2017-01-18

Enabling 5G Technologies

Travis Fredrick Collins
Worcester Polytechnic Institute

Enabling 5G Technologies

Travis Fredrick Collins

A Dissertation
Submitted to the Faculty
of the
WORCESTER POLYTECHNIC INSTITUTE
in partial fulfillment of the requirements for the
Degree of Doctor of Philosophy
in
Electrical and Computer Engineering

December 2016

APPROVED:

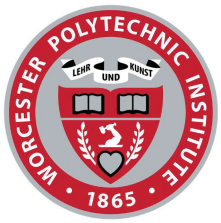
Professor Alexander M. Wyglinski
Primary Advisor
Worcester Polytechnic Institute

Professor Xinming Huang
Committee Member
Worcester Polytechnic Institute

Professor Kaushik Chowdhury
Committee Member
Northeastern University

Enabling 5G Technologies

Travis Fredrick Collins



A Dissertation
Submitted to the Faculty
of the
WORCESTER POLYTECHNIC INSTITUTE
in partial fulfillment of the requirements for the
Degree of Doctor of Philosophy
in
Electrical and Computer Engineering

December 2016

APPROVED:

Professor Alexander M. Wyglinski
Primary Advisor
Worcester Polytechnic Institute

Professor Xinming Huang
Committee Member
Worcester Polytechnic Institute

Professor Kaushik Chowdhury
Committee Member
Northeastern University

Abstract

The increasing demand for connectivity and broadband wireless access is leading to the fifth generation (5G) of cellular networks. The overall scope of 5G is greater in client width and diversity than in previous generations, requiring substantial changes to network topologies and air interfaces. This divergence from existing network designs is prompting a massive growth in research, with the U.S. government alone investing $400 million in advanced wireless technologies. 5G is projected to enable the connectivity of 20 billion devices by 2020, and dominate such areas as vehicular networking and the Internet of Things. However, many challenges exist to enable large scale deployment and general adoption of the cellular industries.

In this dissertation, we propose three new additions to the literature to further the progression 5G development. These additions approach 5G from top down and bottom up perspectives considering interference modeling and physical layer prototyping. Heterogeneous deployments are considered from a purely analytical perspective, modeling co-channel interference between and among both macrocell and femtocell tiers. We further enhance these models with parameterized directional antennas and integrate them into a novel mixed point process study of the network. At the air interface, we examine Software-Defined Radio (SDR) development of physical link level simulations. First, we introduce a new algorithm acceleration framework for MATLAB, enabling real-time and concurrent applications. Extensible beyond SDR alone, this dataflow framework can provide application speedup for stream-based or data dependent processing. Furthermore, using SDRs we develop a localization testbed for dense deployments of 5G smallcells. Providing real-time tracking of targets using foundational direction of arrival estimation techniques, including a new OFDM based correlation implementation.

Acknowledgements

This research was supported through generous contributions of the MathWork Inc., Ettus Research, and the National Science Foundation. I would specifically like to thank my manager at MathWorks Mike McLernon for his guidance and the freedom he gave me to challenge myself. I would also like to thank my committee members, Professor Huang and Professor Chowdhury. Their validation of my own work has solidified my confidence as a researcher.

Personally, I want to thank my advisor Professor Alexander Wyglinski, with whom I have worked with for the last six years of my academic life. He has provided me with the resources to follow my passions and the support to see them through. He has helped mold me into the professional engineer that I am today. I will be forever grateful to him.

Over the past five years, many have passed through the Wireless Innovation Lab, but it has always remained as a safe place to share ideas, tells jokes, and vent frustrations. Support from the lab was something I could always relying upon. Bengi, Le, Paulo, Renato, Sean, Zoe, and last but not least Srikanth, I am very proud to call all of you my friends.

As always, my parents and my brother have supported me during the course of this degree and throughout my life. I am deeply indebted to them for their unconditional love and encouragement.

Finally, during my graduate studies, I was fortunate enough to meet my best friend and wife Lauren. She taught me that I could accomplish anything, and this work is proof of that. This is a stepping stone for our life together and was only possible because of her.

Contents

List of Figures v

List of Tables x

List of Abbreviations xi

1 Introduction 1
- 1.1 Motivation . . . 1
- 1.2 State-of-the-Art . . . 6
- 1.3 Research Questions To Be Answered . . . 10
- 1.4 Research Contributions . . . 11
- 1.5 Dissertation Outline . . . 11
- 1.6 List of Related Publications . . . 12

2 Background Fundamentals 14
- 2.1 Software-Defined Radio Prototyping Technologies . . . 14
- 2.2 Stochastic Network Models . . . 18
- 2.3 Real-Time Localization . . . 21

3 Data-Flow Algorithm Acceleration 26
- 3.1 Architecture Overview . . . 26
- 3.2 Parallel Architecture Systems . . . 27
  - 3.2.1 Limitations . . . 32
- 3.3 Implementation . . . 33
  - 3.3.1 MATLAB Integration and Related Work . . . 34
  - 3.3.2 Code Architecture . . . 37
  - 3.3.3 MATLAB Code Generation . . . 39
- 3.4 Testing and Evaluation . . . 40
  - 3.4.1 Framework Testing . . . 40
  - 3.4.2 Flowgraph Testing . . . 41
- 3.5 Designing For DataFlow . . . 44
  - 3.5.1 Limitations . . . 45
  - 3.5.2 Reaching Real-Time . . . 46
- 3.6 Discussion and Retrospective . . . 48

3.7 Chapter Summary . . . 48

4 Stochastic Networks: Modeling Interference **50**
4.1 Point Process Model . . . 51
4.2 Network Model . . . 52
4.2.1 Closed Subscriber Connectivity . . . 56
4.2.2 Gain Patterns . . . 56
4.3 Success Probability . . . 58
4.4 Mixed PPP/GPP Tiered Networks . . . 62
4.5 Discussion and Retrospective . . . 65
4.6 Chapter Summary . . . 66

5 Real-Time Localization Prototype For Dense Femtocell Networks **68**
5.1 Collaboration Segmentation . . . 69
5.2 Representative Network Level Model . . . 70
5.3 Direction of Arrival Estimation . . . 71
5.3.1 Array Modeling . . . 72
5.4 MUSIC Direction Finding . . . 75
5.4.1 Root-MUSIC . . . 78
5.4.2 Cramér-Rao Lower Bound . . . 79
5.4.3 GNURadio Software Only Implementation . . . 81
5.5 Effective Aperture Distribution Function . . . 82
5.5.1 EADF-DoA Estimation . . . 84
5.5.2 OFDM Receiver, Channel Estimation, and GNURadio . . . 86
5.6 USRP Synchronization . . . 89
5.6.1 TwinRX Daughterboards and Software Libraries . . . 93
5.7 Experimental Setup . . . 95
5.7.1 Initial MUSIC Real-World Testing . . . 96
5.7.2 Antenna Calibration and Refining Spatial Measurements . . . 101
5.7.3 MUSIC Calibration and Testing . . . 103
5.7.4 EADF-FFT DoA Implementation . . . 105
5.8 Discussion and Retrospective . . . 111
5.9 Chapter Summary . . . 111

6 Conclusion and Future Work **113**
6.1 Research Achievements . . . 113
6.2 Future Works . . . 114

A Appendix: Proof of Lemma 1 **116**

B Appendix: Proof of Lemma 2 **117**

C Appendix: Proof of Theorem 1 **118**

Bibliography **124**

List of Figures

1.1 Speed comparison across generations of deployed digital cellular networks. Speed is related to common tasks to provide perspective on user experience [6]. 2
1.2 Frequency allocation of the United States for radio frequency bands [12]. . . 4
1.3 MATLAB SDR platform connectivity for real-world signal analysis [21]. . . 5
1.4 Example intercell deployment of MBS with many FBS. FBS utilize Internet backbones to communicate with the LTE Evolved Packet Core (EPC). . . . 7
1.5 eICIC scheduling example using ABS between a single MBS and a single FBS [28]. 8

2.1 Comparision of matrix multiplication of two square matrices ($A = B \times C$), with loops (red) vs. BLAS (blue) vector calls (dgemm). 16
2.2 Comparision of dataflow architectures in terms of complexity to use vs. their capability as a platform to implement general mathematical processing. . . 17
2.3 Hexagonal cell layout of an ideal network, maximizing spatial coverage. . . 20
2.4 Realistic cell layout, with cell exhibiting some visible structure. 20
2.5 Beam-steering in localization-aware network providing pro-active resource management of spectrum in order to reduce interference. Resulting in less congested links. [73] 22
2.6 Breakdown of common DoA methods including classical, subspace, and Algebraic [77, 78]. 24

3.1 Data parallelism realization for equation (3.1) on a four core system. In the case of a streaming application the buffer block will be utilized to delay input data until four samples are available. 29
3.2 Cascaded task parallelism realization for equation (3.1) on a four core system. If we condition on each core having a different function, this realization is the most efficient. 29
3.3 Processing analysis for a streaming type application comparing the performance of three implementation: data parallel (blue), task parallel (green), and single threaded where no parallelism is exploited (black). Data arrives at the system every one time unit and takes two unit to be processed. . . . 31

3.4 Processing analysis for a off-line type application, where all data is available initially. Three implementations are compared: data parallel (blue), task parallel (green), and single threaded where no parallelism is exploited (black). Data parallelism can utilize all processing units (four) initially. 31
3.5 Mixed parallel flow for Equation (3.4), exploiting data parallel and task parallel advantages. 33
3.6 MATLAB flowgraph code for implementation of a simple noise filtering exampling. The MATLAB class *GraphGenerator* only contains five user visible methods, all of which are shown here. 36
3.7 Flowgraph visualization shown when invoking the *DrawGraph* method as shown in Figure 3.6. The end result is provided in a MATLAB figure. . . . 37
3.8 Transfer latency between blocks for 10^6 iterations for locking queue design. Double precision additions are provided as references to the cost of an individual transfer on average. These reference are provide as vertical lines since on a modern CPU the are roughly deterministic operations. 42
3.9 Transfer latency between blocks for 10^6 iterations for non-locking queue design. Double precision additions are provided as references to the cost of an individual transfer on average. These reference are provide as vertical lines since on a modern CPU the are roughly deterministic operations. 42
3.10 Throughput testing of cascaded FFT operations in the single threaded mode (bottom) and the MLDF implementation (top). These tests were run for two cascaded FFTs to twenty-two FFTs. The random vector generation (Vec Gen) was put into an independent thread to remove the additional computation required for those operations. 43
3.11 This test compares an increase in cascaded FFT operations in the MLDF case and the single threaded standard case. At each number of cascaded FFTs the system processed 10^5 random complex double precision vectors of length 2^{15}. Each test was run four times, with little variance between them. 44
3.12 Complex flowgraph for WLAN 802.11a receiver, based upon the current design available in MATLAB. Significant resources where put into the packet detection phase of the flowgraph, which consists of six threads (Abs, Moving AverageSP, Moving AveragePP, Normalize, Peak Detect, Delay). This design was able to run in real-time and decode beacon frames from commerical routers. DecodePacket contains the majority of the code, but is called drastically less frequent than the upstream blocks. 45
3.13 Relative throughput of computation blocks in WLAN receiver design. In green are blocks requiring continuous operations, in blue are blocks that have dependent operation, and in red is the original MATLAB demo implementation. 47

4.1 Realization comparison of DPP vs. PPP on a 2-dimensional plane. The repulsiveness of the DPP can easily be observed over the complete randomness of the PPP. 54
4.2 Example antenna gain patterns over the orientation angle θ. This pattern must always maintain the TRP and therefore selections of g_2 and *theta* lead to g_1 directly. 58

4.3 Example antenna gain patterns over the orientation angle θ. At θ_{3dB} the 3GPP pattern is at $3dB$ of g_1 and complete gain transition to g_2 at θ_1. Note that the figure is not to scale since there is no combination of g_1 and g_2 that simultaneously achieves unit TRP for both patterns. Note that g_1 and g_2 are not to scale since there are no configuration of values, while maintaining the overall gain constraint, for g_1 and g_2 to be identical between both patterns. 59
4.4 Success probability for different mainlobe beam-widths for antennas of the MBS connected tier. There is only one secondary PPP tier with omni-directional antennas. Here simulation provides matching results against their numerically evaluated analytical equations. 62
4.5 Effect of transmit power and tier density heterogeneity on coverage. Relative power is increased linearly in favor of the MBS tier. Simultaneously, network density is increase linearly in favor of the FBS tier. 64
4.6 Mean rate across different beam-widths of a connected MBS tier, with a single interference FBS tier at increasing density. This is in contrast to the flat omni-directional case of for the MBS without the secondary interference FBS tier. 65

5.1 Collaboration replacement outline with Finnish partner. 69
5.2 Example layout of a dense FBS network in an city environment. MN are located or travel along or near to the green lines in outdoor positions [115]. 70
5.3 Beam-pattern of four element ULA with element spacing of $\frac{\lambda}{4}$. 73
5.4 Beam-pattern of four element ULA with element spacing of $\frac{\lambda}{2}$. 73
5.5 Beam-pattern of four element ULA with element spacing of λ. 73
5.6 Beam-pattern of four element ULA with element spacing of 2λ. 74
5.7 Theoretical response of four element ULA at 2.45GHz with a spacing of $\frac{\lambda}{2}$. 75
5.8 Antenna gain patterns of the VERT2450 antennas provided by Ettus Research [118]. 76
5.9 MUSIC pseudo-spectrum of two transmitted signals at 42 and 120 degrees with an increasing number of elements in a ULA with a spacing of $\lambda/2$. . . 78
5.10 MUSIC pseudo-spectrum of a transmitted signal at 42 and 120 degrees with increasing SNR in a ULA with a spacing of $\lambda/2$. 78
5.11 CRLB of MUSIC across azimuth angles of a ULA for two different SNR conditions. 80
5.12 CRLB of MUSIC over different snapshot lengths averaged across azimuth angles of a ULA. 81
5.13 CRLB of MUSIC over different SNRs averaged across azimuth angles of a ULA. 81
5.14 GNURadio simulation only flowgraph of MUSIC algorithm with received manifold source from a file generated in MATLAB. 82
5.15 EADF of monopole antenna for vertical polarization only before compression is applied through row and column removal. 84
5.16 CRLB of MUSIC over different snapshot lengths averaged across azimuth angles of a ULA. 86

5.17 CRLB of MUSIC over different SNRs averaged across azimuth angles of a ULA. . . . 86
5.18 OFDM packet structure for GNURadio implementation. Preamble is not modulated, unlike the header and payload which are both BPSK/QAM modulated and then OFDM modulated. . . . 87
5.19 OFDM packet structure for GNURadio implementation. Preamble is not modulated, unlike the header and payload which are both BPSK/QAM modulated and then OFDM modulated. . . . 88
5.20 USRP-N210 from Ettus Research front panel. Consists of frequency source (REF CLOCK), clocking source (PPS IN), antenna ports (RF1,RF2), MIMO cabling input, and Gigabit ethernet. . . . 90
5.21 Eight USRP array synchronized with Octoclock and equal length cabling. The Octoclock is driven by external sources to provide REF and PPS signals. 90
5.22 Pre calibration (Bottom) and post calibration (Top) of complex tone feed through cabling to four USRP-N210's. . . . 92
5.23 Multi-USRP configuration for online correction, or in configurations without repeatable phase between initializations. . . . 93
5.24 Performance comparison between Armadillo C++ libraries and raw BLAS calls through the Fortran interface in C++. . . . 94
5.25 ULA fixture with four receiving USRP-N210's, one USRP-N210 used for synchronization tone generation, and an Octoclock used for clocking synchronization. The Vert2450 antennas are space is $\sim 2.4in$ or $\frac{\lambda}{2}$ for 2.45 GHz carrier transmissions. . . . 96
5.26 Relative positioning of transmitter and phased array for DoA over-the-air experimentation. These are positions within the WPI Odeum in the center of the room offset from the wall by 10 ft. . . . 97
5.27 MUSIC RMSE across azimuth for initial large room test at 13 ft. . . . 98
5.28 MUSIC RMSE across azimuth for initial large room test at 29 ft. . . . 98
5.29 Relative locations of target transmitter test positions verses the four antenna array. The chamber itself is only 3×3 meters making space relatively cramped compared with the larger room. . . . 99
5.30 MUSIC and Root-MUSIC RMSE error across azimuth in anechoic chamber. 100
5.31 Pseudo-spectrum estimate of MUSIC algorithm for target placed at 73 degrees azimuth. . . . 100
5.32 Pegboard layout of transmitter in reference to four element array. Pegboard is used to maintain accurate target positioning across azimuth. . . . 102
5.33 RMSE as a result of increasing element positioning noise. . . . 103
5.34 Transmitter positions in anechoic chamber relative to the receiving four element array. . . . 104
5.35 Comparison of calibrated and uncalibrated DoA estimates across azimuth. . 105
5.36 Pseudo-spectrum of MUSIC with calibrated and uncalibrated antennas. . . 106
5.37 Relative locations of target transmitter test positions verses the four antenna array. The conference room itself is only 3×5 meters. . . . 107

5.38 Estimation performance comparison between the baseline MUSIC implementation and the packet based EFD design. Calibration of the antennas were introduced in the second set of measurements with improved in both algorithms. All measurements were taken in the WiLab conference room. 109
5.39 EFD and MUSIC measurements over time of a single position. Measurements show little to no variance for both EFD and MUSIC. These estimations are made for a target positioned at 116 degrees and have included antenna compensation in their software receive chains. 109
5.40 $2D$ localization performance of two arrays at varying distances and DoA estimation error. 110

List of Tables

4.1 Table of variables for stochastic models of randomly deployed BS. 53

List of Abbreviations

1G	First generation
2G	Second generation
3G	Third generation
4G	Fourth generation
5G	Fifth generation
AWGN	Additive White Gaussian Noise
BPSK	Binary Phase Shift Keying
BS	Base Station
CDF	Cumulative Distribution Function
CG	Code Generation
CPP	Cluster Point Process
CPS	Cyber-Physical-Systems
CSN	Closed Subscriber Network
DoA	Direction of Arrival
DP	Data Parallel
DPP	Determinantal Point Process
DSL	Domain Specific Language
EADF	Effective Aperture Distribution Function
EFD	EADF-FFT DoA
EPC	Evolved Packet Core
eICIC	Evolved Inter-cell Interference Coordination
FBS	Femtocell Base Station

FFT	Fast Fouier Transform
FS	Fast Source
GPP	Ginibre Point Process
GPS	Global Positioning Systems
HCPP	Hard-Core Point Process
HetNet	Heterogeneous Network
HIP	Heterogeneous Independent Process
ICIC	Inter-cell Interference Coordination
IoT	Internet of Things
LoS	Line of Sight
LTE	Long-Term Evolution
M2M	Machine-to-Machine
mbps	Megabits per second
MBS	Macrocell Base Station
MIMO	Multi-Input-Multi-Output
MLDF	MATLAB DataFlow
MLGF	MATLAB Generate Function
MN	Mobile Node
MT	Multi-threaded
MUSIC	Multiple Signal Classification
OFDM	Orthogonal Frequency Division Multiplexing
OSN	Open Subscriber Network
PDF	Probability Distribution Function
PP	Point Process
PPP	Poisson Point Process
RMSE	Root Mean Squared Error
RSSI	Received Signal Strength Indicator
SDR	Software-Defined Radio
SG	Stochastic Geometry

SIMO	Single Input Multiple Outputs
SINR	Signal To Interference Plus Noise Ratio
SS	Slow Source
ST	Single Threaded
TDoA	Time Difference of Arrival
ToA	Time of Arrival
TP	Task Parallel
TRP	Total Radiated Power
ULA	Uniform Linear Array
V2V	Vehicle-to-Vehicle
V2X	Vehicle-to-everthing

Chapter 1

Introduction

1.1 Motivation

Wireless communication and networks have become a dominant force in commercial business, military operations, and everyday life of typical consumers. Such networks are an essential component of modern societal infrastructure, and are continually expanding to meet demand. Networks themselves have incrementally advanced over time starting with first generation analog systems of the 1970's [1] and reaching the high-speed digital networks we know today in their third and fourth generations. However, the community at large is a the beginning a new frontier in research where expectations of scale and capacity will be without precedent, all to meet the massive projected hunger for connectivity. This next evolution in cellular technology will be the fifth generation (5G) and is expected to be deployed by 2020.

Unlike previous network generations, there will be a much wider and diverse user equipment (UE) base to support with varying sets of new requirements in 5G. This growth is driven by the higher connectivity of devices, convergence or transition of traditional Wi-Fi targeted markets with cellular technology, and alternative deployment use cases. To meet this demand 5G networks promise link speeds to reach tens of gigabits [2] and larger degrees of connectivity. As shown in Figure 1.1, a hundred fold increase from existing LTE networks and greater than four orders of magnitude increase from 3G networks.

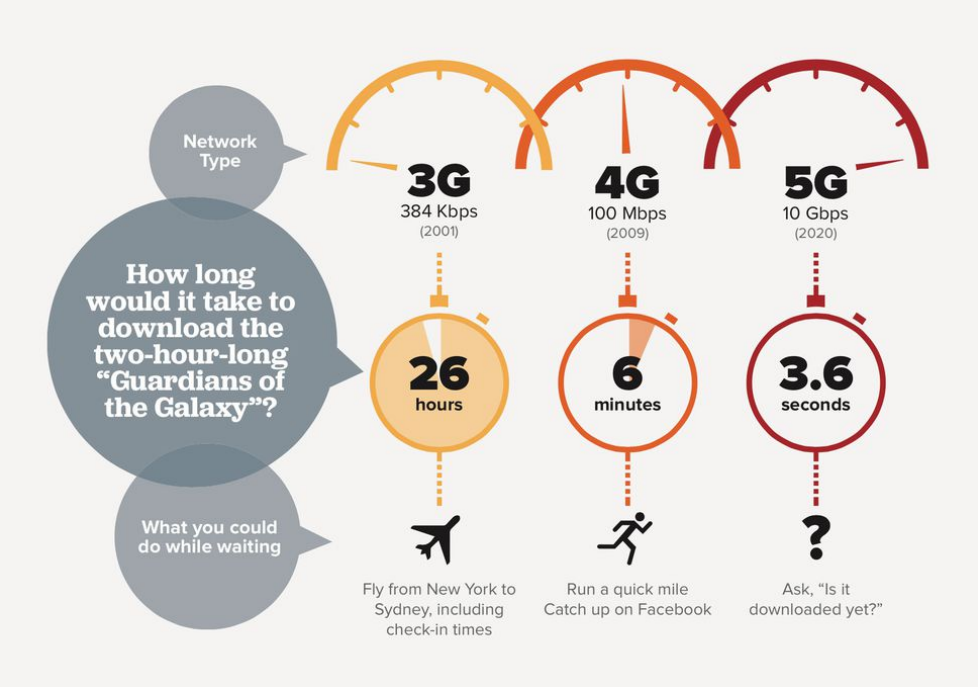


Figure 1.1: Speed comparison across generations of deployed digital cellular networks. Speed is related to common tasks to provide perspective on user experience [6].

Comparatively to next generation networks, 3G and 4G have experienced limited exposure to the Internet since cellular modems are rarely found outside of phones or tables. However, this dynamic is estimated to change in 5G, with an expectation that smart-phone traffic will exceed PC traffic by 2020 [3]. Unlike past networks with a major of growth coming from human-centric communications machine-type or machine-to-machine (M2M) applications will utilize an important portion of the available bandwidth, especially as industries become more vertically integrated and reliant on automation [4]. Overall, the combined user and machine population is projected to reach 25 Billion connected devices by 2020 [5].

This evolution in machine-type communications to this scale is not only driving 5G development, but Wi-Fi and core networking architecture as well. This movement is more generally known as the "Internet of Things" (IoT), as coined by Peter Lewis in 1985 [7], and is a main driver behind 5G network design. IoT itself brings to light this concept that both human devices and machines can be uniquely addressable through the Internet and in

essence everything will be connected. Therefore, mobile and wireless communications will be key enablers for the IoT. 5G in particular will enable IoT for new use cases and commercial sectors where so far mobile communication has been nonexistent. Providing many applications with the required minimal latency and throughput requirements, which are currently impractical at scale. Key examples are Cyber-Physical-Systems (CPS) where sensors, people, and environments are integrated through communication network [4]. Hence, IoT applications expand requirements of future deployments by demanding substantial increases in supported connectivity.

Beyond simplistic sensors, 5G is already being targeted for vehicular networks to provide both vehicle-to-vehicle (V2V) communications and vehicle-to-everything (V2X) as well [8]. V2V and V2X applications can leverage M2M protocols in 5G, as well as take advantage of quality of service (QoS) designs for emergency communications. Unlike the Wi-Fi based standard for vehicular networking IEEE 802.11p, which is more random access based [9], 5G has a significant advantage due to advanced deterministic scheduling techniques along with link directionality. Both these features becoming invaluable as network densities increase.

Besides mobile based clients, 5G is also being considered as a method for providing general broadband access to communities. Taking over the proverbial "last mile", cellular deployments can be an obvious alternative to laying fiber close to communities for connectivity. This is already a common paradigm in developing countries by companies like Vanu [10] with GSM deployments. However, with 5G fiber-like speeds could be realized with a similar deployment and similar remote areas to be easily connected. Besides the general deployments, remote residencies which are serviced by hyperdirectional microwave links could also be replaced by 5G as well. Overall, 5G "last mile" deployments would reduce the cost and construction time required for increased speeds in residential areas.

In general, 5G hopes to touch many new areas that were traditional strongholds for Wi-Fi and even wired networks. However, with this massive projected level of connectivity, especially for alternatives to fiber deployments, 5G is not without many fundamental challenges. The center at many of these problems is the lack of spectrum or electromagnetic real estate available. Due to the proliferation of wireless technology there has become pronounced scarcity in the the UHF through SHF bands as seen by the frequency allocations in

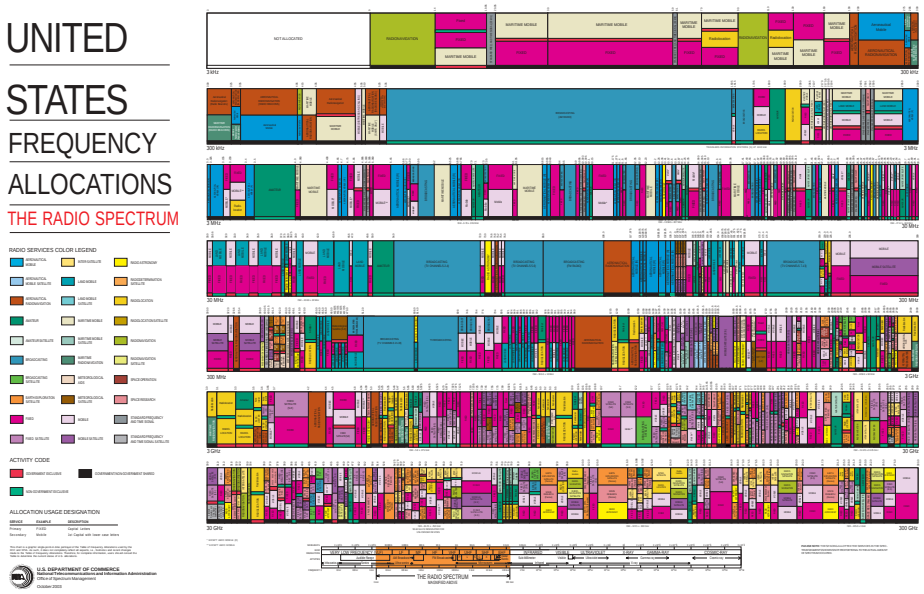


Figure 1.2: Frequency allocation of the United States for radio frequency bands [12].

Figure 1.2. Discussions into spectrum scarcity have been a focus of academic research, government regulation, and commercial business for the past decade [11]. As a result, much of the research for 5G has begun to focus on new spectral efficiency strategies, hybrid network topologies, and techniques requiring advanced radio designs and logic. Alternatively, there is a departure from the traditional frequency bands to *mmWave* ranges, where spectrum is abundant. However, there are still many physical limitations that need to be overcome to operate large networks at these frequencies.

Hybrid or heterogeneous topologies are not a new discussion in cellular networks, existing in both 3G and 4G networks in the form of pico, micro, and femto-cells. In 2013 98% of mobile operators believed that smallcells would be essential for future networks [13]. However, the massive roll out of these smallcells has not been realized in either generation [14]. Nonetheless, dense deployment of these smallcells is a primary strategy in 5G networks to allow for spatial reuse and macro-cell offloading through residential backbones. *mmWave* equipment arrival may be the driving force behind these smallcells due to the limited range and propagation limitations [15]. However, these propagation issues are still an area of

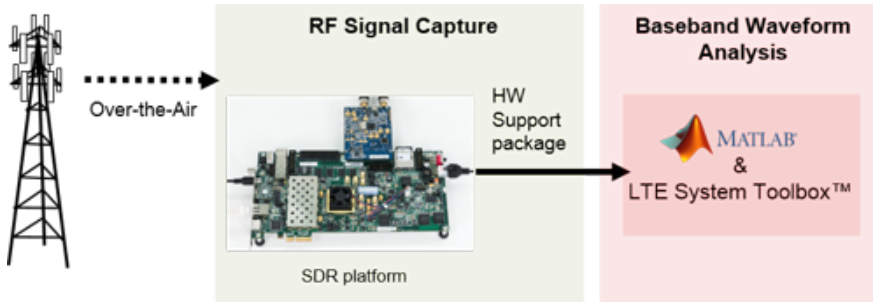


Figure 1.3: MATLAB SDR platform connectivity for real-world signal analysis [21].

concern and require additional research by the community at large.

Uniquely positioned to accelerate this research at the air interface level, are Software-Defined Radios (SDR) [16]. SDRs allow for faster development and testing of waveforms and modems than traditional ASIC only implementations. Since SDRs actually allow for physical simulations we can go beyond synthetic models and operate in these new unknown realms. 5G is especially unique since the community has limited experience in *mmWave* bands, and multi-node networks are especially challenging to model correctly in the electromagnetic space. Therefore, synthetic simulations of environments for 5G networks will not always be enough and practical implementations will be required to prove network feasibility. Nonetheless, even with SDRs prototyping hardware and software have matured slowly in these areas and still present challenges [17]. Such software is particularly absent from widely available or cost effective tools [18], and implementations rarely go beyond single point-to-point (P2P) links. For example in Figure 1.3 MATLAB provides LTE standards compliant reception, but is limited to non-real-time recovery and does not address medium access control. Even so, this bar is being lowered through both closed and open-source packages [19] with heterogeneous computing and natural development. However, SDRs have become a staple in both academia and industry, but software frameworks for these devices are not simply turnkey and require deep knowledge to provide efficient implementations. In general SDRs have become more capable since their introduction in the 1990's [20], but available software tools are limiting.

In this section we have provided motivation and direction of 5G networks, including

their relation to prototyping through use of SDR hardware. In the upcoming years both academic research and commercial investment in 5G technologies will be significant. The US government alone will be investing $400 million into 5G research [22] and the UK is even going beyond that with a $740 million (1 billion pounds) investment [23]. These investments, along with industry ventures, will begin to tackle the problems still left unsolved for the next generation of networks.

1.2 State-of-the-Art

As discussed in the previous section, there is a wide range of applications in 5G networks and research trying to address challenges. However, in this dissertation we focus on dense deployments in heterogeneous networks, and physical layer prototyping of these dense networks with SDRs. In this section we will provide a brief overview of currently adopted technologies that are foundational for 5G, as well as discussions on related topics. This background will help frame the remaining chapters, providing perspectives on current network designs and future enhancements researchers hope to make.

To provide this increased capacity and coverage requires new deployments models beyond only traditional large cells. This introduces the main area of discussion and research of this dissertation, where we investigate hybrid deployment models and network designs. Heterogeneous networks (HetNets) are of specific interest, since they can provide significant capacity gains over traditional networks by exploiting spatial reuse. Spatial reuse increases the number of cells per geographical area through new BS deployments. These deployments will be made up of smaller and less powerful BSs called Femtocell BSs (FBS) and Picocell BSs (PBS), are placed within the transmission domain of larger Macrocell BSs (MBS). However, due to this intracell overlap these deployment will not be without issues.

Figure 1.5 provides a simple example of a HetNet deployment with a single MBS and many FBS. In this type of network, mobile users can be offloaded to nearby FBS instead of all competing for MBS resources, thus drastically changing the user per cell ratio (UCR) to be more favorable. Such network designs have existed in LTE-Advances releases [24], but have yet to be densely commercially deployed as discussed in Section 1.1. However,

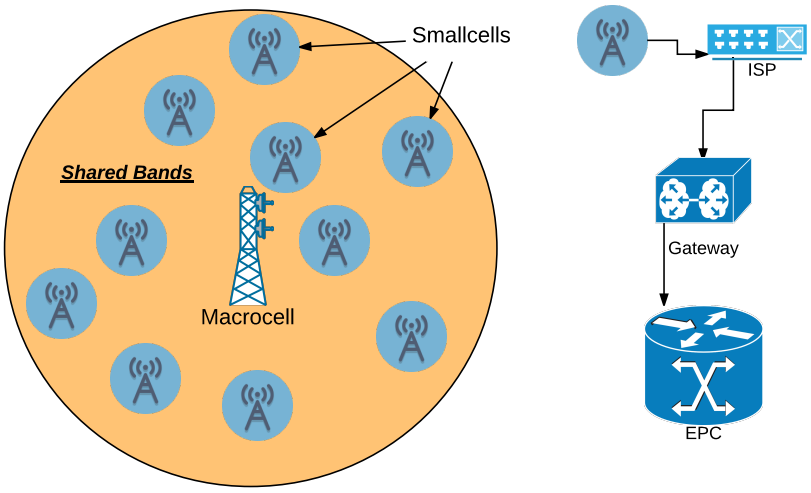


Figure 1.4: Example intercell deployment of MBS with many FBS. FBS utilize Internet backbones to communicate with the LTE Evolved Packet Core (EPC).

with 5G and the introduction of *mmWave* communications, nearby smallcells have become a network requirement for mobile connectivity. This is a direct result of *mmWave* operational bands, which can have significant propagation loss [25] especially in moisture rich environments. FBS can be deployed as Wi-Fi router replacements in consumer homes, or even more generally in public places such as airports or congregation areas. However, their topology compared with MBS is considered unstructured.

When considering traditional, or sub 10 GHz communications, as deployment densities increase that provides new aspects to the network. First, with many more BS new applications can be implemented; such as multi-point communication links, localization, and generally more resources at the BS can be spent on individual users. However, with many more BS devices, co-channel interference will increase and due to the scheduled nature of cellular access, primarily in the down-link, serious collisions can occur. Currently LTE releases 10 and 12 provide inter-cell interference coordination (ICIC) and evolved ICIC (eICIC) [26] to help manage this inter-cell interference. ICIC primarily operates by uniquely scheduling users with adjacent MBS on cell edges in pre-defined resource blocks. eICIC extends this concept to utilize almost blank subframes (ABS) at different but known intervals so FBS or PBS can schedule users on their edges into these resource blocks. However, co-tier interference remains in the lower tiers of HetNets and cell edge users can still be impacted by

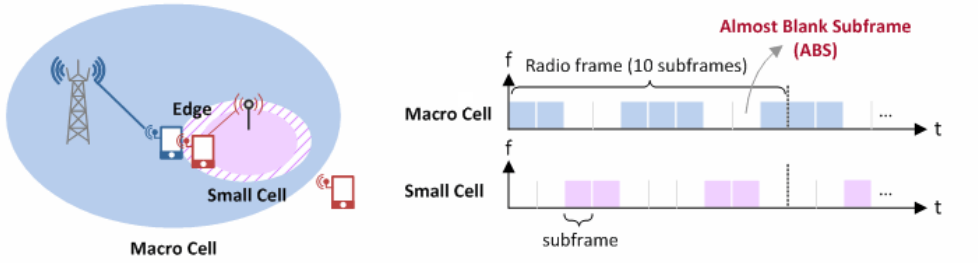


Figure 1.5: eICIC scheduling example using ABS between a single MBS and a single FBS [28].

environmental conditions or miss-configurations [27]. Co-tier interference has a high probability in dense residential or apartment complexes with many closely deployed subscribers. When we expand this discussion to 5G networks, the same problems exist but may be less impact full because of *mmWave* propagation issues. However, due to unknown deployment strategies, control of these interference domains may become important.

The second aspect of this research is based in physical simulation prototyping, allowing design verification and design proof of concept, which will become important due to the environmental considerations in 5G. Currently, for link-level and local area physical simulations there are three major products or projects for implementing system: MATLAB/Simulink, GNURadio, and LabVIEW. All three provide varying levels of performance for P2P links, and limited support for building up bidirectional communications. However, each platform does provide a library of signal processing and data manipulation tools. These tools dominate most of landscape for SDR prototyping, or at least early stage prototyping. MATLAB specifically has extensive support for WLAN [29] and LTE standard waveform generation and recovery [30]. However, real-time operational performance is almost nonexistent at minimum bandwidth specifications for the SDR hardware. Nonetheless, these platforms have begun to expand into the heterogeneous computing architectures to offload computations to FPGA or DSP devices [31]. With their current offerings such offloading are very time consuming to implement and require domain specific knowledge. Therefore, even decades old waveforms can be difficult to realize with the available tools for the common engineer or researcher.

SDRs themselves have become more capable at lower costs. The market now includes

such devices as the USRP [32], LimeSDR [33], FMComms [34], WARP [35], among many others. However, most are supported by the same tool sets already discussed essentially making them limited by those workflows and software capabilities. The obvious paradigm in progressive designs is the use of heterogeneous processing units to enable higher processing rates. To support modern standards and future waveform developments requires larger bandwidth, and therefore more computational power. As a result, large FPGAs, DSPs, and ARM based processors have begun to appear on these devices. However, these devices can be vastly more difficult to utilize and program compared to general purpose processors. Such accelerator type devices can also dramatically increase the cost of an SDR.

In summary, there exists several technical challenges when operating dense HetNets in spite of the advantages associated with current LTE and LTE-A standards. The major challenge is co-tier interference modeling and management in these non-traditional environments. Physical simulation of these links is also still difficult, even with modern advances in SDR technology. Other technical challenges associated with HetNet and physical simulation prototyping are:

- Dynamic loading between tiers and effectively selection of ABS length is difficult to manage to maintain capacity requirements;
- Mobility and handover management with smallcells, specifically when FBS count is high, makes adaptive scheduling computationally complex;
- Unstructured deployments of FBS makes intercell interference more complicated since FBS cross MBS cell boundaries and localized interference can be unknown outside of cell boundaries;
- Alternative resource management techniques such as channelization reduce overall capacity but decrease interference across tier levels but co-tier interference still remains.
- Network level physical simulations are scarce or even non-existent in commercial products utilizing SDR hardware;
- Bandwidth and latency control is difficult with general purpose processor-only designs

and heterogeneous programming still remains non-mainstream or difficult in software packages for SDRs.

In this research we focus on three specific technical challenges for enabling 5G technologies and research:

- Prototyping difficulty of link level communications based on standardized systems for physical simulation.
- Modeling of large deployments spatially with standards base feature set, and future capabilities.
- Applications made possible by 5G dense deployments with a focus on localization.

1.3 Research Questions To Be Answered

The main objective of this dissertation is to advance research of 5G systems further from two distinct perspectives. First, a top down analytical approach that examines interference impacts in dense networks and techniques for reducing affects co-located links. Second, practical implementations and tools for prototyping 5G applications with SDR devices. Therefore, both approaches provide several research questions to be answered, namely:

- What is the affect of co-channel interference in NetNets? What techniques can be used to reduce this interference and what is their effectiveness?
- What are shortcoming in existing tools for prototyping wireless devices and waveforms? How can we provide acceptable performance to such tools without making them difficult or time-consuming to use?
- Can client offloaded tasks to the network be performed reliably and on reasonable hardware? What kind of performance can we expect of real-world applications for clients in these dense networks?

1.4 Research Contributions

Based on the understanding of the state-of-the-art, this PhD dissertation will provide novel strategies for modeling of HetNets stochastically and link level prototyping of physical simulations. The contributions of the research so far are the following:

- **A dataflow algorithm acceleration framework for MATLAB (Chapter 3)**: A proposed framework for algorithm acceleration for real-time signal processing and communications. Here, a function level general concurrency environment is implemented, taking advantage of multi-core CPU architectures, made available to MATLAB user-space. A real-time 802.11a receiver was built with this framework using Software-Defined Radios (SDR) to recover real-world signals.

- **Modeling of K-Tier HetNets using stochastic techniques (Chapter 4)**: Networks of MBSs and FBSs are spatially modeled jointly, and coverage performance is evaluated for connected users. Heterogeneous transmit powers and tier densities are contrasted for a growing FBS population. Finally, directional antennas are explored to reduce interference effect of new FBS on MBS.

- **Prototyping of localization in dense FBS networks (Chapter 5)**: SDR prototype is implemented for real-time direction finding and localization through coordination. Traditional direction of arrival (DoA) are contrasted against an OFDM based architecture, both implemented and tested with real world signals. Indoor and idealized chamber based measurements are explored.

1.5 Dissertation Outline

Chapter 2 discusses the current state-of-the-art for cellular network modeling, and physical simulation techniques and rapid prototyping in SDR. In Chapter 3, a novel prototyping framework for real-time signal processing is presented. First, a theoretical analysis of concurrency in computation is provided for two parallel paradigms. Next, implementation details are explained which include MATLAB integration, as well as the programming interface from MATLAB. Followed by performance analysis of the framework itself and a

demonstration of gains over serialized code. Finally, a case study of a 802.11a receiver built with this framework is discussed.

In Chapter 4, stochastic network modeling is discussed for HetNets. A new mixed point process model is developed specifically for MBS/FBS tiering scenarios. Model descriptions and reasoning is provided for the following analysis using stochastic geometry techniques. Overall derivations for coverage are provided for two point process (PP) models, which include directional antennas, and network tiering. Then, spatial capacity is discuss with this network model certain network scenarios focused on MBS and FBS deployments. Finally, deployment strategies are discussed based on the results of this analysis.

In Chapter 5, a prototype real-time localization system is constructed and evaluated. The analysis focuses on development strategies and numerical performance of different components as well as the overall SDR system. An automated phase synchronization harness was developed to provided the required phase coherence of different SDR receiver chains. Additional antenna compensation provides calibration using transmitters at known positions.

1.6 List of Related Publications

List of Journal Papers

- T. F. Collins, and A. M. Wyglinski, "MATLAB DataFlow: Accelerating Algorithms Through Acceleration" IEEE Access, Under Second Review, 2016.
- T. F. Collins, S. Pagadarai, and A. M. Wyglinski, "Directional Antennas in Closed K-Tier HetNets," IEEE Transactions On Vehicular Technology, Under Review, 2016.
- T. F. Collins, S. Pagadarai, and A. M. Wyglinski, "Direction of Arrival Estimation SDR Testbed," Wiley Wireless Communications and Mobile Computing, In Preparation, 2016.

List of Conference Papers

- T. F. Collins and A. M. Wyglinski, "Co-Channel Interference in Future Femtocell

Networks," in Vehicular Technology Conference (VTC Fall), 2015 IEEE 82nd, Sept 2015.

- T. F. Collins and A. M. Wyglinski, "Skynet: SDR-Based Physical Simulation Testbed," in Vehicular Technology Conference (VTC Fall), 2015 IEEE 82nd, Sept 2015.

Chapter 2

Background Fundamentals

The overarching topics discussed in this work focus around smallcells or FBS. This work examines the effects that small-cells have on an existing network, their use in different sensing strategies, and architectures for performing physical simulation link level analysis. This chapter provides some background knowledge on the existing literature and tooling in these research areas, for which will be expanded upon in the following chapters. Furthermore, since SDR prototyping is the dominant platform for radio development and testing two of the three subjects discussed in this dissertation focus on the SDR area. These SDR topics are approached from two perspectives: the first addresses algorithm acceleration for SDR, and the second uses SDR to perform real-time direction finding and localization. The third topic is a pure analytical analysis of HetNets using stochastic geometry techniques. In general, this research approaches HetNets from a top down and bottom up perspectives, pushing the literature and prototyping capability closer together. Due to the complexity of current network designs and new technologies to be integrated, these perspectives are logical.

2.1 Software-Defined Radio Prototyping Technologies

In this section, prototyping of wireless radio technologies is addressed from a computational perspective. Communication systems in general have been at the forefront of computational speed and efficiency, and even today development of capable hardware and software

is of great interest to commercial and industrial applications. However, software and hardware do not always go instep with one another. Therefore, in this work current software frameworks and how they take advantage of hardware designs are discussed. In Chapter 3 a new framework will be introduced to help reduce this disparity, while reducing developer overhead. As mentioned previously, SDR development will be an overarching theme in this work and is more importantly itself highly focused on computational complexity.

With modern consumer CPU architectures growing in core count instead of higher core clocks, system throughput can be efficiently maximized by expanding an algorithm over many cores. The applications of interest are with respect to areas signal processing, but such ideas can be expanded to other high performance compute applications, such as machine learning, data science, financial analysis, and other similar applications. Signal processing naturally fits into this coding architecture for applications such as signal recovery, tracking, and decoding. In these applications, the tasks can be pipelined, providing increased throughput or reduced output latency, assuming the algorithm is spanned over many cores instead of a single thread.

Parallelism can be performed in computation at many different levels: At the lowest level, modern single CPU cores can execute several basic operations simultaneously with single instruction, multiple data (SIMD) specialized commands [36], or more generically known as vector commands. Programming with a domain-specific language (DSL) provides fast development and natural problem formulation, but can be limiting on actual computational features. In this dissertation we focus primarily on MATLAB [37], but other DSLs such as *R* [38] and *Octave* [39] share in the fact that they provide limited multicore exploitation. This can be even difficult to realize in more general languages that maintain scientific libraries such as *Python* [40] with *SciPy* [41]. It is important to note that behind many functions in these languages/products are implementations of external BLAS (level 3) libraries that use multiple threads to do parts of basic vector and matrix operations in parallel [42], and even higher level functionality is available through *OpenMP* or *pthreads* [43] in these products. However, essentially all general purpose programming languages increase performance with two categorical parallelism approaches, which are vectorization and concurrency/threading. Unfortunately, these implementations require knowledge of the under-

lying hardware, and in many cases are not portable between platforms. Although, if these approaches can be exploited significant gains can be made. An example of this results is in Figure 2.1, where we examine the computation time of a square matrix multiplication of OpenBLAS [44] versus generic loop code. As the matrix size increases, there is obvious performance benefit of vectorization provided by OpenBLAS.

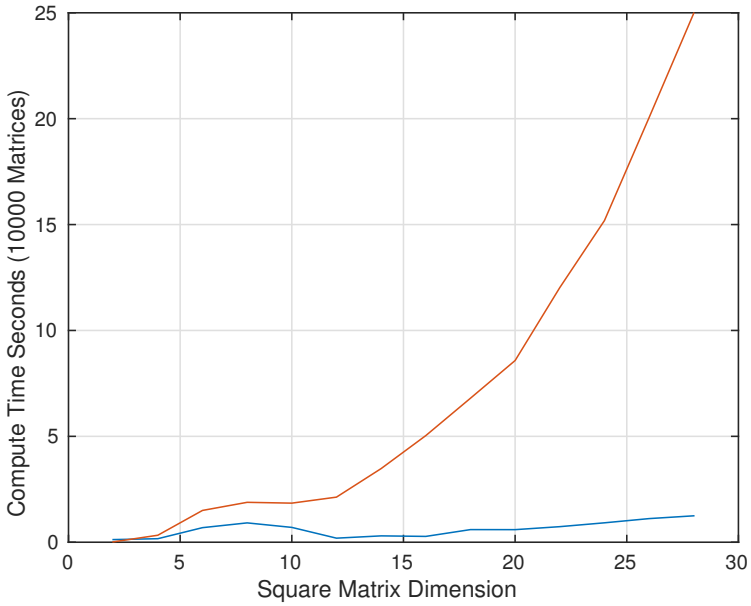


Figure 2.1: Comparision of matrix multiplication of two square matrices ($A = B \times C$), with loops (red) vs. BLAS (blue) vector calls (dgemm).

With the growth of research areas such a Big Data and Machine Learning, it has become more important to utilize parallelism when possible. As a result, we have seen the introduction of Google's *TensorFlow* project [45], *Apache Beam* [46], and migrations to functional programming languages that naturally fit into flow based architectures [47]. Algorithmic acceleration has seen explosive growth in the hardware sectors with GPU and FPGA accelerators becoming more common, and their software architectures (CUDA, OpenCL, HDL/Verilog) becoming easier to use [48]. Such accelerators naturally fit into these flow paradigms, since they operate in tandem with existing general purpose processor operations. As a result, developers can partition algorithms appropriately where highly parallelized task can run on accelerators and more complex serial computations can be general purpose processor

targeted.

However, even though dataflow architectures do exist in many implementations, their usability is questionable. A comparison is made in Figure 2.2 between several of the discussed platforms, primarily those used for signal processing. GNURadio is an extremely capable tool, but has a very large learning curve. In direct contrast to GNURadio is LabVIEW, which relies on a large number of built-in functions to perform process, and technically requires no programming to perform similar processing. However, if built-in functions do not exist, they can be very challenging to be implemented by the user. Therefore, we can

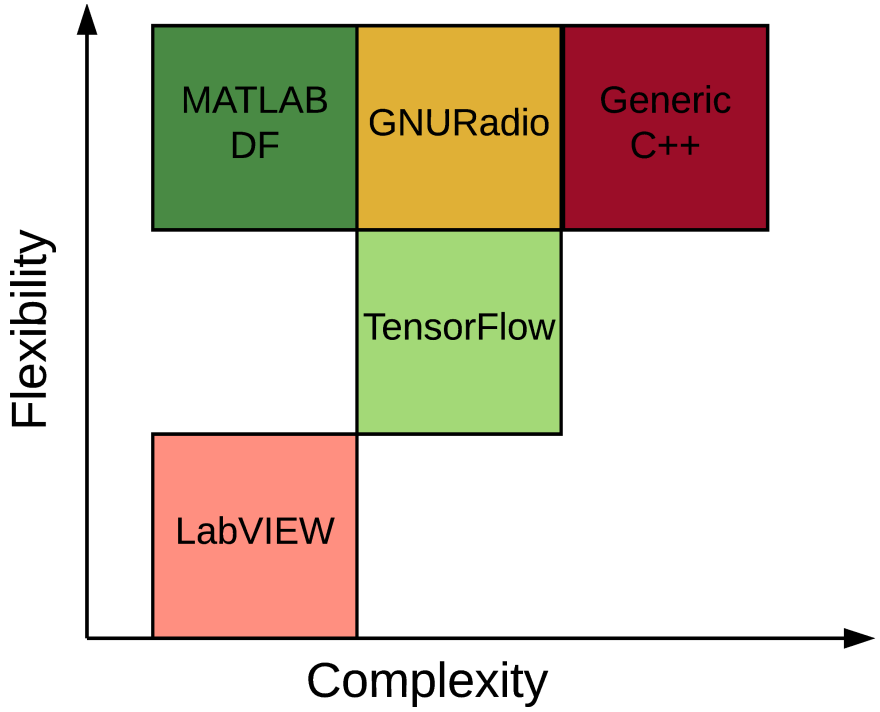


Figure 2.2: Comparision of dataflow architectures in terms of complexity to use vs. their capability as a platform to implement general mathematical processing.

observe this trade-off of capability and usability. DSLs lean toward the side of usability, will frameworks or tools exposing the underlying languages have vastly more capability and control. Now since these tools do not exist in isolation, most developer or scientists tend to design in one platform and deploy in another. It can be challenging for DSLs specifically to cross the boundary into this deployment stage. General languages just tend to inconvenience users, require more work or code to accomplish the same goals.

Focusing on MATLAB specifically, parallelism exploitation has been introduced in many forms but can be categories into two main areas. First are highly data independent implementations, where work can be spread over many cores and many machines. The two main projects in this area include pMATLAB [49] and MathWorks' own Parallel Computing Toolbox [50]. Second are alternative MATLAB language compilers, which convert MATLAB code into C/C++, fortran, or specialized kernels which are parallelized. Such designs include *MATISSE* [51], *MEGHA* [52], and *StencilPaC* [53]. Additional extensions have been added to *MEGHA* introducing runtime thread management with Intel's Threading Building Blocks [54].

In Chapter 3, we will present an architecture to expand MATLAB in order to enable DF based parallelism and general concurrency for the purpose of stream processing acceleration and task concurrency. Helping MATLAB cross the gap into more real-time operations. We provide a user configurable flowgraph, where functions produced from MATLAB are run concurrently and pass data between one another. The goal of this framework is to handle the threading and data passing between MATLAB functions, such that developers can focus on algorithmic aspects of their code rather than handling data flow. MATLAB itself is a dominant tool in data sciences, engineering, and many other fields, reaching the top ten of IEEE Spectrum's programming languages [11]. Currently, MATLAB does not offer any DF or simpler cascaded tasked based parallelism that are available to the user. Expanding MATLAB to include this framework can provide significant acceleration of current scientific work, without difficulties of other languages/frameworks that do provide this capability.

2.2 Stochastic Network Models

In this section, stochastic network modeling is discussed to provide additional motivation and background for Chapter 4 where we will examine HetNets at a macro-level. In general stochastic analysis for cellular networks has been heavily studied, but can be considered a rather minute area of research with a small group of prolific authors. However, results can invaluable for network designers trying to understand network wide effects.

Nonetheless, with the ever increasing demand for wireless capacity and coverage [55]

it is important that the wireless community understands all aspects of wireless resources. Reaching the limits of spectral efficiency only provides a fraction of available capacity in current cellular network topologies. For instance, the capacity enhancement for next generation HetNets comes from the exploitation of aggressive spatial reuse and tiered networks. Furthermore, the densification of base stations from additional deployments of picocells and femtocells can provide significant additional capacity without radical changes to current air interfaces. Such densification comes at the cost of challenging deployment scenarios, as well as significantly increased co-channel interference. In such networks, interference becomes the capacity limiting factor rather than the traditional environmental noise. Fortunately, unlike environmental noise, interference can be controlled or reduced with the help of other technologies, such as directional antennas. Using beam steering, operators can reduce interference to spatially close nodes, or providing a higher degree of densification.

The modeling of such networks can be difficult due to spatial correlations and variance of existing network topologies/configurations. Over the past last decade stochastic geometry (SG), with its rich analytically tool-set, has proven to be an invaluable method for modeling both cellular and Wi-Fi networks [56]. Uncoupled from the deficiencies of fixed spatial configurations, stochastic geometrical models can capture the randomness of cellular deployments instead of utilizing the fixed unrealistic hexagonal model. This is visually compared in Figure 2.3 and 2.4, with a perfect hexagonal deployment and a random/more realistic deployment respectively. This can have significant impact on coverage, and SG allows modeling of these spatial effects.

SG is simply a tool for applying spatial average for the positions of nodes. It possesses a strong basis in applied probability, and general randomness across spatial places across many dimensions. However, most of the literature on wireless network particularly has focused on two or three dimensions only. The analysis itself is technically a branch of point process (PP) theory, which is a natural abstraction for a network of many or infinite nodes. Although most of the growth in the literature has occurred in the late 2000s, the application of percolation theory, a subset of SG, to wireless networks has been seen as early as 1961 [57]. Percolation theory itself is a extremely analytical field of mathematics, with other uses in Fractal Geometry and Graph Theory.

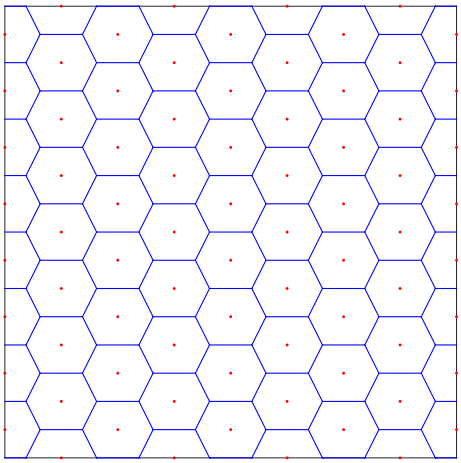


Figure 2.3: Hexagonal cell layout of an ideal network, maximizing spatial coverage.

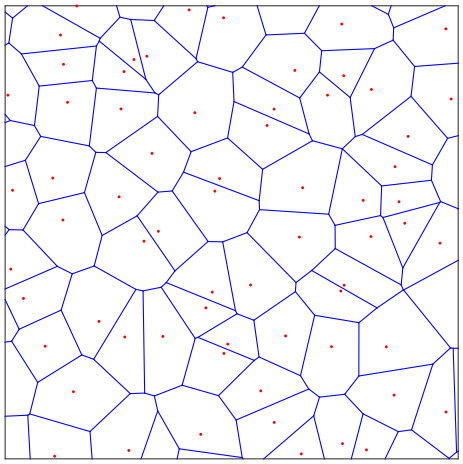


Figure 2.4: Realistic cell layout, with cell exhibiting some visible structure.

A majority of the fundamental application of SG in wireless networks has been concentrated in a a few monographs, primarily by a relatively small group of researchers working in the area [58–60]. Covering the fundamental theory with a focus on modeling based on the Poisson PP (PPP). However, in our own work in Chapter 4 we extend to other PP and applications of communication system details applied to PP theory.

The current related literature to Chapter 4 in this dissertation can be classified into three areas. The first is applications of directional antennas in wireless network-level analysis. Under the stochastic geometry (SG) framework, only PPP type network models have been studied with directional antennas. In [61], ad-hoc networks were studied, where PPP nodes transmit to a desired receivers at fixed distances with fixed patterns. Unique antenna patterns were utilized, and significant effort was spent understanding direction of interference, which is important in ad-hoc and multi-hop type networks. In [62], orientation error was introduced to model beam misalignment, thus providing some insight into capacity reduction for such scenarios. Earlier works include [63], which derived coverage probability for K-tier PPP network with fixed directional antennas in an open access network.

Research focused on tiering network include [64] where open-subscriber PPP networks are considered. Closely related is [65], which uses a more tractable but inaccurate at low SINR PPP model. Most K-tier work utilizes PPP, primarily since they are tractable and

tiers can be modeled independently. The dominant model is called the heterogeneous independent process (HIP) model, which utilizes a coloring concept to model multiple PPP tiers as a single large PPP. Other modifications include cross tier dependence models [66], where MBS have exclusion regions preventing FBSs from being located closely to a MBS. Such models rely on cluster PP (CPP) or hardcore PP (HCPP).

For works focused on soft-core processes in wireless networks, much of the fundamental theory has only been published within the last few years. The literature has focused on the application of determinantal PP (DPP), which have foundations in quantum mechanics for modeling fermions [67]. These PP are useful since they exhibit repulsive properties, provide more regularity to their arrangements. In this work, we focus on two deployment models, the first based upon the PPP, and the second a DPP the Ginibre Point Process (GPP). The PPP is a common model in SG due to its tractability. It can be considered appropriate for femtocell tier networks due to their absence of deployment strategy. However, the PPP is not appropriate for macrocell, or even picocell, base stations [68]. Therefore, we utilized the GPP, which has become popular recently in the literature for modeling more structured deployments. The original work for modeling wireless networks using the GPP were developed here [69], and extended by [68] to include a more general βGPP case. For other DPPs, [70] provided coverage analysis in a comparative study against real deployment data sets. To our knowledge no existing work has been published on integrating the topics of directional antennas with GPP, or more generally, soft-core point process network models, as well as mixed PP types.

2.3 Real-Time Localization

In this section the application of direction finding and localization is discussed, applied specifically for dense deployments of femtocells. As discussed in Section 2.2 future cellular networks will take advantage of spatial reuse in-order to increase capacity, which provides opportunities to take advantage of this increased ratio of tower to mobile. Such conclusions for applications can be drawn from future looking white papers [71, 72]. The application chosen to take advantage of this densification is localization, precisely support for “always

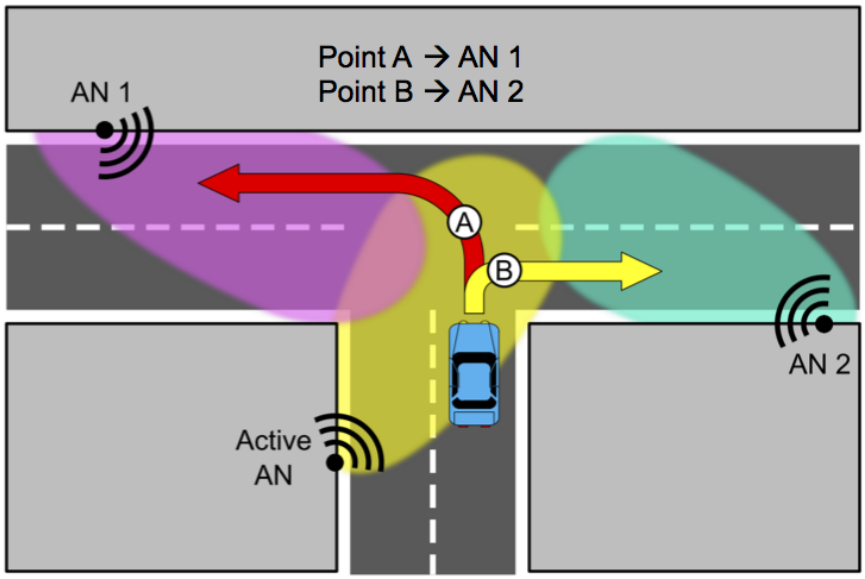


Figure 2.5: Beam-steering in localization-aware network providing pro-active resource management of spectrum in order to reduce interference. Resulting in less congested links. [73]

on" highly accurate positioning user locations, making them available at any given moment without draining the user equipment batteries. However, this presents challenges since traditional networks, or more accurately smallcell deployments, have limited resources to accomplish localization. This research focuses on understanding the real-world requirements for such smallcells, and specifically how direction finding and localization can be performed in practice.

With localization-based information, many optimizations can be made from the perspective of the network. Two obvious applications are interference reduction and load balancing. Through the use of beam-steering smallcell can reduce their transmission footprint to cover only the appropriate areas. This concept is demonstrated in Figure 2.5, where beam overlapping is limited and even handover is optimized by reduce coverage dead-zones. Load balancing can be achieved by turning off individual smallcells or lowering their power, freeing up channels for other nearby smallcells with significant user connections. Since smallcells are likely to be place in a suboptimal way or non-regular way, these techniques will become important since MAC control is more strict than in random access networks like Wi-Fi. Beyond traffic and capacity other possible application include security through

localization, removing spoofing on client side positioning. Driver assisted navigation, with significant applications for smart/autonomous cars. Enabling position information on the network side can provide many benefits to many more applications.

Depending on the sophistication of the localization performed, intrinsically it can be thought of as an extension to time of arrival (ToA) or direction of arrival (DoA) estimation. As the number of estimating nodes increases so does the dimension of the localization planes for which a transmitting node can be tracked. Historic DoA and ToA estimation have been based in radar processing, applied to arrays with large numbers of antenna elements. However arrays are not always required for localization, primarily for ToA or time difference of arrival (TDoA). A common inexpensive method of localization is with received signal strength indicators (RSSI). RSSI data requires no fancy arrays or even specialized receiver hardware, and many of these measurements can be accessed directly through user space APIs. Two primary methods exist for performing localization with RSSI, ranging and fingerprinting. Ranging, as implemented [74], simply estimates the distance based on transmit power. Alternatively, fingerprinting, as shown in [75], requires a mapping of individual locations to specific power levels or RSSI values before hand. However, utilizing RSSI from commercial network interfaces can be inaccurate. There is no standardized relationship of any particular physical parameter to the RSSI value produce by an individual network card. For Wi-Fi, the IEEE 802.11 standard does not define any relationship between RSSI value and measured power level units [76]. One subtlety of the IEEE 802.11 RSSI metric comes from how it is sampled. RSSI is acquired during only the preamble stage of receiving an 802.11 frame, not over the full frame [76]. Therefore results can be questionable depending on estimation technique used.

Now considering modern Wi-Fi and cellular hardware with multiple antennas, more accurate estimation methods can be implemented. Radar DoA algorithms can be implemented around these hardware models, and those of particular interest here are classical and subspace methods. Different DoA methods are related in Figure 2.6, outlining search and algebraic based. It is important to known that search based algorithms are usually applicable to any array structure, unlike algebraic versions which rely on specific arrangements. For example, Root-MUSIC can only be applied to uniform linear arrays (ULA) [79], unlike

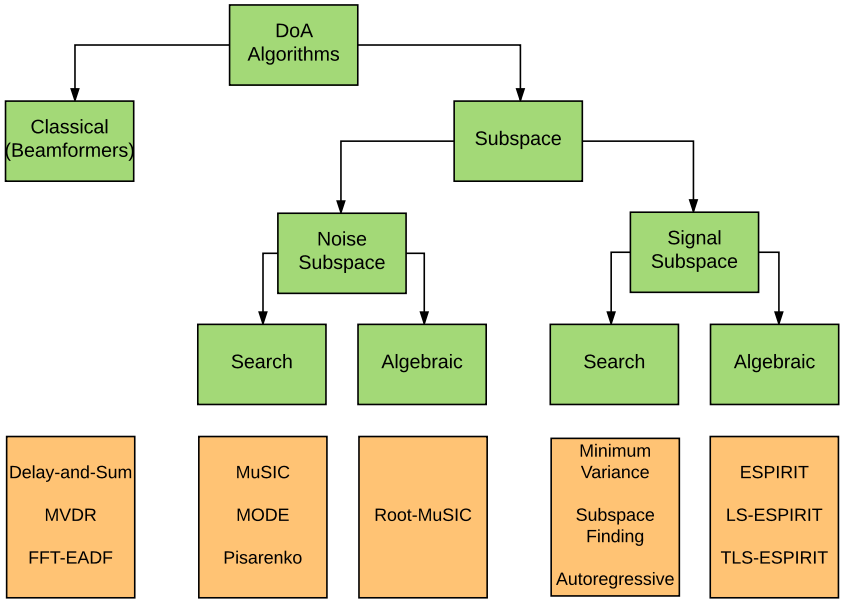


Figure 2.6: Breakdown of common DoA methods including classical, subspace, and Algebraic [77, 78].

MUSIC, which can by applied to any array structure. However, Root-MUSIC provides only single numerical outputs for an estimate, where as MUSIC provides a vector of angles and their likelihoods. These must be searched through, as the name suggests. Classical, also referenced as beamformer algorithms, utilize correlation based approaches rather than subspace processing and handle general array structures [79].

Implementations of these algorithms have historically been limited to military research due to their generally expensive designs with many receivers. However, with new additions in the SDR market these experimental setups are becoming more feasible to academic institutions or even small businesses [80]. Nonetheless, no off-the-shelf implementation exist for real-time DoA or localization. In Chapter 5, real-time DoA estimation is examined with two different methods. First, through traditional work-horse MUSIC algorithms a performance baseline is provided. Then an EADF-FFT algorithm which utilizes channel estimates of a existing communication system is explored. Significant emphasis on physical requirements of the hardware is outlined, along with how these issues were solved with

common SDR hardware and software. Overall, this work is used to present a realistic physical layer for localization in a real-world environmental focused in DoA estimation.

Chapter 3

Data-Flow Algorithm Acceleration

To enable prototyping of modern waveforms requires significant computational power and complex signal processing. From previous work developing communication applications here [81], it became obvious that architectural changes needed to be made to accommodate expansion into more advanced signal processing techniques and network simulation. This chapter addresses this computational bottleneck in MATLAB specifically for SDR prototyping, and provides a unique solution that is applicable also beyond real-time signal processing. This work in general enhances the signal processing libraries of MATLAB to provide real-time performance during development of many applications. Such enhancements help drive faster development for new standards, such as 5G, reducing translation time from MATLAB simulation to physical testing. We have developed the proposed framework for our own SDR development to provide real-time receiver designs. However, the proposed framework is generic and can be applied to any problem with large computational requirements looking to exploit both data dependent and independent parallelism. Such applications include phased array processing, machine learning, data analytics, image processing, and many more.

3.1 Architecture Overview

This Chapter provides the following contributions to existing tools and literature:

- Implementation of a novel DF framework via the MATLAB environment with significant language support.
- Performance analysis of the proposed framework itself and application level speedups possible over current serial code.
- A simplified workflow designed to fit algorithms and current MATLAB code into this framework, including load distribution over the cores of a given machine.
- Discussions on the limitations of the proposed framework and the drawbacks of utilizing some of the associated tooling.
- Presentation of a case study application of a 802.11a receiver implemented using this proposed framework.

3.2 Parallel Architecture Systems

This work focuses on providing a DF parallelism framework to MATLAB, which we call MATLAB Data-Flow (MLDF). The goal of MLDF is to provide increase computational performance, but also limit latency through careful thread orchestration. DF, which can have several meanings in computing, is defined in this work as the division of computational stages into concurrent tasks that can have dependent or independent execution. DF from a simplistic viewpoint is a combination of task and data based parallelism, utilizing both techniques in a cascaded arrangement to increase numerical computation through concurrency. First, it is important to understand the differences between task parallelism (TP) and the more commonly implemented data parallelism (DP). In DP applications, we assume the same operations are performed on different sets of data in parallel across multiple cores. DP traditionally does not share state or handle data dependence across threads or processes, and such operations are considered very computationally expensive. In task-based parallelism, each concurrently running operational unit can have a different function. Generally TP is the simultaneous execution on multiple cores of many different functions across the same or different datasets. Unlike data parallel functions, task parallel functions can be cascaded together in series, essentially pipelining operations, analogous to an assembly

line. Existing frameworks such as OpenMP [43] provide this functionality through parallel for-loops in the case of DP, and as *jobs* or *spawns* in the case of TP, data handling can be limited or expensive in these libraries, with projects like *GNURadio* [19] or *Cilk* [82] providing more finely controlled data passing during concurrency. Such problems have been researched extensively in the literature for more general languages (C/C++) with specialized implementations [83–85]. The goal in this work is to leverage this knowledge for an appropriate integration with MATLAB.

However, since this research is targeted at MATLAB users specifically, it is important to understand the performance benefits of DF. We approach is from the extreme perspectives of DP and TP. Each implementation has their advantages and drawbacks, but for fairness when comparing DP to TP we assume that the number of DP cores is equal to the number of TP functions, unless otherwise specified. When core communication is overlooked, DP and TP will always meet or outperform non-parallel implements, but this will be considered later in this work as well.

To provide a better understanding between these parallel architectures, let us consider a simple example. Note that in a real-world scenario the operations would be much more computationally intense. Figure 3.1 and Figure 3.2 contrast the different structures of parallelism for an evaluation of a simple example equation:

$$f(x) = 3(x+1)^2 - 4, \tag{3.1}$$

where we wish to compute four realizations given different input values (x). As you can see in Figure 3.2, we split the work of each core to be a specific mathematical operation. Therefore, all data passes through each core. In Figure 3.1, we instead supply different data to each core. The design or selection of parallel sections is not always simple to address or optimize due to mismatches in core count and available operations. Therefore, depending on the computational requirements of the algorithms, implementations of TP and DP can perform differently. From an implementers perspective, DP can be more easily applied to problems due to its lack of additional communication required between cores.

If all data to be process is available at the start of the simulation, then DP has an obvious advantage, since there is no startup latency. This provides a direct speedup of

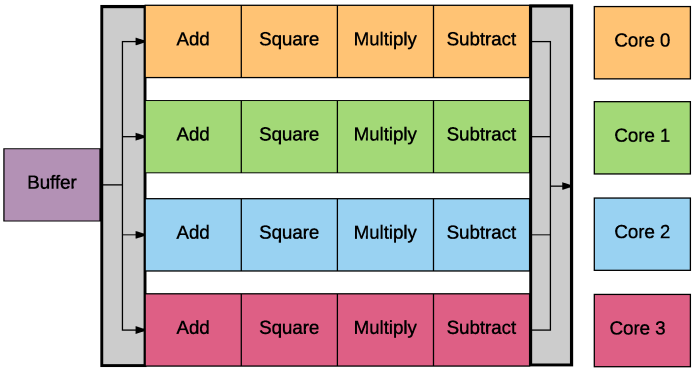


Figure 3.1: Data parallelism realization for equation (3.1) on a four core system. In the case of a streaming application the buffer block will be utilized to delay input data until four samples are available.

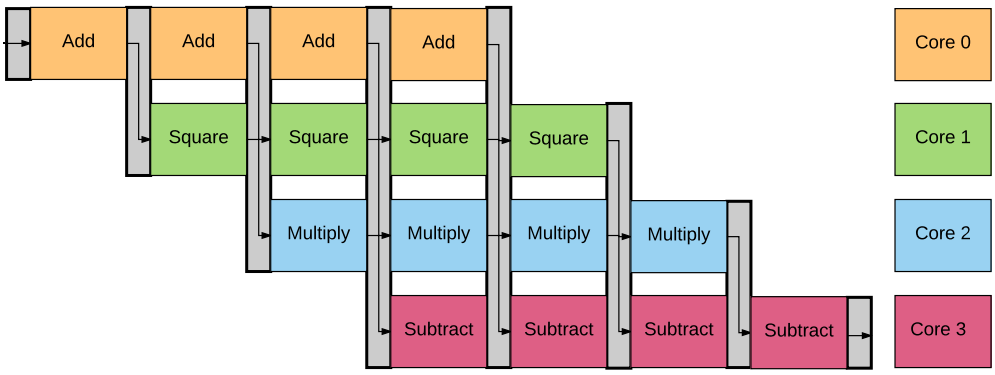


Figure 3.2: Cascaded task parallelism realization for equation (3.1) on a four core system. If we condition on each core having a different function, this realization is the most efficient.

$\sim N_c$ over the single core alternative where N_c is the number of available cores. However, this type of scenario is not always possible. For example, streaming applications generate data over time and tend to have latency or data dependency requirements. The amount of data to be processed in these types of applications can also be unknown. First, let us assume that the overall system stream based and is able to operate in real-time. Therefore, with TP the time between new data arrivals Δt_n and the largest processing time of any individual core/function for that much data $\Delta t_{TP,MAX}$ must have the following relation: $\Delta t_n > \Delta t_{TP,MAX} + \Delta t_{TP,OH}$, where $\Delta t_{TP,OH}$ is the overhead for block data passing. For DP, the processing time of any individual core for that much data Δt_{DP} must have the following relation: $\Delta t_n > \Delta t_{DP}/N_c + \Delta t_{DP,OH}$, where $\Delta t_{DP,OH}$ is the DP overhead. This is true since DP is able to buffer at most N_c data units. Now, the combined TP stages have equal latency as a single DP thread/process $\sum_{k=1}^{K} \Delta t_{TP,k} = \Delta t_{DP}$, where K is the number of TP stages, but the per data unit latency is different.

Furthermore, under these real-time requirements we can determine the average latency to process a unit of data. Note, for real-time applications the overall system throughput is bound by the data arrival rate. For the DP and TP cases where $K = N_c$, the mean processing latencies for data processed by the system are:

$$t_{LAT,RT} = \begin{cases} N_c \Delta t_{TP,OH} + \Delta t_{DP} & TP \\ \frac{1}{N_c} \sum_{n=1}^{N_c} (n-1)\Delta t_n + \Delta t_{DP} + \Delta t_{DP,OH} & DP. \end{cases} \tag{3.2}$$

In Equation (3.2) we observe the startup or buffering latency of DP as $(N_c - 1)\Delta t_n + \Delta t_{DP,OH}$.

Now, in the case of non-streaming applications with a fixed dataset that is entirely available at the start of a simulation, we provide a processing comparison to streaming applications in Figure 3.3 and Figure 3.4. In Figure 3.3, DP needs to buffer inputs while TP can use them right away providing lower latency on average. In Figure 3.4, the throughputs are purely bound on Δt_{DP} and for equal comparison we define $\Delta t_{DP} = N_c \Delta t_{TP,MAX}$. When bounded by Δt_n the average throughput is better for TP, but when unbounded DP is better performing. This discrepancy in performance purely comes from the buffering that occurs in the system and processing lengths.

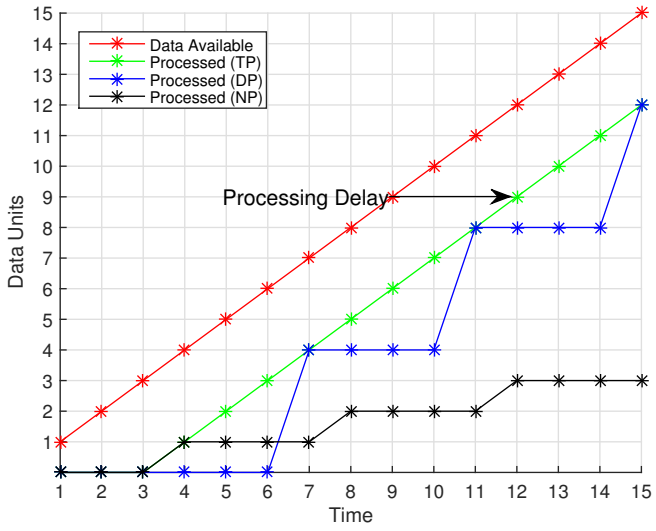


Figure 3.3: Processing analysis for a streaming type application comparing the performance of three implementation: data parallel (blue), task parallel (green), and single threaded where no parallelism is exploited (black). Data arrives at the system every one time unit and takes two unit to be processed.

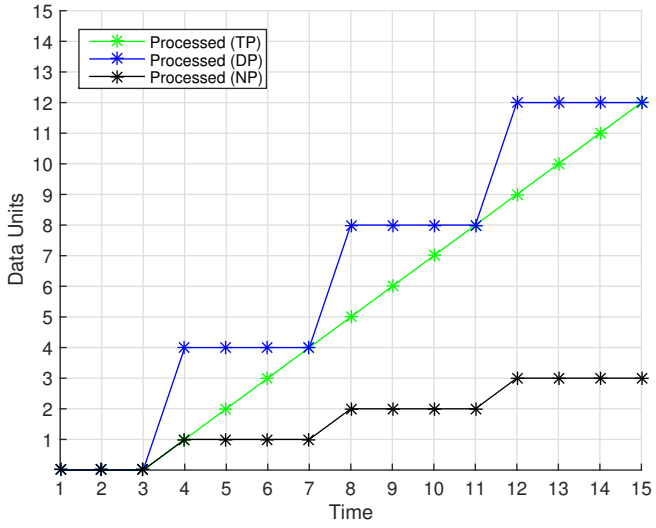


Figure 3.4: Processing analysis for a off-line type application, where all data is available initially. Three implementations are compared: data parallel (blue), task parallel (green), and single threaded where no parallelism is exploited (black). Data parallelism can utilize all processing units (four) initially.

3.2.1 Limitations

Given these insight on the structures and relative performance of DP and TP to non-parallel operations, we can consider the disadvantages of these implementations. The first and most obvious is the overhead required, which is depicted as the grey sections between blocks in Figures 3.1 and 3.2. Due to the asynchronous nature of threads, data must be passed between them in a safe method. This safety comes at the cost of speed. To reduce these overheads, we can either reduce the number of independent threads or increase the data passed at a given time. These have two downsides, first by reducing the number of threads we may not effectively utilize all processing cores. Second, by increasing the data passed between threads we increase the per-data unit latency. To provide gains over non-parallel implementations, for processing B data units we must maintain the following relation:

$$\begin{aligned} B\Delta t_{DP} > \Delta t_{LAT,RT}(TP)+ \\ (B-1)(\Delta t_{TP,K} + \Delta t_{TP,OH}). \end{aligned} \tag{3.3}$$

For useful applications the system should handle a large amount of data units ($B \gg 1$).

When comparing DP and TP, DP implementations tend to be more load confining, but have a reduced communication overhead. From Figures 3.1 and 3.2, we have 8 and 14 communications, respectively. In applications where we have an inherent data dependency, it may not be possible to have efficient data sizes evenly passed to DP threads. TP provides more flexibility and naturally handles data dependency. In particular, TP performs well when data passing between blocks is event driven and not deterministic, such as when streams of data have task boundaries that have time dependence. This is obvious for communication receiver algorithms which rely on all past data in order for perform actions. Such state sharing is difficult with DP, and comes with a significant performance penalties.

Since MLDF is a combination of TP and DP, we can utilizes the advantages of each. Primarily, we can cascade blocks where data dependency exists, and arrange other operation concurrently where there is no dependence. independent task for additional speedup. This independence did not exist in the first example we looked at in Equation (3.1), but instead

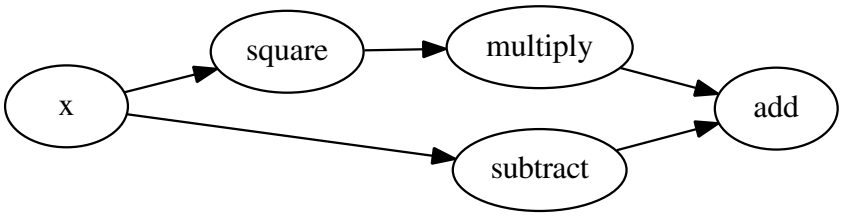


Figure 3.5: Mixed parallel flow for Equation (3.4), exploiting data parallel and task parallel advantages.

becomes obvious in an alternative example:

$$f(x) = 2x^2 + x - 1. \tag{3.4}$$

Here in Equation (3.4), we can compute the squaring operation and multiplication concurrently with the subtraction, but the final addition can only be done once all others have completed. This concurrency flow is illustrated in Figure 3.5. The considered Equations (3.1) and (3.4) are used here as simple examples to demonstrate concurrency of mathematical operations, and in our implementation we restrict these parallel components to the MATLAB functions only. In a realistic implementation, the operations outlined in Figure 3.5 should be replaced by computationally intensive functions, since passing data between these concurrency/threaded domains is not without penalty. This measured penalty will be discussed later in Section 3.4.1 of this paper. The defining aspect of this comparison comes down to data dependency. Whenever data dependence is required that portion of an algorithm should use TP, but without that requirement DP can be used.

3.3 Implementation

Now that we have a general understanding of the certain parallel architectures, we will examine how DF was implemented through the MATLAB language. This will include the underlying threading model, cross-thread data sharing, and integration with MATLAB Coder. Once the implementation details have been provided we will move on to the performance analysis of applications and the underlying architecture.

3.3.1 MATLAB Integration and Related Work

In this work, we expand MATLAB through the code generation tool to provide DF. As stated before, since DF is a super set of TP and DP, we get those extreme cases for free. However, the general DF framework provided here can take advantage of both TP and DP within the same implementation using this framework. To provide concurrency with efficient data sharing, this DF framework must implement threading. Nonetheless, since MATLAB does not inherently have threading tools that are exposed to users in the MATLAB language[1], and MATLAB's interpreter is not thread safe an alternative approach or language domain must be considered. Existing approaches in the literature have utilized external or custom compilers of the MATLAB language, where single vector operations are scaled across cores [54]. Alternatively, code replacement has been used to speedup individual functions or a language subset [51, 53], and even with targeting of hardware [52, 86], these approaches do provide significant speedup. However they only provide an extremely limited subset of MATLAB's functionality. This is especially true when considering MATLAB's toolboxes, which have extensive system object use [87]. The usage of such toolboxes has become the main use cases of the product, beyond just matrix math.

All of the discussed approaches for providing DF-like parallelism to MATLAB involve C/C++ generation/translation, which we utilize here in this framework. However, instead of relying on a custom compiler we utilize MATLAB Coder, which has extensive (but not complete) support for the MATLAB language. Inspirationally, this is how MATLAB provides data parallelism for one other built-in function *dspunfold*. Therefore, like *dspunfold* in this proposed framework, all functions must support Code Generation (CG) [88]. This framework only utilizes threading at the function level, since that is the minimum entry point for CG with MATLAB Coder. Building on top of MATLAB Coder has a clear advantage over custom compilers such as *MEGHA* and *MATISSE*, since it is developed by MathWorks and will grow with the ever changing MATLAB language.

From the generated code, we provide C++ libraries, additional code, and tooling to automatically generate wrappers around these MATLAB generated functions (MGF). The

[1] *par-for* relies on process forking for concurrency and cannot be used for general concurrency.

resulting code is tied together with a single source file describing the flow of data between functions, which we call the *flowgraph*. The C/C++ *flowgraph* is also automatically generated from a custom MATLAB class, which allows users to specify function level interactions, namely which functions pass to data to and receive data from which functions. This work utilizes the build systems already present within MATLAB Coder, providing even *Makefiles* when custom code needs to be edited or manipulated out of the scope of the auto generation process. From the perspective of a user, there is no necessary interaction required outside of MATLAB itself. Therefore, no knowledge of thread handling and efficient data transfer is required. However, function per thread separation is solely up to the user and will more than anything determine performance. Overall this framework provides function level parallelism, which is defined by the user. This function level point of parallelism is very different than the existing literature, which focuses on more instruction parallelism. We believe that this function level parallelism can be desirable since it limits latency and overhead compared with a more granular level of parallelism with a larger number of thread interactions. However, this can be application specific.

It is important to discuss other existing tools within MATLAB in order to differentiate their functionality from this work. Those tools are primarily *par-for* and *dspunfold*. *par-for* is an extremely useful simulation tool for data parallel tasks, and will scale to clusters of computers. *par-for* relies on process forking and requires strict data independence between operations. Utilizing these tools also disables some of the built-in thread spawning functions such as *fft* and *eig* in favor of their single threaded options. Data communication is also rather expensive when working directly in MATLAB but is reduced when CG is applied over these functions. On the other hand, *dspunfold* is purely thread based but is only for data parallel tasks like *par-for*. *dspunfold* can automatically take advantage of data independence for the operation to be threaded, but only applies to a single function. That function must also support CG. *dspunfold* does not create free-running threads, instead they are explicitly launched for each call to a function utilizing *dspunfold*, thus relying on OpenMP under the hood. Both *dspunfold* and *par-for* are very useful tools that can provide significant speedup in DP applications. In the proposed implementation, threads only join the main thread at end of execution. Since this work desires to limit latency, and in essence thread overheads,

```
%% Simple Flowgraph
g = GraphGenerator;

% Add Blocks To Graph
g.AddBlock('src',0,1)
g.AddBlock('awgn',1,2)
g.AddBlock('filter',1,1)
g.AddBlock('sink',2,0)

% Connect Blocks In Desired Order
g.Connect('src_0',0,'awgn_0',0)
g.Connect('awgn_0',0,'sink_0',0)
g.Connect('awgn_0',1,'filter_0',0)
g.Connect('filter_0',0,'sink_0',1)

g.DrawGraph(); % Visualize Graph
g.Build(); % Compile Graph
g.Run(); % Run Flowgraph
```

Figure 3.6: MATLAB flowgraph code for implementation of a simple noise filtering exampling. The MATLAB class *GraphGenerator* only contains five user visible methods, all of which are shown here.

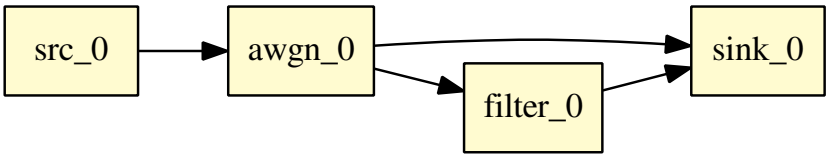


Figure 3.7: Flowgraph visualization shown when invoking the *DrawGraph* method as shown in Figure 3.6. The end result is provided in a MATLAB figure.

this is an alternative to thread re-spawning. [82] can be used to implement threading in a similar way. However, to limit dependencies the threading model used was developed in house from *boost* [89] primitives, which are already a MATLAB dependence. This provided us with full knowledge of thread interaction without relying on external tools.

3.3.2 Code Architecture

Similar to other DF architectures, MLDF is completely characterized by a top-level flowgraph. This flowgraph completely describes function boundaries in the form of blocks, data passing between blocks in the form of connections, as well as sources and sinks inherently. Blocks without connections are called floating blocks. Application examples will focus on signal processing tasks, but as stated previously, can be extended to other applications. Figure 3.7 provides a simple flowgraph that generates a signal, adds noise, filters the signal, and finally saves it to a file. run the eventual graph. The visual flowgraph was also automatically generated from line 16 of MATLAB code in Figure 3.6, implemented by Graphviz [90].

Each of the blocks in the graph runs in their own thread, making them free-running. Blocks themselves can be considered identical from the perspective of the graph except for three properties: (1) the processing algorithm the block is assigned, (2) the number of inputs ports, and (3) the number of output ports. These three parameters, which are provided by the user, allow for automatic parsing of the MGF's function prototypes for template generation. Ports are simply protected FIFO queues of object pointers that are shared between blocks. The queues are type agnostic, which is generally a requirement when working with code produced by MATLAB Coder, due to its heavy use of custom

type definitions. The queues only handle data pointers and never perform expensive data copies. Additionally, queue access was implemented in two flavors, a locking and a non-locking style in order to reduce contention. Due to the fact that the queue model here is a single consumer and single producer between all blocks, we were able in include an efficient non-locking scheme based on [91]. In essence, we are taking advantage of memory barriers and ring buffers. Not all processors support such instructions, therefore the locking implementation is more portable. During operation when data is added to a queue by a block, that block is responsible for signaling downstream blocks. This prevents wasteful polling, as well as queue backlogging.

The graph itself manages block startup, shutdown, and any runtime management that is required, such as benchmarking. However, to limit overhead or performance penalties the graph does not interact with the blocks during runtime by default. Additional flags need to be set with the *GraphGenerator* class in order to enable these runtime options. Other projects, such as *GNURadio*, utilize the graph to help manage flow between blocks during startup. The overall goal of this work is to minimize graph work, as well as latency between block information passing. Therefore, significant effort was put into optimizing inter-thread data passing. As a result, the blocks themselves can have only three states: waiting for new data, processing data, and passing data forward. State transitions only occur when all inputs or outputs are available, depending on the state. Again, when we utilize non-locking queues there should never be contention based waiting. A significant difference between this architecture and others, such as OpenMP *jobs*, is that this framework does not continually launch threads. This occurs once at the start of the flowgraph, and all threads are maintained for the life of the flowgraph.

In this framework, it is possible to implement scenarios where the producer block can run faster than the consumer block. If left unchecked, this would create an ever increasing input queue for the consumer block, consuming all of the system memory. To prevent this, back-pressure is provided at a configurable queue depth. Again, this is also provided through the *GraphGenerator* interface. When queues reach this level, the producer block will sleep periodically and poll the queue until space have freed up. This empirically has proven to be more performant than relying on conditional variable signaling between threads.

3.3.3 MATLAB Code Generation

Since 2011, MATLAB has been able to generate C/C++ code for a large portion of their functions [92], providing ISO compliant full C/C++ source code with no obfuscation through compiled libraries. However, CG is not simple to utilize, and has caveats of its own. This is especially true if the source MATLAB code is not strict about memory management. To help you handle that level of complexity, we have a recommended workflow that we have used to guide code to fit into the eventual MLDF architecture:

1. Confirm current MATLAB code has been tested and provides correct numerical results.

2. Split all code into separate functions that you wish to be individually threaded. Utilize assertions for all function inputs. Make sure again that this code provides desirable results.

3. Utilizing *coder.screener*, evaluate each function to check for CG compatibility. Make necessary corrections where necessary. Again check numerical outputs of functions for correctness.

4. Now that the functions are ready for CG, the flowgraph can be formed with the custom classes provided. These classes utilize the MATLAB build system and generate additional files used for wrapping MLGF's and connecting them together. This class also provides a visual representation of the flowgraph by utilizing *Graphviz*.

5. Generate all code and build executable using the simple interface from the custom provided classes. The end product is a single executable, which again is addressed through the *Run* method of the *GraphGenerator* class.

Making code viable for CG primarily involves making memory allocation strict and removal or replacement of certain functions. It is also important to note that each function will be re-entered, therefore persistence should be utilized for additional speedup. This is true of code executed in MATLAB as well. In summary, this process involves only writing MATLAB code and enforces the CG of functions.

3.4 Testing and Evaluation

As mentioned in Section 2.1, we primarily focus on signal processing and communication applications. Therefore, much of our testing focuses around these type of computations. This work originally provided a mechanism for providing concurrent operations for a communications system rather than a performance framework. However, when spreading work over threads we learned that significant performance enhancements could be gained, and attention was shifted to a more general framework. Therefore, to understand the speedup possible from this framework we provide testing from two perspectives. The first is from the top-level application of a flowgraph and the speedup provided over a non-concurrent implementation. The second is from under the hood of the framework itself, showing the consequences of managing concurrency. All tests outline in this sections are run on a Dual Xeon E5-2690v3, with 24 physical cores (48 Threads) at 2.60 GHz.

3.4.1 Framework Testing

We first examine the latency between blocks, specifically how long it takes for data to be added to the shared queue object between a pair of blocks and then popped off the same queue. In this scenario, we use a simple flowgraph with two blocks, a producer, and consumer sharing one connection. This is tested under two scenarios, the first where the source block produces data at a much slower rate than the downstream block can process it. Second, the source block processing time load is equivalent to the downstream block. Examining a case where the downstream block is much slower provides no perspectives on the framework, since the performance is solely gated on the processing time of the downstream block. Under the condition that the shared queue has already been filled. It is also a poor design where downstream blocks are slower than upstream blocks in a constant streaming data scenario. For an additional comparison, we provide results from our generic lock-based implementation to understand the benefits of the non-locking model.

Now given the implemented tests described above, these provide performance information relative to contention conditions between blocks. Since queues between blocks are shared resources, there is always a natural contention. Data manipulation between blocks

only involves pointer movement, therefore no data copies occur. As a result, the transfer latency between blocks is not dependent on actual size or type of the data transfered. With this knowledge, Figure 3.8 provides the results of 10^6 transfers for each of the test scenarios using the locking-queues. We utilize the *chrono* library within the C++ Standard Library to provide high resolution time information for these tests. For reference, we computed the average time to complete 100 , 10^4 and 2×10^4 double precisions additions to the same memory location on the same machine averaged over 10^6 iterations. The transfer times for the Fast Source (FS) are almost deterministic and better performing than the Slow Source (SS). At first, this is counter intuitive since there is less contention between blocks in the SS case. The reasoning behind this slowdown comes from the signaling delay that occurs when the downstream block sleeps if no data exists in the queue. Therefore, the sleeping block must wait for a signal from the conditional variables associated with the monitored mutexs. In the FS case, the block does not enter this sleep state as often, since the queue will have data ready to be process more often initially. Similarly, Figure 3.9 provides the results for the non-locking case, and performs two orders of magnitude faster on average. In this case, the FS has more variation as expected due to the additional contention, but the SS is almost deterministic. The cases considered here are the most optimal since each block only has one port. The performance here will degrade since exiting the waiting for data state will require data from multiple threads.

3.4.2 Flowgraph Testing

Now that we have an understanding of the low level thread interaction between concurrent functions, we can explore top-level speedup over single threaded non-concurrent versions. The goal is to understand system performance, in the form of data throughput, under significant computational tasks. To examine this we considered an FFT of length 2^{15} over complex double precision data as the basic unit of computation. Chosen to represent an intensive workload over a large amount of data.

To evaluate the performance of this function, testing measured the processing time of 10^5 randomly generated complex vectors of length 2^{15} (matching the FFT size). Across the tests we cascaded additional copies of the FFT baseline function to increase complexity of

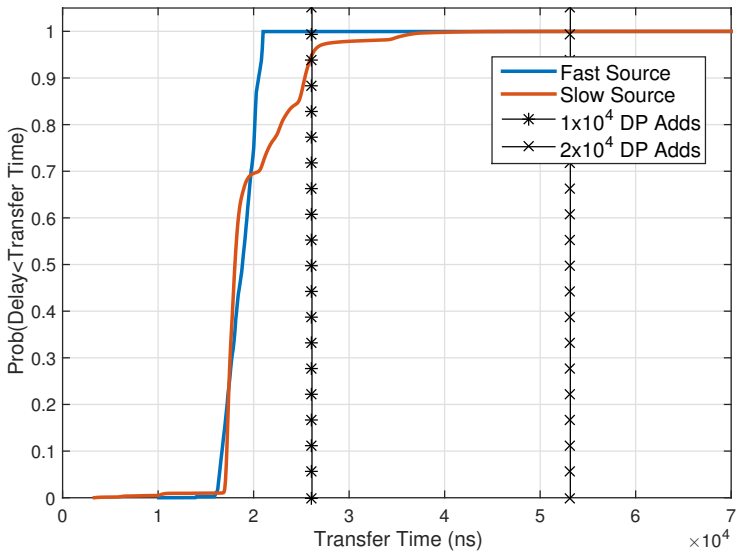


Figure 3.8: Transfer latency between blocks for 10^6 iterations for locking queue design. Double precision additions are provided as references to the cost of an individual transfer on average. These reference are provide as vertical lines since on a modern CPU the are roughly deterministic operations.

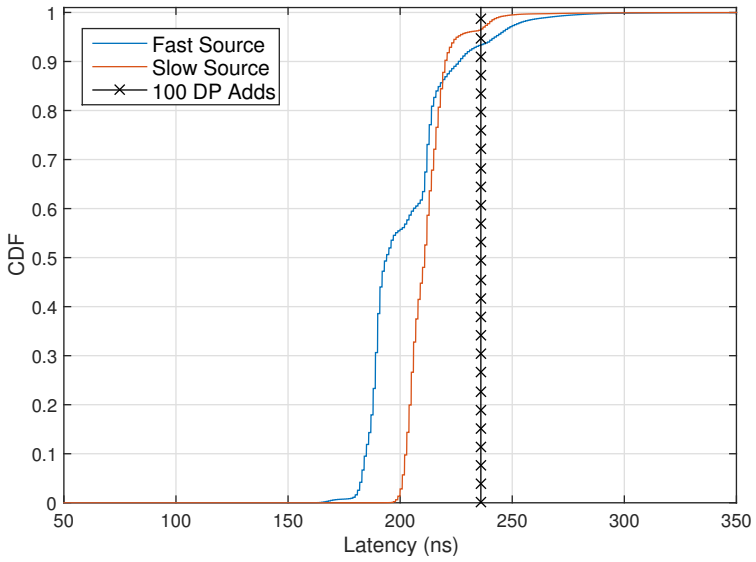


Figure 3.9: Transfer latency between blocks for 10^6 iterations for non-locking queue design. Double precision additions are provided as references to the cost of an individual transfer on average. These reference are provide as vertical lines since on a modern CPU the are roughly deterministic operations.

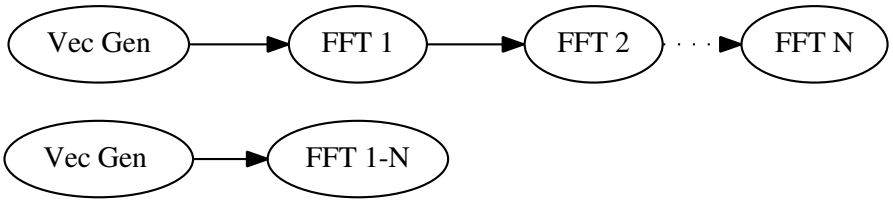


Figure 3.10: Throughput testing of cascaded FFT operations in the single threaded mode (bottom) and the MLDF implementation (top). These tests were run for two cascaded FFTs to twenty-two FFTs. The random vector generation (Vec Gen) was put into an independent thread to remove the additional computation required for those operations.

the problem, ranging from 2 to 22. To remove the effects of the random vector generation, both the single threaded (ST) and multi-threaded (MT) versions were written in the MLDF framework. In the ST case, one block contained the random vector generation code, and a second contained a loop over multiple FFTs. In the MT case, the additional FFTs are applied in additional blocks. This comparison between the ST and MT implementation is shown in Figure 3.10, with each block representing an individual thread. Figure 3.11 provides the results of this test averaged over several iterations for each number of FFTs cascaded. Showing almost no variance in the results in both ST and MT cases. As can be seen, the MT scales significantly better than the ST version. As the number of FFTs does increase, but so does the processing time and individual overhead between blocks. Since the test machine does have 24 physical cores, no bottlenecks are observed. As the flowgraphs become more complex and threads outnumber cores, performance evaluation will become difficult to predict. The ST case simply scales linearly with the increased workload, and will be independent of core count.

The speed advantage in Figure 3.11 is substantial, but it is important to discuss side effects of the speedup. TP will have a constant latency through the system equal to the required N computations plus communication overhead between threads. On the other hand, if DP was applied to this problem, each input vector would have a varying latency. This was theoretically presented in Section 3.2. The single thread case grows in latency since each subsequent vector must wait for each preceding vector to pass through all N FFTs before beginning computation.

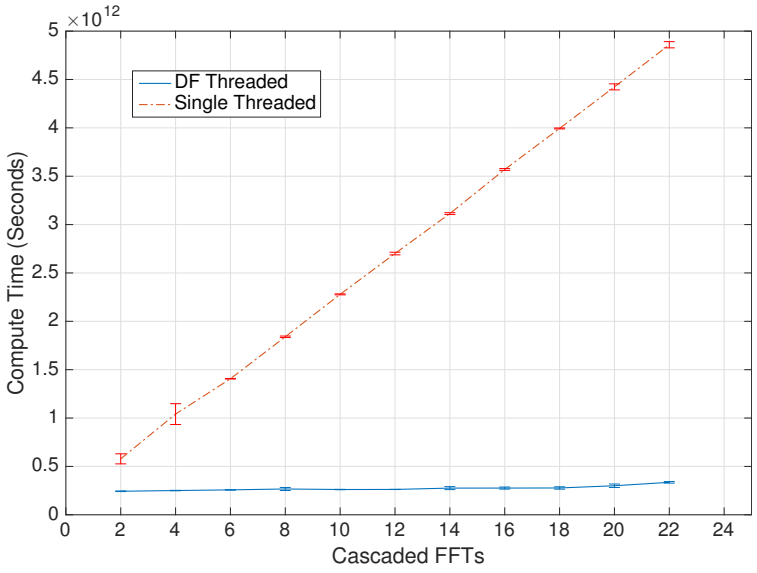


Figure 3.11: This test compares an increase in cascaded FFT operations in the MLDF case and the single threaded standard case. At each number of cascaded FFTs the system processed 10^5 random complex double precision vectors of length 2^{15}. Each test was run four times, with little variance between them.

3.5 Designing For DataFlow

We examined the low-level consequences of using the MLDF architecture and the top-level performance gains it can provide. However, when implementing algorithms within this framework special consideration and thought needs to be provided to function placement and division. Multiple DP and TP sections of algorithms can exist but their implementation is not always necessary, since DF does have overhead. In Figure 3.8 it was demonstrated under the best circumstances we can incur a penalty of $\sim 250ns$ per block transfer. Further synchronization of multiple port blocks can substantially increase this delay. Therefore, to minimize this effect computations of an individual block should take considerably longer than the data transfers. If the computation is not complex, in practice we will increase the amount of data passed per transfer to the block. Since this penalty is fixed, increasing the data to be processed will only increase the computation time of the block.

A second important aspect to consider is load distribution and gating that can occur. For cascade blocks specifically, when upstream blocks are significantly slower than downstream

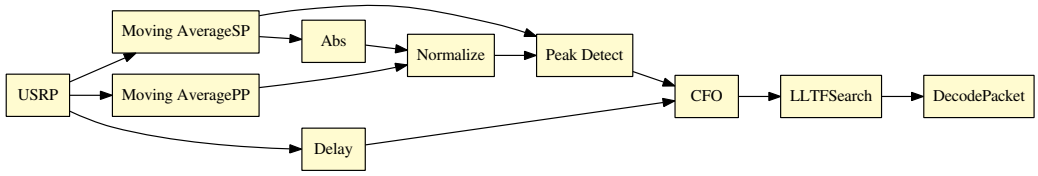


Figure 3.12: Complex flowgraph for WLAN 802.11a receiver, based upon the current design available in MATLAB. Significant resources where put into the packet detection phase of the flowgraph, which consists of six threads (Abs, Moving AverageSP, Moving AveragePP, Normalize, Peak Detect, Delay). This design was able to run in real-time and decode beacon frames from commerical routers. DecodePacket contains the majority of the code, but is called drastically less frequent than the upstream blocks.

blocks this creates block starvation conditions. This will limit core utilization. A similar effect can happen in reverse when downstream blocks are much slower, causing upstream blocks to wait on full queues. Implementations should strive to balance work across blocks as much as possible, but at the same time without thread flooding and choking the processor. We typically approach this by first setting a target throughput rate, and then distributing the algorithm among more blocks until we reach our target.

A final note on implementation details, Based on our second aspect discussed, in scenarios where block operation is conditional, such as those that require existence of specific data in a stream, resource trade-offs can be made. For example, if a piece of code is computationally intensive but called infrequently, it can be beneficial not to pipeline those operations. Reducing contention for available cores, or allowing cores to be more intelligently allocated, this may also reduce latency among those operations.

3.5.1 Limitations

We have examined the possible speed up advantage of this MLDF framework, but there are consequences to using this framework specifically through the use of MATLAB Coder for code translation. First of all, in many circumstances CG does provide speed up over interpreted MATLAB code since it is compiled. However, for highly optimized built-in functions of the MATLAB language there can be a degradation. This comes from the fact that CG produces generic C/C++ code that does not heavily utilize vectorization or

other SIMD type operations. Common advice in MATLAB is to vectorize code, which can provide increased speed up, since under the hood MATLAB will take advantage of processor specific extensions for vectorizations. However, this capability does not always translate into generated code with MATLAB Coder. Through the natural progression of this work we have examined a large amount of generated code, and have found that vectorization can be even tied to data size of each computation. The larger the computational operation in terms of data, the higher the probability of insertion of specific BLAS calls. This is also dependent on the operation itself and the profile selected by MATLAB Coder. Other projects can directly generate to SIMD and parallelized code from MATLAB code [51, 53], but as discussed previously, have very narrow language support. It may be possible to re-vectorize the produced code, but due to the level of obfuscation it will be very challenging.

Besides limited vectorization in generated code, plots and scopes are not available in any form of CG. Leaving only file outputs and textual feedback. It is important to note that plots and scopes in interpreted mode do cause additional latency in the system and hinder performance in the general sense. We are actively exploring other solutions to this problem, since scope can be invaluable for debugging. In this spirit, we are also considering hooks back into MATLAB at runtime through the MEX API [93].

3.5.2 Reaching Real-Time

With the limitations understood and known possible speed increases available, we implemented a real-time 802.11a receiver in the MLDF framework. This work was based upon a serial implementation which is now available as part of the WLAN (System) Toolbox [94]. To reach the real-time objective, the flowgraph needed to handle a data throughput of 20×10^6 double precision floating point numbers per second at the most intensive sections. Data was provided into the flowgraph by a USRP-B210 [95], which displays overflow warnings when the system is unable to keep up with the data stream. To reach this performance goal, drastic modifications were made from the original serial code, primarily in the packet detection portions, which were the most computationally intensive. However, only by expanding into the MLDF framework did many of these bottlenecks become apparent. Many were rather hidden when looking at MATLAB profiles alone. Figure 3.12 outlines the even-

tual flowgraph of the system. Both DP and TP type flows were applied, and the overall structure was influenced by [96]. The workflow to reach real-time relied on the strategy discussed in Section 3.5, where hotspots in the flowgraph were spread across more and more threads. An interesting aspect of this design is that the packet decode operations, which can be considered extremely computationally intensive due to their reliance on a Turbo Decoder [97], are contained in a single thread. This was possible since the rate at which we observed packets was orders of magnitude below the rate at which the packet detection algorithms worked at. With this design, we were able to recover packets from commercial access points without overflow for sustained periods of time. In Figure 3.13 a breakdown of the relative blocks speeds are provided. The blocks with green bars are required to run continuously, however block with green bars only conditionally run. The MATLAB only demo without the MLDF framework provided by MathWorks is shown in red. All continuously running block are well above the 20×10^6 throughput requirement.

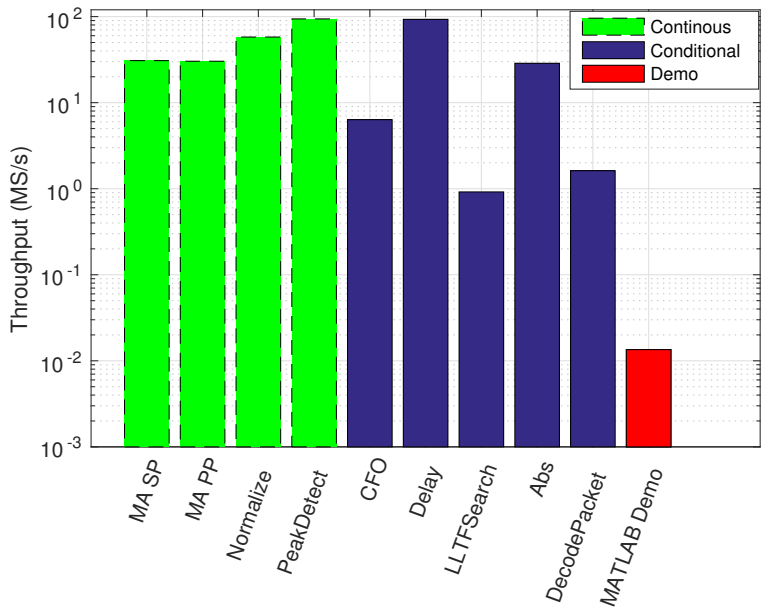


Figure 3.13: Relative throughput of computation blocks in WLAN receiver design. In green are blocks requiring continuous operations, in blue are blocks that have dependent operation, and in red is the original MATLAB demo implementation.

3.6 Discussion and Retrospective

Through this work we have set out to enable high throughout processing for applications with data dependency. These are the main applications where DF truly shines. However, these type of designs are not without limitations as we have discussed. At the macro-scale we can get significant gains if we design the pipelines of the overall system correctly, but there will always be room for improve because of the limitations introduced by MATLAB Coder. The most concerning argument against using MATLAB Coder is the level of optimality of code it can produce, which is quite poor in many scenarios. Nonetheless, MATLAB Coder is a closed source product we have no knowledge on the code translation process. Alternatively, there are still methods which have not been explored here to side step this problem slightly. This method includes use of code replacement libraries within MATLAB Coder itself, which it can use for direct mapping between MATLAB and C/C++. Currently these libraries are targeted at fixed point DSPs with small instructions sets. However, we can utilize these replacement libraries with SIMD libraries for common functions or series of lines of code. Unfortunately this can be a rather tedious process.

Code optimality will always be a concern of developers, but the speed provided for converting algorithms directly to C/C++ into this framework may be more beneficial than rewriting them by hand in C/C++. This is a trade-off implementors will face and is a common problem with automatic code generation tools. Alternatively, it can be useful to partition work so that specific intensive functions are written by hand and other are generate. MATLAB already supplies small capabilities to do this through *coder.ceval* calls which work happily with the proposed MLDF here. We have actually utilizes similar techniques in the WLAN receiver for small functions that have faster built-in C/C++ versions rather than the MATLAB produced functions.

3.7 Chapter Summary

In this chapter, a DF framework is presented for use with the MATLAB DSL. The performance benefits of this MLDF framework were evaluated first from both a high level application perspective, where we considered series throughput compared to single threaded

MATLAB with an intensive workload. Second we studied in-depth benchmarking of the framework itself, focusing on communications between blocks for two different implements for a set of scenarios. Providing solid cost metrics when utilizing data sharing between threads. Overall, the framework can provide significant signal processing algorithm speedup by pipelining and providing general concurrency to the system level design. However, due to limitations in MATLAB and associated components, vectorization performance can be reduced, and only a subset of the MATLAB language functionality can be used with this framework. Nonetheless, compared with existing MATLAB language compilers this MLDF framework has extensive language support, especially when considering user defined functions and classes. This is the result of building on-top of the MATLAB Coder product instead of using these external project for language translation.

Throughout this chapter we have also provided discussion on the trade-offs of using this architecture. In many cases MLDF will not apply, but making this tool available to implementors can provide avenues for code speedup. By both leveraging the deep signal processing libraries of MATLAB and the speed advantages of threading the MLDF, we can significantly decrease the implementation time of real-world designs and testing of those design.

Chapter 4

Stochastic Networks: Modeling Interference

In this chapter, we examine the effect of interference at a macro-scale in a wireless network. Specifically, we utilize SG techniques to model HetNets, combined with directional antenna analysis. This primarily focuses on cellular type networks, but can be applied to sensor networks, Wi-Fi networks, and others. These models help provide a reasonable study of a feature rich communication environment, in a continued effort of the general literature to accurately model future networks.

The work here provides a purely analytical approach to interference characterization, and this perspective must be understood before examining this analysis. SG techniques examine the network from single instances in time and utilize probability methods to average over randomness. In our case randomness only comes from two aspects of the model: (1) locations of BS in the network, and (2) small scale fading. Therefore, considering time series effects of error in traditional approaches is not applicable to this work. However, we are only interested in the average performance of the network and these sources of randomness become understandable in the results.

To provide connections to traditional network analysis and result checking, network simulation are contrasted against the numerical evaluations of the analytical results. Given this check, it is important to understand that these results are only as accurate as the models

we utilize and that we only provide aggregate results. Thus, many network configurations will severely over and under perform these results. However, consequences for specific design decisions can be inferred on network performance from the macro perspective.

Compared to current state-of-the-art, the main contributions of our work are:

- Introduction of a new mixed PP model for a closed subscriber K-Tier HetNet utilizing both PPP and GPP, for more effective combined model of Femtocell Base Station (FBS) and Macrocell Base Stations (MBS).
- An analytic derivation of PPP and GPP HetNet models with transmit directional antennas. This extends [68], [69] and generalizes [62] by removing the the fixed distance constraint between the typical receiver and its paired transmitter. All verified through simulations.
- An evaluation of the effects of directional antenna gain patterns, including tier transmit power and tier density heterogeneity.
- Mean rate analysis of gain pattern selection for interference reduction with an increasingly dense interference tier.

4.1 Point Process Model

In this subsection, we discuss two PP model which will govern BS locations in our networks. The PPP is the simplest PP and is closely related to the binomial point process where fixed number of node are uniformly placed in a certain area. However, in a PPP the number of points in a given area is Poisson distributed. This is known as intensity of density of the PP. PPP enjoy complete spatial randomness or independence property making them easy to analyze. The second PP we utilize is the scaled GPP, which is an example of a determinantal point process (DPP). For further details, refer to [98, 99]. DPP have dependence among points, which makes them difficult to model or even simulate. However, they are completely characterize by their product densities.

A point process, Φ is termed *determinantal* with kernel: $K : S \times S \to \mathbb{C}$ if its n^{th} order

product density is of the form:

$$\rho^{(n)}(x_1, x_2, ..., x_n) = \det(K(x_i, x_j))_{1 \leq i,j \leq n}, \quad (x_1, x_2, ..., x_n) \in S^n \tag{4.1}$$

for a Borel set, $S \subseteq \mathbb{R}^d$. The square kernel matrix K, is any function such that $K : S \times S \to \mathbb{C}$. $K(x_i, x_j)$ in (4.1) denotes the (i, j) element of the matrix K, and det denotes the determinant operation. $\mathbb{C}$ denotes the complex plane.

Moreover, for any Borel function $h : S^n \to \mathbb{R}_+$:

$$\begin{aligned} &E\left[\sum_{x_k \in \Phi}^{\neq} h(x_1, x_2, \ldots, d_n)\right] \\ &= \int_S^{(n)} \int_S \rho^{(n)}(x_1, ..., x_n) h(x_1, ..., x_n) dx_1 \ldots dx_n, \end{aligned} \tag{4.2}$$

where $\neq$ defines the sum as pairwise unique over the n tuples of Φ. In the specific case of a scaled GPP, the kernel matrix is of the form [69]:

$$K(w, z) = \frac{a}{\pi} e^{az\bar{w}} e^{-a(|z|^2+|w|^2)/2}, \quad (w, z) \in \mathbb{C}, \tag{4.3}$$

with respect to the Lebesgue measure on $\mathbb{C}$. For this work we are specifically interested in distance between points and the origin. Therefore, we will utilize alternative distributions to go around this tractability issue while still maintaining the GPP properties.

The main contrast between these distributions is their dependence among their governing points. In Figure 4.1 we provide an example of this repulsiveness property that DPP maintain over the completely random PPP. DPP realizations can be generated with a new software package called *spatstat* [100]. However, single realizations can take many minutes to generate for small sample sizes, which was unsuitable for our purposes here.

4.2 Network Model

We focus on the perceived interference of a typical receiver placed in a tiered network of transmitters arranged according to a scaled GPP, and a PPP. We are utilizing a GPP, since they have shown to provide accurate models of MBS tiers due to their repulsive properties.

Table 4.1: Table of variables for stochastic models of randomly deployed BS.

Variable	**Definition**
β	Pathloss exponent
c	Connection tier index
δ	Density ration between tiers
g_1	Maximum gain of antenna main lobe
g_2	Secondary gain of antenna pattern
G_T	Antenna gain pattern
h	Channel power between BS and receiver
K	Number of tiers
$l(\|x\|)$	Pathloss of BS from x
λ	Density of tier
μ	Mean of channel power
p_s	Success probability
P_t	Transmit power of tier
Φ	Point process
$\mathcal{R}_c$	Mean rate of connection tier
θ_1	Main beamwidth in radians
x_i	BS position

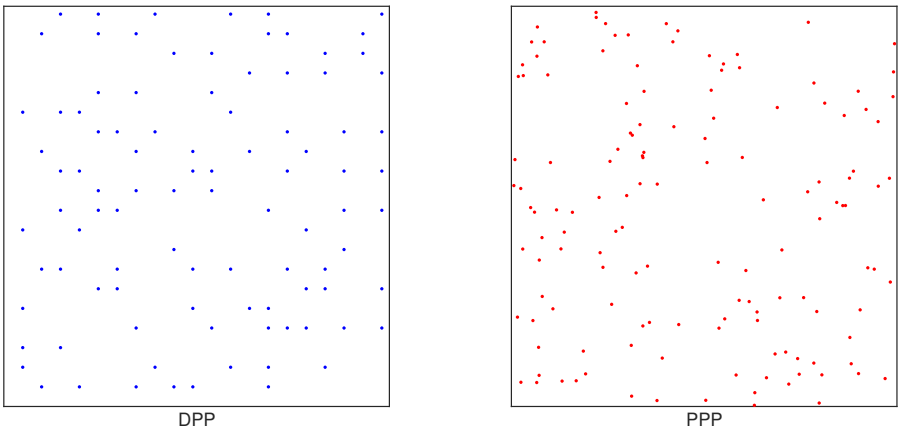


Figure 4.1: Realization comparison of DPP vs. PPP on a 2-dimensional plane. The repulsiveness of the DPP can easily be observed over the complete randomness of the PPP.

However, we employ the PPP for FBS tiers due to their more spatial randomness. Therefore, in combination both allow for study of heterogenous type networks. We provide a list of definitions for these variables in Table 4.1 for convenience.

Now, the PP governing an individual tier is denoted by Φ_k $k \in \{1, .., K\}$ on $\mathbb{R}^2$, where $x_i \in \Phi_k$ represents the node locations in the k^{th} tier. Φ_k can represent either a scaled GPP or PPP. Having multiple GPP tiers can be useful for modeling spatially co-deployed MBS, who even may compete for spectrum in future overlayed deployments in vacant TV-Whitespace [101]. The transmit power for the k^{th} tier is denoted by $P_{t,k}$ and is assumed identical and constant over that tier. Tier powers can be simplified in reference to the connection tier's power $P_{t,c}$ as:

$$P_k = \frac{P_{t,k}}{P_{t,c}}. \tag{4.4}$$

Each user connects to the nearest transmitter of the tier it is associated with $X_{i,k}$, providing the best SIR for a given location. Since Φ_k is motion-invariant and stationary with intensity λ_k, by Campbell's theorem [102], the average interference is same for point processes of the same intensity. Hence, we can generalize the average interference across the network to a user positioned at the origin o.

All tiers are co-located in frequency, providing all available bandwidth to all tiers. This has been shown in the open subscriber case to provide better spatial capacity [103]. This

also does not force operators to reduce MBS resources for an unknown number of FBS, or purchase expensive bandwidth for a currently under-deployed FBS tier. However, an open subscriber network is generally not realistic which we will discuss further in our models. Nevertheless, when considering per tier load analysis can become almost intractable in open GPP networks. As a result, we avoid that type of analysis here but has been considered in [64] for PPP networks.

We assume the fading channels between the typical receiver and the transmitters are *i.i.d.* Rayleigh distributed. Consequently, each channel's power value, h_i, is exponentially distributed. Furthermore, we assume that it has a mean of $1/\mu$. The path-loss attenuation is modeled by the power law as $l(|x|) = r^{-2\beta}$. Moreover, we assume that each transmitter is in the GPP tier equipped with a DA, whose bore-sight is aligned with the paired receiver of the transmitter. We also assume the transmitters in a tier governed by a PPP and each receiver is equipped with an omni-directional antenna. For simplicity in the analysis, environmental noise is ignored. This is a valid assumption since traditional cellular networks (LTE, GSM, *etc...*) are interference limited. Antenna patterns are denoted by $G_{T,k}$ for the given tier.

When considering *mmWave* networks they become no longer limited by interference, therefore many of the assumption made here would require reconsideration. However, due to current models propagation these networks would be limited due to absorption which maintains our assumption on a compact and finite set of interferers. It is our belief that *mmWave* communications will not be used at macro-scale type deployments, instead favoring FBS as primary implementations. This would drive analysis back into the PPP only models, which are much more accessible to the academic community.

Based on the assumptions stated above, the interference term, which is the sum of signal powers from all transmitters excluding the desired transmitter, is given by:

$$I_{o,k}(i) = \sum_{j \in \mathbb{N} \setminus \{i\}} P_{t,k} G_{T,k} h_j l(|X_j|), \tag{4.5}$$

the signal-to-interference ratio (SIR) experienced by the typical receiver connected to the

c^{th} tier where g_1 is the maximum gain of $G_{T,c}$ is given by:

$$SIR_{o,i} = \frac{g_1 \, P_{t,c} \, h_{B_o} \, l(|X_{B_o}|)}{\sum_{k=1}^{K} I_{o,k}(B_o)}. \tag{4.6}$$

where B_o denotes the location of the typical receiver's paired transmitter.

4.2.1 Closed Subscriber Connectivity

In this work, we consider a Closed Subscriber Network (CSN) model. This model assumes that a given receiver is only allowed to connect to a given tier or subset of tiers. The CSN is more realistic than an Open Subscriber Network (OSN) model since FBS owners are unlikely to share their personal Internet with the general population, providing possible security threats and required bandwidth sharing. Therefore, the connection PP is denoted by Φ_c, and this notational pattern is reflected in Section 4.2. The perspective of the MBS users will be the primary focus of this work.

By reducing the perceived interference of users in the MBS tier, we will by consequence reduce the interference observed in the other tiers since all tiers are co-deployed in the same channel. However, we need to be careful when considering users that can connect with the FBS tier since they will almost certainly have access to the MBS tier in a realistic scenario. Therefore, in this Mixed PP model users with freedom to connect to both tiers will naturally have better coverage than MBS only users. However, this is difficult to calculate since we need to consider the joint conditional probability of connecting to either tier. This is beyond the scope of this work and models only a small percentage of users, since users initially will most likely only be able to connect to their own FBS, not others. Modeling such a scenario as a semi-closed network is even more channeling to consider.

4.2.2 Gain Patterns

In this work, we study a parameterized keyhole antenna gain pattern, chosen for its simplicity and tractability, which is necessary specifically for evaluation of GPP tiers. The pattern is symmetric relative to the bore-sight of the antenna, and with associated gain

pattern:

$$G_{T,k}(\theta) = \begin{cases} g_1 = \frac{\pi-(\pi-\theta_1)g_2}{\theta_1} & \text{if } 0 \leq \theta \leq \theta_1 \\ g_2 & \text{if } \theta_1 \leq \theta \leq \pi \end{cases} \tag{4.7}$$

$$\tag{4.8}$$

where, g_1 is the main beam maximum gain, g_2 is the side-lobe gain, and θ_1 is the main beam-width. g_1 and g_2 are constrained such that the total radiated power (TRP) by the antenna is unity, *i.e.*, $\pi^{-1}\int_0^{\pi} G(\theta)\, d\theta = 1$. The orientation angle of the transmitters, ψ is assumed to be distributed in $\mathrm{U}[-\pi, \pi]$ with $\theta := |\psi| \sim \mathrm{U}[0, \pi]$. Due to the random orientations of the interferers with respect to the location of the typical receiver, each interfering signal is affected by a random gain. Consequently, a probability density function (p.d.f.) for the gain is needed for evaluating the success probability in the following section. Based on (4.7), the p.d.f. of the gain pattern is simply:

$$f_{G_{t,k}}(g) = p\delta(g - g_1) + \bar{p}\delta(g - g_2), \tag{4.9}$$

where $p = \theta_1/(\pi)$ and $\bar{p} = 1 - p$ and $0 \leq \theta_1 \leq \pi$. It is obvious that, $g_2 \leq g \leq g_1$. If we consider a more complicated antenna pattern, adopted from the 3GPP technical report [104], this can provide more realistic but complicated result. Again, this pattern is symmetric relative to the bore-sight of the antenna, and with associated gain pattern:

$$G_{3GPP}(\theta) = \begin{cases} g_1\, 10^{-c\,\theta^2} & \text{if } 0 \leq \theta \leq \theta_1 \\ g_2 & \text{if } \theta_1 \leq \theta \leq \pi \end{cases} \tag{4.10}$$

Additionally, for the 3GPP pattern, $c = 0.3/\theta_{3dB}^2$, where, θ_{3dB} is the $3db$ beam-width. Figure 4.3 provides a visualization of the antenna gain as a function of the orientation angle θ for the keyhole antenna patterns. Using a similar technique of taking the CDF from (4.10), we can write the p.d.f.,

$$f_{G_{3GPP}}(g) = \frac{1}{2\pi\sqrt{c\,\log_e(10)}} \frac{1}{g\sqrt{\log_e\left(\frac{g_1}{g}\right)}} + \bar{p}\delta(g - g_2). \tag{4.11}$$

The first term of the above sum is continuous and captures the main lobe, whereas, the second term is similar to that of (4.9) and represents probability of being aligned in the

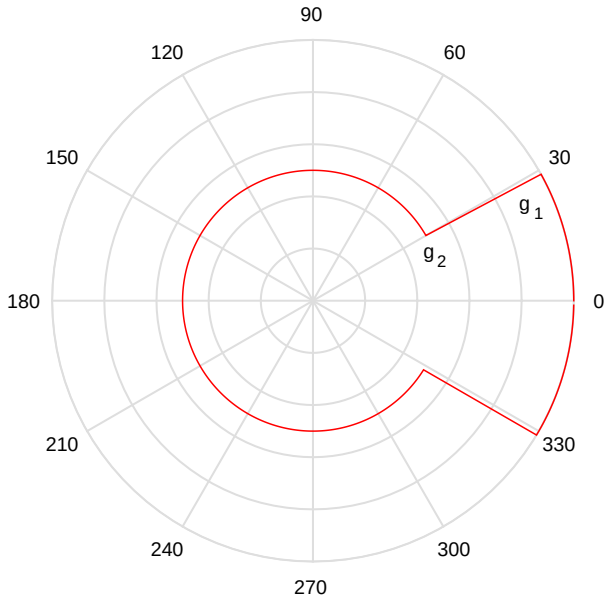


Figure 4.2: Example antenna gain patterns over the orientation angle θ. This pattern must always maintain the TRP and therefore selections of g_2 and *theta* lead to g_1 directly.

direction of the side-lobe. It is obvious that in both cases, $g_2 \leq g \leq g_1$. The derivation for the p.d.f. shown in (4.11) is straightforward and reached through the CDF method. However, since (4.11) is continuous in the main-lobe it can be difficult to simulate, effectively adding another integral to the analysis. Therefore, this pattern can only be applied to the PPP cases, when considering mean rate analysis, since in the GPP case simulation can take around a week to complete. For reference we provide a gain pattern comparison between the Keyhole and 3GPP pattern to identify the steepness of the main lobe roll-offs. Note that to maintain the TRP for the 3GPP pattern from a selection of g_2 and θ_1, the remaining gain g_1 cannot easily be found in a straightforward manner. Solving for it iteratively by checking the resulting sum gain in a brute-force manner proved sufficient but not elegant.

4.3 Success Probability

In this section, we determine the success probability for the system model described in the previous section. Success probability, a standard metric for performance analysis

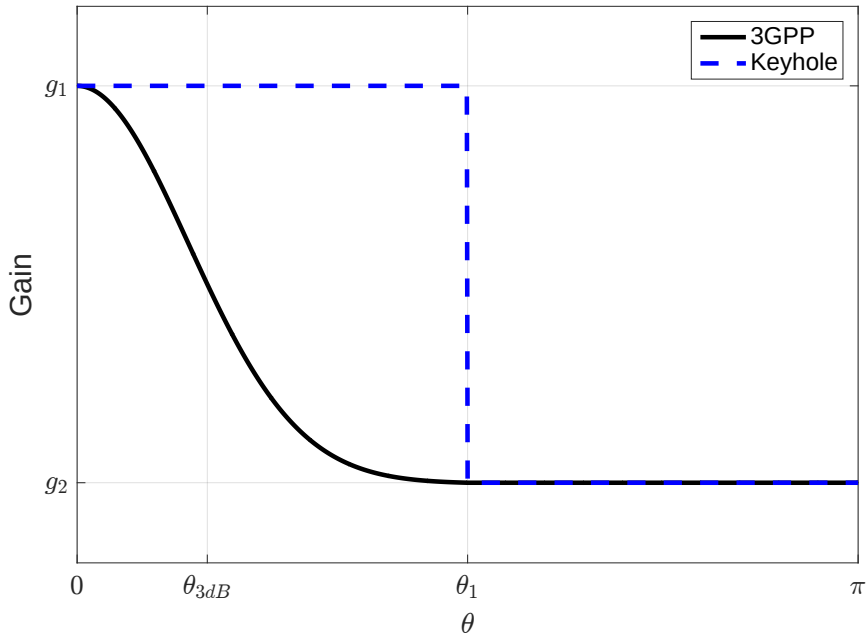


Figure 4.3: Example antenna gain patterns over the orientation angle θ. At θ_{3dB} the 3GPP pattern is at $3dB$ of g_1 and complete gain transition to g_2 at θ_1. Note that the figure is not to scale since there is no combination of g_1 and g_2 that simultaneously achieves unit TRP for both patterns. Note that g_1 and g_2 are not to scale since there are no configuration of values, while maintaining the overall gain constraint, for g_1 and g_2 to be identical between both patterns.

of wireless node configurations [102], is the probability that the SIR of a typical user achieves a predefined threshold, i.e., $p_s(\gamma) = P(SIR_o > \gamma)$, where γ is the SIR threshold. Before proceeding to the success probability evaluation for specific scenarios, we first provide its general representation when the communication link between the transmitters and the typical receiver experiences a random gain of an arbitrary p.d.f.

Lemma 1. *The success probability of a typical receiver for the network model described above, when the transmitters are configured according to K general stationary point processes on $\mathbb{R}^2$ is given by:*

$$p_s(\gamma) = \mathbb{E}_{\Phi, G_T}\left[\prod_{k=1}^{K}\left[\prod_{j\in\mathbb{N}\setminus\{i\}}\left(1+\gamma\frac{P_k G_{T,k}}{g_1}\left|\frac{X_{B_0}}{X_j}\right|^{2\beta}\right)^{-1}\right]\right]. \tag{4.12}$$

Proof. See Appendix 6.2. □

Lemma 1 specializes the result provided in [69] for omni-directional transmitters when $K = 1$. Reference [68] provides a similar result for the more general β-GPP configured nodes with omni-directional antennas.

We can simplify (4.12) to a form that we be notationally convenient and compact for the remaining derivations in the chapter. Therefore, we can split (4.12) as:

$$p_s(\gamma) = \int_0^\infty M(r_c, \gamma) \prod_{k=1}^{K} S_k(r_c, \gamma)\, dr_c, \tag{4.13}$$

where M governs the distance to the closest transmitter of the c^{th} tier and is a result of evaluating the expectation w.r.t. the PP. On the other hand, S_k is the conditional SIR of the individual tiers represented by the inner product term in (4.12).

Now we will consider M and S_k for the scaled GPP and PPP with associated DA.

Lemma 2. *The success probability of a typical receiver for the K-Tier network model described above for the form (4.13), when the transmitters are configured according to a PPP on $\mathbb{R}^2$ with DA is:*

$$M_{PPP}(r_c) = 2\pi\lambda_c r_c e^{-\lambda_c \pi r_c^2}$$

$$S_{k,PPP}(\gamma, r_c) = e^{-\frac{\lambda_k \pi r_c^2}{\lambda_c}\rho(\gamma)}.$$

where:

$$\rho(\gamma) = \gamma^{1/\beta} \int_{\in g} \int_{\gamma^{-1/\beta} \times \mathbb{1}_{\{k=c\}}}^{\infty} f_{G_{k,t}}(g) \left(1 + \frac{g_1}{gP_k} u^\beta\right)^{-1} du\, dg$$

Proof. See Appendix 6.2. □

In the c^{th} tier interferers can only exist beyond r_c, the distance of the closest transmitter. However, the other $K - 1$ tiers are not bound by this limit, and the greater their intensity the higher probability that their transmitters will be positioned at a distance closer than r_c.

For modeling more regularly placed BS such as MBS, we will use the GPP. Again, as in the PPP case, we assume that the tiers are completely independent, with heterogeneous intensities λ_k and transmit powers P_k, and with directional antenna patterns $G_{T,k}$. Furthermore, the typical receiver is assumed to connect to tier c only.

Theorem 1. *The success probability of a typical receiver for the K-Tier network model described above, when the transmitters of each tier are configured according to a GPP on* $\mathbb{R}^2$ *is:*

$$M_{GPP}(\gamma, r_c) = \sum_{i=0}^{\infty} r_c^i e^{-r_c} \Big(\int_{r_c/\lambda_c}^{\infty} \frac{s_c^i e^{-s_c}}{\zeta_k(v,\gamma)} ds_c \Big)^{-1} \tag{4.14}$$

$$S_{k,GPP}(\gamma, r_c) = \prod_{j=0}^{\infty} \frac{1}{j!} \int_{\frac{r_c}{\lambda_k \delta_k} \times \mathbb{1}_{\{k=c\}}}^{\infty} \frac{s_k^j e^{-s_k}}{\zeta_k(v,\gamma)} ds_k \tag{4.15}$$

where:

$$\zeta_k(v, \gamma, r_c) = \int_{\in g} f_{G_{T,k}}(g) \left(1 + P_k \gamma \Big(\frac{r_c}{s_k \delta_k} \Big)^{\beta} \frac{g}{g_1} \right) dg \tag{4.16}$$

$$\delta_k = \frac{v_c}{v_k} = \frac{\lambda_c}{\lambda_k}.$$

Proof. See Appendix 6.2. □

Corollary 1. *The success probability of a typical receiver for the network model described above, with the transmitters configured according to a scaled GPP on* $\mathbb{R}^2$ *and equipped with DA of identical radiation pattern for Equation* (4.16)*, is:*

$$\zeta_k(\gamma, r_c) = p \left(1 + P_k \gamma \Big(\frac{r_c}{s_k \delta_k} \Big)^{\beta} \right) + \bar{p} \left(1 + P_k \gamma \Big(\frac{r_c}{s_k \delta_k} \Big)^{\beta} \frac{g_2}{g_1} \right). \tag{4.17}$$

Proof. By applying the gain distributions shown in (4.9) to (4.16), we get (4.17). □

Remark. *It can be noted that in Lemma 2 and Theorem 1, if we set* $G_T = g_1 = 1$ *and thus, ignore the outer expectation, the resulting probability of success expressions represent the corresponding omni-directional cases.*

Lemma 3. *The ergodic or mean rate in Nats/Hertz for a typical receiver in a K-Tier network connected to the* c^{th} *tier is:*

$$\mathcal{R}_c = \int_0^{\infty} \int_0^{\infty} M(r, e^{\gamma} - 1) \prod_{k=1}^{K} S_k(r, e^{\gamma} - 1) d\gamma dr. \tag{4.18}$$

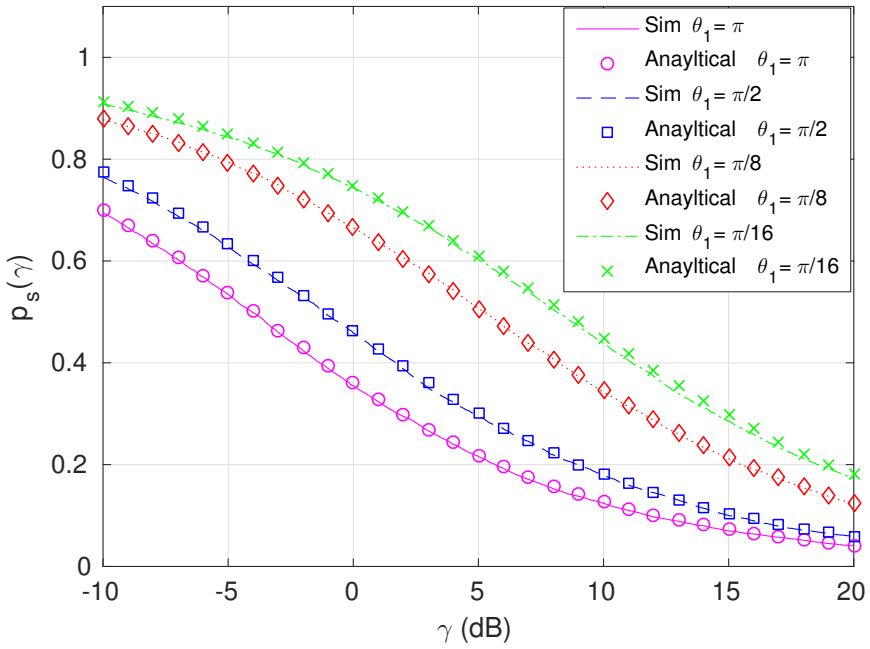


Figure 4.4: Success probability for different mainlobe beam-widths for antennas of the MBS connected tier. There is only one secondary PPP tier with omni-directional antennas. Here simulation provides matching results against their numerically evaluated analytical equations.

Proof. Reached by applying Shannon's equation we have:

$$\mathcal{R}_c = \int_0^\infty \sum_{i=1}^{\infty} \mathbb{P}(ln(1 + SIR_o) > \gamma, B_o = i)\, d\gamma. \tag{4.19}$$

Then continue as in Lemma 1. □

This is similar to References [69] and [103] evaluations. Note that since we treat interference as noise in this equation, the actual channel capacity cannot be reached for Shannon's bound.

4.4 Mixed PPP/GPP Tiered Networks

In this section we first provide a motivational CSN example and then analyze the interference experienced by a receiver connecting to a MBS tier with a secondary FBS tier acting solely as interference. This is further extended to capture effects of directional antenna in these scenarios. However, since any permutation of (4.13) for PPP and GPP tiers is not in

closed form, we evaluated them numerically across γ for all remaining results. First, main beam widths for a range of θ_1 values in Figure 4.4 were explored. In these simulations we set the following parameters: $\delta_k = 1$, $P_k = 1$, $\beta = 2$, $\mu = 1$, and $g_2 = 0.01$. The connection tier is GPP, with a single PPP interference tier ($K = 2$). All success probability curves are verified through simulations, which model 10^3 spatial realizations of the networks with independent channel realizations from each BS to the receiver at the origin.

As shown in Figure 4.4, as the main beam width decreases, probability of success increases. This is as expected, similar to results by Wildman *et al.* [62]. However, in this case there is additional interference that causes a significant reduction in coverage. Alternatively, GPP does provide a slight advantage over PPP offsetting this reduction. The additional gap between the PPP and GPP models is termed *structured network gain* [105], due to the regularity of the GPP over the PPP.

Next we examine tier power and density heterogeneity by adopting parameters from the previous scenarios except: (1) all tiers utilize omni-directional antennas, and (2) $\delta_k \leq 1$ and $P_k \leq 1$. Since only relative differences are needed in Lemma 2 and Theorem 1, only ratios are discussed here. Linearly increasing in both power (in favor of the MBS tier) and density (in favor of the FBS tier) in Figure 4.5, provides evidence that the growing FBS tier's interference cannot be overcome by matched power increases. In the case of a CSN, MBS users can observe substantial interference from other tiers that they are unable to connect. Such conditions exists in LTE-A when MBS-only users are physically close to a FBS and active outside the Almost Blank Subframe (ABS) regions when evolved intercell interference coordination is enabled [106]. This condition is overlooked in LTE-A including co-tier interference between FBS, which are both addressed in the presented framework here.

To reduce this effect of reduced coverage, we explore the use of DA and varying their associated beamwidths and gains. We know from Figure 4.4 that we can increase the coverage with this method. One of our goals in this chapter is to understand at what network density of FBS can the MBS maintain their mean rate by modifying their beam patterns. Maintaining a specific mean rate can be important to network designers for quality of service requirements. For this evaluation we again adopt the original scenario,

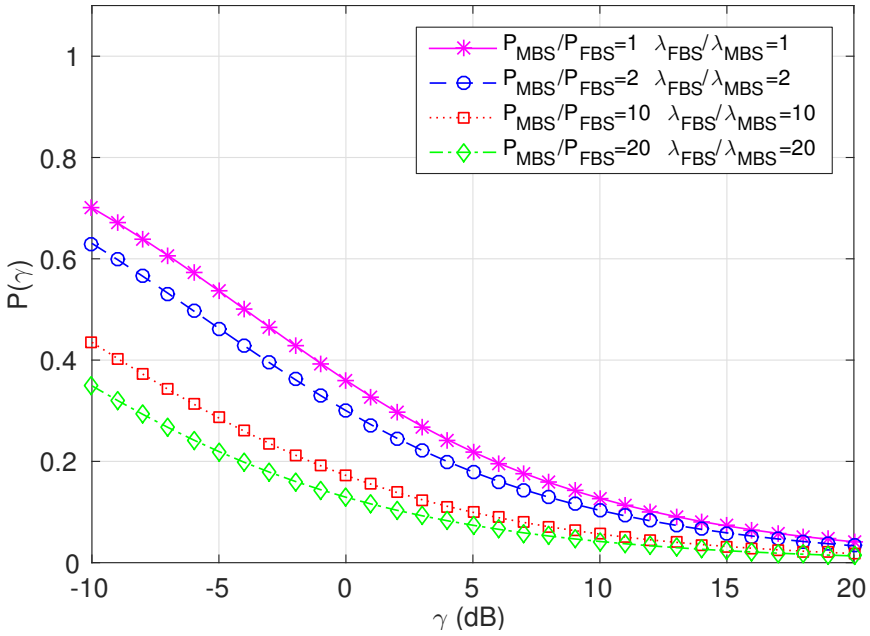


Figure 4.5: Effect of transmit power and tier density heterogeneity on coverage. Relative power is increased linearly in favor of the MBS tier. Simultaneously, network density is increase linearly in favor of the FBS tier.

with one MBS/GPP tier with DA and one FBS/PPP tier. Except the transmit powers for the associated are $1\,Watt$ and $100\,Watts$ for the FBS and MBS tiers, which is appropriate since MBSs are much more powerful than FBSs. Now since FBSs density will increase over time as deployments become built out, we also explore an increasing λ_{FBS} while maintaining a fixed λ_{MBS}.

As we can see in Figure 4.6, there are diminishing returns as the FBS networks get more dense, and the practical size beam-widths are not meeting the rate requirements. Alternatively, the MBS tier could increase the power to increase the rate. However, increasing power also has diminishing returns since it not proportional to the intensity as explored in Figure 4.5. Therefore, as the FBS density increases, DA can provide interference elimination from other tiers, but at certain densities other techniques must be considered to maintain the rate.

To provide alternative avenues for increased coverage there are several approaches that include scheduling of receivers which we can model as network thinning. Second, frequency

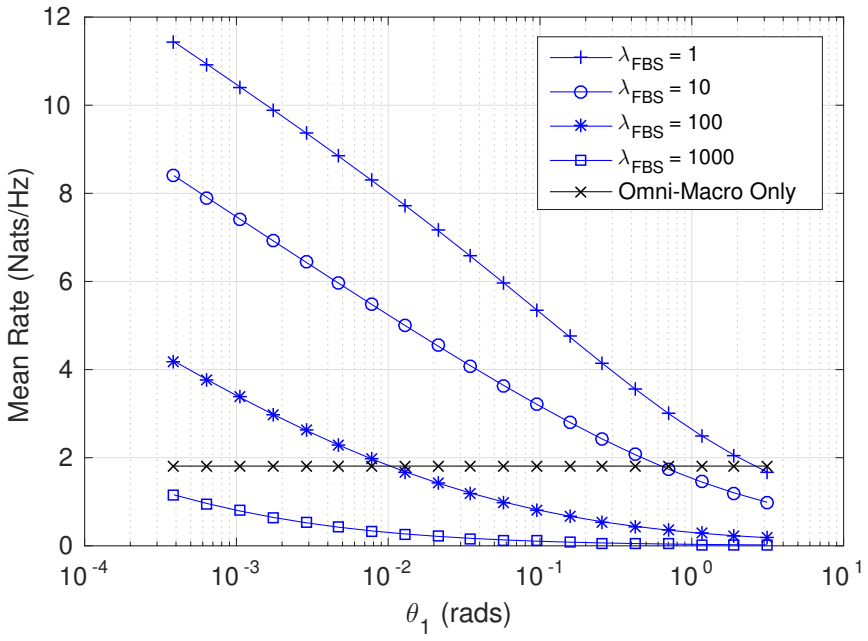


Figure 4.6: Mean rate across different beam-widths of a connected MBS tier, with a single interference FBS tier at increasing density. This is in contrast to the flat omni-directional case of for the MBS without the secondary interference FBS tier.

reuse with overlapping channels that provide trade-off spatially and in frequency. Finally there is *mmWave* communications which are not interference limited, but may not be suitable for macro-scale networks.

4.5 Discussion and Retrospective

In this chapter we derived results for a mixed GPP/PPP wireless K-Tier closed network models with directional antennas. The stochastic models are expanded toward a more accurate HetNet representation, specifically chosen to capture MBS and FBS tiers simultaneously. We demonstrated the reduction in coverage associated with a growing FBS tier in this CSN model. We also provided results on combating these effects with directional antennas in order to maintain specific rate goals. However, as certain densities the required antenna beam-patterns will not be practical to solely compensate for the additional loss in rate. Therefore, alternative strategies, such as frequency reuse should be considered.

From our analytical results we provide more insight into the HetNet deployments, how-

ever there are difficult hurdles to overcome for even the typical SG researcher. GPPs or more generally DPPs have been of recent interest to the community due to their representative properties, but this comes at the price of mathematical tractability. This cost in the long run will limit the complexity of models, which could more closely model network behavior and structure. Therefore, moving forward with this work future researchers must consider two options. First, alternative perspectives to these distributions must be consider that still maintain their usefulness. Such research is currently being performed by Haenggi *et al.* [107], but extensions or manipulations are still difficult. Other work in this simplification or alternative distributions approach include [108] which utilize Meta distribution in-place of more complicated versions. However, even these works require extreme depth in the SG toolset to understand.

The second option is to only utilize simple models such as the PPP or cluster process and construct modifications to the interference. Examples of this include introductions of rate awareness [109], *mmWave* communications [110], and even cell edge placed FBS [111]. Providing more approachable mathematics and simulation environments.

Outside of SG frameworks, cellular standards have begun to address some of these HetNet scenarios with modifications to ICIC. However, these have been heavily analysis in the simulation only frameworks. Such macro-scale simulation environments are generally not publicly available, and those that are have limited spectrum medium access models [112]. Nonetheless, small modifications to SG models have included BS silencing scenarios, which marginally cover ICIC [113]. These modifications are applied locally, and therefore are not representative logically in the network.

4.6 Chapter Summary

In this chapter, a new multi-tier HetNet model is proposed for examining future 5G cellular deployments. This modeling provides three unique perspectives beyond current literature. First, the model using independent PP together, which were defined as mixed PP, which were used to model the independent and structurally different deployments of network tiers. Second, DA were added to the GPP model and relative coverage analysis with respect

to the DA gain patterns were provided. Third, a closed subscriber network was considered for these mixed PP models which are realistic beyond open designs. Together these three unique properties were to examine interference reduction when co-channel interference is capacity limiting. Providing results that show usefulness of DA up to a point in network density.

Supplemental to the technical details we provide discussions throughout and at the end of this chapter to reflect on the meaning behind this work. Since a significant portion of this chapter specifically is heavily analytical we believe in evaluating the useful of this and future analysis using these tools. SG is not the only method for studying network-wide technologies, but it provides a strong mathematical foundation and may be more practical than investing in complicated simulators.

Chapter 5

Real-Time Localization Prototype For Dense Femtocell Networks

In this chapter we examine localization in dense FBS networks. With the densification of access points it becomes possible to perform accurate distributed localization in even indoor environments with some caveats. This represents an example task offloading possible in future 5G or generally dense networks. Unlike traditional cellular localization, precise submeter position could be performed, without the need for power hungry algorithms running on battery dependent mobile devices.

This work extends current research from our Finnish collaborators who are currently utilizing software simulations only to provide feasibility verification [114]. In this dissertation, we focus on the implementation of direction finding algorithms with SDR hardware to provide a representative scenario for eventual tracking of real-world targets. Replacing currently used Cramér-Rao bounds for estimation, which ignore real-world effects such as antenna placement, non-ideal antenna elements, receiver phase misalignment, and antenna coupling.

Compared with the current literature, the main contributions of this work are:

- A phased synchronized (aligned) SDR test harness, applicable to multiple SDR platforms.
- The Effective Aperture Distribution Function FFT DoA method built with respect to

a SIMO OFDM link, providing an integration study or practical use.

- A comparative study of MUSIC and an Effective Aperture Distribution Function FFT DoA method with over-the-air physical simulations is performed.

5.1 Collaboration Segmentation

This work has been a joint collaboration with Tampere University of Technology TUT, for the purposes of developing a physical simulation of their ray-tracing based computer simulation of DoA tracking. Figure 5.1 provides a division of activities between the current industrial partner and the Finnish researchers. The research here partially replaces the industry partners work with an over the air SDR implementation with a small number of interfacing BSs and mobile nodes (MN). The provided simulator is closed source, meaning that the internal implementation details are unknown to the researchers involved at TUT. However, their interface to the simulator is minimal and well defined, requiring only channel and DoA measurements. Both which are replace by the proposed SDR implementations.

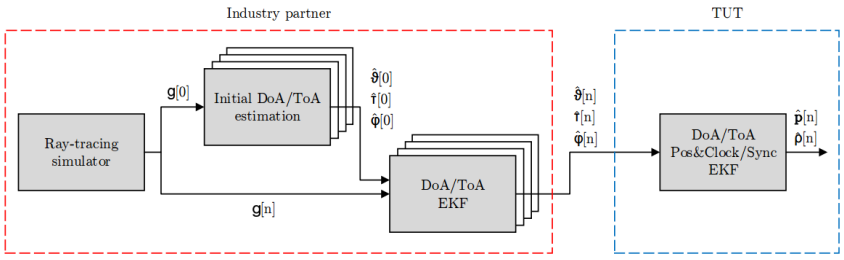


Figure 5.1: Collaboration replacement outline with Finnish partner.

Many modifications needed to be made to the existing research in order to accept the available SDR equipment. These differences will be outlines in the remaining sections. However, significant effort was put into making each member in the network as capable as possible in order to demonstrate realistic conditions. Much of the existing work in [114] relies up Cramér-Rao lower bounds which this work hopes to replace, making the research more practical.

5.2 Representative Network Level Model

In this work, a dense network with a high probability of line of sight (LoS) conditions are assumed. For the sake of simplicity, MN exist outdoors in an environment similar to one presented in [115] and redrawn in Figure 5.2 for reference. Such an environment allows for traditional DoA estimation with high accuracy in the two-dimensional plane, with limited accuracy in elevation. MN self tracking through GPS may be difficult in this environment due to the lack of LoS with orbiting satellites, making network localization favorable. Poor GPS performance is common in cities or areas with dense high-rise developments.

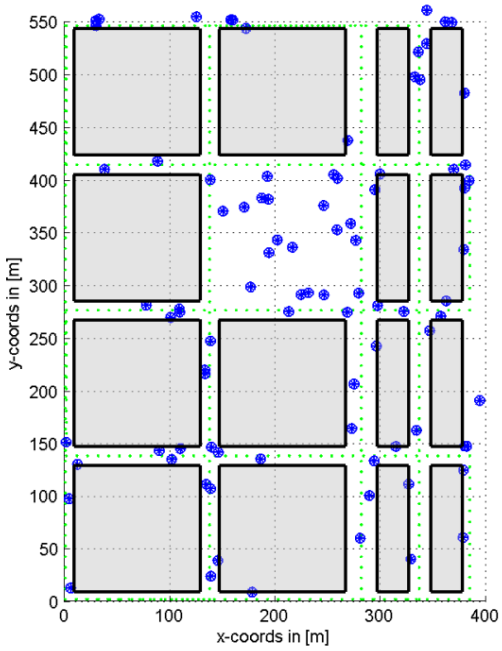


Figure 5.2: Example layout of a dense FBS network in an city environment. MN are located or travel along or near to the green lines in outdoor positions [115].

To perform localization, MN are assume to transmit uplink information to nearby base stations periodically. In LTE this could be a periodic Buffer Status Report [116] which is a required heartbeat-like transmission between a BS and connected nodes. Alternatively, for IEEE 802.11 this could be responses to beacon packets [76]. Wireless devices in general tend to have background contact, or contact not directly related to user actions, such as push email or other common services, which can provide necessary transmissions which we rely

on in this framework for localization. The physical MN themselves are assumed to be equip with single omni-directional antennas, transmitting energy to all nearby BSs. Therefore, only the proximity to a BS and LoS conditions will effect the DoA estimation performance. These constraints on the user are reasonable since its is unlikely for mobile users to be equiped with multiple antennas at sub 10 GHz bandwidths. However, this assumption may not hold in *mmWave* environments as discussed in Chapter 4.

BSs themselves are assumed to be equipped with a set or array of antenna elements, which has become common place in both cellular and commercial Wi-Fi networks. Multiple antenna elements are required for DoA estimation from a single BS or access point. The physical arrangement of the antennas and their number can define the effectiveness in specific dimensions and orientations. However, it is common for FBS or WiFi AP to be equipped with four or eight elements.

5.3 Direction of Arrival Estimation

The remaining sections focus on performing DoA estimation on SDR in an environment outlined in Section 5.2. This analysis is broken down into a series of phases to help understand DoA performance, as well as hardware requirements for such a system. First, we provide a theoretical background on array-processing and translations that have been made from the original model in [115] to the prototype array. These include two specific DoA algorithms, providing a comparative study between a traditional robust subspace technique and one used in [115].

Following the algorithm descriptions, the physical setup details will be outlined including engineering hurdles associated with creating a stable DoA platform with SDR hardware. Next, a baseline performance analysis is conducted using traditional DoA estimation techniques with the built SDR testbed to provide a feasibility study. This feasibility study is important since to our knowledge the secondary approach for DoA estimation is unproven in practical scenarios. Once both techniques provide stable results they can be integrated into the current localization analysis used by our collaborators.

5.3.1 Array Modeling

In a generalize form if we assume that N elements make up the antenna array and are located along the x-axis spaced uniformly (along the azimuth plane perpendicular to a target directly off boresight), the locations of the element are:

$$p_{z_n} = 0,\ \ p_{y_n} = 0,\ \ p_{x_n} = \left(n - \frac{N-1}{2}\right) d, \qquad n = 0,\, 1,\, \ldots,\, N-1, \tag{5.1}$$

where, d is the antenna element spacing. The origin of the array is assumed at the center element if there are an odd number of elements, or between the two central elements with an even number of elements. Now, if we consider the case where $d = \frac{\lambda}{2}$, since the elements are relatively close compared to the wavelength of the signal it can be assumed that the received signals $r(t)$ at the different sensors are correlated. However, the transmitter of $r(t)$ must be positioned in the far field $\frac{2D^2}{\lambda}$, where D is the height of the antenna (or diameter if larger) [117]. Located at this distance allows for the planar wave assumption at the array, and in essence produces delays to $r(t)$ relative to the angle of transmission (distance) to each element. This time delay can be written as a function of the azimuth and elevation angles θ, ϕ respectively, and the element positions:

$$\Delta t_n = -\frac{1}{c}\left[\sin(\theta)\cos(\phi)p_{x_n} + \sin(\theta)\sin(\phi)p_{y_n} + \cos(\theta)p_{z_n}\right]. \tag{5.2}$$

Next, rewriting (5.2) with respect to each axis:

$$u_x = \sin(\theta)\cos(\phi), \quad u_y = \sin(\theta)\sin(\phi), \quad u_z = \cos(\theta),$$

and defining $\mathbf{u} := [u_x\ \ u_y\ \ u_z]^T$, (5.2) can be further simplified into:

$$\mathbf{k}(\theta,\phi) = -\frac{\Omega}{c}\mathbf{u} = -\frac{2\pi}{\lambda}\mathbf{u} = -\frac{2\pi}{\lambda}\left[\sin(\theta)\cos(\phi) \quad \sin(\theta)\sin(\phi) \quad \cos(\theta)\right]^T. \tag{5.3}$$

Since $|\mathbf{k}| = 2\pi/\lambda$ is fixed, only the direction remains as an unknown. Utilizing (5.3), we can now define the array manifold vector (AMV):

$$\mathbf{v}(\mathbf{k}) = \left[e^{-j\mathbf{k}^T p_0} \quad e^{-j\mathbf{k}^T p_1} \ldots \quad e^{-j\mathbf{k}^T p_{N-1}}\right]^T. \tag{5.4}$$

For ULAs with uniform weighting across elements, we can write the azimuth beam-pattern as:

$$B(\psi) = \mathbf{w}^H\,\mathbf{v}(k) = \frac{1}{N}\sum_{n=0}^{N-1} e^{-j\left(n-\frac{N-1}{2}\right)kd} = \frac{1}{N}e^{-j\left(\frac{N-1}{2}\right)\psi}\sum_{n=0}^{N-1} e^{j\,n\psi}. \tag{5.5}$$

A beam-pattern formed from a narrowband beamformer [79], is simply a summed response of the antennas over varying source directions. Depending on the algorithms used, the weighting can vary but uniform is common for linear arrays.

Now if the spacing of the elements is examined as in Figures 5.3, 5.4, 5.5, 5.6 aliasing can be observed when going beyond $\frac{\lambda}{2}$ in spacing. To avoid spatial aliasing in array processing, the element spacing must be at most $\frac{\lambda}{2}$. When $d > \frac{\lambda}{2}$, aliasing occurs which manifests itself in terms of grating lobes. As shown in Figure 5.6 due to aliasing, the array cannot distinguish between signals at -90, -45, 0, 45, and 90 degrees. Alternatively, when $d < \frac{\lambda}{2}$ antennas begin to couple with one another

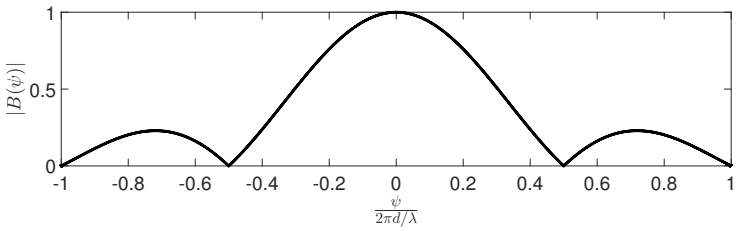


Figure 5.3: Beam-pattern of four element ULA with element spacing of $\frac{\lambda}{4}$.

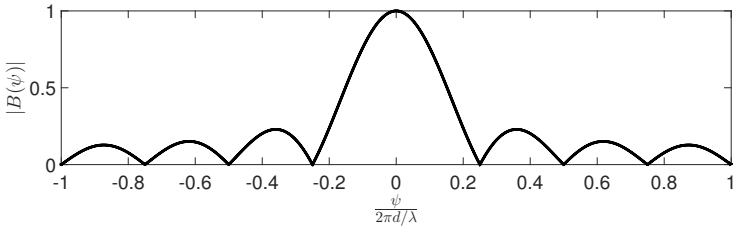


Figure 5.4: Beam-pattern of four element ULA with element spacing of $\frac{\lambda}{2}$.

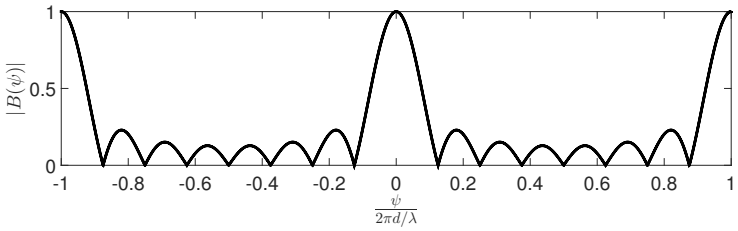


Figure 5.5: Beam-pattern of four element ULA with element spacing of λ.

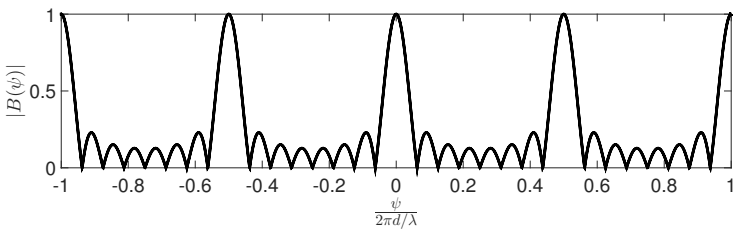


Figure 5.6: Beam-pattern of four element ULA with element spacing of 2λ.

To build a practical array, several modifications needed to be made to the original research. First unlike the expensive circular dual-polarized 3GPP patch elements from [115], simple monopole antennas in a uniform linear array (ULA) designed for the ISM 2.4GHz bands was selected. Due to regulations, transmissions cannot be made in non-public bands due to licensing. Therefore the chosen 2.4GHz ISM band is a reasonable alternative to the 3GPP carrier bands. ULA are well studied in the literature and have many desirable properties. However, we will be limited to azimuth direction finding for angles $\theta \in (0, 180)$ where 90 is the assumed bore-sight of the ULA.

Only a four element ULA was chosen since accurate azimuth estimation angles are the main purpose of the array, and positioning can be limited to a single orientation or perspective of the array. Using four antenna elements is reasonable for small inexpensive devices which these dense FBS networks will be more than likely built from. To provide even more foundation, due to available hardware in the market, there are very few SDR platforms that support more than four receive chains, making implementations rather impractical beyond four elements.

In the theoretical sense, for each antenna element an isotropic gain is assumed along the azimuth, creating an overall combined polar response provided in Figure 5.7 with four elements. Even in the theoretical sense the array has nulls at 30 and -30 degrees with respect to boresight. Beyond |80| to |100| degrees the gain performance is very poor. In the DoA sense, at position in these areas the estimation performance will also be poor for an array using this geometry.

The assumption of isotropic or constant gain is reasonable for commercial omni-directional antennas. In Figure 5.8 we provide the single antenna response measured by the manufac-

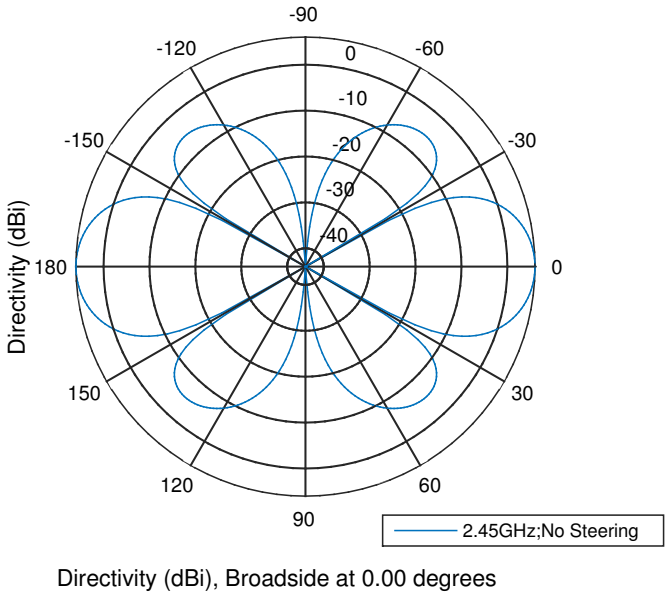


Figure 5.7: Theoretical response of four element ULA at 2.45GHz with a spacing of $\frac{\lambda}{2}$

turer [118] for the VERT2450 antennas using in our physical array setup. As can be observed across azimuth, the response is almost perfectly round as desired. We also operate targets of the array purely at 90 degree elevation, whose position provides the highest gain from Figure 5.8. However, the gain or radius of the azimuth pattern may differ between antennas. Fortunately, this has been shown not to effect estimation performance for DoA applications [119].

5.4 MUSIC Direction Finding

Multiple Signal Classification (MUSIC) is a frequently employed subspace processing method and is widely used for DoA estimation, but can also be used for general frequency estimation [120]. For MUSIC to perform accurate estimation a series of assumptions are required of the target transmitters and of the array itself:

- The received waveform $\mathbf{x}$ is narrowband, meaning the waveform does not significantly exceed the given channel's coherence bandwidth.

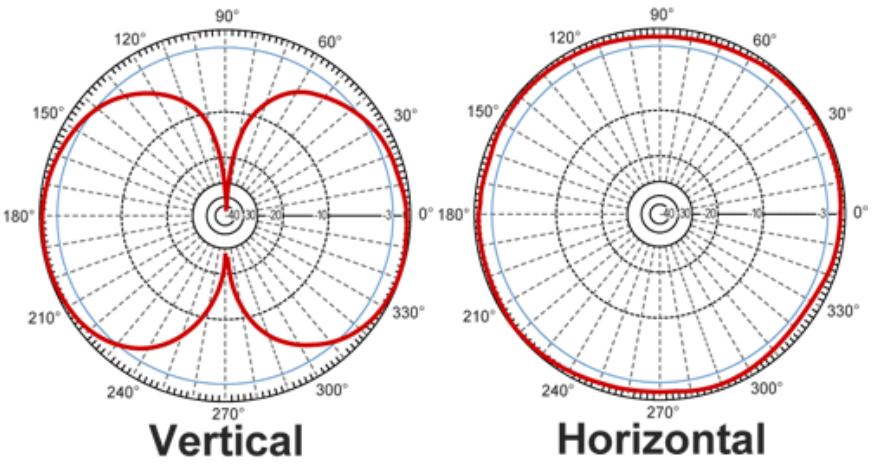


Figure 5.8: Antenna gain patterns of the VERT2450 antennas provided by Ettus Research [118].

- The received waveform $\mathbf{x}$ consists of D plane-wave signals plus uncorrelated noise $\mathbf{n}$.
- The number of plane-wave signals reaching the array, D is known.
- The number of antenna elements, N is at least equal to $D + 1$.

With these assumptions, from a signal processing perspective the received D waveform at an N element array of a transmitted waveform u can be expressed as:

$$\mathbf{x}(t) = \sum_{d=1}^{D} u_d(t)\mathbf{v}(\mathbf{k}_d) + \mathbf{n}(t) \tag{5.6}$$

such that $\mathbf{x} \in \mathcal{C}^{Nx1}$, $\mathbf{n} \in \mathcal{C}^{Nx1}$, and $\mathbf{v} \in \mathcal{C}^{Nx1}$. The noise ($\mathbf{n}$) power is σ^2 and is assumed to be AWGN. The correlation matrix (or spatial covariance matrix) for x can be then written as:

$$\mathbf{R}_{xx} = E\Big[\mathbf{x} \cdot \mathbf{x}^H\Big] = \mathbf{V}\,\mathbf{R}_{uu}\,\mathbf{V}^H + I\sigma^2 \tag{5.7}$$

such that:

$$\mathbf{V} = \mathbf{v}(\mathbf{k}) = \Big[\mathbf{v}(\mathbf{k}_1),\, \mathbf{v}(\mathbf{k}_2),\, \ldots,\, \mathbf{v}(\mathbf{k}_D)\Big]^T \tag{5.8}$$

and R_{uu} is simply the autocorrelation of u. In practice equation (5.7) is replaced by a sample correlation. Now assuming L recorded samples, which is commonly known as the snapshot length, on each antenna producing $\mathbf{X}^{NxL}$, the sample correlation matrix can be

calculated as:

$$\hat{\mathbf{R}}_{xx} = \frac{1}{L}\sum_{l=1}^{L}\mathbf{x}(l)\,\mathbf{x}^H(l) = \frac{1}{L}\,\mathbf{X}\cdot\mathbf{X}^H = \hat{\mathbf{V}}\,\mathbf{R}_{uu}\,\hat{\mathbf{V}}^H + I\sigma^2 \tag{5.9}$$

Now if the received D signals are non-coherent, then $\mathbf{R}_{uu}$ will be positive definite and the DoA of each signal should be separable. With this assumption we can obtain each orthogonal basis of the signal subspace from the matrix of manifolds $\mathbf{V}$ for each of the D dimensions, since each basis (β_d $d = 1, ..., D$) is related by a linear translation of $\mathbf{V}$ from a set of eigenvectors $\mathbf{e}_d$ such that:

$$\beta_d = \mathbf{V}\mathbf{e}_d, \tag{5.10}$$

these eigenvectors can be broken down into two categories or spaces [77]. First, any corresponding eigenvalue of $\mathbf{R}_{xx}$ of an arbitrary eigenvector $\mathbf{e^{\bullet}n}$ that is orthogonal to $\mathbf{V}$ is equal to σ^2 because:

$$\hat{\mathbf{R}}_{xx}\mathbf{e}_n = \hat{\mathbf{V}}\,\mathbf{R}_{uu}\,\hat{\mathbf{V}}^H\,\mathbf{e}_n + I\sigma^2\,\mathbf{e}_n = \sigma^2\,\mathbf{e}_n \tag{5.11}$$

Now there are $N-D$ eigenvectors with eigenvalues equal to σ^2, which we define as spanning the *noise space*. In practice these eigenvalues will not be identical. On the other hand, the remaining eigenvector(s) $\mathbf{e}_s$ of $\mathbf{R}_{xx}$ which are not orthogonal to $\mathbf{V}$ (lie in the column space) make up the *signal space*. There are D vectors spanning the signal space. The collection of noise space and signal space eigenvectors are denoted as $\mathbf{U}_N$ and $\mathbf{U}_S$ respectively.

In practice eigenvalue decomposition of equation (5.9) is used to determine the eigenvectors $\hat{\mathbf{R}}_{xx}$. However, since eigenvalues that are in the signal space have magnitude biased by the noise power, those eigenvalues will always be greater than those in the noise space. Therefore partitioning the two spaces is easily done by the magnitude of each eigenvalue and known signal count.

MUSIC utilizes the fact that the steering vectors $\mathbf{v}$ corresponding to the incoming signals lie in the signal subspace, and are orthogonal to the noise subspace. The directions are determined by searching through the set of all possible steering vectors and finding those that are orthogonal to the noise subspace. Mathematically for a signal received with relation $\mathbf{k}(\theta, \phi)$ to the manifold vector the following relation can be observed:

$$\mathbf{v}^H(\mathbf{k}(\theta, \phi))\mathbf{U}_N = 0 \tag{5.12}$$

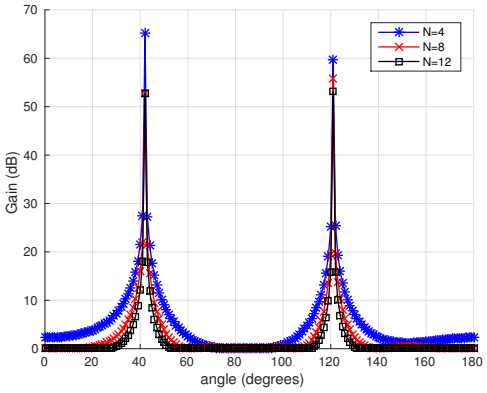


Figure 5.9: MUSIC pseudo-spectrum of two transmitted signals at 42 and 120 degrees with an increasing number of elements in a ULA with a spacing of $\lambda/2$.

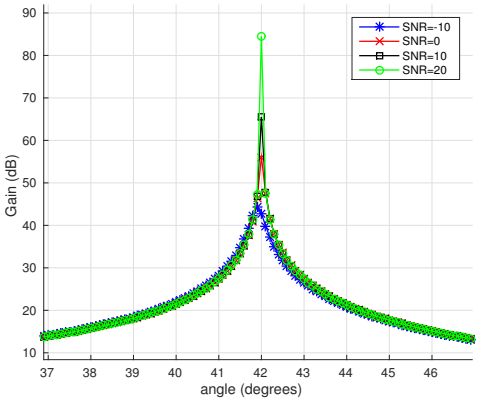


Figure 5.10: MUSIC pseudo-spectrum of a transmitted signal at 42 and 120 degrees with increasing SNR in a ULA with a spacing of $\lambda/2$.

It is common to observe the pseudo-spectrum of MUSIC realized by equation (5.13), which inverts the magnitude of equation (5.12) to pronounce likely angles as peaks.

$$P_{MUSIC} = \frac{1}{\mathbf{v}^H(\mathbf{k}(\theta,\phi))\mathbf{U}_N\mathbf{U}_N^H\mathbf{v}^H(\mathbf{k}(\theta,\phi))} \tag{5.13}$$

An example evaluation of MUSIC's spectrum with multiple sources ($D = 2$) with an array with multiple element counts is presented in Figure 5.9. As the element number increases so does the sharpness of the pseudo-spectrum. However even across significant SNR DoA measurements are still accurate, as observed in Figure 5.10. Therefore, we know that MUSIC is relatively robust in the presence of channel noise.

5.4.1 Root-MUSIC

Root-MUSIC is a simple extension to MUSIC, which assumes a specific array geometry to algebraically solve for numeric angles [121]. This is different than generic MUSIC which requires a linear search for peaks across the MUSIC pseudo-spectrum. RootMUSIC as shown in [121] is only applicable to ULA, but other extensions do exist [122]. If we consider a ULA with manifold:

$$\mathbf{v}(\theta) = [1, e^{j\pi dsin(\theta)}, \ldots, e^{j\pi(M-1)dsin(\theta)}]^T \tag{5.14}$$

with d as the antenna spacing. The inverse MUSIC pseudo-spectrum can be rewritten with use of a nullspace product $\mathbf{J} = \mathbf{U}_N \mathbf{U}_N^H$:

$$P_{MUSIC}^{-1} = \sum_{m}^{N-1} \sum_{n}^{N-1} e^{-j2\pi\, n\, d\, sin(\theta)} J(m,n) e^{j2\pi\, m\, d\, sin(\theta)} \tag{5.15}$$

Now by combining the exponents such that $z^{n-m} = e^{j2\pi\,(m-n)\,d\,sin(\theta)}$, we can define a polynomial of equation 5.15. With a slight abuse of notation define $\mathbf{J}_l$ as the sum across the l^{th} diagonal of $\mathbf{J}$, coefficients are formed as:

$$P(z) = \sum_{l=1-N}^{N-1} \mathbf{J}_l z^{-l} \tag{5.16}$$

Next from the polynomial $P(z)$, D out of $2(N-1)$ roots that lie on the unit circle are chosen as the most likely DoA. In practice, due to the presence of noise, the roots will not necessarily be on the unit circle. So, we select the D roots that are inside the unit circle and closest to the unit circle. An efficient way of solving for roots of P is through use of a companion matrix [123]. Overall this process does not rely on evaluating $\mathbf{v}$ over all θ, instead just solving for $2(N-1)$ roots of a polynomial is required.

5.4.2 Cramér-Rao Lower Bound

In the information theory sense it is important to characterize the estimation performance of MUSIC from a series of perspectives. The Cramér-Rao Lower Bound (CRLB) for DoA estimation of MUSIC was derived in the seminal paper [124] and shown here:

$$var(\theta) \geq \frac{\sigma^2}{2} \left\{ \sum_{k=1}^{L} Re\left[u^*(k) \frac{\partial \mathbf{v}^H}{\partial \theta} \cdot \left[\mathbf{I}_N - \mathbf{v}(\mathbf{v}^H \mathbf{v})^{-1} \mathbf{v}^H \right] \frac{\partial \mathbf{v}}{\partial \theta} u(k) \right] \right\}^{-1} \tag{5.17}$$

For a ULA with a spacing of $\lambda/2$ the steering vector $\mathbf{v}$ and its derivative are:

$$\mathbf{v}(\theta) = [1, e^{j\pi sin(\theta)}, \ldots, e^{j\pi(M-1)sin(\theta)}]^T \tag{5.18}$$

$$\frac{\partial \mathbf{v}(\theta)}{\partial \theta} = \left[0, j\pi cos(\theta) e^{j\pi sin(\theta)}, \ldots, j\pi(M-1) cos(\theta) e^{j\pi(M-1)sin(\theta)} \right]^T \tag{5.19}$$

First examining the root mean squared error (RMSE) across DoA angles θ in Figure 5.11, it can be understood that near the edges of the array performance becomes poor. This

also explains why ULA cannot distinguish beyond 180 degrees of azimuth. Expanding to rectangular or circular arrays can provide these extra perspective for DoA estimations beyond 180 degrees. However, due to shadowing effects and coupling caused by nearby elements, azimuth estimation performance in certain region can suffer. Such effects can be hard to model in simulation because these effects tend to break planar wave assumptions.

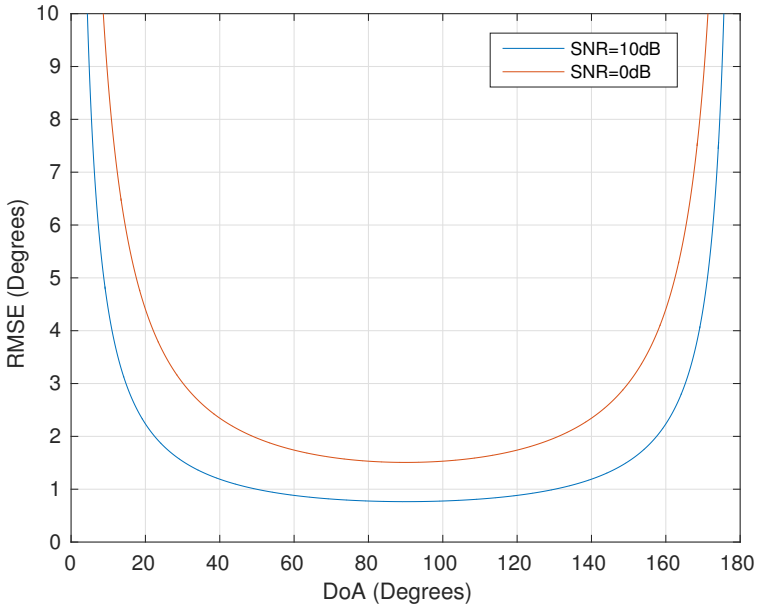


Figure 5.11: CRLB of MUSIC across azimuth angles of a ULA for two different SNR conditions.

Averaging over $\theta \in [-89 : 1 : 89]$ in Figure 5.13 and in Figure 5.12, performance over varying configurations and environments is examined. Average RMSE over SNR shows limited gains, and snapshot length presenting limited effect beyond 40 samples. This provides some useful criteria for implementation, demonstrating that performance is relatively uniform after $L = 40$ and with an SNR of 20 dB. However 20 dB is an unreasonable target SNR, therefore in Figure 5.11 the analyzed choices for SNR $(0, 10)$ are more reasonable especially for indoor conditions.

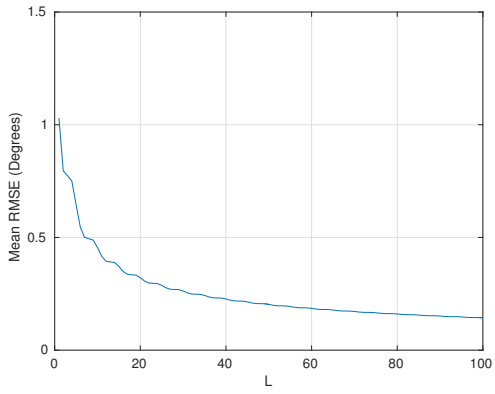


Figure 5.12: CRLB of MUSIC over different snapshot lengths averaged across azimuth angles of a ULA.

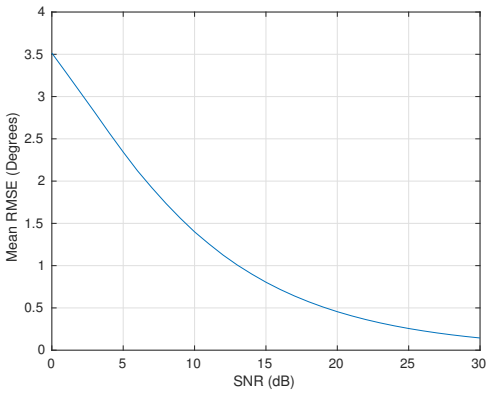


Figure 5.13: CRLB of MUSIC over different SNRs averaged across azimuth angles of a ULA.

5.4.3 GNURadio Software Only Implementation

Before we can go directly to hardware, MATLAB simulations needed to be translated to the GNURadio development environment. GNURadio was chosen as an alternative platform to the proposed framework developed in Chapter 3 due to a lack of hardware support for phase coherence between SDR. As of version R2016a, MATLAB does not support receiver frequency reference and clock synchronization required for phase coherence and alignment on USRP SDR platform which we have available. An alternative device which provides phase alignment between four channels automatically is the FMComms5 SDR. However, we do not have access to such a device.

Porting the algorithms from Sections 5.4 and 5.4.1 involved reimplementation and design of the system to fit into the GNURadio flowbased paradigm. Partitioning of the algorithms was straight-forward since the majority of the computations can be broken down into two sections, autocorrelation and the antenna response search. MUSIC is purely feed-forward, requiring no complicated state maintaining blocks. Figure 5.14 provides a simulation only flowgraph in gnuradio of MUSIC. The algorithm itself is divided into three main blocks: autocorrelate, MUSIC (estimation), and Find Local Max.

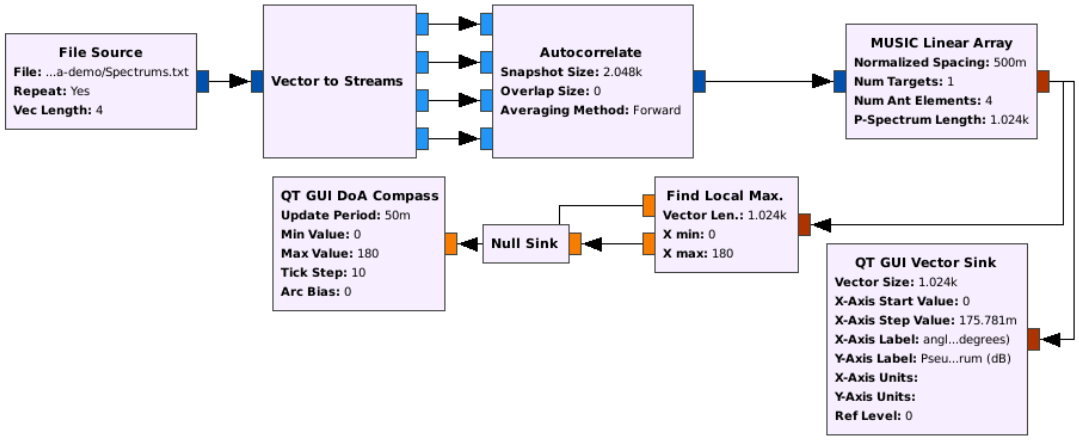


Figure 5.14: GNURadio simulation only flowgraph of MUSIC algorithm with received manifold source from a file generated in MATLAB.

The autocorrelation block is universal between Root-MUSIC and general MUSIC, therefore there are also modularity benefits of partitioning the algorithm in this way. A flowgraph with Root-MUSIC simply replaces the MUSIC block in Figure 5.14 with a Root-MUSIC block, and only exports the numerical angles it finds. However, as mentioned in Section 5.4.1 traditional Root-MUSIC is limited to ULA.

5.5 Effective Aperture Distribution Function

The work outlined for this project as discussed earlier is broken down into several phases. The first was developing a DoA reference implementation for baseline analysis and the second was an integrated solution based on [115]. The discussed MUSIC implementation was designed in Section 5.4, and the integrated solution will be developed in this section. The integrated DoA estimation technique is based off an array response correlation using an Effective Aperture Distribution Function (EADF), with estimates provided by an existing OFDM link. Providing a minor add-on to an existing OFDM MIMO receiver to provide DoA estimates.

In principle, the EADF is a n-dimensional (usually $n = 2$) discrete Fourier transform of an augmented beam pattern. More simply, it is an alternative representation of an

antenna response, which are traditionally stored as spherical coordinates (azimuth and elevation) [125]. For an EADF a frequency domain description brings the advantage of simple and computationally efficient derivative approximation and interpolation capabilities [126]. Differentiation of the array response allows for characterization of the estimation bounds as in Section 5.4.2. The evaluation provide in Section 5.4.2 is relatively straightforward, but under more complex or arbitrary geometries this task can be almost intractable. However, unlike storing the response in spherical coordinates, utilizing the EADF for CRLB can be calculated easily for such arbitrary array geometries. This is primarily a function of EADF's usage of a fourier transform to represent a response, which provides smooth surfaces in the resulting form. Nonetheless, the majority of the literature on EADF focuses on antenna pattern modeling [127–129]. Therefore from this information optimal antennas or beam-patterns can be designed through this technique, but in this work we focus on DoA aspects.

To generate an EADF for a particular antenna let us begin with the antenna response (beam-pattern) $\mathbf{B}^{N_1 \times N_2}$, discretized with N_1 elevations and N_2 azimuth positions. The pattern is only two dimensional in this case but can be extended. First, we assume that measurements are taken in series across azimuth, therefore $\mathbf{B}$ will be periodic in azimuth. However, in order to make measures periodic in elevation $\mathbf{B}$ must be extended for elevations beyond the traditional π to $\sim 2\pi$. This is achieved by creating a new periodic pattern:

$$\mathbf{B}_p = \begin{bmatrix} \mathbf{B} \\ \mathbf{\text{-}B} \end{bmatrix} \tag{5.20}$$

where $\mathbf{B}_r$ is the data shifted 180° in azimuth and flipped across elevation. Taking the two dimensional discrete Fourier Transform results in the EADF of the beam pattern $\mathbf{G}$. This operation is applied for each antenna element of an array. $\mathbf{G}$ can be truncated, which will reduce the resolution, by removing rows and columns of the matrix from the outer edges. Resulting in $\mathbf{G}^{N \times M_A\, M_E}$, where M_A and M_E are the number of azimuth and elevations modes respectively. In the case of a monopole antenna with a height of quarter wavelength and of width 1 mm the resulting $\mathbf{G}$ for the vertical polarization is provided in Figure 5.15. We observe flat areas surrounding the central pillar in Figure 5.15, which tells us that no significant information on the antenna response is provided in these regions. Therefore, we

can simply compress this response as suggested by removing the outer rows and columns in **G**.

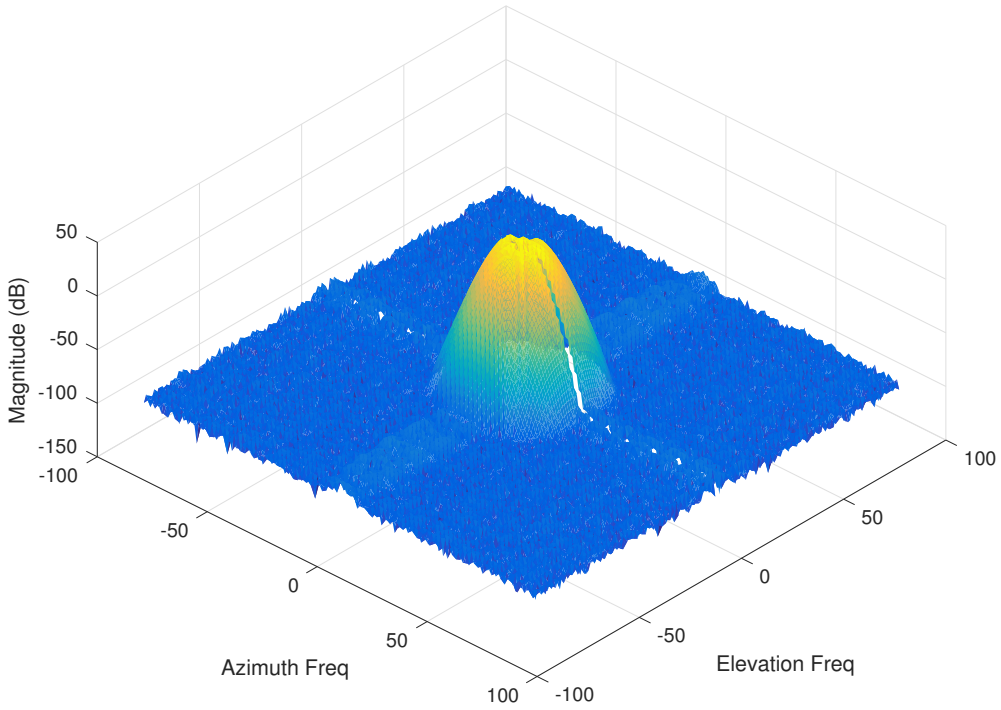


Figure 5.15: EADF of monopole antenna for vertical polarization only before compression is applied through row and column removal.

5.5.1 EADF-DoA Estimation

In essence the EADF provides a compact alternative to the array response, discretizing individual antenna responses over elevation, azimuth, and frequency. This fact can be used to essentially correlate an estimated response $\hat{\mathbf{h}}$ against the EADF responses $\mathbf{G}$ to determine the DoA of the estimated signal. This techniques was derived from work presented here [127]. First we define the channel between the transmitting node and receiving array in terms of the polarimetric response:

$$\mathbf{h} = \mathbf{C}(\theta, \phi)\,\boldsymbol{\gamma} + \mathbf{n}, \tag{5.21}$$

where:

$$\mathbf{C}(\theta,\phi) = \begin{bmatrix} \mathbf{G}_H\,\mathbf{d}(\theta,\phi) \\ \mathbf{G}_V\,\mathbf{d}(\theta,\phi) \end{bmatrix} \tag{5.22}$$

at a single transmission frequency. The mapping vectors $\mathbf{d}(\theta,\phi)$ are of the form:

$$\mathbf{d}(\theta) = \Big[exp(-j\theta(M_A-1)/2),\ldots,exp(j\theta(M_A-1)/2)\Big] \tag{5.23}$$

$$\mathbf{d}(\phi) = \Big[exp(-j\phi(M_E-1)/2),\ldots,exp(j\phi(M_E-1)/2)\Big] \tag{5.24}$$

$$\mathbf{d}(\theta,\phi) = \mathbf{d}(\theta)\otimes\mathbf{d}(\phi) \tag{5.25}$$

In practice $\mathbf{h}^{D\times 1}$ must be estimated as $\hat{\mathbf{h}}$, which is then used to determine θ and ϕ. θ and ϕ are estimated by first convolving them with the independent EADF polarizations:

$$\mathbf{A}_H = \hat{\mathbf{h}}\mathbf{G}_H^* \tag{5.26}$$

$$\mathbf{A}_V = \hat{\mathbf{h}}\mathbf{G}_V^* \tag{5.27}$$

Then converting them back into spherical coordinates with a Fourier Transform again:

$$\mathbf{C}_H = |FFT_{2D}\{A_V\}|^2 \tag{5.28}$$

$$\mathbf{C}_V = |FFT_{2D}\{A_V\}|^2 \tag{5.29}$$

Finally by searching through the combined space $\mathbf{C}_{HV} = \mathbf{C}_H + \mathbf{C}_V$, the maximum point over the surface provides the most likely DoA index for azimuth and elevation indexes. In Figures 5.16 and 5.17 $\mathbf{C}_{HV}$ surface plots are provided for transmissions originating at 90° and 125° in azimuth respectively. In these Figures there is a symmetry maintained as the target moves across azimuth, which shows that if the planar wave approaches the array from behind at 180° the frontal approach, there will be no difference in the array responses. The result would differ in 2D planar arrays. One problem with this response is that it is symmetrical with respect to bore-sight, which can lead to miss-interpretation of the originating direction. For examples with our linear array, with a target place 25 degrees off bore-sight may correlate will with the response at position −25 degrees off bore-sight. The main issue is that this correlation method is not as dependent on the relative nature of the elements as does MUSIC, leading to inconsistencies. This EADF-FFT DoA

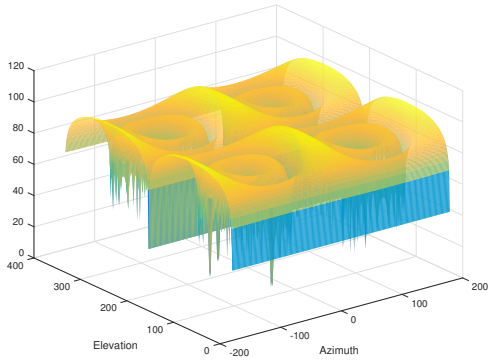


Figure 5.16: CRLB of MUSIC over different snapshot lengths averaged across azimuth angles of a ULA.

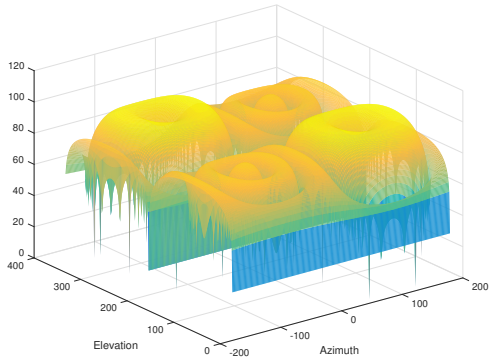


Figure 5.17: CRLB of MUSIC over different SNRs averaged across azimuth angles of a ULA.

(EFD) implementation for comparison is similar to classical fourier transform or correlation method for a ULA. Both methods are searched based, but MUSIC inherently will have more resolution and can be considered more robust to noise [130] due to implicit averaging.

5.5.2 OFDM Receiver, Channel Estimation, and GNURadio

The EFD estimation was chosen for two specific reasons. First, it is effective for arbitrary array geometrics, and second the EFD can easily complement or be integrated into an existing communication system. The integrated DoA method here means that it was built with respect to a specific physical layer system or communication link by simply piggy-backing off channel estimation. Consequently, relying on the link to packet flow itself removes requirements on a signal detection system, since it will be inherent to the system already. However, bolting EFD on top a link does introduce requirements into the receiver itself. Generally, since [115] utilizes a cellular-like OFDM link, an OFDM physical layer was implemented to be used with EFD. Again the goal of this work is to provide real-world analysis and measurement variance. Therefore this OFDM link as well was designed for over-the-air operation with some additional requirements to be discussed.

Like the MUSIC DoA work discussed in Section 5.4.3, the OFDM link for EFD estimation was built in GNURadio. However, unlike some of the existing OFDM solutions that are

in GNURadio, this receiver needed to support a SIMO design with multiple receive paths, resulting in a significant rewrite and ordering of OFDM symbol recovery. Since OFDM reception and recovery is not the purpose of this research, a relatively high level design overview is provided here and details on requirements of EFD.

The transmitted OFDM packets shown in Figure 5.18 are separated into three parts: the preamble, the header, and payload. The preamble is a symmetric OFDM symbol with desirable autocorrelation properties addressable in the time-domain [131]. The remaining pieces, header and payload, are BPSK or QAM modulated based on input parameters to the system. However the header is of fixed length, and only contains the length of the payload in bytes with a single repetition for checking purposes. The payload does contain a CRC check, therefore channel estimates made from packets can maintain some level of validity. In the current implementation header only estimates are used for channel estimation, providing a more compact solution. However if the variance of the estimates is poor, then averaging estimates across payload subcarriers can provide additional stability. Both the header and payload are OFDM modulated as well.

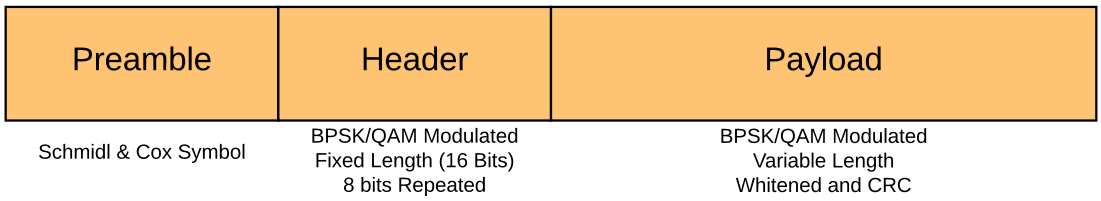


Figure 5.18: OFDM packet structure for GNURadio implementation. Preamble is not modulated, unlike the header and payload which are both BPSK/QAM modulated and then OFDM modulated.

The receiver itself is outlined in Figure 5.18, which presents a four receive path design but programmatically will expand to any number of receive chains. The receiver has five distinct stages to recover a packet and perform DoA estimation. The important design perspective here is that the phase relationship between the receive chains is maintained. These relative phase relations are a product of different signal reception by a different antenna, which we simple model by complex gains in the AMV $v(\mathbf{k})$. However, since there is a significant amount of signal processing needed to recovery a packet, this phase relationship must be

considered at all stages. Now once the signals are coherently received by the USRP(s) only a single chain is used for packet detection and frequency correction. Both operations are a result of the same algorithm from [131]. Identical corrections are applied to each chain, then fed into the third stage for OFDM boundary alignment and header correlation. Again only the first chain is searched across for the header symbols. Once found a zero-forcing channel estimate is made and an equivalent correction is applied to each chain.

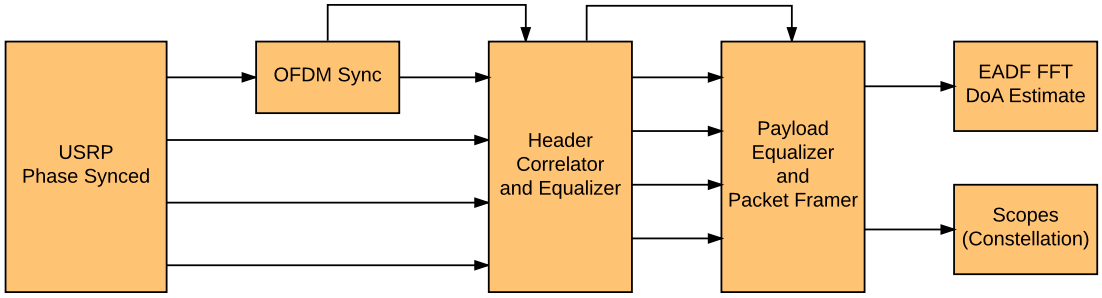


Figure 5.19: OFDM packet structure for GNURadio implementation. Preamble is not modulated, unlike the header and payload which are both BPSK/QAM modulated and then OFDM modulated.

Now that OFDM symbol edges are known, an IFFT is applied, guards carriers removed, and individual symbols are carried forward. With the forwarded data-carriers a flag is forwarded when headers are located, found through a simple correlation. These flags are important since they provide triggers for the state machine in stage four. In stage four, the header is decode, checked, and packet length determined. To help limit phase rotations, phase corrections are applied and updated as the payload is processed. Instead of using pilots, a simple zero-force decision feedback equalizer (ZF-DFE) is used. Now technically the payload are passed into a queue for processing, but at the moment that stage is ignored. Instead, asynchronous messaging [132] is used to send channel estimates of each subcarrier for each receive chain to an EADF-FFT block for DoA estimation. For diagnostic purposes, unequalized subcarriers are output to a set of constellation scopes.

Across each receiver stage a single primary chain is used to estimate channel effects and TX/RX mismatches. This reduces corruption to the phase effects cause by the antenna array orientation to the transmitted signal, which is needed for the EFD estimation. It is important to note that the implemented OFDM link is only one way, and no feedback is

provided to the transmitter.

This link was tested with simple AWGN channels in GNURadio which successive gains being applied to the individual chains to provide array perspectives of directional transmissions. Once this was working and stable physical simulations could be considered, these physical simulation will be discussed the following sections, including the USRP hardware setup, and measurement scenarios.

5.6 USRP Synchronization

In order to perform accurate DoA estimation, the hardware array must be phase coherent, or ideally, phase synchronized across all antenna elements. The original SDRs chosen for each array consists of four USRP-N210 devices with a selection of daughterboards. Since each radio is inherently independent, they require synchronization across them to provide coherence. Ettus Research, the designer of the USRP Series, provides several mechanism to help reach this goal. However, additional methods need to introduced to further match the individual radios together. In this section, a solution or harness is built to effectively synchronize not only USRP-N210, but also other radios with similar features.

Out of the box Ettus Research provides ways to perform motherboard (mixer) frequency matching and timing synchronization. This is accomplished through the bottom right two input SMA port in Figure 5.20. The REF CLOCK input is for a 10 MHz reference to provide a single frequency reference for devices in the array. This is used to drive mixers and other components (DAC/ADC), resulting on zero frequency drift between devices. The PPS IN input utilizes a 1 PPS square or sine wave to set edge clocking in the devices. This forces sampling and device times to periodically (every second) be aligned. Combined usage provides timing and frequency locked radios. However, to eliminate any differences between radios when these events occur it is important to distribute clocking in an equal manor. This is accomplished through the use of an Octoclock [133], which uses active components to amplify and distribute signals with matched-length traces to up to eight devices. The Octoclock here is driven by a two function generators, which are clock off one another. Now to remove any phase ambiguity of the REF and PPS signals, equal length cabling is

Figure 5.20: USRP-N210 from Ettus Research front panel. Consists of frequency source (REF CLOCK), clocking source (PPS IN), antenna ports (RF1,RF2), MIMO cabling input, and Gigabit ethernet.

used from the octoclock to each USRP. In Figure 5.21 for demonstration purpose all eight outputs were used for an 8×8 array, but for this work only four are utilized. With the

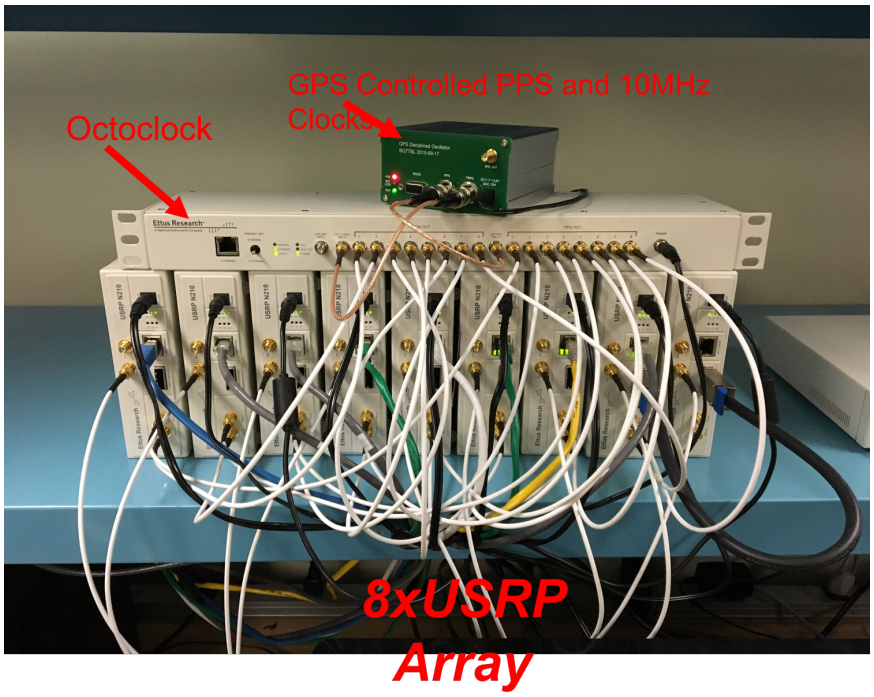


Figure 5.21: Eight USRP array synchronized with Octoclock and equal length cabling. The Octoclock is driven by external sources to provide REF and PPS signals.

Octoclock each radio shares the same frequency source and timing instances. However, since the RF synthesizers are built into the daughterboards, phase ambiguity will still exist across the receive chains. The properties of this phase offset is uniquely dependent on the

daughterboards used, and officially for N210 devices only the SBX daughterboards can be aligned. This is defined in this way since between device retunes (frequency or sampling changes) phase offsets across devices will remain constant only on SBX boards. This is true between power restarts as well. However it must be noted that for this relationship to be maintained requires timed commands [134] to be issued when tuning the devices. Now, since these are still active devices, periodic calibration will be necessary for phase-coherent applications since phase offset will drift over time due to thermal and other characteristics. For reference an example flowgraph using timed commands is provided in Appendix 6.2.

Even though with use of the SBX daughterboards there will still exist a random but constant phase difference between the receive chains. In this work two methods for correcting these residual offsets are explored. The first requires SBX daughterboards and the second method does not have this limitation. However, all USRP devices still require phase and timing synchronize, as previously discussed. In both cases the obvious solution is that we simply measure the relative offsets and compensate for the phase shifts in software. Measuring the offsets is accomplished by transmitting a reference signal r_n into each chain and calculating the phase difference with respect to a selected reference chain k:

$$\Delta\phi_n = tan^{-1}(r_n) - tan^{-1}(r_k) \tag{5.30}$$

Since the phase relationship remains constant between retunes, we can simply apply this to each individual receive chain as:

$$\hat{r}_n = r_n \, exp(j\Delta\phi_n) \tag{5.31}$$

This was simply implemented with a "Const Multiply" block in GNURadio. In Figure 5.22 a four chain correction is demonstrated in GNURadio using this technique with a reference sinusoidal tone. The source or reference signal should be irrelevant because only the relative phase between the signals are important. An alternative approach that may provide more stable results is a phase locked loop (PLL) estimation [135]. Complexity of the PLL can be reduced by driving the transmitter with the same reference as the array, removing the need to estimate the carrier mismatch. Designing the PLL as an overdamped systems provided the best results in experimental testing here, however this may require more convergence time.

In this work a fifth USRP is used to transmit this reference tone using a resistive quad splitter and equal length cabling connected to each receiving port, but a RF tone generator would be effective. Resistive splitters are useful since they have a large frequency range, but can attenuate the signal. This however was combined with a 30 dB attenuator in-line with the transmitting USRP to reduce possible damage due to high input power from the transmitter. Alternatively, if other daughterboards are used a second method can be imple-

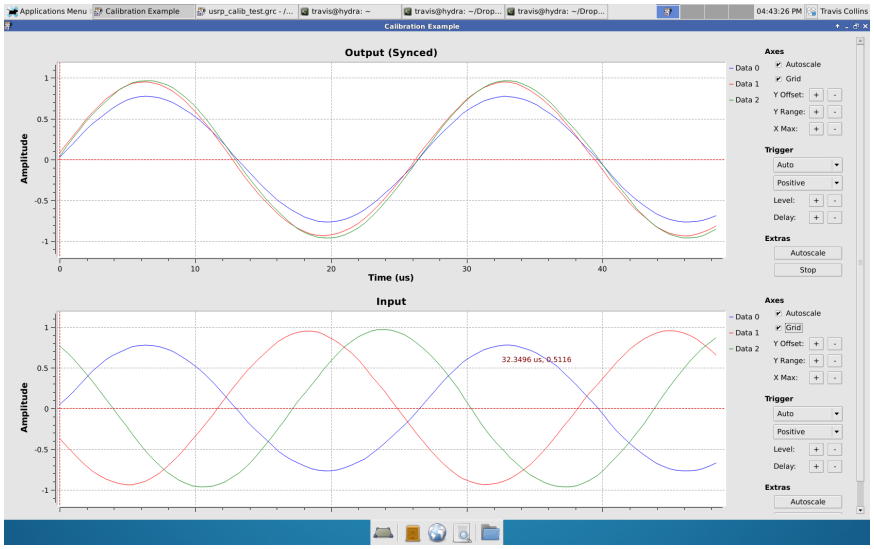


Figure 5.22: Pre calibration (Bottom) and post calibration (Top) of complex tone feed through cabling to four USRP-N210's.

mented for phase correction. However, these daughterboards must guarantee phase offsets less than a single sample. If not, the estimation and correction must be more complicated. Nonetheless, this method is also useful for conditions where temperature swings occur or there is general oscillator instability. In this case the secondary input port, usually the TX/RX port on several daughterboards, is used for the reference tone input. No attenuator is needed in this case, since the daughterboard itself attenuates the signal which bleeds over into the main receive chain. Since this distance is identical on same model daughterboards, phase can be accurately measured. This method requires a synchronization phase every time a flowgraph starts in GNURadio, where the chains are resynced. Furthermore, this technique can phase synchronize any type of daughterboard under certain conditions. The cabling connections are outlined in Figure 5.23, where again a fifth USRP (in blue) was

used for synchronization tone generation. If phase drift is a significant concern, tones can be generated out of band of desired signal to maintain always on phase correction.

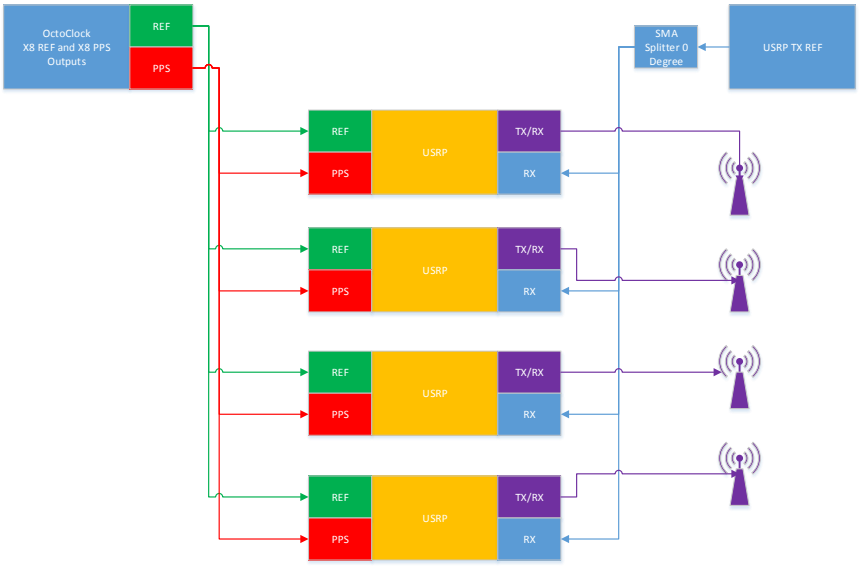


Figure 5.23: Multi-USRP configuration for online correction, or in configurations without repeatable phase between initializations.

There are significant caveats to phase aligning multiple receive chains. A given phase correction is only valid for a specific sample rate, carrier frequency, and gain configuration. Since these parameters heavily influence the receive (and transmit) paths, they will have effects on phases. Timed commands do not remove these limitations, such commands only make phases repeatable in specific configurations.

5.6.1 TwinRX Daughterboards and Software Libraries

After prototypes of the original MUSIC implementation were presented at New England Software-Define Radio (NEWSDR) workshop [136], more advanced USRP-X310s [32] were employed for further research. However, the X310 radio were provided with then unreleased TwinRX daughterboards [80]. These daughterboards are unique since each consists of two receive chains which can share local oscillators (LOs). These LOs can even be exported to a second daughter within the devices, allowing up to four receive chains per radio. The radios also support dual 10-Gigabit Ethernet, for a maximum combined rate of 400 MS/s.

However, even though receive chains share LOs they are not phase synchronized completely, only coherent. Like the N210 with SBX daughterboards, these devices have random but fixed offsets between initializations. The TwinRX do have superior phase performance relative to a multi-N210 setup and are immensely easier to transport. After a stabilization period the TwinRX chains will remain within one degree of one another.

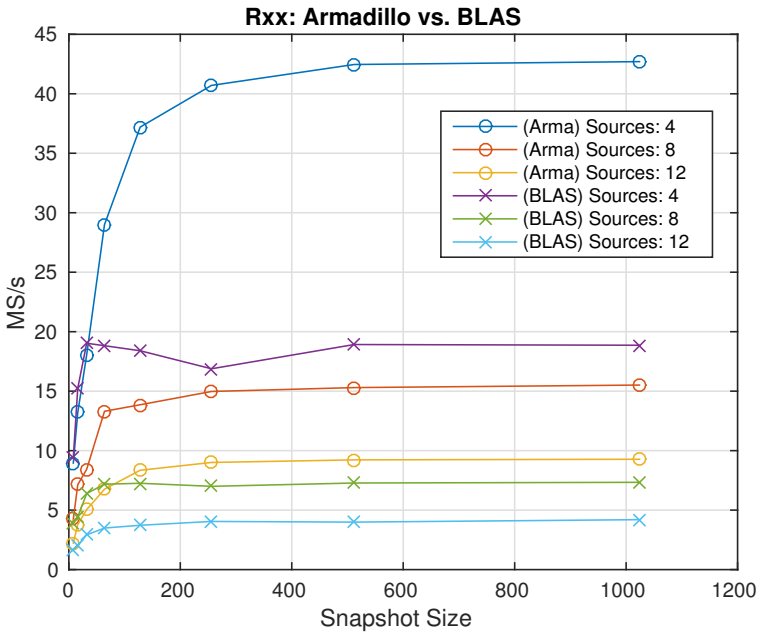


Figure 5.24: Performance comparison between Armadillo C++ libraries and raw BLAS calls through the Fortran interface in C++.

The MUSIC algorithms were re-written for modularity and performance in mind. As outlined in Section 5.4 the majority of computations are linear algebra based. Therefore, two approaches for languages approaches were considered. First the C++ template library Armadillo [137] which is specifically designed for speed and matrices was consider. Second, direct BLAS calls through the Fortran API were implemented. Directly using BLAS was initially considered better since it reduces unneeded memory copies to custom containers in Armadillo. Both BLAS and Armadillo were compared in the "autocorrelate" block designed to form equation (5.9). Although this operation seems trivial, since the dimensions of x are usually large this matrix multiply is by far the most computationally expensive

operation. Figure 5.24 is a resulting benchmark run to examine throughput of this block at different snapshot lengths over various numbers inputs. Armadillo clearly outperforms directly calling BLAS, or BLAS's cgemm [138] function specifically.

5.7 Experimental Setup

So far both algorithms have been implemented in both MATLAB for numerical validation and finally ported to GNURadio for real-time testing. To complete evaluation and characterize performance, a number of test scenarios were developed for a set of available real-world environments. These tests were designed to model the performance of the hardware and software algorithms together in terms of DoA estimation performance. For initial testing the four antenna array fixture shown in Figure 5.25 was used. This setup uses a normalized spacing of $\frac{1}{2}$ or simply $\frac{\lambda}{2}$. All cables were specifically selected to have equal length paths from antennas to daughterboards, as well as from the Octoclock to each radio. Alternatively, the USRP-N210's were only using in the first test then replaced by a single X310 which was much easier to move around. The specific radio(s) used in each test we will explicitly defined, but every effort was made to keep the overall radio platform (no matter the radio used) fully synchronized. The antennas used were VERT2450 monopole antennas with an estimated gain of $3dBi$ at 2.45 GHz. The overall testing in general provides insight into error reduction of DoA estimates and refining of our experimentation methodology.

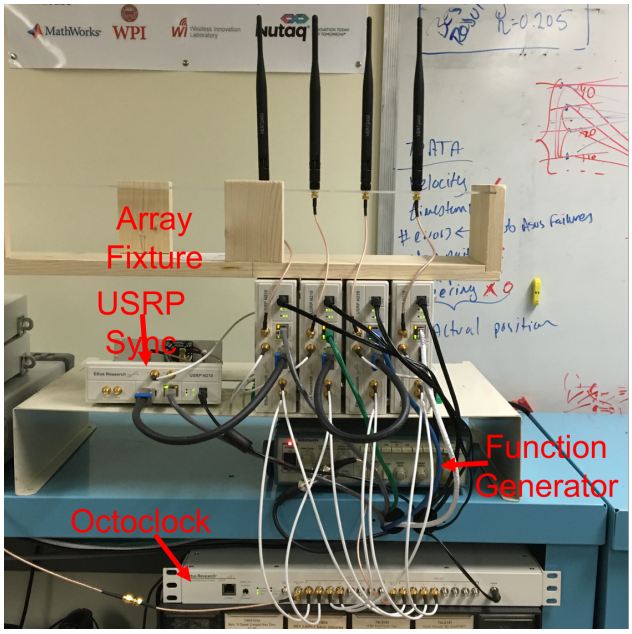


Figure 5.25: ULA fixture with four receiving USRP-N210's, one USRP-N210 used for synchronization tone generation, and an Octoclock used for clocking synchronization. The Vert2450 antennas are space is $\sim 2.4in$ or $\frac{\lambda}{2}$ for 2.45 GHz carrier transmissions.

5.7.1 Initial MUSIC Real-World Testing

Chronologically MUSIC physical simulations began much earlier than even EFD implementations. This is primarily due to delay with the Finnish commercial contractor and implementation details of their ray-tracing simulations. However, this provided time to validate MUSIC and the USRP hardware setup. For testing it was important to reduce possible wave bounce or general reflections in the environment. Reflections undermine the assumptions presented in Section 5.4 about uncorrelated signals, resulting in minimal SNR in the pseudo-spectrum or even angles favoring reflections over LoS observations. Therefore, original proof-of-concept testing with the four USRP-N210 array was performed in a 100×50 meter room without obstacles blocking LoS. Two sets of measurements were taken at different distances from the array, which was placed 10 meters from the wall in the center of the room to remove possible wave bounce. Between tests at these distances the radio were phase synchronized on-line. Figure 5.26 provides the positioning of a transmitter to be estimated with respect to the array. At each position 5 seconds were granted for the algo-

rithm to stabilize and an angle measurement was taken. Since nothing in the environment was moving these angle values are maintained after the initial startup window. As shown in equation (5.9), there is significant averaging performed in MUSIC when L is large. For our experimentation $L \geq 1024$.

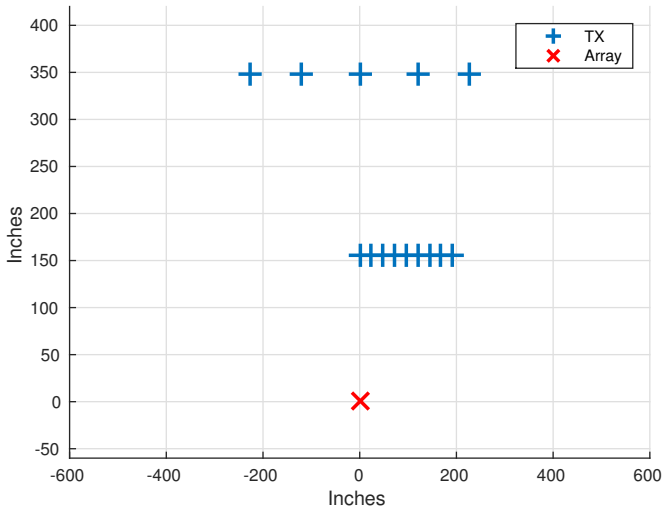


Figure 5.26: Relative positioning of transmitter and phased array for DoA over-the-air experimentation. These are positions within the WPI Odeum in the center of the room offset from the wall by 10 ft.

The y distances (13 ft, 29 ft) were chosen to reduce any near field effects or possible scatters that result from the floor or transmitter itself. However, when comparing the results in Figure 5.27 and 5.28 this distance does not provide a significant advantage. Providing evidence that the transmitter is well into the far field and primarily the LoS path is dominating the results. Furthermore the RMSE roughly increases as the target increase or decreases in azimuth away from boresight in Figure 5.27 as expected for the CRLB analysis. Although, this pattern is similar in Figure 5.28 the effect is not as obvious. Overall these results are not perfect, but for a realistic array with four independent radios with a custom synchronization harness, they are reasonable.

Moving forward, in order to reduce measurement error further the four independent USRP-N210 setup was replaced with a single USRP-X310 with dual TwinRX daughterboards provided by Ettus Research. The TwinRX daughterboards are designed specifically

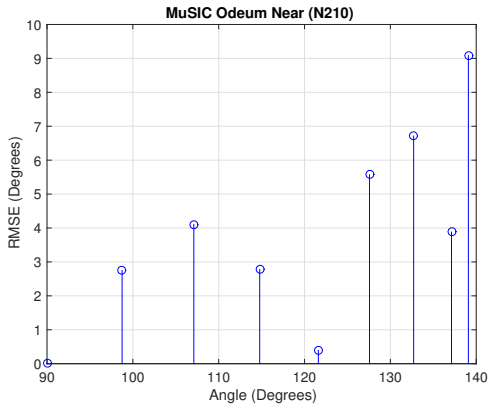


Figure 5.27: MUSIC RMSE across azimuth for initial large room test at 13 ft.

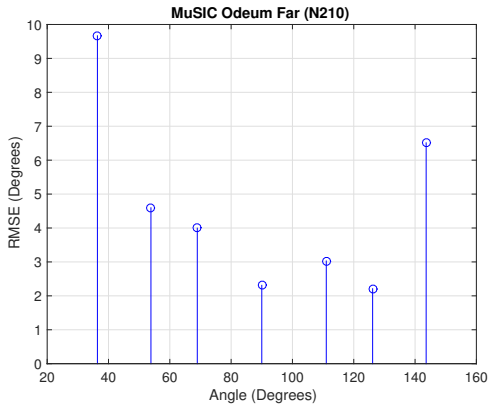


Figure 5.28: MUSIC RMSE across azimuth for initial large room test at 29 ft.

for phase coherent applications such as DoA, and have significant phase stability. Additionally, measurements were moved into a $3 \times 3 \times 3$ meter anechoic chamber. Such an environment provides near ideal testing for RF signals. Therefore, a more thorough testing scenario was used in the chamber itself but distances between target transmitter positions and the array would be much smaller. Figure 5.29 provides the relative positions of the array and transmitter for the chamber, which range from 17.4 to 142 degrees across azimuth. Since the TwinRXs have superior phase performance only a single phase estimation for radio calibration was made after the radios had been running for 10 minutes. This time allowed the RF components to warm-up and reach steady state. This single correction was used to correct the relative phase of the radios for each measurement made in the chamber. At each position 1024 measurements were taken with general MUSIC and Root-MUSIC for comparison. However, there is minimal variance between concurrent measurements as observed in Figure 5.30. There are slight reductions in RMSE compared with the original N210 setup in the large room. Nonetheless, no true increase in estimation performance was gained. The minimal variance shown in Figure 5.30 highlights that there is minimal effect from environmental noise, which is expected from simulations in Figure 5.10 and from previous testing.

To reduce the remaining error we re-evaluated the overall system design and testing methodology. First, we performed additional simulation based testing on the GNURadio

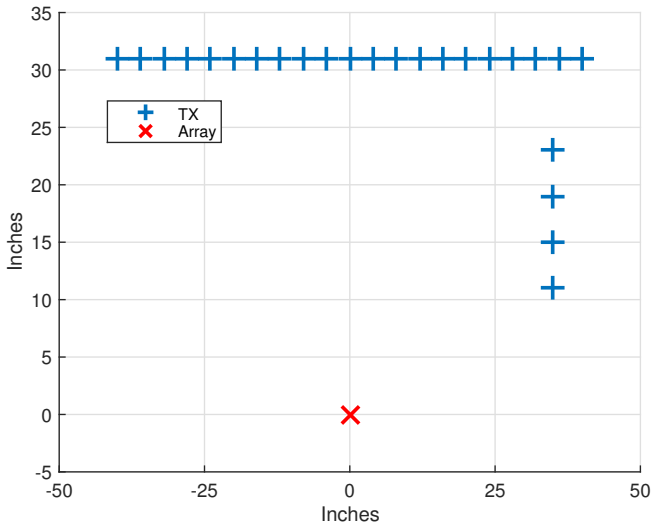


Figure 5.29: Relative locations of target transmitter test positions verses the four antenna array. The chamber itself is only 3×3 meters making space relatively cramped compared with the larger room.

blocks and simplified some of the flowgraph designs. However, from this testing we determined that the remaining error could only be a result of the physical setup. Therefore, we next evaluated the phase calibration of the TwinRX daughterboards. Over a twelve hour period the remained stable within 1 degree of each other, well below the error floor we are working at with these measurements.

Now if we consider the MUSIC pseudo-spectrum when taking measurements in the anechoic chamber, given by Figure 5.31. Here we are estimating a target with true position at 73 degrees, estimated to be at 78 degrees. However, the SNR of the pseudo-spectrum can be considered poor ranging from 4.5 to 5.5 dB. We expected better performance in the ideal setting of the anechoic chamber. Looking back at Figure 5.9 we know we can achieve superior pseudo-spectrum SNR even under high channel noise conditions. Therefore, we moved our attention to antenna phase and positional uncertainty. Nonetheless, a common method to reduce this uncertainty is the take a large number of measurements with variance in the placement. Unfortunately with the current system design this rigorous measurement campaign is impractical. Therefore alternative strategies were considered.

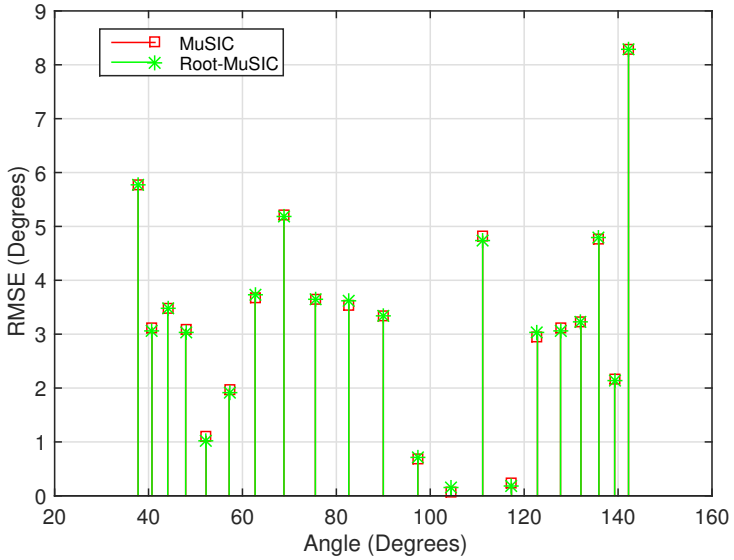


Figure 5.30: MUSIC and Root-MUSIC RMSE error across azimuth in anechoic chamber.

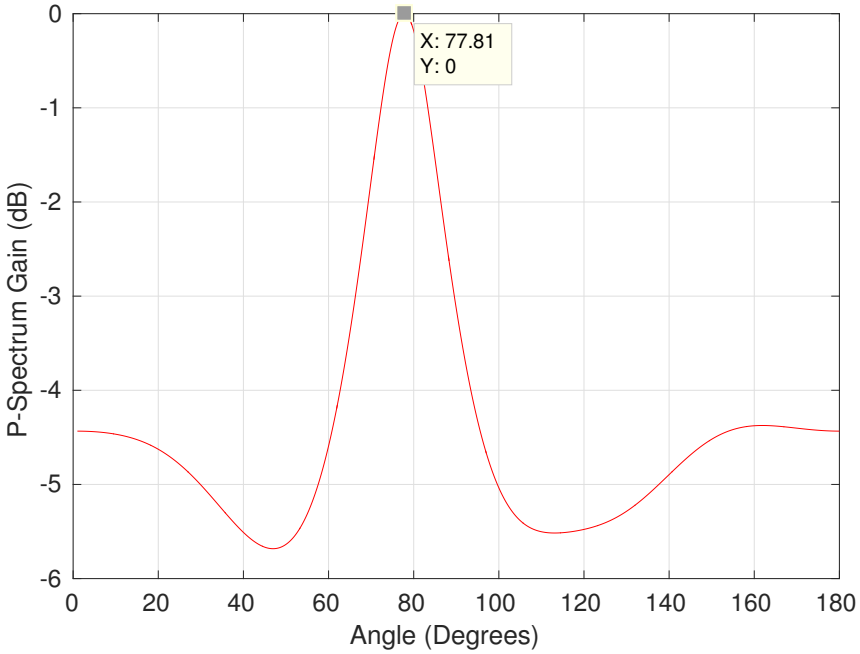


Figure 5.31: Pseudo-spectrum estimate of MUSIC algorithm for target placed at 73 degrees azimuth.

5.7.2 Antenna Calibration and Refining Spatial Measurements

After verification of the GNURadio software only portion of the implementation and TwinRX phase calibration, we moved on to consider the effects of the antennas themselves. Antenna gain, phase, and position mismatches or error can be modeled as a perturbation to the autocorrelation matrix R_{xx} in equation (5.9). The literature has extensively studied this model in [119, 139, 140] and a general framework for modeling an extensive set of non-idealities is provided here [141]. Now if we consider element spacing error alone, we can model this simply as:

$$\mathbf{x}(t) = \sum_{d=1}^{D} u_d(t)\hat{\mathbf{v}}(\mathbf{k}_d) + \mathbf{n}(t) \tag{5.32}$$

where:

$$\hat{\mathbf{v}}(\mathbf{k}_d) = \mathbf{v}(\mathbf{k}_d) + \Big[\mathbf{s}_1, \mathbf{s}_2, \ldots, \mathbf{s}_D\Big]^T \tag{5.33}$$

and $\mathbf{s}_k$ are zero mean gaussian vectors with variance σ_s. Figure 5.33 provides RMSE with respect to Gaussian spacing error at different variances. Although this error can cause estimation error, we believe we are within the acceptable range of 0.1λ due to our designed fixture shown in Figure 5.25. To increase accuracy of the target positioning with respect to the array, which can be modeled as array positioning errors, we utilize a simple pegboard. An example layout of the array and target is shown Figure 5.32 which has hole spacing of 1 in, and we can combine multiple of these boards with bolts to maintain spacing provides a large target positioning.

Although we are able to control the spacing precisely the same in true with the relative gains and phase of the individual antennas. We already perform calibration on the array itself to be phase aligned, but this is done at the base of the daughterboards and does not consider the attached antennas. Therefore, we implemented antenna calibration for random gains and phases into the system. To correct for these issues we utilize the technique from [140] and do not require more complex models such as the general model from [141]. However, we know from [119] that gain perturbations do not effect estimation performance but phase certainly does. We model the non-uniform antenna gains as $\Gamma = diag[1, \alpha_1 e^{-j\theta_1}, \ldots, \alpha_{N-1}^{-j\theta_{N-1}}]$ where α_k and θ_k are the gain and phase shifts for the k^{th}

Figure 5.32: Pegboard layout of transmitter in reference to four element array. Pegboard is used to maintain accurate target positioning across azimuth.

element respectively. Next, modifying the original model in equation (5.7) as:

$$\mathbf{R}_{xx} = \Gamma \mathbf{V}\, \mathbf{R}_{uu}\, \mathbf{V}^H \Gamma^H + I\sigma^2. \tag{5.34}$$

We determine the subspaces of the new signal and noise spaces through eigendecomposition of equation (5.34) and rewriting:

$$\mathbf{C}_{xx,eig} = \mathbf{E}_s \Lambda_s \mathbf{E}_s^H \Gamma^H + I\sigma^2 \mathbf{E}_N \mathbf{E}_N^H, \tag{5.35}$$

where $\mathbf{E}_s$ and $\mathbf{E}_N$ are the new signal and noise spaces respective. With this result we can utilize the properties of the signal space since $\mathbf{E}_s$ is unitary and write:

$$\mathbf{E}_s \mathbf{E}_s^H\, \Gamma \mathbf{V} = \Gamma \mathbf{V}. \tag{5.36}$$

Now if we place a transmitter at a known location resulting in a true manifold response $\mathbf{v}(\mathbf{k})$ we can solve for diagonal elements of $diag(\Gamma) = \gamma$ by the following manipulation:

$$\begin{aligned} \mathbf{E}_s \mathbf{E}_s^H\, \mathbf{V}_d \gamma &= \mathbf{V}_d \gamma \\ \mathbf{V}_d^H \mathbf{E}_s \mathbf{E}_s^H\, \mathbf{V}_d \gamma &= \gamma, \end{aligned} \tag{5.37}$$

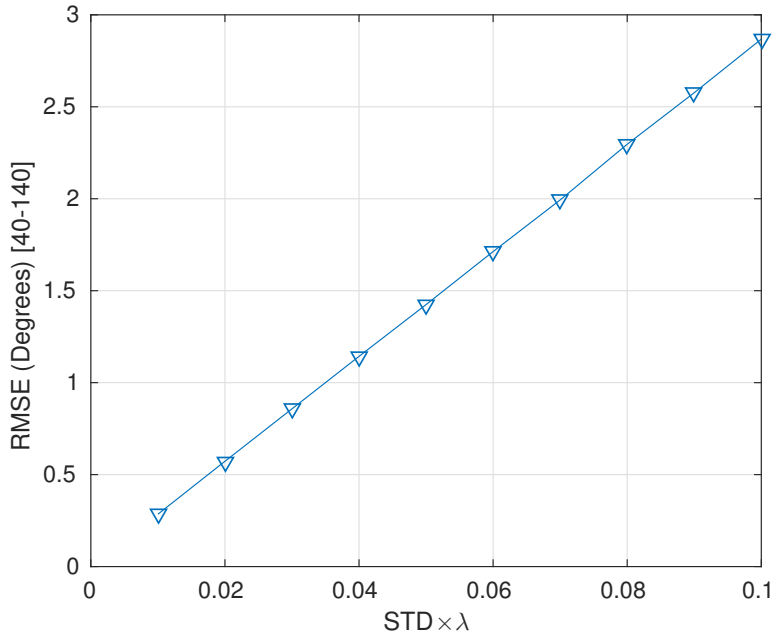


Figure 5.33: RMSE as a result of increasing element positioning noise.

where $\mathbf{V}_d = diag(\mathbf{v}(\mathbf{k}))$. To retrieve γ perform eigen-decomposition on $\mathbf{V}_d^H \mathbf{E}_s \mathbf{E}_s^H \mathbf{V}_d$, and the eigenvector that corresponds to a value of unity is the estimated antenna coefficient vector. This will typically be the N^{th} column of the decomposition if the decomposition is ordered.

Now we can use this estimate γ to correct the gains of the receive chains individually before autocorrelation or simply scale the result of autocorrelation by the outer product of γ with itself. In this work our implement precedes autocorrelation, allowing antenna calibration to be used with our EFD blocks. However, using a version of the block after autocorrelation can be more efficient since it will operate on far less data when L is large.

5.7.3 MUSIC Calibration and Testing

After the antenna calibration blocks were implemented in GNURadio and tested, we then proceeded to perform a comparative study on the performance of these blocks. Again we utilized the anechoic chamber and placed a target transmitter at the locations shown in

Figure 5.34. First DoA measurements were taken with MUSIC without calibration. Next, a transmitter was placed at 45 degrees and the antenna offset gains were estimated. Finally, utilizing these estimates we again made DoA measurements with MUSIC. In Figure 5.35

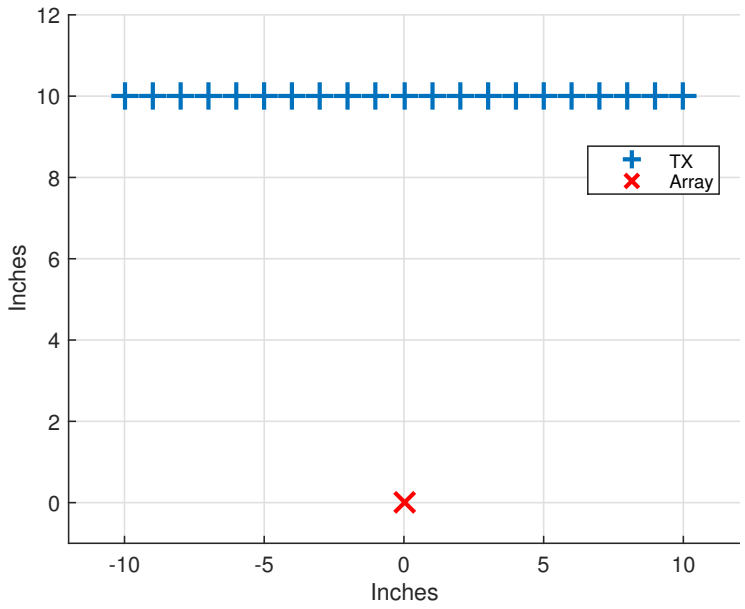


Figure 5.34: Transmitter positions in anechoic chamber relative to the receiving four element array.

we provide the results of these measurements. There are obvious DoA estimation accuracy improvements when utilizing these calibration tools. We observe roughly a 3 degree improve across azimuth in these results. In these results, there is no variance among the measurements over time, therefore we excluded error bars. The plot also include Root-MUSIC results which match MUSIC as expected. For reference we also provide the CRLB for SNRs of 20 and 0 dB, which we can roughly approach at 65, 102, and 115 degrees. To improve these result we could take additional calibration measurements using multiple targets or at different positions with the same target. These techniques have shown to improve DoA to a point [142]. However, at this point to get even better result would require expensive phase-matched cabling and custom antenna elements with specific azimuth performance. Such hardware is purpose built and is beyond practical implementations.

A final aspect to consider is the SNR performance of the pseudo-spectrum when cali-

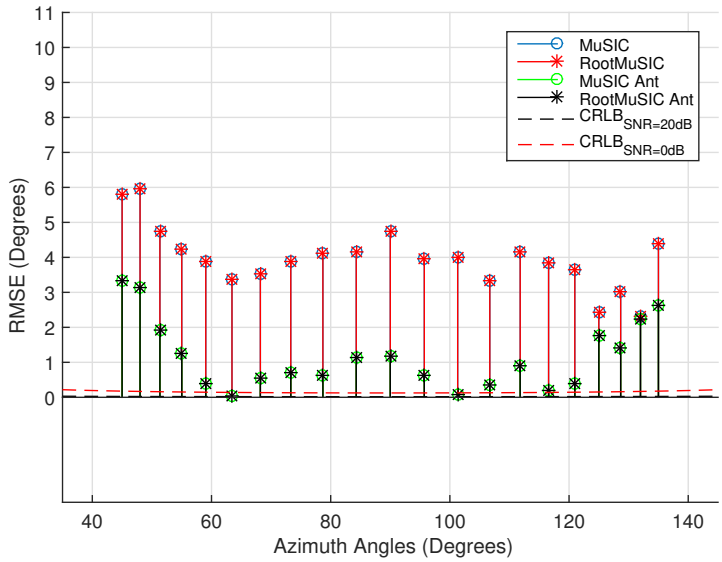


Figure 5.35: Comparison of calibrated and uncalibrated DoA estimates across azimuth.

brated. In Figure 5.36 we provide a pseudo-spectrum comparison again for the calibrated and uncalibrated cases. The SNR increased from between 2 and 5 dB across all angles, however positions below 90 degrees in azimuth generally had better SNR. This is likely due to the calibration target placed at the 45 degree position. Nonetheless, in terms of RMSE the improvement gained by calibration was almost equal across azimuth as seen in Figure 5.35. On a separate point, the pseudo-spectrum presented secondary false peaks at certain positions, pointing to possible reflections in the environment. We believe these are either reflections off the nearby radio devices themselves which have metal panels. Alternatively, they could be reflection off the pegboard which we suspect could act as a waveguide.

5.7.4 EADF-FFT DoA Implementation

After MUSIC was thoroughly testing and characterized, which provided evidence that relatively accurate estimates could be made, resources were shifted to the EFD implementation. Due to existing physical constraints of the EFD implementation over-the-air testing could not be performed in the anechoic chamber. During the initial design stages GNU-Radio simulation and cabling tests were performed to evaluate the OFDM link. Like the

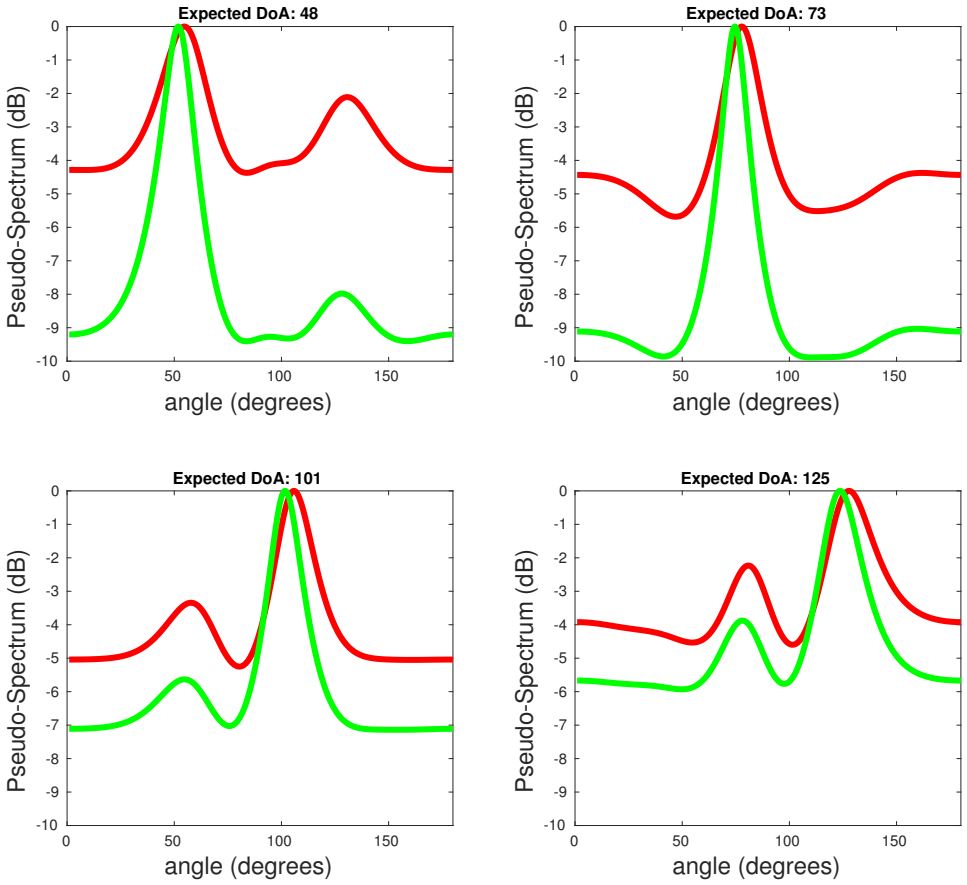


Figure 5.36: Pseudo-spectrum of MUSIC with calibrated and uncalibrated antennas.

final MUSIC system, EFD used X310 radios and the same array fixture at 2.45 GHz. These test were performed in a small conference room with relative array and transmitter spacing provided in Figure 5.37.

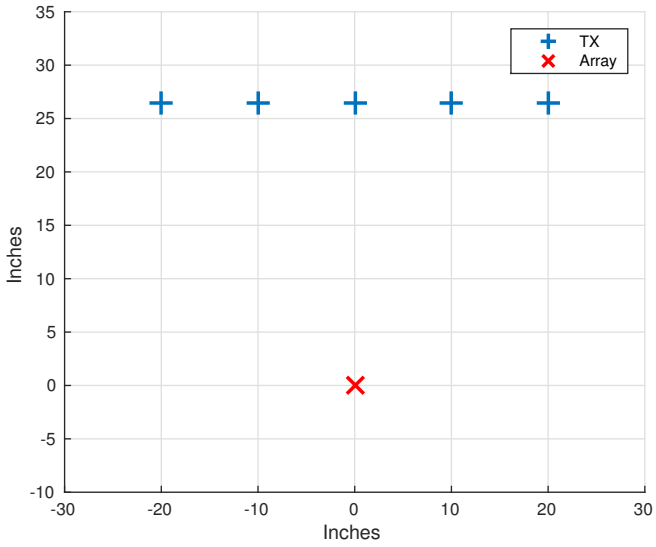


Figure 5.37: Relative locations of target transmitter test positions verses the four antenna array. The conference room itself is only 3×5 meters.

From the MUSIC implementation we were able to utilize the TwinRX calibration block and antenna calibration blocks. Together these should allow EFD to have similar estimation performance as MUSIC. For the tests themselves at each position 1000 packets were collected and Root-MUSIC estimation performed on the corrected input streams as well. MUSIC does not require knowledge of the signal itself, as a result we can directly estimate the source direction and compare to EFD. EFD estimation was implemented in within the flowgraph for online estimation, however due to computation time required for estimation compared to the received packet rate we downsampled the packets into the estimator. Operating the radios at 1 MS/s produced continuous packets at 200 per second, while our implementation of EFD can only withstand ~ 102 packets per second. However, due to the original model the packet rate would be greatly reduce since estimations would be only based on periodic packets. The estimation rate should only affect the system's ability to track users that are moving.

As presented in Section 5.5.1 the periodic antenna response $\mathbf{B}_p$ and resulting EADF $\mathbf{G}$ are a function of measurements taken of the antenna across azimuth and elevation. However, expensive robotic equipment is generally required to perform such measurements

since they are systematically take in space around the antenna themselves. Also, the fixture manipulating the target into various positions must limit their reflectivity. Due to these considerations we decided to consider simulation generated responses of the antenna elements. This approach seemed reasonable since the elements themselves have a simple geometry, have round responses in azimuth according to Figure 5.8, and our estimations are only in reference to azimuth. Therefore, in our EFD testing $\mathbf{G}$ matrices were synthetically generated, modeled simply as monopole antennas with height of 8 in or 2λ. Nonetheless, utilizing these synthetic estimates will inherently reduce our estimation performance.

Figure 5.38 provides results of our DoA measurements, and Figure 5.39 show a estimation of a single target over the collected 1000 packets. As we can observe from Figure 5.39 there is little to no variance among the estimates, and the EFD results match MUSIC closely. Across azimuth in Figure 5.38 EFD performs closely to MUSIC, with mean difference of ~ 0.5 degrees. As in the previous testing with MUSIC alone antenna calibration was performed with a target located at 45 degrees, which resulted in lower RMSE of targets closer to this position. To enhance these results further we have two obvious options: perform antenna calibration at multiple positions, and fully characterize the antenna elements for our EADF matrices. However, these are currently beyond the scope of this work.

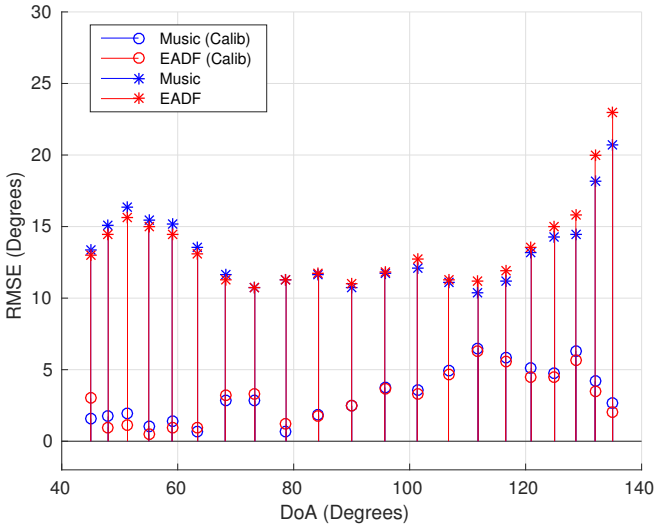


Figure 5.38: Estimation performance comparison between the baseline MUSIC implementation and the packet based EFD design. Calibration of the antennas were introduced in the second set of measurements with improved in both algorithms. All measurements were taken in the WiLab conference room.

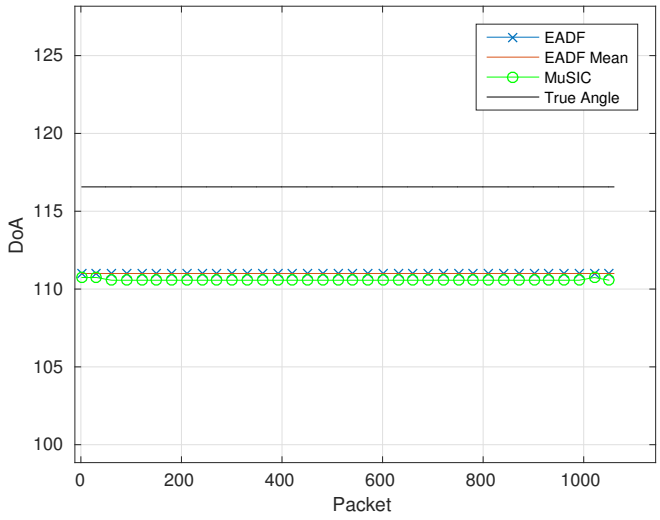


Figure 5.39: EFD and MUSIC measurements over time of a single position. Measurements show little to no variance for both EFD and MUSIC. These estimations are made for a target positioned at 116 degrees and have included antenna compensation in their software receive chains.

Now if we utilize these RMSE values we can determine the $2D$ localization performance of using multiple arrays when estimating a single target. First let us assume that we have two arrays at positions $[x_0, y_0]$ and $[x_1, y_1]$ with DoA RMSE R_0 and R_1 (in radians) respectively. Therefore, the distances and angles from a target at $[x_t, y_t]$ are $D_n = \sqrt{(x_n - x_t)^2 + (y_n - y_t)^2}$ and $\theta_n = tan^{-1}((y_n - y_t)/(x_n - x_t))$ where $n \in [0, 1]$. If we assume the area of error is rectangular for simplicity and similar to [143], then the RMSE $2D$ localization error becomes:

$$RMSE_{2D} = \sqrt{Z_0^2 + Z_1^2 - 2Z_0Z_1cos(\pi - \theta_0 - \theta_1)} \times \sqrt{Z_0^2 + Z_1^2 - 2Z_0Z_1cos(\theta_0 + \theta_1)} \tag{5.38}$$

where $Z_n = tan(R_n)\, D_n$. Based on this evaluation we can construct a simple example to understand the effect of DoA measurement error on localization performance. In Figure 5.40 we consider a scenario with a transmitter placed at $[0, D_n]$ along with two arrays at $[0, 0]$ and $[D_n, D_n]$. We can observe at close distances the localization error is relatively small, but can grow by many orders of magnitude as we place arrays at greater distances. This is an expected result of DoA based localization, and when constructing a system angular resolution should be chosen based on assumed range of targets.

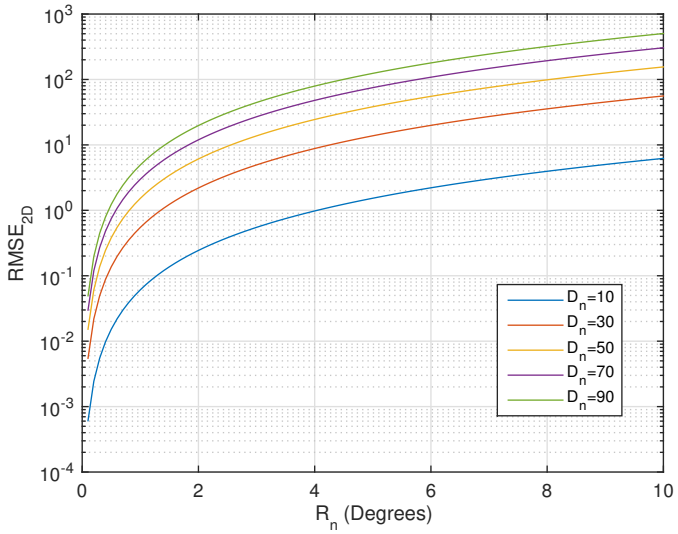


Figure 5.40: $2D$ localization performance of two arrays at varying distances and DoA estimation error.

5.8 Discussion and Retrospective

As in previous sections we provide a retrospective on the work performed and provide commentary for future researchers hoping to utilize or extend these project. Throughout this chapter we have tried to maintain consistent connections to the theoretical concepts governing the implementation and allow this theory to provide goals for system performance. However, reality of the implementation did not always match our models and continual extensions had to be considered. DoA is especially unique for implementations since it has heavy dependence on the physics of the system and operating environment, which is a rather unusual perspective for those coming from a pure communications perspective. However, when we did come across inconsistencies we found it best to isolate the problems. Therefore, we found it invaluable to have simulation only models and utilize cabling between radios in certain scenarios. This was particularly helpful when implementing the OFDM link, understanding the effects of antennas, and coping with general environmental inconsistencies.

Utilizing SDR hardware to implement any design can be challenging, generally resulting from aspects of the physical simulation that were not modeled correctly or at all in the synthetic simulations. By first performing a baseline study of DoA estimation through a robust and ubiquitous algorithm such as MUSIC we believe we saved a significant amount of time, and made our final implementation better. Even simplifying the eventual EFD SDR design. By having such a reference design we could easily check our alternative design, as well as provide respectable results to those unfamiliar with EFD. In the end, we believe this was one of our best decisions.

5.9 Chapter Summary

In this chapter, we implemented a spatial testbed for DoA finding for integration with an existing localization data-fusion framework. Two main DoA algorithms were implemented to provide both validation of the testbed and a proof of concept of the proposed infrastructure based DoA implementation. The first discussed algorithm MUSIC/Root-MUSIC provided a baseline validation for DoA estimates within the SDR system. Significant effort

was made designing a synchronization harness for both the USRP-N210 and USRP-X310 platforms to provide phase alignment between all receive chains. Additional calibration targeted at correcting antenna phase mismatch was implemented and comparative results are provided to show performance gains through their introduction. The combined correction algorithms, support hardware, and experimentation fixtures together reduced system measurement error, providing exact positioning of targets and array elements, and overall reduction in DoA estimation error.

Finally, a novel implementation of EADF modeling for DoA was built with respect to an OFDM communication link. Additional over-the-air testing was performed on the constructed spatial testbed with this EFD technique. The EFD system was able to effectively adopt the synchronization harness and antenna calibration algorithms implemented originally for MUSIC. Results showed similar performance to the MUSIC implementation when DoA estimations were performed across azimuth angles. Overall, we have developed an accurate DoA testbed with SDR hardware. The MUSIC implementation and associated synchronization harness have also been made available through a collaboration with Ettus Research.

Through each implementation step we have provided discussion on design decisions and alternative paths for extensions or compensating for additional implementation issues not specifically addressed in this work. Overall, this work shows how a practical array implementation can be constructed for accurate DoA estimation with connections to the underlying theoretical concepts.

Chapter 6

Conclusion and Future Work

In this dissertation, we have approached 5G network design and physical layer development from top down and bottom up approaches. We proposed a new computational framework to accelerate synthetic simulation, physical simulation, and general prototyping. Next, we targeted SDR hardware to demonstrate localization techniques for future dense FBS based networks. Providing a proof of concept and testbed for target tracking in real-world environments. Both the proposed framework and localization testbed are practical and compatible with the communication standards and implementation practices. Finally, we developed a new K-tier model for a macro-level perspective of network interference. Capturing FBS and MBS co-channel disruptions and an analysis on interference reduction through directional antennas.

6.1 Research Achievements

In this dissertation, several contributions have been made in the areas of SDR prototyping and stochastic network modeling for the purposes of enabling 5G. The research achievements of this dissertation are the following:

- **MATLAB DataFlow Framework for Algorithm Acceleration:** A framework that can provide significant acceleration for existing MATLAB code, specifically for SDR applications with streaming data. The framework's API is fully accessible

through the MATLAB and allows for pipelining and general concurrency of functions. Unlike existing tools within MATLAB, data dependent or time dependent data stream can be handled. This framework in general expands MATLAB to the multi-core world.

- **Stochastic HetNet Model for Multi-Tier Networks with Directional Antennas:** A new probabilistic model set for cellular networks is analyzed and verified through numerical simulation. These models approach multi-tied networks from a closed subscriber view with independent tiers. Since co-channel interference is assumed between tiers, coverage is significantly reduce as tier densities diverge. To reduce this interference and meet coverage targets, directional antennas are introduced. Their associated beam-pattern shapes are explored for macrocells explicitly, and effectiveness relative tier densities are analyzed.

- **Real-Time Localization Implementation for Future Dense Networks:** A comparative study of direction finding algorithms MuSIC and EFD for SDR implementations. This is the first SDR implementation of EFD and was specifically adapted for used with a OFDM link, which was also implemented. To enable these implementations a phased array platform was build from USRP-N210 and USRP-X310 radios with phase aligned reception.

6.2 Future Works

The future work of this PhD research can be classified in extensions based on the above achievements. First, the DataFlow framework can be expanded to incorporate heterogeneous compute units such as GPUs, FPGAs, and other accelerators. Additional extensions include analysis tools of algorithm design for use with the proposed framework would be invaluable for those wishing to implement their work into dataflow. However, other enhancements should include further debugging tools such as scopes and general visualizations.

Second, for stochastic network modeling of *mmWave* propagation scenarios need to be included. In the SG literature *mmWave* communication are a relatively new modeling area

and fundamental knowledges still needs to be captured. An alternative area of focus is overlapping channel allocations which utilize spatial reuse to provide effective overlapping channel design. Through initial simulations we have observed minimal co-channel interference and increased capacity gains with overlapping channels designs with spatially separated users.

Third, for the localization prototyping there a several remaining tasks. The tracking model needs to be integrated into the current measurement system. Targets themselves need to be enhanced with robotic movement to fully evaluate performance of tracking models. The current testbed itself is modular and can support additional arrays given the necessary hardware. However, GNURadio designs would benefit from conversion into RFNoC for higher bandwidth reception.

Appendix A:

Proof of Lemma 1

Based on the definition for probability of success (coverage) we adopted in Section 4.3 and following the approach used in [69], we can write:

$$
\begin{aligned}
p_s(\gamma) &= \sum_{i=1}^{\infty} \mathbb{P}(SIR_o > \gamma, B_o = i) \\
&= \sum_{i=1}^{\infty} \mathbb{P}\left(\frac{g_1 h_i l(|x_i|)}{\sum_{k=1}^{K} \sum_{x_j \in \Phi \setminus \{x_{i_{k=c}}\}} P_k G_{T,k} h_j l(|x_j|)} > \gamma, B_o = i \right) \times \mathbf{1}_{\{|x_i|<|x_j|, j \in \mathbb{N}\}} \Bigg] \\
&\overset{(a)}{=} \sum_{i=1}^{\infty} \mathbb{E}_{\Phi, G_T} \left[\mathbb{E}_{h_j} [e^{-\mu s_{x_i} \sum_{k=1}^{K} \sum_{x_j \in \Phi \setminus \{x_{i_{k=c}}\}} G_{T,k} h_j l(|x_j|)} \mid \Phi] \times \mathbf{1}_{\{|x_i|<|x_j|, j \in \mathbb{N}\}} \right] \\
&\overset{(b)}{=} \sum_{i=1}^{\infty} \mathbb{E}_{\Phi, G_T} \left[\prod_{k=1}^{K} \prod_{j \in \mathbb{N} \setminus \{i_{k=c}\}} \mathbb{E}_{h_j} [e^{-\mu s_{x_i} P_k h_j G_{T,k} l(|x_j|)} \mid \Phi] \times \mathbf{1}_{\{|x_i|<|x_j|, j \in \mathbb{N}\}} \right] \\
&\overset{(c)}{=} \sum_{i=1}^{\infty} \mathbb{E}_{\Phi, G_T} \left[\prod_{k=1}^{K} \prod_{j \in \mathbb{N} \setminus \{i_{k=c}\}} \mathcal{L}_{h_j} \left(\mu s_{x_i} P_k G_{T,k} l(|x_j|) \mid \Phi \right) \times \mathbf{1}_{\{|x_i|<|x_j|, j \in \mathbb{N}\}} \right] \\
&\overset{(d)}{=} \sum_{i=1}^{\infty} \mathbb{E}_{\Phi, G_T} \left[\prod_{k=1}^{K} \prod_{j \in \mathbb{N} \setminus \{i_{k=c}\}} \left(1 + s_{x_i} P_k G_{T,k} l(|x_j|) \right)^{-1} \times \mathbf{1}_{\{|x_i|<|x_j|, j \in \mathbb{N}\}} \right]
\end{aligned}
$$

In equality (a), $s_{x_i} := \gamma/(g_1\, l(|x_i|))$ and we utilized the c.c.d.f of h_i. By identifying that the expectation in (a) is over the independent variables, Φ, G_T and $\{h_j\}$, we first obtain (b) and arrive at (c) by applying the definition of Laplace transform for random variables. Finally, due to our Rayleigh fading assumption, $\mathcal{L}_{h_j}(s)$ will be of the form $\frac{\mu}{\mu+s}$ and hence, we get (d). Substituting the power-law formula gives us (4.12).

Appendix B:

Proof of Lemma 2

Based on the result of Lemma 1 and following the approach used in [103], we first write (4.12) as follows:

$$p_s(\gamma) = \mathbb{E}_r \left[\prod_{k=1}^{K} \mathbb{E}_{G_{T,k},\Phi|r} \left[\prod_{j \in \mathbb{N}\setminus\{i_{k=c}\}} \left(1 + \gamma P_k \frac{G_{T,k}}{g_1} \left| \frac{X_{B_0}}{X_j} \right|^{2\beta} \right)^{-1} \right] \right].$$

Now, we evaluate the inner expectation w.r.t the PPP by conditioning on the nearest transmitter being at a distance $r := |X_{B_0}|$ from the typical receiver. That is:

$$\begin{aligned} S_k &= \mathbb{E}_{G_{T,k},\Phi|r} \left[\prod_{j \in \mathbb{N}\setminus\{i_{k=c}\}} \left(1 + \gamma P_k \frac{G_{T,k}}{g_1} \left| \frac{X_{B_0}}{X_j} \right|^{2\beta} \right)^{-1} \right] \\ &\overset{(a)}{=} \exp\left[\frac{-2\pi\lambda_k}{\lambda_c} \int_{\in g} \int_{r\times\mathbb{1}_{\{k=c\}}}^{\infty} f_{G_{T,k}}(g) \left(1 - \left[1 + \gamma \frac{g}{g_1} \left(\frac{r}{v} \right)^{2\beta} \right]^{-1} \right) v\, dv\, dg \right] \\ &= \exp\left[\frac{-2\pi\lambda_k}{\lambda_c} \int_{\in g} \int_{r\times\mathbb{1}_{\{k=c\}}}^{\infty} f_{G_{T,k}}(g) \gamma \left(\gamma + \frac{g_1}{gP_k} \left(\frac{v}{r} \right)^{2\beta} \right)^{-1} v dv dg \right] \\ &\overset{(b)}{=} \exp\left[\frac{-\pi\lambda_k r^2 \gamma^{1/\beta}}{\lambda_c} \int_{\in g} \int_{\gamma^{-1/\beta}\times\mathbb{1}_{\{k=c\}}}^{\infty} f_{G_{t,k}}(g) \left(1 + \frac{g_1}{gP_k} u^{\beta} \right)^{-1} du dg \right]. \end{aligned}$$

Here, (a) we applied the PGFL for PPPs [102] and apply the definition of expection for the antenna pattern $G_{T,k}$. In (b) performed a change of variables by substituting $u \to v^2/(r^2\gamma^{1/\beta})$. Now we employ the p.d.f. of the distance between a typical receiver and its paired transmitter [103, Section III.A].

$$M(r) = 2\pi\lambda_c r e^{-\lambda_c \pi r^2}$$

Appendix C:

Proof of Theorem 1

Based on the result of Lemma 1 and following the approach used in [69], we first utilize the conditional independence of $\mathbf{1}_{\{|x_i|<|x_j|,\, j\in\mathbb{N}\setminus\{i\}\}}$ write (4.12) as follows:

$$p_s(\gamma) = \sum_{i=1}^{\infty} \mathbb{E}_{\Phi,G_T}\left[\prod_{k=1}^{K}\prod_{j\in\mathbb{N}\setminus\{i_{k=c}\}}\left(1+s_{x_i}P_k G_{T,k} l(|x_j|)\right)^{-1} \times \mathbf{1}_{\{|x_i|<|x_j|,\, j\in\mathbb{N}\}}\right]$$

$$\overset{(a)}{=} \sum_{i=1}^{\infty} E_{G_T,\Phi}\left[\prod_{k=1}^{K}\prod_{j\in\mathbb{N}\setminus\{i_{k=c}\}}\left(1+\gamma P_k \frac{G_{T,k}}{g_1}\left(\frac{Y_i}{Y_j}\right)^{\beta}\right)^{-1} \times \mathbf{1}_{\{|Y_i|<|Y_j|,\, j\in\mathbb{N}\setminus\{i\}\}}\right]$$

$$\overset{(b)}{=} \sum_{i=1}^{\infty}\int_0^{\infty} \frac{\lambda_c^i u^{i-1} e^{-\lambda_c u}}{(i-1)!}\prod_{k=1}^{K}\prod_{j\in\mathbb{N}\setminus\{i_{k=c}\}}\int_{\in g}\int_{u\times\mathbb{1}_{\{k=c\}}}^{\infty}$$

$$\times \frac{\lambda_k^j y^{j-1} e^{-\lambda_k y}}{(j-1)!} f_{G_{T,k}}(g)\left(1+\gamma P_k \frac{g}{g_1}\left(\frac{u}{y}\right)^{\beta}\right)^{-1} dy\, dg\, du$$

In (a), we used the fact that the set, $\{|X_i|\}_{i\in\mathbb{N}}$ will have the same distribution as $\{\sqrt{Y_i}\}_{i\in\mathbb{N}}$, where $Y_i \sim \text{Gamma}(i,\lambda)$ (for $1 \le i \le N$) [69]. By employing the p.d.f. of and the mutual independence among $\{Y_i\}$, we get (b). Again we used the definition of expectation for the antenna gain pattern $G_{T,k}$. In the Gamma distribution p.d.f. substituted in (b), c is a *rate* term which affects the spread of the distribution whereas, u and y are the *shape* terms [144]. After some algebraic manipulations and by performing a change of variables ($s_k \to \lambda_k y$ and $r_k \to \lambda_k u$), we arrive at Theorem 1.

Appendix D:

Timed Commands Flowgraph

```
#!/usr/bin/env python2
# -*- coding: utf-8 -*-
##################################################
# GNU Radio Python Flow Graph
# Title: Top Block
# Generated: Mon Sep 26 17:17:40 2016
##################################################

if __name__ == '__main__':
    import ctypes
    import sys
    if sys.platform.startswith('linux'):
        try:
            x11 = ctypes.cdll.LoadLibrary('libX11.so')
            x11.XInitThreads()
        except:
            print "Warning: failed to XInitThreads()"

from PyQt4 import Qt
from gnuradio import blocks
from gnuradio import eng_notation
from gnuradio import gr
from gnuradio import uhd
```

```
from gnuradio.eng_option import eng_option
from gnuradio.filter import firdes
from optparse import OptionParser
import sys
import time


class top_block(gr.top_block, Qt.QWidget):

    def __init__(self):
        gr.top_block.__init__(self, "Top Block")
        Qt.QWidget.__init__(self)
        self.setWindowTitle("Top Block")
        try:
            self.setWindowIcon(Qt.QIcon.fromTheme('gnuradio-grc'))
        except:
            pass
        self.top_scroll_layout = Qt.QVBoxLayout()
        self.setLayout(self.top_scroll_layout)
        self.top_scroll = Qt.QScrollArea()
        self.top_scroll.setFrameStyle(Qt.QFrame.NoFrame)
        self.top_scroll_layout.addWidget(self.top_scroll)
        self.top_scroll.setWidgetResizable(True)
        self.top_widget = Qt.QWidget()
        self.top_scroll.setWidget(self.top_widget)
        self.top_layout = Qt.QVBoxLayout(self.top_widget)
        self.top_grid_layout = Qt.QGridLayout()
        self.top_layout.addLayout(self.top_grid_layout)

        self.settings = Qt.QSettings("GNU Radio", "top_block")
        self.restoreGeometry(self.settings.value("geometry").toByteArray())

        ##################################################
        # Variables
        ##################################################
        self.samp_rate = samp_rate = 1000000
        self.center_freq = center_freq = 2.45e9
```

```
##################################################
# Blocks
##################################################
self.uhd_usrp_source_0 = uhd.usrp_source(
    ",".join(("addr0=192.168.10.2,addr1=192.168.20.2", "")),
    uhd.stream_args(
        cpu_format="fc32",
        channels=range(2),
    ),
)
self.uhd_usrp_source_0.set_clock_source("external", 0)
self.uhd_usrp_source_0.set_time_source("external", 0)
self.uhd_usrp_source_0.set_clock_source("external", 1)
self.uhd_usrp_source_0.set_time_source("external", 1)
self.uhd_usrp_source_0.set_time_unknown_pps(uhd.time_spec())
self.uhd_usrp_source_0.set_samp_rate(samp_rate)
self.uhd_usrp_source_0.set_gain(10, 0)
self.uhd_usrp_source_0.set_gain(0, 1)

now = self.uhd_usrp_source_0.get_time_now()
command_time = now + uhd.time_spec(0.5)
self.uhd_usrp_source_0.set_command_time(command_time)
self.uhd_usrp_source_0.set_center_freq(uhd.tune_request(
  self.center_freq, dsp_freq=0,
  dsp_freq_policy=uhd.tune_request.POLICY_MANUAL), 0)
self.uhd_usrp_source_0.set_center_freq(uhd.tune_request(
  self.center_freq, dsp_freq=0,
  dsp_freq_policy=uhd.tune_request.POLICY_MANUAL), 1)
self.uhd_usrp_source_0.clear_command_time()


self.blocks_null_sink_0 = blocks.null_sink(gr.sizeof_gr_complex*1)

##################################################
# Connections
##################################################
```

```
        self.connect((self.uhd_usrp_source_0, 0), (self.blocks_null_sink_0, 0))
        self.connect((self.uhd_usrp_source_0, 1), (self.blocks_null_sink_0, 1))

    def closeEvent(self, event):
        self.settings = Qt.QSettings("GNU Radio", "top_block")
        self.settings.setValue("geometry", self.saveGeometry())
        event.accept()


    def get_samp_rate(self):
        return self.samp_rate

    def set_samp_rate(self, samp_rate):
        self.samp_rate = samp_rate
        self.uhd_usrp_source_0.set_samp_rate(self.samp_rate)

    def get_center_freq(self):
        return self.center_freq

    def set_center_freq(self, center_freq):
        self.center_freq = center_freq
        self.uhd_usrp_source_0.set_center_freq(self.center_freq, 0)
        self.uhd_usrp_source_0.set_center_freq(self.center_freq, 1)


def main(top_block_cls=top_block, options=None):

    from distutils.version import StrictVersion
    if StrictVersion(Qt.qVersion()) >= StrictVersion("4.5.0"):
        style = gr.prefs().get_string('qtgui', 'style', 'raster')
        Qt.QApplication.setGraphicsSystem(style)
    qapp = Qt.QApplication(sys.argv)

    tb = top_block_cls()
    tb.start()
    tb.show()
```

```
    def quitting():
        tb.stop()
        tb.wait()
    qapp.connect(qapp, Qt.SIGNAL("aboutToQuit()"), quitting)
    qapp.exec_()


if __name__ == '__main__':
    main()
```

Bibliography

[1] A. Agarwal and K. Agarwal, "The next generation mobile wireless cellular networks–4g and beyond," *American Journal of Electrical and Electronic Engineering*, vol. 2, no. 3, pp. 92–97, 2014.

[2] I. Vision, "Framework and overall objectives of the future development of imt for 2020 and beyond," *Working document toward preliminary draft new recommendation ITU-R M.[IMT. Vision]*, 2014.

[3] Cisco, "The Zettabyte EraTrends and Analysis," jun 2016. [Online]. Available: http://www.cisco.com/c/en/us/solutions/collateral/service-provider/visual-networking-index-vni/vni-hyperconnectivity-wp.html

[4] M. Dohler, T. Nakamura, A. Osseiran, J. F. Monserrat, O. Queseth, and P. Marsch, *5G Mobile and Wireless Communications Technology.* Cambridge University Press, 2016.

[5] Ericsson, "Ericsson Mobility Report, Report No. EAB-15:037849," Nov 2015. [Online]. Available: www.ericsson.com/res/docs/2015/mobility-report/ericsson-mobility-report-nov-2015.pdf

[6] S. Shankland, "How 5g will push a supercharged network to your phone, home, car," mar 2015. [Online]. Available: https://cnet1.cbsistatic.com/img/YP32JeSOmsDJWWa3PySeEYq11ZU=/970x0/2015/02/27/f6842a3c-cc22-4314-830d-de895a5e572e/5g-data-transfer-speed-graphic.jpg

[7] C. Sharma, "Correcting the iot history," Nov 2016. [Online]. Available: http://www.chetansharma.com/IoT_History.htm

[8] S. Mumtaz, K. M. S. Huq, M. I. Ashraf, J. Rodriguez, V. Monteiro, and C. Politis, "Cognitive vehicular communication for 5g," *IEEE Communications Magazine*, vol. 53, no. 7, pp. 109–117, July 2015.

[9] Y. Wang, A. Ahmed, B. Krishnamachari, and K. Psounis, "Ieee 802.11p performance evaluation and protocol enhancement," in *2008 IEEE International Conference on Vehicular Electronics and Safety*, Sept 2008, pp. 317–322.

[10] D. Talbot, "A Tiny Cell-Phone Transmitter Takes Root in Rural Africa," may 2013. [Online]. Available: https://www.technologyreview.com/s/515346/a-tiny-cell-phone-transmitter-takes-root-in-rural-africa/

[11] S. Cass. (2015, Jul.) The 2015 Top Ten Programming Languages. [Online]. Available: http://spectrum.ieee.org/computing/software/the-2015-top-ten-programming-languages

[12] NTIA, "United states frequency allocation chart," jan 2016. [Online]. Available: https://www.ntia.doc.gov/files/ntia/publications/4b_14_5-1.pdf

[13] D. Duffy, "98networks," Dec 2012. [Online]. Available: http://www.smallcellforum.org/press-releases/98-mobile-operators-say-small-cells-essential-future-networks/

[14] M. DeGrasse, "Carrier small cells appear slowly but surely," May 2016. [Online]. Available: http://www.rcrwireless.com/20160524/carriers/carrier-small-cells-tag4

[15] T. Mendelsohn, "Millimeter-wave 5g modem coming mid-2018 with 5gbps peak download," oct 2016. [Online]. Available: http://arstechnica.com/business/2016/10/qualcomm-5g-x50-modem-millimetre-wave-5g-modem/

[16] A. M. Wyglinski and D. Pu, *Digital Communication Systems Engineering with Software-Defined Radio (Mobile Communications)*. Artech House, 2013.

[17] P. Mannion, "Change and Cost: 2016s Test Challenges and Opportunities," jan 2016. [Online]. Available: http://electronicdesign.com/blog/change-and-cost-2016-s-test-challenges-and-opportunities

[18] K. technologies, "W1910 LTE Baseband Veriication Library," jul 2014. [Online]. Available: http://literature.cdn.keysight.com/litweb/pdf/5990-4283EN.pdf?id=1748019

[19] GNU Radio Website, accessed April 2014. [Online]. Available: http://www.gnuradio.org

[20] R. I. Lackey and D. W. Upmal, "Speakeasy: the military software radio," *IEEE Communications Magazine*, vol. 33, no. 5, pp. 56–61, May 1995.

[21] M. Inc., "LTE Receiver using Zynq-based Software-Defined Radio (SDR)," oct 2016. [Online]. Available: http://www.mathworks.com/help/examples/lte_product/sdr_receive_diagram_published.png

[22] "Us government will invest 400 million in 5g research," jul 2016. [Online]. Available: http://www.businessinsider.com/us-government-will-invest-400-million-in-5g-research-2016-7

[23] S. Riaz, "Uk to invest 1b-plus in 5g and fibre," nov 2016. [Online]. Available: http://www.mobileworldlive.com/featured-content/top-three/uk-to-invest-740m-in-5g-trials/

[24] A. Khandekar, N. Bhushan, J. Tingfang, and V. Vanghi, "Lte-advanced: Heterogeneous networks," in *Wireless Conference (EW), 2010 European*, April 2010, pp. 978–982.

[25] T. S. Rappaport, F. Gutierrez, E. Ben-Dor, J. N. Murdock, Y. Qiao, and J. I. Tamir, "Broadband millimeter-wave propagation measurements and models using adaptive-beam antennas for outdoor urban cellular communications," *IEEE Transactions on Antennas and Propagation*, vol. 61, no. 4, pp. 1850–1859, April 2013.

[26] A. S. Hamza, S. S. Khalifa, H. S. Hamza, and K. Elsayed, "A survey on inter-cell interference coordination techniques in ofdma-based cellular networks," *IEEE Communications Surveys & Tutorials*, vol. 15, no. 4, pp. 1642–1670, 2013.

[27] T. F. Collins and A. M. Wyglinski, "Co-channel interference in future femtocell networks," in *Vehicular Technology Conference (VTC Fall), 2015 IEEE 82nd*, Sept 2015.

[28] M. Do and H. Son, "Interference Coordination in LTE/LTE-A (2): eICIC (enhanced ICIC)." [Online]. Available: http://www.netmanias.com/en/?m=view&id=blog&no=6551&xtag=lte-lte-a-eicic&xref=interference-coordination-in-lte-lte-a-2-eicic-enhanced-icic

[29] M. Inc., "WLAN System Toolbox," oct 2016. [Online]. Available: https://www.mathworks.com/products/wlan-system/

[30] ——, "LTE System Toolbox," oct 2016. [Online]. Available: https://www.mathworks.com/products/lte-system/

[31] J. Malsbury and M. Ettus, "Simplifying fpga design with a novel network-on-chip architecture," in *Proceedings of the Second Workshop on Software Radio Implementation Forum*, ser. SRIF '13. New York, NY, USA: ACM, 2013, pp. 45–52. [Online]. Available: http://doi.acm.org/10.1145/2491246.2491251

[32] Ettus Research A National Instruments Company. (2016, Jun.) USRP X300 and X310 X Series. [Online]. Available: https://www.ettus.com/content/files/X300_X310_Spec_Sheet.pdf

[33] L. Microsystems, "LimeSDR: Flexible, Next-generation, Open Source Software Defined Radio." [Online]. Available: https://www.crowdsupply.com/lime-micro/limesdr

[34] R. G. Machado and A. M. Wyglinski, "Software-defined radio: Bridging the analog digital divide," *Proceedings of the IEEE*, vol. 103, no. 3, pp. 409–423, March 2015.

[35] T. Korakis, M. Knox, E. Erkip, and S. Panwar, "Cooperative network implementation using open-source platforms," *IEEE Communications Magazine*, vol. 47, no. 2, pp. 134–141, February 2009.

[36] G. Conte, S. Tommesani, and F. Zanichelli, "The long and winding road to high-performance image processing with mmx/sse," in *Computer Architectures for Machine Perception, 2000. Proceedings. Fifth IEEE International Workshop on*, 2000, pp. 302–310.

[37] S. Chapman, *MATLAB Programming for Engineers*. Cengage Learning, 2007. [Online]. Available: https://books.google.com/books?id=fhpotPvv7v8C

[38] R Development Core Team, *R: A Language and Environment for Statistical Computing*, R Foundation for Statistical Computing, Vienna, Austria, 2008, ISBN 3-900051-07-0. [Online]. Available: http://www.R-project.org

[39] J. W. Eaton, D. Bateman, S. Hauberg, and R. Wehbring, *GNU Octave version 4.0.0 manual: a high-level interactive language for numerical computations*, 2015. [Online]. Available: http://www.gnu.org/software/octave/doc/interpreter

[40] G. Rossum, "Python reference manual," Amsterdam, The Netherlands, The Netherlands, Tech. Rep., 1995.

[41] E. Jones, T. Oliphant, P. Peterson *et al.*, "SciPy: Open source scientific tools for Python," 2001–, [Online; accessed 2016-08-08]. [Online]. Available: "http://www.scipy.org/"

[42] J. J. Dongarra, J. Du Croz, S. Hammarling, and I. S. Duff, "A set of level 3 basic linear algebra subprograms," *ACM Trans. Math. Softw.*, vol. 16, no. 1, pp. 1–17, Mar. 1990. [Online]. Available: http://doi.acm.org/10.1145/77626.79170

[43] L. Dagum and R. Menon, "Openmp: an industry standard api for shared-memory programming," *Computational Science & Engineering, IEEE*, vol. 5, no. 1, pp. 46–55, 1998.

[44] Z. Xianyi, W. Qian, and Z. Yunquan, "Model-driven level 3 blas performance optimization on loongson 3a processor," in *Parallel and Distributed Systems (ICPADS), 2012 IEEE 18th International Conference on*, Dec 2012, pp. 684–691.

[45] M. Abadi *et al.*, "TensorFlow: Large-scale machine learning on heterogeneous systems," 2015, software available from tensorflow.org. [Online]. Available: http://tensorflow.org/

[46] A. Woodie, "Apache Beam's Ambitious Goal Unify Big Data Development," Apr. 2016. [Online]. Available: https://www.datanami.com/2016/04/22/apache-beam-emerges-ambitious-goal-unify-big-data-development/

[47] T. Uustalu and V. Vene, *The Essence of Dataflow Programming*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2006, pp. 135–167. [Online]. Available: http://dx.doi.org/10.1007/11894100_5

[48] Z. OrBach, "FPGAs as ASIC Alternatives Past and Future," Apr. 2014. [Online]. Available: http://www.eetimes.com/author.asp?doc_id=1322021

[49] N. T. Bliss and J. Kepner, "'pmatlab parallel matlab library'," *International Journal of High Performance Computing Applications*, vol. 21, no. 3, pp. 336–359, 2007.

[50] W. Gao, Q. Kemao, H. Wang, F. Lin, and H. S. Seah, "Parallel computing for fringe pattern processing: A multicore {CPU} approach in matlab environment," *Optics and Lasers in Engineering*, vol. 47, no. 11, pp. 1286 – 1292, 2009. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0143816609001079

[51] J. a. Bispo, L. Reis, and J. a. M. P. Cardoso, "Multi-target c code generation from matlab," in *Proceedings of ACM SIGPLAN International Workshop on Libraries, Languages, and Compilers for Array Programming*, ser. ARRAY'14. New York, NY, USA: ACM, 2014, pp. 95:95–95:100. [Online]. Available: http://doi.acm.org/10.1145/2627373.2627389

[52] A. Prasad, J. Anantpur, and R. Govindarajan, "Automatic compilation of matlab programs for synergistic execution on heterogeneous processors,"

SIGPLAN Not., vol. 46, no. 6, pp. 152–163, Jun. 2011. [Online]. Available: http://doi.acm.org/10.1145/1993316.1993517

[53] J. Spazier, S. Christgau, and B. Schnor, "Automatic generation of parallel c code for stencil applications written in matlab," in *Proceedings of the 3rd ACM SIGPLAN International Workshop on Libraries, Languages, and Compilers for Array Programming*, ser. ARRAY 2016. New York, NY, USA: ACM, 2016, pp. 47–54. [Online]. Available: http://doi.acm.org/10.1145/2935323.2935329

[54] P. Ratnalikar and A. Chauhan, "Automatic parallelism through macro dataflow in high-level array languages," in *Proceedings of the 23rd International Conference on Parallel Architectures and Compilation*, ser. PACT '14. New York, NY, USA: ACM, 2014, pp. 489–490. [Online]. Available: http://doi.acm.org/10.1145/2628071.2628131

[55] G. R. Faulhaber, "The Future of Wireless Telecommunications: Spectrum as a Critical Resource," *Information Economics and Policy*, vol. 18, no. 3, pp. 256 – 271, 2006.

[56] J. G. Andrews, R. K. Ganti, M. Haenggi, N. Jindal, and S. Weber, "A Primer on Spatial Modeling and Analysis in Wireless Networks," *IEEE Communications Magazine*, vol. 48, no. 11, pp. 156–163, 2010.

[57] E. N. Gilbert, "Random plane networks," *Journal of the Society for Industrial and Applied Mathematics*, vol. 9, no. 4, pp. 533–543, 1961.

[58] F. Baccelli and B. Blaszczyszyn, *Stochastic Geometry and Wireless Networks, Volume I - Theory*, 2009, vol. I.

[59] S. Weber and J. G. Andrews, "Transmission capacity of wireless networks," *arXiv preprint arXiv:1201.0662*, 2012.

[60] M. Haenggi and R. K. Ganti, *Interference in Large Wireless Networks*, 2008, vol. 3, no. 2.

[61] O. Georgiou, S. Wang, M. Z. Bocus, C. P. Dettmann, and J. P. Coon, "Directional antennas improve the link-connectivity of interference limited ad hoc networks," *ArXiv e-prints*, Sep. 2015.

[62] J. Wildman, P. H. J. Nardelli, M. Latva-aho, and S. Weber, "On the Joint Impact of Beamwidth and Orientation Error on Throughput in Directional Wireless Poisson Networks," *IEEE Transactions on Wireless Communications*, vol. 13, no. 12, pp. 7072–7085, Dec 2014.

[63] H. Wang and M. C. Reed, "Tractable Model for Heterogeneous Cellular Networks with Directional Antennas," in *Communications Theory Workshop (AusCTW), 2012 Australian*, Jan 2012, pp. 61–65.

[64] T. D. Novlan, R. K. Ganti, A. Ghosh, and J. G. Andrews, "Analytical evaluation of fractional frequency reuse for heterogeneous cellular networks," *arXiv preprint arXiv:1112.0674*, 2011.

[65] H. S. Dhillon, R. K. Ganti, F. Baccelli, and J. G. Andrews, "Modeling and analysis of k-tier downlink heterogeneous cellular networks," *CoRR*, vol. abs/1103.2177, 2011. [Online]. Available: http://arxiv.org/abs/1103.2177

[66] N. Deng, W. Zhou, and M. Haenggi, "Heterogeneous cellular network models with dependence," *IEEE Journal on Selected Areas in Communications*, vol. 33, no. 10, pp. 2167–2181, Oct 2015.

[67] T. Shirai and Y. Takahashi, "Random point fields associated with certain Fredholm determinants I: Fermion, poisson and boson point processes," *Journal of Functional Analysis*, vol. 205, no. 2, pp. 414–463, 2003.

[68] N. Deng, W. Zhou, and M. Haenggi, "The Ginibre Point Process as a Model for Wireless Networks With Repulsion," *IEEE Transactions on Wireless Communications*, vol. 14, no. 1, pp. 107–121, Jan 2015.

[69] N. Miyoshi and T. Shirai, "A Cellular Network Model with Ginibre Configured Base Stations," *Advances in Applied Probability*, vol. 46, pp. 832–845, 9 2014.

[70] Y. Li, F. Baccelli, H. S. Dhillon, and J. G. Andrews, "Statistical Modeling and Probabilistic Analysis of Cellular Networks With Determinantal Point Processes," *IEEE Transactions on Communications*, vol. 63, no. 9, pp. 3405–3422, Sept 2015.

[71] N. Alliance, "5G White Paper," Feb. 2015. [Online]. Available: https://www.ngmn.org/uploads/media/NGMN_5G_White_Paper_V1_0.pdf

[72] G. Forum, "5G white paper: New wave towards future societies in the 2020s," Mar. 2015. [Online]. Available: http://media.wix.com/ugd/9b2680_c5a2ce8c8dee4f0d821606f11c302de1.pdf

[73] A. Hakkarainen, "Positioning and Location-Awareness in 5G Networks," Sep. 2016. [Online]. Available: http://www.tut.fi/5G/positioning/

[74] I. Alyafawi, D. C. Dimitrova, and T. Braun, "Sdr-based passive indoor localization system for gsm," in *Proceedings of the 2014 ACM Workshop on Software Radio Implementation Forum*, ser. SRIF '14. New York, NY, USA: ACM, 2014, pp. 7–14. [Online]. Available: http://doi.acm.org/10.1145/2627788.2627790

[75] A. Howard, S. Siddiqi, and G. S. Sukhatme, "An experimental study of localization using wireless ethernet," in *in 4th International Conference on Field and Service Robotics*, 2003.

[76] "802.11-2012 - IEEE Standard for Information technology–Telecommunications and information exchange between systems Local and metropolitan area networks–Specific requirements Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications," IEEE-Inst, Tech. Rep. IEEE Std 802.11-2012. [Online]. Available: http://ieeexplore.ieee.org/servlet/opac?punumber=6178209

[77] J. Foutz, A. Spanias, and M. Banavar, *Narrowband Direction of Arrival Estimation for Antenna Arrays*, ser. Synthesis lectures on antennas. Morgan & Claypool Publishers, 2008. [Online]. Available: https://books.google.com/books?id=28CmfvrOo2YC

[78] Z. Abu-Shaban, "Localisation Testbed using Software-Defined Radio," Master's thesis, Imperial College London, 9 2010.

[79] H. L. Van Trees, *Optimum Array Processing: Part IV of Detection, Estimation, and Modulation Theory*. New York: Wiley-Interscience, 2002.

[80] Ettus Research A National Instruments Company. (2016, Jun.) TwinRX Daughterboard. [Online]. Available: https://www.ettus.com/content/files/TwinRX_Datasheet.pdf

[81] T. F. Collins and A. M. Wyglinski, "Skynet: Sdr-based physical simulation testbed," in *Vehicular Technology Conference (VTC Fall), 2015 IEEE 82nd*, Sept 2015.

[82] R. D. Blumofe, C. F. Joerg, B. C. Kuszmaul, C. E. Leiserson, K. H. Randall, and Y. Zhou, "Cilk: An efficient multithreaded runtime system," *Journal of parallel and distributed computing*, vol. 37, no. 1, pp. 55–69, 1996.

[83] ——, "Cilk: An efficient multithreaded runtime system," in *Proceedings of the Fifth ACM SIGPLAN Symposium on Principles and Practice of Parallel Programming*, ser. PPOPP '95. New York, NY, USA: ACM, 1995, pp. 207–216. [Online]. Available: http://doi.acm.org/10.1145/209936.209958

[84] M. Frigo, "Multithreaded programming in cilk," in *Proceedings of the 2007 International Workshop on Parallel Symbolic Computation*, ser. PASCO '07. New York, NY, USA: ACM, 2007, pp. 13–14. [Online]. Available: http://doi.acm.org/10.1145/1278177.1278181

[85] Y. Guo, J. Zhao, V. Cave, and V. Sarkar, "Slaw: A scalable locality-aware adaptive work-stealing scheduler for multi-core systems," in *Proceedings of the 15th ACM SIGPLAN Symposium on Principles and Practice of Parallel Programming*, ser. PPoPP '10. New York, NY, USA: ACM, 2010, pp. 341–342. [Online]. Available: http://doi.acm.org/10.1145/1693453.1693504

[86] P. Yalamanchili, U. Arshad, Z. Mohammed, P. Garigipati, P. Entschev, B. Kloppenborg, J. Malcolm, and J. Melonakos, "ArrayFire - A high performance software library for parallel computing with an easy-to-use API," Atlanta, 2015. [Online]. Available: https://github.com/arrayfire/arrayfire

[87] The MathWorks Inc., "What Are System Objects?" Feb. 2016. [Online]. Available: https://www.mathworks.com/help/dsp/gs/what-are-system-objects.html

[88] ——, "MATLAB Coder," Feb. 2016. [Online]. Available: http://www.mathworks.com/help/coder/index.html

[89] B. Schling, *The Boost C++ Libraries*. XML Press, 2011.

[90] J. Ellson, E. Gansner, L. Koutsofios, S. North, G. Woodhull, S. Description, and L. Technologies, "Graphviz open source graph drawing tools," in *Lecture Notes in Computer Science*. Springer-Verlag, 2001, pp. 483–484.

[91] D. Vyukov, "Single-Produce/Single-Consumer Queue," Jan. 2009. [Online]. Available: https://software.intel.com/en-us/articles/single-producer-single-consumer-queue

[92] The MathWorks Inc., "Functions and Objects Supported for C and C++ Code Generation," Feb. 2016. [Online]. Available: http://www.mathworks.com/help/simulink/ug/functions-supported-for-code-generation--categorical-list.html

[93] ——, "MEX File Creation API," Feb. 2016. [Online]. Available: http://www.mathworks.com/help/matlab/call-mex-files-1.html

[94] ——, "IEEE 802.11 WLAN - OFDM Beacon Receiver with USRP Hardware," Feb. 2016. [Online]. Available: http://www.mathworks.com/help/supportpkg/usrpradio/examples/ieee-802-11-tm-wlan-ofdm-beacon-receiver-with-usrp-r-hardware.html

[95] Ettus Research A National Instruments Company. (2016, Jun.) USRP B200/B210 Bus Series. [Online]. Available: https://www.ettus.com/content/files/b200-b210_spec_sheet.pdf

[96] B. Bloessl, M. Segata, C. Sommer, and F. Dressler, "An IEEE 802.11a/g/p OFDM Receiver for GNU Radio," in *ACM SIGCOMM 2013, 2nd ACM SIGCOMM Workshop of Software Radio Implementation Forum (SRIF 2013)*. Hong Kong, China: ACM, August 2013, pp. 9–16.

[97] P. H. Y. Wu, "On the complexity of turbo decoding algorithms," in *Vehicular Technology Conference, 2001. VTC 2001 Spring. IEEE VTS 53rd*, vol. 2, 2001, pp. 1439–1443 vol.2.

[98] F. Lavancier, J. Mller, and E. Rubak, "Determinantal Point Process Models and Statistical Inference," *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, vol. 77, pp. 853–877, 2015.

[99] M. Krishnapur, Y. Peres, V. Balint, and B. Hough, *Zeros of Gaussian Analytic Functions and Determinantal Point Processes (University Lecture Series)*. American Mathematical Society, 2009.

[100] A. Baddeley, R. Turner *et al.*, "Spatstat: an r package for analyzing spatial point patterns," *Journal of statistical software*, vol. 12, no. 6, pp. 1–42, 2005.

[101] S. Sun, Y. Ju, and Y. Yamao, "Overlay cognitive radio ofdm system for 4g cellular networks," *IEEE Wireless Communications*, vol. 20, no. 2, pp. 68–73, 2013.

[102] M. Haenggi, *Stochastic Geometry for Wireless Networks*. Cambridge University Press, 2012.

[103] J. G. Andrews, F. Baccelli, and R. K. Ganti, "A Tractable Approach to Coverage and Rate in Cellular Networks," *IEEE Transactions on Communications*, vol. 59, no. 11, pp. 3122–3134, November 2011.

[104] "Spatial channel model for Multiple Input Multiple Output (MIMO) simulations (3GPP TR 25.996 version 12.0.0 Release 12)," *Universal Mobile Telecommunications System (UMTS)*, 2014.

[105] M. Haenggi, "The Mean Interference-to-Signal Ratio and Its Key Role in Cellular and Amorphous Networks," *IEEE Wireless Communications Letters*, vol. 3, no. 6, pp. 597–600, Dec 2014.

[106] K. I. Pedersen, Y. Wang, B. Soret, and F. Frederiksen, "eicic functionality and performance for lte hetnet co-channel deployments," in *Vehicular Technology Conference (VTC Fall), 2012 IEEE*, Sept 2012, pp. 1–5.

[107] R. K. Ganti and M. Haenggi, "Asymptotics and approximation of the SIR distribution in general cellular networks," *CoRR*, vol. abs/1505.02310, 2015. [Online]. Available: http://arxiv.org/abs/1505.02310

[108] M. Haenggi, "The meta distribution of the SIR in poisson bipolar and cellular networks," *CoRR*, vol. abs/1506.01644, 2015. [Online]. Available: http://arxiv.org/abs/1506.01644

[109] Y. Zhong, W. Zhang, and M. Haenggi, "Delay analysis in static poisson network," in *2015 IEEE/CIC International Conference on Communications in China (ICCC)*, Nov 2015, pp. 1–5.

[110] T. Bai and R. W. Heath, "Coverage and rate analysis for millimeter-wave cellular networks," *IEEE Transactions on Wireless Communications*, vol. 14, no. 2, pp. 1100–1114, 2015.

[111] R. K. Ganti and M. Haenggi, "Interference and outage in clustered wireless ad hoc networks," *IEEE Transactions on Information Theory*, vol. 55, no. 9, pp. 4067–4086, 2009.

[112] G. F. Riley and T. R. Henderson, "The ns-3 network simulator," in *Modeling and Tools for Network Simulation*. Springer, 2010, pp. 15–34.

[113] M. Haenggi and R. K. Ganti, *Interference in large wireless networks*. Now Publishers Inc, 2009.

[114] J. Werner, M. Costa, A. Hakkarainen, K. Leppanen, and M. Valkama, "Joint user node positioning and clock offset estimation in 5g ultra-dense networks," in *2015 IEEE Global Communications Conference (GLOBECOM)*, Dec 2015, pp. 1–7.

[115] P. Kyösti and J. Medbo, "A channel model for 5g evaluations," 2015.

[116] 3GPP, "User Equipment (UE) conformance specification; Part 1: Protocol conformance specification," 3rd Generation Partnership Project (3GPP), TS 36.523-1, Jan. 2009. [Online]. Available: http://www.etsi.org/deliver/etsi_ts/136500_136599/13652301/08.00.01_60/ts_13652301v080001p.pdf

[117] G. B. Parrent and B. J. Thompson Jr, "On the fraunhofer (far field) diffraction patterns of opaque and transparent objects with coherent background," *Journal of Modern Optics*, vol. 11, no. 3, pp. 183–193, 1964.

[118] L-Comm, "2.4/4.9/5.8 GHz 3 dBi Tri-band Rubber Duck Antenna - SMA-Male," Nov 2016. [Online]. Available: http://www.l-com.com/multimedia/datasheets/DS_HG2458RD-XXX.PDF

[119] J. X. Zhu and H. Wang, "Effects of sensor position and pattern perturbations on crlb for direction finding of multiple narrow-band sources," in *Spectrum Estimation and Modeling, 1988., Fourth Annual ASSP Workshop on*, Aug 1988, pp. 98–102.

[120] R. Schmidt, "Multiple emitter location and signal parameter estimation," *IEEE Transactions on Antennas and Propagation*, vol. 34, no. 3, pp. 276–280, Mar 1986.

[121] A. Barabell, "Improving the resolution performance of eigenstructure-based direction-finding algorithms," in *Acoustics, Speech, and Signal Processing, IEEE International Conference on ICASSP'83.*, vol. 8. IEEE, 1983, pp. 336–339.

[122] F. Belloni, A. Richter, and V. Koivunen, "Extension of root-music to non-ula array configurations," in *2006 IEEE International Conference on Acoustics Speech and Signal Processing Proceedings*, vol. 4. IEEE, 2006, pp. IV–IV.

[123] A. Edelman and H. Murakami, "Polynomial roots from companion matrix eigenvalues," *Mathematics of Computation*, vol. 64, no. 210, pp. 763–776, 1995.

[124] P. Stoica and A. Nehorai, "Music, maximum likelihood, and cramer-rao bound," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 37, no. 5, pp. 720–741, May 1989.

[125] M. Landmann and G. Del Galdo, "Efficient antenna description for mimo channel modelling and estimation," in *Wireless Technology, 2004. 7th European Conference on.* IEEE, 2004, pp. 217–220.

[126] M. Pralon, D. Schulz, and R. S. Thomä, "Optimization of antenna arrays for 2d doa estimation using eadf for cramér-rao lower bounds computation," in *Antennas and Propagation in Wireless Communications (APWC), 2013 IEEE-APS Topical Conference on.* IEEE, 2013, pp. 1429–1432.

[127] A. Schmitz, T. Karolski, and L. Kobbelt, "Using spherical harmonics for modeling antenna patterns," in *Radio and Wireless Symposium (RWS), 2012 IEEE*, Jan 2012, pp. 155–158.

[128] G. D. Galdo, J. Lotze, M. Landmann, and M. Haardt, "Modelling and manipulation of polarimetric antenna beam patterns via spherical harmonics," in *Signal Processing Conference, 2006 14th European*, Sept 2006, pp. 1–5.

[129] M. Narandzic, M. Kaske, C. Schneider, M. Milojevic, M. Landmann, G. Sommerkorn, and R. S. Thoma, "3d-antenna array model for ist-winner channel simulations," in *2007 IEEE 65th Vehicular Technology Conference - VTC2007-Spring*, April 2007, pp. 319–323.

[130] Y.-S. Yoon, L. M. Kaplan, and J. H. McClellan, *DOA Estimation of Wideband Signals*. Artech House, 2006.

[131] T. M. Schmidl and D. C. Cox, "Robust frequency and timing synchronization for ofdm," *IEEE transactions on communications*, vol. 45, no. 12, pp. 1613–1621, 1997.

[132] The GNURadio Project, "GNU Radio Manual and C++ API Reference: Message Passing," 2016. [Online]. Available: http://gnuradio.org/doc/doxygen/page_msg_passing.html

[133] Ettus Research A National Instruments Company. (2016, Jun.) Octoclock and OctoclockG. [Online]. Available: https://www.ettus.com/content/files/Octoclock_Spec_Sheet.pdf

[134] Ettus Research, "Align los in the front-end (sbx, ubx)," Nov 2016. [Online]. Available: https://files.ettus.com/manual/page_sync.html#sync_phase_lo

[135] M. Rice, *Digital Communications: A Discrete-time Approach*. Pearson/Prentice Hall, 2009. [Online]. Available: https://books.google.com/books?id=EB3r7JtXlWwC

[136] T. F. Collins, "Real-Time SDR Localization: Multiple Phased Array System," Poster, 2016.

[137] C. Sanderson and R. Curtin, "Armadillo: a template-based c++ library for linear algebra," *Journal of Open Source Software*, 2016.

[138] J. Dongarra, I. Duff, J. D. Croz, and S. Hammarling, "cgemm," 1989. [Online]. Available: https://www.math.utah.edu/software/lapack/lapack-blas/cgemm.html

[139] N. Fistas and A. Manikas, "A new general global array calibration method," in *Acoustics, Speech, and Signal Processing, 1994. ICASSP-94., 1994 IEEE International Conference on*, vol. 4. IEEE, 1994, pp. IV–73.

[140] A. Paulraj, T. Shan, V. Reddy, and T. Kailath, "A subspace approach to determining sensor gain and phase with applications to array processing," in *30th Annual Technical Symposium*. International Society for Optics and Photonics, 1986, pp. 102–109.

[141] A. L. Swindlehurst and T. Kailath, "A performance analysis of subspace-based methods in the presence of model errors. i. the music algorithm," *IEEE Transactions on Signal Processing*, vol. 40, no. 7, pp. 1758–1774, Jul 1992.

[142] B. C. Ng and C. M. S. See, "Sensor-array calibration using a maximum-likelihood approach," *IEEE Transactions on Antennas and Propagation*, vol. 44, no. 6, pp. 827–835, 1996.

[143] J. Werner, A. Hakkarainen, and M. Valkama, "Cramer-rao bounds for hybrid rss-doa based emitter location and transmit power estimation in cognitive radio systems," in *Vehicular Technology Conference (VTC Fall), 2013 IEEE 78th*. IEEE, 2013, pp. 1–7.

[144] L. Decreusefond, I. Flint, and A. Vergne, "Efficient simulation of the Ginibre point process," *ArXiv e-prints*, Oct. 2013.