ACOUSTIC SOURCE LOCALISATION

A thesis presented in partial fulfilment of the requirements for the degree of
Master of Science
in
Mathematics
at Massey University, Palmerston North,
New Zealand.

Alexander Lyndon White

2019

Contents

Abstract

Many New Zealand native bird species are under threat, and as such conservationists are interested in obtaining accurate estimates of population density in order to closely monitor the changes in abundance of these species over time. One method of estimating the presence and abundance of birdlife in an area is using acoustic recorders; currently, omnidirectional microphones are used, which provide no estimate of the direction of arrival of the call. An estimate of the direction from which each sound came from would help to discern one individual calling multiple times, from multiple birds calling in succession – thus providing more accurate information to models of population density. The estimation of this direction-of-arrival (or DOA) for each source is known as *acoustic source localisation*, and is the subject of this work.

This thesis contains a discussion and application of two families of algorithm for acoustic source localisation: those based on the Generalised Cross-Correlation (GCC) algorithm, which applies weightings to the calculation of the cross-correlation of two signals; and those based on the Multiple Signal Classification (MUSIC) algorithm, which provides an estimate of source direction based on subspaces generated by the covariance matrix of the data. As the MUSIC algorithm was originally described for narrowband signals – an assumption not applicable to birdsong – we discuss several adaptations of MUSIC to the broadband scenario; one such adaptation requiring the use of polynomial matrices, which are described herein.

An experiment was conducted during this work to determine the effect that the distance between the microphones in a microphone array has on the ability of that array to localise various acoustic signals, including the New Zealand native North Island Brown Kiwi, *Apteryx mantelli*. It was found that both GCC and MUSIC benefit from larger inter-array spacings, and that a variant of the MUSIC algorithm known as autofocusing MUSIC (or AF-MUSIC) provided the most precise DOA estimates.

Though native birdlife was the motivator for the research, none of the methods described within this thesis are necessarily bound only to work on recordings of birdsong; indeed, any multichannel audio which satisfies the necessary assumptions for each algorithm would be suitable.

As well as a description of the algorithms, an implementation of GCC, MUSIC, and AF-MUSIC was produced in the Python 3 programming language, and is available at `https://github.com/alexW335/Locator`.

Acknowledgements

I would like to thank my supervisors, Dr. Richard Brown and Prof. Stephen Marsland, for their encouragement, patience, and advice throughout my degree. I'd like to thank my father for providing me with the theme "quantum electrodynamics" for my poster on the letter 'Q' in primary school, and starting me on a long journey of looking up things I don't understand solely because they sound interesting. I'd like to thank my mother for her unwavering support in whatever I do, my brother for generally being rad 24/7, and I'd like to thank Kelsey for putting up with me for quite some time now; I couldn't have done it without you.

In addition, I am exceptionally grateful to have received funding from the National Science Challenge on Science for Technological Innovation, and the Ministry of Business, Innovation and Employment to enable me to have this opportunity.

Chapter 1

Introduction

1.1 Motivation

The New Zealand native kiwi bird is a national icon, its citizens adopting the bird's name as a demonym, and its name even being adopted by state-owned businesses such as Kiwibank and KiwiRail. Unfortunately, the kiwi, along with many other native birds, is under threat – four out of five living kiwi species are categorised as "Vulnerable" by the IUCN Red List, and two of the five are further decreasing in number [1–5]. In order to preserve the species, conservationists use information about both the number and the whereabouts of animals to effectively target their efforts, and to establish trends in population counts. There are six main methods used by the New Zealand Department of Conservation in their Kiwi Recovery Programme to determine the presence and abundance of kiwi in a given area [6, 7]:

- Casual observations, i.e. reported sightings of signs of kiwi by the general public, including sightings of any of: the birds themselves, their feathers, probe holes (holes left in the ground from the kiwi searching for insects), footprints, or their faeces.
- Kiwi call counts. These are voluntary counts of kiwi vocalisations, often conducted by field workers when working in areas which are already known to be potential kiwi habitats. These call counts involve standing at a location for two hours and listening for birds. Upon hearing a potential kiwi call, the listener records the time, estimated distance, and direction-of-arrival of the call.
- Use of acoustic recorders. This involves placement of a single omnidirectional microphone in a location of interest, and returning later to collect the recorded audio for analysis. These microphones may record for long periods of time, and have a detection rate around 80% that of the human call counts as described above, however they cannot estimate the direction from which each bird call was heard.
- Walk-through surveys. These are where the conservationist walks through an area of interest, recording the time and approximate position of any kiwi calls heard, as well as transmitting pre-recorded kiwi calls to incite replies.

- Dog surveys. In areas which are known to be high-density kiwi habitats, the use of trained "kiwi dogs" is possible. These are highly trained dogs which have been trained to track kiwi using their scent.
- Trail cameras. Motion-activated infra-red cameras have become common-place for their use in game hunting, and may be affixed to a tree and left to record images for up to a year with no necessary maintenance. Trail cameras may potentially enable the identification of individual birds, however they require the birds to pass through the camera's field of view, which is a much smaller area than that covered by acoustic recorders; the kiwi must pass directly in front of the camera within a few meters, whereas in ideal conditions the kiwi call can be heard and recorded from up to 2 km away.

The method using acoustic recorders benefits from having a large area of effect, being minimally invasive, and requiring low human input once set up. However, one omni-directional microphone as currently used can not discern between sources in space; i.e. it may not be possible to distinguish whether two recorded calls arose from two birds at separate locations, or one bird calling twice. To address this, we aim to record on multiple microphones simultaneously, and to use the multi-track audio captured to estimate the birds' relative direction from the microphones at the time of its vocalisations, in a procedure is known as *acoustic source localisation*. These direction estimates may then be used to better distinguish between individuals in situations as described above, and hence provide more precise estimates of population density such as was done in a 2016 study on gibbon populations by Kidney *et al.* [8].

1.2 Objectives

We aim to provide a thorough and cohesive overview of both the GCC algorithm, and several variants of the MUSIC algorithm (MUSIC, ISSM, AF-MUSIC, PS/PSS-MUSIC), the latter of which requires the description of polynomial matrices. As well as a discussion of the theoretical basis for each algorithm, we will provide a software implementation of a subset of them in the Python programming language, to be available for public use. We will demonstrate the use of the code in the localisation of several sounds, including the call of the female North Island Brown Kiwi, *Apteryx mantelli*, and use it to investigate the effect that the geometry of the microphone array has on its ability to effectively localise sources.

1.3 Results

Of the two algorithms tested; GCC, and a variant of the MUSIC algorithm known as autofocusing MUSIC (or AF-MUSIC), both appeared to be able to adequately localise the sources trialled, to different degrees of precision. Several adaptations (known as 'processors') of the GCC algorithm were tested, with one of the more versatile ones known as ρ-PHAT showing the most promise for use in acoustic source localisation. In general, AF-MUSIC provided a more precise DOA estimate than the GCC algorithm, as well as a lower tendency to provide false estimates – however at a higher computational cost.

1.4 Outline

This thesis is organised as follows. Chapter 2 provides a summary of the current state of the field of Acoustic Source Localisation, as well as a discussion of methods for the eigenvalue decomposition of polynomial matrices. Chapter 3 introduces the GCC algorithm; one of the two algorithms which were selected for use in direction-of-arrival estimation for this thesis. Chapter 4 introduces the concept of polynomial matrices, and discusses their eigenvalue decomposition and some related properties. Chapter 5 introduces the second algorithm of interest – the MUSIC algorithm – and some relevant variants, including one which utilises the polynomial matrices described in Chapter 4. Chapter 6 describes the experiment conducted as part of this research, and Chapter 7 both provides an overview of the results, and outlines the scope for future work.

Chapter 2

Literature Review

2.1 Acoustic Source Localisation

Technologically aided Acoustic Source Localisation dates back to at least as far as 1918. Sir Alfred Rawlinson wrote in *The Defence of London* [9] about a 'machine' used for localisation of enemy airships when visibility was low. Two large gramophone trumpets, with doctors' stethoscopes affixed to the base, were attached to either end of a long horizontal pole which was free to rotate around a central pivot. Blind men, with their already elevated sense of hearing [10, 11], were then stationed at each of these machines and instructed to turn their heads – and thus the machine attached to them – towards the sound of an enemy aircraft. While they turned, a pencil and compass pad tracked the bearing towards which they were facing. Several of these contraptions were stationed surrounding London, and were used to track the flight paths of German warplanes as they flew overhead. Though we may not apply this method directly in our goal of localising birdcalls, it sets the scene for the idea of technologically aided acoustic source localisation; in essence, we aim to virtually replace the gramophone trumpets with microphones, and the man steering the machine with a computer.

The act of localising a sound generally involves using in some way the time-delays induced by the signal arriving between pairs of receivers. For example, if a plane wave arrives at your right ear τ seconds before it arrives at your left ear, and your head is d meters in diameter, then the sound must have came from

$$\theta^\circ = \pm \arccos\left(\frac{c_s \tau}{d/2}\right), \tag{2.1}$$

where c_s is the speed of sound in air. Using 2.1, we can reduce the problem down to estimating these time-delays from the observed data. However, estimating the time-delay between two waves is not trivial. One approach is to use the cross-correlation between two functions $f(t)$ and $g(t)$, $R_{fg}(\tau)$, defined by

$$R_{fg}(\tau) = \int_{-\infty}^{\infty} f(t)g(t-\tau)dt, \tag{2.2}$$

which is a measure of the similarity of f and g as a function of the time-delay τ. Even before the advent of digital signal processing (DSP), the cross-correlation between two

signals was being used to produce an estimate of the time delay [12, 13]. It was in the 1960s that Cooley and Tukey [14] described arguably one of the most important algorithms of all time, known as the Fast Fourier Transform (FFT), and with it opened the door for DSP to begin to see widespread use. The Fast Fourier Transform brought with it the ability to speed up the computation of the correlation hugely, thanks to the convolution theorem. This allowed the correlation to be a useful tool in practice, rather than just in theory, as it brought the computation time down to something practical.

The first major work related to Acoustic Source Localisation came from Peter Roth in his 1971 paper *Effective Measurements Using Digital Signal Analysis* [15]. Roth realised there was a problem not with the correlation itself, but with its common interpretation. He showed that the current interpretation of the correlation between two functions was only valid in particular cases, and as a result was in general not showing what it was thought to show. Roth then developed a weighting to apply to the correlation during its calculation which made the end result more easily interpretable, and less ambiguous. The *Roth Processor*, as it is now known, does not appear to have been used widely since then, however it was later shown to be a Maximum Likelihood estimator for the time delay between two signals under certain conditions [16].

2.1.1 The Generalised Cross-Correlation

In 1976, Knapp and Carter proposed the now common-place Generalised Cross-Correlation (GCC) algorithm [16]. Knapp and Carter further investigated Roth's discussion of applying weightings during the calculation of the correlation, and found that applying different weightings corresponded to different forms of a particular kind of pre-processing known as filtering. Filtering is the process of removing or amplifying particular components of a signal; a common example of a filter is the ubiquitous audio equaliser, found in cellphones, televisions and media players. Knapp and Carter described how one particular weighting, the Phase Transform (leading to a variant of GCC known as GCC-PHAT), theoretically provides a very precise estimate for the time delays, provided that the noise in the signal is uncorrelated both with the source and with itself between channels. GCC-PHAT has been used widely since then, for example: Wang and Chu used a modified GCC-PHAT to automatically direct a video camera in a videoconferencing environment [17]; and Gray and Hioka tested the PHAT weighting, along with a Maximum Likelihood weighting and a linear combination of the two, in locating New Zealand Kiwi [18]. Gray and Hioka exploited the known harmonic structure of the kiwi call to automatically distinguish sections in the data which contain calls from those which do not, and thus estimate the harmonic structure of noise for use in their Maximum Likelihood weighting. Other GCC weightings exist, including the Eckart processor [16] and the Smoothed Coherence Transform (SCOT) [19], both of which act to apply a lighter weighting in the calculations to the frequencies which have a lower estimated signal-to-noise ratio, and thus weight the source higher in the calculation. Ritu and Dhull [20] provide a summary of the different processors, and a comparison of their performance.

2.1.2 The Multiple Signal Classification (MUSIC) Algorithm

In 1979, Ralph Schmidt proposed a method with higher resolution than GCC, which takes a different approach altogether. The algorithm, known as Multiple Signal Classification (or MUSIC), uses calculations based on subspaces generated from the data, rather than directly using the data itself. Schmidt originally proposed the algorithm at the RADC Spectrum Estimation Workshop, held in October 1979 at Griffiss Air Force Base, New York – however as the proceedings from that workshop were very limited in circulation, and as the method quickly became the standard for use, Schmidt's paper was republished in IEEE Transactions on Antennas and Propagation in 1986 [21]. MUSIC has an advantage over GCC in that it may estimate multiple simultaneous source directions, as well as the characteristics of the original sources. MUSIC also benefits from 'superresolution', i.e. its method of action does not confine its DOA predictions to be discretised by the discrete nature of the data. MUSIC does come with its disadvantages, however, primarily that in its original form it is confined to work on only one frequency component of the data. This means that for broadband situations, i.e. ones in which the signal to be localised is built up out of a wide range of frequencies, MUSIC only utilises a small amount of the available information.

The extension of traditional MUSIC to a broadband scenario has been done in at least three distinct ways: the first, and simplest of the three, is to break the signal up into many smaller frequency bins to apply regular MUSIC to, and then produce a weighted linear combination of the resulting location estimates [22]. This method is known as the Incoherent Signal Subspace Method (ISSM), and breaks down both in situations where there are a different number of sources across different frequency ranges, and in a low signal-to-noise ratio environment. The second method is an advancement on the first, known as auto-focusing MUSIC, or AF-MUSIC [23]. AF-MUSIC 'shifts' the covariance matrices from all frequencies up to some reference frequency in such a way that leaves their eigen-structures intact, and then runs the traditional MUSIC algorithm on this combined space. The third approach is to use polynomial matrices [24] (a polynomial matrix is a matrix with polynomial entries, or equivalently a polynomial with matrix coefficients) and the polynomial eigenvalue decomposition (PEVD) [25]. The ISSM and polynomial matrix methods for broadband MUSIC are compared in a paper by Alrmah, Weiss, and Lambotharan [26], in which the PEVD-based approach to broadband MUSIC was introduced. Weiss *et al.* [27] showed that the method using polynomial matrices had benefits in accuracy and resolution over the frequency-binned option, assuming one has an efficient method of calculating the PEVD. This assumption is further discussed in Section 2.2.

Returning our attention to the narrowband MUSIC algorithm, Barabell [28] observed that for a particular array design known as a Uniform Linear Array (ULA) – where the microphones are spaced equidistant along a straight line – the search over the angle parameter that MUSIC leaves the user with can be removed, and the problem may be solved directly by finding the roots of a polynomial. This new algorithm, called root-MUSIC, allowed for greatly reduced computational effort at the cost of having an enforced array geometry. Friedlander [29] then extended the root-MUSIC algorithm to arbitrary array designs by using 'interpolated' arrays; virtual ULAs imagined within the actual physical array. Coventry [30] then extended this to the broadband scenario using a polynomial eigenvalue decomposition algorithm, similar to Alrmah *et al.*'s approach

to broadband MUSIC [26].

2.1.3 The Min-Norm Algorithm

In 1983, Tufts and Kumaresan described another subspace-based algorithm, similar to MUSIC, known as Min-Norm [31]. Min-Norm had a similar advantage to root-MUSIC over traditional MUSIC in that it allowed for direct computation of results – however, similarly to root-MUSIC, the original Min-Norm algorithm was only applicable to ULAs. Even with this restriction, the advantages of this method over root-MUSIC were two-fold. Firstly, in low signal-to-noise ratio environments Min-Norm is a more accurate method of DOA estimation by design; and secondly, the algorithm is designed such that false solutions tend to lie on the unit circle and correct solutions far from it – allowing for easy detection of incorrect estimates.

Li, Vaccaro, and Tufts in 1989 described an extension of Min-Norm to arrays of arbitrary design [32], which brought with it both advantages and disadvantages. Two of the main advantages to this extension are that perturbations to an array may simply be calibrated to and dealt with, and that the end user is as such not confined to the ULA array design. With the increased versatility came the disadvantage of increased complexity – instead of finding the roots of a polynomial, the problem again became a search problem, i.e. we are left with a function of the estimated direction-of-arrival θ to maximise, as is done in MUSIC.

Though the statistical performance of Min-Norm has been analysed showing promising results [33–35], it remains less common in practice than MUSIC. Li and Vaccaro [35] showed that the variance in the DOA estimation error for the Min-Norm algorithm is bounded below by that of MUSIC, and that the variances in the error introduced by polynomial root-finding algorithms which are used in the original Min-Norm algorithm are identical to those introduced by extrema-finding algorithms such as are required for MUSIC.

2.1.4 ESPRIT

In 1985, Roy, Paulraj, and Kailath proposed a new algorithm by the name of Estimation of Signal Parameters via Rotational Invariance Techniques, or ESPRIT [36]. ESPRIT is a subspace-based algorithm similar to MUSIC, with several important improvements over MUSIC: it does not require knowledge of the microphone characteristics; it is less complex from a computational perspective, as it directly calculates the DOA estimates rather than searching for them over some space; it does not require calibration of the array; and it simultaneously estimates the DOAs and the number of sources. ESPRIT's main disadvantage is that it requires a very specific array geometry, where each element in the array is in fact a pair of microphones separated by some constant shift in space – for example, if one microphone was 12cm North-East of the other in one microphone pair, then every other pair must have the same separation.

ESPRIT performs similarly to, though slightly worse than, MUSIC [37–39], however if the physical design of the array is free to meet the specifications required for the algorithm and runtime is an important factor in the use-case, then it is still a good choice. ESPRIT was used by Tao alongside time-reversal microwave imaging to locate and image breast tumours [40], and by Sun *et al.* for imaging road surface layers via

ground penetrating RADAR [41]. ESPRIT has been extended for use in uniform circular arrays (UCA-ESPRIT) [42], as well as specialised for use in uniform rectangular arrays [43].

2.1.5 Beamforming

Beamforming is another related algorithm, which is used to focus the constructive interference of multiple transmitters to a localised point in space – commonly used, for example, in radar & sonar systems [44], 5G technology [45–47], and Wi-Fi networks [48]. Beamforming may also be used for "directionally focused recording", where it is used to attract the focus of a microphone array to a particular point in space – for example, to focus recording on the speaker at a conference or in a videoconferencing environment [49–52].

Though beamforming has often been applied to the situation of unknown DOA [53, 54] this is often an extension of the 'focused recording' problem, rather than solely to provide a location estimate for the source, and it is common to find papers which refer to 'unknown DOA' in the context of enhancing a component of interest from a signal without knowing from which direction that component came. Nevertheless, Krishnaveni *et al.* surveyed the use of beamforming as it pertains to DOA estimation [55]; this consists of retrospectively applying beamformers to observed data, and finding which direction maximised the sum of the beamformers' output.

2.1.6 Machine Learning

The recent explosion in popularity of artificial neural networks (ANNs) has not left the field of acoustic source localisation untouched. Though the physical interpretation in the situation of DOA estimation is clear, and we may directly establish deterministic or statistical estimates for the DOA, several groups have applied ANNs to the problem of acoustic source localisation. Agatonovic *et al.* [56] designed a system in which a multilayer perceptron performed a crude discretisation of the search space, and a radial basis function neural network was used to get a finer approximation of the DOA from within the reduced search space. The results were comparable to two-dimensional MUSIC, but with a considerable increase in speed owing to the computationally inexpensive nature of the ANN feed-forward (prediction) process. This was offset somewhat by the large amount of time taken to train the model (though this is typically a task completed during development and not at the time of use). Unlersen and Yaldiz [57] fed the covariance matrix of the data from a five element ULA into a simple two-layer neural network with a hardmax 19-node output layer. They only trained the model on sources in front of the array, and obtained an output resolution of 10 degrees. The results appear promising, however no comparisons were made with the predictions from a well-established method such as MUSIC or ESPRIT. Yang, Chan, and Chang [58] designed a complex-valued neural network with limited results, which were again not compared to those from any other algorithms. Adavanne, Politis, and Virtanen [59] designed a deep neural network (referred to as DOAnet), consisting of two key components: a convolutional neural network (CNN), stacked on top of a recurrent neural network (RNN). This structure is known as a convolutional recurrent neural network (CRNN). The CRNN was trained directly on a portion of the Fourier Transform of the data.

DOAnet was incredibly promising, being shown to perform considerably better than MUSIC in most scenarios in which it was tested. Chakrabarty and Habets [60] opted for a simpler CNN which they trained on white noise, which they showed to generalise to speech signals. The trained CNN was robust to both noise and small perturbations in array geometry, however did not allow for localisation of multiple simultaneous sources.

2.2 History of the Polynomial Eigenvalue Decomposition

It was mentioned that the adaptations to the MUSIC algorithm using polynomial matrices had significant benefits in accuracy over the frequency-binned option, assuming one has an efficient method of calculating the Polynomial Eigenvalue Decomposition (PEVD). This is not a trivial matter; in fact, particularly in recent years, there has been extensive work on obtaining computationally tractable PEVD algorithms, as well as in the underlying theory of polynomial matrices.

The road to a complete theory of polynomial matrix algebra, including the PEVD, begins at least as early as 1993 with Vaidyanathan [61], who investigated the idea of paraunitary matrices (the polynomial matrix equivalent of unitary matrices), and constructing them with a series of delays interspersed by rotations. In 1996, Russell Lambert gave explicit mention of "FIR Matrix Algebra" with respect to the separation of multipath mixtures and blind decorrelation [62]. FIR stands for "finite impulse response", a term which implies that the polynomial representation of a filter has finitely many non-zero coefficients. Vaidyanathan, in 1998, published a paper on the theory of optimal subband coders [63], and introduced the concept of strong (or total) decorrelation. This work would later be used to show the tendency of a particular PEVD algorithm, known as SBR2, to impose a strict ordering on the returned eigenvalues; an attribute known as spectral majorisation. Gao and Nguyen [64] described in 2001 the factorisation of matrices which contain a specific type of polynomial – ones that describe a particular type of filter known as a *causal* FIR filter. In 2003, Do-Hong and Russer [65] described a way to split an aspect of the calculation for spatially-close wideband sources into two separate components via Bessel functions – one component modelling source direction, and one modelling source separation.

The first method for numerical calculation of the PEVD, known as the Sequential Best Rotation algorithm with Second Order Statistics (or simply SBR2), was introduced as a prototype by Baxter and McWhirter in 2004 [66]. SBR2 is an iterative algorithm which effectively treats the polynomial matrix as a stack of scalar matrices: one matrix corresponding to each polynomial degree. At each iteration, some off-diagonal energy is shifted to the constant coefficient matrix, and then that shifted energy is subsequently rotated down onto the diagonal. In 2007, SBR2 was properly introduced by McWhirter, Baxter, Cooper, Redif and Foster [25], alongside a proof of convergence and demonstration of its tendency for spectral majorisation. In 2010, Andre Tkacenko released an algorithm similar to SBR2 which didn't require full knowledge about the matrix to be diagonalised, and could thus be used in the frequency domain as well as the time domain [67]. Another advantage of the PEVD algorithm suggested by Tkacenko is that the size of the 'stack' of matrices only increases by one in each iteration, in contrast to SBR2, in which this increase is variable and on average much larger. The same year, Weiss, Millar and Stewart published a paper on the inversion

of parahermitian polynomial matrices [68] (a parahermitian matrix is the polynomial matrix equivalent of a complex hermitian matrix, and appears as the generalisation of the covariance matrix in the setting of polynomial matrices). This paper relies on the existence and uniqueness of the EVD of a parahermitian matrix, which had not yet been proven. Also in 2010, Rasmus Brandt published his thesis entitled "Polynomial Matrix Decompositions" [69], which summarised the methods available at the time.

2011 brought with it the extension of the MUSIC algorithm to the broadband scenario via the use of polynomial matrices, in a paper published by Alrmah, Weiss, and Lambotharan [26]. With this came two versions of polynomial MUSIC, known as polynomial spatial MUSIC (PS-MUSIC) and polynomial spatio-spectral MUSIC (PSS-MUSIC). The same year, Redif, Weiss, and McWhirter published an extension of the algorithm proposed by Tkacenko based on Householder transformations, which could handle arbitrary-degree FIR paraunitary matrices. Though broadband MUSIC via polynomial matrices was now established, one aspect of the calculation known as the steering vector was difficult to produce accurately. Alrmah, Weiss, and McWhirter saw to this issue in their 2013 paper "Implementation of Broadband Steering Vectors for Broadband Angle-Of-Arrival Estimation" [70]. This year, other common scalar matrix techniques were posed in the polynomial framework, including comparing AF-MUSIC to both PS-MUSIC and PSS-MUSIC, and showing their equivalence in certain situations [27]. Another method, known as the parametrised spatial covariance method, was later added to the comparison [71]. Tohidian, Amindavor, and Reza [72] introduced another way of obtaining the PEVD through the two-dimensional Discrete Fourier Transform (DFT) and a point-wise calculation around the unit circle.

In 2014, the Sequential Matrix Diagonalisation (SMD) algorithm for calculation of the PEVD was introduced by Redif, Weiss, and McWhirter [73]. This is similar to SBR2, but delays more off-diagonal energy per iteration, and diagonalises the entire constant coefficient matrix, instead of only rotating a small amount of energy onto the diagonal. That year, the same authors alongside Corr, Thompson, and Proudler, published an improved version of SMD known as Multiple-Shift Maximum-Element SMD (MSME-SMD) [74], which makes multiple shifts per iteration and has a different, restricted, search phase as compared to regular SMD. Alrmah, Corr, Alzin, Thompson, and Weiss compared use of MSME-SMD and SBR2 in the MUSIC algorithm, and showed MSME-SMD produced more accurate DOA estimates [75]. To avoid the computational cost of a full diagonalisation of the constant coefficient matrix via an EVD in SMD algorithms, Corr, Thompson, Weiss, McWhirter, and Proudler proposed the use of a series of Givens rotations performed in a particular order to minimise residual off-diagonal elements [76].

The following year, the same "multiple shift" idea from MSME-SMD was applied to SBR2 by Wang, McWhirter, Corr, and Weiss, to create the MS-SBR2 algorithm [77], which was published alongside a proof of convergence. In general, the memory required by the SBR2 algorithm (or SMD, or any variant of these two) grows rapidly at each iteration of the process, and Corr, Thompson, Weiss, McWhirter, and Proudler proposed a way of keeping these memory requirements low for multiple-shift variants by utilising the commutativity of delay operations [78]. Also in 2015, Corr, Thompson, Weiss, Proudler, and McWhirter proposed a method of reducing the complexity of the initial search step in MSME-SMD, leading to Reduced Search Space MSME-SMD (RS-MSME-SMD) [79].

Several papers dealing with memory and complexity reduction for PEVD algorithms were released in 2016. Wang, McWhirter, Corr and Weiss adapted the delay strategy of the MS-SBR2 algorithm to create order-controlled MS-SBR2 (OC-MS-SBR2) [80]. Coutts et al. proposed utilising the symmetry present in a parahermitian polynomial matrix to reduce the storage requirements by a factor of two, and illustrated how to perform delays in this reduced representation [81]. The same authors also demonstrated how to reduce computational complexity in algorithms requiring a cyclic sweep of Jacobi rotations, and proposed thresholding the delayed elements in the constant coefficient matrix to decide whether it was worth expending the computational effort required to rotate them [82].

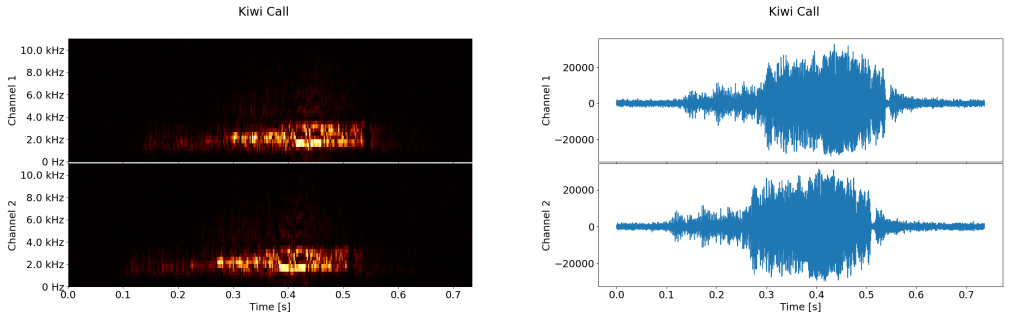
In 2017, Dr Jamie Corr published his PhD thesis entitled *Advanced Algorithms for Polynomial Eigenvalue Decomposition* [83]. This was a comprehensive account of the available algorithms at the time, including SBR2, SMD, ME-SMD, MSME-SMD, and MS-SBR2; as well as discussion of truncation methods, search-space restriction, and complexity reduction. Also included were discussions of applications of the PEVD, including polynomial MUSIC algorithms, the generalised polynomial EVD, and parahermitian matrix inversion. Dr Corr's thesis is an excellent reference for this material, and includes visualisations which help build further intuition about the algorithms discussed. The same year, Redif, Weiss, and McWhirter discussed the relevance of the PEVD to broadband blind signal separation; noting its uses in underwater acoustics and suppression of unwanted radar back-scattering [84]. Coutts, Thompson, Weiss, and Proudler published a divide-and-conquer variant of SMD named DC-SMD this year [85], and analysed its performance on data from large broadband sensor arrays [86]. The same authors also compared SBR2, SMD, and DC-SMD for use in the SSP-MUSIC algorithm on a 12-element sensor array, with DC-SMD showing the best performance [87], and published a comparison between iterative and DFT-based PEVD algorithms for different scenarios [88]. Iterative methods were shown to be a good choice for diagonalising low-dimensional matrices, and DFT-based methods were shown to perform well in scenarios with a low initial lag parameter.

The following year, Coutts, Thompson, Pestana, Proudler, and Weiss extended the most common DFT-based method for calculation of the PEVD to not require *a priori* information of the order of the eigenvalues/eigenvectors to be calculated. The same paper introduced a new metric for the smoothness of a function on the unit circle, and used this to enforce smoothness of the eigenvectors [89]. Despite all the effort going into calculating the PEVD of a parahermitian polynomial matrix, the existence and uniqueness of such a decomposition was not proven until the 2018 paper *On the Existence and Uniqueness of the Eigenvalue Decomposition of a Parahermitian Matrix* by Weiss, Pestana, and Proudler [90].

Chapter 3

Generalised Cross-Correlation (GCC)

Consider some real signal $s(t)$ impinging on an array of 2 sensors x_1 and x_2 from some angle of arrival θ. For demonstration, in this chapter we shall use a 22050 Hz recording of the female North Island Brown Kiwi call, simulated as if captured on two microphones 10 meters apart. Figure 3.1 shows two visualisations of the data; in both the spectrogram in 3.1(a), and the raw data plot in 3.1(b), the two channels appear similar, with a visible horizontal (time-axis) shift.



(a) Spectrogram of Kiwi Data

(b) Raw Plot of Kiwi Data

Figure 3.1: Simulated recording of a North Island Brown Kiwi call with a 642.67 sample delay between channels, caused by the signal travelling 10 meters at 343.1 meters per second, while being sampled at 22050 samples per second.

The data captured at the two sensors may be modelled by:

$$x_1(t) = s(t) + n_1(t) \tag{3.1}$$
$$x_2(t) = \alpha s(t - D) + n_2(t), \tag{3.2}$$

where D is the relative delay caused by the sound having to traverse the distance between the two microphones, α is an attenuation factor to model the decay in volume as the signal travels between the microphones, and $n_1(t)$ and $n_2(t)$ are independent and identically distributed additive Gaussian noise. As the time delay is relative to the

two microphones, we may without loss of generality assume that the signal is at sensor x_1 at time $t = 0$ with attenuation factor 1, which is why α and D only appear in the equation for $x_1(t)$. Notice that D is governed by the angle of arrival θ, as well as the array geometry. Assuming that the array geometry is known *a priori*, we aim to work backwards and estimate θ through the delay D. But how do we find an estimate of D?

3.1 The Fourier Transform

A tool of particular use in the remainder of this thesis is an integral transform known as the Fourier Transform, which will be introduced here before continuing on to the Generalised Cross-Correlation. The Fourier Transform is a common tool in engineering applications, as it acts to change a function of time into a function of frequency; for example, one may transform a recording of a musical chord to be able to investigate its constituent notes, or transform a set of time-series data to identify different periodic components.

The Fourier Transform of a continuous time function $g(t)$, denoted $\mathscr{F}\{g(t)\}(f)$, is given by the integral

$$\mathscr{F}\{g(t)\}(f) = \int_{-\infty}^{\infty} g(t)e^{-2\pi i f t}dt. \tag{3.3}$$

In practice, such as in the case of working with audio data sampled by a computer, the signal of interest is a function of discrete time ($n \in \mathbb{Z}$) instead of continuous time ($t \in \mathbb{R}$), and one wishes to obtain a discrete function of f. In this case, we may use the Discrete Fourier Transform (or simply DFT) of a discrete signal $x[k]$, given by:

$$X[f_n] = \mathrm{DFT}\{x[k]\} = \sum_{k=0}^{N-1} x[k]e^{-2\pi i \frac{nk}{N}}, \tag{3.4}$$

which turns a series of N samples in time $x[0], ..., x[N-1]$ into a series of N samples in frequency $X[0], ..., X[N-1]$. Here, rather than use the "$\mathscr{F}$" notation, we simply capitalise the variable name to symbolise that it is now a function of discrete frequency rather than discrete time. The inverse of Eq. (3.4) is the Inverse Discrete Fourier Transform (or IDFT), given by

$$x[n] = \mathrm{IDFT}[X[n]] = \frac{1}{N}\sum_{k=0}^{N-1} X[f_n]e^{2\pi i \frac{nk}{N}}. \tag{3.5}$$

For analytic work, it is sometimes desirable to transform a function of discrete time into a function of continuous frequency. This is indeed possible, and is done with the Discrete Time Fourier Transform (DTFT), which is given by:

$$X(f) = \mathrm{DTFT}\{x[k]\} = \sum_{k=-\infty}^{\infty} x[k]e^{-2\pi i f n}. \tag{3.6}$$

or if working in units of ω radians per sample:

$$X(\omega) = \text{DTFT}\{x[k]\} = \sum_{k=-\infty}^{\infty} x[k]e^{-i\omega n}. \tag{3.7}$$

3.2 Cross-Correlation

We have seen in Eq. (2.1) that the time-delays induced by a signal arriving at spatially separated receivers may be used to estimate the direction-of-arrival of a signal, and thus finding some way to estimate these delays seems like a good initial approach to the problem of acoustic source localisation. One method of estimating the time-delay D between two signals is using something known as the *cross-correlation* of the two signals; indeed we will see that the time-delay τ which maximises the cross-correlation provides an estimate of D. The cross-correlation, also known as the sliding dot product, of two real, continuous-time, observed signals $x_1(t)$ and $x_2(t)$, is given by [16]:

$$R_{x_1x_2}(\tau) = \mathbb{E}\left[x_1(t)x_2(t-\tau)\right] \tag{3.8}$$

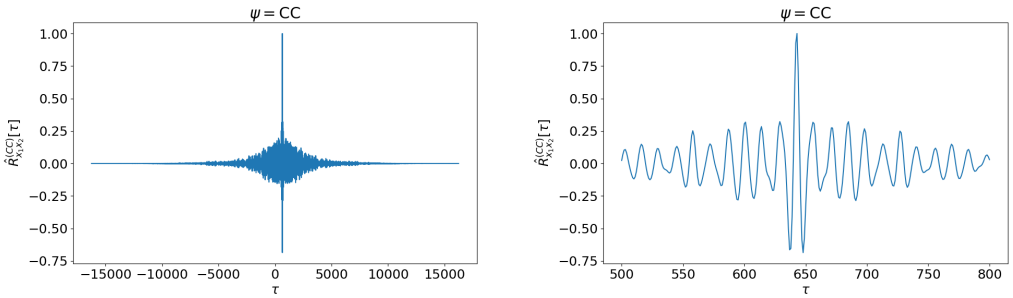
$$= \int_{-\infty}^{\infty} x_1(t)x_2(t-\tau)dt, \tag{3.9}$$

where $\mathbb{E}[x(t)]$ denotes the expected value of $x(t)$ over all t; or for real discrete signals $x_1[n]$ and $x_2[n]$:

$$R_{x_1x_2}[\tau] = \mathbb{E}\left[x_1[n]x_2[n-\tau]\right] \tag{3.10}$$

$$= \sum_{n=-\infty}^{\infty} x_1[n]x_2[n-\tau]. \tag{3.11}$$

An example of the cross-correlation can be seen in Figure 3.2, which shows the cross correlation of the signals shown in Figure 3.1 as a function of the time-delay τ.



(a) Full Time-Scale

(b) Behaviour Around $\tau = 642.67$

Figure 3.2: The cross-correlation of the simulated data as introduced in Figure 3.1. (a) shows the full cross-correlation, and (b) shows a close-up of the behaviour of the cross-correlation around the true delay $D = 642.67$ samples.

Wide-Sense Stationarity

A random process $X(t)$ with mean function $\mu(t)$ and variance function $\sigma^2(t)$ is said to be *wide-sense stationary* (WSS) if its mean, variance, and autocorrelation function do not change over time; i.e. if both $\mu(t) = \mu(0)$ and $\sigma^2(t) = \sigma^2(0)$ for all t, and $\mathbb{E}\{X(t)X^*(t-\tau)\} = \mathbb{E}\{X(0)X^*(-\tau)\}$ for all t.

For discrete wide-sense stationary signals $x_1[n]$ and $x_2[n]$ of length L, we may estimate (3.11) by

$$\begin{aligned} R_{x_1x_2}[\tau] &\approx \hat{R}_{x_1x_2}[\tau] \\ &= \sum_{n=0}^{L} x_1[n]x_2[n-\tau], \end{aligned} \tag{3.12}$$

which is defined for $-L \leq \tau \leq L$ by letting $x_2[n] = 0$ for $n \notin [0, L]$. Where we may estimate $R_{x_1x_2}[\tau]$ as done in Eq. (3.12) by the wide-sense stationarity of $x_1[n]$ and $x_2[n]$.

Consider the two continuous-time signals $f(t)$ and $g(t)$. The cross-correlation of f with g may be visualised as lining the end of g up with the start of f, i.e. letting $\tau = -L$, and taking the dot product of the pieces which overlap. Then, slide g along f, taking the dot product of the overlapping pieces every step of the way. The function of τ given by the dot product of the overlapping parts is the cross-correlation of f with g; hence the alternate name "sliding dot product".

3.3 Generalised Cross-Correlation

For the model given in (3.1), as we have assumed the signal and noise are uncorrelated (i.e. their cross-correlation is 0), (3.8) gives

$$R_{x_1x_2}(\tau) = \alpha R_{ss}(\tau - D) + R_{n_1n_2}(\tau) \tag{3.13}$$

$$= \alpha R_{ss}(\tau) \circledast \delta(t - D), \tag{3.14}$$

where δ denotes the Dirac delta function, and (3.13) to (3.14) is by the assumption that the noise channels $n_1(t)$ and $n_2(t)$ are uncorrelated; both with each other, and with any sources present. The Fourier Transform of (3.14) is what is known as the cross-power spectrum, $G_{x_1x_2}(f)$, of $x_1(t)$ and $x_2(t)$, and is given by:

$$G_{x_1x_2}(f) = \alpha G_{ss}(f)e^{-2\pi i f D}. \tag{3.15}$$

Thus, the cross-correlation of $x_1(t)$ and $x_2(t)$ is a Dirac delta function which has been convolved by the inverse Fourier Transform of $G_{ss}(\omega)$, the autospectrum[1] of the source signal $s(t)$ [16].

In 1976, Knapp and Carter described a method to extend the cross-correlation to incorporate the pre-filtering of the input signals as a weighting in the frequency domain [16]. To see why this is advantageous, consider the convolution theorem:

[1]The *autospectrum* is to the cross-power spectrum what the autocorrelation is to the cross-correlation; i.e. the autospectrum of a signal is the cross-power spectrum of the signal and itself.

Convolution Theorem

The Convolution Theorem [91] for two functions $f(x)$ and $g(x)$ in the L^p space $L^1(\mathbb{R})$ (i.e. for absolutely integrable real-valued functions f and g) states:

$$f(x) \circledast g(x) = \mathscr{F}^{-1}\left(\mathscr{F}\left\{f(x)\right\} \cdot \mathscr{F}\left\{g(x)\right\}\right), \tag{3.16}$$

where $f(x) \circledast g(x)$ denotes the circular convolution of $f(x)$ and $g(x)$.

The convolution theorem allows us to perform the correlation by taking the Fourier transform of $f(x)$ and $g(x)$, conjugating the transform of $g(x)$, multiplying the results point-wise, and taking the inverse Fourier transform of the product. This gives a computational complexity of $\mathcal{O}(n \log n)$ thanks to the Fast Fourier Transform (FFT) [14], as compared to cross-correlation via traditional sliding convolution, which is $\mathcal{O}(n^2)$ in complexity.

Another advantage of Eq. (3.16), and the basis of the Generalised Cross-Correlation (GCC) algorithm, is as follows. Consider the situation in which the autocorrelation of $s(t)$, R_{ss}, is a Dirac delta function. In this case, (3.14) reduces to $\delta(t - D)$, providing an easy estimate for D. In general, however, the autocorrelation of a source signal will not be of this form, and so we must modify our received data in some way for this to be the case. As seen in (3.15) and (3.14), the inverse Fourier Transform of a constant term is a Dirac delta function, and thus if we can normalise the autospectrum of $s(t)$ to have unit magnitude at all frequencies, i.e. if we can whiten the signal component of the received data, then we will obtain the desired result. This effect can be approximated by with a procedure known as filtering, whereby the frequency spectrum of the data is shaped in the desired way by a purpose-built filter. We take further advantage of the Convolution Theorem, and apply the filters by multiplying the Fourier Transform of $x_1(t)$ and $x_2(t)$ by the Fourier Transform of the appropriate filters h_1 and h_2, H_1 and H_2 respectively. As multiplication is associative in $\mathbb{C}$, we may combine the transformed filters into one general frequency weighting $\psi_g(f) = H_1 \cdot H_2$ to apply to the cross-power spectrum of the observed data.

This leads to a 'weighted' cross-correlation that Knapp and Carter called the Generalised Cross-Correlation, and is given by

$$\begin{aligned} \hat{R}^{(g)}_{x_1x_2}(\tau) &= \int_{-\infty}^{\infty} H_1(f) H_2(f) \widehat{G}_{x_1x_2}(f) e^{2\pi i f \tau} df \\ &= \int_{-\infty}^{\infty} \psi_g(f) G_{x_1x_2}(f) e^{2\pi i f \tau} df, \end{aligned} \tag{3.17}$$

estimated in practice, where $X_i[f_n]$ denotes the n^{th} frequency component of the discrete Fourier transform of $x_i(t)$, by

$$\hat{R}^{(g)}_{x_1x_2}[\tau_n] = IDFT\left[\psi_g[f_n] \cdot X_1[f_n] \cdot \overline{X_2[f_n]}\right], \tag{3.18}$$

using (3.4), (3.5), the discrete version of the convolution theorem in (3.16) [92], and

the approximation

$$G_{x_1x_2}[f_n] \approx \widehat{G}_{x_1x_2}[f_n] \tag{3.19}$$
$$= X_1[f_n] \cdot \overline{X_2[f_n]}. \tag{3.20}$$

In (3.18), we have used square brackets and the notation τ_n to illustrate that the calculation as shown is a function of integer time delays and is as such not defined for non-integer τ. To get around this restriction in practice, we may use some form of interpolation (such as cubic spline or linear interpolation) to approximate the GCC at non-integer time delays for discrete data.

This is a very versatile approach to the direction of arrival estimation problem; different processors can be designed to suit different environments. Note that for $\psi_g = 1$ $(= \psi_{\text{CC}})$, the GCC is simply the cross-correlation between $x_1(t)$ and $x_2(t)$.

3.3.1 GCC Processors

The different weightings ψ in the GCC procedure are commonly known as 'processors'. Several have been described and reviewed by many, including Knapp and Carter [16], Marinescu *et al.* [93] and Ritu and Dhull [20]. We discuss some here.

Phase Transform

The Phase Transform (PHAT) [16, 19] was developed as an *ad hoc* method to whiten the signals and thus make the cross-correlation a delta function for easy peak detection. The PHAT weighting is given by

$$\psi_{\text{PHAT}}(f) = \frac{1}{|G_{x_1x_2}(f)|}, \tag{3.21}$$

which, with $|G_{x_1x_2}(f)| = \alpha G_{ss}(f)$ by our assumptions about the noise, gives

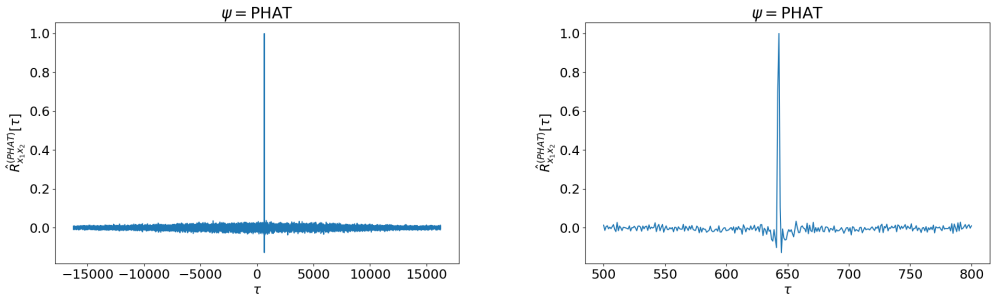
$$\frac{G_{x_1x_2}(f)}{|G_{x_1x_2}(f)|} = \frac{\alpha G_{ss}(f) e^{-2\pi i f D}}{\alpha G_{ss}(f)} = e^{-i2\pi f D}, \tag{3.22}$$

and thus combined with (3.17),

$$\hat{R}^{(\text{PHAT})}_{x_1x_2}(\tau) = \int_{-\infty}^{\infty} \frac{G_{x_1x_2}(f)}{|G_{x_1x_2}(f)|} e^{2\pi i f \tau} df \tag{3.23}$$
$$= \int_{-\infty}^{\infty} e^{-2\pi i f D} e^{2\pi i f \tau} df \tag{3.24}$$
$$= \int_{-\infty}^{\infty} e^{2\pi i f (\tau - D)} df \tag{3.25}$$
$$= \delta(\tau - D) \tag{3.26}$$
$$\tag{3.27}$$

The PHAT processor provides some potential issues; frequencies where the source is not present, or has low power, are weighted just as high as all other frequency components in the data. Thus for a narrowband signal (a signal for which the total

power is very localised in frequency, for example a 440 Hz sine wave) this weighting is potentially not a wise choice, but for a broadband signal (a broadband signal is the opposite of a narrowband signal; the total power is very spread out in frequency, for example white noise) such as human speech, it may perform better.



(a) Full Time-Scale

(b) Behaviour Around $\tau = 642.67$

Figure 3.3: The PHAT-weighted cross-correlation of the data as introduced in Figure 3.1. (a) shows the full GCC output, and (b) shows a close-up of the behaviour of the GCC output around the true delay $D = 642.67$ samples.

Figure 3.3 shows the output of the GCC algorithm with the PHAT weighting for the kiwi call data as introduced in Figure 3.1. Comparing Figure 3.2(b) and Figure 3.3(b) we can see how the attempted whitening of the signal acts to flatten out the GCC output either side of the true time-delay D.

A few adaptations of the PHAT processor exist, which do not attempt to whiten the signal entirely as (3.21) does; the rationale being that the signal components with high magnitude are more likely to have come from the source than the noise, and so should still contribute more strongly to the GCC calculation.

The first such adaptation is the ρ-PHAT processor [94], as given by

$$\psi_{\rho\text{-PHAT}}(f) = \frac{1}{|G_{x_1x_2}(f)|^{\rho}}. \tag{3.28}$$

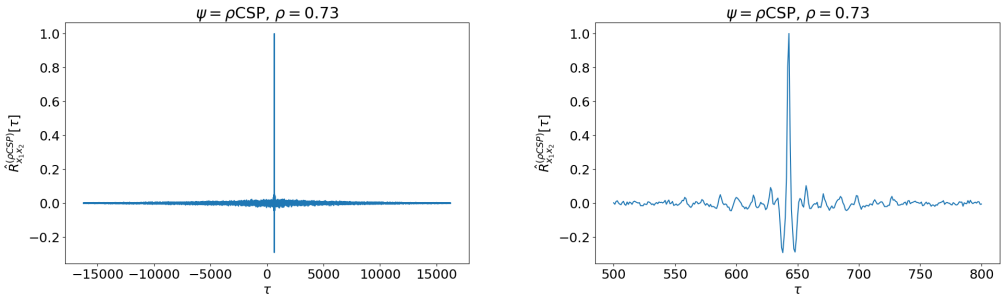
Notice that for $\rho = 0$,

$$\begin{aligned}\psi_{0\text{-PHAT}}(f) &= \frac{1}{|G_{x_1x_2}(f)|^{0}} \\ &= 1 \\ &= \psi_{CC}(f),\end{aligned}$$

which gives the normal cross-correlation, and for $\rho = 1$:

$$\begin{aligned}\psi_{1\text{-PHAT}}(f) &= \frac{1}{|G_{x_1x_2}(f)|^{1}} \\ &= \frac{1}{|G_{x_1x_2}(f)|} \\ &= \psi_{\text{PHAT}}(f).\end{aligned}$$

Thus by choosing $0 \leq \rho \leq 1$, the user has control over the level of whitening of the signal in the GCC calculation. The optimal value of ρ can be established experimentally, but a common choice for a default value in audio applications appears to be $\rho \approx 0.73$ [93, 94].



(a) Full Time-Scale

(b) Behaviour Around $\tau = 642.67$

Figure 3.4: The ρ-PHAT (ρ= 0.73) weighted cross-correlation of the data as introduced in Figure 3.1. (a) shows the full GCC output, and (b) shows a close-up of the behaviour of the GCC output around the true delay $D = 642.67$ samples.

An example output for the GCC algorithm with ρ-PHAT weighting (ρ= 0.73) for the kiwi data is shown in Figure 3.4.

Roth Impulse Response

The Roth Processor [15] aims to estimate the optimum linear Wiener-Hopf filter

$$H_m(f) = \frac{G_{x_1x_2}(f)}{G_{x_1x_1}(f)}, \tag{3.29}$$

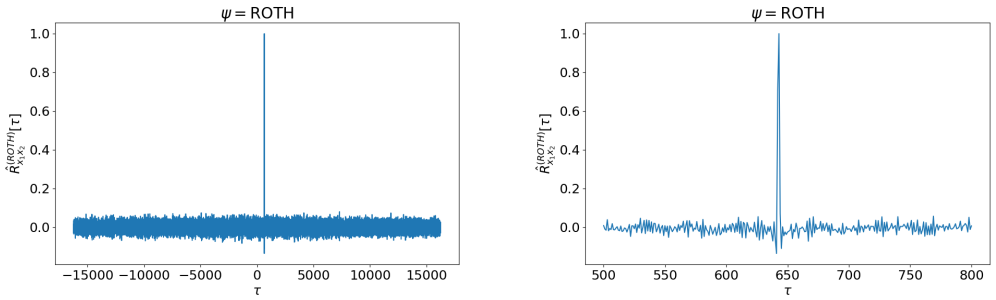
which best approximates the mapping of $x_1(t)$ to $x_2(t)$ [16]. The Roth processor uses the weighting

$$\psi_{\text{ROTH}}(f) = \frac{1}{G_{x_1x_1}(f)}. \tag{3.30}$$

Combining (3.17) and (3.30) give

$$\hat{R}^{(\text{ROTH})}_{x_1x_2}(\tau) = \int_{-\infty}^{\infty} \frac{\widehat{G}_{x_1x_2}(f)}{G_{x_1x_1}(f)} e^{2\pi i f \tau} df \tag{3.31}$$

and we can see the approximation of 3.29. The Roth Processor weights low the frequencies at which $G_{n_1n_1}(f)$ is large, with the idea that the estimate $\widehat{G}_{x_1x_2}(f)$ of the cross-power spectra of x_1 and x_2 is more likely to be erroneous at these frequencies.



(a) Full Time-Scale

(b) Behaviour Around $\tau = 642.67$

Figure 3.5: The ROTH-weighted cross-correlation of the data as introduced in Figure 3.1. (a) shows the full GCC output, and (b) shows a close-up of the behaviour of the GCC output around the true delay $D = 642.67$ samples.

An example output of the ROTH-weighted GCC algorithm for the kiwi data is shown in Figure 3.5.

Smoothed Coherence Transform

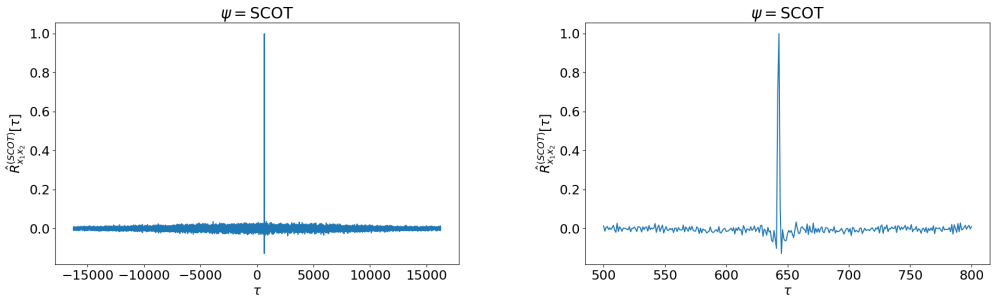
The Smoothed Coherence Transform, or SCOT [19], aims to perform a similar weighting to the PHAT in order to tend $R_{x_1x_2}(\tau)$ towards a delta function. The difference, however, is that the SCOT aims to mitigate the errors due to low signal power by weighting by an estimate of how 'likely' it is that the signals at a particular frequency are related by a linear filter, and thus weighting the signal components which do not appear to be related less than components which do appear to be related. The statistic used for this weighting is called the *coherence* between x_1 and x_2, denoted $\gamma_{12}(f)$, and is low when $x_2(t)$ is not likely to be the result of applying a linear filter to $x_1(t)$. The SCOT weighting is given by

$$\psi_{\text{SCOT}}(f) = \frac{1}{\sqrt{G_{x_1x_1}(f)G_{x_2x_2}(f)}}, \tag{3.32}$$

in which can be seen an estimate for $\gamma_{12}(f)$:

$$\hat{\gamma}_{1,2}(f) = \frac{|G_{x_1x_2}(f)|^2}{G_{x_1x_1}(f)G_{x_2x_2}(f)}. \tag{3.33}$$

The SCOT processor considers both channels, and will broaden the peak of cross-correlation function when compared to the ROTH processor.



(a) Full Time-Scale

(b) Behaviour Around $\tau = 642.67$

Figure 3.6: The ROTH-weighted cross-correlation of the data as introduced in Figure 3.1. (a) shows the full GCC output, and (b) shows a close-up of the behaviour of the GCC output around the true delay $D = 642.67$ samples.

An example output of the SCOT-weighted GCC algorithm for the kiwi data is shown in Figure 3.6.

Other Processors

Several other processors also exist which have not been implemented or investigated in this thesis. These include the Wiener Processor:

$$\psi_{\text{Wiener}}(f) = |\gamma_{12}(f)|^2 , \tag{3.34}$$

proposed by Hero and Schwartz [95]; the HB processor

$$\psi_{\text{HB}}(f) = \frac{|G_{x_1x_2}(f)|}{G_{x_1x_1}(f) \cdot G_{x_2x_2}(f)}, \tag{3.35}$$

described by Hassan and Boucher [96]; and a maximum likelihood processor known as the Hannan and Thomson processor [97]:

$$\psi_{\text{HT}}(f) = \frac{1}{|G_{x_1x_2}(f)|} \cdot \frac{|\gamma_{12}(f)|^2}{[1 - |\gamma_{12}(f)|^2]}. \tag{3.36}$$

For more information about these processors, Marinescu *et al.* [93] provide an overview of all processors described above.

This concludes our discussion of GCC; the first of the two algorithms for acoustic source localisation to be discussed in this thesis. Before moving on to the second – the Multiple Signal Classification (MUSIC) algorithm – we will first introduce the idea of polynomial matrices, as they will be of use in the application of the MUSIC algorithm to broadband signals.

Chapter 4

Polynomial Matrices

The MUSIC algorithm, to be discussed in Chapter 5, is an algorithm capable of superior precision to the GCC algorithm, with the significant drawback that it was originally only designed for use with narrowband sources; the data is modelled explicitly as a single complex sinusoid. In the case of working with audio data, true narrowband sources are rare, and thus it would be advantageous to extend MUSIC to the broadband scenario. One such method for this extension is by incorporating a time aspect into the covariance matrix of the data; possible by the use of *polynomial matrices*. In this chapter we will define polynomial matrices, as well as present some results about them pertinent to their use in the MUSIC algorithm.

4.1 Overview

A polynomial matrix R in the variable z, denoted by $R(z)$, is a matrix whose entries are polynomials, for example:

$$R(z) = \begin{bmatrix} 2z^2 - 1 & 3z - 5 + 4iz^{-1} \\ 0 & 6z \end{bmatrix}. \tag{4.1}$$

Equivalently, a polynomial matrix may be thought of as a polynomial with matrix coefficients; e.g. for $R(z)$ as above:

$$R(z) = \begin{bmatrix} 2 & 0 \\ 0 & 0 \end{bmatrix} z^2 + \begin{bmatrix} 0 & 3 \\ 0 & 6 \end{bmatrix} z + \begin{bmatrix} -1 & -5 \\ 0 & 0 \end{bmatrix} + \begin{bmatrix} 0 & 4i \\ 0 & 0 \end{bmatrix} z^{-1}.$$

Assuming that the polynomials populating the matrices satisfy some conditions to be outlined in this chapter, polynomial matrices can be manipulated in much the same ways as scalar matrices, with two notable differences. Firstly, note that the element-wise multiplication involved in matrix multiplication is now *polynomial* multiplication, not scalar multiplication, for example:

$$\begin{bmatrix} 2z^2 - 1 & 3z - 5 + 4iz^{-1} \\ 0 & 6z \end{bmatrix} \begin{bmatrix} z + 3z^{-2} & 2z \\ 0 & z^{-1} \end{bmatrix} = \begin{bmatrix} -z + 6 - 3z^{-2} & -2z + 3 - 5z^{-1} + 4iz^{-2} \\ 0 & 6 \end{bmatrix}.$$

Secondly, we may define an additional operation as follows. We define the *parahermitian transpose* of a polynomial matrix $R(z)$, denoted $R^P(z)$, to be:

$$R^P(z) = R^*(1/z),$$

where $R^*(z)$ denotes the conjugate transpose of matrix $R(z)$, for example for $R(z)$ as given in (4.1);

$$R^P(z) = R^*(1/z) = \begin{bmatrix} 2z^{-2} - 1 & 0 \\ -4iz - 5 + 3z^{-1} & 6z^{-1} \end{bmatrix}. \tag{4.2}$$

This can be viewed as the application of both the complex conjugate transpose and a "time reversal" of the coefficients of z. A *parahermitian matrix* is a polynomial matrix which is equal to its own parahermitian transpose, and a *paraunitary matrix* is a polynomial matrix $U(z)$ such that $U(z)U^P(z) = U^P(z)U(z) = I$.

Some terminology used here has also been borrowed from signal processing, as the polynomials of interest are also used to describe an important object in signal processing known as filters. A filter can be thought of as a set of coefficients used to generate a linear combination of the input signal and time-delayed versions of itself. Letting $z^{-\tau}$ denote a delay of the input signal by τ samples, we see how a polynomial in z can describe a filter. For example, after applying the filter described by

$$H(z) = 0.2z^3 - 0.5 + 0.6z^{-1},$$

the n^{th} sample of the output signal would be the sum of 0.2 times the $(n+3)^{\text{rd}}$ input sample, 0.5 times the n^{th} input sample, and 0.6 times the $(n-1)^{\text{th}}$ input sample. Notice that this particular filter has finitely many non-zero coefficients (three); we call filters with this property *finite impulse response* (FIR) filters. A *causal* filter is one in which the n^{th} term in the result only depends on samples at indices prior to n; these are of particular interest as they may be implemented in real-time (for more information, see any reference text on digital signal processing e.g. [98]). A *stable* filter is one whose series of coefficients $h[n]$ approaches 0 as $n \to \infty$.

While much work has been done utilising polynomial matrices (e.g. [26]), the nature of the polynomials involved, and the resultant algebraic properties of the set from which we populate our matrices is often left as assumed knowledge. We present a foundation for Finite Impulse Response (FIR) matrices as given by R. Lambert in his 1996 thesis, *Multichannel Blind Deconvolution: FIR Matrix Algebra and Separation of Multipath Mixtures* [62], particularly Chapter 3.

Definition 4.1.1. Laurent Polynomial. A Laurent Polynomial over $\mathbb{C}$ is an expression of the form

$$p(z) = \sum_{n=-j}^{k} a[n]z^n, \tag{4.3}$$

with $j, k \in \mathbb{N}$, and $a[n] \in \mathbb{C}\ \forall n$.

Here, there are a finite number of terms with negative powers of z, as well as a finite number of terms with positive powers of z. Laurent Polynomials are commonly found as the impulse responses of Finite Impulse Response (FIR) filters. Note that Laurent polynomials form a ring $\mathcal{R}$ with unity under term-wise addition (with identity

polynomial 0), and polynomial multiplication (with unity polynomial 1), which may be calculated via convolution of the series of coefficients.

It is important to notice, however, that in general there do not exist multiplicative inverses in this ring, e.g. for $p(z) = 1 + z \in \mathcal{R}$,

$$(1+z)^{-1} = \sum_{n=0}^{\infty} (-1)^n z^n, \tag{4.4}$$

which is not an element of the ring. Thus $\mathcal{R}$ does **not** constitute a field - a desired property of $\mathcal{R}$ so that we may construct a vector space over it to transfer familiar techniques from Linear Algebra. Note that $\mathcal{R}$ is an integral domain, as $\mathbb{C}$ is an integral domain. To be able to include inverses, we must expand our definition to accommodate expressions such as Eq. (4.4).

Definition 4.1.2. Laurent Series. A Laurent series over $\mathbb{C}$ is an expression of the form

$$p(z) = \sum_{n=-j}^{\infty} a[n] z^n \tag{4.5}$$

with $j \in \mathbb{N}$, and $a[n] \in \mathbb{C}\ \forall n$.

Notice that this is in fact the field of quotients of the integral domain $\mathcal{R}$ as above, and thus is a field [99]. Here we have only finitely many negative powers of z, but potentially infinitely many non-zero positive powers of z. This now includes elements such as (4.4).

Is this all we require to form our vector space? Recall that an operation of particular interest when dealing with polynomial matrices is the parahermitian transpose of polynomial matrix $A(z)$, $A^P(z) = A^*(1/z)$. Consider the (1×1) polynomial matrix

$$Q(z) = \left[\sum_{n=-j}^{\infty} a[n] z^n \right].$$

We have that

$$Q^P(z) = \left[\sum_{n=-\infty}^{j} b[n] z^n \right],$$

with appropriate $b[n]$. Notice, however, that the lone element in $Q^P(z)$ is **not** an element of $\mathcal{R}$, as there exist infinitely many terms in z with negative exponent. Hence we do not have closure under the parahermitian transpose as it stands, and we must expand our definition of $\mathcal{R}$ further to include such matrices.

Definition 4.1.3. Formal Laurent Series. A *formal* Laurent series over $\mathbb{C}$ is an expression of the form

$$p(z) = \sum_{-\infty}^{\infty} a[n] z^n \tag{4.6}$$

with $a[n] \in \mathbb{C}\ \forall n$.

Including infinitely many terms with both positive and negative exponent such as this gives us closure under the parahermitian transpose, but brings about its own set of problems. Consider two formal Laurent series $p(z) = \sum_{-\infty}^{\infty} a[n]z^n$ and $q(z) = \sum_{-\infty}^{\infty} b[n]z^n$. Multiplying them as we would Laurent Polynomials, for example, we see that the series of coefficients of their product is given by

$$c[n] = \sum_{m=0}^{\infty} a[n+m]b[n-m]$$

and it becomes apparent that we must stipulate some further conditions on the series $a[n]$ and $b[n]$ for the elements of $c[n]$ to be finite.

4.1.1 Description

We restrict our attention to formal Laurent series whose series of coefficients $a[n]$ are absolutely summable, i.e. those for which $\sum_{-\infty}^{\infty} |a[n]|$ exists and is finite.

In this case, we may perform the convolution of two series of coefficients $a[n]$ and $b[n]$ (and hence multiplication of their corresponding formal Laurent Series) by making use of the Discrete Time Fourier Transform (DTFT), and the Convolution Theorem.

Discrete Time Fourier Transform

Recall from (3.6) that the DTFT of an infinite, non-periodic, absolutely summable sequence $x[n]$ is given by:

$$X(\omega) = DTFT\{x[n]\} = \sum_{n=-\infty}^{\infty} x[n]e^{-i\omega n}, \qquad \text{(3.6, in radians/sample)}$$

with inverse transform (IDTFT)

$$x[n] = IDTFT[X(\omega)] = \frac{1}{2\pi}\int_{-\pi}^{\pi} X(e^{i\omega})e^{i\omega n}d\omega. \tag{4.7}$$

Convolution Theorem (Discrete Version)

The Convolution Theorem for sequences, where $x[n]*y[n]$ denotes the linear convolution of $x[n]$ and $y[n]$, states that:

$$x[n] * y[n] = IDTFT[DTFT\{x[n]\} \cdot DTFT\{y[n]\}]. \tag{4.8}$$

This is the discrete version of the convolution theorem in (3.16).

4.1.2 Implications

Under our assumption of absolute summability, we are now able to: add, subtract, multiply, and divide formal Laurent series, and thus we have a field. Our field is also closed under time inversion, an important property to ensure that we may make use of the parahermitian transpose. Provided that absolute summability is a reasonable assumption (this will be addressed later), we can now construct a vector space over the

field and put its elements inside matrices, and generalise results from traditional linear algebra to our new space.

4.2 Existence and Uniqueness of Parahermitian Matrix EVDs

This partial proof of the existence and uniqueness of parahermitian matrix eigenvalue decompositions comes from the Weiss *et al.* paper *On the Existence and Uniqueness of the Eigenvalue Decomposition of a Parahermitian Matrix* [90]. Not all cases detailed in the original paper will be covered here (they do not all apply to the direction-of-arrival estimation problem - in particular, we expect by design to have fewer sources than we have sensors and thus have noise-space eigenvalues of multiplicity greater than one). Before we proceed, we must define the *z-transform* of a series $x[n]$.

Definition 4.2.1. The z-transform. Given a series $x[n]$, we define its z-transform $X(z)$ to be

$$X(z) = \sum_{n=-\infty}^{\infty} x[n]z^{-n}. \tag{4.9}$$

Notice that for z restricted to the unit circle (i.e. for $z = e^{i\omega}, 0 \leq \omega \leq 2\pi$), we have:

$$X(\omega) = \sum_{n=-\infty}^{\infty} x[n]e^{-i\omega n} = \text{DTFT}\left\{x[n]\right\}.$$

Note that when working with polynomial matrices in software, one is in fact manipulating the inverse z-transform of the polynomial; the series of coefficients of the polynomial. This is indeed the natural way to store and work with polynomials in software.

4.2.1 Analyticity

Consider the $M \times M$ parahermitian space-time covariance matrix, $R[\tau]$, of M-channel data x, defined by:

$$R[\tau] = \mathbb{E}\left\{x[n]x^*[n-\tau]\right\}, \tag{4.10}$$

where $\mathbb{E}$ denotes the expectation operator. This can be visualised as a 'stack' of covariance matrices - each layer in the stack corresponding to a different time-delay. Assuming the power spectral density (PSD) of each of the $L < M$ sources present is generated by a stable and causal innovation filter $F_l(z)$, $l = 1, 2, \cdots, L$, and if the (m, l) entry $H_{m,l}(z)$ of $M \times L$ matrix $H(z)$ is the (causal and stable) transfer function between the l^{th} source and the m^{th} sensor, then the z-transform of (4.10) becomes

$$R(z) = H(z)F(z)F^P(z)H^P(z) \tag{4.11}$$

$$= H(z)\begin{bmatrix} S_1(z) & & \\ & \ddots & \\ & & S_L(z) \end{bmatrix} H^P(z), \tag{4.12}$$

where $F(z) = \text{diag}\{F_1(z), F_1(z), \ldots, F_L(z)\}$, and $S_l(z) = F_l(z)F_l^P(z)$ is the PSD of the l^{th} source [90].

An innovation filter is a filter used to generate the signal of interest from white noise input. For a signal to be produced in this way, it must be that the signal under consideration satisfies the Paley-Wiener condition [100]:

$$\int_{-\infty}^{\infty} \frac{|\log F_i(e^{i\omega})|}{1+\omega^2} d\omega < \infty, \tag{4.13}$$

for $\omega \in [0, 2\pi)$, which implies that the signal must not be bandlimited [101].

By the assumption of stability of each of their respective entries, the maximum modulus of any pole of $H(z)$ or $F(z)$, ρ, lies within the open interval $(0, 1)$, and thus the region of convergence (ROC) for $H(z)F(z)$ is $|z| > \rho$. Similarly, the maximum modulus of any pole of $H^P(z)$ or $F^P(z)$ is ρ^{-1}, and thus the product $F^P(z)H^P(z)$ has ROC $|z| < \rho^{-1}$. Hence $R(z) = H(z)F(z)F^P(z)H^P(z)$ has ROC $\rho < |z| < \rho^{-1}$ and, importantly, is thus analytic on an annulus containing the unit circle. Analyticity on the unit circle implies absolute convergence of the series of coefficients there, i.e. for space-time covariance matrices, the assumption of absolute summability from 4.1.1 is appropriate. Note also that any finite polynomial has an absolutely summable series of coefficients and is thus analytic, and so the assumption is also valid for polynomials of this type. For brevity we refer to a polynomial matrix comprised of analytic polynomials as an analytic matrix. The product of two analytic matrices is also analytic.

4.2.2 EVD on the Unit Circle

As the inverse z-transform

$$x[n] = \frac{1}{2\pi i} \oint_C X(z) z^{n-1} dz \tag{4.14}$$

requires evaluation on a closed path C around the origin, and we have that the matrix of Laurent polynomials and/or rational functions $R(z)$ is analytic on the unit circle, the equivalent (on the unit circle) inverse Discrete Time Fourier Transform

$$x[n] = \frac{1}{2\pi} \int_{-\pi}^{\pi} X(e^{i\omega}) e^{i\omega n} d\omega \tag{4.15}$$

takes us from $R(e^{i\omega})$ to the time domain. We may then take the z-transform of the resultant time-domain function to get back to an $R(z)$ consisting of only Laurent polynomials, which we know is analytic. Notice then that $R(z)$ is completely characterised by $R(e^{i\omega})$.

Now, for each value of $\omega \in [0, 2\pi]$, we may perform a regular scalar eigenvalue decomposition

$$R(e^{i\omega}) = Q(e^{i\omega})\Lambda(e^{i\omega})Q^H(e^{i\omega}), \tag{4.16}$$

where $\Lambda(e^{i\omega})$ is the diagonal matrix of eigenvalues of $R(e^{i\omega})$, $Q(e^{i\omega})$ is the matrix whose columns are the eigenvectors of $R(e^{i\omega})$ (the modal matrix of $R(e^{i\omega})$), and with existence and uniqueness coming from the usual scalar matrix complex positive-definite case. All that remains is to show the eigenvalues and vectors vary smoothly with ω.

4.2.3 Continuity of Eigenvalues

The continuity of the eigenvalues of $R(e^{i\omega})$ comes as a corollary to a theorem in matrix perturbation theory known as the Wielandt-Hoffman Theorem.

Theorem (Wielandt-Hoffman). *Let A and M be $(n \times n)$ Hermitian matrices, and let α_i and μ_i $(i = 1, ..., n)$ be the eigenvalues of A and M, respectively, with each set arranged in non-increasing order.*

Then

$$\sum_{j=1}^{n}(\alpha_j - \mu_j)^2 \leq ||A - M||_F^2,$$

where $||M||_F$ denotes the Frobenius norm of matrix M. Assuming the eigenvalues $\lambda_i(e^{i\omega})$ of $R(e^{i\omega})$ are in non-increasing order of magnitude and letting $A = R(e^{i\omega})$ and $M = R(e^{i(\omega-\Delta\omega)})$ in the above statement, the Wielandt-Hoffman Theorem [102, 103] gives

$$\sum_i |\lambda_i(e^{i\omega}) - \lambda_i(e^{i(\omega+\Delta\omega)})| \leq ||R(e^{i\omega}) - R(e^{i(\omega-\Delta\omega)})||_F. \tag{4.17}$$

By assumption, $R(z)$ is analytic and thus continuous, and so the right-hand side of (4.17) approaches 0 as $\Delta\omega \to 0$. This then implies

$$\lim_{\Delta\omega\to 0}(\sum_i |\lambda_i(e^{i\omega}) - \lambda_i(e^{i(\omega+\Delta\omega)})|) = 0, \tag{4.18}$$

i.e. the eigenvalues are continuous in ω. Analyticity of the eigenvalues for $R(z)$ on an annulus containing the unit circle is given by the spectral theorem.

4.2.4 Existence of Parahermitian Polynomial Matrix Inverses

A parahermitian polynomial matrix $R(z)$, by Section 4.2.2, admits an eigenvalue decomposition

$$R(z) = Q(z)\Lambda(z)Q^P(z) \tag{4.19}$$

with paraunitary $Q(z)$ (i.e. $Q^P(z)Q(z) = Q(z)Q^P(z) = I$), and diagonal matrix $\Lambda(z)$ containing eigenvalue polynomials $\lambda_1(z), \lambda_2(z), \ldots, \lambda_L(z)$. Consider now the matrix

$$S(z) = Q(z)\Lambda^{-1}(z)Q^P(z), \tag{4.20}$$

where $Q(z)$ is the same as in $R(z)$, and

$$\Lambda^{-1}(z) = \begin{bmatrix} (\lambda_1(z))^{-1} & & \\ & \ddots & \\ & & (\lambda_L(z))^{-1} \end{bmatrix}, \tag{4.21}$$

where $(\lambda(z))^{-1} = IFFT\left\{(FFT\left[\lambda[n]\right])^{-1}\right\}$, with $\lambda[n]$ the series of coefficients of $\lambda(z)$. Proof of existence of these inverses is given by Weiss, Millar, and Stewart [104]. We

then have

$$\begin{aligned} R(z)S(z) &= Q(z)\Lambda(z)Q^P(z)Q(z)\Lambda^{-1}(z)Q^P(z) \\ &= Q(z)\Lambda(z)\Lambda^{-1}(z)Q^P(z) \\ &= Q(z)Q^P(z) \\ &= \mathbb{I} \\ &= Q(z)\Lambda^{-1}(z)\Lambda(z)Q^P(z) \\ &= Q(z)\Lambda^{-1}(z)Q(z)Q^P(z)\Lambda(z)Q^P(z) \\ &= S(z)R(z) \end{aligned}$$

where $\mathbb{I}$ is the identity matrix, i.e. $S(z) = (R(z))^{-1}$, and thus the inverses exist.

4.3 Calculation of the PEVD

We have seen that the EVD of a parahermitian polynomial matrix $R(z) = Q(z)\Lambda(z)Q^P(z)$ exists, but how does one calculate it in practice? The most common way is through the use of the algorithm known as the Sequential Best Rotation Algorithm Using Second Order Statistics (known as the SBR2 algorithm) [66], or one of its variants. This was originally an *ad hoc* method which was later proven to indeed converge to the PEVD. The aim of the SBR2 algorithm is to generate a paraunitary matrix $Q(z)$ such that $Q^P(z)R(z)Q(z) = \Lambda(z)$, where $\Lambda(z)$ is a diagonal matrix of polynomial eigenvalues of $R(z)$.

4.3.1 The SBR2 Algorithm

Two types of construction are required to proceed: the first is the delay matrix $D_{k,\tau}(z)$, where

$$D_{k,\tau}(z) = \mathrm{diag}\{1, \ldots, 1, \underbrace{z^\tau}_{k^{\text{th}}\text{ entry}}, 1, \ldots, 1\}. \tag{4.22}$$

Note that $D_{k,\tau}(z)$ is paraunitary, as

$$\begin{aligned} D_{k,\tau}(z)D^P_{k,\tau}(z) &= \mathrm{diag}\{1, \ldots, 1, z^\tau, 1, \ldots, 1\}\mathrm{diag}\{1, \ldots, 1, z^{-\tau}, 1, \ldots, 1\} \\ &= \mathrm{diag}\{1, \ldots, 1, 1, 1, \ldots, 1\} \\ &= \mathbb{I} \\ &= D^P_{k,\tau}(z)D_{k,\tau}(z), \end{aligned}$$

and that $D_{k,\tau}(z)$ is also analytic, as each entry has a finite number of polynomial coefficients. The delay matrix is so called because it acts to shift the k^{th} column of a matrix by τ samples, equivalent to applying a delay.

The second construction is the rotation matrix, $J_{j,k}(\theta, \phi)$:

$$J_{j,k}(\theta,\phi) = \begin{bmatrix} 1 & & & & & & \\ & \ddots & & & & & \\ & & \cos\theta & \dots & e^{i\phi}\sin\theta & & \\ & & \vdots & \mathbb{I} & \vdots & & \\ & & -e^{-i\phi}\sin\theta & \dots & \cos\theta & & \\ & & & & & \ddots & \\ & & & & & & 1 \end{bmatrix}, \tag{4.23}$$

or explicitly,

$$\{J_{j,k}(\theta,\phi)\}_{l,m} = \begin{cases} \cos\theta, & \text{if } l = m = j, \text{ or } l = m = k \\ e^{i\phi}\sin\theta, & \text{if } l = j, \text{ and } m = k \\ -e^{-i\phi}\sin\theta, & \text{if } l = k, \text{ and } m = j \\ \{\mathbb{I}\}_{l,m}, & \text{otherwise.} \end{cases}$$

Notice that $J_{j,k}(\theta,\phi)$ is also paraunitary, as by taking the submatrix of non-identity parts of $J_{j,k}(\theta,\phi)$ we see

$$\begin{aligned} J_{j,k}(\theta,\phi)J^P_{j,k}(\theta,\phi) &= \begin{bmatrix} \cos\theta & e^{i\phi}\sin\theta \\ -e^{-i\phi}\sin\theta & \cos\theta \end{bmatrix} \begin{bmatrix} \cos\theta & -e^{i\phi}\sin\theta \\ e^{-i\phi}\sin\theta & \cos\theta \end{bmatrix} \\ &= \begin{bmatrix} \cos^2\theta + \sin^2\theta & e^{i\phi}\sin\theta\cos\theta - e^{i\phi}\sin\theta\cos\theta \\ e^{-i\phi}\sin\theta\cos\theta - e^{-i\phi}\sin\theta\cos\theta & \cos^2\theta + \sin^2\theta \end{bmatrix} \\ &= \mathbf{I} \end{aligned}$$

with a similar calculation for $J^P_{j,k}(\theta,\phi)J_{j,k}(\theta,\phi)$. We may now introduce the SBR2 algorithm for polynomial eigenvalue decomposition.

The SBR2 Algorithm

The SBR2 algorithm is an iterative algorithm, where each iteration is as follows:

1. Search the off-diagonal elements of parahermitian polynomial matrix $R(z)$ for the coordinates (j,k,τ) such that the coefficient of z^τ in the polynomial $R_{j,k}(z)$ has the largest absolute value of all coefficients of all off-diagonal polynomials in $R(z)$.

2. Use the coordinates (j,k,τ) to create matrix $D_{k,\tau}(z)$ as per (4.22), and conjugate $R(z)$ by $D_{k,\tau}(z)$ to produce

$$R'(z) = D_{k,\tau}(z)R(z)D^P_{k,\tau}(z). \tag{4.24}$$

   The matrix $R'(z)$ will now have its largest off-diagonal coefficient as a coefficient of z^0, in the position (j,k).

3. With $r'_{l,m}(z)$ being the entry in position (l,m) of matrix $R'(z)$, and $r'_{l,m}(0)$ being its evaluation at $z = 0$, evaluate

$$\phi = \arg\left(r'_{j,k}(0)\right) \tag{4.25}$$

and

$$\theta = \frac{1}{2}\arctan\left(\frac{2|r'_{j,k}(0)|}{r'_{j,j}(0) - r'_{k,k}(0)}\right), \tag{4.26}$$

calculated in practice with

$$\theta = \text{atan2}\left(2|r'_{j,k}(0)|,\ r'_{j,j}(0) - r'_{k,k}(0)\right) \tag{4.27}$$

as implemented in many modern programming languages, such as Python and MATLAB. Using ϕ and θ as well as coordinates j and k, create rotation matrix $J_{j,k}(\theta, \phi)$ as per (4.23), and conjugate $R'(z)$ to give

$$R''(z) = J_{j,k}(\theta, \phi)R'(z)J^P_{j,k}(\theta, \phi). \tag{4.28}$$

This Jacobi transformation has the effect of shifting the 'energy' of the largest off-diagonal coefficient (the one which was found and via $D_{k,\tau}(z)$) down onto the diagonal.

4. Repeat from step 1 with $R(z) = R''(z)$ until the matrix is sufficiently diagonalised.

Each iteration of the SBR2 algorithm shifts the largest off-diagonal element of $R(z)$ onto the diagonal, and thus the algorithm iteratively diagonalises the matrix. A visual representation of the SBR2 algorithm is provided in Appendix A, Figures A.1-A.9.

4.3.2 Example Use of SBR2

Here we give an example of use of the SBR2 algorithm on the parahermitian matrix $R(z)$, where

$$R(z) = \begin{bmatrix} 1 & -0.4iz & 0 \\ 0.4iz^{-1} & 1 & 0.5z^{-2} \\ 0 & 0.5z^2 & 1 \end{bmatrix}. \tag{4.29}$$

This matrix is the same as is given as an example in Eq. (2.21) from [25].

1. We begin by searching $R(z)$ for the coordinates of the coefficient pair with maximum absolute value. Note that we only need to search the above-diagonal (or equivalently the below-diagonal) entries, as $R(z)$ is parahermitian. In this case, the maximum element is found in position $(2, 3)$ with lag parameter -2, and has magnitude 0.5.

2. We now create the delay matrix $D_{3,-2}(z)$,

$$D_{3,-2}(z) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & z^{-2} \end{bmatrix}, \tag{4.30}$$

and conjugate $R(z)$ by $D_{3,-2}(z)$ to give:

$$\begin{aligned} R'(z) &= D_{3,-2}(z)R(z)D^P_{3,-2}(z) \\ &= \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & z^{-2} \end{bmatrix} \begin{bmatrix} 1 & -0.4iz & 0 \\ 0.4iz^{-1} & 1 & 0.5z^{-2} \\ 0 & 0.5z^2 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & z^2 \end{bmatrix} \\ &= \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & z^{-2} \end{bmatrix} \begin{bmatrix} 1 & -0.4iz & 0 \\ 0.4iz^{-1} & 1 & 0.5 \\ 0 & 0.5z^2 & z^2 \end{bmatrix} \\ &= \begin{bmatrix} 1 & -0.4iz & 0 \\ 0.4iz^{-1} & 1 & 0.5 \\ 0 & 0.5 & 1 \end{bmatrix} \end{aligned} \tag{4.31}$$

3. We now calculate ϕ and θ according to (4.25) and (4.26)

$$\begin{aligned} \phi &= \arg\left(r'_{2,3}(0)\right) \\ &= \arg\left(0.5\right) \\ &= 0 \end{aligned}$$

and

$$\begin{aligned} \theta &= \frac{1}{2}\arctan\left(\frac{2|r'_{2,3}(0)|}{r'_{2,2}(0) - r'_{3,3}(0)}\right) \\ &= \frac{1}{2}\,\mathrm{atan2}\left(1,\ 0\right) \\ &= \pi/4, \end{aligned}$$

and use them to create Jacobi rotation matrix $J_{2,3}(\frac{\pi}{4}, 0)$,

$$J_{2,3}(\tfrac{\pi}{4}, 0) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(\pi/4) & \sin(\pi/4) \\ 0 & -\sin(\pi/4) & \cos(\pi/4) \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ 0 & \frac{-1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \end{bmatrix}. \tag{4.32}$$

Next we conjugate $R'(z)$ by $J_{2,3}(\frac{\pi}{4}, 0)$ to give $R''(z)$,

$$\begin{aligned} R''(z) &= J_{2,3}(\tfrac{\pi}{4}, 0)R'(z)J^P_{2,3}(\tfrac{\pi}{4}, 0) \\ &= \begin{bmatrix} 1 & 0 & 0 \\ 0 & \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ 0 & \frac{-1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \end{bmatrix} \begin{bmatrix} 1 & -0.4iz & 0 \\ 0.4iz^{-1} & 1 & 0.5 \\ 0 & 0.5 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & \frac{1}{\sqrt{2}} & \frac{-1}{\sqrt{2}} \\ 0 & \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \end{bmatrix} \\ &= \begin{bmatrix} 1 & \frac{-i\sqrt{2}}{5}z & \frac{i\sqrt{2}}{5}z \\ \frac{i\sqrt{2}}{5}z^{-1} & 1.5 & 0 \\ \frac{-i\sqrt{2}}{5}z^{-1} & 0 & 0.5 \end{bmatrix}. \end{aligned} \tag{4.33}$$

The off-diagonal energy has been transferred down onto the diagonal, and one iteration is complete. The process repeats from step 1, with $R(z) = R''(z)$.

To assure oneself that the on-diagonal energy has in fact increased, we may compare the Frobenius norm of the diagonal entries of the original matrix, $\sqrt{1^2+1^2+1^2} = \sqrt{(3)} \approx 1.732$, with the Frobenius norm *after* the SBR2 iteration: $\sqrt{1^2+1.5^2+0.5^2} = \sqrt{(\frac{14}{4})} \approx 1.871$. Notice that the Frobenius norm of the entire matrix remains unchanged at

$$\sqrt{1+1+1+2(\frac{1}{2})^2+2(\frac{4}{10})^2} = \sqrt{1+(\frac{3}{2})^2+(\frac{1}{2})^2+4(\frac{\sqrt{2}}{5})^2} = \sqrt{\frac{191}{50}},$$

and so we can be assured that information was only shifted and not lost.

4.3.3 Notes on Practical Implementation

We may model the matrix $R(z)$ as a 3D array, for example with a four microphone array, one may have a $(4 \times 4 \times 101)$ space-time covariance matrix. Each matrix 'slice' along the third dimension corresponds to the matrix of coefficients of a different power of z. For example, we may store the polynomial

$$P(z) = az^{\tau_{max}} + bz^{\tau_{max}-1} + \cdots + cz + d + ez^{-1} + \cdots + fz^{-(\tau_{max}-1)} + gz^{-\tau_{max}} \quad (4.34)$$

in a Python/NumPy array like so:

`p = [a,b,...,c,d,e,...,f,g]`.

This means that the constant coefficient of the $(2,3)$ term in $R(z)$, for example, is accessed by

`d = R[2,3,(p.shape[-1]-1)//2]`,

and thus for our example of a $(4 \times 4 \times 101)$ NumPy array, we will be able to store polynomials with powers of z of magnitude up to $\lfloor (101-1)/2 \rfloor = 50$. If we require higher order polynomials, we must increase the size of the array.

Implicit Use of $D_{k,\tau}(z)$

Though it is possible to create the full matrix $D_{k,\tau}(z)$ and to do explicit polynomial matrix multiplication for the conjugation during the delay step in SBR2, this is not necessary in practice. For a delay of $\tau \in \mathbb{Z}$, one may simply pad the stack of matrices with a stack zeros of depth τ on each end, and then use `numpy.roll` (or an equivalent method) to delay the appropriate rows and columns by τ or $-\tau$. `numpy.roll(x,n)`, for a vector of length k and an integer n, acts to map the element at index i $(0 \leq i \leq k-1)$ to index $(i+n) \mod k$; when $n = \tau$, adding the stack of τ zeros on either end ensures that no coefficients of positive powers of z erroneously become coefficients of negative powers of z, and *vice versa*.

Memory Issues

Consider multiplying z^{49} with z^2. The answer is indeed z^{51}. If, however, you were trying to do this with the polynomials stored in arrays of size 101 as described earlier,

you would reach a problem - there is nowhere to store the 1 coefficient of the z^{51} term. There are two options: either let that piece of information disappear and suffer a loss of accuracy, or expand your matrix to be able to hold coefficients of larger exponents of z. If we choose the latter option, then during every iteration of SBR2 our matrix must increase in size by $|\tau|$, where τ is the exponent of z which corresponds to the largest magnitude coefficient. This may cause rapid growth, and the user may quickly find themselves either out of memory or waiting a long time for each SBR2 iteration to run. One option, as suggested by the authors of the SBR2 algorithm [25], is to "trim" the matrix at the end of each SBR2 iteration. This trimming procedure involves removing high-order terms layer by layer until a predefined proportion of the total energy has been removed - 'energy' here simply referring to the squared Frobenius norm of the matrix stack. In this way the end user may perform a trade-off of speed against information loss.

4.3.4 Extensions to SBR2

As noted in Section 2.2, several adaptations of the SBR2 algorithm exist which have been designed to exhibit faster convergence, including Multiple-Shift SBR2 (MS-SBR2) and Sequential Matrix Diagonalisation (SMD). MS-SBR2 is similar to SBR2, except that at each time step one may choose multiple terms to shift, under some constraints regarding which rows and columns will be affected by the Jacobi transformation. SMD is also reminiscent of SBR2, but replaces the Jacobi transformation by a complete scalar EVD at each iteration. For a comparison and review of these algorithms and more, refer to the PhD thesis of Dr Jamie Corr [105].

Chapter 5

The MUSIC Algorithm

5.1 Multiple Signal Classification (MUSIC)

In 1986, in *Multiple Emitter Location and Signal Parameter Estimation* [21], Schmidt proposed an algorithm which allows for the simultaneous location estimation of multiple sources. This method, known as MUSIC (**Mu**ltiple **Si**gnal **C**lassification), is based on the properties of spaces generated from the data, rather than directly derived from the data itself akin to GCC. MUSIC thus benefits from 'superresolution', i.e. it is not confined by the discrete nature of the data. Schmidt modelled the received vector X (one frequency component of the Fourier transform of the received data) by

$$\begin{bmatrix} X_1 \\ X_2 \\ \vdots \\ X_M \end{bmatrix} = \begin{bmatrix} & & & \\ \mathbf{a}_{\theta_1}(\omega_1) & \mathbf{a}_{\theta_2}(\omega_2) & \dots & \mathbf{a}_{\theta_D}(\omega_D) \\ & & & \end{bmatrix} \begin{bmatrix} f_1 \\ f_2 \\ \vdots \\ f_D \end{bmatrix} + \begin{bmatrix} w_1 \\ w_2 \\ \vdots \\ w_M \end{bmatrix}$$

or equivalently

$$X = A\mathbf{f} + \mathbf{w}, \tag{5.1}$$

where a uppercase variable denotes a matrix, and an bold lowercase variable a vector. Though X is a vector, here it remains uppercase to symbolise that it is the result of a Fourier transform.

These terms require some explanation. Beginning on the right-hand side, w_i ($1 \leq i \leq M$) is the total noise component of the data in channel i, whether it originates from within the receiver/array itself or during transmission; we model this with a vector of independent identically distributed Gaussian random variables. The next term to the left, f_j ($1 \leq j \leq D$), where D is the number of sources present in the signal, is the complex exponential emitted from the j^{th} source, of frequency ω_j radians/sample. Finally, $\mathbf{a}_{\theta_j}(\omega_j)$ is the j^{th} complex *steering vector*, a function of the direction-of-arrival θ which acts to shift the phase of f_j to match how it would be observed at each microphone, had the source indeed come from angle θ_j. These phase shifts are used to model the delays that occur between the wavefront hitting each of the microphones,

and for some frequency ω will look like so:

$$\mathbf{a}_\theta(\omega) = \begin{bmatrix} e^{-i\omega\tau_1} & e^{-i\omega\tau_2} & \dots & e^{-i\omega\tau_M} \end{bmatrix}^T, \tag{5.2}$$

where τ_k is the relative time taken for the wavefront to travel from some reference point (be it the source location or otherwise) to microphone k. If working in units of Hertz (samples/second), (5.2) becomes:

$$\mathbf{a}_\theta(f) = \begin{bmatrix} e^{-i2\pi f\tau_1} & e^{-i2\pi f\tau_2} & \dots & e^{-i2\pi f\tau_M} \end{bmatrix}^T. \tag{5.3}$$

The steering vectors may be readily extended to be a function of both angle and azimuth, or of angle and radius, if required – this will only act to change the calculation of the value of τ in each entry. The following explanation of the MUSIC algorithm is adapted from Schmidt's 1986 paper [21].

Under the assumption that the sources and noise are uncorrelated (so that the expected value of the source/noise cross-terms is zero), the $M \times M$ covariance matrix S of the vector X is given by

$$S = \mathbb{E}[XX^*] = A\mathbb{E}[\mathbf{f}\mathbf{f}^*]A^* + \mathbb{E}[\mathbf{w}\mathbf{w}^*] = APA^* + \lambda S_0, \tag{5.4}$$

where denotes X^* the Hermitian transpose (i.e. conjugate transpose) of X, $\mathbb{E}[XX^*]$ denotes the expectation[1] of the outer product XX^*, and where $P = \mathbb{E}[\mathbf{f}\mathbf{f}^*]$. Note that P is positive semidefinite (it is a rank 1 matrix; the columns are linear combination of the vector $\mathbf{f}$) and thus, for A full rank, APA^* is positive semidefinite; conjugation by A is merely a change of basis and does not effect the dimensionality of the eigen-structure of P.

If the number of signals to be localised is less than the number M of microphones in the array, then APA^* is singular with rank $(M-D) < M$, and thus $\det(APA^*) = 0$. By (5.4), then, we have that $\det(S - \lambda S_0) = 0$. Notice that this is only true with λ equal to one of the eigenvalues of S in the metric of S_0 (i.e. solutions to the generalised eigenvalue problem $S\mathbf{v} = \lambda S_0\mathbf{v}$), and as APA^* is positive semi-definite, λ must be equal to the smallest eigenvalue $\lambda_{\min} \gtrapprox 0$. Hence we may write (5.4) as

$$S = APA^* + \lambda_{\min} S_0, \tag{5.5}$$

where $\lambda_{\min} \geq 0$ is the smallest solution to $\det(S - \lambda S_0) = 0$. As the positive semidefinite matrix $APA^* = S - \lambda_{\min} S_0$ is singular with rank $M - D$, its minimum eigenvalue is 0, with multiplicity $M - D$. Notice now that the eigenvalues of S and the eigenvalues of $S - \lambda_{\min} S_0$ all differ by $\lambda_{\min}$; this implies that the multiplicity of the minimum eigenvalue $\lambda_{\min}$ must also be $M - D$.

The last observation required before presenting the final result for the MUSIC algorithm is as follows: by definition, the M eigenvectors $\mathbf{e}_i$ $(1 \leq i \leq M)$ and their corresponding eigenvalues λ_i of S in the metric of S_0 must satisfy

$$S\mathbf{e}_i = \lambda_i S_0 \mathbf{e}_i. \tag{5.6}$$

[1] i.e. the average value of the $M \times M$ matrix XX^* over several partitions of the data.

Now by substituting (5.6) into (5.5), we have

$$(APA^* + \lambda_{\min})\mathbf{e}_i = \lambda_i S_0 \mathbf{e}_i \implies APA^*\mathbf{e}_i = (\lambda_i - \lambda_{\min})S_0\mathbf{e}_i, \tag{5.7}$$

and so for the $M - D$ eigenvectors with corresponding eigenvalues equal to $\lambda_{\min}$, we must have $APA^*\mathbf{e}_i = 0$, and hence the eigenvectors associated with $\lambda_{\min}$ are orthogonal to the column-space of A.

We now have two orthogonal subspaces of $\mathbb{C}^M$: the space spanned by the eigenvectors corresponding to $\lambda_{\min}$ (known as the *noise subspace*) which is governed by the noise in the data and is orthogonal to the column-space of A; and a subspace governed by the D 'signal' eigenvectors corresponding to the D eigenvalues not equal to $\lambda_{\min}$ (we call this the *signal subspace*).

The continuum of possible steering vectors $\mathbf{a}_\theta(\omega)$ winds through $\mathbb{C}^M$, and intercepts the signal subspace precisely D times. When an intercept occurs, the angle θ from $\mathbf{a}_\theta(\omega)$ points in the same direction as one of the D sources - and thus we may search over the continuum of possible steering vectors to find these intercepts. Schmidt [21] gives a graphical representation of this process, shown in Figure 5.1.

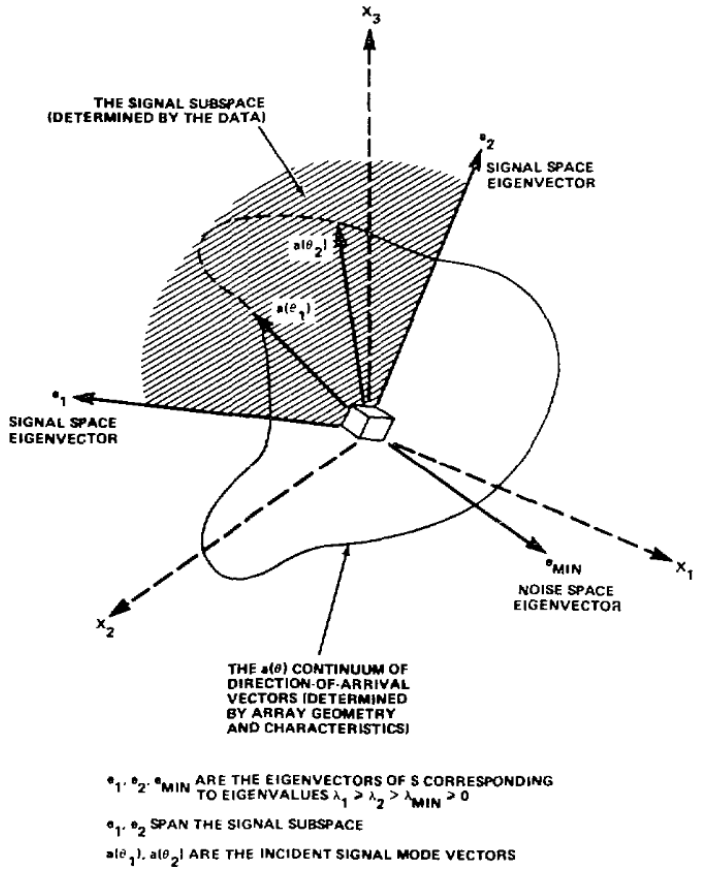


Figure 5.1: A visualisation of the MUSIC algorithm. The image shown is for the case with three microphones and two sources, hence why the space depicted is $\mathbb{R}^3$ with a 2D signal subspace. $x_{1,\cdots,3}$ are the standard basis vectors. θ_1 and θ_2 are the angles at which the steering vectors intercept the signal subspace, and thus are the angles at which the sources were positioned during the recording of the data. Image taken from [21] ©1986 IEEE.

Given any vector $\mathbf{a}_\theta(\omega)$, we may calculate the distance between it and the signal subspace using its orthogonal projection onto the noise eigenvectors. The MUSIC spectrum P_{MU} is defined to be the reciprocal of this squared distance; the closer the steering vector is to the signal subspace, the higher the value of P_{MU}. The MUSIC spectrum is given by:

$$P_{\mathrm{MU}}(\theta) = ||E_N^*(\omega)\mathbf{a}_\theta(\omega)||_2^{-2} \tag{5.8}$$

$$= (\mathbf{a}_\theta^*(\omega)E_N(\omega)E_N^*(\omega)\mathbf{a}_\theta(\omega))^{-1}\,, \tag{5.9}$$

where the columns of the matrix $E_N(\omega)$ are the $M - D$ noise eigenvectors as discussed previously. To find $E_N(\omega)$ in practice, we sort the eigenvectors of the covariance matrix S in non-increasing order by the magnitude of their corresponding eigenvalues. The eigenvectors corresponding to the $M - D$ smallest eigenvalues of S then form the columns of E_N. The $M - D$ smallest eigenvalues of S should be approximately 0, and

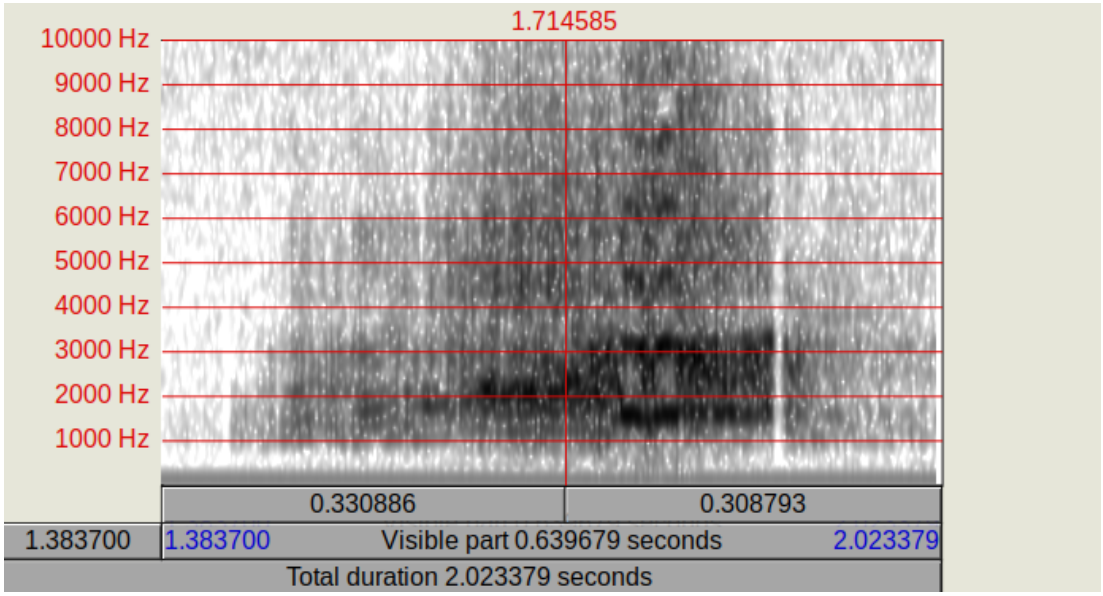


Figure 5.2: Spectrogram of the call of a captive female North Island Brown Kiwi. Here, a darker colour corresponds to a higher signal strength. Horizontal axis is time as shown in blue, and vertical axis is frequency. Audio taken from `http://www.nzbirdsonline.org.nz/sites/all/files/09%20-%20Track%209_12.mp3`, spectrogram generated with the use of the free software Praat, available from `http://www.fon.hum.uva.nl/praat/`. Note the spread of intensity over a large range of frequencies.

so we may get an estimate of D, if it is not known *a priori*, by thresholding this set of eigenvalues.

Though not of further interest in this thesis, Schmidt shows that under the assumption that the noise is uncorrelated and white (i.e. has identity covariance matrix), we may also estimate the original signal parameters by approximating the covariance matrix $P = \mathbb{E}[\mathbf{f}\mathbf{f}^*]$ by

$$P \approx (A^*A)^{-1}A^*(\mathbb{E}[XX^*] - \lambda_{\min}\mathbb{I})A(A^*A)^{-1}. \tag{5.10}$$

For situations where the assumption of whiteness does not hold, another estimator which whitens as necessary is also provided by Schmidt [21].

5.1.1 Limitations

It was briefly mentioned in the foundations of the MUSIC algorithm in Section 5.1, that the algorithm only deals with one frequency component of the data. This means that the end user must either be attempting to localise a narrowband source of known frequency, or must choose one frequency to look at from a range of active frequencies in a broadband signal. The former is not always the case, especially when working with acoustic signals, and the latter poses some problems of its own. For example, consider the call of the female North Island Brown Kiwi, *Apteryx mantelli*, with spectrogram shown in Figure 5.2. The call of this bird is a harsh screech, with total energy spread

out over a rather large range of frequencies. This spread makes for a lower signal-to-noise ratio (SNR) at each frequency, reducing the effectiveness of the algorithm at each individual frequency. How then does one get around these limitations?

5.1.2 Extensions to the MUSIC Algorithm

The most obvious initial choice is to sum the output of the MUSIC algorithm over all K frequency bins ω_k ($k = 1, \cdots, K$) obtained from the DFT of the data. This weights all frequencies equally, which may in fact make performance worse when the source does not cover the entire frequency spectrum, as the noise in the signal will be treated with equal importance to the signal at the frequencies at which the source is not present. To get around this, we may weight the sum by the square root of the largest eigenvalue for each frequency bin, which is roughly proportional to the strength of the source component of the signal [22]. This method is known as the Incoherent Signal Subspace Method (ISSM) [106], and is formally:

$$P_{\text{ISSM}}(\theta) = \sum_{k=1}^{K} \left(\sqrt{\lambda_{\max}(\omega_k)} ||E_N^*(\omega_k)\mathbf{a}_\theta(\omega_k)||_2^{-2} \right) \tag{5.11}$$

$$= \sum_{k=1}^{K} \left(\frac{\sqrt{\lambda_{\max}(\omega_k)}}{\mathbf{a}_\theta^*(\omega_k)E_N(\omega_k)E_N^*(\omega_k)\mathbf{a}_\theta(\omega_k)} \right), \tag{5.12}$$

where K is the number of frequencies at which the DFT is evaluated, and $\lambda_{\max}(\omega_k)$ is the maximum eigenvalue of the covariance matrix $\mathbb{E}\{X[k]X^*[k]\}$. Here we have explicitly stated each term's dependence on the frequency ω. This is a good first-attempt method at applying MUSIC to broadband scenarios, but breaks down in the situation where the eigenspaces have different structures at different frequencies; for example, if there is one source active over the 1000-2000Hz range, but two active over the 50-100Hz range.

To work with situations with a variable number of sources across the frequency spectrum, Wang and Kaveh [23] first transformed each frequency to some reference frequency ω_0, and then performed the MUSIC algorithm on the 'combined eigenspace' at ω_0. However, Wang and Kaveh's algorithm required initial estimates of the direction-of-arrival, and designing individual focusing matrices for each frequency bin. In 2009, Pal and Vaidyanathan [107] introduced a new method which avoids these problems by performing the 'focusing' from Wang and Kaveh's work automatically, and was thus coined autofocusing MUSIC, or AF-MUSIC. The AF-MUSIC spectrum is the same as the regular MUSIC spectrum, but performed on the "universal focused sample covariance matrix", R_{coh} [107]:

$$R_{coh} = E_S(\omega_0)R_s E_S^H(\omega_0) + \sigma^2 I \tag{5.13}$$

$$R_s = \frac{1}{K}\sum_{k=0}^{K-1} E_S^H(\omega_k)A(\omega_k)R(\omega_k)A^H(\omega_k)E_S(\omega_k), \tag{5.14}$$

where K is the number of frequency bins, $E_s(\omega_k)$ is a matrix of orthogonal spanning vectors for the signal subspace at frequency ω_k, $A(\theta_k)$ is the matrix of array steering vectors at frequency ω_k, and $R(\omega_\text{k})$ is the covariance matrix of the data at frequency

ω_k. The universal focused sample covariance matrix, R_{coh}, is a matrix whose rows span the same subspace of $\mathbb{C}^M$ as the matrix of array steering vectors at reference frequency ω_0, but is built to coherently combine the information available from the data at all K available frequencies, and is given by

$$R_{coh} = \sum_{k=0}^{K-1} R_{yy}[k], \tag{5.15}$$

$$R_{yy}[k] = \mathbb{E}\left[Y[k](Y[k])^H\right], \tag{5.16}$$

$$Y[k] = T_{auto}(\omega_0, \omega_k)X[k], \tag{5.17}$$

$$T_{auto}(\omega_0, \omega_k) = \frac{1}{\sqrt{K}} E(\omega_0)(E(\omega_k))^H, \tag{5.18}$$

where $T_{auto}(\omega_0, \omega_k)$ is known as the 'focusing matrix' and acts to transform the information at frequency ω_k to ω_0, and $E(\omega_k)$ is the matrix of eigenvectors of $R_{xx}[k] = \mathbb{E}\left[X[\omega_k]X^*[\omega_k]\right]$ sorted in non-increasing order by the magnitude of their corresponding eigenvalues. Combining (5.15)-(5.18) gives

$$R_{coh} = \frac{1}{K} \sum_{k=0}^{K-1} \mathbb{E}\left[E(\omega_0)(E(\omega_k))^H X[k](X[k])^H E(\omega_k)(E(\omega_0))^H\right] \tag{5.19}$$

$$= \sum_{k=0}^{K-1} T_{auto}\left(\omega_0, \omega_k\right) R_{xx}[k] T_{auto}^H\left(\omega_0, \omega_k\right) \tag{5.20}$$

$$\tag{5.21}$$

which only requires the use of known matrices – we do not need to know the number of sources at frequency ω_k to formulate $E(\omega_k)$. The AF-MUSIC Spectrum is then given by [107]:

$$P_{AF}(\theta, \omega_0) = ||E_N^*(\omega_0)\mathbf{a}_\theta(\omega_0)||_2^{-2} \tag{5.22}$$

$$= \left(\mathbf{a}_\theta^*(\omega_0)E_N(\omega_0)E_N^*(\omega_0)\mathbf{a}_\theta(\omega_0)\right)^{-1}, \tag{5.23}$$

where $E_N(\omega_0)$ is the matrix of noise-only eigenvectors of the universal focused sample covariance matrix, R_{coh}, and $\mathbf{a}_\theta(\omega_0)$ is the steering vector as in Eq. (5.2), evaluated at reference frequency ω_0. We may estimate the number of sources active over the entire frequency range by thresholding the eigenvalues of R_{coh} similarly to the narrowband case.

AF-MUSIC allows the end user to make use of the information present at all frequencies in the signal, however they are not explicitly able see to what extent each frequency contributes to the directional estimate like as is possible with (5.11). To gain access to this information, we may utilise polynomial matrices.

5.2 Broadband MUSIC

Introduction of the concept of polynomial matrices to the MUSIC algorithm allows us to change from a narrowband covariance matrix to one which captures the broadband

nature of the data, namely $R(z)$, the z-transform of the space-time covariance matrix (4.10) as introduced in Section 4.2.1. This is a polynomial matrix in the variable z which describes the covariance between all frequencies of all channels of the observed data.

5.2.1 The MUSIC Spectrum

Similarly to the narrow-band case, we may scan over the nullspace of the matrix $E_N(z)$ of noise-space eigenvectors of $R(z)$ using the broadband steering vector

$$\mathbf{a}_\theta(z) = \begin{bmatrix} D_0(z,\theta) \\ \vdots \\ D_{M-1}(z,\theta) \end{bmatrix} \tag{5.24}$$

with $D_i(z,\theta)$ the z-transform of the fractional delay filter $d_i[n] = \text{sinc}\,(nT_s - \Delta\tau_i)$, where T_s is the sample period of the array, and $\Delta\tau_i$ is the time delay at channel i. However, now

$$\Gamma_\theta(z) = \mathbf{a}_\theta^P(z) E_N(z) E_N^P(z) \mathbf{a}_\theta(z)$$

is not the squared norm of the distance from the steering vector estimate to the signal-subspace, but a power spectral density, and hence contains frequency information. This motivates two versions of polynomial MUSIC: Polynomial Spatio-MUSIC, which integrates over the frequencies to provide only a function of the direction-of-arrival; and Polynomial Spatio-Spectral MUSIC, which leaves the frequency information in to be used or searched over.

Polynomial Spatio-MUSIC

We can use the autocorrelation-type sequence $\gamma_\theta[\tau]$, the inverse z-transform of $\Gamma_\theta(z)$, to get to the required DOA estimate. In particular, as we are dealing with an autocorrelation, the zero-lag term will be representative of the total energy of the signal, and thus we may search over

$$P_{\text{PS}}(\theta) = (\gamma_\theta[0])^{-1} \tag{5.25}$$

$$= \left(\frac{1}{2\pi} \int_{-\pi}^{\pi} \Gamma_\theta(e^{i\omega}) d\omega \right)^{-1} \tag{5.26}$$

to find a global maxima, such as in narrow-band MUSIC [26]. In practice, as we know parahermitian matrices are analytic on the unit circle, we may approximate the integral in (5.26) by a sum over K discrete frequency bins ω_k $(k = 1, \cdots, K)$ in a DFT, i.e.

$$P_{\text{PS}}(\theta) \approx K \left(\sum_{k=1}^{K} \Gamma_\theta(\omega_k) \right)^{-1} \tag{5.27}$$

$$= K \left(\sum_{k=1}^{K} \mathbf{a}_\theta^P(\omega_k) E_N(\omega_k) E_N^P(\omega_k) \mathbf{a}_\theta(\omega_k) \right)^{-1}. \tag{5.28}$$

An example output from the Polynomial Spatio-MUSIC algorithm is shown in Figure 5.3.

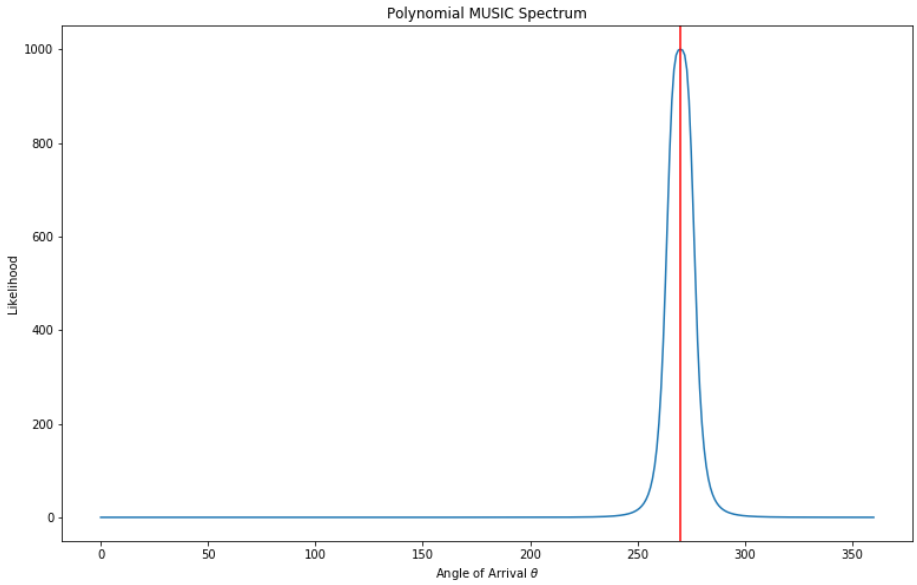


Figure 5.3: An example output of the Polynomial Spatio-MUSIC algorithm. The horizontal axis here is the direction-of-arrival θ, and the vertical axis is an indication of how likely it is that the source came from that angle.

Polynomial Spatio-Spectral MUSIC

If, however, we would like to make use of the spectral component of the PSD,

$$P_{\text{PSS}}(\theta, e^{i\omega}) = \left(\Gamma_\theta(e^{i\omega})\right)^{-1} \tag{5.29}$$

$$= \left(\mathbf{a}_\theta^P(e^{i\omega}) E_N(e^{i\omega}) E_N^P(e^{i\omega}) \mathbf{a}_\theta(e^{i\omega})\right)^{-1} \tag{5.30}$$

will allow us to search over both the angle of arrival, θ, and the normalised frequency ω. In this way we may examine not only estimates of direction, but the frequencies which contributed to them, for example in Figure 5.4.

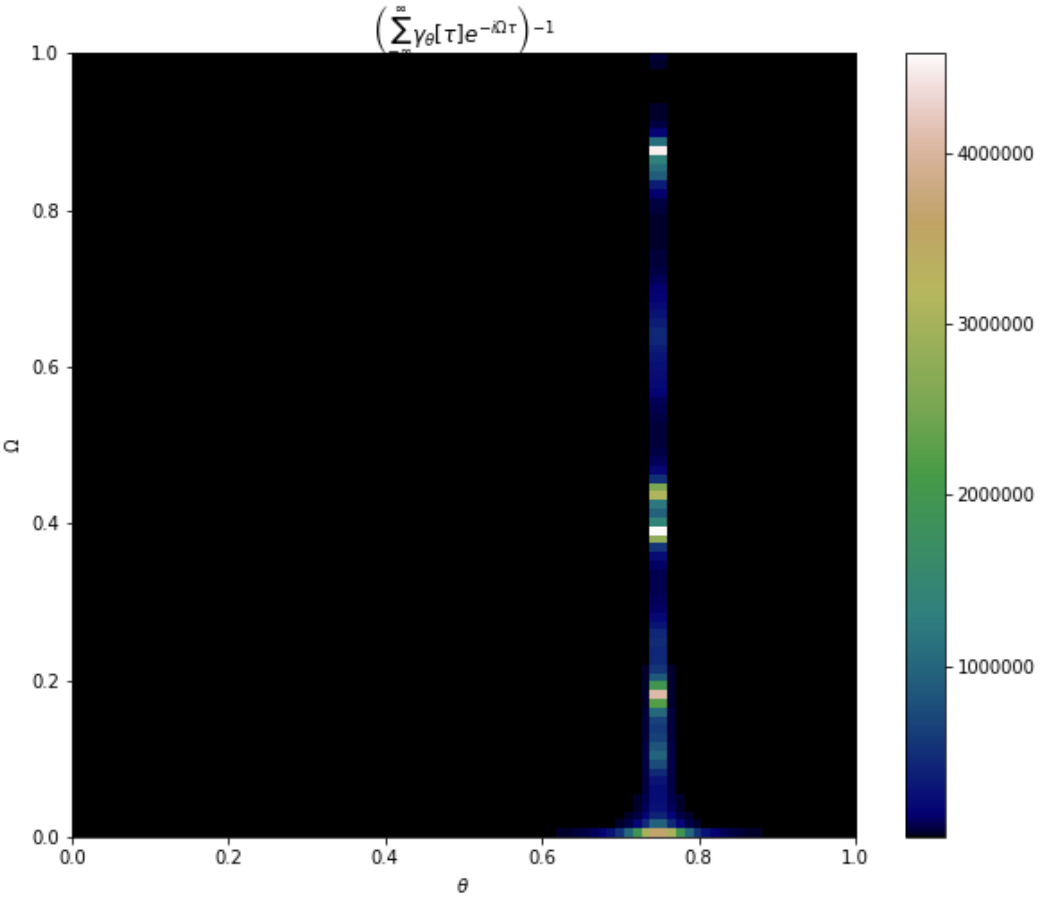


Figure 5.4: An example output of the Polynomial Spatio-Spectral MUSIC algorithm. The horizontal axis here is the direction-of-arrival $\theta/2\pi$, the vertical axis is the normalised frequency Ω, and the colour intensity is proportional to the likelihood of the source being in that position in $\theta - \Omega$ space.

Notice that PS-MUSIC and PSS-MUSIC are closely related, explicitly:

$$\begin{aligned}
\frac{1}{2\pi}\int_{-\pi}^{\pi} P_{\mathrm{PSS}}(\theta, e^{i\Omega})d\Omega &= \frac{1}{2\pi}\int_{-\pi}^{\pi} \left(\mathbf{a}_\theta^P(e^{i\Omega})E_N(e^{i\Omega})E_N^P(e^{i\Omega})\mathbf{a}_\theta(e^{i\Omega})\right)^{-1} d\Omega \\
&= \frac{1}{2\pi}\int_{-\pi}^{\pi} \left(\Gamma_\theta(e^{i\Omega})\right)^{-1} d\Omega \\
&= \left(\frac{1}{2\pi}\int_{-\pi}^{\pi} \Gamma_\theta(e^{i\Omega})d\Omega\right)^{-1} \\
&= \left(\gamma_\theta[0]\right)^{-1} \\
&= P_{\mathrm{PS}}(\theta).
\end{aligned}$$

5.2.2 Limitations

Upon application of Broadband MUSIC via polynomial matrices to the problem of acoustic source localisation with the array configurations used in our experiment (a full

explanation of which is provided in Chapter 6), we noticed that none of the PEVD algorithms tested could sufficiently diagonalise the required space-time covariance matrix. We think this is because the initial maximum lag parameter is so large; indeed, as the maximum inter-microphone distance in our four-microphone array was 0.5447m with a sample rate of 96kHz, we had an initial maximum lag parameter τ of

$$\tau = \left\lceil \frac{0.5447 * 96000}{c_{\text{sound}}} \right\rceil \approx 153 \text{ samples},$$

where c_{sound} is the speed of sound in air, which requires diagonalisation of a $(4 \times 4 \times 307)$ array. It would appear that such a level of diagonalisation is not attainable with current algorithms (S. Weiss, personal communication, February 14, 2019). However, we will show that AF-MUSIC may be used as a low computational-cost approximation to the polynomial MUSIC algorithms.

5.2.3 Equivalence of PS(S)-MUSIC to AF-MUSIC

Returning to AF-MUSIC and the definition of the universal focused sample covariance matrix given by (5.19), we can write:

$$R_{coh} = \frac{1}{K} \sum_{k=0}^{K-1} \mathbb{E}[E(\omega_0)(E(\omega_k))^H X[k](X[k])^H E(\omega_k)(E(\omega_0))^H] \qquad \text{(5.19, revisited)}$$

$$= \frac{1}{K} \sum_{k=0}^{K-1} T(\omega_k) R(\omega_k)(T(\omega_k))^H. \qquad (5.31)$$

If we replace the summation over frequencies in (5.31) with an integration over the Fourier transform (i.e. letting the number of frequency bins K tend to infinity), we get:

$$R_{coh} \approx \frac{1}{2\pi} \oint \left\{ T(z) R(z) (T(z))^H \right\}_{z=e^{i\omega}} d\omega \qquad (5.32)$$

$$= E(\omega_0) \left(\frac{1}{2\pi} \oint \left\{ E(z) R(z) (E(z))^H \right\}_{z=e^{i\omega}} d\omega \right) (E(\omega_0))^H, \qquad (5.33)$$

assuming that the DFT is a good approximation of the Fourier transform [27]. However, as we know from Chapter 4.2, the matrix $E(z)$ diagonalises the polynomial matrix $R(z)$ to give diagonal matrix $\Lambda(z)$, and thus we have

$$R_{coh} \approx E(\omega_0) \left(\frac{1}{2\pi} \oint \left\{ \Lambda(z) \right\}_{z=e^{i\omega}} d\omega \right) (E(\omega_0))^H \qquad (5.34)$$

$$= E(\omega_0) \Lambda[0] (E(\omega_0))^H \qquad (5.35)$$

$$\implies \Lambda[0] \approx (E(\omega_0))^H R_{coh} E(\omega_0), \qquad (5.36)$$

where (5.35) comes from a direct application of the definition of the inverse Fourier transform, specifically for $n = 0$;

$$\begin{aligned}\Lambda[0] &= \frac{1}{2\pi}\int_{\pi}^{\pi} \Lambda(e^{i\omega})e^{i\omega n}d\omega \\ &= \frac{1}{2\pi}\oint \{\Lambda(z)\}_{z=e^{i\omega}}\, d\omega.\end{aligned}$$

Hence conjugation of R_{coh} by $E(\omega_0)$ approximately yields the zero-lag coefficient of the diagonal matrix of polynomial eigenvalues of the space-time covariance matrix $R(z)$.

Now, for K suitably high such that the DTFT is a good approximation of the Fourier transform (3.3):

$$P_{AF}(\theta,\omega_0) = \left(\mathbf{a}_\theta^P(\omega_0)E_N(\omega_0)E_N^P(\omega_0)\mathbf{a}_\theta(\omega_0)\right)^{-1} \tag{5.23}$$

$$\approx \left\{\left(\mathbf{a}_\theta^P(z)E_N(z)E_N^P(z)\mathbf{a}_\theta(z)\right)^{-1}\right\}|_{z=e^{i\omega_0}}, \tag{5.30}$$

and thus AF-MUSIC with reference frequency ω_0 is approximately equal to SSP-MUSIC evaluated at ω_0 [27, 71].

Chapter 6

Experiment

During the course of this thesis, two experiments were conducted; one with the general aim of collecting data for future use, and one to investigate the effect of array size on the DOA estimation ability of the various implemented algorithms. The former was only of use for initial testing during development, until the ability to simulate data was added to the code. This chapter will focus on the latter experiment.

6.1 Aim & Overview

The aim of this experiment is to determine the effect of the array size on the result of the currently implemented source localisation algorithms. The array configuration was an equilateral triangular array with a fourth microphone in the center. Beginning at minimum arm length of 0.07m, we linearly and uniformly increased the distance from the origin to the three non-central microphones at each iteration, stopping at a limit of 0.346m (governed by the physical constraints of the board housing the microphones). This design is illustrated in Figure 6.1.

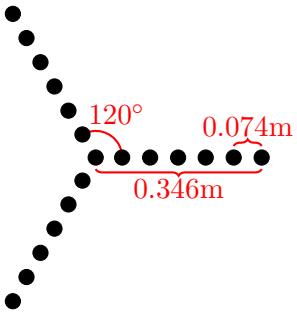


Figure 6.1: Illustration of the array configurations.

6.2 Audio File

The 7.5 second, 44.1 kHz, single-channel audio file for use in the experiment was constructed with the following components:

- 1.0 seconds of a bandlimited 440 Hz sawtooth wave (seen at $t = 0.5$ to $t = 1.5$ in Figure 6.2). This was chosen for use with broadband source localisation algorithms.
- 1.0 seconds of a 4439.5 Hz sine wave ($t = 2.0$ to $t = 3.0$ in Figure 6.2). This frequency was picked as it coincided with the strongest power frequency in the Stitchbird call situated at the end of the audio file. This frequency corresponds to a wavelength of $\frac{343.1}{4439.5} = 0.077283$ m.

  All of the above sections of audio are inter-dispersed with short, half-second, periods of silence for ease of data separation – bringing the total playback time to 7.5 seconds. A spectrogram of the file is shown in Figure 6.2.
- 1.25 seconds of a female North Island Brown Kiwi call ($t = 5.0$ to $t = 6.25$ in Figure 6.2). This was chosen as it is a harsh, broadband, call and also a species of particular interest due to both it's 'Declining' conservation status [108], and its status as a *de facto* national icon.
- 0.75 seconds of a Stitchbird/Hihi call ($t = 6.75$ to $t = 7.5$ in Figure 6.2). This bird was chosen as it has a more melodic, higher-pitched call, and a 'Nationally Vulnerable' conservation status [108].

All of the above sections of audio are inter-dispersed with short, half-second, periods of silence for ease of data separation – bringing the total playback time to 7.5 seconds. A spectrogram of the file is shown in Figure 6.2.

6.3 Method

Due to the vertical symmetry of the array, either positive or negative angles may be used to transmit the sound from the source, and due to the rotational symmetry, it is only necessary to consider angles with magnitude less than or equal to $\frac{\pi}{3}$. In this case, to ensure a clear path between source and microphones on the sloped ground in the forest clearing where the experiment was conducted, negative angles were used as illustrated in Figure 6.3.

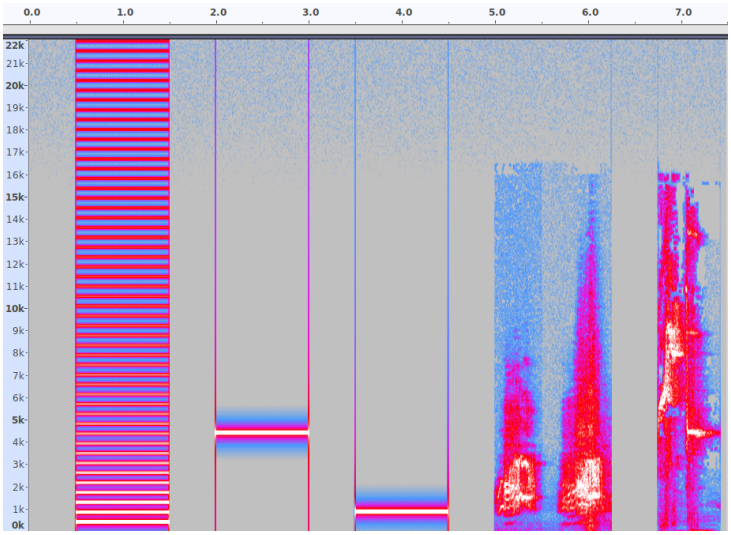


Figure 6.2: Spectrogram of the audio file used for the experiment. The time axis is shown along the top of the figure, with the frequency (in Hz) shown on the left-hand-side. The brightness of colour at any point on the time-frequency graph corresponds to the strength of the signal at that point. The apparent 'clicks' at the start and end of each piece of audio (e.g. at 2.0s and 3.0s) are artefacts of the procedure used to generate the spectrogram.

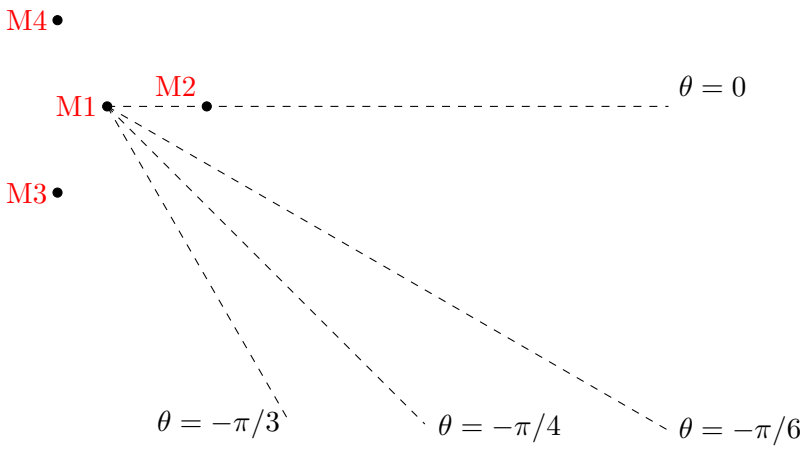


Figure 6.3: Visual demonstration of the source directions-of-arrival, with microphones corresponding to channels 1-4 labelled M1-M4, respectively. The speaker was placed 8m along the shown rays.

The method for the experiment is outlined in Algorithm 1. All coordinates are in meters, relative to the center of the array. The method involves the physical handling of the array mid-experiment to adjust the position of the microphones. As a result of this, it is possible that small biases in the direction-of-arrival estimates may appear as a result of the array not being placed perfectly in alignment with its position at previous iterations. Indeed for the experiment conducted by the author, the board in which the microphones were housed was lined up with four marks spray-painted on the

ground at each of its four corners – however this was difficult to do precisely, and some small biases may be seen in the DOA estimates. Some sort of rigid, semi-permanent, physical mount that holds the array firmly in position could be used to counteract this in future experiments.

initialisation place microphone 1 at position $(0, 0)$;
for $d \in \{0.07, 0.125, 0.181, 0.236, 0.291, 0.346\}$ **do**
  for $m \in \{2, 3, 4\}$ **do**
    place microphone m at coordinates $(d \cos \frac{(2-m)\pi}{3}, d \sin \frac{(2-m)\pi}{3})$;
  end
  begin audio capture;
  for $\vartheta \in \{0, \frac{-\pi}{6}, \frac{-\pi}{4}, \frac{-\pi}{3}\}$ **do**
    place the speaker at coordinate $(8 \cos \vartheta, 8 \sin \vartheta)$;
    silently:
      begin audio file playback;
      while *audio playing* **do**
        wait;
      end
    end silence
  end
  cease audio capture;
end

Algorithm 1: Method for the experiment.

Chapter 7

Results

Of the two algorithms tested; GCC and AF-MUSIC, both appear to work well. Several GCC processors were tested, with ρ-PHAT showing the most promise for use in acoustic source localisation. In general, AF-MUSIC provided a more precise estimate, as well as a lower tendency to provide false estimates when compared to GCC, however at a notably higher computational cost.

7.1 GCC Performance

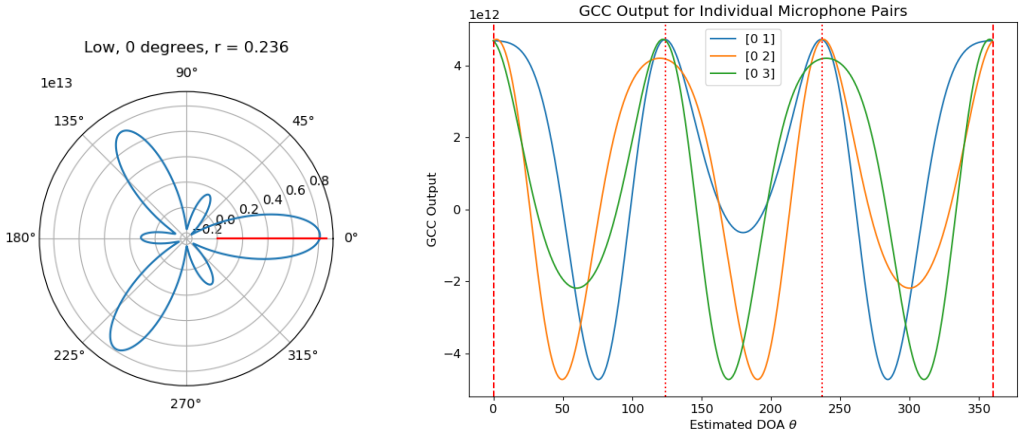
In general, GCC appears to be be a reasonably accurate, low-cost, algorithm which is not particularly difficult to implement. The choice of multiple processors such as those described in Section 3.3.1 provides some versatility. From simulation and manual inspection, GCC works best in broadband high signal-to-noise ratio environments. It would thus be a better choice for localisation of birds with lower-pitched, harsh, calls such as the female North Island Brown Kiwi, rather than high pitched birds such as the Hihi/Stitchbird (whose high-pitched calls drop in power rapidly over a distance), or more melodic calls such as the Grey Warbler.

7.1.1 False Positives

For highly periodic signals (such as the two sine waves included in the experiment), the cross-correlation between any pair of microphones is also periodic, with period the same as the period of the signal. This means that for delays an integer period out from the true delay the cross-correlation will be nearly as high as the cross-correlation at the true delay, and thus any angles which correspond to these false GCC peaks for that microphone pair will contribute a significant proportion of their global maxima to the GCC sum as in Eq. (B.2). If the wavelength of the signal and array geometry is such that multiple pairs of microphones happen to have this occur at the same time, the effect will be exacerbated, such as in Figure 7.1.

This phenomenon was also present in any noise-free simulation of GCC with the high pitched 4439.5 Hz call, however for this high of a frequency sound, the source was massively overpowered in magnitude by the noise in the data, and as such the output from almost all GCC calculations on the experimental data was akin to Figure 7.2.

It should be noted that the false positives were only of major concern in low-frequency, narrowband, signals – indeed, for a more complex broadband call such as



(a) Total GCC Output

(b) Example Correlations

Figure 7.1: (a) shows the GCC output obtained from experimental data for localisation of a 909.5 Hz sine wave at an angle of 0 degrees relative to the array, as illustrated by the red line. (b) shows a subset of the correlation functions for some microphone pairs (legend "[a b]" indicates the line corresponds to the GCC output of microphones $a+1$ and $b+1$); the red dashed line shows the true DOA, and the red dotted lines indicate the angles of the false positives in (a).

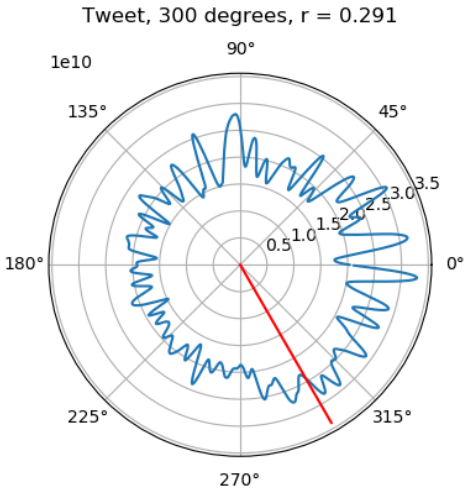
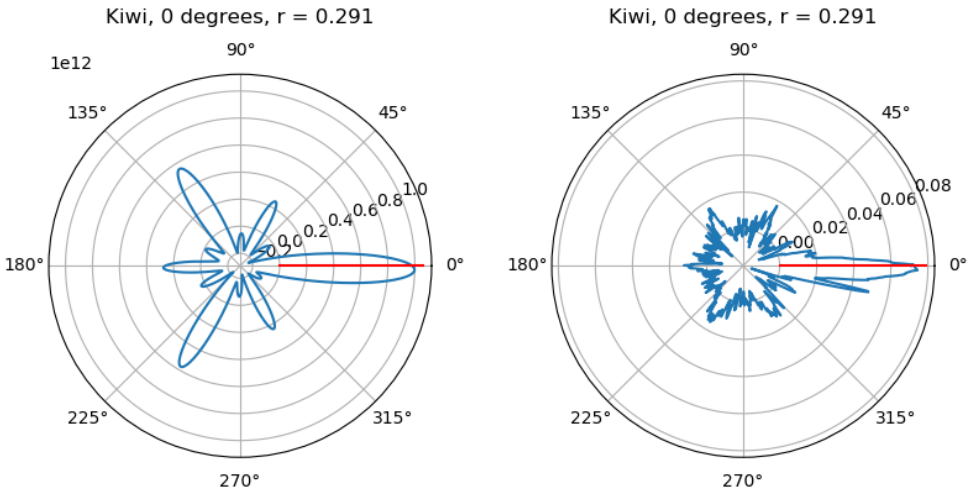


Figure 7.2: A representative example of the GCC output for the 4439.5 Hz sine wave at all array configurations. The red line illustrates the true signal DOA.



(a) The Cross-Correlation

(b) The PHAT Processor

Figure 7.3: The effect of different GCC processors on localisation of a strong, narrowband signal – in this case a 909.5 Hz sine wave at an angle of 315 degrees relative to the array, as illustrated by the red line in each figure.

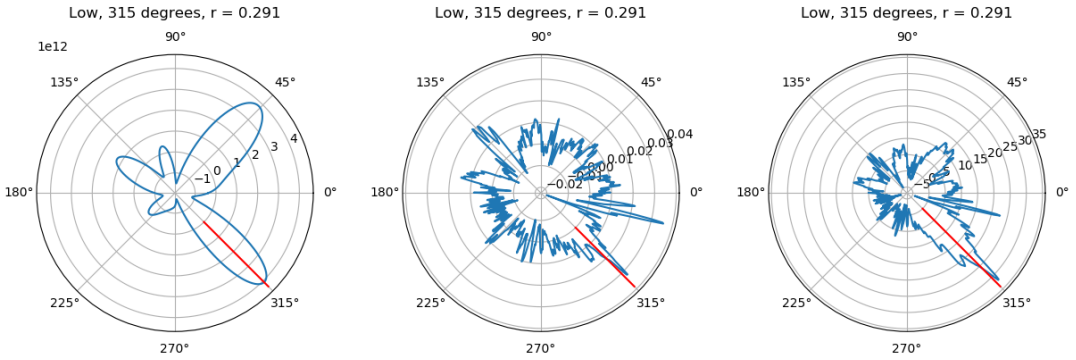
from the female kiwi, these false positives did not regularly appear. This is good, as the kiwi has already been identified as a species of particular interest.

7.1.2 Choice of Processor

In initial testing of the GCC algorithm, only the regular cross-correlation was used (i.e. $\psi(\omega) = 1 \quad \forall\omega$ in (3.17)). Later, the PHAT processor was trialled, which showed promising results, in particular by reducing the false positives discussed in 7.1.1, for example as shown in Figure 7.3.

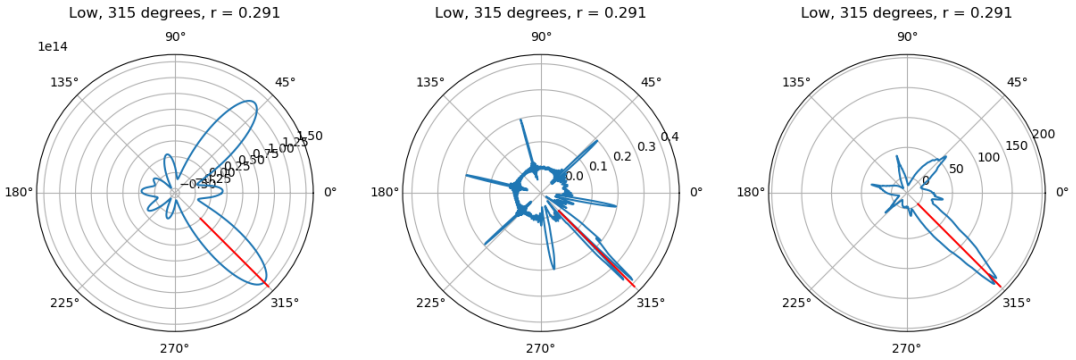
For narrowband sources such as the 909.5 Hz sine wave, the performance of the PHAT processor degrades, as the number of frequency components with no active source begin to outnumber the source component of the signal. To counteract this, the ρ-PHAT processor allows the end user to essentially scale back the amount of whitening applied to the data, to allow strong tonal components to have more of an impact, but still apply some whitening in an attempt to sharpen the peak of the cross-correlation functions. This is shown in Figure 7.4. The regular cross-correlation in Figure 7.4(a) displays a strong false positive, likely caused by the high periodicity of the signal leading to a spurious false peak in the cross-correlations which happens to line up in the $\approx 45°$ direction.

Figure 7.5(a) helps to strengthen the hypothesis that the false positive in Figure 7.4(a) was from a coincidental lining-up of the highly periodic data at that angle, rather than an unintended second source in the recording – it shows a simulation of the experiment in the noise-free case. The second peak occurs in the simulation, where there is no added noise and we can be sure of no additional source. Notice that the



(a) The Cross-Correlation (b) The PHAT Processor (c) The ρ-PHAT Processor, with ρ=0.73

Figure 7.4: The effect of different GCC processors on localisation of a strong, narrowband signal – in this case a 909.5 Hz sine wave at an angle of 315 degrees relative to the array, as illustrated by the red line in each figure.



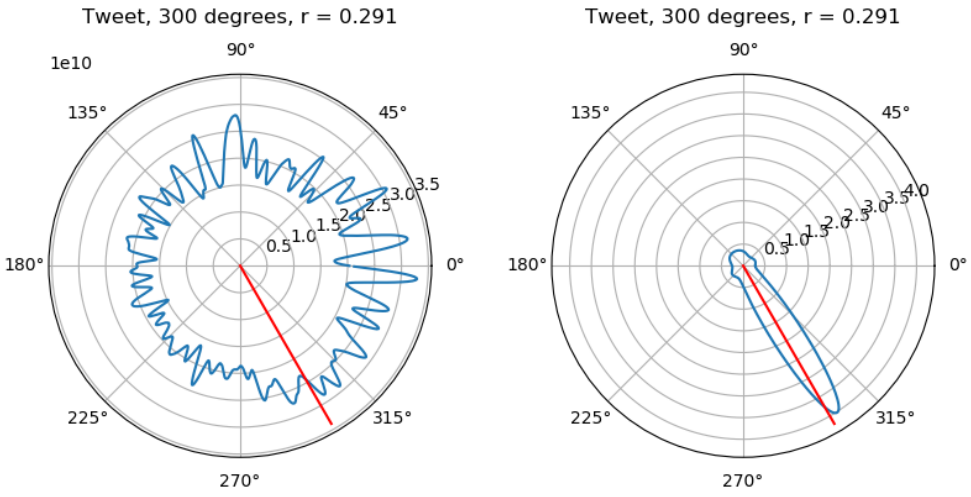
(a) The Cross-Correlation (b) The PHAT Processor (c) The ρ-PHAT Processor, with ρ=0.73

Figure 7.5: The effect of different GCC processors on localisation of a *simulated* (noise-free) strong, narrowband signal – in this case a 909.5 Hz sine wave at an angle of 315 degrees relative to the array, as illustrated by the red line in each figure.

second peak is gone in Figure 7.5(b) as in Figure 7.4(b), and that the true direction-of-arrival is accentuated when using ρ-PHAT in Figure 7.5(c).

7.2 AF-MUSIC Performance

Though we have presented no metric for accuracy, in general AF-MUSIC appeared to perform better than GCC; the peaks were sharper, there were fewer false positives, and it allowed for localisation in situations where GCC did not – for example in the situation shown in Figure 7.6, which shows the localisation of a Hihi call at 300° on an array of 0.291m radius with both the GCC and AF-MUSIC algorithms.



(a) Localisation via GCC

(b) Localisation via AF-MUSIC

Figure 7.6: This figure shows an example of AF-MUSIC providing a direction-of-arrival estimate where GCC fails to do so. (a) is the same as Figure 7.2, and (b) is the same situation analysed with the AF-MUSIC algorithm. Both the PHAT and ρ-PHAT processors performed as badly as the CC processor shown in (a) in this case.

7.3 Remarks on Computational Cost

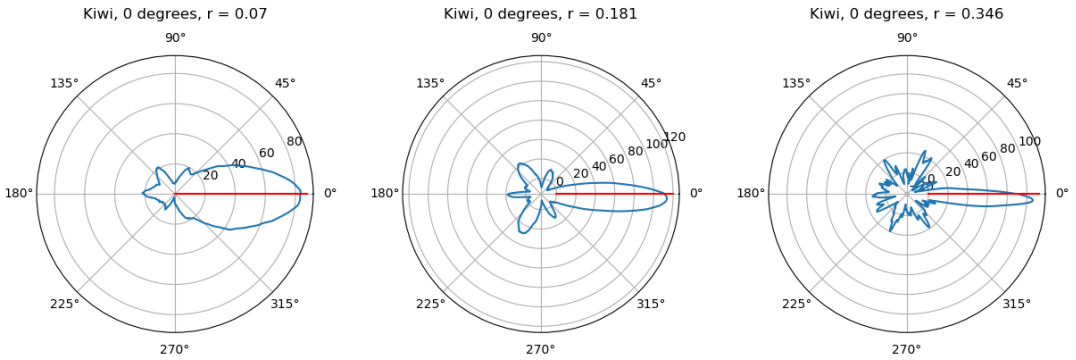
Though the AF-MUSIC algorithm showed much greater localisation abilities than any of the GCC variants tested, it also comes with the burden of much higher computational cost. If the data on M microphones of length N samples is split up into B bins for the expectation in the calculation of the eigenvectors of $R_{xx}(\omega_k)$ for the generation of the focusing matrices $T_{auto}(\omega_0, \omega_k)$, and if there are K DFT coefficients calculated, then there are least B DFTs (of runtime proportional to $\frac{N}{B} \log \frac{N}{B}$) to calculate, as well as BK eigendecompositions of Hermitian matrices of size $M \times M$. In contrast, the cross-correlation only requires two K-bin DFTs, K scalar multiplications, and one inverse DFT. The code developed to accompany this thesis is largely not optimised, however

a comparison of runtimes between GCC, MUSIC, and AF-MUSIC may be found in Appendix C.3.

7.4 Array Radius

The experiment described in Chapter 6 was conducted during the course of this thesis to determine the effect of the physical size of the array on localisation ability. In particular, we hypothesised that a larger inter-microphone distance would increase the resolution of the array. This indeed seems to be the case, however the GCC algorithm appears more sensitive to microphone separation than AF-MUSIC, as is shown in the following figures.

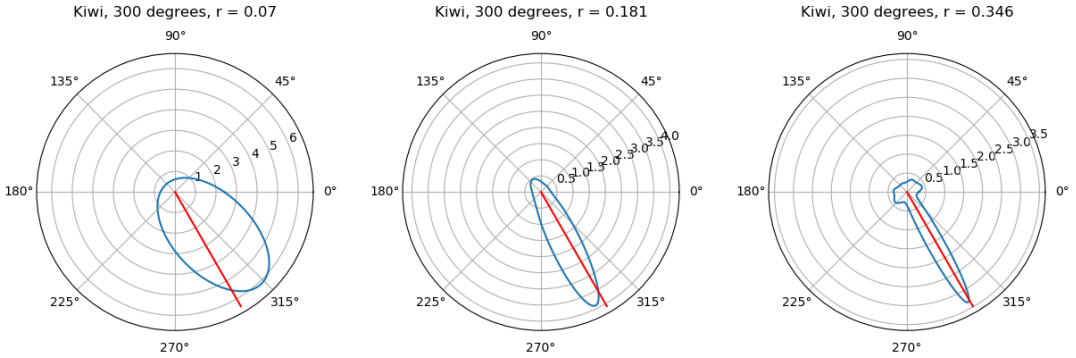
Figure 7.7 shows the output of the ρ-PHAT weighted GCC algorithm (ρ= 0.73) for a small (0.07 m), medium (0.181 m), and large (0.346 m) array arm radius – here, "arm radius" refers to the distance from the array center at which the non-central microphones are placed; refer to 6.1 for a diagram of the array geometry.



(a) Array Radius: 0.070 m (b) Array Radius: 0.181 m (c) Array Radius: 0.346 m

Figure 7.7: The effect of varying array radius on the output of the GCC algorithm with ρ-PHAT weighting (ρ=0.73). Here, the kiwi call was broadcast at 0 degrees as indicated by the red line in each graph.

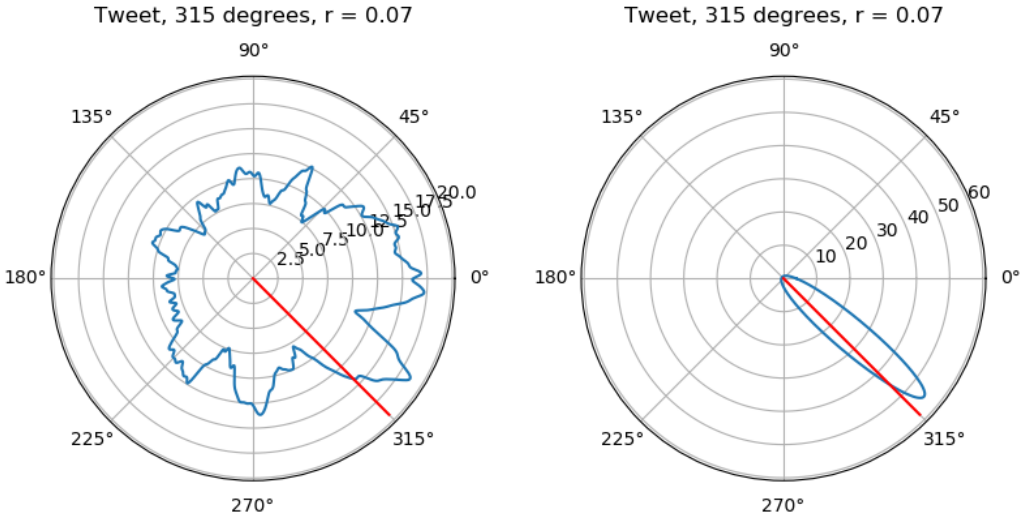
As the array radius increased, we saw an increase in precision in the DOA estimate, as illustrated by Figure 7.7. The small biases in the estimate, especially present in 7.7(c), are believed to be due to physical handling of the array between trials as discussed in Section 6.3.



(a) Array Radius: 0.070 m (b) Array Radius: 0.181 m (c) Array Radius: 0.346 m

Figure 7.8: The effect of varying array radius on the output of the AF-MUSIC algorithm. In this case, the kiwi call was broadcast at 300 degrees, as indicated by the red line in each graph.

In general, an increase in array radius also acted to increase the precision of the AF-MUSIC algorithm similarly to with GCC, as illustrated by Figure 7.8. This effect was not as profound for AF-MUSIC based localisation; it was common for the 7 cm array to lose all GCC-based localisation ability and yet still provide a reasonably precise estimate using AF-MUSIC, such as in Figure 7.9.



(a) ρ-PHAT GCC (b) AF-MUSIC

Figure 7.9: Localisation via 0.07m array of the Hihi/Stitchbird call at 315 degrees, as indicated by the red line in each figure. Figure 7.9(a) shows GCC-based localisation with the ρ-PHAT weighting ($\rho = 0.73$), and Figure 7.9(b) shows AF-MUSIC based localisation.

7.5 Future Work

Though the motivation for this work was the localisation of native birds for aid in conservation, none of methods described herein are bound to work solely on recordings of birdsong. Gray and Hioka [18] used the known harmonic structure of the kiwi call to automatically segment the data into sections in which bird's call was present and sections with only noise, allowing them to obtain an estimate of the noise spectra to be used in a maximum-likelihood GCC processor. As this thesis was part of an ongoing project[1] which includes work into the automatic detection and classification of birdsong, this could possibly prove to be a fruitful avenue for further exploration – however from the experiments conducted so far, it is anticipated that AF-MUSIC will still outperform any GCC variants.

The current array design, while it benefits from its small size by being light and portable, is too small to allow for estimation of the distance from the array to the signal source. To counteract this, two or more arrays may be used to provide approximate bearings, and then a least squares fit for their intersection may be performed. Expected challenges with this approach include inter-array synchronisation, and accurate measurement of the arrays' relative location in potentially dense native bush.

[1] The AviaNZ project, for more information visit `http://www.avianz.net/`

Appendices

Appendix A

Figures

A.1 SBR2 Visualisation

These figures are taken from slides prepared for UDRC Summer School 2017, entitled *Polynomial Matrices and Decompositions; Tutorial Examples Including Beamforming and Source Localisation*, with permission from Prof. Stephan Weiss. They provide a visual representation of an iteration of the SBR2 algorithm as presented in 4.3.1.

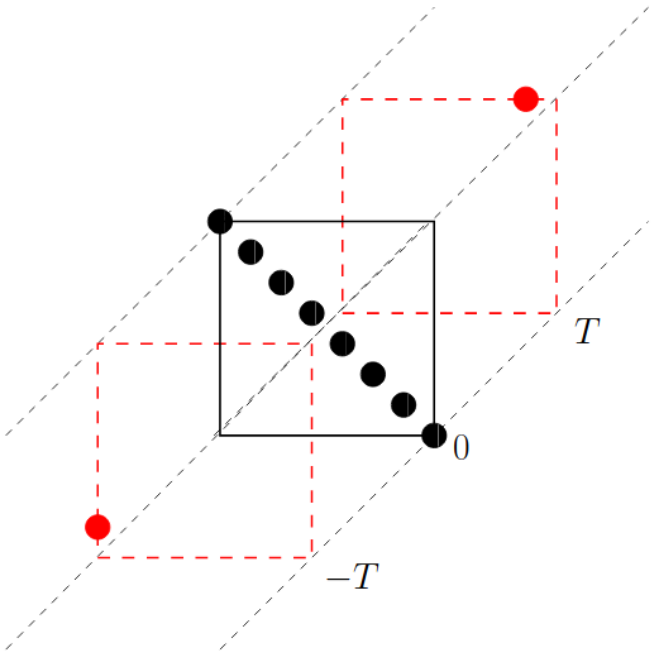


Figure A.1: The first step of the SBR2 algorithm is to locate the pair (by symmetry) of elements in the 3D array of coefficients with the highest absolute value. In this case, they have been displayed in red, and are found at time slice $\pm T$.

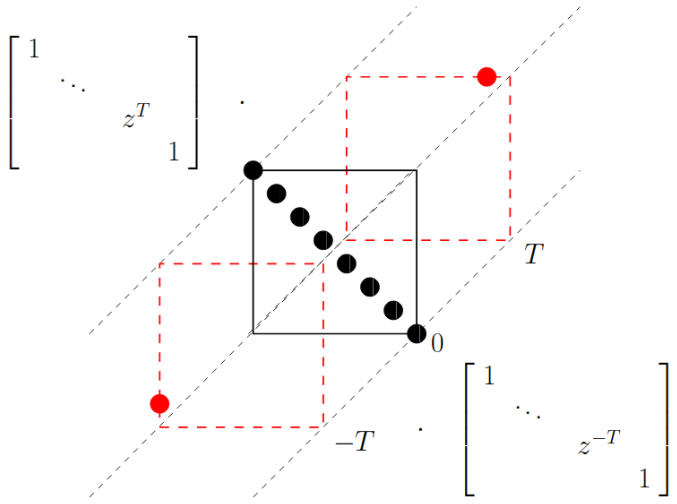


Figure A.2: Next, we use the coordinates of the red entries to create the delay matrix $D_{k,T}(z)$. The aim of these matrices is to shift the red entries onto the same plane as the black diagonal entries at the coefficient plane of exponent 0.

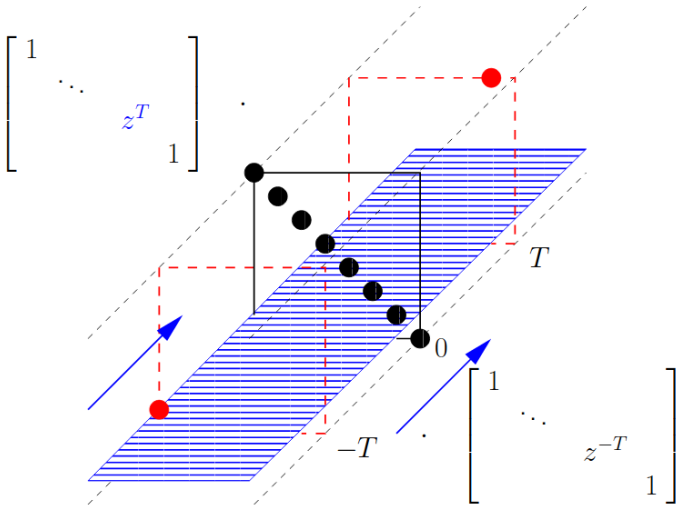


Figure A.3: Applying $D_{k,T}(z)$ to the matrix from the left, we see that it acts to shift the blue highlighted row slice forward towards the zero coefficient plane.

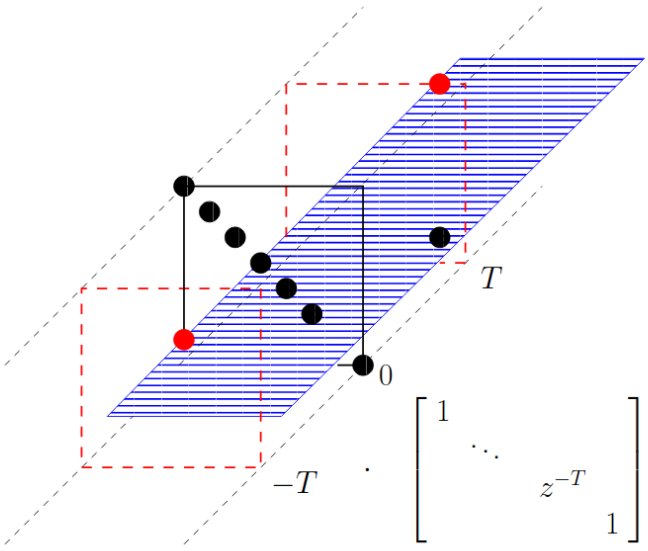


Figure A.4: After the application of $D_{k,T}(z)$, we see that the red element which was at the slice corresponding to time T is now at the constant coefficient plane. Notice also that the rear portion of the row has been shifted further out off the end - this is where the expansion of the matrix comes in; we may either truncate those entries off, or expand the array to make room to keep them. Also notice that the diagonal entry in the T^{th} row has been displaced.

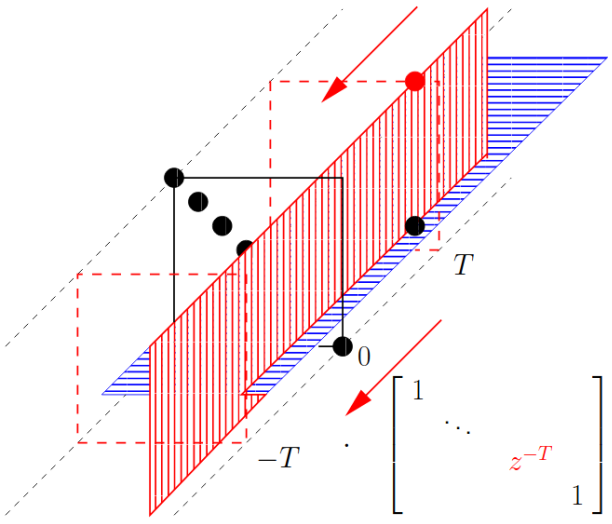


Figure A.5: Now applying $D^{P}_{k,T}(z)$ on the right, we shift the T^{th} column slice back towards the zero coefficient plane. Notice that this action restores the black diagonal entry to where is was previously.

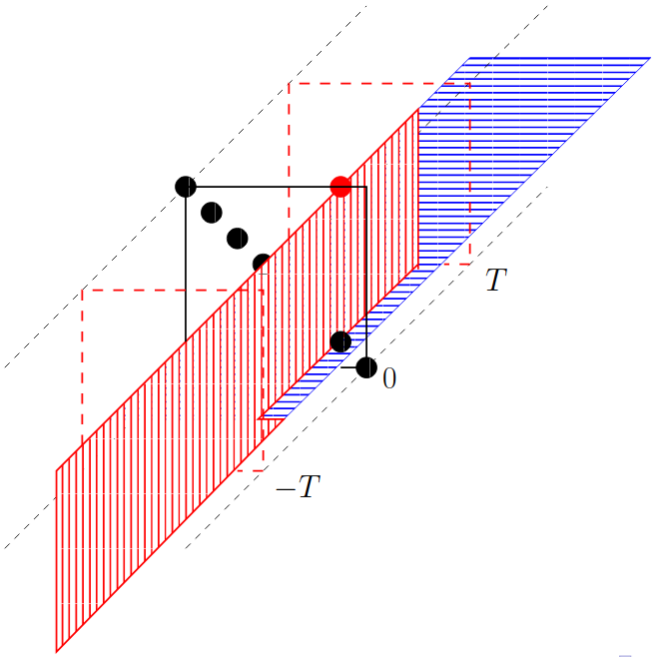


Figure A.6: The result of applying $D^P_{k,T}(z)$ to the right of the matrix.

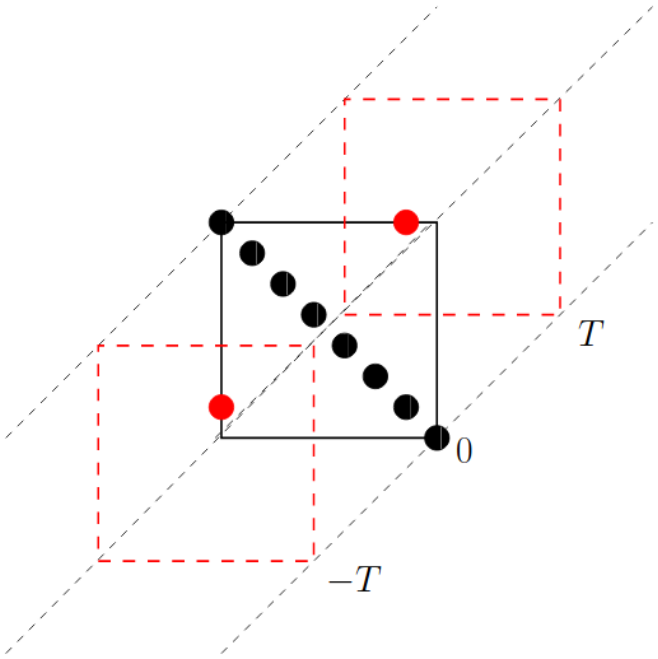


Figure A.7: The state of the matrix after completion of the delay step. The red terms have both been shifted onto the zero coefficient plane. All that remains is to apply a Jacobi transformation to shift them down onto the diagonal.

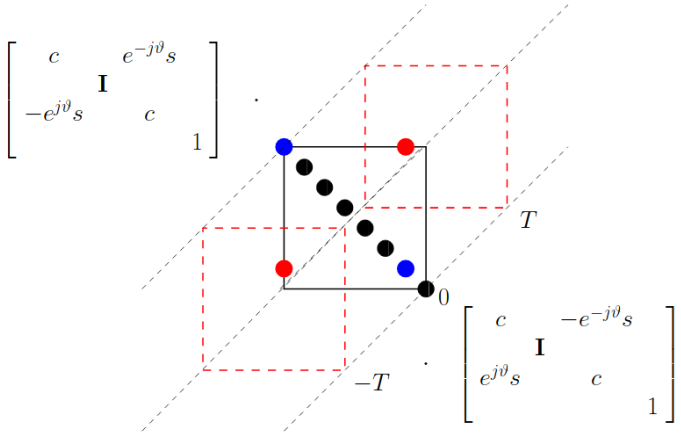


Figure A.8: Using the coordinates and details of the red entries, we now create the rotation matrix $J_{j,k}(\theta, \phi)$ which we will use to shift the red entries down to the blue highlighted positions. This step is simply a regular scalar Jacobi transformation, but must be carried out across the matrices at all lag indices.

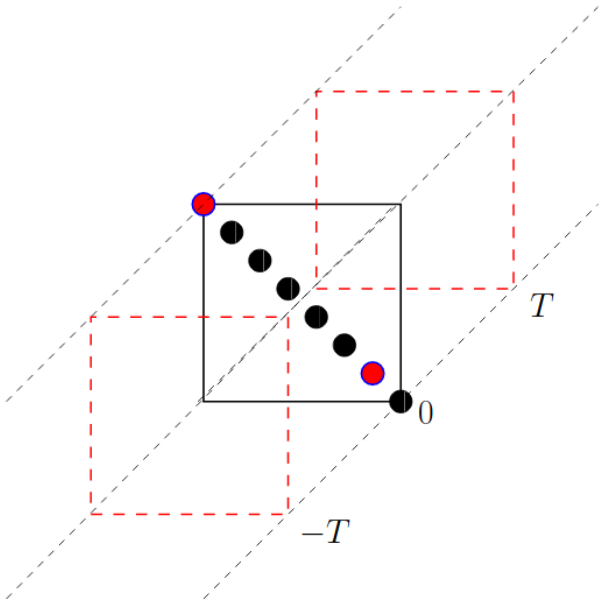


Figure A.9: The result of conjugating by $J_{j,k}(\theta, \phi)$. We see that the red entries have been rotated down onto the diagonal; this concludes one iteration of the SBR2 algorithm. Repeat these steps until the matrix is sufficiently diagonalised.

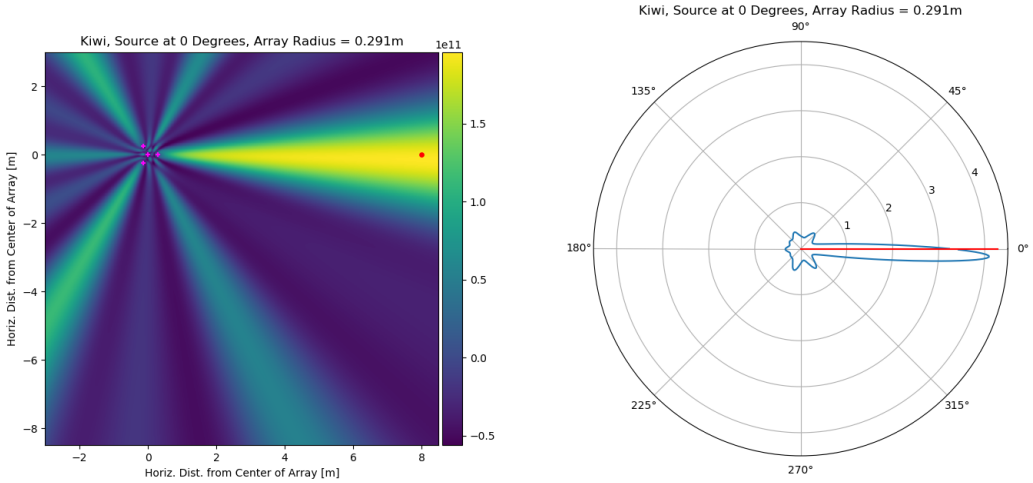
Appendix B

Image Generation

In his chapter we discuss what is done with the output of the algorithms presented in Chapters 3 & 5, including the generation of visual direction estimates.

B.1 Graphical Representation

A visual representation of the algorithm output is a useful tool for both development and quick inspection. For this, the provided `Locator` class can produce both heatmaps (e.g. Figure B.1(a)) and radial plots (e.g. Figure B.1(b)) from the output of both the GCC and the AF-MUSIC algorithm.



(a) A heatmap.

(b) A radial plot.

Figure B.1: Examples of both a heatmap (a), and a radial plot (b) generated by the `Locator` class. Both figures were generated from experimental data – in the heatmap, the source was at the position of the red dot, and the magenta '+' symbols represent the microphone positions. In the radial plot, the source was at the angle $\theta = 0$, as shown by the red line.

B.1.1 GCC Heatmap Generation

A visual representation of the GCC output was produced as follows. First, create some representation of the domain of interest; in this case, a mesh of two-dimensional points was created with `numpy.meshgrid`, with each point corresponding to a unique point in space. Then, for each point (d_x, d_y) in the domain, calculate what the observed time delays would be if the signal had indeed come from this point in space. There will be one time delay per pair of microphones, i.e. for each pair of microphones with indices j and k in an M-microphone array, we get

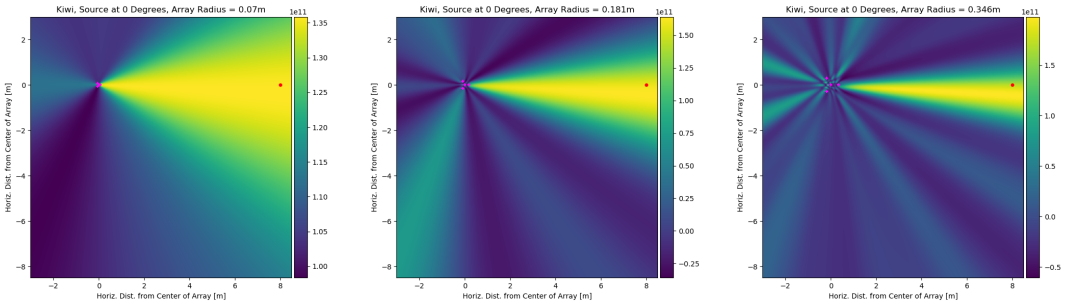
$$\tau_{j,k} = \frac{1}{c_{sound}} \left(\sqrt{\left(m_x^k - d_x\right)^2 + \left(m_y^k - d_y\right)^2} - \sqrt{\left(m_x^j - d_x\right)^2 + \left(m_y^k - d_y\right)^2} \right), \quad \text{(B.1)}$$

where m_x^l is the x-coordinate of microphone l, m_y^l its y-coordinate, and c_{sound} the speed of sound in air.

Next, evaluate the interpolated GCC function for each microphone pair at it's corresponding simulated time delay from (B.1), and sum the results. Explicitly, for each point (d_x, d_y) calculate

$$\sum_{j=1}^{M-1} \sum_{k=j+1}^{M} \hat{R}_{x_j x_k}^{(g)}(\tau_{j,k}). \quad \text{(B.2)}$$

The heatmaps produced by the `Locator` class (as seen in Figures B.2(a)–B.2(c)) are produced by calculating these sums for each point in the domain, and mapping the highest GCC sums to highest intensity colours, and *vice versa*. The point (d_x, d_y) in the domain which maximises the sum (B.2) provides an estimate of source position, akin to a maximum likelihood method. Some example heatmaps are shown in Figure B.2.

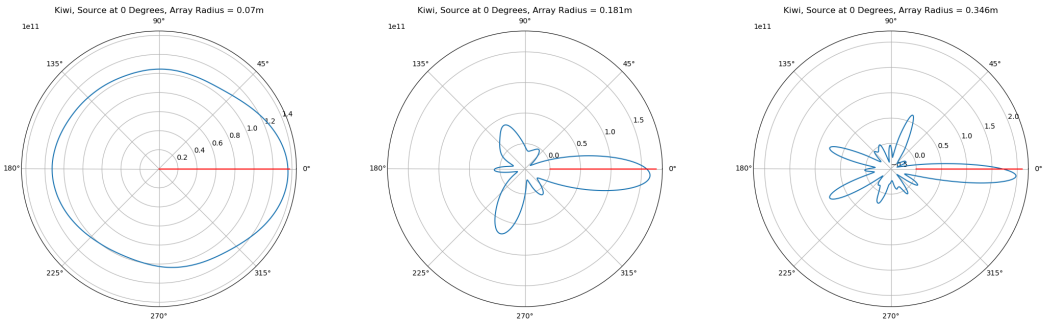


(a) Array Radius: 0.070 m (b) Array Radius: 0.181 m (c) Array Radius: 0.346 m

Figure B.2: Experimental results of GCC-based localisation of the call of the Female North Island Brown Kiwi call with varying array radii. In each image, the red dot illustrates the position of the speaker playing the call, and the pink '+' symbols represent the microphones.

B.1.2 Radial Plots

Though the heatmaps are a nice visual representation of the space in which the experiments were conducted, they illustrate the main inadequacy of the currently implemented methods on such a physically small array; the apparent inability to estimate the distance from the array to the source. From simulation, this only appears possible either in the near-field case where the source is within the space spanned by the array, or when there are multiple arrays all giving directional estimates which intersect at a unique point. To this end, instead of generating the heatmaps and evaluating Eq. (B.2) potentially millions of times over a large two-dimensional domain, we may reduce the computational load by modelling the source as a plane-wave, i.e. assuming that it is far enough from the array that the curvature of the wavefront is essentially 0 by the time it passes over the microphones. This assumption reduces the search space down to one dimension, namely the angle-of-arrival θ.



(a) Array Radius: 0.070 m (b) Array Radius: 0.181 m (c) Array Radius: 0.346 m

Figure B.3: Radial plot of GCC-based far-field localisation of the call of the Female North Island Brown Kiwi call with varying array radii. In each image, the red line illustrates the angle of the speaker playing the call.

Figures B.3(a)-B.3(c) show exactly this. Using the same data as was used to create Figure B.2, we model the time-delays as if the source were emitting a plane wave, and plot the GCC output as a function of the direction-of-arrival of that plane wave. Notice that the radial plots are qualitatively very similar to what is seen in the heatmaps when tracing a circle of fixed radius about the microphones with the eye.

B.1.3 AF-MUSIC Image Generation

The procedure for creating visual results via the AF-MUSIC algorithm is similar to the procedure for creating the images for GCC as detailed in B.1.1, up to and including creating the domain of points (d_x, d_y). At this stage, the calculation of the delays of interest differ; rather than calculate the delay between the signal reaching two microphones, for each point in the domain we calculate the time it took for the sound to travel from that point to each of the microphones in the array, i.e. for microphones

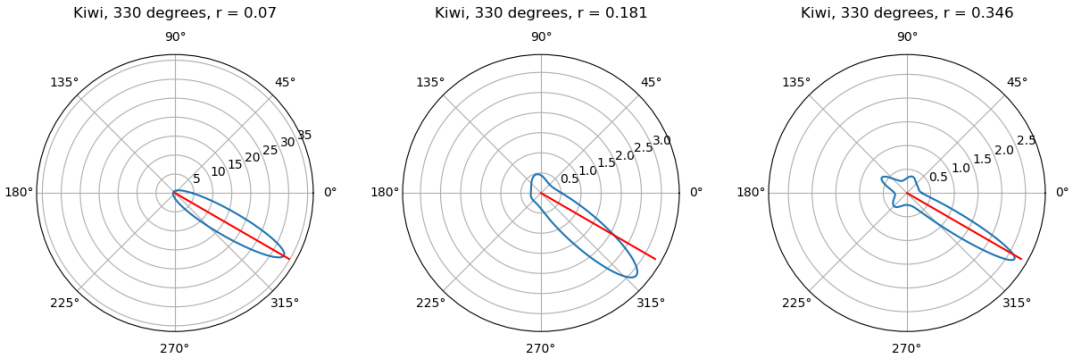
$m = 1, 2, \cdots, M$, calculate:

$$\tau_m = \frac{1}{c_{sound}} \left(\sqrt{(m_x - d_x)^2 + (m_y - d_y)^2} \right). \tag{B.3}$$

We then use these time delays to create the steering vector $\mathbf{a}_\theta(\omega)$ as follows:

$$\mathbf{a}_\theta(\omega) = \begin{bmatrix} e^{-i\omega\tau_1} & e^{-i\omega\tau_2} & \dots & e^{-i\omega\tau_M} \end{bmatrix}^T, \tag{5.2}$$

and then proceed to calculate the AF-MUSIC output $P_{AF}(\theta, \omega_0)$ as per (5.22). This procedure is done for every point in the domain, and the output is plotted either as a heatmap or a radial plot, depending on the dimensionality of the domain of interest. Some example radial plots of AF-MUSIC output are shown in Figures B.4(a)-B.4(c).



(a) Array Radius: 0.070 m (b) Array Radius: 0.181 m (c) Array Radius: 0.346 m

Figure B.4: Radial plot of AF-MUSIC-based far-field localisation of the call of the Female North Island Brown Kiwi call with varying array radii. In each image, the red line illustrates the angle of the speaker playing the call.

Appendix C

Code Demonstration

The `Locator` class

During the course of this degree, a Python package was developed for use in Acoustic Source Localisation, with the main class being coined the `Locator` class. The code is hosted at `https://github.com/alexW335/Locator`. In this chapter, we will provide a demonstration of its use, similar to the `Demo.ipynb` Jupyter notebook also hosted in the GitHub repository. The code that follows was run in an ipython shell with the following imports:

```
%pylab
from scipy import signal
from scipy.io import wavfile as wav
import locator
```

C.1 Simulating and Loading Data

The `locator` class has the functionality to artificially shift a signal to simulate as if it had come from some position in space, as well as to load in actual array data to process. To begin the demonstration, we will create a mono 'WAVE' file comprised of white noise, and then artificially shift it as if it had come from the direction-of-arrival $\theta = \pi/5$.

```
noise = random.normal(0., 1., size=44100//5)
noise *= 2**15/max(noise)
wav.write(``noisedemo.wav'', 44100, noise)
```

Now that we have our source file saved as "noisedemo.wav", we instantiate a Locator object, and use it to create our artificial array data, like so:

```
loc = locator.Locator(mic_locations=locator.UCA(n=4, r=0.346,
                                                centerpoint=True,
                                                show=True))
```

When using the helper function `locator.UCA` to simulate a Uniform Circular Array, we may pass in the named keyword `show=True` to produce a visual representation of the microphone positions, as shown in Figure C.1

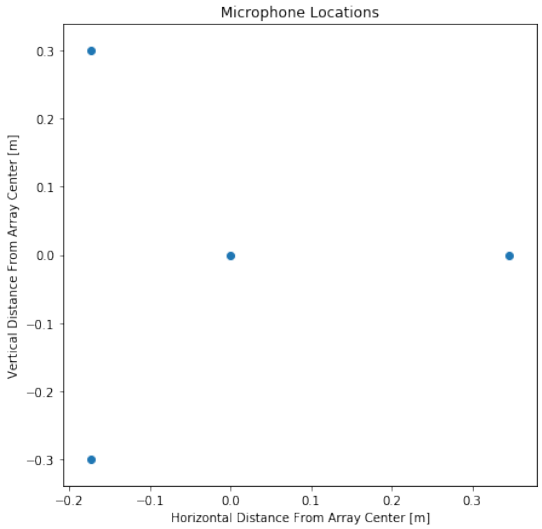


Figure C.1: The position of the simulated microphones, shown by the blue dots.

Now that we have a `Locator` object, we may make use of the `shift_sound` method to create our artificial data. The `shift_sound` method actually takes in a pair of coordinates (x, y) as its position argument, because some of the methods implemented in the `locator` work for near-field sources as well as far-field. In this case, we just shift the source very far away.

```
# Define an (x,y) tuple for the source position
source_position = (100000*cos(pi/5), 100000*sin(pi/5))

# Artificially shift the source to that position,
# save as ``noisedemoSHIFTED.wav''.
loc.shift_sound(source_position, "noisedemo.wav",
                "noisedemoSHIFTED.wav", noisescale=0.0)
```

You may notice that the `shift_sound` method has a "noisescale" parameter. You can introduce artificial noise in the data with this parameter – it generates Gaussian i.i.d. noise with mean 0 and variance `noisescale` times the variance of the source file.

```
# Load the newly-shifted file.
loc.load("noisedemoSHIFTED.wav", GCC_processor="CC")
```

The `load` method also generates the weighted cross-correlation functions using the weighting specified by the `GCC_processor` keyword. Options available are: CC, PHAT, pPHAT (for ρ-PHAT), RIR, HB, and SCOT. You can avoid calculation of the cross-correlations (e.g. if only utilising MUSIC-based algorithms) by passing the parameter `do_FFTs=False`.

C.2 Localisation

The next step is to try and localise our artificial array data using the `Locator` functionality. GCC-based methods available are:

- display_heatmap()
- display_radial()

MUSIC-based methods available are:

- display_radMUSIC(frequency)
- display_MUSICheatmap(frequency)

and AF-MUSIC-based methods available are:

- AF_MUSIC()

C.2.1 GCC-Based Methods

To display GCC-based heatmaps (discussed in B.1.1), we use the `display_heatmap` method as follows:

```
loc.display_heatmap()
```

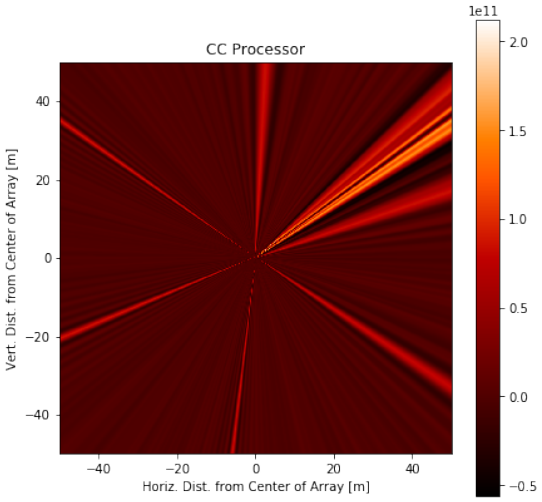


Figure C.2: Example output of the `display_heatmap` method.

For a radial plot (discussed in B.1.2), we use the `display_radial` method. This method has one required parameter `radius`, which is the radius from the array center at which to evaluate the GCC. The method is called as so:

```
loc.display_radial(1000)
```

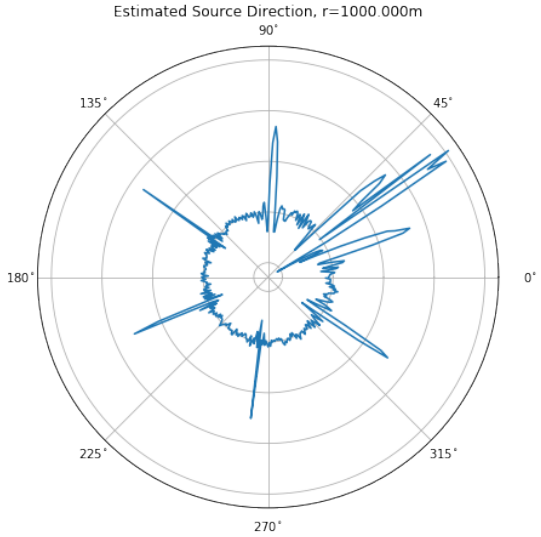


Figure C.3: Example output of the `display_radial` method.

C.2.2 MUSIC-Based Methods

Both `display_MUSICheatmap` and `display_radMUSIC` require the selection of a frequency at which to evaluate the MUSIC algorithm. For this demonstration, I picked an arbitrary frequency to use (500 Hz), as the white noise signal should have equal power at all frequencies. For MUSIC, there is the option of a two-dimensional heatmap like so:

```
loc.display_MUSICheatmap(500, xrange=(0,40), yrange=(0,40), xstep=0.01,
                         ystep=0.01)
```

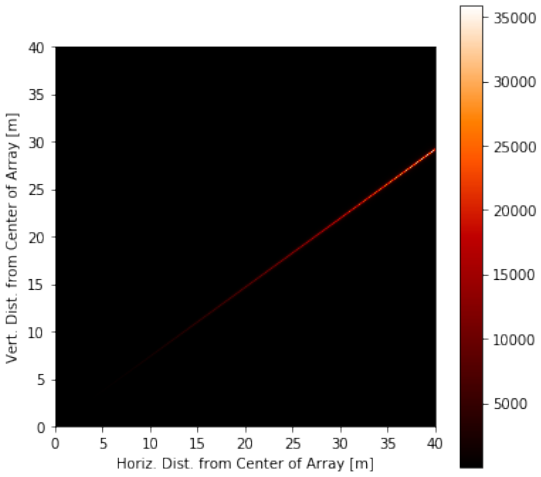


Figure C.4: Example output of the `display_MUSICheatmap` method with `frequency` = 500.

or the radial plots for far-field DOA estimation, as follows:

```
loc.display_radMUSIC(500)
```

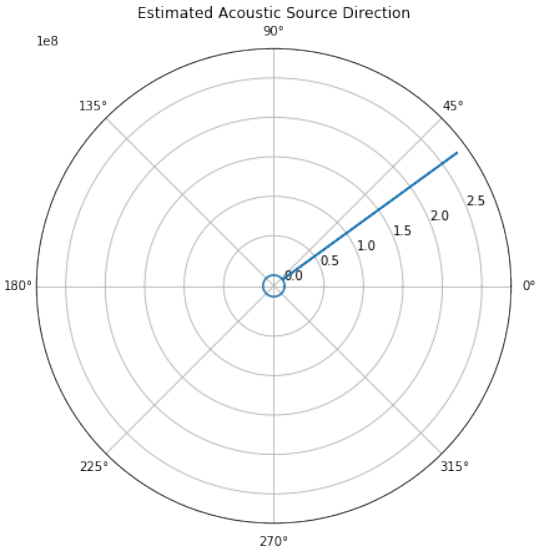


Figure C.5: Example output of the `display_radMUSIC` method with `frequency` = 500.

AF-MUSIC

The AF-MUSIC algorithm does not require the selection of a single frequency to work on, and as such the related methods do not require any arguments. To make a radial plot of DOA estimation, we use the `AF_MUSIC` method as follows:

```
loc.AF_MUSIC()
```

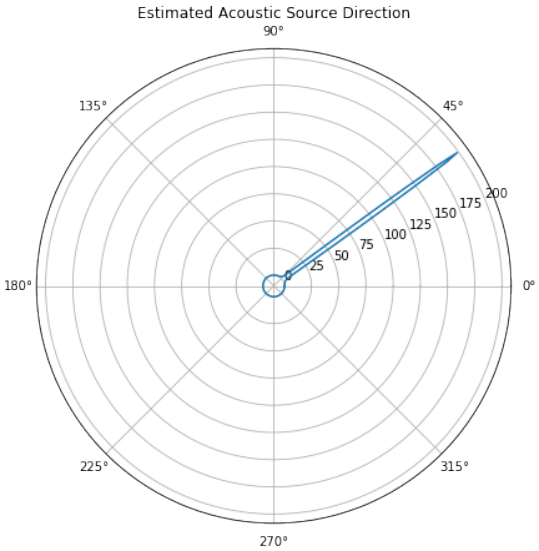


Figure C.6: Example output of the `AF_MUSIC` method.

C.2.3 Example Using Real Data

Both algorithms appear to work well on the simulation with no noise. But do they work for real data? In practice, the `Locator` is used in exactly the same way. The file "kdemo.wav" was recorded as part of a larger experiment on how array size effects the localisation ability of the array, as described in Chapter 6. It was captured with a 0.346 m radius 4-element UCA with one microphone in the center (the same shape as in the simulations above). The source was placed at an angle of 0 degrees from the array at a distance of 8 m. As before, the first thing we must do is instantiate a `Locator`, however this time we must be sure to place the microphones in the same positions as they were during recording of the data.

```
# Load the file
l_exp = locator.Locator(mic_locations=locator.UCA(n=4, r=0.346,
                        centerpoint=True))
l_exp.load("kdemo.wav")
```

To begin with, we will try the GCC-based heatmap and radial plot. First we generate the heatmap as follows:

```
l_exp.display_heatmap(xrange=(-10,10), yrange=(-10,10))
```

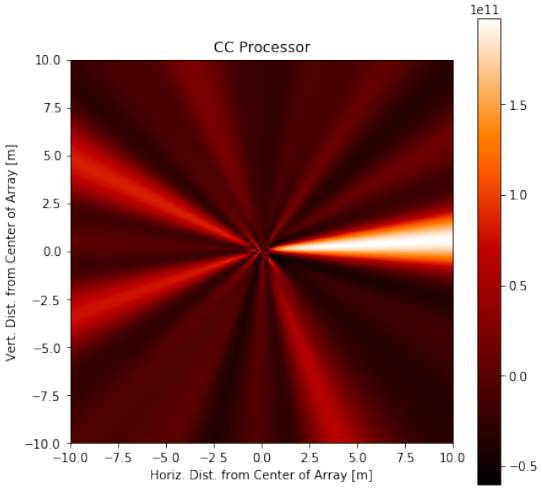


Figure C.7: The output of the `display_heatmap` method for the "kdemo.wav" file.

For the `display_radial` method, we must pass in a radius at which to evaluate the GCC. We know *a priori* that the source was 8 meters away from the array, so we shall pass in this distance for the `radius` parameter:

```
l_exp.display_radial(8)
```

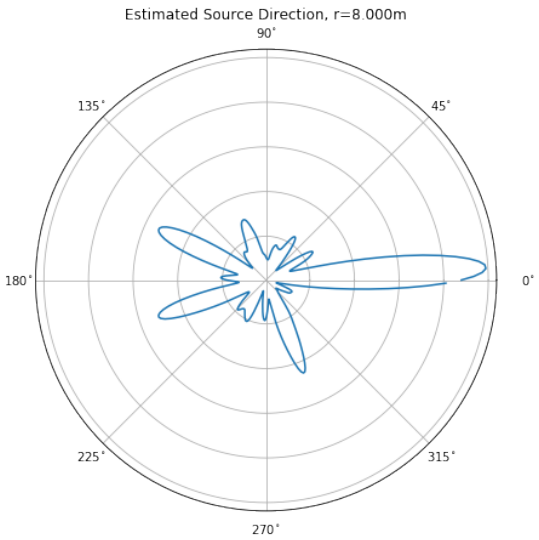


Figure C.8: The output of the `display_radial` method for the "kdemo.wav" file.

Now we shall try the MUSIC-based algorithms – but at which frequency do we evaluate the MUSIC spectrum? One suggestion is to look at the Fourier transform of the recorded call of the female North Island Brown Kiwi; the frequency with the highest magnitude seems to be a sensible choice for use, as it contributes the most to the data.

```
plot(fft.rfftfreq(l_exp.data[:,0].shape[0])*l_exp.sample_rate,
                  np.abs(fft.rfft(l_exp.data[:,0])))
xlabel("Frequency [Hz]")
```

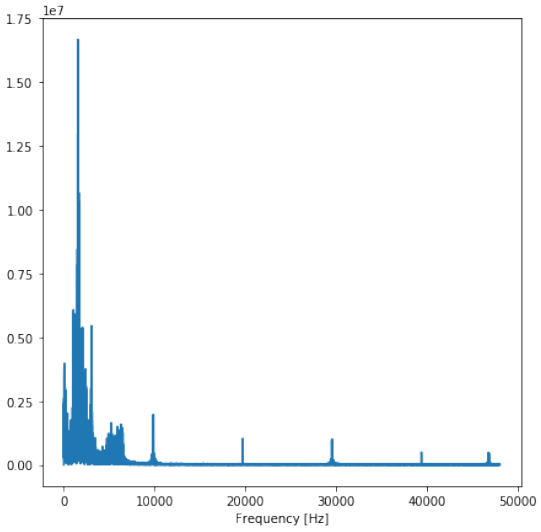


Figure C.9: The Fourier transform of the kiwi call.

Notice how the total signal power is spread out over a large range of frequencies.

```
frq = fft.rfftfreq(l_exp.data[:,0].shape[0])[argmax(
                    np.abs(fft.rfft(l_exp.data[:,0])))]*l_exp.sample_rate
print("The frequency with highest power is", frq, "Hz")
>>> The frequency with highest power is 1559.2987858217573 Hz
```

We will then use this frequency as the choice for the MUSIC algorithm, like so:

```
l_exp.display_radMUSIC(frq)
```

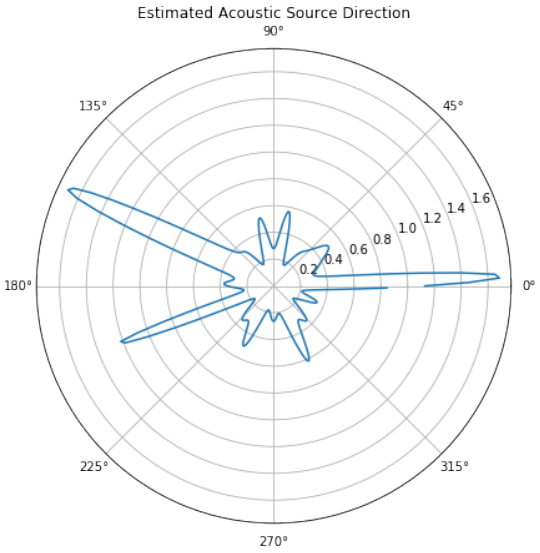


Figure C.10: The output of the `display_radMUSIC` method for the experimental data, evaluated at ∼1559 Hz.

For the ∼1559 Hz component of the signal, it appears that the kiwi call at 0 degrees is not the only active source; there are multiple peaks in the output. Rather than pick one frequency for localisation, by using AF-MUSIC we may combine the information available at all frequencies in the data, like so:

```
l_exp.AF_MUSIC()
```

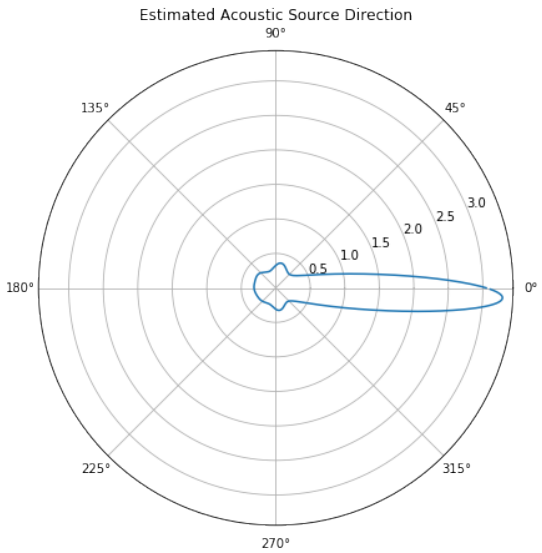


Figure C.11: The output of the `AF_MUSIC` method for the experimental data.

C.3 Runtime Comparison

As AF-MUSIC combines the information available at all frequencies, there is a lot more computation involved. Indeed, regular MUSIC takes approximately 34 milliseconds per run:

```
%%timeit
_ = l_exp.display_radMUSIC(frq, shw=False)
>>> 33.9 ms +- 2.65 ms per loop (mean +- std. dev. of 7 runs,
    10 loops each)
```

Whereas AF-MUSIC takes around 3.9 seconds:

```
%%timeit
_ = l_exp.AF_MUSIC(shw=False)
>>> 3.66 s +- 11.3 ms per loop (mean +- std. dev. of 7 runs,
    1 loop each)
```

For comparison, the cross-correlation based method only takes around 9 milliseconds.

```
%%timeit
_ = l_exp.display_radial(8, shw=False)
>>> 8.91 ms +- 683 us per loop (mean +- std. dev. of 7 runs,
    100 loops each)
```

Bibliography

[1] BirdLife International 2016. *Apteryx haastii*. 2016. DOI: http://dx.doi.org/10.2305/IUCN.UK.2016-3.RLTS.T22678132A92756666.en. URL: https://www.iucnredlist.org/species/22678132/92756666.

[2] BirdLife International 2016. *Apteryx australis*. 2016. DOI: http://dx.doi.org/10.2305/IUCN.UK.2016-3.RLTS.T22678122A92756034.en. URL: https://www.iucnredlist.org/species/22678122/92756034.

[3] BirdLife International 2017. *Apteryx owenii*. 2017. DOI: http://dx.doi.org/10.2305/IUCN.UK.2016-3.RLTS.T22678129A92756395.en. URL: https://www.iucnredlist.org/species/22678129/92756395.

[4] BirdLife International 2017. *Apteryx mantelli*. 2017. URL: https://www.iucnredlist.org/species/45353580/119177586.

[5] BirdLife International 2017. *Apteryx rowi*. 2017. DOI: http://dx.doi.org/10.2305/IUCN.UK.2017-3.RLTS.T22732871A119169794.en. URL: https://www.iucnredlist.org/species/22732871/119169794.

[6] Hugh Robertson and Rogan Colbourne. *Kiwi Best Practice Manual*. Tech. rep. Department of Conservation, 2017. URL: https://www.doc.govt.nz/Documents/science-and-technical/sap262entire.pdf.

[7] Kiwis for kiwi. *Call count monitoring*. 2019. URL: https://www.kiwisforkiwi.org/resources/call-count-monitoring/.

[8] Darren Kidney et al. "An efficient acoustic density estimation method with human detectors applied to gibbons in Cambodia". In: *PLoS ONE* 11.5 (2016), pp. 1–16. ISSN: 19326203. DOI: 10.1371/journal.pone.0155066.

[9] A. Rawlinson. *The Defense of London 1915-1918*. 2nd ed. London: Andrew Melrose Ltd., 1923, p. 267.

[10] Brigitte Röder et al. "Improved auditory spatial tuning in blind humans". In: *Nature* 400.6740 (1999), pp. 162–166. ISSN: 00280836. DOI: 10.1038/22106.

[11] Patrice Voss et al. "Early- and Late-Onset Blind Individuals Show Supra-Normal Auditory Abilities in Far-Space". In: *Current Biology* 14.1 (2004), pp. 1734–1738. ISSN: 00114162. DOI: 10.1016/j.

[12] Hunter C. Jr. Harris, Meyer Leifer, and Desmond W. Cawood. *Modified Cross-Correlation Radio System and Method*. 1960.

[13] Norman L. Harvey. *Radio Navigation System*. 1954.

[14] James W. Cooley and John W Tukey. "An Algorithm for the Machine Calculation of Complex Fourier Series". In: *Mathematics of Computation* 19.90 (1964), pp. 297–301.

[15] Peter R. Roth. "Effective measurements using digital signal analysis". In: *IEEE Spectrum* 8.4 (1971), pp. 62–70. ISSN: 00189235. DOI: 10.1109/MSPEC.1971.5218046.

[16] Charles H. Knapp and G. Clifford Carter. "The Generalized Correlation Method for Estimation of Time Delay". In: *IEEE Transactions on Acoustics, Speech, and Signal Processing* 24.4 (1976), pp. 320–327. ISSN: 00963518. DOI: 10.1109/TASSP.1976.1162830.

[17] Hong Wang and P. Chu. "Voice source localization for automatic camera pointing system in videoconferencing". In: *Proceedings of 1997 Workshop on Applications of Signal Processing to Audio and Acoustics* 2.2 (1997), pp. 187–190. ISSN: 1520-6149. DOI: 10.1109/ASPAA.1997.625639.

[18] C Gray and Y Hioka. "Direction of arrival estimation of kiwi call in noisy and reverberant bush". In: *2014 IEEE Sensors Applications Symposium, SAS 2014 - Proceedings* 1 (2014), pp. 258–262. DOI: 10.1109/SAS.2014.6798957.

[19] G. Clifford Carter, Albert H. Nuttall, and Peter G. Cable. "The Smoothed Coherence Transform". In: *Proceedings of the IEEE* 61.10 (1973), pp. 1497–1498. ISSN: 15582256. DOI: 10.1109/PROC.1973.9300.

[20] Ritu and Sanjeev Kumar Dhull. "A Comparison of Generalized Cross-Correlation Methods for Time Delay Estimation". In: *IUP Journal of Telecommunications* 8.4 (2016).

[21] Ralph O. Schmidt. "Multiple emitter location and signal parameter estimation". In: *IEEE Transactions on Antennas and Propagation* AP-34.3 (1986), pp. 276–280. ISSN: 0096-1973. DOI: 10.1109/9780470544075.ch2.

[22] Takuma Otsuka. "Bayesian Microphone Array Processing". PhD thesis. Kyoto University, 2014. URL: https://doi.org/10.14989/doctor.k18412.

[23] H Wang. "Coherent Signal-Subspace Processing for the Detection Wide-Band Sources". In: *Ieee Transactions on Acoustics, Speech and Signal Processing* (1985), pp. 823–831.

[24] Sylvie Icart et al. "Some Properties of Laurent Polynomial Matrices". In: (2012).

[25] John G. McWhirter et al. "An EVD algorithm for para-Hermitian polynomial matrices". In: *IEEE Transactions on Signal Processing* 55.5 II (2007), pp. 2158–2169. ISSN: 1053587. DOI: 10.1109/TSP.2007.893222.

[26] Mohamed A. Alrmah, Stephan Weiss, and Sangarapillai Lambotharan. "An extension of the MUSIC algorithm to broadband scenarios using a polynomial eigenvalue decomposition". In: *European Signal Processing Conference* Eusipco (2011), pp. 629–633. ISSN: 22195491.

[27] Stephan Weiss et al. "Broadband angle of arrival estimation methods in a polynomial matrix decomposition framework". In: *2013 5th IEEE International Workshop on Computational Advances in Multi-Sensor Adaptive Processing, CAMSAP 2013* (2013), pp. 109–112. DOI: 10.1109/CAMSAP.2013.6714019.

[28] A. Barabell. "Improving the resolution performance of eigenstructure-based direction-finding algorithms". In: *ICASSP '83. IEEE International Conference on Acoustics, Speech, and Signal Processing* 8 (1983), pp. 336–339. DOI: 10.1109/icassp.1983.1172124.

[29] B. Friedlander. "Direction finding using an interpolated array". In: *International Conference on Acoustics, Speech, and Signal Processing* (1990), pp. 2951–2954. ISSN: 15206149. DOI: 10.1109/ICASSP.1990.116245.

[30] William Coventry et al. "Polynomial Root-MUSIC Algorithm for Efficient Broadband Direction Of Arrival Estimation". In: *Sensor Signal Processing for Defence Conference*. 2017, pp. 5–9. ISBN: 9781538616635.

[31] D W Tufts and R Kumaresan. "Estimating the angle of arrival of multiple plane waves". In: *IEEE Transactions on Aerospace and Electronic Systems* 19.1 (1983), pp. 134–139.

[32] Fu Li, R J Vaccaro, and D W Tufts. "Min-Norm linear prediction for arbitrary sensor arrays". In: *Acoustics, Speech, and Signal ...* 2 (1989), pp. 2613–2616. ISSN: 07367791.

[33] Mostafa Kaveh and Arthur J. Barabell. "The Statistical Performance of the MUSIC and the Minimum-Norm Algorithms in Resolving Plane Waves in Noise". In: *IEEE Transactions on Acoustics, Speech, and Signal Processing* 34.3 (1986), p. 633. ISSN: 00963518. DOI: 10.1109/TASSP.1986.1164841.

[34] Xiao Liang Xu and Kevin M. Buckley. "Bias and Variance of Direction-of-Arrival Estimates from MUSIC, MIN-NORM, and FINE". In: *IEEE Transactions on Signal Processing* 42.7 (1994), pp. 1812–1816. ISSN: 19410476. DOI: 10.1109/78.298288.

[35] Fu Li and R J Vaccaro. "Statistical Comparison of Subspace Based DOA Estimation Algorithms in the Presence of Sensor Errors". In: *Fifth ASSP Workshop on Spectrum Estimation and Modeling, 1990*. IEEE, 1990.

[36] A Paulraj, R Roy, and T Kailath. "Estimation Of Signal Parameters Via Rotational Invariance Techniques - Esprit". In: *Circuits, Systems and Computers, 1985. Nineteeth Asilomar Conference on* (1985), pp. 83–89. ISSN: 1058-6393. DOI: 10.1109/ACSSC.1985.671426.

[37] Ts Dhope. "Application of MUSIC, ESPRIT and ROOT MUSIC in DOA Estimation". In: *Ieee.Hr* September (2010), pp. 1–5.

[38] Ousmane Abdoulaye Oumar, Ming Fei Siyau, and Tariq P. Sattar. "Comparison between MUSIC and ESPRIT direction of arrival estimation algorithms for wireless communication systems". In: *The First International Conference on Future Generation Communication Technologies* (2012), pp. 99–103. DOI: 10.1109/FGCT.2012.6476563.

[39] V. C. Soon and Y.F. Huang. "An Analysis of ESPRIT Under Random Sensor Uncertainties". In: *IEEE Transactions on Signal Processing* 40.9201603 (1992).

[40] Yu Tao, Tong Mu, and Yaoliang Song. "Time Reversal Microwave Imaging Method Based on SF-ESPRIT for Breast Cancer". In: *IEEE International Conference on Computer and Communications*. 2017, pp. 1–5. ISBN: 9781509063529.

[41] M. Sun et al. "Road surface layers geometric parameters estimation by ground penetrating radar using Estimation of Signal Parameters via Rotational Invariance Techniques method". In: *IET Radar, Sonar and Navigation* 10.3 (2016), pp. 603–609. ISSN: 17518784. DOI: 10.1049/iet-rsn.2015.0374.

[42] Michael D. Zoltowski and C. P. Mathews. "Closed-form 2D angle estimation with uniform circular array via phase mode excitation and ESPRIT". In: *Signals, Systems and Computers, 1993. 1993 Conference Record of The Twenty-Seventh Asilomar Conference on* (1993), pp. 169–173. ISSN: 10586393.

[43] Michael D. Zoltowski, Martin Haardt, and Cherian P. Mathews. "Closed-form 2-D angle estimation with rectangular arrays in element space or beamspace via unitary ESPRIT". In: *IEEE Transactions on Signal Processing* 44.2 (1996), pp. 316–328. ISSN: 1053587X. DOI: 10.1109/78.485927.

[44] Marlene Harter, Andreas Ziroff, and Thomas Zwick. "Three-Dimensional Radar Imaging by Digital Beamforming". In: *European Radar Conference*. June 2017. 2011. ISBN: 9782874870255.

[45] Hang Yu. "Beamforming on Mobile Devices : A First Study". PhD thesis. 2011.

[46] Wonil Roh et al. "Millimeter-Wave Beamforming as an Enabling Technology for 5G Cellular Communications : Theoretical Feasibility and Prototype Results". In: *IEEE Communications Magazine* February (2014), pp. 106–113.

[47] Yongjiu Du et al. "iBeam : Intelligent Client-Side Multi-User Beamforming in Wireless Networks". In: *IEEE INFOCOM 2014*. April. 2014. ISBN: 9781479933600. DOI: 10.1109/INFOCOM.2014.6848009.

[48] Shuangfeng Han, Zhikun Xu, and Corbett Rowell. "Large-Scale Antenna Systems with Hybrid Analog and Digital Beamforming for Millimeter Wave 5G". In: *IEEE Communications Magazine* January (2015), pp. 186–194.

[49] Michael S. Brandstein and H. F. Silverman. "A practical methodology for speech source localization with microphone arrays". In: *Computer Speech & Language* 11.2 (1997), pp. 91–126. ISSN: 08852308. DOI: 10.1006/csla.1996.0024.

[50] Shmulik Markovich, Sharon Gannot, and Israel Cohen. "Multichannel eigenspace beamforming in a reverberant noisy environment with multiple interfering speech signals". In: *IEEE Transactions on Audio, Speech and Language Processing* 17.6 (2009), pp. 1071–1086. ISSN: 15587916. DOI: 10.1109/TASL.2009.2016395.

[51] Cha Zhang et al. "Maximum likelihood sound source localization and beamforming for directional microphone arrays in distributed meetings". In: *IEEE Transactions on Multimedia* 10.3 (2008), pp. 538–548. ISSN: 15209210. DOI: 10.1109/TMM.2008.917406.

[52] Sharon Gannot, David Burshtein, and Ehud Weinstein. "Signal enhancement using beamforming and nonstationarity with applications to speech". In: *IEEE Transactions on Signal Processing* 49.8 (2001), pp. 1614–1626. ISSN: 1053587X. DOI: 10.1109/78.934132.

[53] Chunwei J. Lam and Andrew C. Singer. "Bayesian beamforming for DOA uncertainty: Theory and implementation". In: *IEEE Transactions on Signal Processing* 54.11 (2006), pp. 4435–4445. ISSN: 1053587X. DOI: 10.1109/TSP.2006.880257.

[54] Jean Jacques Fuchs. "On the application of the global matched filter to DOA estimation with uniform circular arrays". In: *IEEE Transactions on Signal Processing* 49.4 (2001), pp. 702–709. ISSN: 1053587X. DOI: 10.1109/78.912914.

[55] V Krishnaveni and T Kesavamurthy. "Beamforming for Direction-of-Arrival (DOA) Estimation-A Survey". In: *International Journal of Computer Applications* 61.11 (2013), pp. 975–8887. ISSN: 09758887. DOI: 10.5120/9970-4758.

[56] Marija Agatonović et al. "Application of artificial neural networks for efficient high-resolution 2D DOA estimation". In: *Radioengineering* 21.4 (2012), pp. 1178–1186. ISSN: 12102512.

[57] Muhammed Fahri Unlersen and Ercan Yaldiz. "Direction of Arrival Estimation by Using Artificial Neural Networks". In: *2016 European Modelling Symposium*. 2016, pp. 242–245. DOI: 10.1109/EMS.2016.51.

[58] Y. T. Chan and K. C. Ho. "A Simple and Efficient Estimator for Hyperbolic Location". In: *IEEE Transactions on Signal Processing* 42.8 (1994), pp. 1905–1915. ISSN: 19410476. DOI: 10.1109/78.301830.

[59] Sharath Adavanne, Archontis Politis, and Tuomas Virtanen. "Direction of arrival estimation for multiple sound sources using convolutional recurrent neural network". 2017. URL: http://arxiv.org/abs/1710.10059.

[60] Soumitro Chakrabarty and Emanufffdfffdl. A. P. Habets. "Broadband DOA estimation using Convolutional neural networks trained with noise signals". In: *2017 IEEEWorkshop on Applications of Signal Processing to Audio and Acoustics*. 2017, pp. 136–140. ISBN: 9781538616314. URL: http://arxiv.org/abs/1705.00919.

[61] P.P. Vaidyanathan. "Multirate Systems and Filter Banks". In: (1993), p. 911. URL: https://books.google.nl/books?id=6lMKQak%5C_VgkC.

[62] Russel H. Lambert. "Multichannel blind deconvolution: FIR matrix algebra and separation of multipath mixtures". In: *Russell The Journal Of The Bertrand Russell Archives* May (1996).

[63] P. P. Vaidyanathan. "Theory of optimal orthonormal subband coders". In: *IEEE Transactions on Signal Processing* 46.6 (1998), pp. 1528–1543. ISSN: 1053587X. DOI: 10.1109/78.678466.

[64] Xiqi Gao, Truong Q. Nguyen, and Gilbert Strang. "On factorization of M-channel paraunitary filterbanks". In: *IEEE Transactions on Signal Processing* 49.7 (2001), pp. 1433–1446. ISSN: 1053587X. DOI: 10.1109/78.928696.

[65] Tuan Do-Hong and Peter Russer. "An analysis of wideband direction-of-arrival estimation for closely-spaced sources in the presence of array model errors". In: *IEEE Microwave and Wireless Components Letters* 13.8 (2003), pp. 314–316. ISSN: 15311309. DOI: 10.1109/LMWC.2003.815701.

[66] P. D. Baxter and J. G. McWhirter. "A Novel Technique for Broadband SVD". In: *Proceedings 12th Annual Workshop of Adaptive Sensor Array Signal Processing* (2004).

[67] Andre Tkacenko. "Approximate Eigenvalue Decomposition of Para-hermitian Systems Through Successive FIR Paraunitary Transformations". In: *Simulation* (2010), pp. 4074–4077.

[68] Stephan Weiss, Andrew P. Millar, and Robert W. Stewart. "Inversion of parahermitian matrices". In: *European Signal Processing Conference* (2010), pp. 447–451. ISSN: 22195491.

[69] Rasmus Brandt. "Polynomial Matrix Decompositions". In: *Convergence* November (2010).

[70] M Alrmah, Stephan Weiss, and John G. McWhirter. "Implementation of accurate broadband steering vectors for broadband angle of arrival estimation". In: *IET Intelligent Signal Processing Conference 2013 (ISP 2013)* (2013), pp. 1–6. DOI: 10.1049/cp.2013.2057.

[71] Mohamed A. Alrmah et al. "Angle of arrival estimation for broadband signals: A comparison". In: *Intelligent Signal Processing Conference 2013 (ISP 2013), IET* (2013), pp. 1–6. DOI: 10.1049/cp.2013.2066.

[72] Mahdi Tohidian, Hamidreza Amindavar, and Ali M. Reza. "A DFT-based approximate eigenvalue and singular value decomposition of polynomial matrices". In: *Eurasip Journal on Advances in Signal Processing* 2013.1 (2013), pp. 1–16. ISSN: 16876172. DOI: 10.1186/1687-6180-2013-93.

[73] Soydan Redif, Stephan Weiss, and John G. McWhirter. "Sequential matrix diagonalization algorithms for polynomial EVD of parahermitian matrices". In: *IEEE Transactions on Signal Processing* 63.1 (2015), pp. 81–89. ISSN: 1053587X. DOI: 10.1109/TSP.2014.2367460.

[74] Jamie Corr et al. "Multiple shift maximum element sequential matrix diagonalisation for parahermitian matrices". In: *IEEE Workshop on Statistical Signal Processing Proceedings* (2014), pp. 312–315. DOI: 10.1109/SSP.2014.6884638.

[75] Mohamed A. Alrmah et al. "Polynomial subspace decomposition for broadband angle of arrival estimation". In: *2014 Sensor Signal Processing for Defence, SSPD 2014* (2014), pp. 1–5. DOI: 10.1109/SSPD.2014.6943305.

[76] Jamie Corr et al. "Cyclic-by-row approximation of iterative polynomial EVD algorithms". In: *2014 Sensor Signal Processing for Defence, SSPD 2014* (2014). DOI: 10.1109/SSPD.2014.6943330.

[77] Zeliang Wang et al. "Multiple shift second order sequential best rotation algorithm for polynomial matrix EVD". In: *2015 23rd European Signal Processing Conference, EUSIPCO 2015* (2015), pp. 844–848. DOI: 10.1109/EUSIPCO.2015.7362502.

[78] Jamie Corr et al. "Row-Shift Corrected Truncation of Paraunitary Matrices for PEVD Algorithms". In: (2015), pp. 854–858.

[79] J.G. McWhirter et al. "Reduced search space multiple shift maximum element sequential matrix diagonalisation algorithm". In: (2016). DOI: 10.1049/cp.2015.1769.

[80] Zeliang Wang et al. "Order-controlled multiple shift SBR2 algorithm for para-Hermitian polynomial matrices". In: *Proceedings of the IEEE Sensor Array and Multichannel Signal Processing Workshop* 2016-Septe.July (2016), pp. 1–6. ISSN: 2151870X. DOI: 10.1109/SAM.2016.7569742.

[81] Fraser K. Coutts et al. "Memory and complexity reduction in parahermitian matrix manipulations of PEVD algorithms". In: *European Signal Processing Conference* 2016-Novem (2016), pp. 1633–1637. ISSN: 22195491. DOI: 10.1109/EUSIPCO.2016.7760525.

[82] Fraser K. Coutts et al. "Complexity and search space reduction in cyclic-by-row PEVD algorithms". In: *Conference Record - Asilomar Conference on Signals, Systems and Computers* 0 (2017), pp. 1349–1353. ISSN: 10586393. DOI: 10.1109/ACSSC.2016.7869595.

[83] Jamie Corr. "Advanced algorithms for polynomial matrix eigenvalue decomposition". PhD thesis. University of Strathclyde, 2017.

[84] Soydan Redif, Stephan Weiss, and John G. McWhirter. "Relevance of polynomial matrix decompositions to broadband blind signal separation". In: *Signal Processing* 134.May 2016 (2017), pp. 76–86. ISSN: 01651684. DOI: 10.1016/j.sigpro.2016.11.019.

[85] Fraser Coutts et al. "Divide-and-Conquer Sequential Matrix Diagonalisation for Parahermitian Matrices". In: *2017 Sensor Signal Processing for Defence Conference* 5 (2017), pp. 1–5. DOI: 10.1109/SSPD.2017.8233228.

[86] Fraser K. Coutts et al. "Analysing the performance of divide-and-conquer sequential matrix diagonalisation for large broadband sensor arrays". In: *IEEE Workshop on Signal Processing Systems, SiPS: Design and Implementation* 3 (2017). ISSN: 15206130. DOI: 10.1109/SiPS.2017.8109976.

[87] Fraser Coutts et al. "Impact of Fast-Converging PEVD Algorithms on Broadband AoA Estimation". In: *2017 Sensor Signal Processing for Defence Conference, SSPD 2017* 5 (2017), pp. 1–5. DOI: 10.1109/SSPD.2017.8233238.

[88] Fraser K Coutts et al. "A Comparison of Iterative and DFT-Based Polynomial Matrix Eigenvalue Decompositions". In: *IEEE 7th International Workshop on Computational Advances in Multi-Sensor Adaptive Processing* (2017).

[89] Fraser K Coutts et al. "Enforcing eigenvector smoothness for a compact DFT-based polynomial eigenvalue decomposition". In: *Proceedings of the IEEE Sensor Array and Multichannel Signal Processing Workshop*. Vol. 2018-July. 5. IEEE, 2018, pp. 159–163. ISBN: 9781538647523. DOI: 10.1109/SAM.2018.8448895.

[90] Stephan Weiss, Jennifer Pestana, and Ian K. Proudler. "On the Existence and Uniqueness of the Eigenvalue Decomposition of a Parahermitian Matrix". In: *IEEE Transactions on Signal Processing* 66.10 (2018), pp. 2659–2672. ISSN: 00029939. DOI: 10.2307/2036109.

[91] Julius O. Smith. *Introduction to Digital Filters with Audio Applications*. W3K Publishing, 2007. ISBN: 978-0-9745607-1-7. URL: https://ccrma.stanford.edu/~jos/filters06/Convolution_Theorem.html.

[92] Julius O. Smith. *Mathematics of the Discrete Fourier Transform (DFT)*. W3K Publishing, 2007. ISBN: 978-0-9745607-4-8. URL: https://ccrma.stanford.edu/~jos/mdft/Convolution_Theorem.html.

[93] Rs Marinescu and A Buzo. "Applying the Accumulation of Cross-Power Spectrum Technique for Traditional Generalized Cross-Correlation Time Delay Estimation". In: *International Journal on Advances in Telecommunications* 6.3 (2013), pp. 98–108.

[94] Daniel V. Rabinkin et al. "A DSP implementation of source location using microphone arrays." In: *The Journal of the Acoustical Society of America* 99.4 (2005), pp. 2503–2529. ISSN: 0001-4966. DOI: 10.1121/1.415678.

[95] A.O. Hero and S. Schwartz. "A New Generalized Cross Correlator". In: *IEEE Transactions on Acoustics Speech and Signal Processing* 1 (1985).

[96] J. Hassab and R. Boucher. "Optimum Estimation of Time Delay". In: *IEEE Transactions on Acoustics Speech and Signal Processing* 4 (1979), pp. 373–380.

[97] E. J. Hannan and P. J. Thomson. "The estimation of coherence and group delay". In: *Biometrika* 58.3 (1971), pp. 469–481. ISSN: 00063444. DOI: 10.1093/biomet/58.3.469.

[98] Alan V. Oppenheim and Ronald W. Schafer. *Discrete-time signal processing*. Prentice Hall signal processing series. Upper Saddle River [N.J.] : Pearson, c2010, 2010. ISBN: 9780131988422. URL: http://ezproxy.massey.ac.nz/login?url=http://search.ebscohost.com/login.aspx?direct=true&db=cat00245a&AN=massey.b2201767&site=eds-live&scope=site.

[99] Joseph A. Gallian. *Contemporary Abstract Algebra*. 8th ed. Cengage Learning, 2012, p. 583. ISBN: 9781133599708.

[100] Raymond Edward Alan Christopher Paley and Norbert Wiener. *Fourier transforms in the complex domain*. American Mathematical Soc., 1934.

[101] Athanasios Papoulis and S. Unnikrishna Pillai. *Probability, Random Variables and Stochastic Processes*. 4th. New York: McGraw-Hill Higher Education, 2002, p. 852. ISBN: 0073660116.

[102] A. J. Hoffman and H. W. Wielandt. "The Variation of the Spectrum of a Normal Matrix". In: *Duke Mathematical Journal* 20.1 (1953), pp. 37–40. URL: https://doi-org.ezproxy.massey.ac.nz/10.1142/9789812796936_0011.

[103] J. H. Wilkinson. *Elementary Proof of the Wielandt-Hoffman Theorem and of its Generalization*. Tech. rep. Stanford University, 1970.

[104] Stephan Weiss, Andrew P. Millar, and Robert W. Stewart. "Inversion of Parahermitian Matrices". In: *18th European Signal Processing Conference*. Aalborg, Denmark, 2010. URL: http://strathprints.strath.ac.uk/20620/.

[105] Jamie Corr. "Advanced Algorithms for Polynomial Matrix Eigenvalue Decomposition PhD Thesis". PhD thesis. 2017.

[106] Mati Wax, Tie-jun Shan, and Thomas Kailath. "Spatio-Temporal Spectral Analysis". In: *IEEE Transactions on Acoustics Speech and Signal Processing* 32.4 (1984).

[107] Piya Pal and P. P. Vaidyanathan. "A novel autofocusing approach for estimating directions-of-arrival of wideband signals". In: *Conference Record - Asilomar Conference on Signals, Systems and Computers* (2009), pp. 1663–1667. ISSN: 10586393. DOI: 10.1109/ACSSC.2009.5469796.

[108] Hugh A. Robertson et al. "Conservation status of New Zealand birds, 2016". In: *New Zealand Threat Classification Series* 19 (2016), 26 p. ISSN: 0029-4470. URL: http://www.doc.govt.nz/upload/documents/science-and-technical/nztcs4entire.pdf.