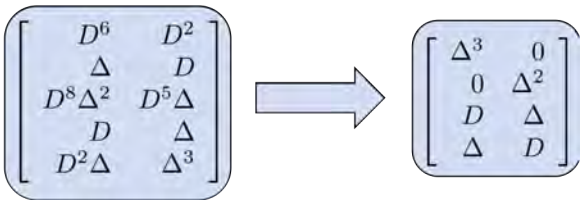


Figure 23. The initial channel matrix is shown along with the matrix containing (in rows) the module's Groebner basis. The row-space of the channel matrix on the left, as generated by multiplying on the left with Ore polynomials and then adding componentwise, is also generated by the rows on the right. However, the right-hand-side generators form a Groebner basis. In particular, this basis solves the membership problem for determining whether an arbitrary row vector is in the module. Furthermore, in the process of constructing the Groebner basis, we build the zero-forcing weights for the original channel matrix.

5.4 PERFORMANCE EXAMPLES

The basic flow for computing zero-forcing weights using Groebner basis techniques is shown in Fig. 24. First, the interference channel is estimated. This can be accomplished with cooperative channel sounding techniques or estimation can be based on geolocation techniques. Only the signals' differential delays and Dopplers are required. The complex weights needed for phase and gain control can be estimated from the data once the delay-Doppler taps are located algebraically. Second, a Groebner basis is computed for the module determined by the channel matrix (see above). This step provides some generators for the zero-forcing module. The remainder of the generators are determined by using both the original channel matrix and its Groebner basis, in conjunction with relations between the two. This process is complicated even in simple cases and relies on efficient software implementations of the calculations. The final step uses generators of the zero-forcing weights to find weights that both zero-force and equalize a desired signal.

We present an example of algebraic weights that involves all but the equalization step in the signal processing chain of Fig. 24.

A known channel matrix is shown in Fig. 25a. Five sensors are used to null two interferers. Both delay and Doppler spreads are large in the sense that the exponents of the delay Δ and Doppler D shifts are large. Typically, these shifts correspond to multiples of fundamental resolution cells ($\frac{1}{B} \times \frac{1}{T}$) associated with bandwidth B and observations time T. Shown in Fig. 25b are the locations of the delay-Doppler taps for one zero-forcing weight. The axes indicate the exponents of the corresponding operators. Each color is associated with a different sensor. We can see that not all sensors are used for nulling. This is a common situation for generators constructed with algebraic techniques.

The extent of the delay and Doppler spreads is roughly 36×165. If one were to fill the entire extent with delay-Doppler taps, zero-forcing could be accomplished using a dense grid of $5 \cdot 36 \cdot 165 = 29700$ taps. Zero-forcing would then involve solving a very underdetermined system of equations. This is an approach that will be examined in Section 6. However, the algebraic solution uses well less that 100 taps. Besides a dramatic improvement in computational complexity, a small number of taps requires much less training data in order to determine the tap coefficients. Other generators for the zero-forcing weights of this channel matrix have similar delay-Doppler scatter plots.

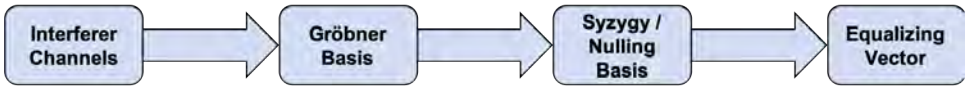
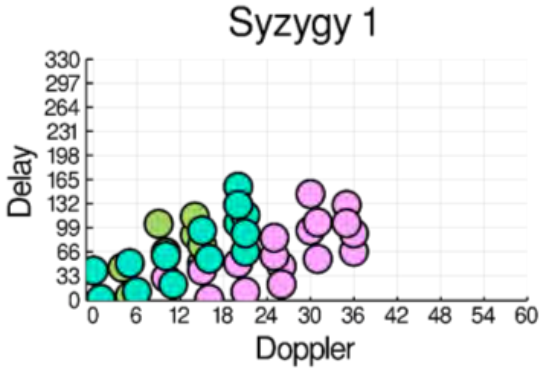


Figure 24. The interference channel matrix is first converted to a module Groebner basis. The combinations of rows of the channel matrix that zero-force are computed from the coefficients used to form the Groebner basis. This provides generators for all zero-forcing weights. If required, the zero-forcing generators are applied to the array response of the desired signal. The resulting Ore polynomials generate a left ideal whose Groebner basis is evaluated. If the ideal contains a monomial, equalization is possible.

$$\begin{bmatrix} 3D^{10}\Delta^{30} & D^{6}\Delta^{6} \\ 6D^{21}\Delta^{36} & 4D^{16}\Delta^{26} \\ 12D^{26}\Delta^{41} & 8D^{21}\Delta^{31} \\ D^{5} & \Delta^{15} \\ D^{20} & 2D^{10}\Delta^{5} \end{bmatrix}$$

(a) Shown is the channel matrix of a 5 element array for two interferers, whose response vectors are shown in each column. The responses have delay-Doppler monomials as entries that express the appropriate time and frequency shifts associated with signal (column) and sensor (row).



(b) One of the zero-forcing weights that generates the module of syzygies (zero-forcing weights) is shown. Each color represents a different sensor. The position of a disk indicates a nonzero monomial term for that sensor (color). Note that not all sensors are used for this nulling weight. This is a common situation.

Figure 25. Zero-forcing weights are shown for the channel with the algebraic channel matrix. One typical weight from a generating set of zero-forcing weights is illustrated. The disks in the subplot indicate nonzero terms (i.e., delay-Doppler taps) in the zero-forcing weight. The number of taps required to cover densely the full range of the shown delay and Doppler taps is very large, equaling $N_{\text{sensors}} \cdot N_{\text{delay}} \cdot N_{\text{Dop}}$. The sparse-taps solution shown above uses less the .3% of the taps in the dense solution. As a consequence, the amount of data needed to train the unknown tap coefficients is dramatically reduced.

This page intentionally left blank.

6. COMPRESSIVE SENSING AND SPARSE TAPS

6.1 INTRODUCTION TO COMPRESSIVE SENSING

Compressive sensing (CS) addresses the solution of an underdetermined set of equations given a linear, functional relationship between a model and observations. In concrete terms, denote the observations by the vector $\mathbf{z}$ and model by the vector $\mathbf{x}$. In the absence of noise, we have $\mathbf{z} = \mathbf{A}\mathbf{x}$. Since this system of equations is underdetermined when $\mathbf{A}$ has more columns than rows, the degeneracy of solutions must be broken in some manner.

The least-squares approach handles the degeneracy by asking for the solution to $\mathbf{z} = \mathbf{A}\mathbf{x}$ with smallest L_2 norm (Euclidean). This procedure is illustrated in Fig. 26, which shows a 2-D problem with a linear constraint space of candidate solutions. The solution closest to the origin is the intersection of a contour of constant L_2 norms with the constraint space. The contour value is the smallest that yields an intersection.

We are interested in signals that have sparse representations in some basis. Since we allow degeneracies, and hence linear dependancies between representing vectors, it is more proper to use the term frame rather than basis. For the linear relationship $\mathbf{z} = \mathbf{A}\mathbf{x}$, sparseness is equivalent to a small number of nonzero components in $\mathbf{x}$ given the frame formed by the columns of $\mathbf{A}$. The approach illustrated in Fig. 27 can provide sparse solutions. The degeneracy is broken by choosing the smallest contour of L_1 norms (the box contours) that intersects the constraint space. The nature of these contours forces a solution that, generically, lies on one of the axes. Thus L_1 norm minimization can yield sparse solutions.

The appropriate problem to solve, if a sparse solution is desired, is the minimization of the L_0 norm under the linear constraints. The L_0 norm of a vector equals the number of nonzero components, which is exactly what we want to minimize. If there exists a solution to $\mathbf{z} = \mathbf{A}\mathbf{x}$ with sufficiently sparse $\mathbf{x}$, then minimizing the L_1 norm yields the same solution as minimizing the L_0 norm (see [Don06] for a similar result in the presence of noise).

There are a variety of computational techniques and applications of CS and an extensive literature. Among the variety of computational approaches, CoSaMP [NT09] offers good efficiency as well as performance. We use this technique here for sparse tap placement.

CoSaMP requires a sparsity goal as an input argument and has certain technical requirements on the linear system being solved in order to provide performance guarantees. These technical requirements are difficult to verify, in practice, but often true with high probability when the linear systems are randomly chosen. In short, the guarantees possible with algebraic methods are not readily available with CS. Nonetheless, CoSaMP offers a very useful tool for tap placement. To handle the unknown degree of sparsity, we employ a binary search to find the sparsest solution, when possible. We check residuals to make sure a near-solution is found at each stage.

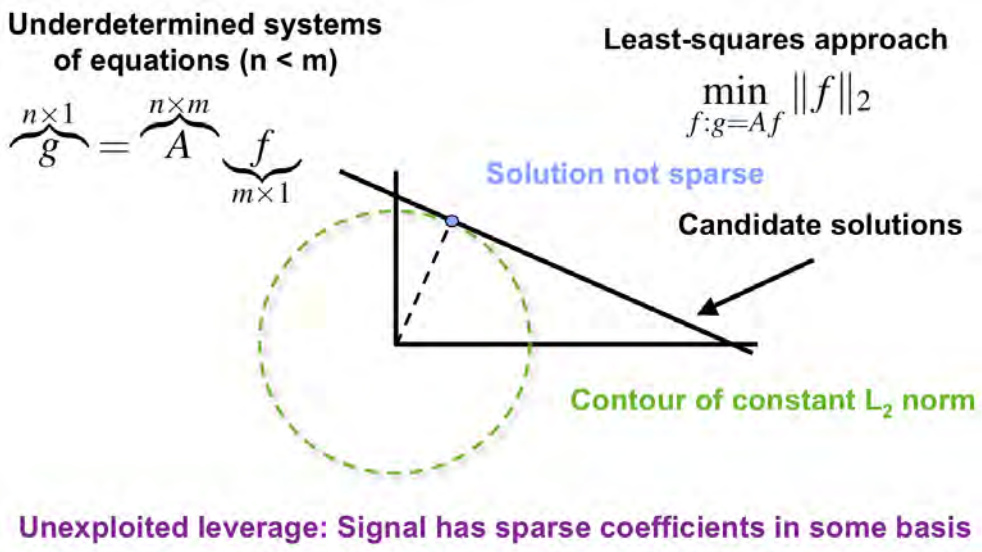


Figure 26. Degeneracies in the space of candidate solutions can be resolved by finding the solution closest to the origin. This procedure amounts to determining the smallest contour of constant Euclidean norm that intersects the solution space. Note that the solution is almost never sparse.

6.2 APPLICATION: SPARSE TAP PLACEMENT

To apply compressive sensing to sparse delay-Doppler taps, we first layout a dense grid of delay-Doppler taps for each sensor. Typically, both the delay and Doppler extents of the grid will be small multiples of the differential delay and Doppler spreads. Based on the regularity of the grid and the channel matrix (really, just delay and Doppler shifts at each sensor for each signal) of the interfering signals, we can write down the nulling equations in matrix-vector form, as given, for example, in Equ. (107). In the notation of that equation, $\mathbf{x}$ represents the nulling vector with operator entries. To cast the equation in the form of matrices over the complex numbers, we need to expand $\mathbf{x}$ into a vector with entries labeled by sensor, delay shift, and Doppler shift. The matrix $\mathbf{\Gamma}$, with operator entries, is similarly expanded in size. This gives us a known matrix, call it $\tilde{\mathbf{\Gamma}}$, that expresses the channel. Only the differential delays and Dopplers are available to build $\tilde{\mathbf{\Gamma}}$, so the complex signal amplitudes must be supplied in a somewhat arbitrary fashion. Remember, tap placement is the goal. The values of the amplitudes should have little effect on where the taps need to go for zero-forcing.

The linear operator relation $\mathbf{\Gamma x} = \mathbf{b}$ can also handle equalization through $\mathbf{b}$. As we have seen, however, equalization tends to fill in otherwise sparse taps. Since the CS algorithms require the existence of sparse solutions, they are not able to handle equalization as well as the algebraic techniques of Section 5.

We have an example, in Fig. 28 and Fig. 29, of tap placement in a scenario involving strong cochannel interferers. Delay-only taps based on the algebraic techniques from Section 5 were used

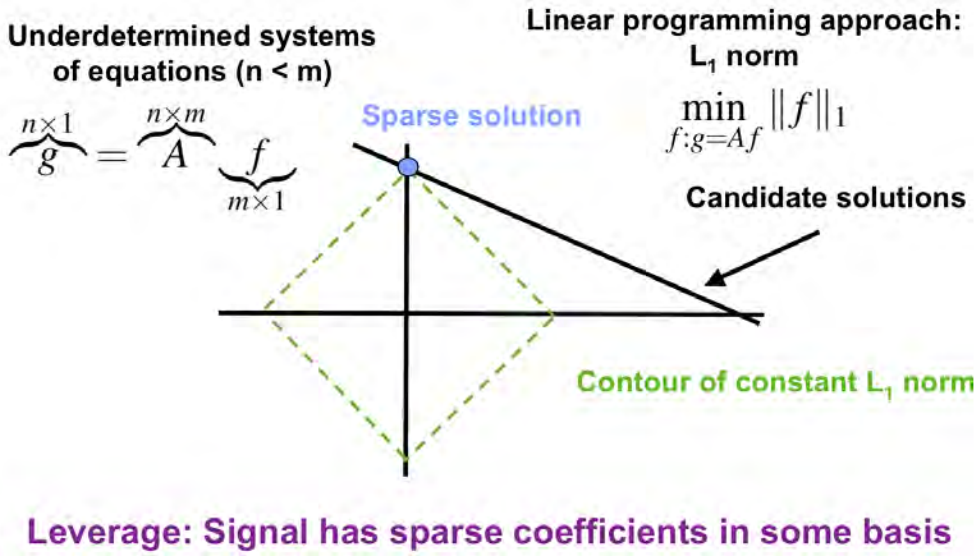


Figure 27. When sparse solutions are desired, a better approach finds the smallest contour of the L_1 norm that intersects the solution space. These contours are shaped in a way that results in solutions with few nonzero coordinates.

in Fig. 28, with poor performance. The algebraic construction achieved equalization but was unable to do a good job mitigating interference.

For comparison, in Fig. 29 a 50×15 delay-Doppler grid of potential tap locations was used, along with the example's differential delays and Dopplers, to select sparse tap patterns for zero-forcing using compressive sensing. The selected tap locations are shown. No equalization was attempted. Performance is substantially improved with the use of delay-Doppler taps. Furthermore, the number of delay-Doppler taps employed for this sparse solution is a small fraction of the number of taps required for a dense solution. There would not be enough training data to determine the unknown tap weights for the dense solution, but there would be enough data to train the sparse taps.

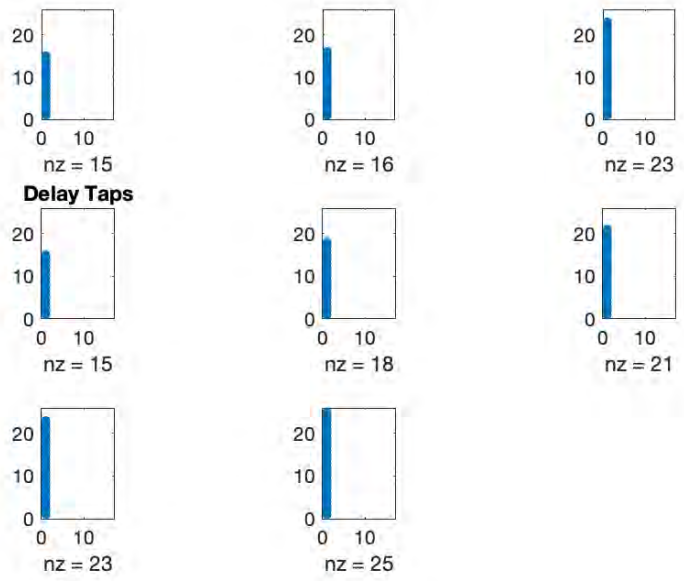


Figure 28. Shown are the tap placements for delay-only taps using the algebraic techniques of Section 5. Taps are placed for the purpose of zero-forcing and equalization based on the differential delays of the signals. The subplots, one for each sensor, show tap locations within a dense grid of potential tap locations. Although the subplots show an integer grid, true grid spacings are $\frac{1}{2B}$ in delay (y-axis) and $\frac{1}{2T}$ in frequency (x-axis). The number of taps used for each sensor is indicated by the notation nz = tap count. Characteristic of the tap placements when equalization is required is the filling-in of the tap regions. Thus, the taps are generally uniformly spaced, with different numbers and offsets at each sensor. The signal bandwidth is 50 kHz and performance is measured over a time extent of 20 ms. There are 8 sensors and 4 signals. The desired signal, for which performance is measured, has an element SNR of 10 dB while the interfering signals have equal SNRs of 40 dB. Delays spreads are around $200\mu s$ and Doppler spreads are about 75 Hz. Performance with delay-only taps is poor, with an effective SNR of about -23 dB. The number of taps used is 156.

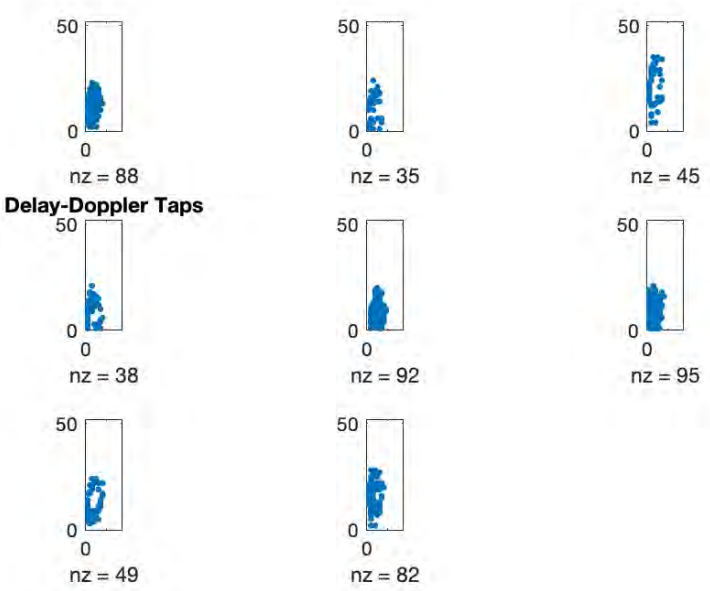


Figure 29. The same example used in Fig. 28 is treated here with 524 sparse delay-Doppler taps derived using compressive sensing. Performance is much better, with an effective SNR of 7 dB. For a dense tap solution over the same time and bandwidth, we would require at least 3200 delay-Doppler taps. Since the TB product is 1000, there are enough samples to support sparse taps, but not the dense tap placement. Note that the real delays and Dopplers are randomized versions of the design values, with the random displacements sized to $\frac{1}{2B}$ *in delay and* $\frac{1}{2T}$ *in Doppler.*

This page intentionally left blank.

7. CONCLUSIONS

We examined four techniques for designing sparse layouts of delay-Doppler taps to support adaptive beamforming with distributed arrays. Delay-Doppler tap placement is a natural wideband extension of the layout of antenna elements for narrowband adaptive beamforming. For distributed arrays, tap placement is used to compensate for the large variations in delay and Doppler seen by each sensor for each signal. Delay and Doppler shifts applied to the sensors are used to align cochannel signals in a manner that supports adaptive beamforming, which minimizes interference and equalizes a desired signal.

Three of the tap-placement techniques investigated, greedy mutual information maximization, algebraic zero-forcing, and compressive sensing, are used to provide examples of tap layout and beamforming performance. The fourth technique, fraction-free Gaussian elimination is described and illustrated in detail for a simple, but useful example. The performance of all of the tap layouts can be evaluated for Gaussian signals using statistics developed here.

The tap placement techniques treated have different strengths and weaknesses with regard to computational challenges, extensibility to large problems, and flexibility. The algebraic zero-forcing techniques are the most flexible in that they provide a description of all tap patterns that support cancellation of interference as well as equalization of a desired signal. Algebraic techniques are also the most difficult to execute efficiently. Compressive sensing can also provide zero-forcing solutions, without as much of a computational challenge. However, compressive sensing provides little control over the form of the solutions and does not provide the nonsparse solutions that are generally associated with equalization. Maximizing mutual information in a greedy manner is the most extensible of all the techniques, but does not offer any performance guarantees for the interference environments that we care about. Nevertheless, all of the algorithms we treat provide sparse tap layouts that offer good performance in challenging environments involving large delay and Doppler spreads.

This page intentionally left blank.

REFERENCES

[BDES19] E. Basor, J. Dubail, T. Emig, and R. Santachiara. Modified Szegö–Widom Asymptotics for Block Toeplitz Matrices with Zero Modes. *Journal of Statistical Physics*, 174(1):28–39, Jan 2019.

[BGTV03] José Bueso, José Gómez-Torrecillas, and Alain Verschoren. *Algorithmic Methods in Non-Commutative Algebra: Applications to Quantum Groups*. Springer Netherlands, Dordrecht, 2003.

[BM04] Nicolas Le Bihan and Jérôme Mars. Singular value decomposition of quaternion matrices: a new tool for vector-sensor signal processing. *Signal Processing*, 84(7):1177 – 1199, 2004.

[BWHZ10] C. R. Berger, Z. Wang, J. Huang, and S. Zhou. Application of compressive sensing to sparse channel estimation. *IEEE Communications Magazine*, 48(11):164–174, November 2010.

[Don06] David Donoho. For Most Large Underdetermined Systems of Equations, the Minimal L1-norm Near-Solution Approximates the Sparsest Near-Solution. *Communications in Pure and Applied Mathematics*, 59:907–934, 2006.

[GGRW05] Israel Gelfand, Sergei Gelfand, Vladimir Retakh, and Robert Lee Wilson. Quasideterminants. *Advances in Mathematics*, 193(1):56 – 141, 2005.

[HC12] B. W. Han and J. H. Cho. Capacity of Second-Order Cyclostationary Complex Gaussian Noise Channels. *IEEE Transactions on Communications*, 60(1):89–100, January 2012.

[JT00] Michele Jr and P. Tilli. Asymptotic Spectra of Hermitian Block Toeplitz Matrices and Preconditioning Results. *SIAM Journal on Matrix Analysis and Applications*, 21:867–881, 02 2000.

[KM71] G. Kallianpur and V. Mandrekar. Spectral Theory of Stationary H-Valued Processes. *Journal of Multivariate Analysis*, (1):1–16, 1971.

[LS95] Hong R. Lee and B.David Saunders. Fraction Free Gaussian Elimination for Sparse Matrices. *Journal of Symbolic Computation*, 19(5):393 – 402, 1995.

[LS12] Viktor Levandovskyy and Kristina Schindelar. Fraction-free algorithm for the computation of diagonal forms matrices over Ore domains using Gröbner bases. *Journal of Symbolic Computation*, 47(10):1214 – 1232, 2012. Symbolic Computation and its Applications.

[NT09] D. Needell and J.A. Tropp. Cosamp: Iterative signal recovery from incomplete and inaccurate samples. *Applied and Computational Harmonic Analysis*, 26(3):301 – 321, 2009.

[Ore33] Oystein Ore. Theory of Non-Commutative Polynomials. *Annals of Mathematics*, July, 1933.

[SJW17] G. Shulkind, S. Jegelka, and G. W. Wornell. Multiple wavelength sensing array design. In *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 3424–3428, March 2017.

[SJW19] G. Shulkind, S. Jegelka, and G. W. Wornell. Sensor array design through submodular optimization. *IEEE Transactions on Information Theory*, 65(1):664–675, Jan 2019.

[War98] James Ward. Space-time adaptive processing for airborne radar. *IET*, 1998.

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. REPORT DATE *(DD-MM-YYYY)*	2. REPORT TYPE	3. DATES COVERED *(From - To)*
07-01-2020	Report	

4. TITLE AND SUBTITLE

Beamforming with Distributed Arrays

5a. CONTRACT NUMBER
FA8702-15-D-0001

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER

6. AUTHOR(S)

K.W. Forsythe, L. Collins, H. Naumer

5d. PROJECT NUMBER
2239

5e. TASK NUMBER

5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

MIT Lincoln Laboratory
244 Wood Street
Lexington, MA 02421-6426

8. PERFORMING ORGANIZATION REPORT NUMBER

LSP-270

9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)
Under Secretary of Defense for Research and Engineering, and the U.S. Air Force

10. SPONSOR/MONITOR'S ACRONYM(S)
USDR&E, and USAF

11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION / AVAILABILITY STATEMENT

Approved for public release: distribution unlimited.

13. SUPPLEMENTARY NOTES

14. ABSTRACT

Spatial beamforming using distributed arrays of RF sensors is treated. Unlike the observations from traditional RF antenna arrays, the distributed array's data can be subjected to widely varying time and frequency shifts among sensors and signals. These shifts require compensation upon reception in order to perform spatial filtering. To perform beamforming with a distributed array, the complex-valued observations from the sensors are shifted in time and frequency, weighted, and summed to form a beamformer output that is designed to mitigate interference and enhance signal energy. The appropriate time-frequency shifts required for good beamforming are studied here using several different methodologies.

15. SUBJECT TERMS

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			19b. TELEPHONE NUMBER *(include area code)*
Unclassified	Unclassified	Unclassified	Same as report	102	