ACHIEVING NEAR MAP PERFORMANCE WITH AN EXCITED MARKOV CHAIN MONTE CARLO MIMO DETECTOR

by

Jonathan Carl Hedstrom

A dissertation submitted to the faculty of
The University of Utah
in partial fulfillment of the requirements for the degree of

Doctor of Philosophy

Department of Electrical and Computer Engineering

The University of Utah

August 2017

The University of Utah Graduate School

STATEMENT OF DISSERTATION APPROVAL

The dissertation of **Jonathan Carl Hedstrom**

has been approved by the following supervisory committee members:

Behrouz Farhang ,	Chair(s)	**March 24, 2017** Date Approved
Rong-Rong Chen ,	Member	**March 24, 2017** Date Approved
Osama S. Haddadin ,	Member	**March 24, 2017** Date Approved
Kenneth Stevens ,	Member	**March 24, 2017** Date Approved
Jeffrey Walling ,	Member	**March 24, 2017** Date Approved

by **Gianluca Lazzi** , Chair of

the Department of **Electrical and Computer Engineering**

and by **David B. Kieda** , Dean of The Graduate School.

ABSTRACT

The continuous growth of wireless communication use has largely exhausted the limited spectrum available. Methods to improve spectral efficiency are in high demand and will continue to be for the foreseeable future. Several technologies have the potential to make large improvements to spectral efficiency and the total capacity of networks including massive multiple-input multiple-output (MIMO), cognitive radio, and spatial-multiplexing MIMO. Of these, spatial-multiplexing MIMO has the largest near-term potential as it has already been adopted in the WiFi, WiMAX, and LTE standards.

Although transmitting independent MIMO streams is cheap and easy, with a mere linear increase in cost with streams, receiving MIMO is difficult since the optimal methods have exponentially increasing cost and power consumption. Suboptimal MIMO detectors such as K-Best have a drastically reduced complexity compared to optimal methods but still have an undesirable exponentially increasing cost with data-rate. The Markov Chain Monte Carlo (MCMC) detector has been proposed as a near-optimal method with polynomial cost, but it has a history of unusual performance issues which have hindered its adoption.

In this dissertation, we introduce a revised derivation of the bitwise MCMC MIMO detector. The new approach resolves the previously reported high SNR stalling problem of MCMC without the need for hybridization with another detector method or adding heuristic temperature scaling terms. Another common problem with MCMC algorithms is an unknown convergence time making predictable fixed-length implementations problematic. When an insufficient number of iterations is used on a slowly converging example, the output LLRs can be unstable and overconfident, therefore, we develop a method to identify rare, slowly converging runs and mitigate their degrading effects on the soft-output information. This improves forward-error-correcting code performance and removes a symptomatic error floor in bit-error-rates. Next, pseudo-convergence is identified with a novel way to visualize the internal behavior of the Gibbs sampler. An effective and efficient pseudo-convergence detection and escape strategy is suggested. Finally, the new excited MCMC (X-MCMC)

detector is shown to have near maximum-a-posteriori (MAP) performance even with challenging, realistic, highly-correlated channels at the maximum MIMO sizes and modulation rates supported by the 802.11ac WiFi specification, 8×8 256 QAM.

Further, the new excited MCMC (X-MCMC) detector is demonstrated on an 8-antenna MIMO testbed with the 802.11ac WiFi protocol, confirming its high performance.

Finally, a VLSI implementation of the X-MCMC detector is presented which retains the near-optimal performance of the floating-point algorithm while having one of the lowest complexities found in the near-optimal MIMO detector literature.

For my parents, Paulette and Rocky.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGEMENTS

So many people have helped me throughout the years that it would take at least the length of this dissertation to acknowledge them. In particular, I would like to highlight the specific contributions of a few people and organizations that have made this work possible either through guidance, example, or contribution.

My parents, Rocky and Paulette Hedstrom, have always been the source of my success. They taught me the power of hard work, determination, when to do things your own way, and when to follow. I will always happily be in their debt.

My advisor, Behrouz Farhang, is a strong mentor and I'm thankful for the opportunity to work and learn from him over the last few years. None of this could have been done without him. I couldn't imagine a better advisor for the complicated task of commercializing this technology. He's always made the time to be intimately involved when needed, has trusted me with immense responsibility, is a patient guide, and happens to be a great person.

My committee has been amazingly supportive, understanding, and helpful with the difficulties of having this research done partially at the university and partially in our startup. Rong-Rong Chen in particular has contributed a lot of great insight in preparing thorough derivations of our techniques.

My colleague, George Yuen, has made huge contributions to this project. He's responsible for much of the work with the WiFi protocol and his strong mathematical background has strengthened and refined my often ad-hoc approach. It's been a pleasure to work with him over the last few years and I look forward to our continued success.

My colleague, Ahamd RezazadehReyhani, has contributed an impressive fixed point library, methods to estimate fixed-point bit widths, and GPU implementations of MAP. Now that he's dedicated to extending our MCMC work full time, I'm really looking forward to seeing what new insight and innovations he will bring.

The students who've previously worked on the MCMC detector under Behrouz and Rong-Rong laid a strong foundation for me to build on. It was a great learning experience

digging into their work and a lot of fun extending it, especially Andy Laraway's.

I would like to thank my prior colleagues from MIT Lincoln Laboratory from whom I learned crucial skills that I keep in my toolbox of tricks. In particular, this includes Tim Schiefelbein who taught me by example to code like my hair's on fire, John Stueve who guided me on managing chaotic R&D projects, Seth Kosowsky who initiated me in the mystic art of heuristic algorithm development, and late nights in the desert where I learned indomitable debugging determination.

I would like to thank the University of Utah Technology and Venture Commercialization (TVC) organization. They provided early funding for our work and for our first testbed through the Engine Fund. Lance Pedersen, Danuta Petelenz, Matthew Gardner, and Natan Chetrit have provided guidance and assistance in commercialization and are valued stakeholders in the technology's success.

I would like to thank the Utah Science Technology and Research (USTAR) initiative for interim funding through the TAP grant in 2016 that was critical to maintaining momentum in the gap between our NSF STTR Phase 1 and Phase 2 funding.

Additionally, I would like to thank the the University of Utah USTAR initiative for teaching me the lean business model canvas method. The first cohort with Phillip Grimm, David Robinson, Greg Jones, Angela Mitcham, and Kevin Jessing as trainers and advisors helped tremendously in developing our commercialization strategy, refining our message, and in winning our NSF STTR Phase 2 grant.

I would like to thank the National Science Foundation (NSF) for their financial support during my studies and for their continued support in commercializing the technology. This work has been funded under Small Business Technology Transfer (STTR) Program Phase 1 and 2 grant numbers 1449033 and 1632569.

I would also like to thank the Linux, Python, Numpy, CFFI, Matplotlib, and Seaborn open-source teams for the excellent software used in this work. I hope to contribute back to the community in the near future by open-sourcing some of our own code to help others.

CHAPTER 1

INTRODUCTION

Wireless communication systems have a strong long-term trend towards ever-increasing data throughput requirements while having a fixed amount of scarce and expensive frequency bandwidth available. Recent growth in cellular smartphone and wireless broadband use have largely exhausted the available licensed cellular bands. This has resulted in the recent record-breaking wireless spectrum auction of a mere 65 MHz of total bandwidth in the 1700-2100 MHz band in the United States for $45 billion [2]. Increasing the efficiency of spectral use is an important and valuable technological problem now and for the foreseeable future.

It is useful to understand how spectral efficiency has improved thus far so that the remaining opportunities can be better appreciated. For the first 100 years of radio-frequency communications technology, incremental improvements in analog technology brought slow but steady improvements. Then in the early 1990s the first digital cellular phone standards were introduced which brought a large jump in spectral efficiency as well as massive set of new opportunities available with complex digital signal processing techniques [3]. As Moore's Law drove onward increasing the processing capabilities available to digital systems, the complexity of implementable DSP algorithms was also able to increase [4]. Introduction of more efficient modulation and coding schemes such as quadrature amplitude modulation (QAM) with orthogonal frequency domain multiplexing (OFDM) [5] and low-density-parity-check (LDPC) coding [6] have enabled some of the most advanced wireless protocols in use today such as WiFi [7, 8] and LTE cellular [9, 10].

Now that Moore's Law is slowing [11] and the potential gains from improved modulation and coding are largely exhausted, it is becoming ever harder to continue improving spectral efficiency to meet demand. Several of the remaining high value paths to increasing efficiency include massive multiple-input multiple-output (MIMO) [12], cognitive radio [13], and

improved spatial-multiplexing MIMO [14, 15]. Massive MIMO and cognitive radio are still actively being researched, and are not yet being used in a commercial settings, whereas limited spatial-multiplexing MIMO has already been deployed in WiFi since 802.11n (2009) and in cellular since 3GPP Release 7 (2007).

By using spatial-multiplexing MIMO, data rates can theoretically be increased linearly without requiring additional spectral resources [16, 17]. All that is needed is a rich scattering environment and multiple antennas which means that it is best used exactly where it is needed, high density urban and indoor areas. For spatial-multiplexing to be an effective solution, it needs a high performing, low cost, and low power MIMO detector to separate the mixed together spatial streams at the receiver.

High performance, low complexity MIMO detectors is an active area of research. The type of detectors currently favored in the literature is the K-Best variant of sphere-decoding [18] which is known to have near-optimal performance though with a cost that exponentially increases with modulation density. Markov Chain Monte Carlo (MCMC) has been proposed as an alternative approach believed to have a less than exponentially increasing cost [19]. Much of the literature since the introduction of MCMC has been centered on resolving a high SNR stalling problem which degrades performance [20–24].

The focus of this dissertation is to make the MCMC detector viable for mainstream use. We propose a new excited MCMC (X-MCMC) detector with dramatically improved performance compared to the prior work, demonstrate its effectiveness on an 8-antenna MIMO testbed with the 802.11ac WiFI protocol, and implement a low cost VLSI design.

In this chapter, we review some basic information that will be useful in understanding the chapters to come. This includes an overview of spatial-multiplexing MIMO communication techniques in Section 1.1. Then in Section 2, the basics of Markov Chain Monte Carlo is introduced. Turbo loops which improve receiver performance by making a joint decision between the detector and decoder is described in Section 1.4. Finally, the main contributions of the dissertation are outlined in Section 1.5 and the structure of the dissertation is presented in Section 1.6.

1.1 Spatial-Multiplexing MIMO

Information in a wireless communication system can generally exploit 3 dimensions: time, frequency, and space. Multiantenna systems exploit spatial information to improve reliability, capacity, or some combination of the two. Information can be transmitted and/or received redundantly, using the additional antennas for diversity gain with space-time-codes [25, 26], beamforming [27], or maximum-ratio-combining [28]. Information can also be sent independently over each antenna to increase the total data-rate as in spatial-multiplexing, also sometimes referred to as vertical Bell Laboratories Space-Time (V-BLAST) due to the foundational work at Bell Laboratory [29, 30].

The goal of a spatial-multiplexing MIMO transceiver is to exploit the spatial multipath of the environment to support overlapping data streams. This concept allows reuse of the spectrum and therefore has the potential to dramatically increase the data rates and capacity of wireless networks [16, 17]. To perform data transmission with spatial-multiplexing, the transmitter sends independent data streams simultaneously on a set of N_t transmit antennas which are then received at a set of N_r receive antennas. The channel can be represented in the frequency-domain by the complex matrix $\mathbf{H}$ where each element represent the path gain and delay between a pair of transmit and receive antennas, as shown in Fig. 1.1. This allows the system to be modeled as

$$\mathbf{y} = \mathbf{H}\mathbf{s} + \mathbf{n} \tag{1.1}$$

where $\mathbf{y}$ is the received signal vector, $\mathbf{s}$ is the vector of transmitted QAM constellation symbols, and $\mathbf{n}$ is a noise vector. The noise elements are assumed to be independent and identically distributed (i.i.d.) complex Gaussian random variables with variance of σ_n^2 per

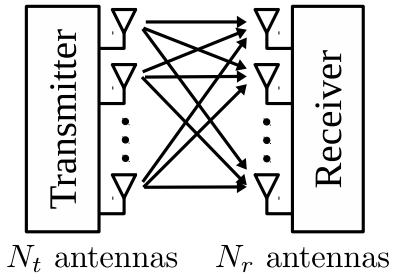


Figure 1.1. Channel model showing the pairwise paths between the N_t transmit and N_r receive antennas.

each real and imaginary dimension. Note that in this dissertation we assume that the channel can be represented by a slow flat-fading model which means that the channel **H** can be assumed to be constant for duration of use. This is appropriate for 802.11ac WiFi simulations since it is orthogonal frequency domain multiplexing (OFDM) based, there are generally no fast moving objects like cars within the environment, and the packets are short compared to the fading time.

We will adopt a compact notation to express the size of a spatial-multiplexing MIMO system with $N_t \times N_r : N_s$. This represents that there are N_s independent spatial streams being transmitted over N_t transmit antennas to N_r receive antennas. Therefore, a symmetric system using 4 antennas to send 3 spatial streams would be referred to as a 4x4:3 system.

Compared to a single-input single-output (SISO) system, there is an additional performance limiter beyond noise and channel gain. Notice that the previous conversation assumes a rich scattering environment. Mathematically, this means that the environment is complex with sufficiently independent signal paths that the channel matrix **H** is strongly orthogonal. As the channel becomes more ill-conditioned the capacity will be degraded, possibly meaning that the system should switch from a high throughput spatial-multiplexing mode to a redundant one such as beamforming. This is the reason why spatial-multiplexing is not generally applicable. Free-space line-of-sight communications in particular, such as microwave backhauls, aircraft, and satellites, cannot exploit more than 2-stream MIMO by using simple polarization diversity.

When there is a 2-way MIMO communications link, it may be possible to augment the transmission to improve performance. This can be thought of as the opposite of beamforming, where the goal is to spread the signals into independent paths. To do this, channel reciprocity can be used by the transmitter to identify the channel seen by the receiver. Then a matrix can be identified, which, when used to premultiply against the transmitted vector before transmission, improves the orthogonality of the channel at the receiver. This precoding technique, sometimes referred to as waterfilling, can be used to artificially improve the capacity of a MIMO system [31–33].

The degree that the channel impacts performance is strongly determined by the receiver's MIMO detector which identifies the data symbols mixed together at the receiver. This will be the topic of the next section.

1.2 MIMO Detectors

The MIMO detector is responsible for using the signal at the N_r receive antennas to identify the transmitted sequence. The performance and complexity of this operation is often the limiting factor in building large spatial-multiplexing MIMO systems [14]

Zero-forcing (ZF) and minimum-mean-square-error (MMSE) detectors [34] essentially apply a channel inverse to the received signal $\mathbf{y}$. They are attractive because of their low complexity, but suffer from poor performance with ill-conditioned channels due to noise enhancement. Maximum-likelihood (ML) and maximum-a-posteriori (MAP) detectors [35] essentially check every possibly transmitted bit permutation, testing which matches the received signal most closely when the channel is applied. They have attractive optimal performance, but are limited by their exponentially increasing complexity as the number of antennas and modulation density increases. Successive interference cancelers (SIC) are a class of interesting detectors suggested with the introduction of V-BLAST [30] and have a performance and complexity somewhere between straight inverse methods and maximum-likelihood methods.

The sphere-decoding (SD) class of MIMO detectors are known to have near maximum-a-posteriori (MAP) performance with a much lower complexity [35–41]. Essentially, it searches iteratively within a spherical subset of the solution space. Since the hyper-sphere volume grows exponentially with the number of bits transmitted and the radius must increase as SNR decreases, the complexity can be too great to implement in practice for large numbers of transmitted bits and when the receiver operates near the capacity bound.

The K-Best detector modifies the sphere-decoder algorithm with a breadth-first tree search strategy in which only the best K-candidates at each layer of the tree are kept, thus reducing the number of bit sequences in the search to a smaller one and making the search size deterministic. These detectors have been demonstrated in effective VLSI designs, but their complexity increases quickly with the number of antennas, the number of transmitted bits per channel use, and the list size [18, 42]. Therefore, the search for lower complexity MIMO detectors is ongoing.

Markov Chain Monte Carlo is a statistical search technique which may be able to meet near ML and MAP performance with a less than exponentially increasing complexity [19]. This is the focus of the following section.

1.3 Markov Chain Monte Carlo

Markov Chain Monte Carlo (MCMC) has been shown to have near optimal performance at low signal-to-noise-ratio (SNR) and to have efficient hardware implementations [19, 43]. The foundation of this technology is associated with 4 main patents held by the University of Utah [44–47]. It uses a random walk through the permutations of the transmitted bit sequence to estimate the posterior probability distribution and accordingly generate soft-output information, estimates of the transmitted bit sequence with an associated confidence.

Unlike other detectors, MCMC performance degrades as SNR increases. This undesirable behavior is caused by stalling [20]. Most of the attempted solutions to this problem can be grouped into either hybridization or temperature scaling approaches.

The hybridization schemes combine a method with good, low complexity performance at high SNR with MCMC to combine the best traits of both. Examples are ZF, MMSE, and sphere-decoding initialized MCMC methods [21, 48].

The temperature scaling approaches are so called because they recognize that the probabilities generated by the Gibbs sampler are too cold, resulting in slow convergence. A linear scaling coefficient is therefore applied to artificially increase the noise temperature. This coefficient must be heuristically optimized depending on system parameters and SNR [23, 49, 50], though a recent derivation of a near optimal value is showing promise under some testing conditions [24].

An additional general problem of all MCMC-based algorithms is that the mixing time, the number of iterations needed to converge to the posterior distribution, is variable and notoriously difficult to predict [51]. This makes the design of a realistic fixed iteration length MCMC detector challenging, either erring on wasting resources with an algorithm that is always safely run for too many iterations, or a cheaper implementation that risks producing poorly converged erroneous output.

In this dissertation we will explore the MCMC detector in detail, propose improvements, demonstrate the improvements on a MIMO testbed, and implement a VLSI design.

1.4 Turbo Loops

Many of the MIMO detectors discussed in this dissertation are compatible with turbo iterations, the exchange of soft-information between the detector and decoder as in Fig. 1.2 [52]. This allows for the iterative joint detection of the signal for enhanced performance [53]. The soft-information is in the form of log likelihood ratios (LLRs) and represented by λ^a and λ^e for their *a priori* input and extrinsic output versions.

1.5 Contributions of Dissertation

This dissertation thoroughly explores the potential use of the MCMC detector for mainstream MIMO applications. We have developed the high performance X-MCMC detector, demonstrated its effectiveness with 802.11ac WiFi on an 8-antenna MIMO testbed, and implemented an X-MCMC VLSI design through synthesis. This work has resulted in our early testbed results being published and presented at IEEE International Conference on Communications (ICC'15) [54], our new testbed results being accepted at ICC'17 [55], and a journal paper submission to IEEE Transactions on Wireless Communications [56]. Two patents have been submitted through the University of Utah [57, 58].

The X-MCMC detector is comprised of 3 main components: the excited Gibbs sampler, pseudo-convergence detection and escape, and output LLR overconfidence conditioning. These components are the primary contributions of the dissertation and represent a significant contribution to the body of knowledge on MCMC detectors. Through statistical derivations and novel visualization methods we have shown that these methods resolve high SNR stalling,

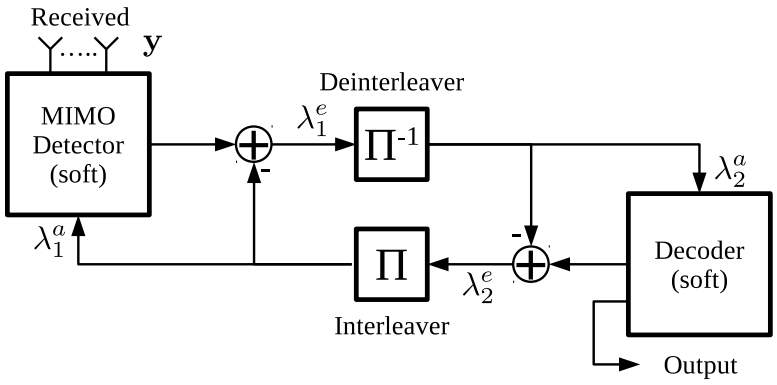


Figure 1.2. Turbo loop structure with a MAP, MCMC, X-MCMC, or K-Best MIMO detector iteratively exchanging soft information with a forward error correction decoder.

improve sampling efficiency, and compensate for variable convergence times. Using these techniques has made our X-MCMC detector the highest performing MCMC detector in the literature.

To demonstrate the high performance of the X-MCMC detector, 3 generations of testbeds were created. The final testbed successfully validated near maxlog-MAP performance at up to 8x8 MIMO 64 QAM sizes with the 802.11ac WiFi protocol. This is the first real-world published demonstration of any MCMC detector at 8-antenna MIMO sizes that we know of. Auxiliary contributions from this work include development of a new interference metric and use of channel matrix condition number to match testbed measurement results to simulation. Our detailed analysis has shown that the Gaussian i.i.d. channel model broadly used in the MCMC literature is insufficient to represent indoor environments and can lead to misleading conclusions. The final 8-antenna MIMO testbed based on the Ettus B210 software defined radio is potentially useful to other researchers as it is user friendly, provides high performance, and is 1/20 the cost of some commercial alternatives.

High performance must be paired with an implementable low cost for the X-MCMC detector to see mainstream adoption, therefore, we have created a VLSI design of the algorithm in System Verilog. Synthesis results show that it is among the lowest cost MIMO detectors in the literature. The implementation of the Version A X-MCMC detector resulted in several important innovations. First, the use of power-of-2 division is used to simplify the excitation and pseudo-convergence escape calculations. Next, we extend the previous approximations of the probability function to include a crude rectangular option. The partial list update is proven to be effective compared to a more costly asymmetric one. Finally, ideas from feedback control theory are used to allow delay in the calculations and thereby reduce the length of each Gibbs step by 1-clock-cycle.

1.6 Structure of Dissertation

This dissertation proceeds as follows. We begin with an overview of the Markov Chain Monte Carlo MIMO detector in Chapter 2, along with a derivation of the technique. The limitations of the algorithm are shown and several improvements from the literature are reviewed. The MMSE initialized MCMC detector is described for use as a reference method used elsewhere in the dissertation.

In Chapter 3, we present our main contributions in the form of the excited MCMC (X-MCMC) detector. This includes a derivation of the excited Gibbs sampler and output LLR overconfidence conditioning. Pseudo-convergence is identified and a strategy to detect and escape this efficiency reducing effect are suggested. Simulations are shown with near maxlog-MAP performance at up to 8x8 MIMO 256QAM sizes.

In Chapter 4, we verify the X-MCMC high performance by sending 802.11ac packets over 4- and 8-antenna MIMO testbeds. Methods to match measurement results to simulation are provided which are used in other sections of the dissertation. We demonstrate the importance of using a representative channel model for simulations and use the condition-number of the channel matrix to gain a deeper understanding of the impact of ill-conditioned channels. Interference metrics are investigated resulting in the development of our new HSADR measurement to provide consistent measurement analysis.

Chapter 5 outlines changes to the floating-point X-MCMC detector needed to create an effective VLSI hardware implementation. The impact of these methods and approximations are analyzed and are shown to still allow near maxlog-MAP performance. Synthesis results of our HDL design are used to show that the low complexity and cost of our is among the best in the literature.

Concluding remarks and suggestions for potential future work are made in Chapter 6.

CHAPTER 2

THE MARKOV CHAIN MONTE CARLO MIMO DETECTOR

In this chapter, we present the basics of the Markov Chain Monte Carlo (MCMC) MIMO detector. We include a detailed derivation of the Gibbs sampler transition probabilities and of the output LLR calculation so that the differences with the new X-MCMC detector presented in Chapter 3 can be better understood. After deriving the basic equations needed by the Gibbs sampler and output LLR calculation, we explain the high signal-to-noise-ratio (SNR) stalling problem that limits performance as the signal strength increases. Potential solutions to the stalling problem found in the literature are reviewed and their deficiencies explained. Finally, the MMSE initialized MCMC detector is described for use as a reference algorithm in future chapters.

2.1 Overview

The MCMC detector is an interesting class of detectors that is very different from other detector methods in the literature. Generally, MIMO detectors use some method to reverse the system model with linear algebra such as with a channel inverse in soft-MMSE [34] or using a tree-search process on a QR decomposition in K-Best [18]. The MCMC detector instead estimates the output log likelihood ratio (LLR) by means of Monte Carlo sampling [19]. This can be thought of as a method to identify a list of important bit permutation samples with a random walk and use them to approximate the posterior distribution of the likelihood ratios, similar to how maximum-likelihood detection works but with an exponentially shorter list. The challenge is to make the sampling process computationally efficient by using an easy-to-calculate short list that accurately captures the statistics of the full permutation list used by maximum-likelihood. We select the bitwise method described in [43, 59, 60] as our foundational algorithm because it has been shown to be efficiently implementable in hardware.

The two main components of the MCMC detector is the Gibbs sampler and the LLR output calculation. The Gibbs sampler starts with an initial estimate of the transmitted bit sequence, either randomly selected or initialized with prior information. It then cycles through the bits, calculating the probability of a bit being a 1 or 0 and uses it to weight a random decision to change the bit. Each cycle through the bits is an iteration. The list of permutations visited by the Gibbs sampler is used to calculate the output LLR of each bit.

2.2 System Model and Notation

In an orthogonal frequency division multiplexing (OFDM) system, for each subcarrier, this can be represented with a flat-fading frequency-domain system model as

$$\mathbf{y} = \mathbf{H}\mathbf{s} + \mathbf{n} \ . \tag{2.1}$$

Here, $\mathbf{y}$ is the received signal vector, $\mathbf{H}$ is a slow flat fading complex channel matrix containing the pairwise gain and phase between antennas, $\mathbf{s}$ is the vector of transmitted complex constellation symbols mapped from the bit sequence $\mathbf{x}$, and $\mathbf{n}$ is a noise vector. Assuming the transmit and receive side have the same number of antennas $N_t = N_r = N$, the dimensions of the corresponding vectors and matrices are $N \times 1$ and $N \times N$, respectively. The channels for this dissertation are produced with the method described by the WiFi TGn Model-D specification [61] which creates a correlated channel matrix $\mathbf{H}$. The TGn-D channel is much more challenging for the detector compared to the independent and identically distributed (i.i.d.) complex Gaussian channel typically used in the literature [18,19,24,35,43]. The bit vector $\mathbf{x}$ may be described as comprising of 1's and 0's or equivalently +1's and −1's, depending on context. In principle, $\mathbf{s}$ can be a vector of any complex modulated symbols, but here we use the quadrature amplitude modulated (QAM) symbols defined in 802.11ac [7]. The elements of $\mathbf{n}$ are assumed to be i.i.d. complex Gaussian random variables with variance of σ_n^2 per each real and imaginary dimension.

The primary challenge in a MIMO receiver, and the focus of this dissertation, is to accurately and efficiently estimate the $K = N \log_2(N_{\text{qam}})$ simultaneously transmitted bits in $\mathbf{x}$ per realization, where N_{qam} is the QAM constellation size.

In the equations that follow, some specialized notation is used for compactness and clarity. Vectors and matrices are expressed with bold fonts and the latter are capitalized. The removal of the k^{th} element of a vector is shown with set notation as $\{\cdot\}^{\setminus k}$. A variable or

vector derived from the bit sequence $\mathbf{x}$ with the k^{th} bit forced to a 1 or 0 is shown with $\{\cdot\}^{k+}$ and $\{\cdot\}^{k-}$, respectively. When two nearly identical equations are needed differing only in use of $k+$ or $k-$, $k\pm$ is used to represent both versions. If the k^{th} bit is forced to the correct transmitted value, either 1 or 0, it is shown with $\{\cdot\}^{k*}$.

2.3 Gibbs Sampler

The Gibbs sampler is at the core of the MCMC algorithm. It is used in difficult multivariate posterior probability estimation problems when sampling probabilities jointly across all variables is too complex [62–66]. It cycles across the variables, calculating probabilities conditioned on all other variables being fixed to the current state. In the bitwise MCMC MIMO detector, this means that the Gibbs sampler calculates the probability of a specific bit x_k being a $+1$ or -1 conditioned on the current state of $\mathbf{x}^{\backslash k}$. Therefore, the probability $P(x_k = +1|\mathbf{y}, \mathbf{x}^{\backslash k}, \boldsymbol{\lambda}^a)$ is needed which we refer to henceforth as P_{gibbs} for brevity.

To derive P_{gibbs} we begin with the definition of the LLR for the k^{th} bit at the current Gibbs sampler state $\mathbf{x}$

$$\gamma_k = \ln \frac{P(x_k = +1|\mathbf{y}, \mathbf{x}^{\backslash k}, \boldsymbol{\lambda}^a)}{P(x_k = -1|\mathbf{y}, \mathbf{x}^{\backslash k}, \boldsymbol{\lambda}^a)} \ . \tag{2.2}$$

Then, by noting that

$$P(x_k = -1|\mathbf{y}, \mathbf{x}^{\backslash k}, \boldsymbol{\lambda}^a) = 1 - P(x_k = +1|\mathbf{y}, \mathbf{x}^{\backslash k}, \boldsymbol{\lambda}^a) \ , \tag{2.3}$$

we can rearrange to have the definition of the Gibbs probability as

$$P_{\text{gibbs}} = P(x_k = +1|\mathbf{y}, \mathbf{x}^{\backslash k}, \boldsymbol{\lambda}^a) = \frac{1}{1 + e^{-\gamma_k}} \ . \tag{2.4}$$

Next, Bayes' Theorem is applied to γ_k to separate the contribution of the prior $\boldsymbol{\lambda}^a$. The result is

$$\begin{aligned}
\gamma_k &= \ln \frac{p(\mathbf{y}|x_k = +1, \mathbf{x}^{\backslash k}, \boldsymbol{\lambda}^a) P(x_k = +1|\mathbf{x}^{\backslash k}, \boldsymbol{\lambda}^a) p(\mathbf{y})}{p(\mathbf{y}|x_k = -1, \mathbf{x}^{\backslash k}, \boldsymbol{\lambda}^a) P(x_k = -1|\mathbf{x}^{\backslash k}, \boldsymbol{\lambda}^a) p(\mathbf{y})} \\
&= \ln \frac{p(\mathbf{y}|x_k = +1, \mathbf{x}^{\backslash k}) P(x_k = +1|\lambda_k^a)}{p(\mathbf{y}|x_k = -1, \mathbf{x}^{\backslash k}) P(x_k = -1|\lambda_k^a)} \\
&= \ln \frac{p(\mathbf{y}|x_k = +1, \mathbf{x}^{\backslash k})}{p(\mathbf{y}|x_k = -1, \mathbf{x}^{\backslash k})} + \lambda_k^a
\end{aligned} \tag{2.5}$$

where in the second line $\boldsymbol{\lambda}^a$, $\mathbf{x}^{\backslash k}$, and $\boldsymbol{\lambda}^{a,\backslash k}$ are removed from their respective conditionals because of the independence among the $\mathbf{x}$ bits created by the interleaving effect in the turbo loop. The final line follows from the definition of the *a priori* LLR $\boldsymbol{\lambda}^a$.

The system model is now used to provide the probability of the received sequence $\mathbf{y}$ given $\mathbf{x}$ and noise with variance σ_n^2 per dimension. This leads to

$$p(\mathbf{y}|\mathbf{x}) = \left(2\pi\sigma_n^2\right)^{-N} \exp\left(-\frac{\|\mathbf{y}-\mathbf{Hs}\|^2}{2\sigma_n^2}\right) \quad . \tag{2.6}$$

Substituting (2.6) into (2.5) and simplifying the result, we get

$$\gamma_k = \frac{1}{2\sigma_n^2}\left(\left\|\mathbf{y}-\mathbf{H}\mathbf{s}^{k-}\right\|^2 - \left\|\mathbf{y}-\mathbf{H}\mathbf{s}^{k+}\right\|^2\right) + \lambda_k^a \quad . \tag{2.7}$$

This may be used with (2.4) to calculate the needed P_{gibbs} probability for the Gibbs sampler. Note that this will only be an accurate calculation if additive white Gaussian noise (AWGN) with variance σ_n^2 is the only contributer to the error residual in $\mathbf{y}-\mathbf{Hs}$. This assumption is revisited in the derivation of the X-MCMC detector in Chapter 3 where we find that error in the Gibbs sampler's current state of $\mathbf{s}$ introduces additional error that should be accounted for.

2.4 Output LLR Calculation

To calculate the extrinsic output LLR $\boldsymbol{\lambda}^e$ we first derive the MAP output LLR. The definition of LLR for the k^{th} bit is the log ratio of probabilities of x_k being a +1 or −1. This is expressed as

$$\begin{aligned}\lambda_{k,\text{MAP}}^e &= \ln\frac{P(x_k=+1|\mathbf{y},\boldsymbol{\lambda}^{a,\backslash k})}{P(x_k=-1|\mathbf{y},\boldsymbol{\lambda}^{a,\backslash k})} \\ &= \ln\frac{\displaystyle\sum_{\mathbf{x}\in\mathbb{X}^{k+}}\left(p(\mathbf{y}|\mathbf{x})\prod_{j\neq k}P(x_j|\lambda_j^a)\right)}{\displaystyle\sum_{\mathbf{x}\in\mathbb{X}^{k-}}\left(p(\mathbf{y}|\mathbf{x})\prod_{j\neq k}P(x_j|\lambda_j^a)\right)}\end{aligned} \tag{2.8}$$

where $\mathbb{X}^{k+}$ and $\mathbb{X}^{k-}$ are the sets of all permutations of $\mathbf{x}$ with the k^{th} bit forced to a +1 and −1, respectively. To simplify (2.8), we recall from [67] that

$$P(x_j|\lambda_j^a) = \frac{e^{-\lambda_j^a/2}}{1+e^{-\lambda_j^a}}e^{x_j\lambda_j^a/2} = A(\lambda_j^a)e^{x_j\lambda_j^a/2} \quad . \tag{2.9}$$

Since the coefficient $A(\lambda_j^a)$ is independent of x_j, it may be separated from the summation and canceled. This leads to

$$\lambda^e_{k,\text{MAP}} = \ln \frac{\displaystyle\sum_{\mathbf{x}\in\mathbb{X}^{k+}} \left(p(\mathbf{y}|\mathbf{x}) \exp \sum_{j\neq k} \frac{1}{2} x_j \lambda^a_j \right)}{\displaystyle\sum_{\mathbf{x}\in\mathbb{X}^{k-}} \left(p(\mathbf{y}|\mathbf{x}) \exp \sum_{j\neq k} \frac{1}{2} x_j \lambda^a_j \right)} \quad . \tag{2.10}$$

Finally, the max-log approximation to the Jacobian logarithm [68] is used to simplify (2.10) to

$$\begin{aligned} \lambda^e_{k,\text{Max-MAP}} \approx & \frac{1}{2} \max_{\mathbf{x}\in\mathbb{X}^{k+}} \left(-\frac{1}{\sigma_n^2} \left\| \mathbf{y} - \mathbf{H}\mathbf{s}^{k+} \right\|^2 + \mathbf{x}^{\setminus k} \cdot \boldsymbol{\lambda}^{a,\setminus k} \right) \\ & - \frac{1}{2} \max_{\mathbf{x}\in\mathbb{X}^{k-}} \left(-\frac{1}{\sigma_n^2} \left\| \mathbf{y} - \mathbf{H}\mathbf{s}^{k-} \right\|^2 + \mathbf{x}^{\setminus k} \cdot \boldsymbol{\lambda}^{a,\setminus k} \right) \end{aligned} \tag{2.11}$$

where $\mathbf{x}^{\setminus k} \cdot \boldsymbol{\lambda}^{a,\setminus k}$ is a vector dot product.

To calculate the output LLR with MCMC, the max-log MAP calculation of (2.11) is used but with the set $\mathbb{X}$ replaced with the list Z of the sampled permutations. This, with some minor rearrangements leads to

$$\begin{aligned} \lambda^e_{k,\text{MCMC}} \approx & \frac{1}{2} \min_{\mathbf{x}\in Z^{k-}} \left(\frac{1}{\sigma_n^2} \left\| \mathbf{y} - \mathbf{H}\mathbf{s}^{k-} \right\|^2 - \mathbf{x}^{\setminus k} \cdot \boldsymbol{\lambda}^{a,\setminus k} \right) \\ & - \frac{1}{2} \min_{\mathbf{x}\in Z^{k+}} \left(\frac{1}{\sigma_n^2} \left\| \mathbf{y} - \mathbf{H}\mathbf{s}^{k+} \right\|^2 - \mathbf{x}^{\setminus k} \cdot \boldsymbol{\lambda}^{a,\setminus k} \right) \end{aligned} \tag{2.12}$$

where the max() operation has been changed to an equivalent min() operation to put the equation in a more intuitive cost-function minimization form.

2.5 Algorithm Outline

The MCMC MIMO detector algorithm starts with an initial sequence of bits $\mathbf{x}$. It then cycles across the K bits in a bitwise fashion for N_{iter} iterations, using the probabilities calculated with (2.4) and (2.7) to determine state transitions. This process effectively guides the Gibbs sampler towards the more important regions. After a sufficient quantity of samples have been taken in this manner, the algorithm approximates the max-log MAP detector's extrinsic output LLR λ^e_k in (2.11) with (2.12). This procedure is outlined in Algorithm 1 for one Gibbs sampler, though N_{gibbs} can be used in parallel to increase sampling speed and increase sample diversity.

2.6 High SNR Stalling Problem

At high SNR, the MCMC detector derived here is known to stall. This means that the rate at which new samples are generated per iteration approaches zero [69]. This

Algorithm 1 Basic MCMC Gibbs Sampler

```
Initialize x.
for 1 to N_iter do
    for k = 0 to N log2(N_qam)-1 do
        Update min_{x∈Z^{k+}}( ) and min_{x∈Z^{k-}}( ) for (2.12).
        Calculate P_gibbs with (2.4) and (2.7).
        Generate a uniform random variable 0 ≤ r ≤ 1.
        if r < P_gibbs then
            x_k ← +1
        else
            x_k ← -1
Compute output LLR with (2.12).
```

results in slow convergence and an enormous number of iterations needed to reach near MAP performance, potentially more than the calculations needed for MAP itself. To help understand stalling more thoroughly and the improvements proposed in the subsequent sections, a detailed view of the underlying Gibbs samplers is presented that will lead to some insight.

In Fig. 2.1, we see the internal behavior of a pair of randomly initialized Gibbs samplers working on a 4×4 MIMO system with a 16 QAM symbol constellation, thus $4\log_2(16) = 16$ bits per $\mathbf{x}$. The only difference between the subfigures is the magnitude of AWGN noise, 6 dB vs 12 dB E_b/N_0, which is sufficient to strongly instigate the high SNR stalling problem. Each subplot has the bit index k over the horizontal axis from left to right, and the Gibbs iterations starting at the top and descending over time. Accordingly, the Gibbs sampler moves from left to right and top to bottom.

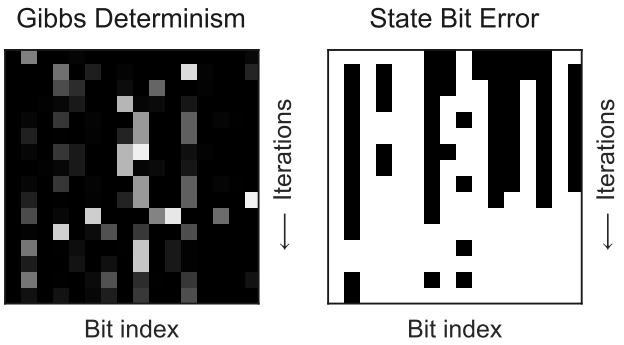
The left subplots show the probability of deterministic and nondeterministic flips where

$$\text{determinism} = |2P_{\text{gibbs}} - 1| \quad . \tag{2.13}$$

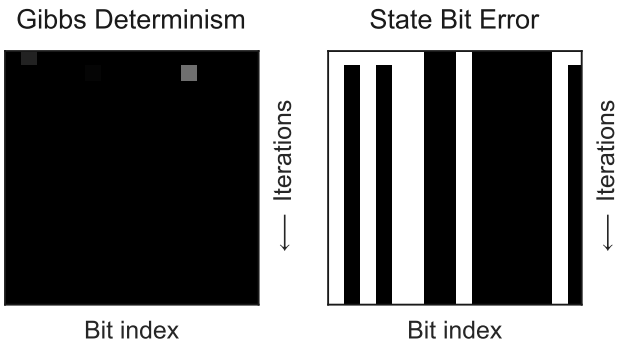
With a gray-scale color mapping determinism = 1 is shown as black (fully deterministic) and determinism = 0 as white (fully random).

The right subplots show the error in the Gibbs state $\mathbf{x}$ when compared to the true transmitted bit sequence, where black indicates a bit error. State error is used instead of simply $\mathbf{x}$ because it simultaneously shows if the state is changing and where the random walk is relative to the transmitted bit sequence.

In Fig. 2.1, we see the curious trend that at low SNR the Gibbs sampler probability is



(a) Correct behavior at low SNR (6 dB E_b/N_0).



(b) Stalling at high SNR (12 dB E_b/N_0).

Figure 2.1. Gibbs sampler detail with the original randomly initialized MCMC algorithm. Both subfigures have the exact same bit sequence, channel, random number generator, and additive noise. The only difference is in noise magnitude which instigates the high SNR stalling problem. Parameters: 4 antennas, 16 QAM, $N_{\text{iter}} = 16$, WiFi TGn Model-D channel.

moderate and the sampler quickly converges to having only 1-bit error, whereas at high SNR the Gibbs probabilities are persistently at extreme values and no change is seen after the first iteration.

Stalling is not seen under all conditions. If a smaller QAM size of 2 to 16 is used at a low SNR, near the BER cliff, stalling will not be observed [70, 71]. Another condition where the full extent of stalling may not be obviously observed is when using well-conditioned, Rayleigh channels with uncorrelated i.i.d. Gaussian channel matrix elements [55]. Real-world channels, with correlation between antennas, display stalling much more strongly than the uncorrelated channels often used in the literature to simulate MCMC performance [18, 20, 24, 43]. There are many variations of the MCMC detector which attempt to mitigate the high SNR stalling problem. Most can be broadly classified as either hybrid or temperature scaling methods.

The idea behind the hybrid approaches is to combine the MCMC detector, which has decreasing performance as SNR increases, with a complementary detector, which has low performance at low SNR but high performance at high SNR. There are many examples of this using the sphere-decoder such as [21, 48, 72, 73] and with MMSE [20]. Further examples include ways of combining successive interference cancellation with MCMC [74, 75]. We note that this class of detectors partially avoids, but does not solve MCMC's high SNR stalling problem.

The temperature scaling class of MCMC detectors recognize that the superficial cause of stalling is that the P_{gibbs} values calculated are too extreme. They suggest that, as the SNR increases, σ_n^2 acts as a gain factor in (2.7), therefore they heuristically include an additional scaling factor α which either linearly scales or replaces σ_n^2 to counteract stalling [23, 24, 76].

Using a constant temperature factor shows a general improvement, but a close inspection of the Gibbs sampler performance shows that the scaling factor is not small enough when the Gibbs sampler first starts, meaning that the walk moves too slowly and deterministically, and is too large when the Gibbs sampler has converged to the region of a solution, meaning the walk moves too quickly and may diverge from the region.

In [19] it was suggested that using a scaling factor which changes size gradually over time may improve performance, but no specific algorithm is provided. The suggested method would predict the average convergence behavior of all Gibbs samplers, and adjust a scaling factor to moderate the probabilities in (2.4) accordingly. Such a method would be nonoptimal as the scaling factor would not be matched to any specific Gibbs sampler which may converge more or less quickly than the average, and thus the scaling factor could reduce rather than improve performance on individual Gibbs samplers.

The randomized-MCMC (R-MCMC) method alternates between using the standard P_{gibbs} calculation and a full random flip [70] to mitigate the high SNR stalling problem. Although not using an explicit temperature factor, the rate at which the randomized probability is used reminds us of pulse-width-modulation (PWM) and therefore on average the effect is similar to temperature scaling. This method has the same basic problems of temperature methods where the parameter is fixed and must be heuristically identified.

Not all of the MCMC methods in the literature fit into the two hybrid and temperature classes. An interesting example is the constrained MCMC method [22, 77]. After using the

standard MCMC detector, it switches to a refinement mode where some of the bits are constrained to specific values. Since this method still suffers from high SNR stalling problem, its sampling efficiency is relatively low.

Stalling will be further explained in the framework of the X-MCMC detector variation in Chapter 3.

2.7 MMSE Initialized MCMC

To better understand the improvements made by the X-MCMC detector that will be introduced in Chapter 3, an MCMC method is needed for comparison. We have selected the MMSE initialized MCMC detector described in [20]. This is a hybrid type of detector since it combines MMSE, which has good performance at high SNR, with MCMC to compensate for the high SNR stalling effect. The reason why we have selected this as apposed to some of the other excellent QRD-hybrid or temperature scaling varieties in the literature is that those methods are generally loosely defined with parameters that need to be heuristically tuned to the application. The MMSE initialized variety is useful as a benchmark because it has an explicit implementation regardless of channel, SNR, number of antennas, or modulation order.

The only extension needed for the MMSE initialized MCMC hybrid algorithm is to initialize one of the parallel Gibbs samplers with the hard decision from an MMSE solution as in

$$\hat{\mathbf{s}}_{\text{MMSE}} = \left(\mathbf{H}^{\dagger}\mathbf{H} + 2\sigma_n^2\mathbf{I}\right)^{-1}\mathbf{H}^{\dagger}\mathbf{y} \tag{2.14}$$

where † is the conjugate transpose. This method solves the issue where the MCMC detector does not converge for small QAM sizes at high SNR. It works because the MMSE solution will result in a correctly signed LLR output without any need of additional MCMC Gibbs iterations. For turbo iterations to work properly and to generate reasonable extrinsic information transfer (EXIT) charts, every turbo iteration has one parallel Gibbs sampler initialized with the original MMSE solution.

As described in Section 2.6, the MMSE initialized MCMC detector does not prevent stalling because it is a hybrid method, thus it is far from MAP performance when higher QAM sizes are used. Examples that show such behavior are presented in Section 3.6.

2.8 Summary

In this chapter, we derived the basic algorithm of the MCMC detector. The high SNR stalling problem was introduced and possible solutions from the literature were reviewed. To help in understanding the stalling problem approaches we broadly classified most of the techniques under hybridization and temperature scaling methods which we found to be useful in understanding their benefits and drawbacks in a broader context. Finally, the MMSE initialized MCMC detector was described for future reference in comparisons against our new X-MCMC detector presented in Chapter 3.

CHAPTER 3

EXCITED MARKOV CHAIN MONTE CARLO

In the previous chapter, the Markov Chain Monte Carlo (MCMC) detector algorithm was derived as presented in much of the literature [19, 24, 43]. These detectors suffer from a characteristic stalling problem that decreases performance as SNR increases. Here, we introduce a revised derivation of the bitwise MCMC detector that we refer to as excited MCMC (X-MCMC). The new approach resolves the stalling problem of MCMC without the need for hybridization with another detector method or adding heuristic temperature scaling factors.

The primary insight needed for the X-MCMC detector is that there are error contributions from both noise and incorrect bit estimates when computing all conditional probabilities. Previous derivations have only included AWGN noise in their derivations. We found that the inclusion of bit error in the statistical model completely solves the high SNR stalling problem covered in Section 2.6. It also provides a method to detect and mitigate poorly converged and overconfident output LLRs that otherwise confuse the decoder, potentially creating error floors in BER curves.

In [55], we use a heuristic explanation to justify the dynamic scaling used in the X-MCMC detector. Those ideas will be expanded on here with a thorough theoretical understanding and some algorithmic improvements. The new X-MCMC detector is shown to have near max-log maximum-a-posteriori (MAP) performance even with challenging, realistic, highly-correlated channels at the maximum MIMO sizes and modulation rates supported by the 802.11ac WiFi specification, 8×8 MIMO 256 QAM.

This chapter is organized as follows. After briefly covering the system model and specialized notation used in this section, we present the derivation of one of our significant contributions, the excited Gibbs sampler, in Section 3.2. Following this mathematical

foundation, we expand on it with conditioning the output LLR confidence in Section 3.3. Then, we introduce the idea of pseudo-convergence and suggest a strategy to detect and remedy it in Section 3.4. An outline of the combined algorithm can be seen in Section 3.5 and simulation results in Section 3.6. Finally, a brief summary is provided in Section 3.7.

3.1 System Model and Notation

In the discussions and derivations that follow, the notation is the same as elsewhere in the dissertation where the system model is represented by

$$\mathbf{y} = \mathbf{H}\mathbf{s} + \mathbf{n} \ . \tag{3.1}$$

Here, $\mathbf{y}$ is the received signal vector, $\mathbf{H}$ is a slow flat fading complex channel matrix containing the pairwise gain and phase between antennas, $\mathbf{s}$ is the vector of transmitted QAM constellation symbols, and $\mathbf{n}$ is a noise vector. The noise elements are assumed to be independent and identically distributed (i.i.d.) complex Gaussian random variables with variance of σ_n^2 per each real and imaginary dimension. Assuming the transmit and receive side have the same number of antennas N, the dimensions of the vectors and matrices are $N \times 1$ and $N \times N$. The complex symbols $\mathbf{s}$ are mapped from the bit vector $\mathbf{x}$, which may be described as comprising of 1's and 0's or equivalently +1's and −1's.

Some specialized notation is used for compactness and clarity. Vectors and matrices are expressed with bold fonts and the latter are capitalized. The removal of the k^{th} element of a vector is shown with set notation as $\{\cdot\}^{\backslash k}$. A variable or vector derived from the bit sequence $\mathbf{x}$ with the k^{th} bit forced to a 1 or 0 is shown with $\{\cdot\}^{k+}$ and $\{\cdot\}^{k-}$, respectively. When two nearly identical equations are needed differing only in use of $k+$ or $k-$, $k\pm$ is used to represent both versions. If the k^{th} bit is forced to the correct transmitted value, either 1 or 0, it is shown with $\{\cdot\}^{k*}$.

The concept of distance is repeatedly used. It is the closeness of the current state $\mathbf{x}$ to the transmitted sequence. To simplify its use, it will be defined as the square Euclidean distance

$$d = \|\mathbf{y} - \mathbf{H}\mathbf{s}\|^2 \tag{3.2}$$

where $\mathbf{s}$ is the complex symbol mapped version of the bit state $\mathbf{x}$. If the k^{th} bit of $\mathbf{x}$ is forced to a 1 or 0, then this can be indicated on all dependent variables and vectors with a

superscript $k+$ or $k-$ as in

$$d^{k\pm} = \left\| \mathbf{y} - \mathbf{H}\mathbf{s}^{k\pm} \right\|^2 \ . \tag{3.3}$$

3.2 Excited Gibbs Sampler

We begin the derivation of the excited Gibbs sampler by recalling the derived Gibbs transition probability of the original MCMC detector in (2.4), (2.5), and (2.7) as

$$P_{\text{gibbs}} = P(x_k = +1 | \mathbf{y}, \mathbf{x}^{\setminus k}, \boldsymbol{\lambda}^a) = \frac{1}{1 + e^{-\gamma_k}} \tag{3.4}$$

where

$$\gamma_k = \ln \frac{p(\mathbf{y} | x_k = +1, \mathbf{x}^{\setminus k})}{p(\mathbf{y} | x_k = -1, \mathbf{x}^{\setminus k})} + \lambda_k^a \ . \tag{3.5}$$

If AWGN noise is the only source of error used for the probability distributions, then we have the typical MCMC detector calculation

$$\gamma_k = \frac{1}{2\sigma_n^2} \left(\left\| \mathbf{y} - \mathbf{H}\mathbf{s}^{k-} \right\|^2 - \left\| \mathbf{y} - \mathbf{H}\mathbf{s}^{k+} \right\|^2 \right) + \lambda_k^a \ . \tag{3.6}$$

The cause of Gibbs sampler stalling in the MCMC detector is constant production of extreme P_{gibbs} values close to 0% or 100%. This creates a nearly deterministic walk which quickly stalls. Extreme probabilities are readily caused by (3.4) and (3.6). At just $|\gamma_k| = 3$ there is a mere 2% chance of a nondeterministic transition. In simulations of 4-antenna 64 QAM systems, γ_k values greater than 10 are commonly seen and values greater than 100 are not unusual at high SNRs.

In the MCMC derivation of Chapter 2, an assumption is implicitly made that the only form of error is from the channel model's AWGN noise. It is a useful assumption because it allows the substitution of a Gaussian probability distribution into (3.5) that simplifies to the explicit γ_k of (3.6). However, this causes problems since the incorrect bits in $\mathbf{x}$ also contribute to error and in general are much larger contributors, often by a factor of 1000 at high SNRs. This provides insight as to why the stalling problem only appears at high SNRs, since at low SNRs the condition is less strongly violated. Therefore, the goal here is to find a way to calculate the needed probabilities in γ_k without this assumption.

The implicit statistical model used for the probability distributions in the original MCMC derivation is

$$\mathbf{n} = \mathbf{y} - \mathrm{H}\mathbf{s} \ . \tag{3.7}$$

But a more accurate model is needed when the Gibbs sampler is far from the true transmitted bit sequence. The error $\mathbf{e}^{k*}$ must include both AWGN interference and the incorrect bits in the state. This is captured with the new model

$$\mathbf{e}^{k*} = \mathbf{y} - \mathrm{H}\mathbf{s}^{k*} = \mathbf{H}\left(\mathbf{s}_{\mathrm{tx}} - \mathbf{s}^{k*}\right) + \mathbf{n} \tag{3.8}$$

where $\mathbf{s}_{\mathrm{tx}}$ is the transmitted symbol vector and $\mathbf{s}^{k*}$ is the current Gibbs state with the k^{th} bit of $\mathbf{x}$ correct, i.e. matches the respective transmitted bit. This model does not include error from the k^{th} bit in $\mathbf{s}$ so that the distributions of the numerator and denominator of (3.5) will be the same.

Therefore, we see that to calculate the probability of $x_k = \pm 1$ we must include conditioning based on the error from the bits $\mathbf{x}^{\setminus k}$ as well as from the AWGN noise. It is not generally possible to separate the contributions of x_k and $\mathbf{x}^{\setminus k}$ to the error. In the steps that follow, we begin by assuming that we have a metric for the error, excluding that caused by the k^{th} bit, and finish with approximations sufficient for development of a detector that works well in practice.

Quantifying the contributions of error only from $\mathbf{x}^{\setminus k}$ and noise is equivalent to having the distance d with the k^{th} bit known to be correct. Thus, we need $d^{k*} = \left\|\mathbf{y} - \mathbf{H}\mathbf{s}^{k*}\right\|^2$. If we assume that the combined bits in error and noise represented by $\mathbf{e}^{k*}$ is a Gaussian random vector with variance σ_{k*}^2 per dimension we have

$$p(\mathbf{y}|x_k = \pm 1, \mathbf{x}^{\setminus k}) = \left(2\pi\sigma_{k*}^2\right)^{-N} \exp\left(-\frac{\left\|\mathbf{y} - \mathbf{H}\mathbf{s}^{k\pm}\right\|^2}{2\sigma_{k*}^2}\right) \ . \tag{3.9}$$

Then, by substituting this Gaussian model into (3.5), we have

$$\gamma_k = \frac{d^{k-} - d^{k+}}{2\sigma_{k*}^2} + \lambda_k^a \ . \tag{3.10}$$

A single sample estimate of the error variance σ_{k*}^2 can be made by using an *oracle* with knowledge of d^{k*}.

$$\sigma_{k*}^2 \approx \frac{d^{k*}}{2N} \tag{3.11}$$

The challenge is to make an approximation of σ_{k*}^2 without directly having d^{k*} which is unknown to the receiver.

An estimate $\hat{d}^{k*}$ of d^{k*} can be made with the weighted average of d^{k+} and d^{k-}.

$$\begin{aligned}\hat{d}^{k*} &= d^{k+}P(x_k = +1|\mathbf{x}^{\backslash k}, \mathbf{y}) + d^{k-}P(x_k = -1|\mathbf{x}^{\backslash k}, \mathbf{y}) \\ &= \frac{d^{k+}p(\mathbf{y}|x_k = +1, \mathbf{x}^{\backslash k})P(x_k = +1|\mathbf{x}^{\backslash k})}{p(\mathbf{y}|\mathbf{x}^{\backslash k})} \\ &\quad + \frac{d^{k-}p(\mathbf{y}|x_k = -1, \mathbf{x}^{\backslash k})P(x_k = -1|\mathbf{x}^{\backslash k})}{p(\mathbf{y}|\mathbf{x}^{\backslash k})} \\ &= \frac{d^{k+}p(\mathbf{y}|x_k = +1, \mathbf{x}^{\backslash k}) + d^{k-}p(\mathbf{y}|x_k = -1, \mathbf{x}^{\backslash k})}{2p(\mathbf{y}|\mathbf{x}^{\backslash k})} \end{aligned} \tag{3.12}$$

where Bayes' rule is first applied and then $P(x_k = \pm 1|\mathbf{x}^{\backslash k}) = 1/2$ because of the independence between bits. This estimate can be bounded by the minimum and mean of $d^{k\pm}$ since $P(x_k = -1|\mathbf{x}^{\backslash k}, \mathbf{y}) = 1 - P(x_k = +1|\mathbf{x}^{\backslash k}, \mathbf{y})$ and the larger probability will correspond to the smaller distance.

$$\hat{d}^{k*}_{\min} = \min\left(d^{k+}, d^{k-}\right) \leq \hat{d}^{k*} \tag{3.13}$$

$$\hat{d}^{k*} \leq \frac{1}{2}\left(d^{k+} + d^{k-}\right) = \hat{d}^{k*}_{\text{mean}} \tag{3.14}$$

These bounds are useful rough approximations to $\hat{d}^{k*}$ as they are readily available to the Gibbs sampler and are computationally efficient. We will refer to these as the *min* and *mean* approximations.

A more accurate approximation to the weighted average of (3.12) can be made by first replacing the denominator, which would require the unknown $\mathbf{x}^{k*}$, with the average of the probabilities with $x_k = \pm 1$.

$$p(\mathbf{y}|\mathbf{x}^{\backslash k}) = \frac{1}{2}\left(p(\mathbf{y}|x_k = +1, \mathbf{x}^{\backslash k}) + p(\mathbf{y}|x_k = -1, \mathbf{x}^{\backslash k})\right) \tag{3.15}$$

Then, single sample estimates of the variances $\sigma^2_{k\pm} \approx d^{k\pm}/2N$ are used to approximate the Gaussian distributions

$$\begin{aligned} p(\mathbf{y}|x_k = \pm 1, \mathbf{x}^{\backslash k}) &\approx \left(2\pi\sigma^2_{k\pm}\right)^{-N} \exp\left(-\frac{d^{k\pm}}{2\sigma^2_{k\pm}}\right) \\ &\approx \left(\pi\frac{d^{k\pm}}{N}\right)^{-N} \exp\left(-\frac{d^{k\pm}}{d^{k\pm}/N}\right) \quad . \end{aligned} \tag{3.16}$$

This estimate is too crude to be used directly in (3.5), but by using (3.15) and (3.16), an approximation of (3.12) can be made

$$\begin{aligned} \hat{d}^{k*}_{\text{weighted}} &= \frac{d^{k+}(d^{k+})^{-N} + d^{k-}(d^{k-})^{-N}}{(d^{k+})^{-N} + (d^{k-})^{-N}} \\ &= \frac{d^{k+} + d^{k-}(d^{k+}/d^{k-})^{N}}{1 + (d^{k+}/d^{k-})^{N}} \quad . \end{aligned} \tag{3.17}$$

For comparison, it is useful to include the original MCMC method in Chapter 2 in this framework. It implicitly uses the approximation

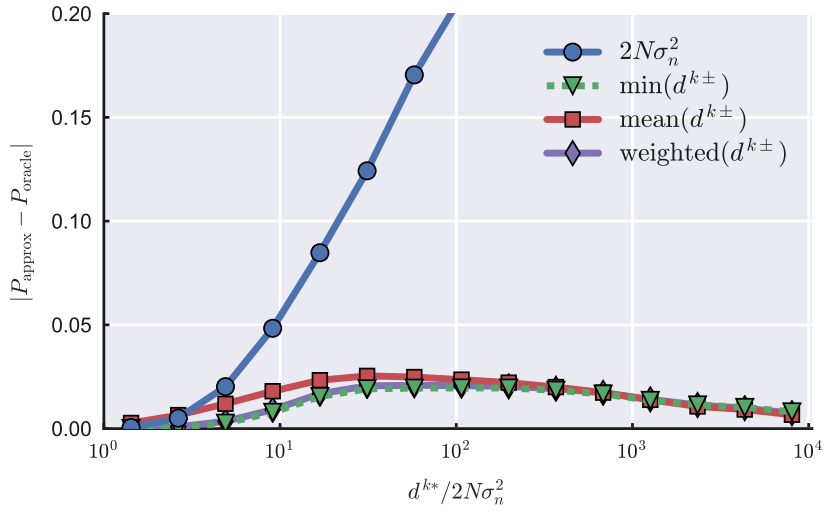
$$\hat{d}^{k*}_{\text{original}} = 2N\sigma_n^2 \quad . \tag{3.18}$$

The result of using the *min* (3.13), *mean* (3.14), *weighted* (3.17), and *original* (3.18) estimates of d^{k*} to generate P_{gibbs} is shown in Fig. 3.1a, where the error is the mean of $|P_{\text{approx}} - P_{\text{oracle}}|$ in each x-axis bin. Based on these plots the new approximations are a vast improvement over the *original* method, but it may not be clear which should be used in practice. To make a selection, we need to understand how error in approximating P_{gibbs} affects the Gibbs sampler.

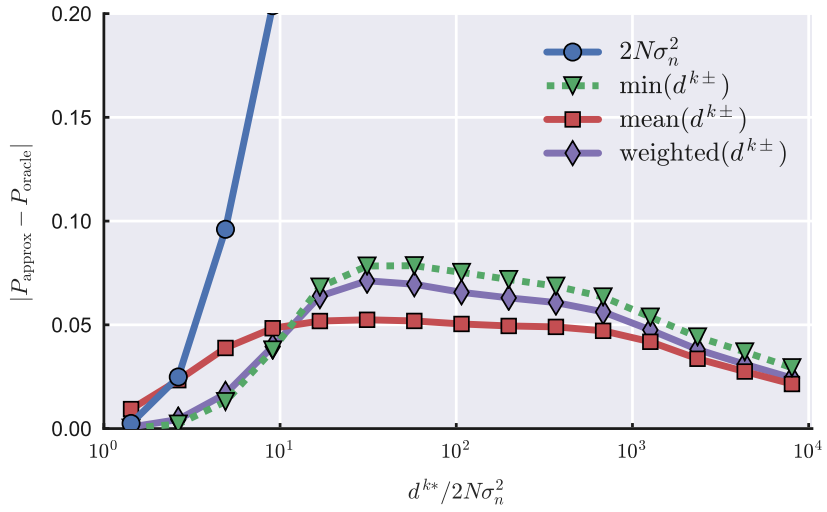
Calculations that underestimate or overestimate d^{k*} have specific qualitative effects on the Gibbs sampler's behavior. By acting as a scaling factor applied to the difference $(d^{k-} - d^{k+})$ of (3.10), the estimation error will tend to either make the Gibbs sampler more deterministic, by pushing probabilities to the 0% and 100% extremes, or make the Gibbs sampler more random, by moderating probabilities towards 50%. Therefore, underestimates like *min* and *original* are more deterministic with a cooler/slower movement, and overestimates like *mean* are more random with a hotter/faster movement. The symptom of a large underestimate is stalling whereas for large overestimates it is failing to converge to important regions. This observation is similar to the observations in [24] about how nonoptimal temperature scaling affects convergence time. These effects are difficult to differentiate within BER curves since in both cases performance goes down and more iterations are needed. In Sections 2.6 and 3.4, we introduce a way to visualize the internal state of the Gibbs sampler sufficient to distinguish between these types of errors and others.

Using this understanding of the effects of under and over estimates in P_{gibbs}, it is clear why the original MCMC method in Chapter 2 has a stalling problem. Since it always underestimates $\hat{d}^{k*}$ it creates a nearly deterministic, gradient descent style Gibbs sampler walk which quickly stalls at the first local minima. At low SNRs the underestimate is less severe so the algorithm works but is less efficient.

To help select the best approximation, Fig. 3.1b shows the mean error of only the worst 20% of samples in each bin for each method. The *mean* method has the worst performance at small distances since it overweights the larger of $d^{k\pm}$, making the sampler too random.



(a) P_{gibbs} approximation error including all samples.



(b) P_{gibbs} approximation error including only worst 20% of samples in each bin for each method to emphasize differences.

Figure 3.1. Comparison of using various d^{k*} approximations to calculate P_{gibbs} with (3.4), (3.10), and (3.11). Samples are generated by a set of Gibbs samplers using $P_{\text{gibbs}} = P_{\text{oracle}}$. Each plotted data point is the mean error over a range of x-axis values, i.e. average per bin. Parameters: 4 antennas, 64 QAM, $N_{\text{gibbs}} \times N_{\text{iter}} = 24 \times 24$, WiFi TGn Model-D channel, 22dB E_b/N_0.

The *min* method has worse performance at large distances because it overweights the smaller of $d^{k\pm}$, making the sampler too deterministic. The *weighted* method performs slightly better than *min*, but the large number of multiplications and divisions makes the added complexity too large for the small improvement. In practice we have found that having more accurate approximations at low distance is desirable because it results in slower sampling near important regions whereas overestimates such as *mean* tend to diverge from important regions too quickly. Therefore the *min* approximation is selected for σ^2_{k*} in calculating

$$\gamma_k = \frac{d^{k-} - d^{k+}}{\hat{d}^{k*}_{\min}/N} + \lambda^a_k \quad . \tag{3.19}$$

This selection will be reaffirmed in Section 3.6.1 with BER curves, but first, output LLR conditioning and pseudo-convergence enhancements must be introduced so that the combined effects can be accounted for.

3.3 Output LLR Overconfidence and Conditioning

From the original MCMC algorithm derivation in Chapter 2, we recall that the output LLR can be calculated with (2.12) as in

$$\begin{aligned}\lambda^e_{k,\mathrm{MCMC}} \approx & \frac{1}{2}\min_{\mathbf{x}\in Z^{k-}}\left(\frac{1}{\sigma_n^2}\left\|\mathbf{y}-\mathbf{H}\mathbf{s}^{k-}\right\|^2 - \mathbf{x}^{\setminus k}\cdot\boldsymbol{\lambda}^{a,\setminus k}\right) \\ & -\frac{1}{2}\min_{\mathbf{x}\in Z^{k+}}\left(\frac{1}{\sigma_n^2}\left\|\mathbf{y}-\mathbf{H}\mathbf{s}^{k+}\right\|^2 - \mathbf{x}^{\setminus k}\cdot\boldsymbol{\lambda}^{a,\setminus k}\right) \quad .\end{aligned} \tag{3.20}$$

It uses the list $Z \subset \mathbb{X}$ of sampled bit permutations as an approximation to the full permutation list used by the MAP detector in (2.11). This is the method used by list based MIMO detectors such as MCMC [19], list sphere-decoding [35], and K-Best [18]. Notice that in order to use this method, one must have a list which accurately represents the statistics of the full permutation list. If the list is too short or poorly selected the statistics break down, the calculation becomes unreliable, and the output LLRs can be wildly inaccurate.

One can state that a requirement of the algorithm is for a sufficient number of samples to be taken, but there are two reasons why this is undesirable in practice. First, an implementable design has the goal of minimizing computational complexity and therefore real-world implementations need to use the minimum number of samples possible. As a result, a minority of realizations are likely to be poorly converged, leading to invalid output statistics. Second, many modern communication systems have multiple channel realizations

per codeword, for example OFDM. When there are multiple channel realizations, some will be more ill-conditioned than others with a longer convergence time and therefore requiring more Gibbs iterations than the average. During our analysis we have observed situations where the worst channel, often during a deep fade, requires 10x more iterations than average while using a WiFi TGn Model-D channel model, 4 antennas, 64 QAM, a WiFi OFDM system using 52 active subcarriers, and an LDPC 3/4 code. Using 10x more iterations consistently is undesirable, but if the slowly converging realizations are halted early while still statistically unstable, then their large incorrect soft-output values can easily corrupt the entire codeword despite comprising a small number of bit errors.

In the context of suboptimal iterative detection and decoding schemes, a constant positive scaling coefficient less than 1 is sometimes used to scale the extrinsic LLRs [78–82]. This has the effect of removing divergence and thus increase stability. Therefore, an alternative to using a large list to deal with outlier difficult channel realizations is to decrease the confidence of their respective LLRs so that they no longer have the strength to corrupt the entire codeword. Unlike other scaling methods which suggest a heuristic constant scaling coefficient to reduce LLR confidence, we propose using the statistics of the sample list to compute a dynamic coefficient that may be used with any list based detector.

If we allow that the sample list may be of poor quality, then there are two contributions to error: the AWGN noise in the system model and the quality of the list. The quality of the list can be computed using the probability that the list contains a sample that is the true transmitted signal without error. As in Section 3.2, if we assume that the combination of both forms of error are Gaussian distributions then the output LLR can be changed to

$$\begin{aligned}\lambda^{e}_{k,\text{X-MCMC}} \approx & \frac{1}{2}\min_{\mathbf{x}\in \mathrm{Z}^{k-}}\left(\frac{1}{\sigma_z^2}\left\|\mathbf{y}-\mathbf{H}\mathbf{s}^{k^-}\right\|^2 - \mathbf{x}^{\backslash k}\cdot\boldsymbol{\lambda}^{a,\backslash k}\right) \\ & -\frac{1}{2}\min_{\mathbf{x}\in \mathrm{Z}^{k+}}\left(\frac{1}{\sigma_z^2}\left\|\mathbf{y}-\mathbf{H}\mathbf{s}^{k^+}\right\|^2 - \mathbf{x}^{\backslash k}\cdot\boldsymbol{\lambda}^{a,\backslash k}\right) \end{aligned} \tag{3.21}$$

where σ_z^2 represents the variance of the combined contributions of both AWGN noise and list error. This can be estimated with the minimum distance sample in the list similarly to (3.11) using $\hat{d}^{k*}_{\min}$. That is,

$$\sigma_z^2 \approx \frac{1}{2N}\min_{\mathbf{x}\in \mathrm{Z}}\left(\|\mathbf{y}-\mathbf{H}\mathbf{s}\|^2\right) \quad . \tag{3.22}$$

Note that additionally we should limit $\sigma_z^2 \geq \sigma_n^2$. This protects against a sample over-fitting

the noise and therefore having a distance less than the expectation of correct bit sequences, $2N\sigma_n^2$, which would incorrectly create overconfidence in the output LLR.

In Fig. 3.2, the effects of output conditioning on a short set of realizations, where a smaller than normal X-MCMC detector is used to increase the occurrence of poor convergence. Several important things should be noted in these plots. First, a large final distance (row 2) is highly correlated to bit error (row 1), where final distance is calculated using the bit sequence from a hard-decision on the output LLR. When there are bit errors and the distance is low, the output LLR of incorrect bits should be small, indicating a low confidence in the incorrect hard-decision (row 4) which is seen to be universally true for MAP. Realization indexes 14 and 24 are clear examples of the X-MCMC detector failing to converge and producing dramatic overconfidence on incorrect bits. Passing this invalid information to a FEC decoder will propagate to create many more errors, possibly corrupting the entire codeword. Notice that the scaled version, using (3.21) in place of (3.20), suppresses the overconfidence on incorrect bits (row 4) while having minimal impact on the overall LLR (row 3).

The corruption of codewords by rare poorly converged realizations causes an error floor to appear on BER curves, as will be shown in Section 3.6.3. Using output LLR conditioning reduces this effect and allows a short, fixed number of iterations to be used stably and reliably in real-world applications.

We also note that since the output conditioning is derived using the sample list, this method is applicable to other list based algorithms such as list sphere-decoding and K-Best.

3.4 Pseudo-Convergence

Now that the high SNR stalling problem is solved by the X-MCMC detector and the Gibbs sampler moves efficiently at all SNR levels, a new issue is encountered. The Gibbs sampler may stop moving due to an effect referred to as pseudo-convergence. This problem appears to be less understood in MIMO communications applications, but has been noted in the wider MCMC literature [51]. Although symptomatically similar, it is different from the stalling problem discussed previously. It occurs when the posterior probability distribution is multi-modal with weak connections between modes, that is, important regions are weakly connected to other important regions through very low probabilities. Thus, the Gibbs sampler

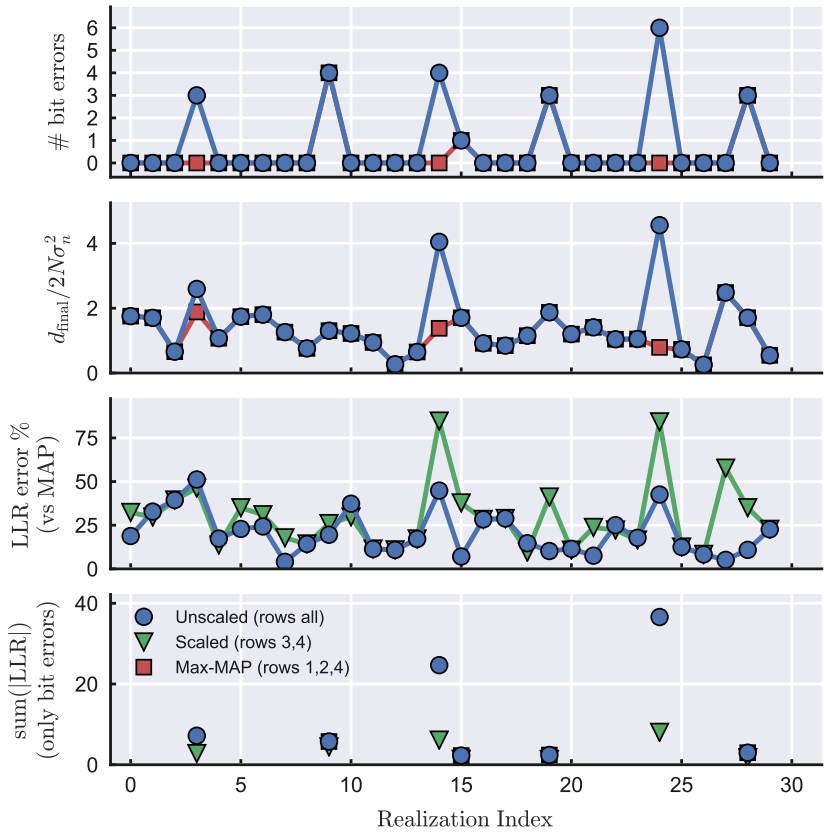


Figure 3.2. Impact of LLR output conditioning on X-MCMC where unscaled is (3.20) and scaled is (3.21). Extrinsic mutual information quality: $I_{e,\text{unscaled}}$=.82, $I_{e,\text{scaled}}$=.91, $I_{e,\text{MAP}}$=.96 (see [1] and Fig 3.7). Parameters: 4 antennas, 64 QAM, $N_{\text{gibbs}} \times N_{\text{iter}} = 22 \times 22$, WiFi TGn Model-D channel, 19dB E_b/N_0.

may stay in one mode (i.e. similar bit permutations connected with sufficient probability) and collect highly correlated samples. When encountered, pseudo-convergence decreases the sampling efficiency of MCMC, resulting in a need for a large number of iterations and/or parallel Gibbs samplers. Others appear to be encountering this effect as reports of performance improvements when adding random walk restarting have been observed [70, 76].

3.4.1 Gibbs Detail Plots

To understand the pseudo-convergence phenomena more thoroughly, we have developed the Gibbs details plots of Fig. 3.3, 3.4, and 3.5 introduced partially in Section 2.6. For consistency, these figures all use the same input data and initialized states. The first row shows information on a single Gibbs sampler whereas the second row shows information on

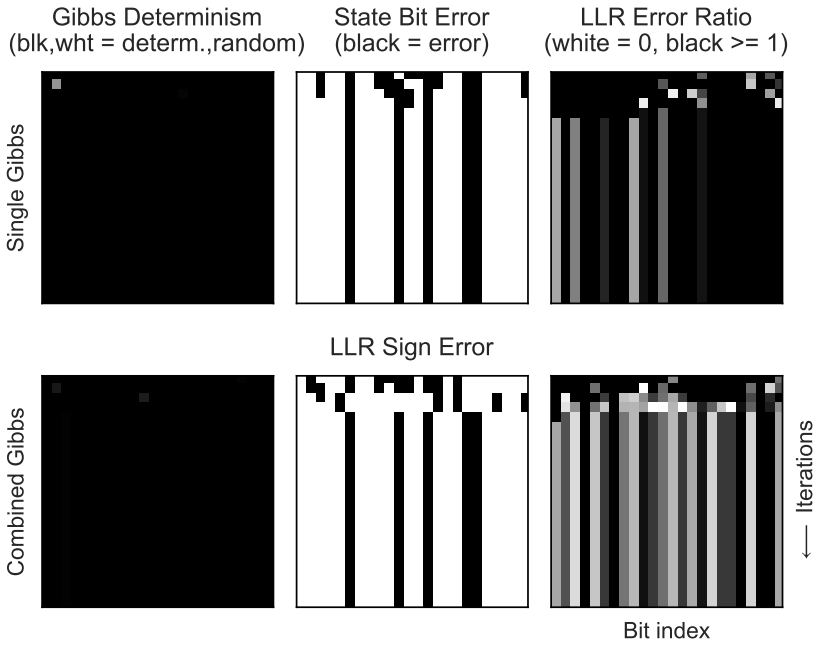


Figure 3.3. Detail on internal MCMC behavior with the original random initialized MCMC. Notice high SNR stalling problem. Parameters: 4 antennas, 64 QAM, $N_{\text{gibbs}} \times N_{\text{iter}} = 6 \times 24$, WiFi TGn Model-D channel, 22dB E_b/N_0.

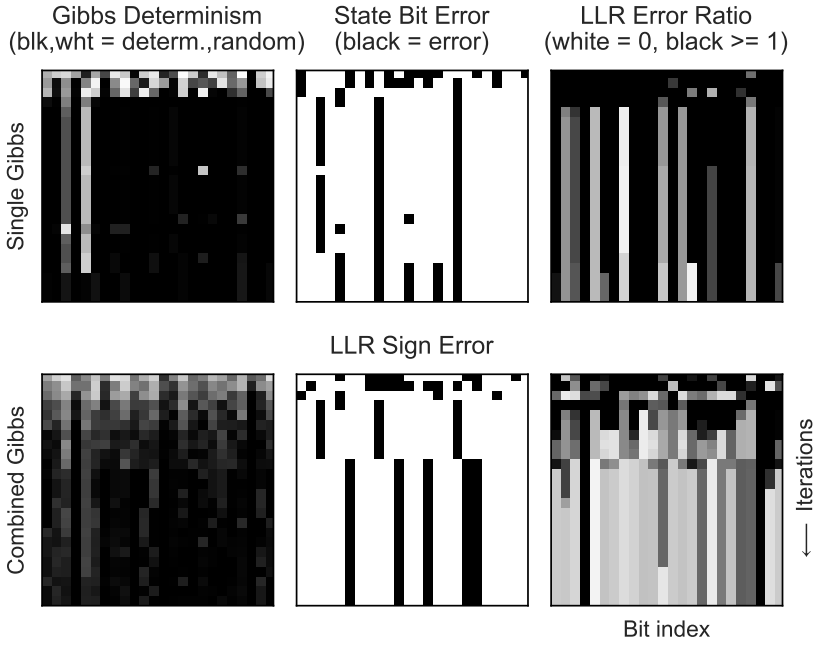


Figure 3.4. Detail on internal MCMC behavior with partial X-MCMC ('x - -' = Gibbs excitement only). Stalling fixed but pseudo-convergence stopping is present. Parameters: 4 antennas, 64 QAM, $N_{\text{gibbs}} \times N_{\text{iter}} = 6 \times 24$, WiFi TGn Model-D channel, 22dB E_b/N_0.

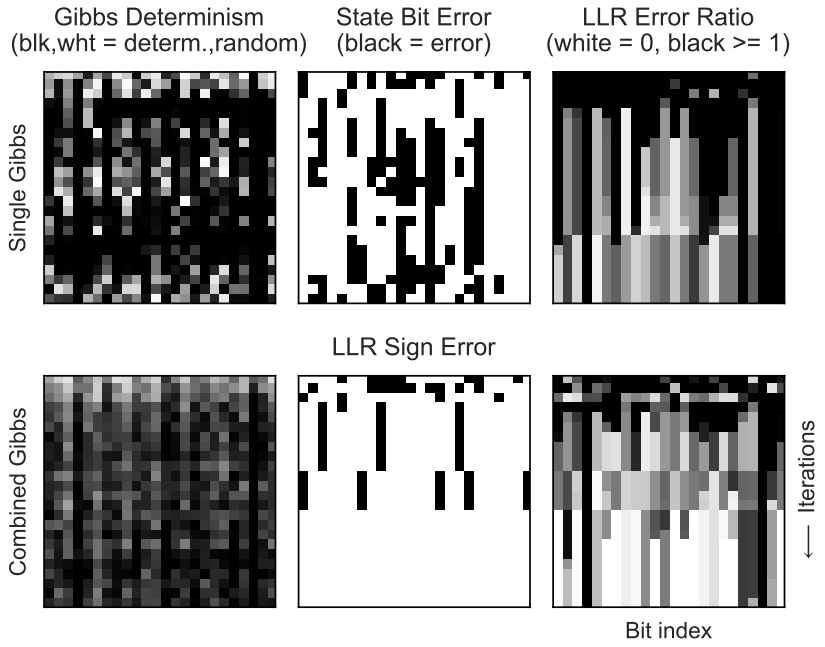


Figure 3.5. Detail on internal MCMC behavior with partial X-MCMC ('x - p' = Gibbs excitement and pseudo-convergence stopping mitigation). Stalling and stopping resolved. Parameters: 4 antennas, 64 QAM, $N_{\text{gibbs}} \times N_{\text{iter}} = 6 \times 24$, WiFi TGn Model-D channel, 22dB E_b/N_0.

the combination of all parallel Gibbs samplers.

In the first row, the first subplot shows the likelihood of a nondeterministic flip where

$$\text{determinism} = |2P_{\text{gibbs}} - 1| \quad . \tag{3.23}$$

The second subplot is the error compared to the transmitted bit sequence, and the third subplot shows the relative amount of error in the output LLR using the single Gibbs sampler. The normalized LLR error,

$$\text{LLR error ratio} = |\lambda_k^e - \lambda_{k,\text{MAP}}^e| / \underset{k}{\text{mean}} \left(|\lambda_{k,\text{MAP}}^e| \right) \tag{3.24}$$

is calculated against the optimal max-log MAP solution. The max-log MAP solution is what MCMC should converge to if run for an infinite number of iterations. In Fig. 3.3, 3.4, and 3.5, the LLR error ratio values are shown with a gray-scale color mapping, 0 to 1 as white to black where values above one saturate to black.

The second row shows a combined view of all of the N_{gibbs} parallel Gibbs samplers. The first subplot is the average determinism, whereas the second and third are the LLR sign error relative to Max-MAP and the LLR error ratio from (3.24).

The desired behavior of the first column is to show signs of the random walk being guided with a variation of determinism, not fully random or deterministic. In the second column, the single Gibbs sampler should not stay converged to any state and instead should continuously explore the state space, whereas the combined Gibbs samplers should converge to no LLR sign error relative to Max-MAP. Finally, the third column should continue to converge to the Max-MAP solution, displayed by white.

Fig. 3.3 shows that the original MCMC detector described in Chapter 2 is strongly stalled at this high SNR, is almost completely deterministic, and does not improve the output LLR after only a few iterations.

By using the excited Gibbs sampler described in Section 3.2, Fig. 3.4 shows a large improvement in behavior. It is no longer stalled on the first step and after a few Gibbs iterations the algorithm has mostly converged. In the second row, after an initial very active period, the guided random walk slows as all parallel samplers become locked into an isolate posterior mode due to pseudo-convergence.

3.4.2 Detection and Escape

As pseudo-convergence is a byproduct of the structure of the posterior distribution, the algorithm is behaving correctly and as expected. Given enough iterations a single Gibbs sampler will eventually leave an isolated mode and sample others. Instead of waiting for the low probability of transition, we prefer a more computationally efficient method of detecting when pseudo-convergence has occurred and then forcing state divergence. This allows the algorithm to collect more unique samples with fewer iterations, thus improving the sampling efficiency of the MCMC detector.

Two effective and computationally efficient methods to detect pseudo-convergence include what we refer to as the *distance* and *motion* methods. The *distance* method tracks the best (smallest) distance d sampled over time, including both d^{k+} and d^{k-}. If this distance does not improve in N_{motion} steps, then pseudo-convergence is detected. Alternatively, the *motion* method detects no change in Gibbs state $\mathbf{x}$ for N_{motion} steps.

The choice of using the $\hat{d}^{k*}_{\min}$ estimate in Section 3.2 causes the Gibbs sampler to move slightly more slowly and deterministically, thus we have found that it tends to stop moving when in pseudo-convergence, therefore using the *motion* pseudo-convergence detection strategy works well with the choice of $\hat{d}^{k*}_{\min}$. For the detection threshold we use $N_{\text{motion}} = N \log_2(N_{\text{qam}})$ steps which is one full Gibbs iteration.

Once pseudo-convergence is detected, the most robust solution is to restart the Gibbs sampler with a new random state. This is a solution mentioned in the wider MCMC literature beyond MIMO communications applications [51]. A drawback of the full restart approach is that it requires reinitialization of the Gibbs sampler which may be an expensive and time consuming operation in VLSI implementations. We have found that a full restart is not necessary in a bitwise MCMC detector. Forcing a 1-bit state change immediately following pseudo-convergence detection generally results in a cascade of state changes. This can be thought of as adding an impulse of energy or excitation to the random walk which assists the Gibbs sampler in escaping the isolated posterior mode. Using this strategy incurs no additional complexity from reinitializing the sampler and is trivial to implement in VLSI designs.

By adding the *motion* based pseudo-convergence detection along with the 1-bit forced-flip method to the excited Gibbs sampler we see the results presented in Fig. 3.5. Here, we see that both the stalling problem seen in Fig. 3.3 and the stopping problem seen in Fig. 3.4 are now resolved. Both the single and combined determinism show that the MCMC algorithm is consistently excited. Combined LLR quickly converges to a correct output bit sequence and then continues to improve the output LLR until near Max-MAP performance is achieved.

3.5 X-MCMC Algorithm Outline

In practice, the X-MCMC algorithm is executed similarly to the original MCMC detector outlined in Algorithm 1. The changes include the use of the newly introduced dynamic scaling of $(d^{k-} - d^{k+})$, pseudo-convergence detection and escape, and output conditioning applied to $\lambda^{e}_{k,\text{X-MCMC}}$. The full X-MCMC algorithm is outlined in Algorithm 2 with the necessary equations specified. Note that many Gibbs samplers may be used in parallel to increase sampling speed and increase sample diversity. When using parallel Gibbs samplers each $\hat{d}^{k*}_{\min}$ should be calculated independently for each Gibbs sampler whereas the output

LLR should be calculated with a list Z comprising a combination of all samplers. Moreover, σ_z^2 in (3.22) should be obtained by taking the minimum of $\hat{d}_{\min}^{k*}$ of all the Gibbs samplers.

Algorithm 2 X-MCMC Gibbs Sampler

```
Initialize x.
for 1 to N_iter do
    for k = 0 to N log2(N_qam)-1 do
        Calculate d^{k+} and d^{k-} with (3.3).
        Update min_{x∈Z}( ) for (3.22).
        Update min_{x∈Z^{k+}}( ) and min_{x∈Z^{k-}}( ) for (3.21).
        Calculate P_gibbs with (3.4) and (3.19).
        if no state change in N_c steps then
            x_k ← ∼x_k
        else
            Generate a uniform random variable 0 ≤ r ≤ 1.
            if r < P_gibbs then
                x_k ← +1
            else
                x_k ← -1
Compute output LLR with (3.21), (3.22).
```

3.6 Results

3.6.1 Excitation Approximations

The previous sections detail the development of the X-MCMC algorithm to solve the high SNR stalling problem and improve output LLR quality. This method was extended with pseudo-convergence enhancements which improves sampling efficiency. We may now revisit the selection of $\hat{d}_{\min}^{k*}$ made in Section 3.2. The reason for doing this verification after the development of the other X-MCMC components and enhancements is that their combined interaction affects the final choice. Therefore, in Fig. 3.6 we show BER curves comparing the potential approximations to d^{k*} while also using the output LLR conditioning and pseudo-convergence enhancements. Both the first and final turbo iterations need to be shown to check for issues with the σ_{k*}^2 scaling relative to the prior $\boldsymbol{\lambda}^a$ in (3.19). As expected, the BER curves show that all of the new approximations perform well and will approach near MAP performance given sufficient iterations. The question is which is the most efficient. The *min* and *weighted* methods are slightly superior to *mean*, but the *weighted* method has much greater computational complexity making it a poor choice to use in

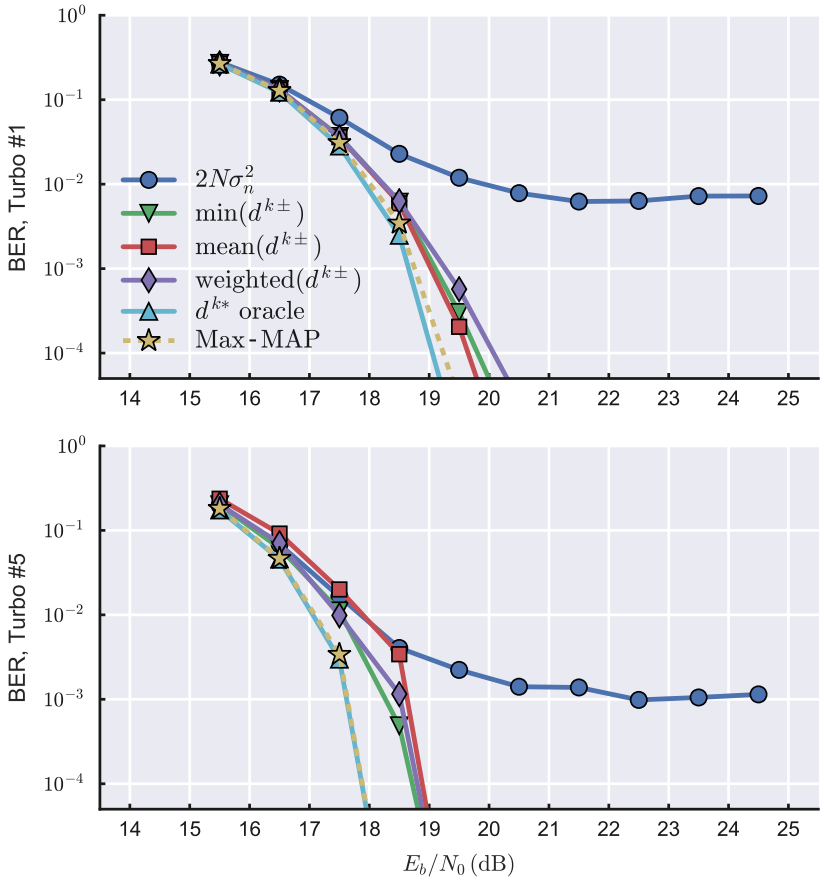


Figure 3.6. Comparison of d^{k*} approximations. This simulation includes the pseudo-convergence mitigation but not the output LLR conditioning technique. Showing with and without turbo loops here is important because it shows the effects with and without *a priori* information. 4 antennas, 64 QAM, TGn-D channel model, $N_{\text{Gibbs}} \times N_{\text{iter}} = 42 \times 42$.

real-world implementations. Therefore $\hat{d}^{k*}_{\min}$ is verified as the preferred approximation for d^{k*} as suggested by the analysis in Section 3.2.

3.6.2 EXIT Chart

To deepen the understanding of the X-MCMC detector, an extrinsic information transfer (EXIT) chart is presented in Fig. 3.7. EXIT charts are commonly used to evaluate the performance of detectors and decoders [1, 83–85]. As expected, the output extrinsic information I_e of the X-MCMC detector improves as the excited Gibbs sampler, output LLR conditioning, and pseudo-convergence enhancements are included in the algorithm. With 48 parallel Gibbs samplers run for 48 iterations (48×48), it is indistinguishable from Max-MAP.

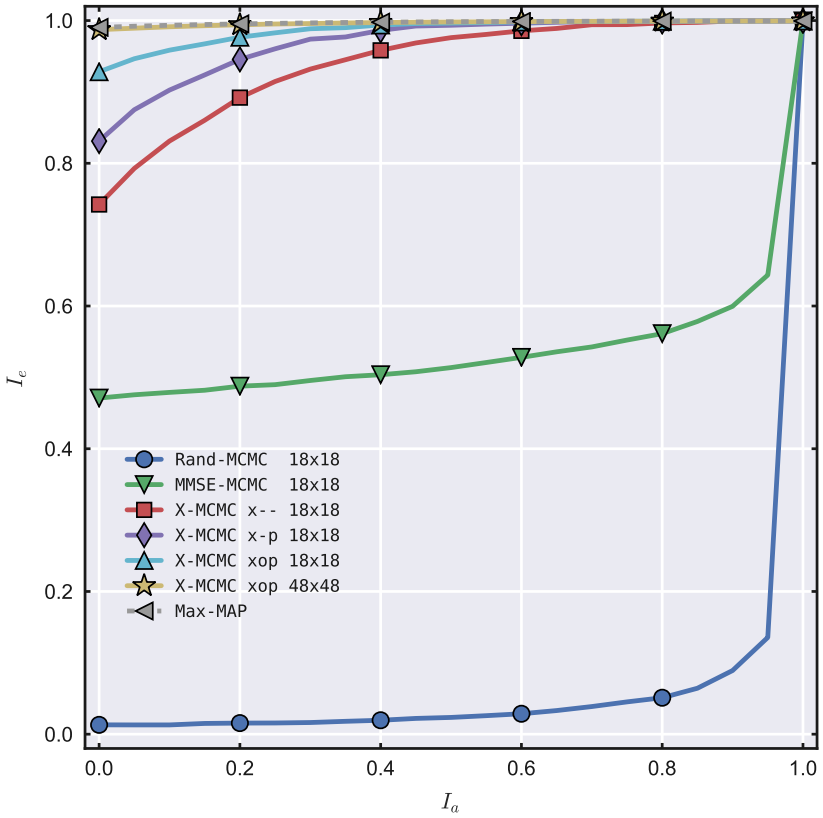


Figure 3.7. EXIT chart showing random and MMSE initialized original MCMC methods versus X-MCMC components ('x o p' are flags representing inclusion of Gibbs excitation, output LLR conditioning, and pseudo-convergence mitigation). Parameters: 4 antennas, 64 QAM, WiFi TGn Model-D channel, 22dB E_b/N_0, WiFi 3/4 LDPC encoding.

The introduction of the excited Gibbs sampler is the most important contribution as it fixes the strange EXIT curve shapes presented by the original and MMSE initialized MCMC methods, caused by high SNR stalling. Their prolonged, flat shape with a sharp rise at the end of the curves is produced by the large d^{k*} underestimate from using (3.18). This means that the $(d^{k-} - d^{k+})/(\hat{d}^{k*}_{\text{original}}/2N)$ is far overweighted compared to the prior $\boldsymbol{\lambda}^a$ in (3.10). Once $I_a > 0.8$ the prior becomes strong enough to overcome the imbalance at this SNR. Using an MMSE initialization does not change the problem, it only provides the algorithm with a good starting point which it may never leave.

3.6.3 BER Curves

Now with the X-MCMC MIMO detector and enhancements fully described, its high performance may be demonstrated under various conditions and some interesting observations can be made. All BER plots include an MMSE initialized MCMC detector for reference as described in Section 2.6. The random initialized version is not generally included as it performs worse than the MMSE initialized version under all conditions. Max-MAP is shown as the optimal performance bound when possible. By using a highly optimized GPU implementation we are able to compute MAP up to 4 antennas with 64 QAM. For the 8 antenna with 256 QAM case we use a very large K-Best as an approximation of Max-MAP limit since it is known to have near MAP performance [18]. A full comparison between X-MCMC and K-Best is outside the scope of this dissertation. One moderate sized K-Best is generally included so that the reader may do some initial comparisons with the literature on K-Best.

When using the WiFi channel model we generate a single time domain realization of the channel for each packet, convert it to the frequency domain, extract the 52 active subcarriers used in a 20MHz WiFi packet, and repeat the same 52 extracted channels until a full LDPC codeword is filled. Creating the channels for a codeword from one time-domain realization in this way creates samples with more extreme, difficult conditions.

The BER curves of Fig. 3.8 confirm the relationships shown in the EXIT chart of Fig. 3.7. There is an incremental improvement in performance as each of the Gibbs excitation, pseudo-convergence enhancement, and output LLR conditioning are included. As predicted by the EXIT chart, a 48×48 X-MCMC detector achieves near Max-MAP performance, and a smaller 30×30 detector is within 1dB. The most interesting feature of these curves is the error floor seen in the X-MCMC curves without output LLR conditioning. This is caused by rare realizations with slow convergence that poorly converge with the fixed number of Gibbs iterations provided. The LLR overconfidence in the poorly converged cases are capable of corrupting entire codewords even when representing a small minority of realizations. For more details on how output LLR conditioning resolves this effect see Section 3.3.

The most important observation in the remaining BER figures is that the X-MCMC detector is capable of achieving near Max-MAP performance under all conditions tested. This is especially impressive at the maximum 802.11ac WiFi protocol size of 8 antenna

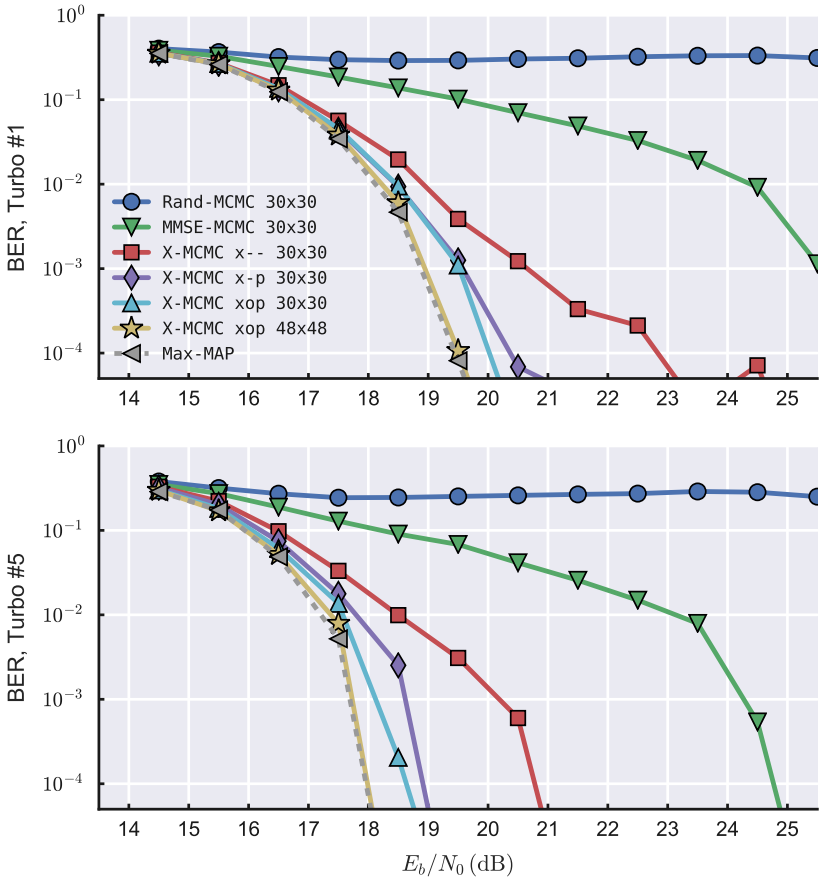


Figure 3.8. BER curves showing original MCMC methods versus X-MCMC components ('x o p' are flags representing inclusion of Gibbs excitation, pseudo-convergence mitigation, and output LLR conditioning). Parameters: 4 antennas, 64 QAM, WiFi TGn Model-D channel, WiFi 3/4 LDPC encoding.

MIMO with 256 QAM modulation shown in Fig. 3.9. Compared to MMSE-Initialized MCMC there is a massive > 6 dB improvement. Similar results are seen in Fig. 3.10 with X-MCMC again achieving Max-MAP performance.

It is relatively easy to achieve near Max-MAP performance on low-order modulation with low-SNR, as seen in Fig. 3.11 and reported in [20] and [54]. Though MMSE-MCMC works under these conditions, it is at a lower efficiency than X-MCMC. This is predicted by our excited Gibbs derivation since the poor approximation $\hat{d}^{k*}_{\text{original}}$ in (3.18) becomes more accurate at lower SNRs and therefore the impact of stalling is limited.

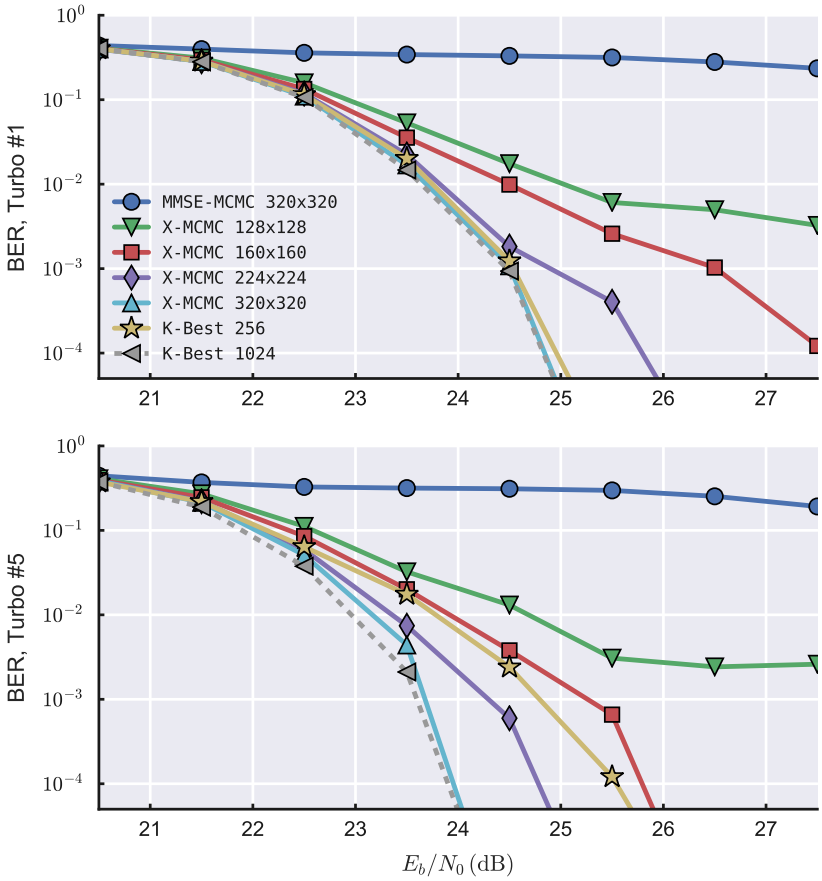


Figure 3.9. BER curves showing X-MCMC achieves near Max-MAP performance (approximated with large K-Best) even at the maximum 802.11ac MIMO and QAM sizes. Parameters: 8 antennas, 256 QAM, WiFi TGn Model-D channel, WiFi 3/4 LDPC encoding.

3.7 Summary

We presented a new derivation of the MCMC detector which solves the high SNR stalling problem without use of hybridization or heuristic temperature scaling terms. Output LLR quality was improved for poorly converged cases by conditioning output confidence on sample list statistics. Output LLR conditioning is shown to moderate soft-output overconfidence and allow a low complexity fixed length Gibbs sampler to be used in practice, eliminating error floors caused by rare slowly converging realizations. This conditioning may have application to other list based detectors such as list sphere-decoding and K-Best.

Additionally, we identified pseudo-convergence conditions which lowers the efficiency of the MCMC detector. A 1-bit randomization procedure was proposed as a low complexity

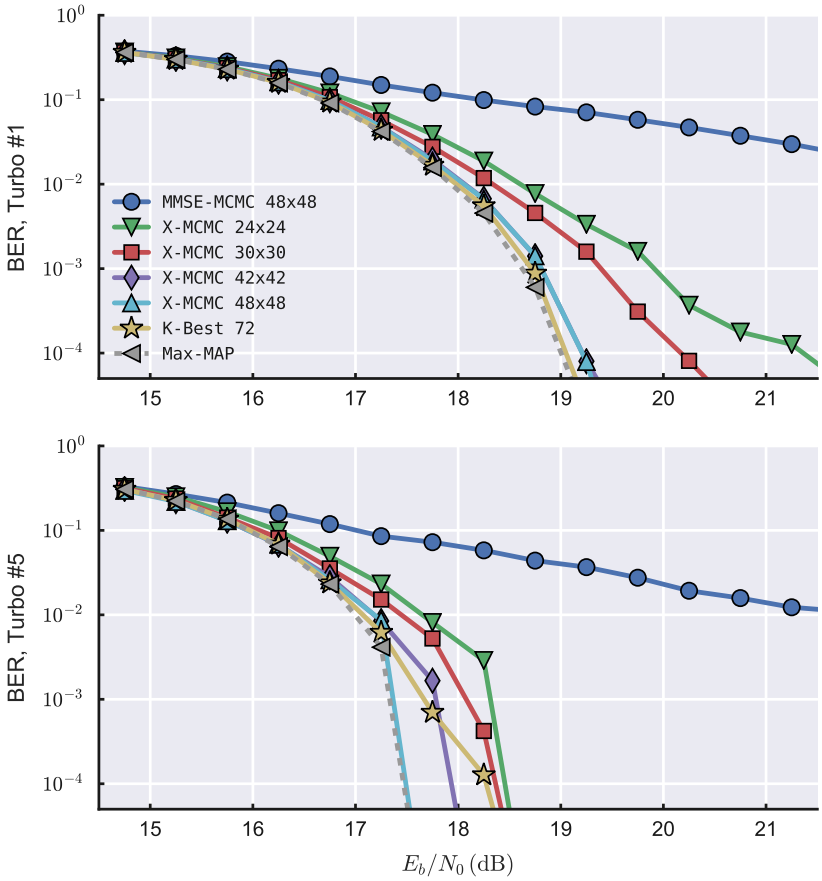


Figure 3.10. BER curves showing near Max-MAP performance for X-MCMC. Parameters: 4 antennas, 64 QAM, WiFi TGn Model-D channel, WiFi 3/4 LDPC encoding.

alternative way to leave pseudo-convergence compared to using a full random-walk restart.

Results showed that the combined improvements allow near Max-MAP performance at all SNR regimes with large numbers of antennas and high-order modulation. This is true even with highly correlated, WiFi TGn Model-D channels which are significantly more challenging than the Gaussian i.i.d. channels commonly used in the literature. No heuristic optimizations are needed, making the proposed method straightforward to effectively implement in practice.

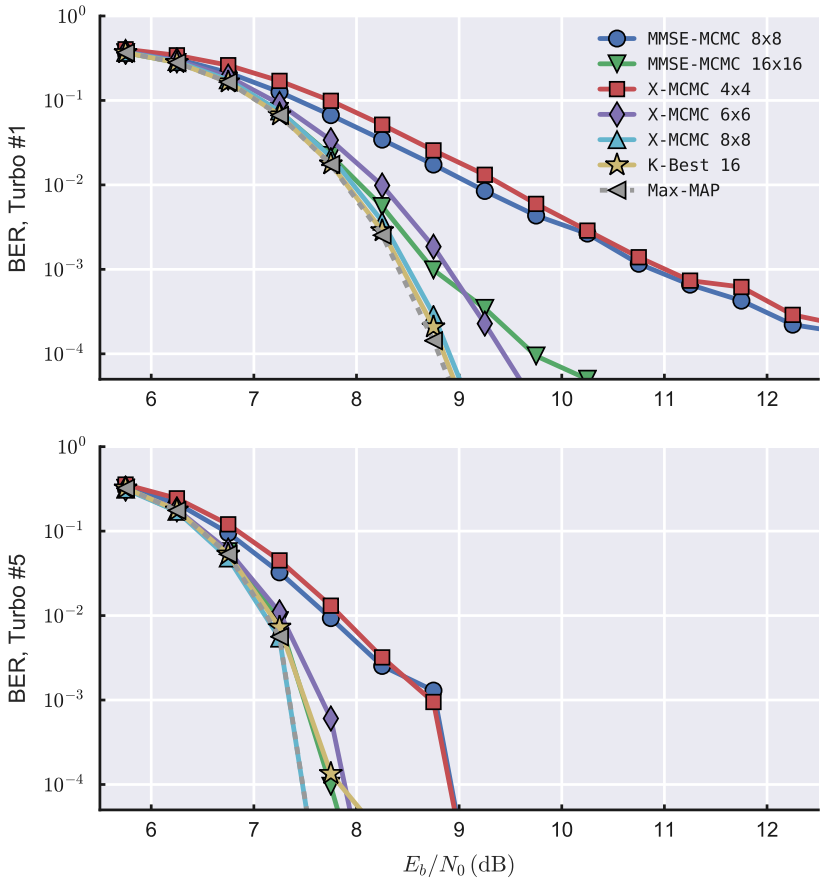


Figure 3.11. BER curves showing that all methods at low SNR though X-MCMC is more efficient than previous MCMC methods. Parameters: 4 antennas, 4 QAM, WiFi TGn Model-D channel, WiFi 3/4 LDPC encoding.

CHAPTER 4

TESTBED DEMONSTRATION: MIMO CHANNELS, INTERFERENCE METRICS, AND 802.11AC MEASUREMENTS

The purpose of this chapter is to verify that the simulation methods used in other chapters accurately represent real-world conditions and to demonstrate the high performance of the X-MCMC detector with the 802.11ac WiFi protocol. This is a useful exercise as we will show that the channel models often used in the MCMC detector literature are insufficient to represent indoor environments and thus can result in misleading analysis conclusions.

We will review several potential MIMO channel models including Gaussian i.i.d., Rayleigh with and without correlation between the antennas, and the official WiFi TGn model based on clusters of scatterers. To help compare the resulting channels and identify which represent our real-world measurements most closely, we introduce the condition-number (CN) of a channel matrix. Simulations are used to compare the relative performance of several different MIMO detectors on these different channels, with the varied results demonstrating the importance of using a representative model.

Interference metrics are briefly reviewed including the signal-to-noise-ratio (SNR) and the error-vector-magnitude (EVM). We then extend EVM to MIMO and make improvements to overcome several limitations. We refer to the resulting new metric the harmonic-mean-signal to arithmetic-mean-distortion ratio (HSADR) which we use to compare the 8x8 Ettus B210 measurement results to simulation, where 8x8 is shorthand for 8-antenna MIMO with 8 spatial streams.

The 4x4 National Instruments MIMO testbed was used before the X-MCMC detector was developed, therefore these results use MMSE initialized MCMC detector from Chapter 2. The detailed analysis on this testbed is still interesting because it presents the 3-dimensional relationship between BER, SNR, and channel CN. This concept is used to create two-

dimensional slices through the data that allow almost perfect matches between simulation and measurement BER curves.

The 8x8 Ettus based MIMO testbed results demonstrate our new X-MCMC detector with near Max-MAP performance verified using the 802.11ac WiFi protocol. Additional information is provided on the methods and techniques to match simulation to measurement and to construct a low cost and effective 8-antenna MIMO testbed.

This chapter is structured as follows. First, the methods needed to produce simulated channels similar to real-world measurements is covered in Section 4.1. This is followed by Section 4.2 with metrics needed to correctly identify interference and distortion in measurements for production of consistent analysis results. Next, the methods and results for 3 MIMO testbeds are presented in Sections 4.3, 4.4, and 4.5 for the 4x4 National Instruments, 8x8 Ettus USRP2, and 8x8 Ettus B210 testbeds, respectively. A summary is provided in Section 4.6.

4.1 Channel Models

Using the correct channel model in simulation has proven to be an important aspect of understanding the performance, complexity, and overall behavior of the MCMC and X-MCMC detectors. If the channel model used is less ill-conditioned than real-world channels, the MCMC detector converges easily with a small number of Gibbs samples. This can lead to an overoptimistic assessment of performance and complexity.

The channel can be represented in the frequency-domain by the complex matrix $\mathbf{H}$ where each element represent the path gain and delay between a pair of transmit and receive antennas. This allows the system to be modeled with

$$\mathbf{y} = \mathbf{H}\mathbf{s} + \mathbf{n} \tag{4.1}$$

where $\mathbf{y}$ is the received signal vector, $\mathbf{s}$ is the vector of transmitted QAM constellation symbols, and $\mathbf{n}$ is a noise vector. The noise elements are assumed to be independent and identically distributed (i.i.d.) complex Gaussian random variables with variance of σ_n^2 per each real and imaginary dimension. Assuming the transmit and receive side have the same number of antennas N, the dimensions of the vectors and matrices are $N \times 1$ and $N \times N$.

Note that here we will assume that the channel can be represented by a slow flat-fading model. This is appropriate for our 802.11ac WiFi simulations since it is orthogonal frequency

domain multiplexing (OFDM) multicarrier based, assumes low mobility indoor environments, and the packets are short compared to the long coherence time of the channel.

Here, we will briefly describe 3 pertinent channel models: Gaussian i.i.d. which is the most prevalent in the MCMC literature, Rayleigh with correlation which we used early in our research, and WiFi TGn which is currently our preferred method. Next, we present the condition number (CN) of a channel matrix so that the characteristics of different channel models may be easily quantified and compared against real measurements. Finally, the BER performance of MCMC and X-MCMC are shown on various channel models to understand how large of a difference the model makes. This will be an important foundation to understanding the results of the testbed demonstrations presented later in this chapter.

4.1.1 Gaussian i.i.d.

The Gaussian i.i.d channel is the simplest model and is generated with normally distributed i.i.d. complex Gaussian elements, $\mathcal{CN}(0,1)$. This is the method most commonly used in the MCMC literature [19, 23, 43]. Use of this model results in simulations which display overoptimistic performance leading to potentially wrong conclusions, as will be seen in Section 4.1.5.

There are two unrealistic features of the Gaussian i.i.d. model which makes it easier than typical real-world channels. First, there is no correlation introduced between elements, resulting in ill-conditioned channels rarely being produced. Second, each channel realization is produced independently in the frequency-domain, resulting in no correlation between the creation of the rare ill-conditioned channels.

4.1.2 Rayleigh With Correlation

After performing measurements on the 4-antenna MIMO testbed of Section 4.3, it was clear that the Gaussian i.i.d. model was inappropriate, therefore, we adopted a Rayleigh model with added correlation similar to [86]. Our variation does not contain Doppler spreading as here we are focused on indoor environments common to 802.11 WiFi.

Let N_t and N_r represent the number of transmit and receive antennas, N_{P} the number of time-domain samples in a packet, N_{taps} the number of taps in the time-domain channel, k and l the discrete time indices, and $\{\cdot\}'$ a time-domain version of a vector or matrix that also has a related frequency-domain representation. The time-domain model (4.2) takes

the generated signal $\mathbf{s}'$, applies a correlated Rayleigh channel, and adds the white Gaussian noise $\mathbf{n}'$ to produce the received signal $\mathbf{y}'$.

$$\mathbf{y}'[k] = \sum_{l=0}^{N_{\text{taps}}-1} \left(\mathbf{R}_r^{1/2} \, \mathbf{H}'[l] \, \left(\mathbf{R}_t^{1/2} \right)^T \, \mathbf{s}'[k-l] \right) + \mathbf{n}'[k] \tag{4.2}$$

An $N_t \times N_r$ Rayleigh distributed channel matrix $\mathbf{H}'[l]$ with elements $h'_{ij}[l]$ is generated for each packet realization. It is given an exponentially decaying power delay profile (PDP) as in equation (4.3). The time constant $\tau_s = 50$ ns is used as specified for a typical office environment in [87]. Each element uses a unique i.i.d. normally distributed complex random number with zero mean and unit variance $\psi_{ij}[l]$. The number of taps is limited to $5\tau_s$ where τ_0 is the sampling period; therefore the PDP tail is trimmed at power < -21.7dB which is negligible. The 802.11ac minimum OFDM guard interval of 400ns easily handles such a delay-spread.

$$h'_{ij}[l] = \begin{cases} \psi_{ij}[l] \sqrt{\exp(-l\tau_0/\tau_s)} & \text{for } 0 \le l \le \text{ceil}(5\tau_s/\tau_0) \\ 0 & \text{otherwise} \end{cases} \tag{4.3}$$

To introduce correlation among the elements of $\mathbf{H}'[l]$, we follow [88]. A complex Toeplitz matrix is used for the transmit and receive correlation covariance matrices $\mathbf{R}_t$ and $\mathbf{R}_r$, respectively. They are constructed of elements r_{ij} using (4.4) where $c_R = 0.25$ is a correlation scaling coefficient, ψ_{ij} is a standard normal random complex variable similar to that in (4.3) but without the need for temporal indexing, and * represents complex conjugation. These elements produce an exponentially decaying correlation away from the main diagonal as suggested by [89]. Using this matrix structure attempts to capture the spatial relationship in the linear antenna array used in the hardware implementation where more correlation is expected between neighboring elements.

$$r_{ij} = \begin{cases} 1, & \text{for } i = j \\ c_R^{|i-j|} \psi_{ij}, & \text{for } i < j \\ r_{ji}^*, & \text{for } i > j \end{cases} \tag{4.4}$$

Note that by increasing correlation coefficient c_R we can generate channels with larger condition number.

In order to use this time-domain model for OFDM simulations, we first generate a single time-domain (TD) channel realization for each packet. Thus we are implicitly assuming that the channel remains stationary over the duration of the packet. This TD channel is

converted to the frequency-domain (FD) where only the 52 active subcarriers defined in the 20 MHz WiFi specification are used. These are repeated as many times as needed to match the number of MIMO realizations in one codeword.

Notice that the correlated Rayleigh time-domain model solves the two primary undesirable characteristics of the Gaussian i.i.d frequency-domain model. First, correlation between elements is applied which will result in ill-conditioned channels being more common. Second, by generating a single TD realization per packet there is correlation in neighboring subcarriers, therefore there packets with concentrations of deep-fades will be more likely, thus the forward error correcting (FEC) code will more likely to fail.

We have found this model to work well in practice, as will be shown in Section 4.3. It is a potentially useful model as it is conceptually easy to understand and implement, but we do not recommend it to be generally adopted as there is no advantage other than simplicity over the more advanced WiFi TGn model presented in the next section.

4.1.3 WiFi TGn

The WiFi TGn channel model [61] was released in 2004 to assist in adding MIMO to the WiFi standard, resulting in MIMO being included in 802.11n-2009 [90]. It is a physical model based on clusters of scatters first introduced by Saleh and Valenzuela [91] and expanded on by many others [92–97]. There are multiple models, A-F, with specific combinations of power profiles, delay profiles, angles of arrival, and correlation. The intention is that each model designated by a different letter captures the important aspects of a different type of indoor environment. An example of the kinds of differences would be the long delay profile with strong LOS typical to a warehouse compared to the short delay with no LOS in a small multi-room home.

The general idea is that the channel can be represented by the combination of a line-of-sight (LOS) and a non-line-of-sight (NLOS) component as in

$$\mathbf{H}' = \sqrt{P}\left(\sqrt{\frac{K}{K+1}}\mathbf{H}'_{\text{LOS}} + \sqrt{\frac{1}{K+1}}\mathbf{H}'_{\text{NLOS}}\right) \tag{4.5}$$

where $\{\cdot\}'$ designates a time-domain representation, P is a power scaling factor, and K is the Ricean K-factor [98, 99].

The elements of $\mathbf{H}'_{\text{LOS}}$ are generated from a defined set of scatterers with specific incoming angles occurring at specific delays impinging a uniform linear array of antennas. The power

from the scatterers are based on a double exponentially decaying power-delay-profile (PDP) as in

$$\beta_{kl}^2 = \beta_{0,0}^2 e^{-T_l/\Gamma} e^{-\tau_{kl}/\gamma} \tag{4.6}$$

where β_{kl}^2 specifies the average power of the k^{th} arrival of the l^{th} cluster, T_l and τ_{kl} are arrival times, and Γ and γ determine the rate of decay. This is similar to the specification of the single PDP used in the Rayleigh model of Section 4.1.2.

The phase of $\mathbf{H}'_{\text{LOS}}$ is based on the specified angle-of-arrival, thus creating correlation between the elements.

The $\mathbf{H}'_{\text{NLOS}}$ matrix [100] is represented with a Rayleigh model where correlation is applied with $\mathbf{R}$ as in

$$\mathbf{H}'_{\text{NLOS}} = \mathbf{R}_r^{1/2} \mathbf{H}'_{\text{iid}} \left(\mathbf{R}_t^{1/2}\right)^T \quad . \tag{4.7}$$

This is similar to the method of adding correlation in Section 4.1.2.

Similar to the previous section, this is a time-domain model which can be converted to the frequency-domain for use in simulations. By generating a single time-domain realization per packet, the chance of having multiple ill-conditioned realizations over a packet is increased.

Since there are multiple models specified by TGn, a method is needed to select a good match to a specific situation. The most accurate method is to compare the delay and power profiles of measured channels and compare it with the models. This is a difficult and time consuming approach, therefore, in the next section, we present a simple alternative method to assess channel quality which uses the condition number of the frequency-domain channel matrix.

4.1.4 Condition Number of Channel Matrix

For MIMO systems, the detection performance depends on not only the signal-to-noise (SNR) ratio, which is related to the channel additive-noise vector $\mathbf{n}$, but also the condition number of the channel matrix $\mathbf{H}$ [101]. In the next section, we will investigate the effect of the condition number on the detection of the received signal in a MIMO system.

For signal transmission over a MIMO channel, receivers typically perform detection of the transmitted symbols using the estimated channel matrix. The performance of undoing the channel matrix using this method is dependent on the condition of the channel matrix. When the matrix is ill-conditioned, having columns that are correlated to each other, a small

disruption of the transmitted signal by additive noise will result in a larger change in the received signal. This will prevent the reliable separation of the spatial streams due to poor spatial multiplexing. This means that in addition to the signal-to-noise ratio (SNR) of the channel, the condition number is also an important parameter affecting the capacity of a MIMO channel [101].

For the channel matrix $\mathbf{H}$, the condition number CN is defined as:

$$\mathrm{CN}(\mathbf{H}) = \frac{\sigma_{\max}}{\sigma_{\min}} = \frac{\sqrt{\lambda_{\max}}}{\sqrt{\lambda_{\min}}} \tag{4.8}$$

where $\sigma_{\max}$ and $\sigma_{\min}$ are the maximum and minimum singular values of $\mathbf{H}$ [101], and $\lambda_{\max}$ and $\lambda_{\min}$ are the maximum and minimum eigenvalues of the associated correlation matrix $\mathbf{H}^{\mathrm{H}}\mathbf{H}$. To express the CN in dB, we use the following equation [102, 103]:

$$\mathrm{CN}(\mathbf{H}) \text{ in dB} = 20\log_{10}\left(\frac{\sigma_{\max}}{\sigma_{\min}}\right) = 10\log_{10}\left(\frac{\lambda_{\max}}{\lambda_{\min}}\right) \quad . \tag{4.9}$$

For an ill-conditioned channel matrix, the spread of the singular values will be large causing the magnitude of CN to also be large. This relates to correlated crosstalk between paths reducing the spatial multiplexing capacity in the MIMO channel. If we use linear equalization based detection methods, then in environments with larger CN the receiver will be sensitive to perturbations caused by the channel's additive noise. An example of a linear equalization based method is the MMSE detector

$$\begin{aligned}\hat{\mathbf{s}} = &\left(\mathbf{H}^{\mathrm{H}}\mathbf{H} + \frac{\sigma_n^2}{\sigma_s^2}\mathbf{I}\right)^{-1}\mathbf{H}^{\mathrm{H}}\mathbf{H}\mathbf{s} \\ &+ \left(\mathbf{H}^{\mathrm{H}}\mathbf{H} + \frac{\sigma_n^2}{\sigma_s^2}\mathbf{I}\right)^{-1}\mathbf{H}^{\mathrm{H}}\mathbf{n}\end{aligned} \tag{4.10}$$

where $\hat{\mathbf{s}}$ is the estimate of $\mathbf{s}$. The first term on the right-hand side of (4.10) provides an estimate of $\mathbf{s}$. It will be very close to the true value if the SNR is high.

The second term represents corruption arising from equalization applied to the noise vector $\mathbf{n}$. Its negative effect will be enhanced by the equalization under certain bad channel matrices $\mathbf{H}$ indicated by large CN. In this situation the components of the noise vector at the output of the MMSE detector will be correlated and have a larger variance. As a result, the detector performs poorly and the estimate $\hat{\mathbf{s}}$ may no longer be in the neighborhood of the true transmitted symbols $\mathbf{s}$.

The condition number of the channel matrix $\mathbf{H}$ can be estimated efficiently using the singular value decomposition (SVD), QR factorization, or even a simple L1-norm calculation [104]. The SVD of the matrix $\mathbf{H}$ is defined as

$$\mathbf{H} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^{\mathrm{H}} \tag{4.11}$$

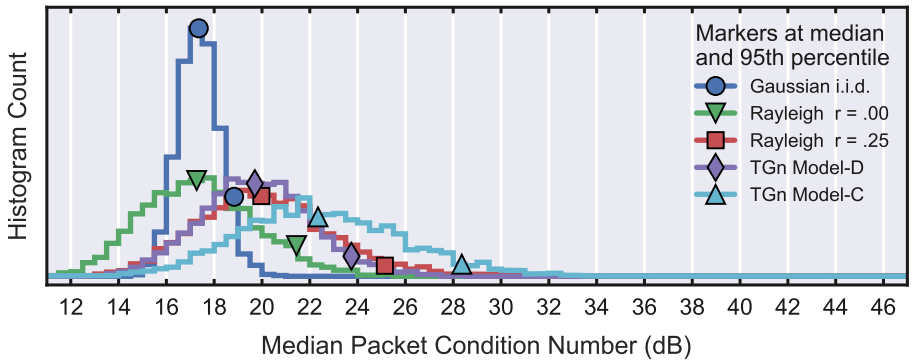
where $\mathbf{\Sigma}$ is a diagonal matrix containing the singular values of $\mathbf{H}$ and the columns of $\mathbf{U}$, $\mathbf{V}$ are the associated singular vectors.

In [101], it is suggested that linear equalization based detection will be affected significantly for CN($\mathbf{H}$) above 12 dB. For maximum likelihood (ML) based detection methods, the tolerance level is higher with a threshold at 28 dB. The condition numbers in dB have an approximately linear relationship with the increase in size of the channel matrix. In other words, when we have more transmit or receive antennas in the system, the condition number is on average larger. This is important as IEEE 802.11ac features up to 8 spatial streams and linear equalization based detection methods will suffer from performance degradation due to the increase in condition number.

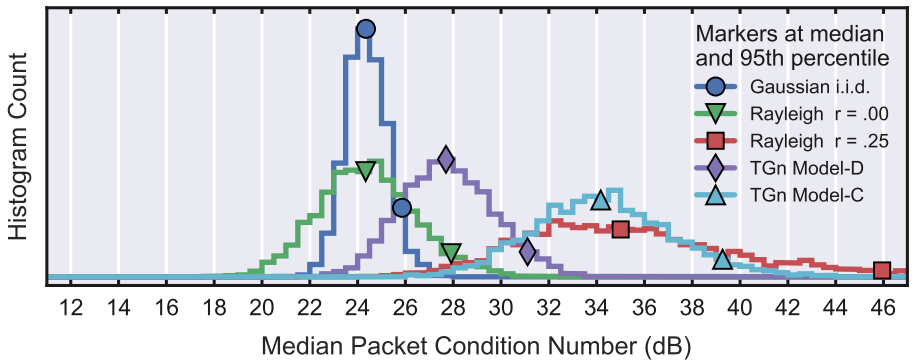
In [105], the distribution and impact of condition number from Rayleigh multipath channels are investigated. The condition number has a probability density function that is right-skewed and decays polynomially. Thus ill-conditioned channels with large condition number values are likely to occur.

To compare the orthogonality and therefore difficulty of several common channel models, see the CN histograms in Fig. 4.1. Each data point is the median CN of 52 frequency-domain channels corresponding to the active OFDM subcarriers in 20 MHz bandwidth 802.11ac. For the Gaussian i.i.d. model, all channels were independently generated from i.i.d. complex Gaussian variables of zero mean and 0.5 variance per dimension. In the Rayleigh and TGn models, a single time-domain realization is used to produce the 52 frequency-domain channels per data point. The Rayleigh model with nonzero r is generated as in [54] where r adjusts the strength of the added correlation. The TGn models are from the 802.11n WiFi specification in [61] and is based on defined clusters of scatterers. The TGn model letter designates differing clusters and delay profiles which affects the degree of correlation and severity of fading.

The Rayleigh $r = 0.25$ model was confirmed as a valid model for 4-antenna indoor



(a) 4-antenna channels generated.



(b) 8-antenna channels generated.

Figure 4.1. Comparison of channel distributions using various models. Each data point is the median of a set of 52 frequency-domain channels corresponding to the active OFDM subcarriers in 802.11ac.

environments with measurements in our prior work [54], but to allow better comparisons with other papers in the literature, we now use the standard 802.11n WiFi TGn specified channel models. As can be seen in Fig. 4.1, the previous model matches well with the statistics of the TGn Model-D for 4-antenna situations.

The most important observation to note in Fig. 4.1 is that the Gaussian i.i.d. channels, the most commonly used model in the MCMC literature [19, 23, 43], have a much lower condition number and therefore is much easier than the other models. This means that any performance and complexity analysis performed using an i.i.d. Gaussian channel model should be considered overoptimistic compared to real-world indoor environments. This will be revisited with the results in Section 4.5.3.

4.1.5 Impact of Channel Model Selection

In the preceding sections, several channel models have been described and a simple method to compare them using condition number. Here, we will briefly demonstrate the performance difference created by the use of those channel models. This is important as the choice of model can have a significant impact on conclusions made during analysis and comparison of different MIMO detector algorithms.

Here we only show the difference between the MMSE initialized MCMC detector, as described in Chapter 2, and the X-MCMC detector of Chapter 3 with Max-MAP shown as a performance bound. In Fig. 4.2, BER curves are presented for Gaussian, Rayleigh, and WiFi TGn channel models. Compared to the others, the Gaussian i.i.d. model is dramatically easier with even MMSE-MCMC, which suffers from stalling problems, converging to within 1.5 dB of Max-MAP performance with a 24x24 Gibbs sampler. This is in stark contrast to the more realistic models which show that MMSE-MCMC is strongly stalled.

Next, notice that the Rayleigh model without correlation ($r = 0$) is significantly easier with the old MMSE-MCMC algorithm struggling, though still converging with a large 72x72 Gibbs sampler. With the high correlation of Rayleigh with $r = 0.25$, TGn Model-D, and Model-C the MMSE-MCMC detector is strongly stalled and does not work.

Most of the MCMC literature uses an i.i.d. complex Gaussian channel model [19, 23, 43]. Compared to the other models of Fig. 4.2 which more accurately represent real-world indoor-channels, a much smaller MCMC Gibbs sampler is needed for the BER to converge to Max-MAP performance. Also, there is a large disagreement over the effectiveness of the MMSE initialized MCMC detector. This is because the Gaussian model has fewer and less severe deep fades and the channels are generally better conditioned than real channels which more often suffer from strong correlation.

4.2 Noise and Interference Metrics

One of the challenging aspects of comparing communications system simulations to testbed measurements is quantifying interference. During simulations, often the only form of external interference is additive white Gaussian noise (AWGN) with constant variance over a test. This type of interference is quantified well with a typical signal-to-noise-ratio (SNR) metric comprising the average signal energy over the average noise energy.

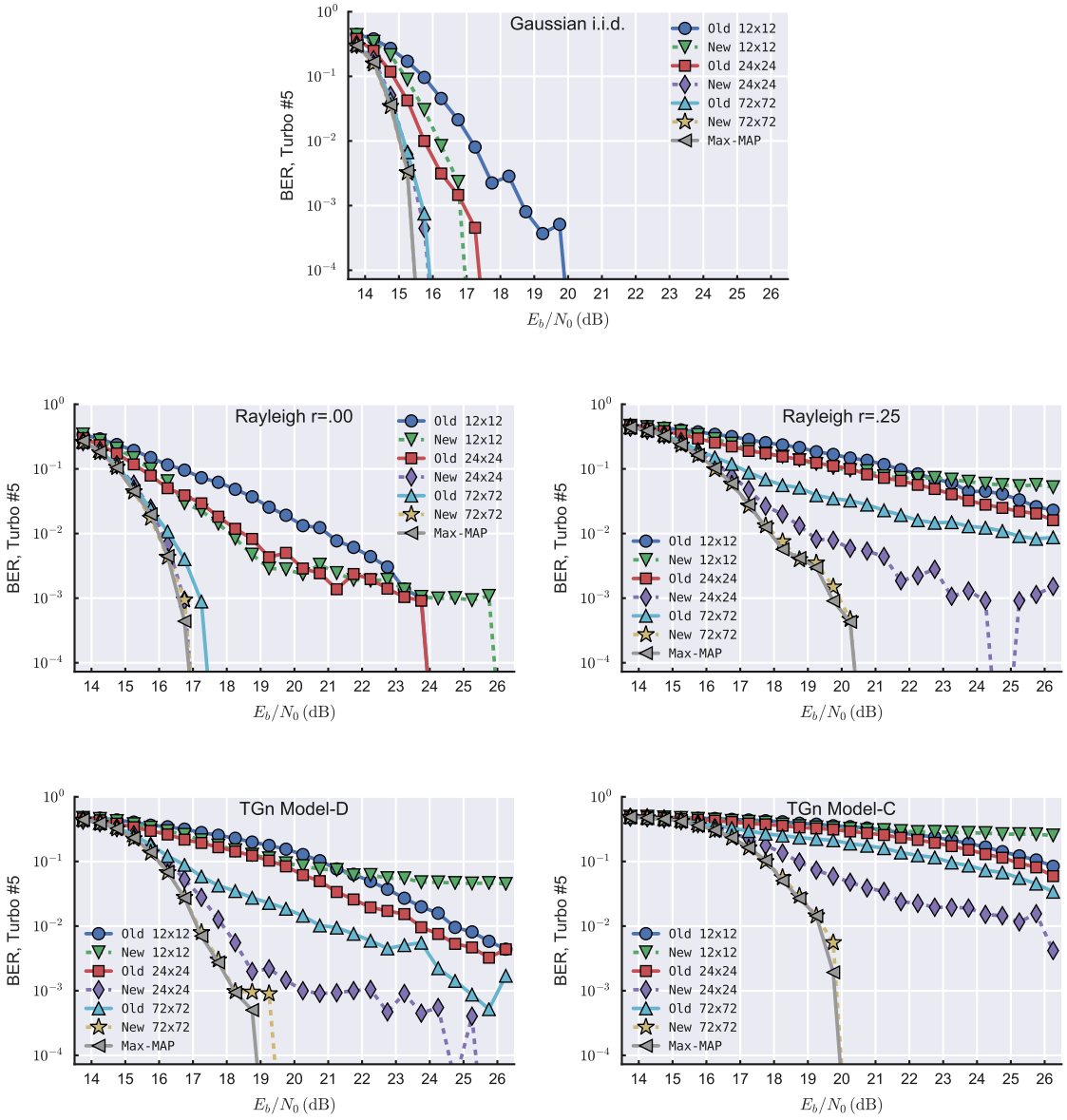


Figure 4.2. Impact of channel model choice on BER curve results. "Old" designates the MMSE initialized MCMC detector of Chapter 2 whereas "New" designates X-MCMC of Chapter 3 with Gibbs excitation, 1-bit forced flip to escape convergence, and output overconfidence conditioning.

$$\mathrm{SNR} = \frac{E_{\mathrm{signal}}}{\hat{E}_{\mathrm{noise}}} = \frac{\sum_{i=1}^{N_{\mathrm{subc}}} \|\mathbf{y}_i\|^2}{\sum_{i=1}^{N_{\mathrm{subc}}} \|\hat{\mathbf{n}}_i\|^2} \tag{4.12}$$

Notice that the ratio is of the arithmetic mean over N_{subc} subcarrier realizations of the signal and noise separately. Doing the mean separately improves stability and moderates outliers.

In real-world systems, there are many types of interference including noise, distortion, channel estimation error, and other active transmissions which all vary over both time and frequency. This variability is not effectively represented by an SNR measurement made only with training fields or pilot symbols. Physical methods to control a test such as cabling the receiver to transmitter or using an anechoic chamber are inappropriate for spatial-multiplexing MIMO because it removes the multipath environment needed for spatial reuse.

Physically, there are several possible solutions to control external interference, but all are inappropriate in this situation. Large data-sets are insufficient as active interference can change dramatically between test locations and collection times, making experiments unreproducible. Anechoic chambers cannot be used because they lack the rich multipath environment needed for spatial-multiplexing. Using an isolated test range, far away from other electronic emissions, is generally not available to most researchers. Manually inspecting measured data and attempting to remove unusual data points is undesirable as it is time consuming and unreproducible. Therefore, we will examine a modified noise metric useful in quantifying interference in real-world MIMO measurements.

The standard signal-to-noise-ratio is a reasonable, consistent, and stable measure when interference is primarily additive white Gaussian noise (AWGN) with constant variance. In testbed measurements the amount of noise is only an estimate, often based on the known training-field in the header of a packet. As a result, this metric cannot reliably estimate irregular interference that may affect packet header and payload differently.

A metric often used in lab testing is the error-vector-magnitude (EVM) [106, 107].

$$\mathrm{EVM}^2_{\mathrm{SISO}} = \frac{\sum_{i=1}^{N_{\mathrm{subc}}} \|\hat{s}_i - s_i\|^2}{\sum_{i=1}^{N_{\mathrm{subc}}} \|s_i\|^2} = \frac{\sum_{i=1}^{N_{\mathrm{subc}}} \left\|\frac{n_i}{\hat{h}_i}\right\|^2}{\sum_{i=1}^{N_{\mathrm{subc}}} \|s_i\|^2} \tag{4.13}$$

It uses knowledge of the transmitted data to calculate the ratio of interference to signal energy directly using the payload. This makes it useful in capturing time varying conditions. The inverse-EVM (IEVM) can be extended to MIMO with

$$\mathrm{IEVM}^2_{\mathrm{MIMO}} = \frac{\sum_{i=1}^{N_{\mathrm{subc}}} \|\mathbf{s}_i\|^2}{\sum_{i=1}^{N_{\mathrm{subc}}} \|\hat{\mathbf{s}}_i - \mathbf{s}_i\|^2} = \frac{\sum_{i=1}^{N_{\mathrm{subc}}} \|\mathbf{s}_i\|^2}{\sum_{i=1}^{N_{\mathrm{subc}}} \left\|\hat{\mathbf{H}}_i^{-1}\mathbf{y}_i - \mathbf{s}_i\right\|^2} \tag{4.14}$$

where N_{subc} is the number of active OFDM subcarriers within a frame and $\hat{\{\cdot\}}$ designates an estimate. The problem with this method is that when the channel is ill-conditioned, the matrix inverse $\mathbf{H}_i^{-1}$ can create a large amount of noise enhancement. This results in an overestimate of the interference which near maximum-likelihood detectors such as MAP, MCMC, X-MCMC, and K-Best do not experience. During our testing, this overestimate was commonly in the order of 10-20 dB which made it misleading and unreliable. Improvements could be made by replacing the channel inverse based estimate of $\hat{\mathbf{s}}$ with an alternative, such as MMSE, which creates less noise enhancement, but instead we propose removing the need for the estimate completely.

To remove the noise enhancement caused by the channel inverse, first the channel effects must be moved to the signal portion of the metric. We refer to this as the arithmetic-mean-signal to arithmetic-mean-distortion ratio and is defined as

$$\mathrm{ASADR} = \frac{\frac{1}{N_{\mathrm{subc}}} \sum_{i=1}^{N_{\mathrm{subc}}} \left\|\hat{\mathbf{H}}_i \mathbf{s}_i\right\|^2}{\frac{1}{N_{\mathrm{subc}}} \sum_{i=1}^{N_{\mathrm{subc}}} \left\|\mathbf{y}_i - \hat{\mathbf{H}}_i \mathbf{s}_i\right\|^2} \quad . \tag{4.15}$$

This metric is able to detect and represent many forms of time varying distortion that may be present in the payload. Unfortunately, using the arithmetic-mean of $\mathbf{Hs}$ deemphasizes the effects of weak channels, unlike $\mathrm{EVM}_{\mathrm{SISO}}$ in (4.13). This is because the arithmetic-mean emphasizes large values.

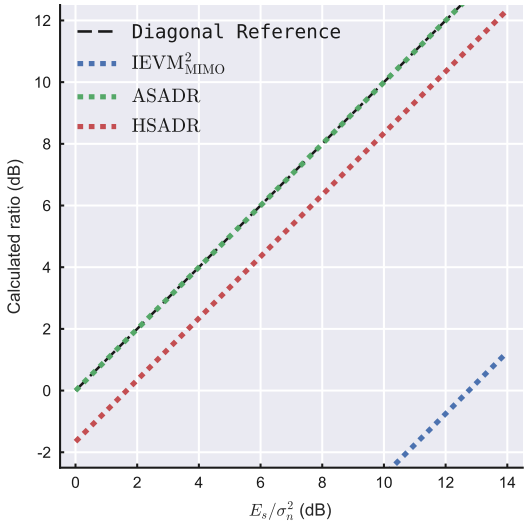
To improve the ASADR metric by emphasizing weak channels, we have developed the harmonic-mean-signal to arithmetic-mean-distortion ratio (HSADR) metric as an SNR measure that uses the payload and avoids the noise enhancement of IEVM. It is defined as

$$\text{HSADR} = \frac{\left(\frac{1}{N_{\text{subc}}} \sum_{i=1}^{N_{\text{subc}}} \left(\left\| \hat{\mathbf{H}}_i \mathbf{s}_i \right\|^2 \right)^{-1} \right)^{-1}}{\frac{1}{N_{\text{subc}}} \sum_{i=1}^{N_{\text{subc}}} \left\| \mathbf{y}_i - \hat{\mathbf{H}}_i \mathbf{s}_i \right\|^2} \quad . \tag{4.16}$$

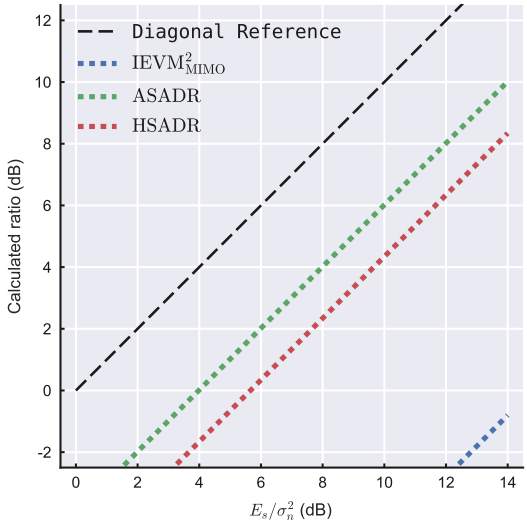
There are two important features of this definition. First, the channel in IEVM is moved from the bottom of the ratio to the top, thus removing the problematic noise enhancement. Next, the arithmetic-mean-signal, which emphasizes large values, is replaced with the harmonic-mean, which emphasizes small values [108]. In a single-input single-output (SISO) EVM calculation, taking the arithmetic-mean of the noise over channel gain (n/h) results in emphasizing small, weak channel conditions, that is, deep fades. When the channel is moved to the top of the ratio and no longer inverted, poor channels have little effect on the metric. By taking the harmonic-mean of $\hat{\mathbf{H}}\mathbf{s}$, weak channels have a larger impact on the mean which is desirable.

Using the HSADR metric of (4.16) on testbed measurements with irregularly interfered and distorted packet payloads results in a more consistent analysis with smooth BER curves. Note that the HSADR metric requires prior knowledge of the transmitted payload and so is only used for analysis and plotting purposes and not by the signal processing blocks.

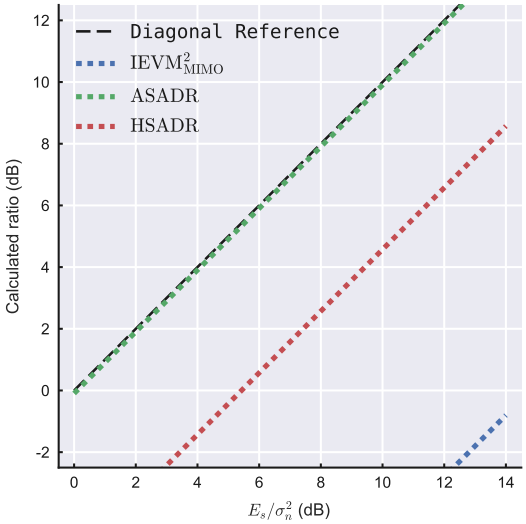
To make the differences between these various definitions of SNR more clear we present IEVM, ASADR, and HSADR in Fig. 4.3. First, note that even for these Gaussian i.i.d. channels which tend to be much better conditioned than real-world channels, the noise enhancement is so large for IEVM that it is on average 14 dB low. This clearly makes it an unusable measure of SNR. Next, ASADR is a good measure for Gaussian affects and unusual noise but fails to detect channel irregularities. Finally, notice that HSADR can detect both unusual noise and channel irregularities. The only drawback of HSADR is that it is consistently low. This is because the harmonic-mean of a quadrature-amplitude-modulated (QAM) signal is smaller than the arithmetic-mean. If it is desirable to match HSADR with other results, it may be useful to correct for this known offset. Note that even after correcting for the harmonic-mean QAM offset that HSADR will still be low depending on



(a) Gaussian noise and Gaussian i.i.d. channel.



(b) Gaussian noise and Gaussian i.i.d. channel. Augmented with 2% of noise realizations increased by 20 dB to simulate active interferer or distortion.



(c) Gaussian noise and Gaussian i.i.d. channel. Augmented with 2% of channel realizations decreased 20 dB by to simulate deep fades or distortion.

Figure 4.3. Comparison of interference metrics with 4-antenna 64 QAM data. E_s/σ_n^2 is the SNR using the initial Gaussian simulation parameters which is then augmented by the specified method in the subfigures.

channel realizations. This is desirable as it emphasizes poor quality channel outliers which are likely to dominate performance characteristics.

4.3 4-Antenna 'National Instruments FlexRIO' MIMO Testbed

The 4x4 National Instruments testbed was our first attempt to demonstrate the MCMC MIMO detector over real transmissions with the 802.11ac protocol [54]. The version of MCMC used was the MMSE initialized variety described in Chapter 2 which does not include any of our new MCMC techniques. At the time, we did not fully understand the high SNR stalling problem or have a resolution to it, therefore, only low SNR 4-antenna 4-QAM results were shown because higher QAM sizes needing higher SNR did not perform well. This work was important because it helped us develop the use of condition number to quickly identify the quality of a channel and helped us discover that the Gaussian i.i.d. model is insufficient for use in representative simulations.

In this section we briefly outline the hardware and algorithms used in this series of tests. Then we present a method to get accurate matches between testbed measurement and simulation BER curves by using condition number slicing. Finally, results are presented verifying the effectiveness of the methods and the potential improvement of MCMC over a soft-MMSE solution alone.

4.3.1 Hardware

The development system shown in Fig. 4.4 is a state of the art platform based on the National Instruments FlexRIO with NI5792/5793 RF transceiver cards and Xilinx Kintex-7 FPGAs. It has the capability for 4x4 MIMO with 0.2-4.4 GHz frequency range, 200 MHz instantaneous bandwidth, and 20 dBm transmit power.

Above the National Instruments hardware, a custom antenna stand and plate is used constructed of RF transparent HDPE plastic and PVC. Using a variable template of holes, a square or linear arrangement of antennas may be placed with separations of 1/4, 1/3, /1/2, 2/3, 3/4, or 1 wavelength at 2.45 GHz. A linear 2/3 wavelength spacing was used with dual band 2.4/5 GHz monopole antennas for the measurements that follow.

To be precise, 802.11ac is limited to the 5 GHz frequency band. This development system is used to transmit at a lower frequency due to hardware limitations, but this should not

Figure 4.4. Development system based on National Instruments FlexRIO platform.

effect BER performance and the conclusions drawn from our study.

For more information on the 4x4 National Instruments testbed, see Appendix A.

4.3.2 Algorithm

The MCMC detector is only a single component in a digital communication system. In order to evaluate its real-world performance it must be implemented within a relevant and compatible framework. The IEEE 802.11 WiFi standard is an ideal choice for this purpose. This standard has very high relevance with the multi-billion dollar WiFi industry based upon it. The newest specification revision released in 2013, 802.11ac, defines MIMO systems of up to 8x8 in size [7], but interestingly no system of such a large size is commercially available. One of the reasons that systems larger than 3x3 are rare is that a high-performance, low-complexity detector is still needed.

In summary, the 802.11ac standard is an OFDM based wireless communication system used for computer networking. Depending on channel conditions the protocol dynamically changes between many possible modes to maximize reliable data-rate. Transmissions can be sent using 20, 40, 80, or 160 MHz bandwidth over 1 to 8 antennas. Modulation types include BPSK, 4-QAM, 16-QAM, 64-QAM, and 256-QAM. Interleaved low-density parity-check

coding (LDPC) at 1/2, 2/3, 3/4, and 5/6 rates are used. OFDM symbol guard intervals of 400 and 800 nanoseconds are possible. When multiple antennas are present they can be used in beamforming, spatial-multiplexing, or multiuser modes. Maximum transmitted power varies according to local regulations, but 20 dBm is a common baseline in retail products for use in most countries.

Our implementation allows testing of most possible 802.11ac configurations by adjusting number of antennas, bandwidth, packet size, QAM size, and coding rate. To keep the analysis here concise, only results for 4x4 antennas, 20MHz bandwidth, 400ns guard interval, 604 byte packet size, 4 QAM, and 5/6 LDPC coding rate are presented.

Although technical details on the detectors used in commercial MIMO WiFi systems are not available, it is assumed that most use a form of soft-MMSE feeding into the LDPC decoder, where *soft* means that no hard decisions have been made on symbol content. Under limited conditions, an approximate ML method is likely used. The limitations being that it will only be used in situations with lower numbers of antennas, smaller QAM sizes (meaning low data rates), and at high condition number. These restrictions keep complexity and therefore chip cost, processing lag, and power requirements down. For comparison to the MCMC method, soft-MMSE has been chosen [34]. This solution also acts as an initializer to MCMC's search.

The 802.11ac protocol uses a packetized structure with 52 active OFDM subcarriers. Each subcarrier has its own channel characterized by a different condition number. Because of the LDPC coding which correlates the BER across all active subcarriers in the packet it is necessary to condense the 52 condition numbers to a single value defined for a packet. This is done by performing an average in dB scale.

Our implementation of MCMC in this section matches that described in Chapter 2 and [43] with the MMSE initialization suggested by [20]. The MCMC detector can be adjusted to have a varying number of iterations, number of Gibbs samplers, and number of Gibbs sampler iterations. For this implementation 5 iterations with 12 Gibbs samplers each running 3 times was chosen as a balance between performance, processing lag, and complexity. Complexity increases with the number of Gibbs samplers and this should be scaled in proportion to the number of antennas and constellation size, that is, the number of bits per channel use.

4.3.3 Results

The overall relationship between BER/SNR/CN is best viewed in 3-dimensions as in Fig. 4.5, though in print this is difficult to use for understanding relationship detail; therefore the following results are presented with 2D slices through this space.

The motivation for using results in a 3-dimensional space can be seen in Fig. 4.6b. Despite collecting over 100,000 packets over 7-days with different locations and antenna orientations, the CN distribution is still irregular and biased. In the University of Utah Merrill Engineering Building, recording real-world data in a single location tends to create a data spike with only a 5 dB range in CN; therefore measurement results are difficult to reproduce and compare against simulation. Real-world measurements displaying BER results must include CN as a third dimension to take into account the biasing from specific, unreproducible channel conditions.

If considering only a simulated channel such as in Fig. 4.6a, the condition numbers display a fairly even, reproducible shape with a slight tail to the right. As shown in [105], with high numbers of antennas this becomes a Gamma-like distribution with a polynomial decay for higher condition numbers. With large numbers of simulated packets, the BER will consistently settle to a result similar to a slice at the condition number average, just to the right of the peak. Therefore it is acceptable to exclude condition number when comparing simulated results between themselves, but not when real-world channels are included.

There is one additional difficulty in comparing the results between different methods using measured data. A typical approach would be to take a BER average within SNR bins to create a trend line and then compare these performance lines against each other. This method works well in simulation because the SNR and CN can be controlled, allowing collection of many packets for a specific SNR and CN point. This also can work well for real measurements with SNR and CN ranges until the region of the BER waterfall. With the current hardware test setup, using multithreaded and optimized C-code to process incoming packets, it would take several months to collect the number of points needed to create a smooth, consistent curve in the region of the waterfall for several CN slices.

Fortunately, at a fixed CN the BER vs SNR plot appears to display a linear relationship, see Fig. 4.7 where dots show single 802.11ac packet performance and solid lines show trends through the 2-dimensional subspace. By using linear regression to fit a line to the data,

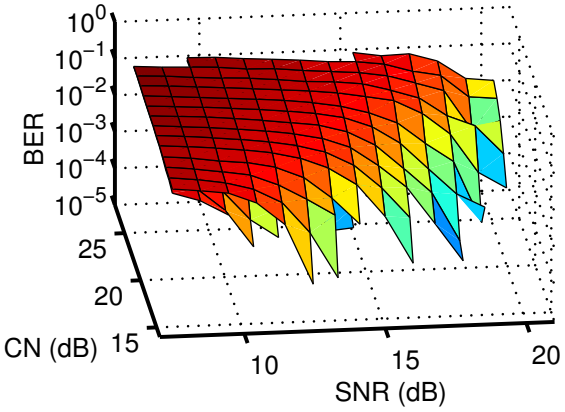
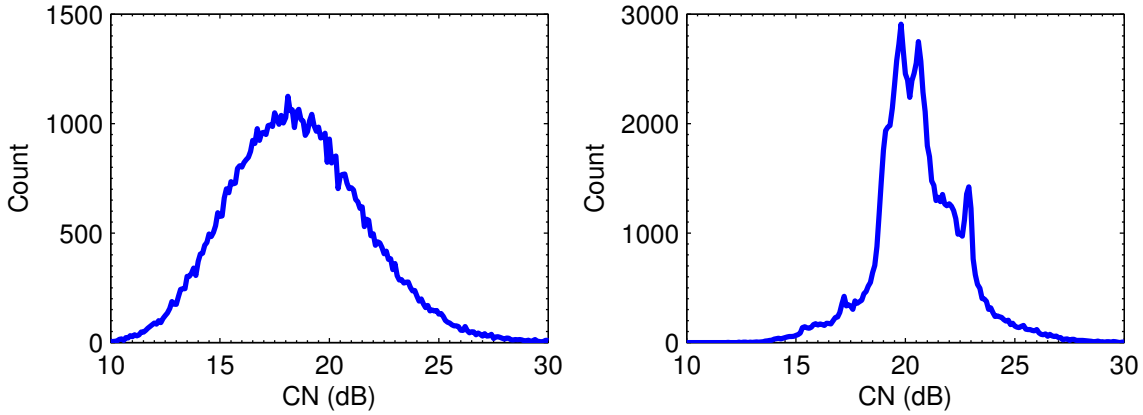


Figure 4.5. Measured BER surface.



(a) Simulated.

(b) Measured.

Figure 4.6. CN 0.1dB bin width histogram for 4x4 MIMO. Simulated results created with correlated Raleigh fading channel model. Measured results collected over 7-days in University of Utah Merrill Engineering Building in multiple locations and antenna orientations. Note that all results presented elsewhere first randomly decimate the data to a uniform CN distribution to prevent biasing.

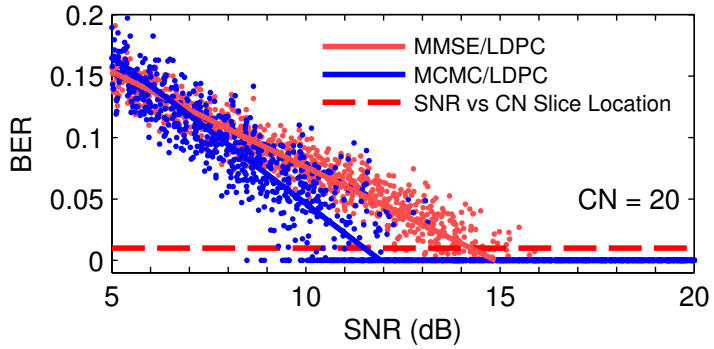


Figure 4.7. Simulated BER slices through CN $= 20 \pm 1$dB. Solid line is linear estimate of trend excluding BER $= 0$. Dashed line is location of horizontal slice showing SNR vs CN trend in Fig. 4.10a.

excluding zero BER which would bias the relationship, a consistent and reliable quantitative comparison between different methods' simulated and measured data can be made with a fraction of the data.

The data around the trend lines shown in Fig. 4.8 and 4.9 have SNR standard deviations of $0.6 - 1.3$dB for simulation and $0.9 - 1.1$dB for measurement, depending on CN slice. The results are presented in semi-log space for better visualization. Now that the concept of condition number has been included in displaying the BER performance and a trend line used, the results in Fig. 4.8 and 4.9 show good, consistent, and reliable agreement between simulation and measurement.

To view the effect of condition number more directly, see Fig. 4.10. This presents a slice through a specific BER to show the shift of the waterfalls toward higher SNR as condition number increases, meaning that more SNR is needed for packet transmission at higher values of CN. Notice that MCMC makes an increasing improvement over MMSE as condition number increases. This is expected since high channel matrix condition numbers result in increased noise enhancement which decreases performance in detectors such as ZF and MMSE. ML based methods such as MCMC are specifically designed to avoid this effect, giving near-optimal performance. An additional advantage of MCMC over other approximate ML methods such as sphere-decoding is that the method's complexity does not increase with increasing condition number [105].

In Table 4.1, the trend lines from Fig. 4.8 and 4.9 have been extended to BER $= 10^{-5}$ to facilitate comparisons to a larger, more common, default computer networking packet

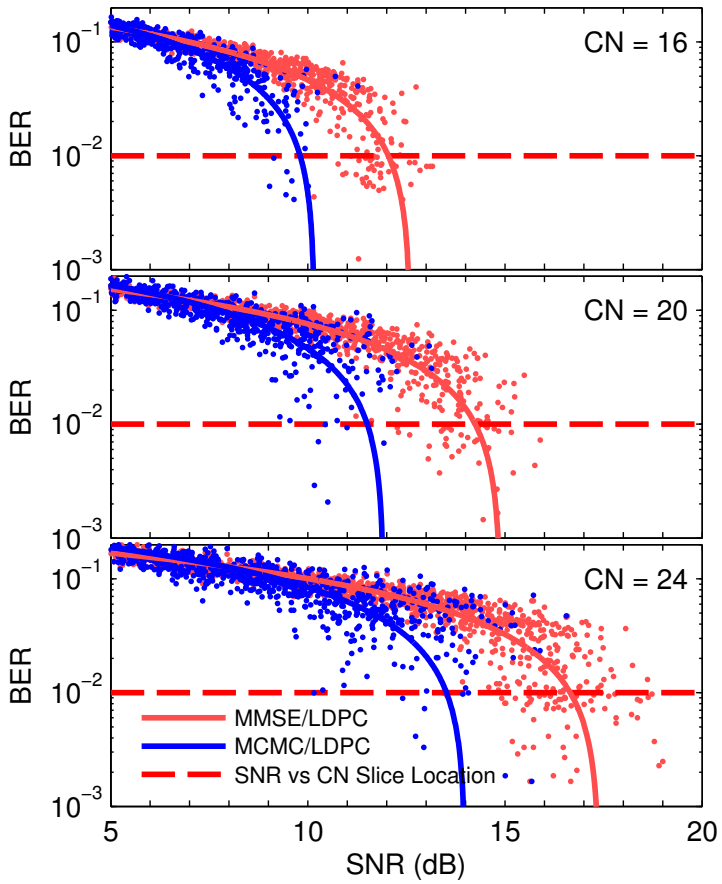


Figure 4.8. Simulated BER slices through CN $= [16, 20, 24] \pm 1$dB. Solid line is linear estimate of trend. Dashed line is location of horizontal slice showing SNR vs CN trend in Fig. 4.10a.

size of 1500 bytes.

The previous results are good for quantitative comparison between methods, but in indoor environments it is generally not obvious how SNR and CN are related to range because of the complex, unusual combinations of multipath and line-of-sight. This makes it difficult to decide whether the increased computation of an MCMC iteration is worth the performance improvement. An additional qualitative data set was collected as shown in Fig. 4.11. Primary test locations are in offices to the left of the hallway. Results were recorded as a simple pass/fail for connectivity in the room and compiled into a representative image to show maximum range. In this specific scenario, an approximate 60% improvement in range was seen with MCMC.

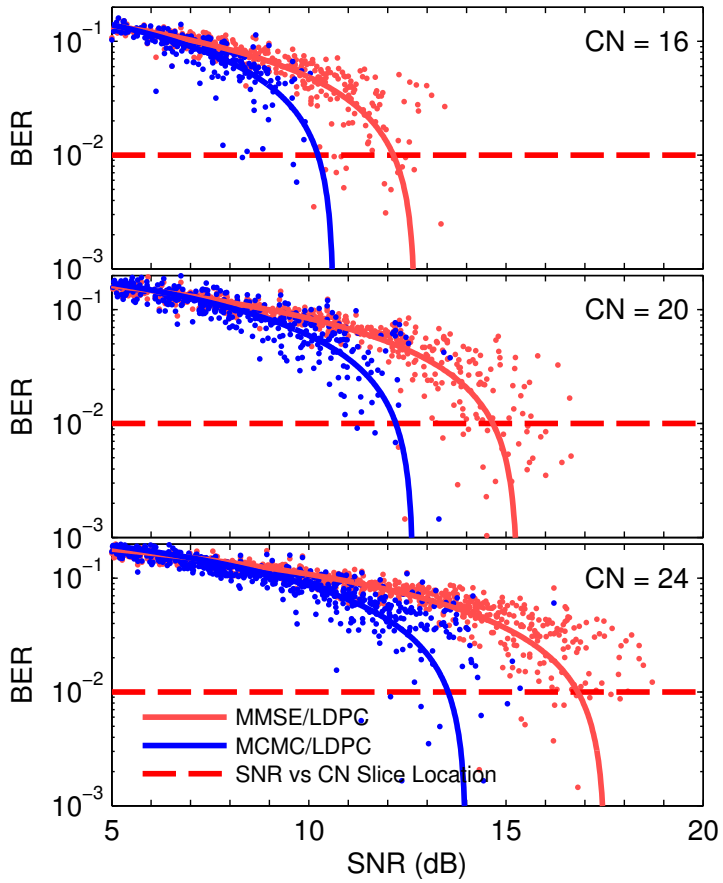
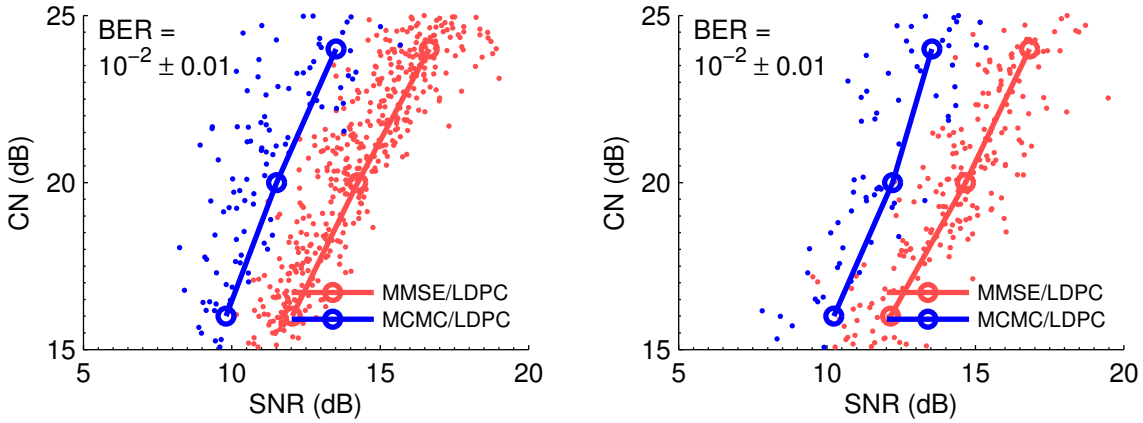


Figure 4.9. Measured BER slices through CN $= [16, 20, 24] \pm 1$dB. Solid line is linear estimate of trend. Dashed line is location of horizontal slice showing SNR vs CN trend in Fig. 4.10b.



(a) Simulated (b) Measured

Figure 4.10. 2D slice through the 3D BER/SNR/CN space. Dots are packets with BER $= 10^{-2} \pm 0.01$. Circles and lines show the intersection of the BER trend lines from Fig. 4.8 and 4.9.

Table 4.1. SNR at BER $= 10^{-5}$. Compiled from extending trend lines on Fig. 4.8 and 4.9.

SNR(dB) @ BER=10^{-5}				
	Condition Number (dB)	**16**	**20**	**24**
Simulated	**MMSE/LDPC**	12.6	14.9	17.4
	MCMC/LDPC	10.2	12.1	14.1
	Improvement	2.4	2.8	3.3
Measured	**MMSE/LDPC**	12.6	15.2	17.6
	MCMC/LDPC	10.6	12.6	14.0
	Improvement	2.0	2.6	3.6

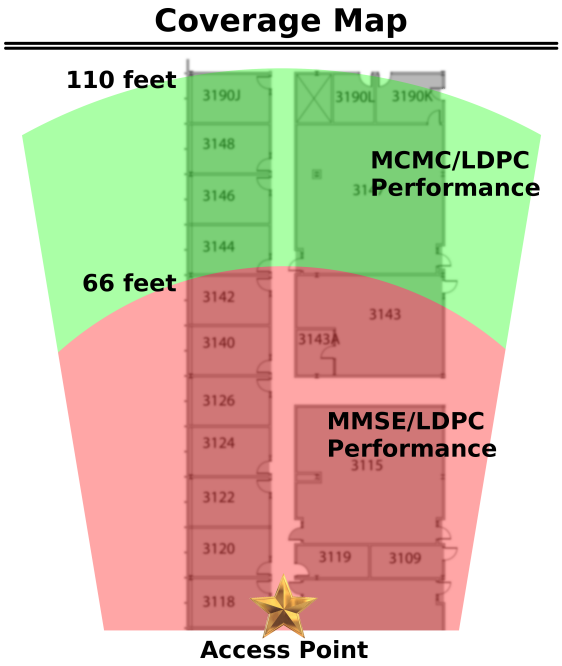


Figure 4.11. Example coverage map of real-world performance on the 3rd floor of the University of Utah Merrill Engineering Building. Transmitter placed in hallway at location labeled "Access Point".

4.4 8-Antenna "Ettus USRP2" MIMO Testbed

The 8x8 MIMO testbed based on the now discontinued Ettus USRP2 software-defined-radio was a temporary system used as a proof of concept before building the newer and higher performing Ettus B210 based system described in Section 4.5. The most important aspects of the USRP2 system shown in Fig. 4.12 is that it is a completely open-source architecture making it much easier to work with and solving many of the problems encountered with the 4x4 National Instruments testbed. It was limited to only the 2.4 GHz and 5 GHz WiFi bands and so could not be used in future LTE testing. For more information on the 8x8 Ettus USRP2 testbed, see Appendix A.

No results will be shown for this specific testbed because the results in Section 4.5 supersede them.

4.5 8-Antenna "Ettus B210" MIMO Testbed

The goal of the 8-antenna B210 testbed is to perform thorough 8x8 MIMO testing with the new X-MCMC detector as described in Chapter 3. Here, we present measurement-based results demonstrating that X-MCMC attains near maxlog-MAP performance for both 4-antenna and 8-antenna 64-QAM 802.11ac WiFi communications. The testbed measurements are matched to simulations to verify the validity of our simulation methodology.

4.5.1 Hardware

The 8x8 MIMO testbed based on the Ettus B210 software-defined-radio shown in Fig. 4.13 is our third MIMO testbed. The requirements in its construction were to have a tunable range including all of the LTE and WiFi bands, have a minimum 20 MHz of usable bandwidth, and have 30 dB of SNR so that 64 QAM WiFi results could be collected. This testbed has met all of these requirements at approximately 1/20 of the cost of a comparable National Instruments FlexRIO system and 1/3 of an equivalent USRP2 system. We are the first to demonstrate greater than 2x2 MIMO with this SDR. For more information on the 8x8 Ettus B210 testbed, see Appendix A.

4.5.2 Algorithm

The primary algorithm that will be demonstrated in the following section is the X-MCMC detector introduced in Chapter 3. This X-MCMC detector includes the excited Gibbs

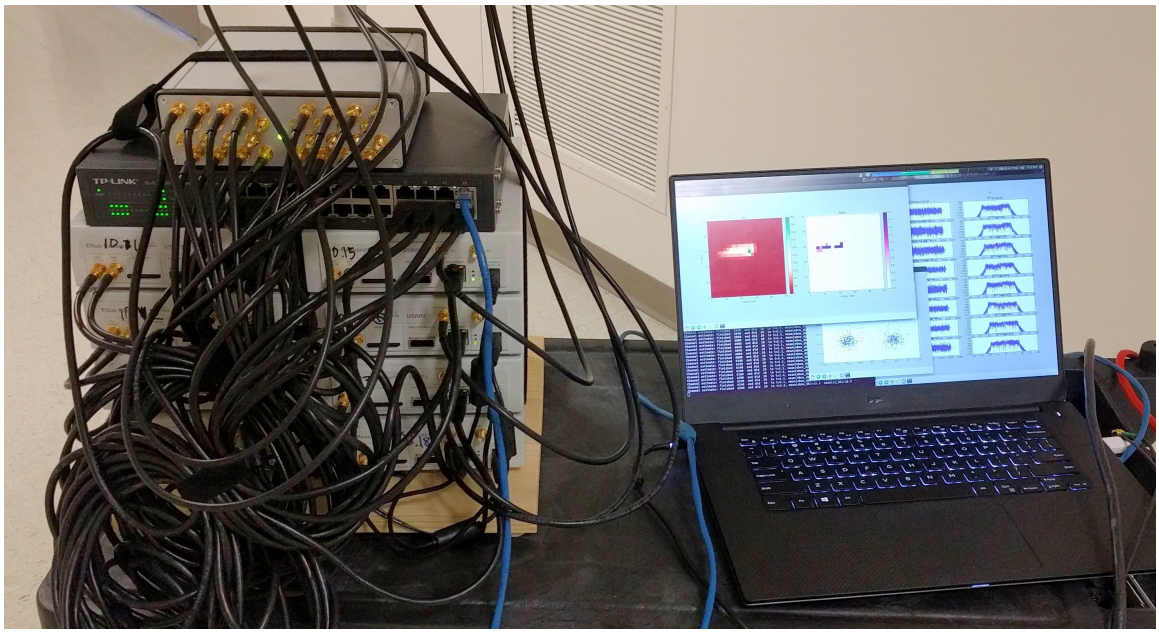


Figure 4.12. 8-antenna experimental testbed based on Ettus USRP2 software defined radios.

Figure 4.13. 8-antenna experimental testbed based on Ettus USRP B210 software defined radios.

sampler which solves the high SNR stalling problem, a 1-bit forced bit flip to escape pseudo-convergence, and output LLR overconfidence conditioning. There are also several additional algorithms used for comparison. The MMSE initialized MCMC detector described in Section 2.7 and Chapter 2 is used to demonstrate how large of an improvement X-MCMC is over prior work. The K-Best variety of sphere-decoding is also shown as a reference point for the larger MIMO detector literature which commonly uses this algorithm [18]. The Max-MAP algorithm [35] is used as a theoretical bound for 4-antenna results whereas a very large K-Best detector is used as an approximate bound on the 8-antenna results since such a large number of simultaneous bits is too complex to compute with Max-Map and K-Best is known to have near-MAP performance [18].

4.5.3 Results

The following testbed measurement results were transmitted using an 802.11ac packet structure over the system described in Section 4.5.1 in the 2.4 GHz WiFi band. The low-density parity-check (LDPC) code described in the protocol was used for both measurement and simulation. Simulations were performed in the frequency-domain without use of the WiFi time-domain packet structure.

The most important factor for matching testbed to simulation results is selection of a channel model with sufficient correlation to produce similar distributions of ill-conditioned $\mathbf{H}$ matrices. In Fig. 4.14, condition number histograms are shown using the same testbed data set as in Fig. 4.15 and 4.16. By comparing these distributions to the channel models in Fig. 4.1, we see that the WiFi TGn Model-C is a reasonable match for the 4-antenna measurements and Model-D for the 8-antenna measurements. These specific models have been used in the corresponding simulation results of Fig. 4.17 and 4.18.

Secondly, channel realizations simulated over an LDPC block should not be independent. A good procedure is to generate a single time-domain channel realization for each LDPC block, and then extract the same 52 active OFDM subcarrier realizations as used in 802.11ac. If more than 52 realizations are needed for the LDPC block, then they are reused. This results in the likelihood of an LDPC block encountering many deep fades simultaneously much more likely resulting in a significant shift of the BER curves to the right.

In the 4-antenna results of Fig. 4.15 and 4.17, it can be seen that X-MCMC approaches

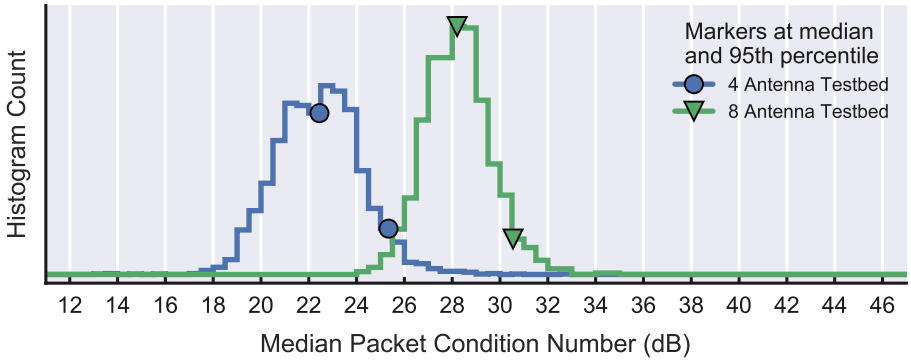


Figure 4.14. Distribution of the channels observed in the testbed data used to generate BER curve results. Median is across the N_{subc} channels needed for one LDPC codeword.

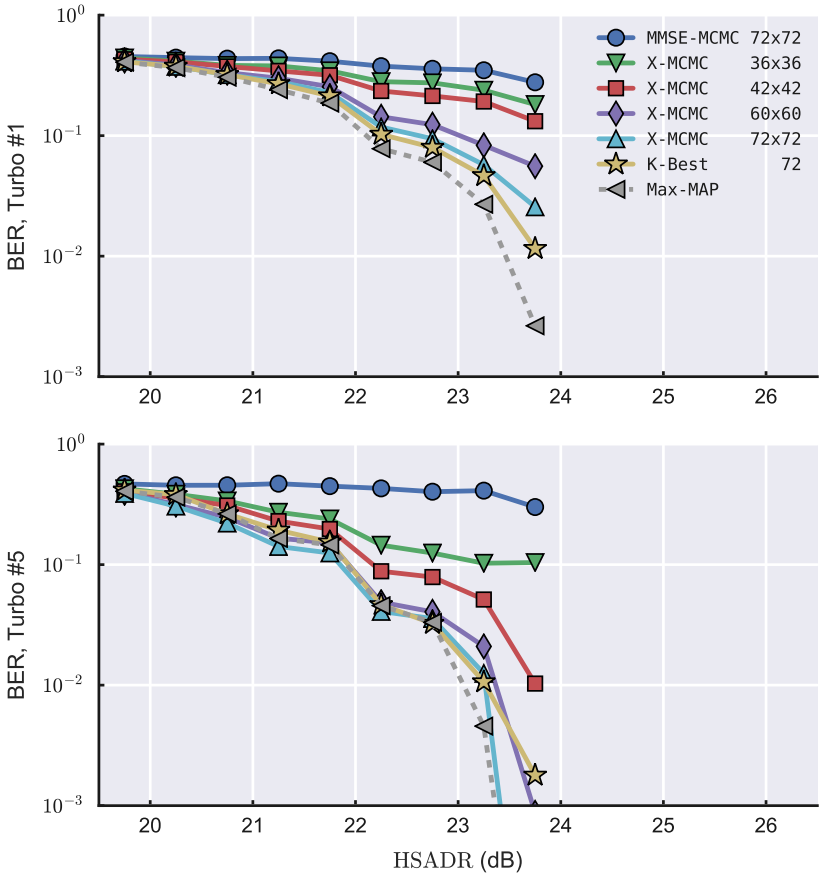


Figure 4.15. Testbed transmitted data using 802.11ac. 4 antenna, 64 QAM, 3/4 LDPC coding rate BER curves. MCMC values are $N_{\text{gibbs}} \times N_{\text{iter}}$ and K-Best values are the list length.

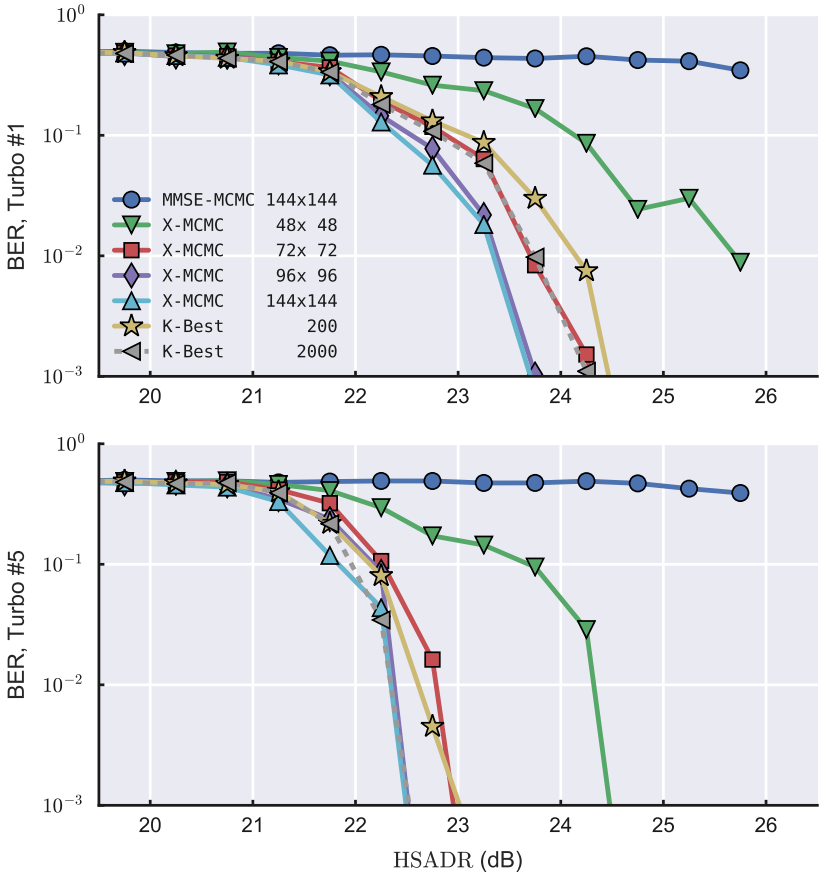


Figure 4.16. Testbed transmitted data using 802.11ac. 8 antenna, 64 QAM, 3/4 LDPC coding rate BER curves. MCMC values are $N_{\text{gibbs}} \times N_{\text{iter}}$ and K-Best values are the list length.

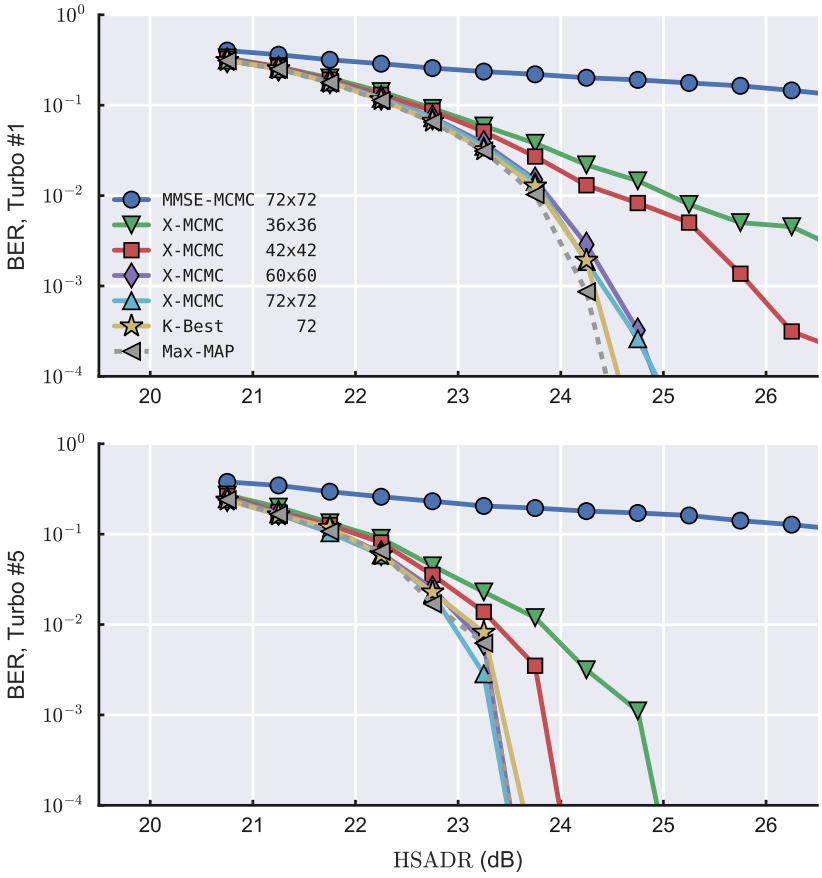


Figure 4.17. Simulation with WiFi TGn Model-C channel model. 4 antenna, 64 QAM, 3/4 LDPC coding rate BER curves. MCMC values are $N_{\text{gibbs}} \times N_{\text{iter}}$ and K-Best values are the list length.

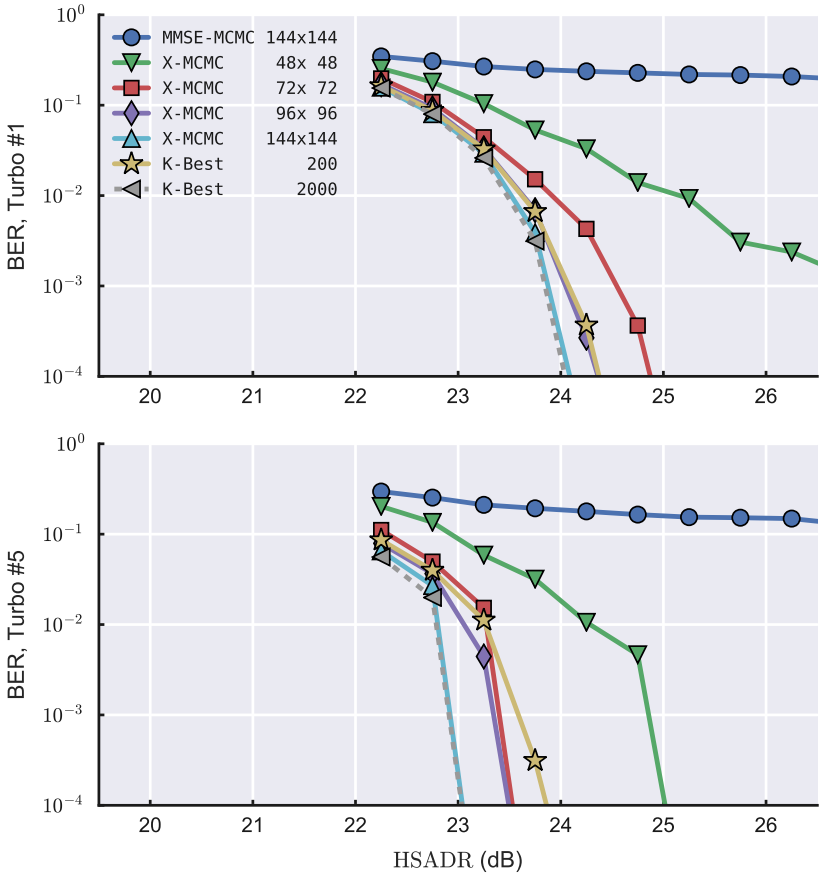


Figure 4.18. Simulation with WiFi TGn Model-D channel model. 8 antenna, 64 QAM, 3/4 LDPC coding rate BER curves. MCMC values are $N_{\text{gibbs}} \times N_{\text{iter}}$ and K-Best values are the list length.

near Max-MAP performance as the number of Gibbs samplers increases. Similarly, the 8-antenna results of Fig. 4.16 and 4.18 confirm this, but since MAP is too complex to calculate on the 8-antenna results, a very large K-Best detector is used as an estimate of the Max-MAP performance bound. A moderate-sized K-Best detector is used as a reference point so that the difficulty of this channel can be better appreciated. Notice that the MMSE initialized MCMC detector does not converge whereas the X-MCMC detector converges to near Max-MAP performance. This verifies our claims that X-MCMC has solved the high SNR stalling problem.

Even after using a well matched channel model and the HSADR interference metric in (4.16), the waterfall locations of the simulation results do not perfectly match measurement in Fig. 4.15 and 4.16. This is acceptable and to be expected as the distribution of condition numbers do not perfectly match and HSADR is an imperfect metric. What is important is that they are close (within 1-2 dB), the convergence behavior of all detector methods agree, and the parameters of MCMC and K-Best necessary for convergence are similar. This means that in the future these simulation techniques and models can be used to perform a much deeper analysis of MCMC and X-MCMC without performing real-world tests. If a closer match between simulation and measurement results is desired, see our previous work in Section 4.3.3 [54] which uses condition number slicing to control channel distributions. Slicing is no longer our preferred method as it removes the contribution of outliers from the data sets.

4.6 Summary

In this chapter, we presented methods to match simulation to testbed measurements. This includes using more representative ill-conditioned channel models, using condition-number of the channel matrix to identify and possibly sort realizations through slicing, and using the new HASDR distortion metric during analysis to properly display BER curves. Through this detailed work with real-world measurements, we also showed that the Gaussian i.i.d. channel models commonly used for MCMC simulations are insufficient to demonstrate indoor WiFi performance. The construction of a low cost and effective 8-antenna MIMO testbed was described which others may replicate to do testing at all 802.11ac and worldwide LTE frequency bands for 1/20 the cost of some alternative systems. Finally, we verified the high

performance of the X-MCMC detector. The X-MCMC detector was demonstrated to have near maxlog-MAP performance with a reasonable number of Gibbs samplers at up to 8x8 MIMO 64 QAM sizes on the 802.11ac WiFi protocol.

CHAPTER 5

VLSI IMPLEMENTATION

Creating a near optimal MIMO detector is not difficult in simulation. This can be achieved with max-log MAP, sphere-decoding, K-Best, and MCMC to mention just a few. The difficulty is in creating a MIMO detector with both near optimal performance and low complexity. Early estimates of the complexity and cost of an algorithm can be made by simply counting multiplications and additions, but this does not capture all of the potential implementation issues of an algorithm. A perfect example of this is that one of the most difficult and expensive operations needed for real-world implementations of sphere-decoding and K-best is in sorting the list, which is not captured in simple measures of multiplications and additions [18]. For this reason, it is common to write a register-transfer-level (RTL) hardware description and synthesize the design to more throughly understand the details of an algorithm's implementation.

The focus of this chapter is to consider the limitations and opportunities presented by the X-MCMC MIMO detector and demonstrate a viable RTL design. First, we will describe the older algorithm that this work is based upon, Version A, in Section 5.2.1. Then, we follow with some initial changes to the algorithm which do not have an impact on algorithm performance or behavior but that lays the foundation for an efficient hardware implementation. Next, in Section 5.3 we identify the most complex and costly operations in the algorithm, propose suitable approximations, and present simulation results that show they work well in practice. Finally, we include all of the methods and approximations into our proposed X-MCMC hardware design in Section 5.4. Performance and complexity metrics are provided and compared to other MCMC and K-Best implementations found in the literature.

5.1 System Model and Notation

In the discussions and derivations that follow, the notation is the same as elsewhere in the dissertation where the system model is represented by

$$\mathbf{y} = \mathbf{H}\mathbf{s} + \mathbf{n} \ . \tag{5.1}$$

Here, $\mathbf{y}$ is the received signal vector, $\mathbf{H}$ is a slow flat fading complex channel matrix containing the pairwise gain and phase between antennas, $\mathbf{s}$ is the vector of transmitted QAM constellation symbols, and $\mathbf{n}$ is a noise vector. The noise elements are assumed to be independent and identically distributed (i.i.d.) complex Gaussian random variables with variance of σ_n^2 per each real and imaginary dimension. Assuming the transmit and receive side have the same number of antennas N, the dimensions of the vectors and matrices are $N \times 1$ and $N \times N$. The complex symbols $\mathbf{s}$ are mapped from the bit vector $\mathbf{x}$, which may be described as comprising of 1's and 0's or equivalently +1's and −1's.

Some specialized notation is used for compactness and clarity. Vectors and matrices are expressed with bold fonts and the latter are capitalized. The removal of the k^{th} element of a vector is shown with set notation as $\{\cdot\}^{\backslash k}$. A variable or vector derived from the bit sequence $\mathbf{x}$ with the k^{th} bit forced to a 1 or 0 is shown with $\{\cdot\}^{k+}$ and $\{\cdot\}^{k-}$, respectively. When two nearly identical equations are needed differing only in use of $k+$ or $k-$, $k\pm$ is used to represent both versions. If the k^{th} bit is forced to the correct transmitted value, either 1 or 0, it is shown with $\{\cdot\}^{k*}$.

The concept of distance is repeatedly used. It is the closeness of the current state $\mathbf{x}$ to the transmitted sequence. To simplify its use, it will be defined as the square Euclidean distance

$$d = \|\mathbf{y} - \mathbf{H}\mathbf{s}\|^2 \tag{5.2}$$

where $\mathbf{s}$ is the complex symbol mapped version of the bit state $\mathbf{x}$. If the k^{th} bit of $\mathbf{x}$ is forced to a 1 or 0, then this can be indicated on all dependent variables and vectors with a superscript $k+$ or $k-$ as in

$$d^{k\pm} = \left\|\mathbf{y} - \mathbf{H}\mathbf{s}^{k\pm}\right\|^2 \ . \tag{5.3}$$

5.2 Algorithm

5.2.1 X-MCMC: Version A

The algorithm used for the VLSI implementation of X-MCMC is not exactly the same one used in Chapters 3 and 4, but will be shown to be similar in performance. This difference is only due to changes made to the design over time, where the algorithm presented here is an earlier, more complex version based on heuristic motivations. Therefore, we will refer to the older design presented here as *Version A* and the newer one described in Chapter 3 as *Version B*.

There are two differences between Version A and Version B. We recall that Version B is comprised of an excited Gibbs sampler using $\min(d^{k\pm})$ for the scaling factor, a pseudo-convergence detection method based on a lack of state change, a pseudo-convergence escape method based on a forced 1-bit change, and output LLR overconfidence scaling based on $\min(d^{k\pm})$ across all samples. Here, we present Version A which is similar except that the Gibbs excitation uses the recent history of $\min(d^{k\pm})$ and the pseudo-convergence escape is made by further exciting the Gibbs sampler.

5.2.1.1 Excited Gibbs Sampler

Version A of the X-MCMC detector is based on the observation that the root cause of the high SNR stalling problem described in Chapter 2 is that γ_k from (2.7) as in

$$\gamma_k = \frac{(d^{k-} - d^{k+})}{2\sigma_n^2} + \lambda_k^a \tag{5.4}$$

reaches extreme values when sampling far from a correct solution region. This results in the probability values in (2.4) as in

$$P_{\text{gibbs}} = P(x_k = +1 \,|\, \mathbf{y}, \mathbf{x}^{\setminus k}, \boldsymbol{\lambda}^a) = \frac{1}{1 + e^{-\gamma_k}} \quad . \tag{5.5}$$

saturating, creating a nearly deterministic walk that quickly stalls.

When the γ_k values have a small magnitude, P_{gibbs} probabilities are moderate and the desired guided random walk behavior is seen. This occurs when the Euclidean squared distance $\|\mathbf{y} - \mathbf{Hs}\|^2$ is small, and thus the Gibbs sampler is likely close to a correct solution.

Based on these observations, it is apparent that a scaling variable is needed. It should have no effect near the correct solution, and it must keep the sampler active with moderated values of γ_k when far away from the correct solution region. This can be accomplished by

applying a linear scaling factor to (5.4). Thus the P_{gibbs} calculation should use the modified value

$$\gamma_k' = \frac{1}{r_\gamma}\frac{(d^{k-} - d^{k+})}{2\sigma_n^2} + \lambda_k^a \tag{5.6}$$

where r_γ is a time varying, confidence adjusting factor. This factor should be unity when the sampler is close to the solution region, and should become larger as the distance from the correct solution becomes greater. This behavior can be accomplished by taking the ratio between the distance in the current state space region and the expectation of all realizations of the system with correct bit sequences

$$E\left[\|\mathbf{y} - \mathbf{H}\mathbf{s}_{\text{correct}}\|^2\right] = E\left[\|\mathbf{n}\|^2\right] = 2N\sigma_n^2 \tag{5.7}$$

This suggests that r_γ should take the form

$$r_\gamma = \frac{\min_{d\in \mathrm{D}_\gamma}(d)}{2N\sigma_n^2} \tag{5.8}$$

where the minimum is taken over the list D_γ, the $d^{k\pm}$ distances observed since the last change in the Gibbs state $\mathbf{x}$. By using the recent history of distances, the scaling factor takes into account the error in the current region instead of just the most recently calculated k^{th} distance. This heuristic approach has the desired scaling behavior and has proven to work in our analysis. Note that in Version B the list D_γ is only comprised of the current distances without any previous history.

5.2.1.2 Pseudo-Convergence

After applying the dynamic excitation coefficient r_γ to solve the high SNR stalling problem, one finds that the Gibbs sampler will converge quickly to the region of a potentially valid solution. Once this happens, r_γ will be approximately 1 and the Gibbs sampler will stop exploring new solution regions, repeatedly sampling the same bit permutations and collect no new samples. Although symptomatically similar, pseudo-convergence is different from the stalling problem discussed previously. This can be thought of as being trapped in a local-well of a gradient descent method. Therefore, we measure the depth of the well, and increase the Gibbs excitation by adding the factor r_p to escape as in

$$\gamma_k'' = \frac{1}{r_\gamma + r_{\text{p}}}\frac{(d^{k-} - d^{k+})}{2\sigma_n^2} + \lambda_k^a \ . \tag{5.9}$$

Here, the desired r_{p} should be zero normally, but needs to add enough excitation to exit the current well when stalling has occurred. This can be done with

$$r_{\mathrm{p}} = \begin{cases} 0, & \text{normally} \\ \frac{3}{2N\sigma_n^2} \min_{d \in \mathrm{D_p}}(d)\,, & \text{pseudo-convergence} \end{cases} \tag{5.10}$$

where the minimum is across the list of neighboring distances $\mathrm{D_p}$ and the pseudo-convergence case is triggered when the Gibbs sampler has not moved during the previous full cycle of K-bits. In this context, a neighboring distance is created from a bit sequence with Hamming distance of 1 compared to the currently pseudo-converged state, assumed to be the bottom of a local-well. This definition and depth estimate is appropriate since we assume that the depth of the well is much larger than the distance at the bottom, and therefore, we can approximate the depth with only the neighboring distances.

An intuitive way to think of this is that the Gibbs sampler is stuck in the bottom of a mountain valley. It looks around at the mountain ridges around it, that is, the neighboring distances, and selects the lowest point to exit the valley, that is, the pseudo-converged region.

Note that there is a heuristically determined extra factor of 3 which increases the moderation of probabilities, thus increasing the speed of exiting the well.

5.2.1.3 Output LLR Conditioning

Since X-MCMC has a variable convergence time, it is still possible that for a small percentage of cases it does not converge to a stationary posterior distribution in the predetermined fixed number of iterations. With a large enough number of samples, the probability of poor convergence approaches zero, but since it is desirable to minimize the complexity of the design and therefore the number of iterations, it should be considered likely in practice. This can be detected and mitigated by similar metrics and methods as in (5.6) and (5.8) by expanding it to use the best distance observed across all Gibbs samplers. Therefore we specify the metric

$$r_\lambda = \frac{\min_{d \in \mathrm{D}_\lambda}(d)}{2N\sigma_n^2} \tag{5.11}$$

where the minimum is taken over the list D_λ, of all $d^{k\pm}$ distances calculated in all parallel Gibbs samplers.

This distance ratio has a desirable characteristic that it becomes approximately unity when the MCMC detector has sampled the transmitted bit sequence. If the ratio r_λ is much

larger than 1, it can be directly used to scale down the overly confident extrinsic LLR values that otherwise could confuse the forward-error-correcting (FEC) decoder with poor quality statistics. Thus we propose applying (5.11) to condition the output LLRs as in

$$\begin{aligned}\lambda_k^{e\prime} =& \frac{1}{2}\min_{\mathbf{x}\in\mathrm{Z}^{k-}}\left(\frac{1}{r_\lambda\sigma_n^2}\left\|\mathbf{y}-\mathbf{H}\mathbf{s}^{k-}\right\|^2-\mathbf{x}^{\backslash k}\cdot\boldsymbol{\lambda}^{a,\backslash k}\right)\\ &-\frac{1}{2}\min_{\mathbf{x}\in\mathrm{Z}^{k+}}\left(\frac{1}{r_\lambda\sigma_n^2}\left\|\mathbf{y}-\mathbf{H}\mathbf{s}^{k+}\right\|^2-\mathbf{x}^{\backslash k}\cdot\boldsymbol{\lambda}^{a,\backslash k}\right) \quad . \end{aligned} \tag{5.12}$$

5.2.1.4 Summary

Here, we have presented a brief heuristic explanation of the Version A X-MCMC detector; see Chapter 3 for a complete explanation and derivation of the very similar Version B. The differences between Version A and Version B X-MCMC detectors is in how the Gibbs excitation is computed and how pseudo-convergence escape is achieved. In Fig. 5.1, the effect of these differences can be seen. Notice that the performance is indistinguishable between using the minimum-history versus only the most recent minimum distance ('h' vs 'm') for Gibbs excitation. There is a small difference between the well-depth and 1-bit change ('w' vs 'b'), with the Version A well-depth method presented here performing better. Since there is no difference with larger numbers of Gibbs samplers, it is clear that the well-depth method is improving the sampling efficiency slightly. Since this method is more complex and not fully understood, it is not analyzed in detail in Chapter 3.

Since the performance of Versions A and B are nearly identical and have only minor algorithmic differences, we will use our early work on Version A for the analysis that follows.

5.2.2 Prior Information and Turbo Loops

The X-MCMC detector is compatible with turbo iterations, the exchange of soft-information between the detector and decoder as in Fig. 1.2. This allows for the iterative joint detection of the signal for enhanced performance. The soft-information is in the form of log likelihood ratios (LLRs) and represented by λ^a and λ^e for their *a priori* input and extrinsic output versions.

Using turbo loops is interesting from a theoretical point of view but is generally not used in modern 2-way communications systems such as WiFi 802.11ac [7] and LTE cellular [9]. This is due to the fact that tight latency requirements must be met which limit the amount of processing time allowed. Therefore we will not include prior information or turbo iterations

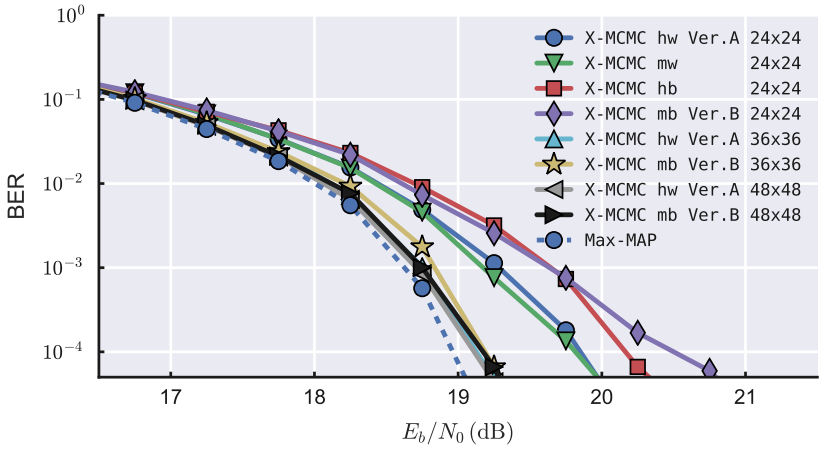


Figure 5.1. Performance difference between the Version A and Version B X-MCMC detectors. The Gibbs excitation of Ver. A using the minimum-history is represented with "h" whereas the Ver. B minimum with "m". The pseudo-convergence escape of Ver. A using the well-depth is represented with "w" whereas the Ver. B escape using a 1-bit forced flip with "b". Parameters: 4 antennas, 64 QAM, WiFi TGn-D channel, 3/4 rate WiFi LDPC encoding.

in any of the analysis that follows, though it may be simply added in the future.

5.2.3 Real-Valued Representation

The MIMO system model and X-MCMC algorithms are generally presented in a representation with complex numbers. In a VLSI design, it is easier to work with a real-valued representation which performs the same calculations. This can be be done by replacing the complex system model of (5.1) with the real-valued versions

$$\mathbf{y} = \begin{bmatrix} \Re\{\mathbf{y}_c\} \\ \Im\{\mathbf{y}_c\} \end{bmatrix}, \ \mathbf{H} = \begin{bmatrix} \Re\{\mathbf{H}_c\} & -\Im\{\mathbf{H}_c\} \\ \Im\{\mathbf{H}_c\} & \Re\{\mathbf{H}_c\} \end{bmatrix},$$
$$\mathbf{s} = \begin{bmatrix} \Re\{\mathbf{s}_c\} \\ \Im\{\mathbf{s}_c\} \end{bmatrix}, \text{ and } \ \mathbf{n} = \begin{bmatrix} \Re\{\mathbf{n}_c\} \\ \Im\{\mathbf{n}_c\} \end{bmatrix}. \tag{5.13}$$

Here, $\{\cdot\}_c$ represents the complex version of a variable and $\Re\{\cdot\}$ and $\Im\{\cdot\}$ extract the real and imaginary components.

5.2.4 Delta Calculation

The most complex portion of the MCMC algorithm is in computing $\|\mathbf{y} - \mathbf{Hs}\|^2$. To avoid repeating this full calculation at every step of the Gibbs sampler, we follow the approach in [43] where only the difference between steps is calculated. We refer to this generically as

a *delta* calculation because only the difference, or delta, between steps is needed. Here, we will present a brief explanation of the method, for details see [43]. Note that the real-valued representation of all variables is used here for later convenience in use with the hardware implementation.

The algorithm begins by pre-calculating the values $\tilde{\mathbf{H}}$ and $\tilde{\mathbf{y}}$.

$$\tilde{\mathbf{H}} = \frac{1}{2\sigma_n^2}\mathbf{H}^T\mathbf{H} \tag{5.14}$$

$$\tilde{\mathbf{y}} = \frac{1}{2\sigma_n^2}\mathbf{H}^T\mathbf{y} \tag{5.15}$$

These values may be calculated once and provided to all of the parallel Gibbs samplers.

Each Gibbs sampler computes an initial distance $\tilde{d}^{(0)}$ using a unique and random starting $\mathbf{x}$ sequence where

$$\tilde{d} = \frac{1}{2\sigma_n^2}\|\mathbf{y} - \mathbf{H}\mathbf{s}\|^2 \ . \tag{5.16}$$

When using the real-representation of the symbol vector $\mathbf{s}$, each element is a half-symbol representing either the real or imaginary portion of a full complex symbol. The s_m half-symbol is the element affected by the x_k bit. Therefore, at each step of the Gibbs sampler, we compute the intermediate variables

$$\kappa_1 = \frac{1}{2}\left(\left(s_m^{k+}\right)^2 - \left(s_m^{k-}\right)^2\right) \tag{5.17}$$

$$\kappa_2 = s_m^{k+} - s_m^{k-} \ . \tag{5.18}$$

Next, we compute the change in distance produced from a change in the state of bit x_k with

$$\delta = \tilde{d}^{k-} - \tilde{d}^{k+} = \kappa_2\left(\tilde{y}_m - \sum_{j\neq m}\tilde{h}_{mj}s_j\right) - \kappa_1\tilde{h}_{mm} \tag{5.19}$$

where $\tilde{h}_{ij}$ is the $(i,j)^{\text{th}}$ element of $\tilde{H}$. This is used in producing

$$\gamma_k = \frac{1}{r_\gamma + r_\text{p}}\left(\tilde{d}^{k-} - \tilde{d}^{k+}\right) = \frac{\delta}{r_\gamma + r_\text{p}} \tag{5.20}$$

where r_γ and r_p are the excitation and pseudo-convergence scaling factors found in (5.8) and (5.10).

Then P_{gibbs} is computed with (5.5) and compared to a uniform random variable to make a choice of changing the k^{th} bit of the current state. Once this choice is made, the current distance $\tilde{d}^{(n)}$ at time n can be updated as in

$$\tilde{d}^{(n)} = \begin{cases} \tilde{d}^{(n-1)}, & \text{bit not changed} \\ \tilde{d}^{(n-1)} - \delta, & \text{bit changed to +1} \\ \tilde{d}^{(n-1)} + \delta, & \text{bit changed to -1} \end{cases} . \tag{5.21}$$

Rather than keeping a list of all visited states as in (5.12) and finding the minimum after all Gibbs iterations are complete, we save the minimum values as the algorithm progresses. Starting from a large initialization value

$$\eta_{j,\pm 1}^{(0)} = \infty \tag{5.22}$$

we update the best minimum distance at every step

$$\eta_{j,x_j}^{(n)} = \min\left(\eta_{j,x_j}^{(n-1)}, \tilde{d}^{k\pm}\right) . \tag{5.23}$$

Here, j is a bit index $0 \leq j < K$ and there are two versions of η_j, where the corresponding j^{th} bit is +1 or -1. In the original delta-calculation in [43], only $\eta_{k,\pm 1}$ is updated at each step, that is, only 2 of the $2K$ values as in

$$\eta_{k,\pm 1}^{(n)} = \min\left(\eta_{k,\pm 1}^{(n-1)}, \tilde{d}^{k\pm}\right) . \tag{5.24}$$

Generally, it is possible to update $K+1$ of the $\eta_{j,\pm 1}$ values for very little additional hardware cost since at each Gibbs step we have the distances $\tilde{d}^{k\pm}$ calculated for the bit states $\mathbf{x}^{k+}$ and $\mathbf{x}^{k-}$. The performance difference between the $\eta_{k,\pm 1}$ update and full asymmetric η_{j,x_j} update will be examined in Section 5.3.3.

Finally, after a sufficient number of Gibbs sampler iterations have been made, the final η values may be used to compute the extrinsic output LLR

$$\lambda_k^e = \frac{1}{r_\lambda}\left(\eta_{k,\text{-1}} - \eta_{k,\text{+1}}\right) \tag{5.25}$$

where r_λ is the output conditioning value in (5.11).

5.3 Approximations

The delta-calculation is a good foundational algorithm for a potential VLSI hardware design, but it can be simplified further by making approximations to some of the most

expensive steps. Here, we use the term expensive or costly as a general term to mean a combination of large silicon area and power consumption which is undesirable. In the algorithm outlined in Section 5.2.4, there are several expensive calculations that we will examine in this section for simplification. These include the reciprocal operations of (5.20) and (5.25), the complex P_{gibbs} calculation in (5.5), the η update of (5.23), and the question of whether the scaling factors r_γ and r_λ can be updated 1-step late.

5.3.1 Reciprocal

Using a generic division operation should be avoided at all costs in a potential VLSI design. When absolutely necessary, they are often implemented as a reciprocal and multiply operation. Although an improvement, an accurate reciprocal is also very expensive, often using an iterative approach such as an approximate Newton-Raphson method [109]. Here, we suggest that for the MCMC algorithm an extremely crude power-of-2 accurate reciprocal is sufficient. If valid, this is extremely efficient as the only operations necessary are to detect the location of the leading bit in the denominator and then apply a shift operation to the numerator.

In Fig. 5.2, we see the effect of some potential gross approximations to the reciprocal operation. The $1/x$ operation is a 64-bit floating-point operation. The LUT approximations designate the number of leading bits of denominator to use in a reciprocal LUT. Thus the LUT options would be implemented as a look-up and a multiply. These options are just used as a quick test, and if found useful we would explore them further by carefully choosing a LUT value which is just a sum of several power-of-2 values, thus converting the multiply to a few shift-add operations rather than a generic multiply. Notice in the figure that using the LUT based approximations always results in an overestimate because of truncating the denominator.

First, we will examine the effect of the reciprocal approximation on the Gibbs excitation factor of (5.20). In the block diagrams found in Section 5.4, this is referred to as σ_e^{-2} where $\sigma_e^2 = r_\gamma + r_{\text{p}}$. We can see in the BER curve of Fig. 5.3 that all of the approximations work well and that surprisingly, the 1-bit LUT actually outperforms the floating-point operation. This crude approximation is sufficient in this part of the calculation because it is in the stochastic decision portion of the algorithm where there are no wrong answers, only statistical

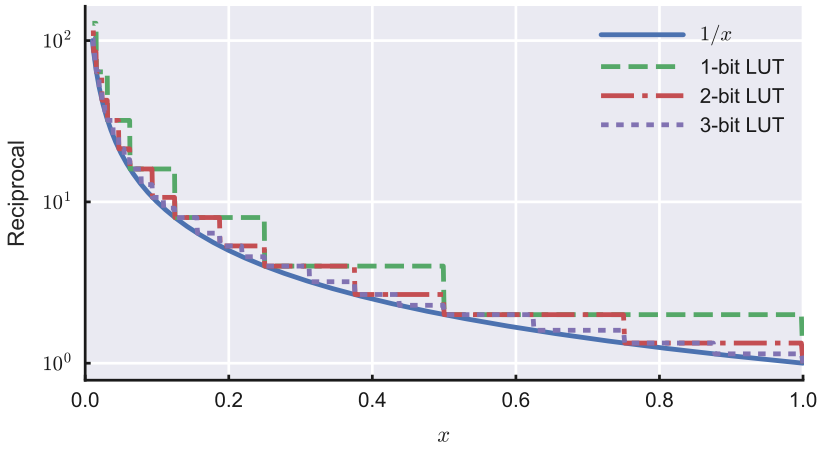


Figure 5.2. Various approximations to a $1/x$ reciprocal operation.

biases. Therefore, the power-of-2 inverse with its bias towards always overestimating the inverse excitation factor is of benefit to the algorithm by making the statistical search slightly colder, i.e. slower and more deterministic. This gives us a clue that, in the future, purposefully biasing the excitation scaling factor should be explored to enhance performance.

Next, we will check if the output LLR calculation of (5.25) can also be crudely approximated with a power-of-2 reciprocal. In the block diagrams found in Section 5.4, this is referred to as σ_z^{-2} where $\sigma_z^2 = r_\lambda$. We can see in the BER curve of Fig. 5.4 that all of the approximations work well, with no perceivable difference between methods. This can be understood because the most important part of the output LLR is the sign which informs the forward-error-correcting (FEC) decoder of what bits have been estimated. Then the magnitude of the LLR specifies a confidence. Using a power-of-2 approximation on σ_z^{-2} results in the values being on average overconfident by a factor of 1.5 and never more than 2, therefore, we assume that this amount of additional variation is safely within the inherent variability of the LLR calculation itself and thus has no ill affect.

5.3.2 Probability Calculation

The calculation of $P_{\text{gibbs}} = 1/(1 + \exp(-\gamma_k))$ in (5.5) is quite expensive but can be approximated quite easily with a LUT containing simpler functions. Here, we show several such approximations in Fig. 5.5 where the $\gamma/4$ approximation is the one proposed by [43]. Note that a piecewise combination of the $\gamma/4$ and $\gamma/8$ functions or higher order could be

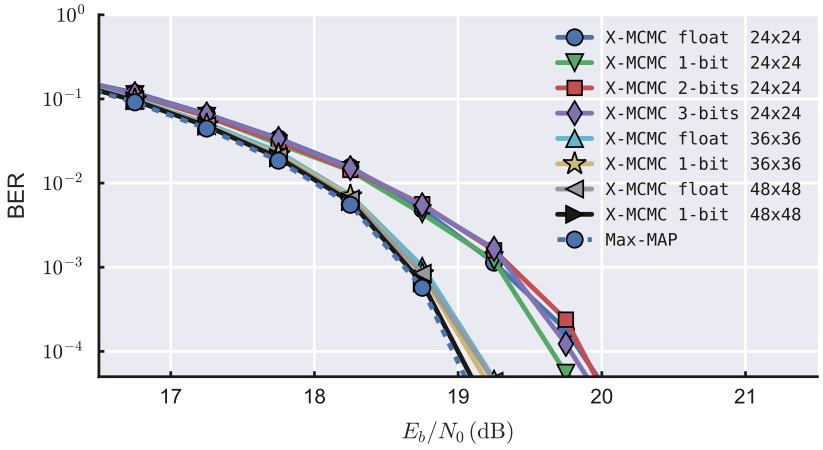


Figure 5.3. BER curves showing effect of approximating σ_e^{-2}. Parameters: 4 antennas, 64 QAM, WiFi TGn-D channel, 3/4 rate WiFi LDPC encoding.

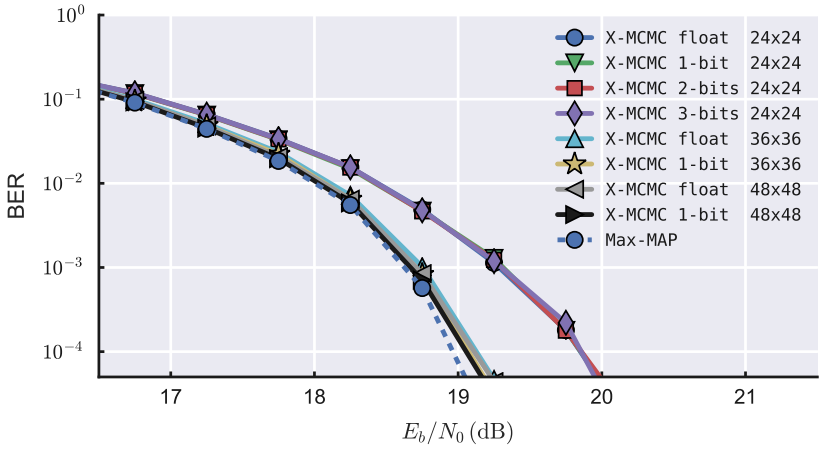


Figure 5.4. BER curves showing effect of approximating σ_z^{-2}. Parameters: 4 antennas, 64 QAM, WiFi TGn-D channel, 3/4 rate WiFi LDPC encoding.

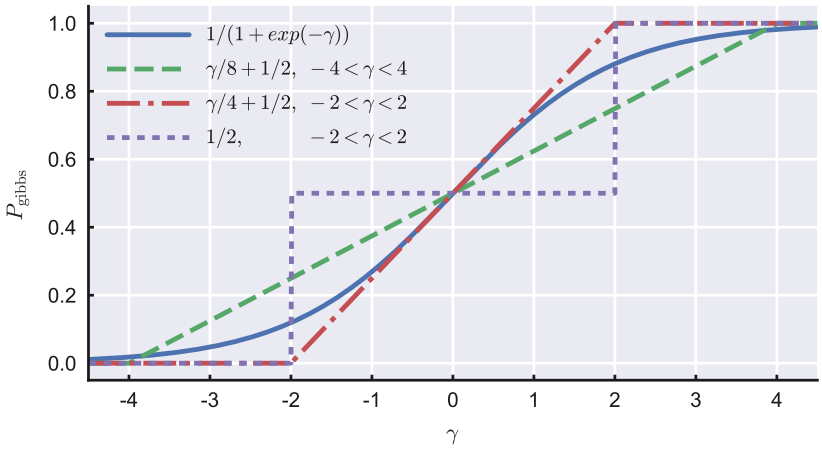


Figure 5.5. Various approximations to P_{gibbs}.

easily implemented, though as we shall see is not necessary.

In Fig. 5.6, we see that the only poor performing approximation is $\gamma/4$, though even that approximation attains near Max-MAP performance with sufficient samples. This makes sense as making a poor approximation here will only reduce the sampling efficiency and not directly affect the final output. The $\gamma/4$ estimate tends to lead to more extreme probabilities which is similar to overestimating γ_k, therefore, we observe that, in this approximation, making the sampler colder is bad whereas the very crude, rectangular approximation is sufficient as it is resulting in a slightly hotter, less deterministic Gibbs sampler.

5.3.3 List Update for Output LLR Calculation

The derivation of X-MCMC results in the output LLR calculation of (5.12) which requires a list of samples to be saved. A straightforward manipulation leads to (5.25) which simply takes the minimum at each Gibbs step instead of performing it at the end. This needs no approximation and results in an asymmetric update in (5.23). By asymmetric we mean that the $\eta_{j,+1}$ and $\eta_{j,-1}$ values are not generally updated at the same time which could be a problem in certain edge cases. In [43], they instead implement the k^{th} update of (5.24) without explanation. In practice, we have found that updating only $\eta_{k,\pm 1}$ has similar performance to a full asymmetric update of η_{j,x_j}, see Fig. 5.7, therefore, we choose to also use the simpler k^{th} update approach.

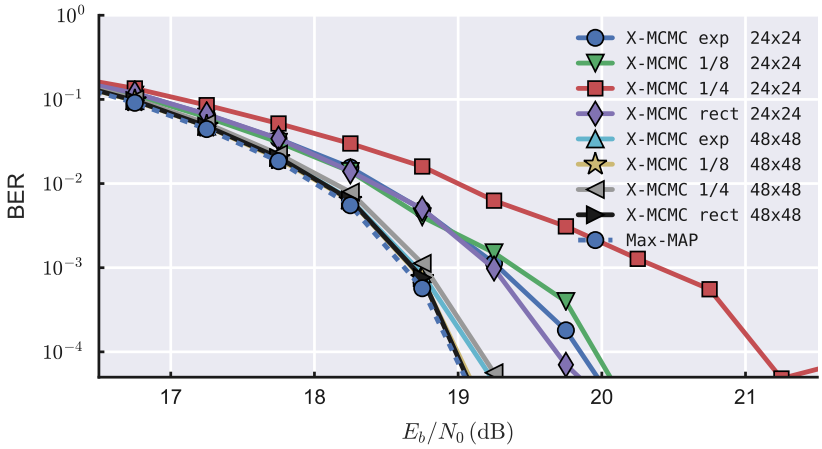


Figure 5.6. BER curves showing effect of approximating P_{gibbs}, where exp is the full calculation, fractions represent the slope used in Fig. 5.5, and rect is the rectangular approximation. Parameters: 4 antennas, 64 QAM, WiFi TGn-D channel, 3/4 rate WiFi LDPC encoding.

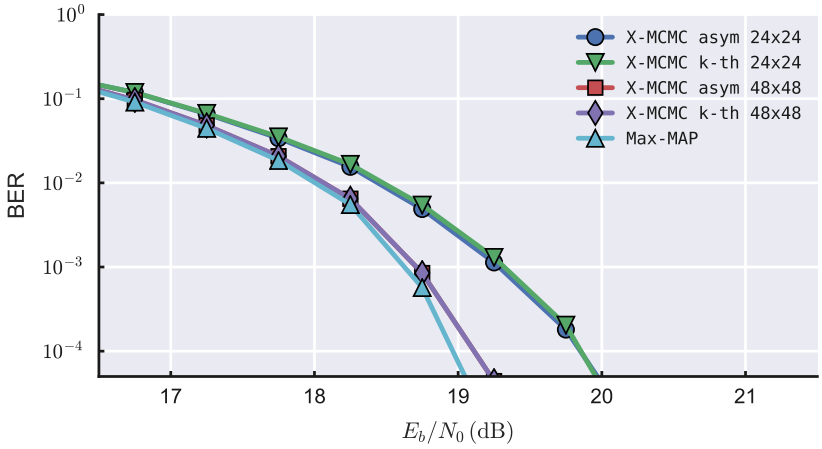


Figure 5.7. BER curves showing effect of asymmetric vs k^{th} update of η . Parameters: 4 antennas, 64 QAM, WiFi TGn-D channel, 3/4 rate WiFi LDPC encoding.

5.3.4 Excitation and Pseudo-Convergence Delay

One of the problems in real-world communication systems is in meeting tight latency requirements, therefore, it is desirable to make the Gibbs sampler run at both a high clock speed and have each Gibbs sampler step occur in as few of clock cycles as possible. One cause of needing multiple clock cycles is in computing values that depend on previous values, i.e. serial calculations in the Gibbs sampler core. The X-MCMC is potentially less efficient to implement than MCMC because it adds two variables, r_γ and r_p, which must be computed after $\tilde{d}^{k\pm}$. Thus, the X-MCMC detector is likely to increase the length of each Gibbs sampler calculation step by 1 clock cycle compared to a base MCMC design. One approach to remove this undesirable increase is to simply calculate the values 1-clock-cycle late. Although not common in digital signal processing, delays commonly occur in analog feedback control loops. From feedback control theory we know that if the feedback delay is short compared to the rate of change of the variables, this approximation should have little effect.

In Fig. 5.8, we examine the effects of adding a 1-clock delay to both r_γ and r_p. For a 24x24 Gibbs sampler, it is clear that adding delay to the pseudo-convergence scaling factor r_p has no effect. This is expected since it is only nonzero after the sampler has not changed state for K steps and therefore by definition there will be no difference. Unfortunately, the 1-clock cycle delay in the Gibbs excitation factor r_γ has a small negative effect. Note that for larger sampler sizes that delay has no effect which means that it is only reducing the sampling efficiency slightly which in our case is likely an acceptable trade-off for reducing the time of a Gibbs sampler step by 1 clock cycle.

5.4 VLSI Design

In the previous sections, we have taken Version A of the X-MCMC detector and have shown methods and approximations in isolation which allow it to be efficiently implemented in hardware. Here, we only consider the full Version A floating-point implementation from Section 5.2.1 and the delta-calculation of Section 5.2.4 with power-of-2 reciprocals, a rectangular LUT for P_gibbs, k^th update on η, and 1-clock delays on both r_γ and r_p.

We present the results of the floating-point Version A vs the simplified fixed-point Version A in the simulation of Fig. 5.9. The floating-point and fixed-point versions are seen to be capable of near Max-MAP performance, though the floating-point design has

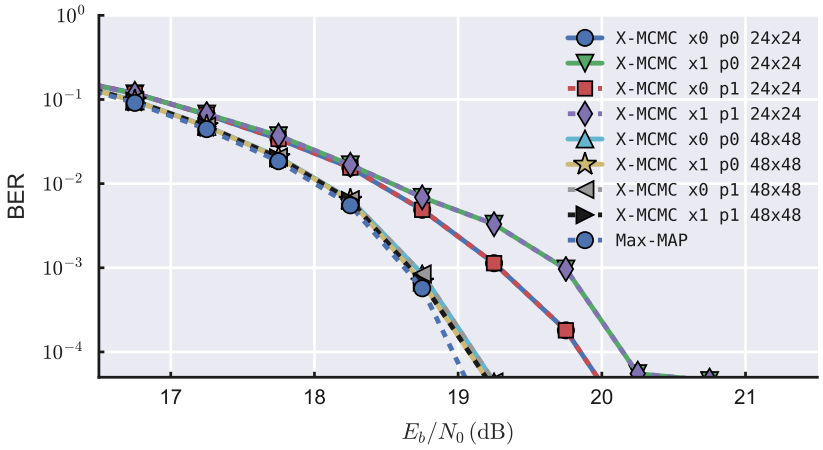


Figure 5.8. BER curves showing the effect of delaying the r_γ and r_p calculations by 1 clock cycle, where $x1$ is a 1-clock delay on the excitation factor and $p1$ is a 1-clock delay on the pseudo-convergence factor. Parameters: 4 antennas, 64 QAM, WiFi TGn-D channel, 3/4 rate WiFi LDPC encoding.

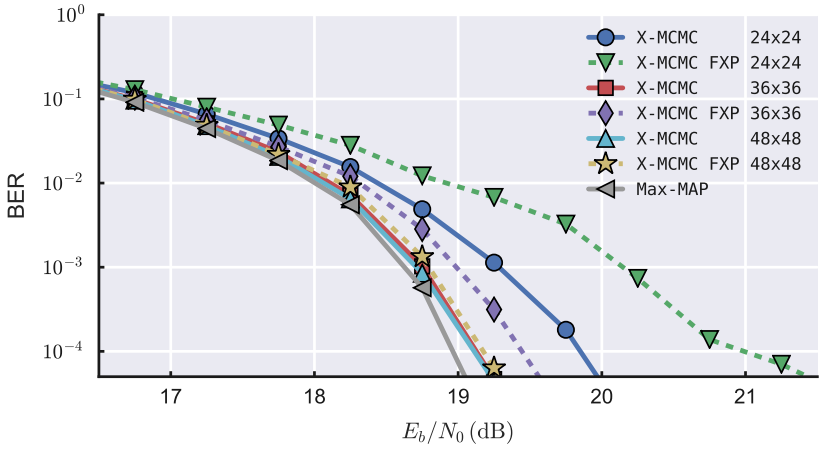


Figure 5.9. BER curves showing the effect of the full floating-point calculation of Section 5.2.4 and a fixed-point (FXP) using 1-bit LUTs for reciprocals, a rectangular LUT for P_gibbs, k^th update on η, and 1-clock delays on both r_γ and r_p. Parameters: 4 antennas, 64 QAM, WiFi TGn-D channel, 3/4 rate WiFi LDPC encoding.

slightly higher sampling efficiency resulting in the performance difference at 24x24. For this simulation, the primary bit widths selected were $\tilde{\mathbf{H}} = Q10.8$, $\tilde{\mathbf{y}} = Q13.8$, and $\sigma_n^2 = Q0.8$, where $Qi.f$ is Q-notation for i integer bits and f fractional bits. These were selected to have minimal impact on the performance of the system.

5.4.1 Block Diagram

Next we show our suggested implementation of the fixed-point hardware design with the block diagrams of Fig. 5.10 and 5.11. In the compact notation of the block diagram, trapezoids are muxes with the dot at the binary zero (false) case, rectangles are registers placed on the thick gray dashed lines of a clock boundary, squares represent combinational logic, large circles are arrays of registers, and thick lines assist in identifying buses of multiple values. Generic components such as the linear-feedback-shift-register (LFSR) random number generator and LUTs are not shown in detail. This design represents a 2 clock cycle core, meaning that each step of the Gibbs sampler takes 2 clock cycles, thus this core can contain 2 Gibbs samplers by doubling all registers and alternating computations between the 2 independent samplers. This is an improvement over [43] which presents a 16-clock-cycle core. Remember that since minimizing latency is an important aspect of most communication systems, that the factor-of-four improvement is significant.

5.4.2 HDL Implementation

The performance of a MIMO detector has little meaning without consideration of its potential cost to use in practice. The most thorough way to measure a designs full complexity and cost is to bring it all the way through synthesis, place and route, and silicon fabrication. This provides a thorough understanding of cost, timing, and power consumption. The drawbacks of a full implementation to silicon is that it is expensive, very time consuming, and the detailed understanding is only relevant to the specific process used in manufacturing the chip. Thus it is desirable to find a middle ground where most implementation difficulties are identified and quantified, the understanding gained is generally applicable, and the time needed to produce the work is small enough to allow quickly iterating and learning from the process. This combination of desirable attributes is mostly captured by bringing a design through synthesis. There are some drawbacks including losing some accuracy in timing and power consumption estimates. Also, some limitations in physical placement and routing will

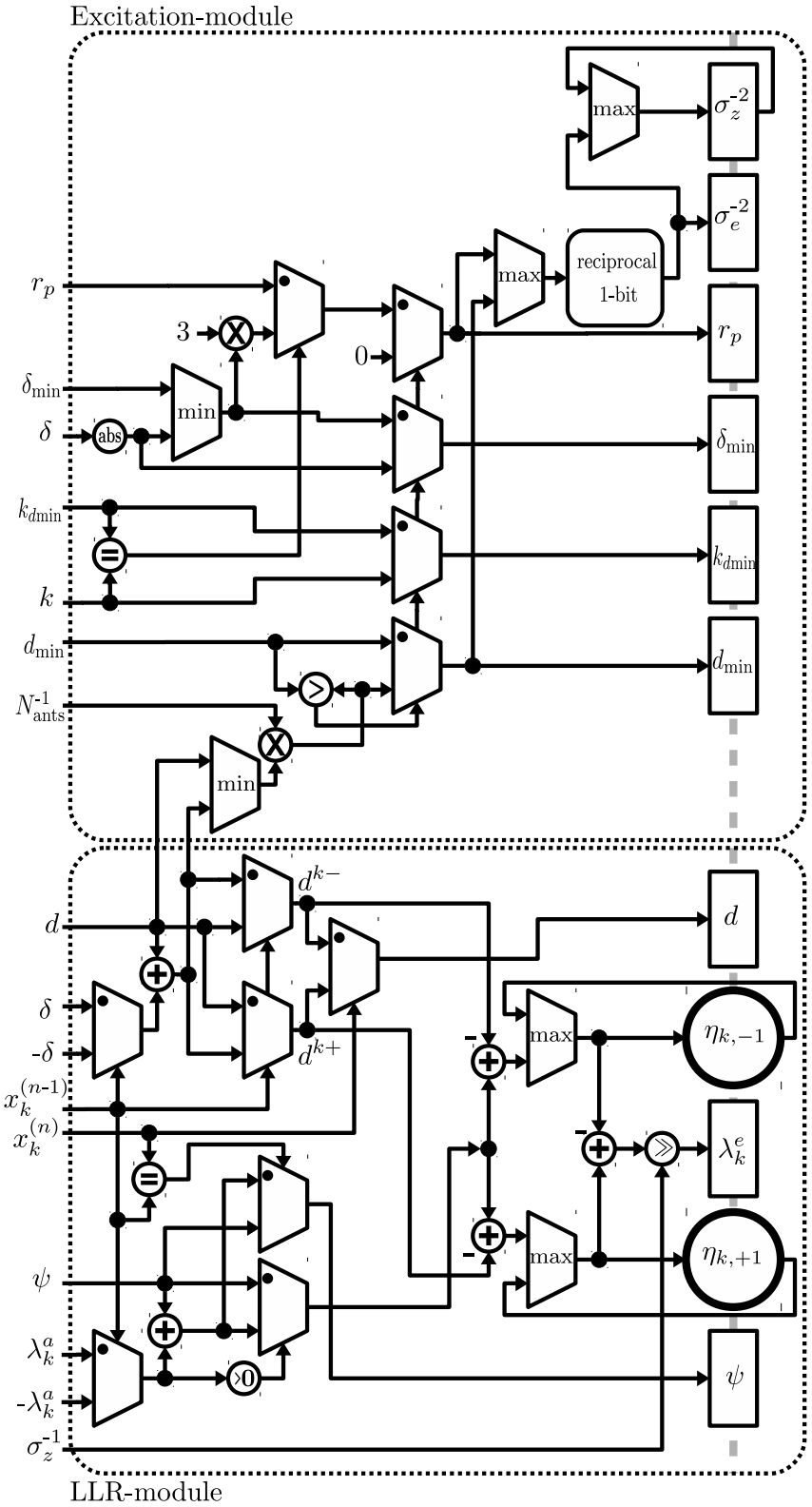


Figure 5.10. Block diagram of X-MCMC Version A including some of the key components needed for excitation, pseudo-convergence mitigation, and LLR calculation.

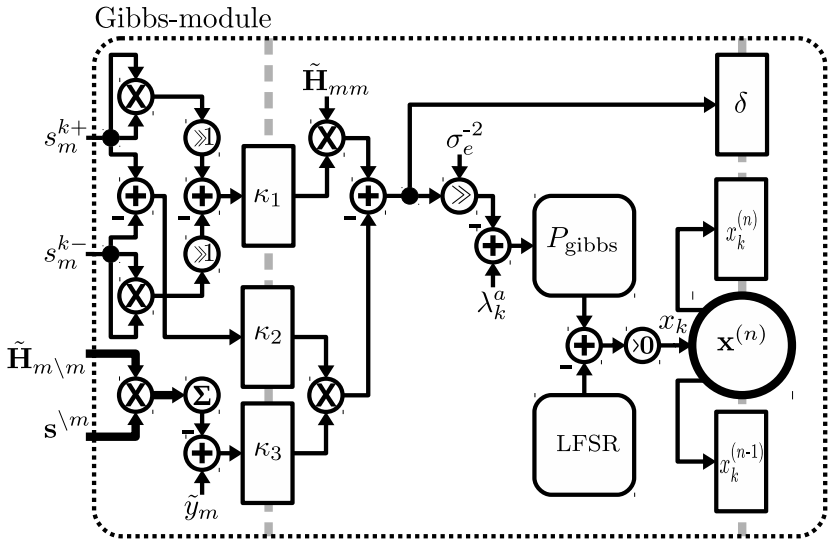


Figure 5.11. Block diagram of X-MCMC Version A including the main components of the Gibbs sampler.

not be identified, but they can largely be avoided by a knowledgeable designer.

To compare the cost of our X-MCMC detector implementation to other MIMO detectors in the literature, we have created a System Verilog implementation of the algorithm. System Verilog is a hardware-design-language (HDL) used to describe the register-transfer-level (RTL) interconnection and functionality of a design suitable for very-large-scale-integration (VLSI). This HDL design can be *synthesized* with a tool such as Synopsys Design Compiler to basic logic gates that can then be *placed and routed* in silicon.

After synthesis, it is possible to compare two designs by their kilo NAND2 gate count equivalent (kGE). This is convenient as it is somewhat process independent, meaning that we can compare results using our open-source 350 nanometer (nm) cell library from Oklahoma State University [110] with another team at a large company who are targeting a 10 nm process with expensive private cell libraries. This is done by taking the final synthesized area and dividing it by the area of a NAND2 logic gate using the same cell library as for synthesis. The next step would be to take the mass of logic cells and place and route them in silicon which would allow for accurate timing and throughput estimates, though limited to that specific cell library and process. Here, we only compare designs through synthesis and do not place and route.

The results of synthesizing our HDL implementation is presented in Table 5.1. This is

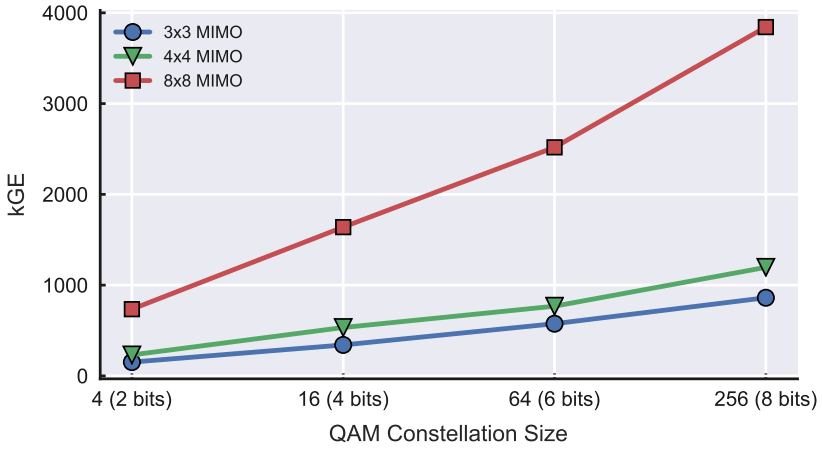
useful for direct comparisons with other designs in the literature as using kGE is standard practice. For a more intuitive understanding of how the number of antennas, QAM size, and cost are related, see Fig. 5.12. Several assumptions have been made to generate the synthesized areas in the table and figures. First, a conservatively large number of Gibbs samplers was selected with $N_{\text{gibbs}} = N_{\text{iter}} = 2N_{\text{ants}} \log_2(N_{\text{qam}})$, i.e. the number of parallel Gibbs samplers and number of iterations is equal to twice the number of bits transmitted in $\mathbf{x}$. Second, that the communications protocol allows for the processing time to increase with increased number of bits transmitted. This last assumption potentially allows for a misinterpretation of Fig. 5.12b which seems to suggest that the complexity increases nearly linearly with the number of transmitted bits. In fact, the complexity here is also increasing quadratically in processing time because the number of bits per iteration and number of iterations are both increasing linearly, therefore, when including both kGE and the number of clock cycles the complexity appears to have a roughly cubic, polynomial growth rate with bits estimated.

To better understand the contribution of the X-MCMC detector hardware design in comparison with other MIMO detector designs we present Table 5.2 with MCMC detectors and Table 5.3 with K-Best detectors. We have attempted to extrapolate the published results to a larger MCMC or K-Best size which should have near Max-MAP performance under our more difficult simulations using a WiFi TGn Model-D channel model. Most authors use an unrealistically easy Gaussian i.i.d. channel model to validate their performance. Since many of these designs use crude approximations and algorithmic shortcuts to reduce the complexity, although they may work for the Gaussian channel they may not function even with the increased Gibbs sampler size, therefore, despite having quantitative numbers, our comparison with other MIMO detectors should only be considered roughly qualitative.

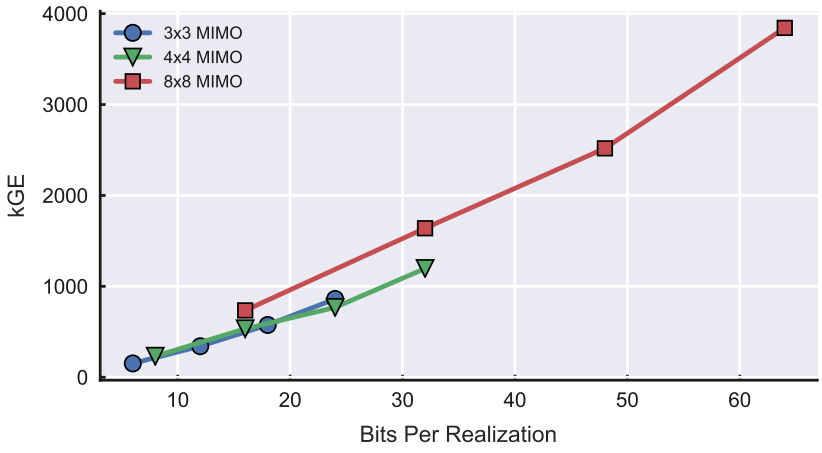
Not considering the fact that the MCMC detectors of Table 5.2 would likely fail due to high SNR stalling, our hardware design is still one of the lowest complexity versions presented. The design by Auras [50] has a much lower complexity than the others because they suggest using a clever way to avoid full multiplications. It should be possible to borrow some of these ideas for our design in future work. The excellent design by [43], using the delta-calculation ours is based on, is not included as it is more focused on FPGA implementation and is therefore inappropriate for silicon design comparison.

Table 5.1. X-MCMC VLSI size in kilo-gate-equivalent (kGE).

	4 QAM	16 QAM	64 QAM	256 QAM
3x3 MIMO	152	341	575	861
4x4 MIMO	231	533	768	1196
8x8 MIMO	735	1639	2518	3843



(a) Analysis of the design complexity vs the QAM size in bits.



(b) Analysis of the design complexity vs the number of bits transmitted per realization.

Figure 5.12. Comparison of the relationship between the number of antennas, QAM size, and number of transmitted bits. Raw data can be found in Table 5.1.

Table 5.2. K-Best VLSI size for 4x4 64 QAM. Needed and extrapolated values are based on simulations with WiFi TGn-D channel models that show a larger list size is needed than what may be selected by the source.

	N_{gibbs}	kGE	needed N_{gibbs}	extrapolated kGE
Deidersen [111]	8	265	192 [1]	6360
Deidersen [111]	8	265	48 [2]	1590
Auras [50]	8	127	192 [1]	3048
Auras [50]	8	127	48 [2]	762
This work	48	768	48	768

[1]: 4x size penalty for stalling.

[2]: No size penalty for stalling. Useful to estimate the design with X-MCMC features added.

Table 5.3. K-Best VLSI size for 4x4 64 QAM. Needed and extrapolated values are based on simulations with WiFi TGn-D channel models that show a larger list size is needed than what may be selected by the source.

	K size	kGE	needed K size	extrapolated kGE
Wenk [112]	10	135	72	972
Chen [113]	64	5270	72	5270
Lin [114]	5	294	72	4234
Shabany [115]	10	114	72	821

There are other excellent hardware designs that we have not included in the table for clarity. Specifically, there are several examples of using stochastic computing in place of true multiplications and additions in the MCMC detector [116–118]. The idea with stochastic math is that true arithmetic can be replaced by surrogates that are fast but less accurate [119]. Although an interesting idea, we believe that stochastic MCMC will likely have poor performance at higher order QAM sizes and with ill-conditioned channels.

The K-Best detector is one of the more common MIMO detector explored in the literature [18, 112–115, 120, 121], therefore, it is important to compare our contribution against this competing design as well. For a 4 antenna 64 QAM design, it appears that our X-MCMC implementation has similar cost to the best K-Best implementations. After making the changes described earlier of moving the noise variance use and adding an additional

pipelining stage we believe that our design will beat K-Best by a factor-of-two in cost.

5.5 Summary

In the preceding sections, we presented methods and approximations which allow for an efficient and high performing X-MCMC detector to be implemented. We showed that the methods used retain the desirable near maxlog-MAP performance while having minimal impact on the sampling efficiency of the algorithm. When compared to other MCMC and K-Best MIMO detectors, we found that our implementation was among the best available. Several potential improvements were identified including moving the noise variance use and additional pipelining which may reduce implementation cost and decrease latency further.

CHAPTER 6

CONCLUSION AND FUTURE WORK

6.1 Conclusion

This dissertation studied the proposed excited Markov Chain Monte Carlo (X-MCMC) MIMO detector. We showed that this improved detector resolves the two most prominent limitations in the MCMC literature, the high SNR stalling problem and decoding failure due to rare poorly converged realizations. It has near maxlog-MAP performance even at large MIMO sizes, with high-order modulation, and while using ill-conditioned channels. Our VLSI implementation of X-MCMC shows that it is one of the lowest complexity MIMO detectors available.

Through detailed analysis of the MCMC detector, we found that the cause of high SNR stalling is unusually extreme Gibbs sampler probabilities being generated. Attempts by others to mitigate this problem through hybridizing with another detector or applying temperature scaling factors have not been completely effective.

In the X-MCMC detector, we introduced a more accurate system model that includes error from both noise and the current Gibbs state. This led to a high efficiency algorithm without high SNR stalling. We then recognized an undesirable pseudo-converge behavior and suggested strategies to avoid its negative impacts, thus improving sampling efficiency further. The effects of poorly converged realizations was analyzed and a method to detect them using sample-list quality metrics was introduced. By reducing output LLR confidence with conditioning, we showed that the negative impact from poor convergence is reduced. When combined, these improvements were demonstrated to have near maxlog-MAP performance at up to 8x8 MIMO 256 QAM.

Testbed demonstrations were used to verify the high performance of the X-MCMC detector with the 802.11ac WiFi protocol over 4- and 8-stream spatial-multiplexing MIMO. To match measurement results to simulation, we presented methods to compare channels

including condition number and slicing and the new interference metric HSADR. We showed that the Gaussian i.i.d. channel models commonly used in the MCMC literature lead to potentially misleading results. Generating the correct distribution and severity of ill-conditioned channels is crucial in verifying MIMO detector performance. The most important aspects of our custom, high performing, and cost effective 8x8 MIMO testbed was described, enabling others to collect similar measurements for 1/20 the cost of some of the similar commercial systems available.

A VLSI implementation of the X-MCMC detector was presented which retains the near maxlog-MAP performance of the full floating-point version of the algorithm. This required methods to approximate the high cost aspects of the design including division in excitation, division in output conditioning, exponential probability functions, list updates, and serial calculations in excitation and pseudo-convergence escape. The final implementation has among the lowest complexities of the K-Best and MCMC algorithms in the literature.

6.2 Future Work

As presented in this dissertation, the X-MCMC detector and its VLSI implementation has largely solved the known limitations of MCMC. We have identified further open questions and opportunities that should be addressed in future works.

6.2.1 Gibbs Excitation

When testing the approximations used in Chapter 5, it appeared that both overestimates of the excitation factor and underestimates of the probability function slightly improved performance. This is likely due to approximations needed in the derivation of the excited Gibbs sampler in Chapter 3 producing a biased excitation factor. Therefore, it should be tested if adding an additional coefficient to correct this bias will improve performance. Since the VLSI design creates additional biases, the optimization study should include all relevant approximations.

6.2.2 Pseudo-Convergence

The proposed pseudo-convergence detection and 1-bit forced change escape procedure were found to be effective, but other approaches should be explored. Some initial possibilities include changing multiple bits, assuring that the bit change has a large impact on distance,

and changing the detection threshold to be shorter or longer than the number of bits.

6.2.3 Output LLR Conditioning

The output LLR conditioning can potentially be applied to any soft-output MIMO detector. It should be tested against other methods found in the literature such as K-Best.

Another form of output LLR augmentation mentioned in the sphere-decoding literature is saturation [18, 35]. LLR saturation should be tested in the context of MCMC because it may recognize types of convergence instabilities that our proposed conditioning method misses.

6.2.4 Testbed and Measurements

This dissertation contained extensive testing with the 802.11ac WiFi protocol in indoor environments. This work should be extended with LTE protocol testing to verify that the X-MCMC detector still has near maxlog-MAP performance with the more severely ill-conditioned channels expected in cellular environments.

One aspect of the 802.11ac protocol that was not tested is precoding. It will be interesting to see how much of an impact the improved, augmented channels will have on convergence times of the X-MCMC detector. This may reduce our conservative estimate of generally needing $2N_{\text{bits}} \times 2N_{\text{bits}}$ Gibbs sampler sizes to attain near maxlog-MAP performance on indoor channels.

Not mentioned in this dissertation is how difficult it was to collect a high quality data set with the MIMO testbeds. As testbed complexity increases so too does the likelihood of problems such as failed components, loose connections, and user error occurring as well as the difficulty in identifying the issue. It would be useful to develop a robust set of self tests to assist in quickly identifying potential problems.

6.2.5 VLSI Design

Two potential improvements include moving the use of noise variance and additional pipe-lining in the block-diagrams of Section 5.4.1. Currently, the noise variance σ_n^2 is applied to all of the pre-calculated and initialized values so that these divisions do not need to be made at every Gibbs steps. Unfortunately, the variability of this value leads to an increase in all bit widths throughout the design which increases complexity. We suggest to simply move

the noise variance use to the power-of-2 accurate σ_e^{-2} and σ_z^{-2} calculations, thus reducing most bitwidths.

Next, we suggest adding an additional pipe-line stage inbetween the multiplication and summation portions of the dot product performed for (5.19) as seen in Fig. 5.11. This will allow the clock speed to approximately double while only increasing the core step duration to 3 clock-cycles, leading to an overall system latency decrease of 25%.

The ideas recently presented in the multiplierless MCMC detector design by Auras [50] should be explored for use by the X-MCMC detector.

There has been recent interest in using stochastic calculations in the MCMC detector [116–118]. The benefit of stochastic math is that it can replace true multiplication and addition operations with extremely fast, imprecise alternatives. As noted in Chapter 5, we believe that stochastic calculations will be too imprecise for higher-order modulation, though it is worth exploring due to the large potential it presents.

APPENDIX

MIMO TESTBEDS

Over the last 3 years of our research and development on MCMC, we have built 3 different MIMO testbeds and a custom MIMO synchronizer as described in the following sections. All 3 testbeds proved to be effective tools, though the final Ettus B210 testbed in Section A.3 is the most effective for our specific work. There are several key considerations when designing such a system beyond the obvious specification of tunable frequency and bandwidth. These include choice of development environment and inter-SDR bandwidth requirements.

In our opinion, the most important aspect of testbed design is the choice of development environment. We have found that closed source, non-standard platforms are a massive hindrance to productivity and are generally much more expensive. Also, FPGA development should be avoided unless absolutely necessary as it can dramatically increase development time. Therefore, a good starting point for anyone deciding on developing a testbed is to limit themselves to software-defined-radios with at least partially open-source code bases. Open-source hardware is also nice to have but hard to find. Thanks to the new Analog Devices AD9361 chipset there are a myriad of new SDR products coming to the market that fit these suggestions and cost one tenth of the previous generation of commercially available SDR testbeds.

The next question to consider when designing a testbed is the inter SDR bandwidth requirement. When using SDRs, massive amounts of data can be produced trivially. For example, just 25 MHz of sampled RF data can produce 800 Mbps which nearly saturates a 1-gigabit ethernet link. If larger MIMO sizes are needed with continuous operation, then usual connectivity options are needed such as SFP+, PXI-express, or 10-gigabit ethernet. Most often this is a poor design choice because it is also unlikely that the data can be processed or stored effectively. Therefore, our preferred method is to make early design decisions that allow the use of bursted packets with long delays inbetween. This allows short,

packet bursts to be kept in local SDR FPGA buffers and slowly sent and processed between the host computer and SDR. For more information and suggestions on how to design a bursted system see Section A.3 on the 8x8 Ettus B210 based MIMO testbed.

A.1 4x4 MIMO Testbed - National Instruments FlexRIO

The testbed shown in Fig. A.1 is our first MIMO testbed and is based on the National Instruments FlexRIO platform. It is built from an assortment of standard cards which fit into a proprietary PXI express chassis. This allows a modular architecture with different combinations of cards for different purposes. The platform cost was in excess of $100,000 for 4-stream MIMO. When we needed to expand to 8-stream MIMO, this system was retired as it was too expensive to upgrade and had proven to be difficult to use in practice.

The transceiver cards used in this testbed were the NI-5792 receiver and NI-5793 transmitter. These high performance ADC and DAC cards have a 250 Msps rate and can provide up to 200 MHz of usable RF bandwidth. The tunable frequency range includes 200-4400 MHz which is sufficient to test all of the worldwide LTE cellular bands and the 2.4 GHz WiFi band. Unfortunately, the 5 GHz WiFi band is outside of its capabilities. Since each transceiver has a peak transmit power of 7 dBm, additional amplifier cards are needed, the NI-5691, which provides a maximum 20 dBm output.

Each transceiver cards must be paired with a field-programmable-gate-array (FPGA) card, in our case the 7975R with the Xilinx Kintex-7 FPGA. The FPGA controls the

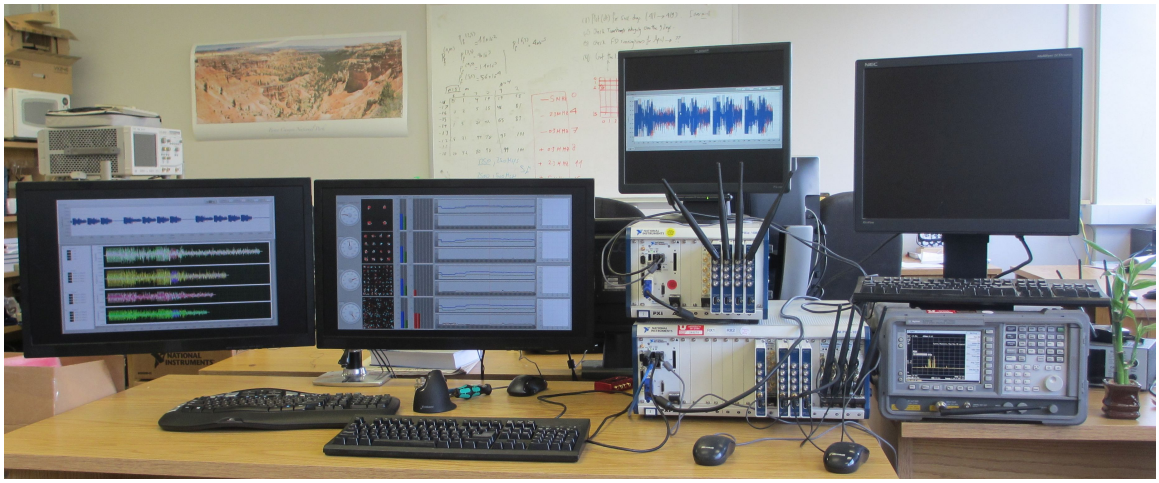


Figure A.1. 4x4 MIMO testbed based on the National Instruments FlexRIO platform.

transceiver card and does the high bandwidth intermediate processing necessary, such as re-sampling and filtering the digital data stream. It is possible to load custom software onto the FPGA, further expanding the possibilities of this modular architecture. Unfortunately, it is necessary to use Labview FPGA graphical code which is clumsy to use and required use of Labview software with many bugs. Additionally, hidden closed source blocks are added when compiling which was found to limit clock rates. It is possible to use VHDL, but only when wrapped in the proprietary graphical environment which made it difficult and time consuming to debug.

Each PXI chassis includes a 25 ppm reference clock. For our application, we wanted to have sufficient accuracy to do 802.11 WiFi testing without need of any frequency offset corrections, therefore an additional timing card was needed. The 6674T timing card includes an ovenized oscillator with 80 ppb accuracy. Once configured, the chassis distributes this improved reference clock to all cards through the backplane without additional cabling.

The chassis is controlled from an embedded computer with an Intel CPU running Microsoft Windows. We found that the low performance of the expensive, outdated, thermally limited embedded computer limited our capabilities.

The platform must be controlled from within the proprietary National Instruments Labview environment. This requires use of "G-code" which is a graphical programming method. To use this approach, closed-source functional blocks are connected with lines to create a flow graph. For basic processing this may be fine, but for the complex algorithms used in our MIMO detector research it was clumsy to use, difficult to debug, and had low performance, therefore, we used the minimum amount of G-code necessary to run the system. For all DSP algorithms, we first wrote and tested them in Matlab. Then we exported the Matlab code to C. Next, the C-code was wrapped in a compatibility shim and compiled into a Windows DLL. This DLL is imported into Labview at run-time. Although this multistep process worked, it was error prone and is not recommended

One additional difficulty of using the National Instruments system is that all G-code is kept within encrypted binaries. This means that using a version control system such as Git does not work well since any simple change requires full new copies of the large and bloated binaries. Because of this, it is standard practice to not use a version tracking system and instead make full manual copies of the entire project. After 1 year of work, our project

and backups exceeded 100 GB in size. There was also no way to identify changes between versions or revert specific portions of the project.

Overall, we would not recommend the use of National Instruments equipment or Labview software for MIMO testing. It is expensive, buggy, and difficult to use. Because National Instruments aggressively practices vendor lock-in strategies, the software and skills developed on their equipment and software are not generally usable on other equipment or for other uses.

A.2 8x8 MIMO Testbed - Ettus USRP2

The testbed shown in Fig. A.2 was a temporary 8-stream MIMO system based on the Ettus USRP2 software defined radio. It was primarily used to de-risk the transition from the earlier National Instruments based system to the later Ettus B210 testbed presented in Section A.3. One of the best aspects of the Ettus based systems is that they are completely open-source. This includes the FPGA code, interface libraries, examples, and even hardware designs with circuit diagrams. This allows for easier development, easier debugging, and the possibility to expand the system beyond what is currently supported by the manufacturer. A good example of this is that despite only 4x4 MIMO being supported by Ettus, we were able to expand our system to up to 16x16 with little difficulty.

Each USRP2 is comprised of a Xilinx Spartan-3 FPGA and a separately purchased daughter card. For our system, we used the Ettus XCVR2450 which is primarily intended for WiFi applications. It has a maximum 50 Msps rate and 36 MHz of usable bandwidth. Its two tunable frequency bands of 2400-2500 MHz and 4900-6000 MHz make it inappropriate for LTE testing, though other daughter cards could be used to provide that functionality for the USRP2. The on-board frequency reference has 20 ppm accuracy. The custom MIMO synchronizer of Section A.4 was used as an improved 5 ppb frequency reference as well as an accurate 1-pulse-per-second (PPS) source. If purchased, this system would have cost $30,000. This is roughly 1/7th of what a National Instruments based system would cost. The USRP2 and XCVR2450 have been discontinued, though they are similar to the newer Ettus N200 with CBX daughterboard.

The most difficult part of building this system was increasing the FPGA receive buffers. Unfortunately, the provided USRP2 FPGA image has an insufficient receive buffer to hold

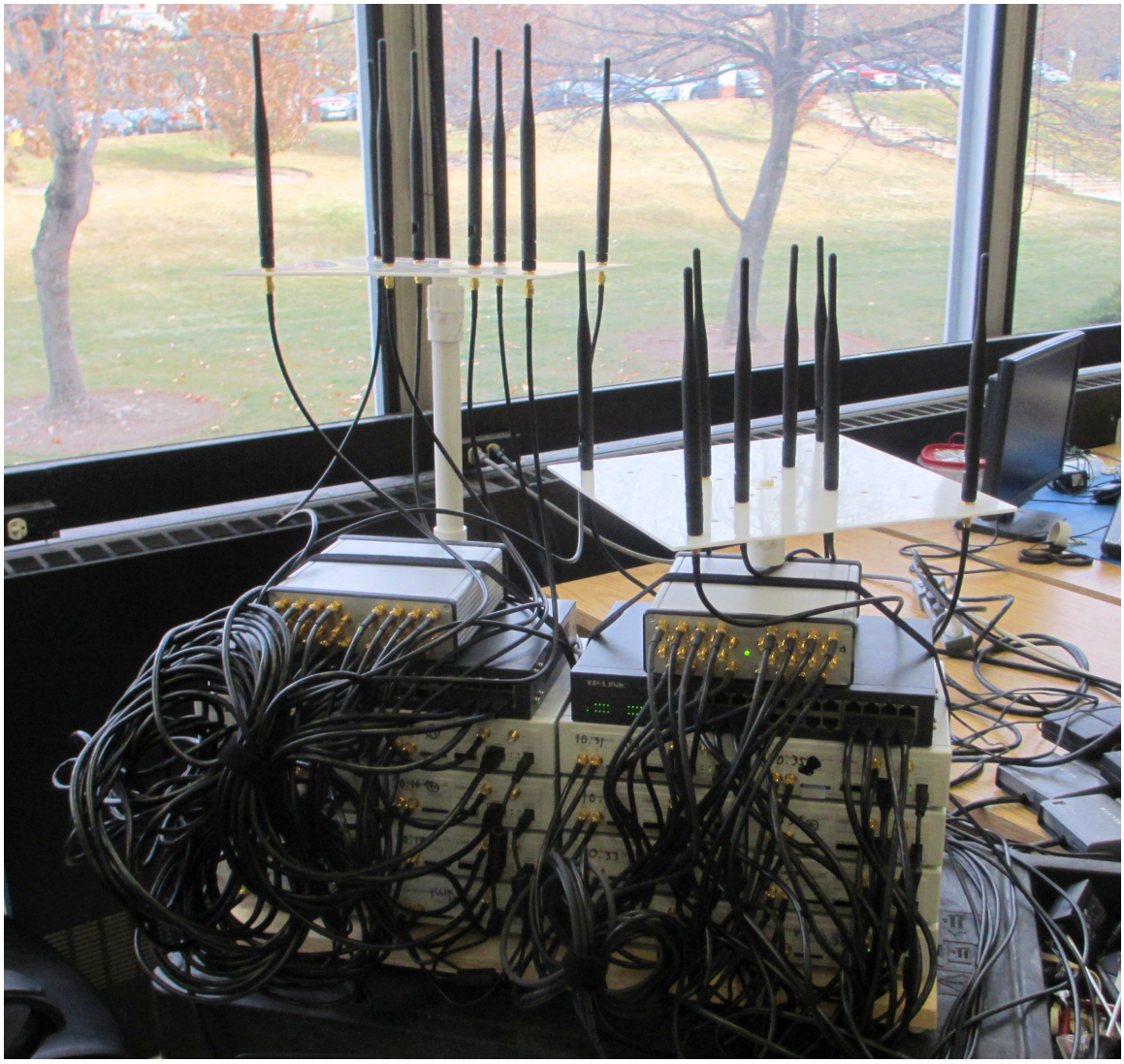


Figure A.2. 8x8 MIMO testbed based on Ettus USRP2 software defined radios and the 16x MIMO synchronizer described in Section A.4.

one of our full RF packets at 25 Msps, therefore, we created a custom FPGA image with increased buffer sizes. This was difficult to do as the FPGA design was already too large to consistently meet timing, thus, we carefully selected unneeded components of the design to remove, freeing resources without breaking the overall system. Note that this was only possible because Ettus provides their FPGA designs as open-source.

Control of the testbed is provided by a separate computer connected via 1-gigabit ethernet. Interfaces are provided through either the Gnuradio graphical programming environment or directly with a C++ API. Both Linux and Microsoft Windows is supported. Because of the

earlier bad experiences with using graphical programming environments on the National Instruments testbed, we opted to use the C++ API. Since our MIMO detector development code is based in Python, we open a socket between a C++ program which controls the radios and the Python code which produces and consumes the data. This architecture has proven to be flexible, powerful, and easy to use. One of the advantages of using Python for data creation and consumption is that is quick and easy to make high quality and informative graphs, as seen in Fig. A.3. It is even possible to manipulate and inspect the data interactively similar to Matlab.

The primary limitation of this system is the 1-gigabit ethernet bandwidth which is only capable of streaming 1 radio continuously with the needed 25 Msps, 16-bit IQ samples. Luckily, our MIMO detector testing can be performed using bursted packets. This means that, as long as a packet is kept short enough that it can completely fit within the FPGA buffers, the system can be run using timed, synchronous bursts. In theory, this can allow the system to grow to any MIMO size by simply adding more radios and more delay between bursts to provide time to transfer data slowly between host and radio FPGA.

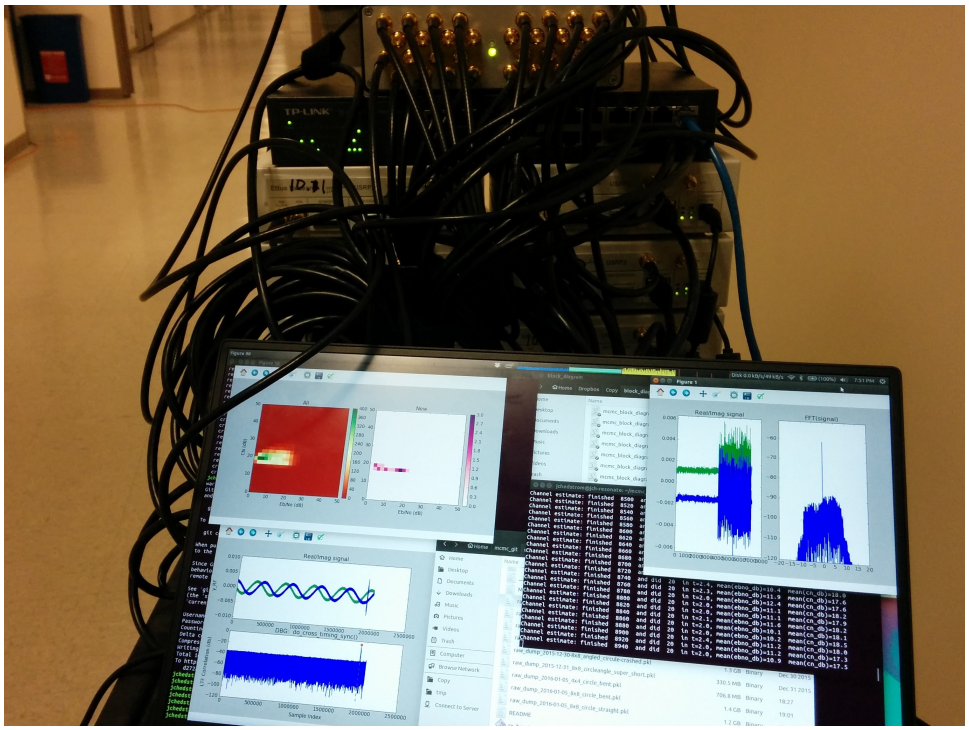


Figure A.3. Laptop showing the testbed interface using the Numpy and Matplotlib modules in Python.

There were several reasons for retiring this system. First, the radios were quite old and beginning to fail. Since the hardware was discontinued it would not be possible to replace with identical parts or get repaired. Second, the goal was to eventually expand to LTE testing which would have required the purchase of all new daughterboards. Finally, the large system complexity with 72 dangling cables made faults more likely and limited portability.

A.3 8x8 Broadband MIMO Testbed - Ettus USRP B210

The testbed shown in Fig. A.4 is our third generation MIMO testbed and is a useful contribution to the research community. Based on the Ettus B210 software defined radio and Analog Devices AD9361 chipset, it cost $10,000 US dollars to build, making it affordable for many wireless researchers. This is in contrast to most commercially available alternatives which cost $60,000 to $250,000 US dollars, usually with less frequency coverage. The main additional features that these expensive options include are increased bandwidth, improved noise figure, *official* >2-stream MIMO support, and larger capture capabilities. Here, *official* means that a company supports an intended functionality whereas *unofficial* means it is possible but without the support of the manufacturer. Our B210 solution is capable of 30.7

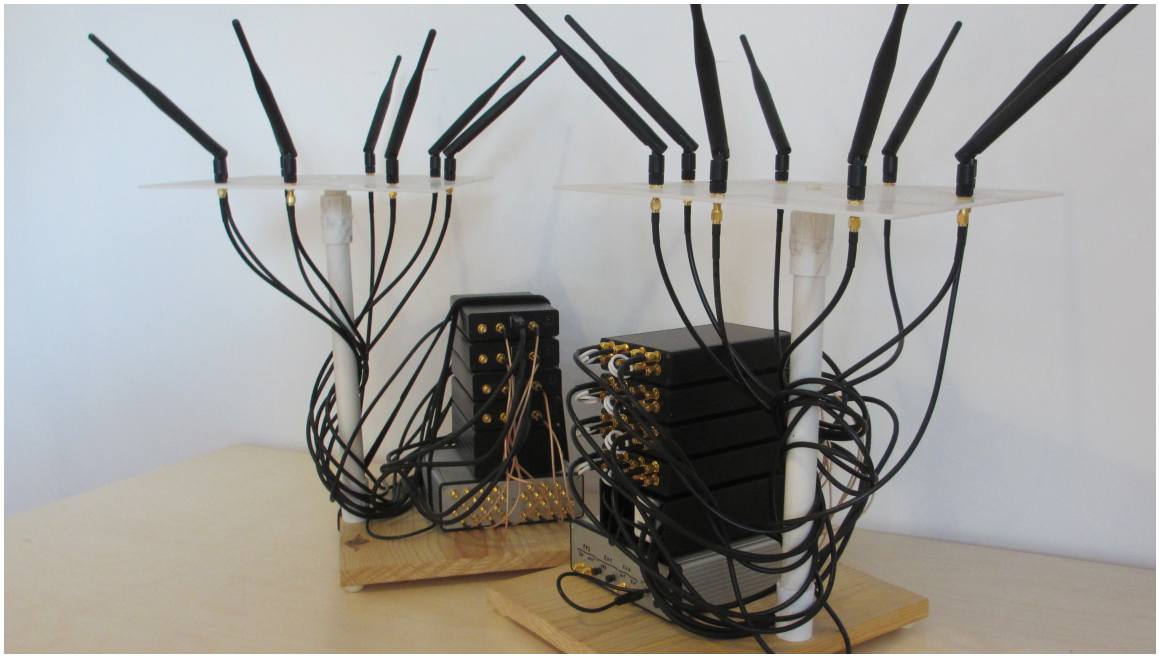


Figure A.4. 8x8 MIMO testbed based on Ettus USRP B210 software defined radios and the 16x MIMO synchronizer described in Section A.4.

MHz instantaneous bandwidth, 70MHz-6GHz frequency range, and *unofficial* 16-stream MIMO in bursts (largest number of streams verified, larger may be possible). This provides the ability to work with all of the 802.11ac and worldwide LTE bands as well as produce the 20MHz bandwidth needed for portions of the WiFi and LTE protocols.

The main hardware components of each side of the testbed are 4 Ettus B210 radios each containing two transceivers, 1 synchronized source of 4 10MHz clock and pulse-per-second (PPS) signals, a powered 4-port USB3 hub, and a linux computer. Optionally, we have added the Mini-Circuits ZX60-83LN-S+ 0.5-7GHz, 21dB gain, broadband amplifiers to the transmit side as seen in Fig. A.5. The powered USB3 hub is capable of powering both a B210 and two amplifiers off of 1 cable, reducing system cabling and complexity. A potential issue in this design is avoiding low quality USB3 chipsets which are known to create problems with the B210 radios. We have seen dropped data, re-ordered data, and overheating chips that require hard resets. Using a quality hub with good linux drivers avoids this irregular behavior.

Synchronization sources for MIMO are available from Ettus and others, but we elected to build our own with lower cost, higher performance, and a more convenient form factor, see Section A.4.

Overall, the custom software was the most time consuming and difficult portion of building the testbed. We based our design on Python and C++. This provides flexibility and a great degree of control of the hardware while still being developer friendly. As Python is being used to produce all of our simulation results, it was natural to use it to synthesize and process 802.11ac packets. The packets are sent and received through a socket to a separate C++ program controlling the radios with the Ettus USRP hardware driver (UHD) API. Using the burst mode is essential in doing large MIMO sizes as we have found that the USB3 connection is limited to 1.1 Gbps in practice, much less than the needed 6.4 Gbps for 8-antennas streaming at 25Msps and 32 bits per raw RF sample. These radios readily buffer 10k samples, allowing individual high bandwidth MIMO packets to be sent and received using timed commands despite the USB3 bottleneck. Finally, it should be noted that the Ettus UHD library does not officially support greater than 2x MIMO on the B210 as of UHD library version 3.9. Since we were using C++, it was straightforward to make a custom wrapper which synchronizes and controls multiple radios.

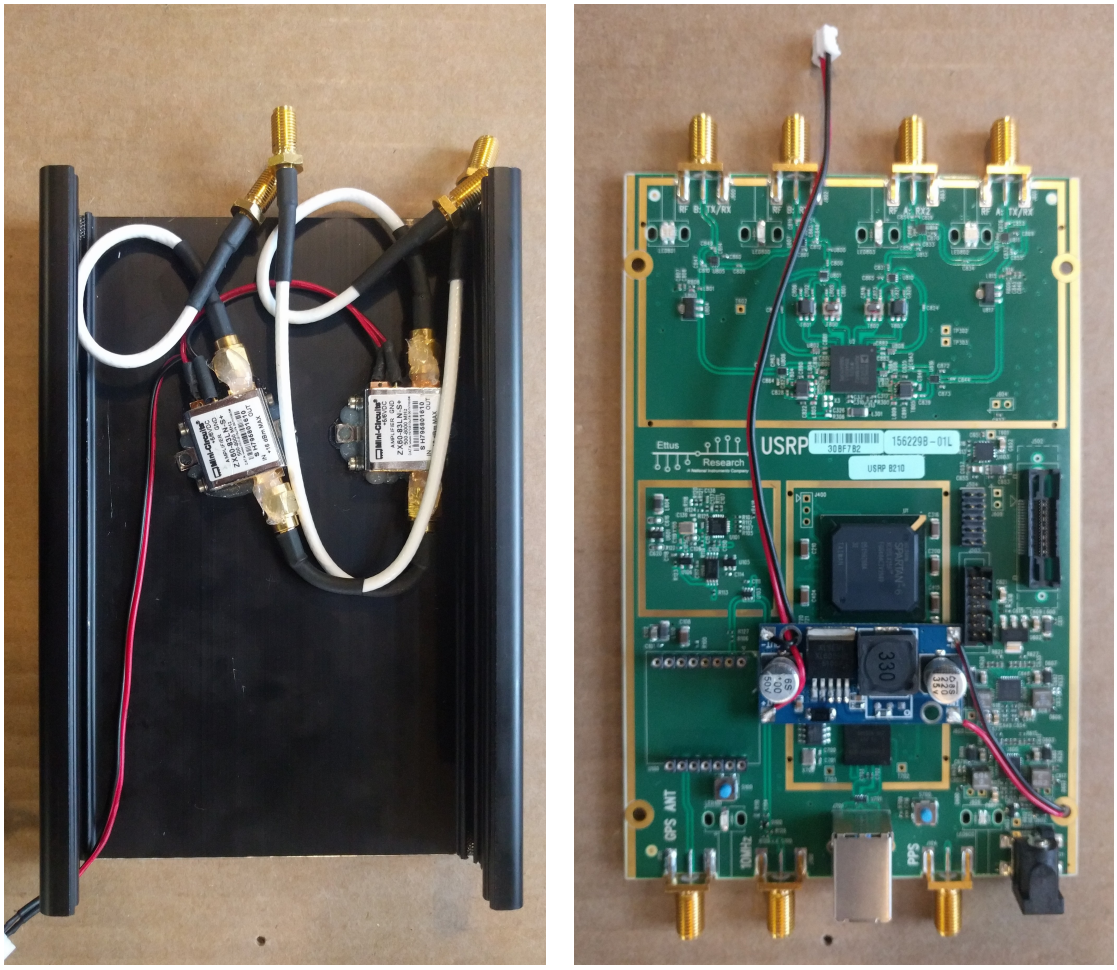


Figure A.5. Inside of the custom B210 transmitter-side case. Two Mini-Circuits ZX60-83LN-S+ amplifiers were added to optionally increase transmit power. A DC-to-DC converter is used to power the amplifiers off of the USB cable.

Several more implementation details specific to our measurements may be helpful to those designing similar systems. To align the receiver's short capture window with the sparsely transmitted packets, on startup it first captures 1 stream of 11 ms of data. By transmitting data at exactly 10 ms intervals, the long 1-stream collection can be used to identify a timing offset between the two systems and align future 8-stream MIMO collections which can only be done in short bursts. To make the data in each packet randomized but still known, a heavily coded extra field was added to the packet header containing the random seed used in creating the payload data. This allows the receiver to reconstruct the true transmitted bits for error analysis. On each burst mode transmission of the B210, there appears to be a phase instability for the first 10 microseconds of nonzero data, thus a random pad must be

placed at the beginning of each packet.

A.4 16x MIMO Synchronizer

All MIMO testbeds comprised of independent transceivers require a method of synchronization. This means that a common frequency reference and method of aligning time is necessary. The most commonly used frequency reference for SDR equipment is a 10 MHz clock. The most commonly used method to align time for SDR equipment is to use a 1-pulse-per-second (PPS) signal. This is true of the testbeds described earlier in this section based on the National Instruments FlexRIO, Ettus USRP2, and Ettus B210.

Synchronization of systems which share a clock reference and 1 PPS are done in the following manner. First, the radio tuners all derive themselves from the 10 MHz reference clock with a combination of multipliers and phase-locked-loops (PLL). This means that all of the radios will have near identical frequency though usually not identical phase since it allows for cheaper hardware. Having a random phase offset is generally not a problem for MIMO communication systems since it is corrected by the channel equalizer.

Second, the FPGAs derive their internal clocks from the reference clock. This is important because it allows counters to be used in each separate FPGA with zero drift between them. During initialization, the FPGAs are instructed to reset their counters on the next PPS. This results in the counters being identical within the error of 1 internal clock cycle. Assuming the PPS signal has a fast rise time and the internal FPGA clock is 100 MHz, 10 ns synchronization accuracy can be achieved. This is sufficient for most MIMO communications systems as this is a fraction of the channel duration and much less than the guard interval. Note that for 802.11 WiFi the short guard interval is 400 ns making the 10 ns synchronization accuracy sufficient. The cable lengths need to be only roughly matched as a 150-cm difference will only create roughly a 1-ns offset. After the internal clocks are reset, the FPGAs can operate synchronously by instructing them to all perform an action at a specific counter based time. This time must be slightly in the future to assure the instruction and any required data has been received.

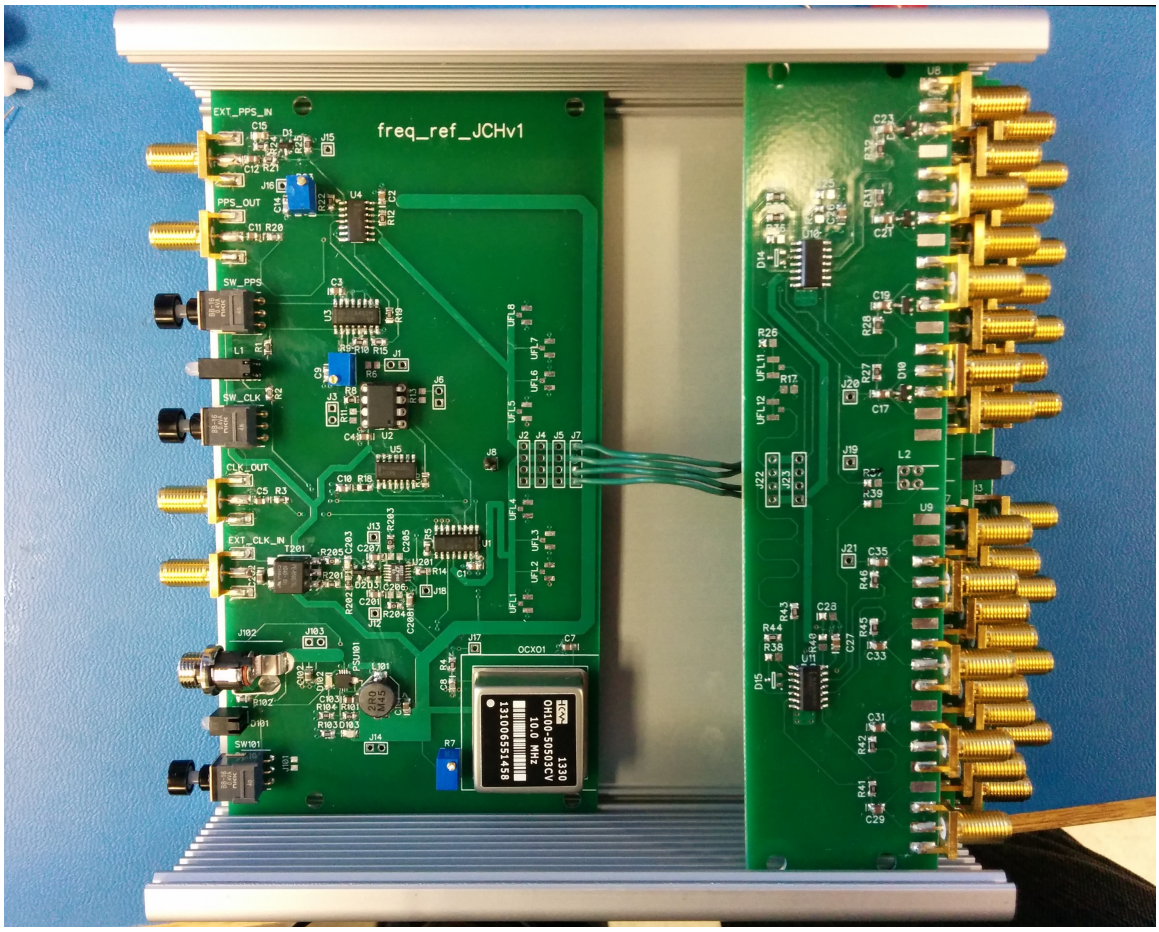
When designing the Ettus based testbeds, we were expecting to eventually expand to 16-stream MIMO, therefore, the MIMO synchronizers available for purchase were considered expensive and clumsy since they generally are limited to 8 or fewer outputs. To synchronize

16 radios with the Ettus Octoclock, 3 would be needed plus an added GPSDO. This would cost $3,750 and require 3U of 19" rack space for 25 ppb of accuracy. Instead we chose to construct our own synchronizer.

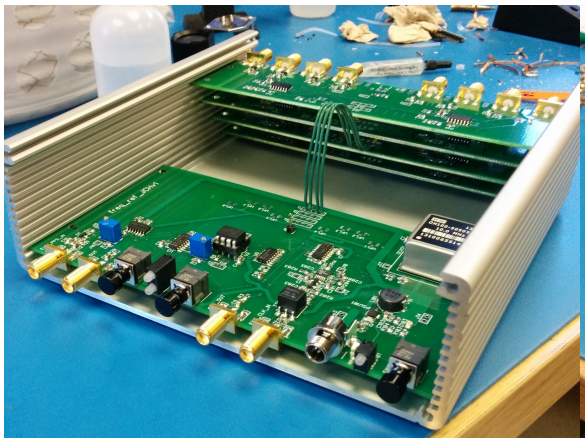
Our custom synchronizer shown in Fig. A.6 cost only $400, has 16 outputs, and 5 ppb accuracy. It also includes the option to generate a PPS from an external clock source which is not possible on the Ettus Octoclock. Its size is also much smaller and more convenient being only 2"x6.5"x6.3". Our custom synchronizer has twice as many outputs, five times the accuracy, one fifth the size, and is one tenth the cost of the commercially available Octoclock synchronizer. Note that if we redesigned a similar synchronizer for the 8x8 B210 MIMO testbed we could easily reduce both the size and cost by an additional factor-of-four. This is because the B210 has 2 streams per board, requiring only 4 synchronizer outputs, and the accuracy could be reduced to 50 ppb.

The main components of our custom synchronizer are the Connor-Winfield OH100 5 ppb 10 MHz ovenized oscillator as frequency reference, Atmel ATTiny85 microcontroller for PPS generation, and SN74AC logic inverters as buffers for both PPS and 10 MHz clock. In testing, we achieved better than 10 ns synchronization offset between radios which was at the limit of our testing equipment and a small fraction of the shortest 400 ns guard interval of 802.11ac WiFi. This is also at the limit of the synchronization abilities of the Ettus B210 radios regardless of external reference accuracy due to their 100 MHz FPGA clocks.

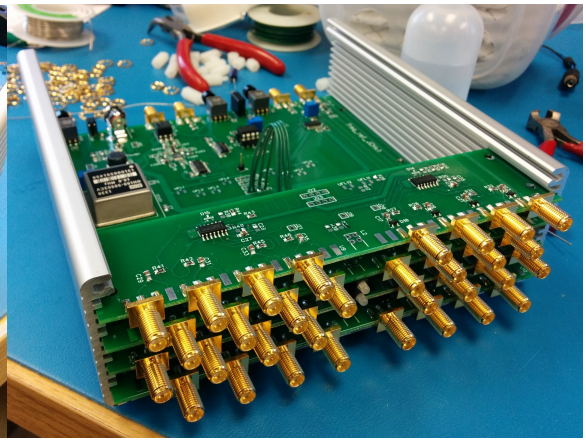
One negative of using such a highly accurate ovenized oscillator is that it has a relatively long warmup settling time when first turned on. As a result, users should not attempt to use the system until it has been on for at least 5 minutes. This is a minor point but easily forgotten, therefore, if building a new system we would select a faster settling, less accurate, smaller, and cheaper ovenized oscillator.



(a)



(b)



(c)

Figure A.6. Custom 16 output 10 MHz 5 ppb clock reference and <10 ns accurate PPS synchronizer. On the front the left half of SMAs are clock and the right half are PPS. The rear has switches which select from optional external clock and PPS sources.

REFERENCES

[1] S. Ten Brink, "Convergence behavior of iteratively decoded parallel concatenated codes," *IEEE Trans. Commun.*, vol. 49, no. 10, pp. 1727–1737, 2001.

[2] P. Cramton and P. Sujarittanonta, "Bidding and prices in the aws-3 auction," *Competitive Carrier Association*, 2015.

[3] T. Halonen, J. Romero, and J. Melero, *GSM, GPRS and EDGE performance: Evolution towards 3G/UMTS*. New York: John Wiley & Sons, 2004.

[4] A. J. Viterbi and J. K. Omura, *Principles of digital communication and coding*. Courier Corporation, 2009.

[5] B. Sklar, *Digital communications*. Upper Saddle River, New Jersey: Prentice Hall, 2001, vol. 2.

[6] S. Lin and D. J. Costello, *Error control coding*. India: Pearson Education, 2004.

[7] "802.11ac-2013 - IEEE standard for information technology," iEEE Std 802.11-2013.

[8] D. Halperin, W. Hu, A. Sheth, and D. Wetherall, "802.11 with multiple antennas for dummies," *ACM SIGCOMM Computer Communication Review*, vol. 40, no. 1, pp. 19–25, 2010.

[9] "LTE cellular specification - release 13," http://www.3gpp.org/release-13.

[10] F. Boccardi, B. Clerckx, A. Ghosh, E. Hardouin, G. Jongren, K. Kusume, E. Onggosanusi, and Y. Tang, "Multiple-antenna techniques in LTE-advanced," *IEEE Commun. Mag.*, vol. 50, no. 3, pp. 114–121, 2012.

[11] C. A. Mack, "Fifty years of Moore's law," *IEEE Trans. on Semicond. Manuf.*, vol. 24, no. 2, pp. 202–207, 2011.

[12] E. G. Larsson, O. Edfors, F. Tufvesson, and T. L. Marzetta, "Massive mimo for next generation wireless systems," *IEEE Commun. Mag.*, vol. 52, no. 2, pp. 186–195, 2014.

[13] B. Wang and K. R. Liu, "Advances in cognitive radio networks: A survey," *IEEE J. of Select. Topics in Signal Process.*, vol. 5, no. 1, pp. 5–23, 2011.

[14] H. Bolcskei, "MIMO-OFDM wireless systems: Basics, perspectives, and challenges," *IEEE Wireless Commun.*, vol. 13, no. 4, pp. 31–37, 2006.

[15] A. J. Paulraj, D. Gore, R. U. Nabar, H. Bölcskei *et al.*, "An overview of MIMO communications - a key to gigabit wireless," *Proc. IEEE*, vol. 92, no. 2, pp. 198–218, 2004.

[16] G. J. Foschini and M. J. Gans, "On limits of wireless communications in a fading environment when using multiple antennas," *Wireless Personal Commun.*, vol. 6, no. 3, pp. 311–335, 1998.

[17] E. Telatar, "Capacity of multi-antenna gaussian channels," *Eur. Trnas. on Telecommun.*, vol. 10, no. 6, pp. 585–595, 1999.

[18] Z. Guo and P. Nilsson, "Algorithm and implementation of the K-best sphere decoding for MIMO detection," *IEEE J. on Select. Areas in Commun.*, vol. 24, no. 3, pp. 491–503, 2006.

[19] B. Farhang-Boroujeny, H. Zhu, and Z. Shi, "Markov chain Monte Carlo algorithms for CDMA and MIMO communication systems," *IEEE Trans. Signal Process.*, vol. 54, no. 5, pp. 1896–1909, 2006.

[20] X. Mao, P. Amini, and B. Farhang-Boroujeny, "Markov chain Monte Carlo MIMO detection methods for high signal-to-noise ratio regimes," in *IEEE Global Telecommun. Conf. (GLOBECOM)*, 2007, pp. 3979–3983.

[21] R. Peng, K. H. Teo, J. Zhang, and R.-R. Chen, "Low-complexity hybrid QRD-MCMC MIMO detection," in *IEEE Global Telecommun. Conf. (GLOBECOM)*, 2008, pp. 1–5.

[22] S. Akoum, R. Peng, R.-R. Chen, and B. Farhang-Boroujeny, "Markov chain Monte Carlo detection methods for high SNR regimes," in *IEEE Int. Conf. Commun. (ICC)*, 2009, pp. 1–5.

[23] M. Hansen, B. Hassibi, A. G. Dimakis, and W. Xu, "Near-optimal detection in MIMO systems using gibbs sampling," in *IEEE Global Telecommun. Conf. (GLOBECOM)*, 2009, pp. 1–6.

[24] B. Hassibi, M. Hansen, A. G. Dimakis, H. A. J. Alshamary, and W. Xu, "Optimized Markov chain Monte Carlo for signal detection in MIMO systems: An analysis of the stationary distribution and mixing time," *IEEE Trans. Signal Process.*, vol. 62, no. 17, pp. 4436–4450, 2014.

[25] V. Tarokh, N. Seshadri, and A. R. Calderbank, "Space-time codes for high data rate wireless communication: Performance criterion and code construction," *IEEE Trans. Inf. Theory*, vol. 44, no. 2, pp. 744–765, 1998.

[26] S. M. Alamouti, "A simple transmit diversity technique for wireless communications," *IEEE J. on Select. Areas in Commun.*, vol. 16, no. 8, pp. 1451–1458, 1998.

[27] B. D. Van Veen and K. M. Buckley, "Beamforming: A versatile approach to spatial filtering," *IEEE ASSP Mag.*, vol. 5, no. 2, pp. 4–24, 1988.

[28] T. K. Lo, "Maximum ratio transmission," in *IEEE Int. Conf. on Commun. (ICC)*, vol. 2, 1999, pp. 1310–1314.

[29] G. J. Foschini, "Layered space-time architecture for wireless communication in a fading environment when using multi-element antennas," *Bell Labs Techn. J.*, vol. 1, no. 2, pp. 41–59, 1996.

[30] P. W. Wolniansky, G. J. Foschini, G. Golden, and R. A. Valenzuela, "V-BLAST: An architecture for realizing very high data rates over the rich-scattering wireless channel," in *URSI Int. Symp. on Signals, Syst., and Electron. (ISSSE)*. IEEE, 1998, pp. 295–300.

[31] M. Vu and A. Paulraj, "MIMO wireless linear precoding," *IEEE Signal Process. Mag.*, vol. 24, no. 5, pp. 86–105, 2007.

[32] V. Stankovic and M. Haardt, "Generalized design of multi-user MIMO precoding matrices," *IEEE Trans. on Wireless Commun.*, vol. 7, no. 3, pp. 953–961, 2008.

[33] M. Lamarca, "Linear precoding for mutual information maximization in MIMO systems," in *Proc. Int. Symp. on Wireless Commun. Systems*, 2009, pp. 26–30.

[34] I. B. Collings, M. R. Butler, and M. R. McKay, "Low complexity receiver design for MIMO bit-interleaved coded modulation," in *IEEE Int. Symp. on Spread Spectrum Techn. and Appl.*, 2004, pp. 12–16.

[35] B. M. Hochwald and S. Ten Brink, "Achieving near-capacity on a multiple-antenna channel," *IEEE Trans. Commun.*, vol. 51, no. 3, pp. 389–399, 2003.

[36] U. Fincke and M. Pohst, "Improved methods for calculating vectors of short length in a lattice, including a complexity analysis," *Math. of Computation*, vol. 44, no. 170, pp. 463–471, 1985.

[37] C.-P. Schnorr and M. Euchner, "Lattice basis reduction: Improved practical algorithms and solving subset sum problems," *Math. Programming*, vol. 66, no. 1-3, pp. 181–199, 1994.

[38] H. Vikalo, B. Hassibi, and T. Kailath, "Iterative decoding for MIMO channels via modified sphere decoding," *IEEE Trans. on Wireless Commun.*, vol. 3, no. 6, pp. 2299–2311, 2004.

[39] H. Vikalo and B. Hassibi, "On the sphere-decoding algorithm II. generalizations, second-order statistics, and applications to communications," *IEEE Trans. Signal Process.*, vol. 53, no. 8, pp. 2819–2834, 2005.

[40] J. Jaldén and B. Ottersten, "On the complexity of sphere decoding in digital communications," *IEEE Trans. Signal Process.*, vol. 53, no. 4, pp. 1474–1484, 2005.

[41] B. Hassibi and H. Vikalo, "On the sphere-decoding algorithm I. expected complexity," *IEEE Trans. Signal Process.*, vol. 53, no. 8, pp. 2806–2818, 2005.

[42] K.-w. Wong, C.-y. Tsui, R.-K. Cheng, and W.-h. Mow, "A VLSI architecture of a K-best lattice decoding algorithm for MIMO channels," in *IEEE Int. Symp. on Circuits and Syst. (ISCAS)*, vol. 3, 2002, pp. III–III.

[43] S. A. Laraway and B. Farhang-Boroujeny, "Implementation of a Markov chain Monte Carlo based multiuser/MIMO detector," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 56, no. 1, pp. 246–255, 2009.

[44] B. Farhang-Boroujeny, H. Zhu, Z. Shi, and S. A. Laraway, "Detector and method for estimating data probability in a multi-channel receiver," Nov. 25 2008, US Patent 7457367.

[45] ——, "Detector and method for estimating data probability in a multi-channel receiver," Oct. 12 2010, US Patent 7813438.

[46] B. F. Boroujeny and P. Amini, "Multi-channel communication method and apparatus using plural Markov chain Monte Carlo simulations," Dec. 7 2010, US Patent 7848440.

[47] B. Farhang-Boroujeny and S. Akoum, "Estimation of log-likelihood using constrained Markov-chain Monte Carlo simulation," Oct. 25 2011, US Patent 8045604.

[48] F.-L. Yuan, C.-H. Yang, and D. Marković, "A hardware-efficient VLSI architecture for hybrid sphere-MCMC detection," in *IEEE Global Telecommun. Conf. (GLOBECOM)*, 2011, pp. 1–6.

[49] M. Senst and G. Ascheid, "A Rao-Blackwellized Markov chain Monte Carlo algorithm for efficient MIMO detection," in *IEEE Int. Conf. Commun. (ICC)*, 2011, pp. 1–6.

[50] D. Auras, U. Deidersen, R. Leupers, and G. Ascheid, "VLSI design of a parallel MCMC-based MIMO detector with multiplier-free gibbs samplers," in *IEEE Int. Conf. on Very Large Scale Integration (VLSI-SoC)*, 2014, pp. 1–6.

[51] S. Brooks, A. Gelman, G. L. Jones, and X.-L. Meng, *Handbook of Markov Chain Monte Carlo*. United Kingdom: Chapman and Hall/CRC, 2011.

[52] P. Robertson, P. Hoeher, and E. Villebrun, "Optimal and sub-optimal maximum a posteriori algorithms suitable for turbo decoding," *Eur. Trans. on Telecommun.*, vol. 8, no. 2, pp. 119–125, 1997.

[53] C. Berrou, A. Glavieux, and P. Thitimajshima, "Near shannon limit error-correcting coding and decoding: Turbo-codes. 1," in *IEEE Int. Conf. Commun.*, vol. 2, 1993, pp. 1064–1070.

[54] J. C. Hedstrom, C. H. Yuen, and B. Farhang-Boroujeny, "Markov chain Monte Carlo based multiuser/MIMO detector: 802.11ac implementation and measurement," in *IEEE Int. Conf. Commun. (ICC)*, 2015, pp. 4846–4852.

[55] J. C. Hedstrom, C. H. Yuen, R.-R. Chen, and B. Farhang-Boroujeny, "Excited Markov chain Monte Carlo MIMO detector with 8-antenna 802.11ac testbed demonstration," in *IEEE Int. Conf. Commun. (ICC)*, 2017.

[56] ——, "Achieving near MAP performance with an excited Markov chain Monte Carlo MIMO detector," *Under preparation*, 2017.

[57] J. Hedstrom, C. H. Yuen, and B. Farhang-Boroujeny, "Markov chain monte carlo MIMO detector method with gibbs sampler excitation," Patent Pending 2015, US Patent 62/386762.

[58] ——, "Soft-information moderation for MIMO detectors," Patent Pending 2015, US Patent 62/386764.

[59] R.-R. Chen, R. Peng, A. Ashikhmin, and B. Farhang-Boroujeny, "Approaching MIMO capacity using bitwise Markov chain Monte Carlo detection," *IEEE Trans. Commun.*, vol. 58, no. 2, pp. 423–428, 2010.

[60] R.-H. Peng, R.-R. Chen, and B. Farhang-Boroujeny, "Markov chain Monte Carlo detectors for channels with intersymbol interference," *IEEE Trans. Signal Process.*, vol. 58, no. 4, pp. 2206–2217, 2010.

[61] V. Erceg, L. Schumacher, P. Kyritsi, A. Molisch, D. Baum, A. Gorokhov, C. Oestges, Q. Li, K. Yu, N. Tal *et al.*, "IEEE P802.11 wireless LANs - indoor MIMO WLAN TGn channel models," 2004, iEEE 802.11 document 03/940r4.

[62] C. J. Geyer, "Practical Markov chain Monte Carlo," *Statistical science*, pp. 473–483, 1992.

[63] A. F. Smith and G. O. Roberts, "Bayesian computation via the gibbs sampler and related Markov chain Monte Carlo methods," *J. of the Roy. Statist. Soc.: Series B Statist. Methodology*, pp. 3–23, 1993.

[64] N. Metropolis, A. W. Rosenbluth, M. N. Rosenbluth, A. H. Teller, and E. Teller, "Equation of state calculations by fast computing machines," *J. of Chemical Physics*, vol. 21, no. 6, pp. 1087–1092, 1953.

[65] W. K. Hastings, "Monte Carlo sampling methods using Markov chains and their applications," *Biometrika*, vol. 57, no. 1, pp. 97–109, 1970.

[66] G. O. Roberts and S. K. Sahu, "Updating schemes, correlation structure, blocking and parameterization for the gibbs sampler," *J. of the Roy. Statist. Soc.: Series B Statist. Methodology*, vol. 59, no. 2, pp. 291–317, 1997.

[67] J. Hagenauer, E. Offer, and L. Papke, "Iterative decoding of binary block and convolutional codes," *IEEE Trans. Inf. Theory*, vol. 42, no. 2, pp. 429–445, 1996.

[68] P. Robertson, E. Villebrun, and P. Hoeher, "A comparison of optimal and sub-optimal MAP decoding algorithms operating in the log domain," in *IEEE Int. Conf. Commun. (ICC)*, vol. 2, 1995, pp. 1009–1013.

[69] H. Zhu, Z. Shi, and B. Farhang-Boroujeny, "MIMO detection using Markov chain Monte Carlo techniques for near-capacity performance," in *IEEE Int. Conf. on Acoust., Speech, and Signal Process (ICASSP)*, vol. 3, 2005.

[70] A. Kumar, S. Chandrasekaran, A. Chockalingam, and B. S. Rajan, "Near-optimal large-MIMO detection using randomized MCMC and randomized search algorithms," in *IEEE Int. Conf. Commun. (ICC)*, 2011, pp. 1–5.

[71] Z. Shi, H. Zhu, and B. Farhang-Boroujeny, "Markov chain Monte Carlo techniques in iterative detectors: A novel approach based on Monte Carlo integration," in *IEEE Global Telecommun. Conf. (GLOBECOM)*, vol. 1, 2004, pp. 325–329.

[72] B. Liu, J. Zhang, K.-K. Wong, L. He, and C. Hao, "Low-latency near-capacity MIMO detection using parallel and hybrid QRD-MCMC algorithm," in *IEEE China Summit and Int. Conf. on Signal and Inform. Process. (ChinaSIP)*, 2015, pp. 903–907.

[73] M. Senst, G. Ascheid, and H. Lüders, "Performance evaluation of the Markov chain Monte Carlo MIMO detector based on mutual information," in *IEEE Int. Conf. Commun. (ICC)*, 2010, pp. 1–6.

[74] M. Zhao, "Iterative receiver techniques for data-driven channel estimation and interference mitigation in wireless communications," Ph.D. dissertation, College of Computer Science and Engineering, Research School of Information Sciences and Engineering, National University of Singapore, 6 2009.

[75] M. Zhao, Z. Shi, and M. C. Reed, "A reduced-state-space Markov chain Monte Carlo method for iterative spatial multiplexing MIMO," in *IEEE IEEE Global Telecommun. Conf. Workshops*, 2009, pp. 1–6.

[76] T. Datta, N. A. Kumar, A. Chockalingam, and B. S. Rajan, "A novel MCMC algorithm for near-optimal detection in large-scale uplink mulituser MIMO systems," in *IEEE Inform. Theory and Appl. Workshop (ITA)*, 2012, pp. 69–77.

[77] R.-R. Chen, R. Peng, and B. Farhang-Boroujeny, "Markov chain Monte Carlo: Applications to MIMO detection and channel equalization," in *Inform. Theory and Appl. Workshop*. IEEE, 2009, pp. 44–49.

[78] C. Douillard, M. Jézéquel, C. Berrou, D. Electronique, A. Picart, P. Didier, and A. Glavieux, "Iterative correction of intersymbol interference: Turbo-equalization," *Eur. Trans. on Telecommun.*, vol. 6, no. 5, pp. 507–511, 1995.

[79] A. Elkhazin, K. N. Plataniotis, and S. Pasupathy, "Reduced-dimension map turbo-BLAST detection," *IEEE Trans. Commun.*, vol. 54, no. 1, pp. 108–118, 2006.

[80] G. Colavolpe, G. Ferrari, and R. Raheli, "Extrinsic information in iterative decoding: A unified view," *IEEE Trans. Commun.*, vol. 49, no. 12, pp. 2088–2094, 2001.

[81] G. Colavolpe, D. Fertonani, and A. Piemontese, "SISO detection over linear channels with linear complexity in the number of interferers," *IEEE J. of Select. Topics in Signal Process.*, vol. 5, no. 8, pp. 1475–1485, 2011.

[82] J. Vogt and A. Finger, "Improving the max-log-MAP turbo decoder," *Electron. Lett.*, vol. 36, no. 23, pp. 1937–1939, 2000.

[83] R.-R. Chen, B. Farhang-Boroujeny, and A. Ashikhmin, "Capacity-approaching LDPC codes based on Markov chain Monte Carlo MIMO detection," in *IEEE Workshop on Signal Process. Advances in Wireless Commun.*, 2005, pp. 285–288.

[84] J. Hagenauer, "The exit chart - introduction to extrinsic information transfer in iterative processing," in *Proc. Eur. Signal Process. Conf. (EUSIPCO)*. Citeseer, 2004, pp. 1541–1548.

[85] S. Ten Brink, G. Kramer, and A. Ashikhmin, "Design of low-density parity-check codes for modulation and detection," *IEEE Trans. Commun.*, vol. 52, no. 4, pp. 670–678, 2004.

[86] J.-P. Kermoal, L. Schumacher, K. I. Pedersen, P. E. Mogensen, and F. Frederiksen, "A stochastic MIMO radio channel model with experimental validation," *IEEE J. on Select. Areas in Commun.*, vol. 20, no. 6, pp. 1211–1226, 2002.

[87] A. F. Molisch, *Wireless Communications*, 2nd ed. New York: John Wiley & Sons, 2011.

[88] C.-N. Chuah, D. N. C. Tse, J. M. Kahn, and R. A. Valenzuela, "Capacity scaling in MIMO wireless systems under correlated fading," *IEEE Trans. Inf. Theory*, vol. 48, no. 3, pp. 637–650, 2002.

[89] S. L. Loyka, "Channel capacity of MIMO architecture using the exponential correlation matrix," *IEEE Commun. Lett.*, vol. 5, no. 9, pp. 369–371, 2001.

[90] "802.11n-2009 - IEEE standard for information technology," iEEE Std 802.11-2009.

[91] A. A. Saleh and R. Valenzuela, "A statistical model for indoor multipath propagation," *IEEE J. on Select. Areas in Commun.*, vol. 5, no. 2, pp. 128–137, 1987.

[92] Q. H. Spencer, B. D. Jeffs, M. A. Jensen, and A. L. Swindlehurst, "Modeling the statistical time and angle of arrival characteristics of an indoor multipath channel," *IEEE J. on Select. Areas in Commun.*, vol. 18, no. 3, pp. 347–360, 2000.

[93] R.-M. Cramer, R. A. Scholtz, and M. Z. Win, "Evaluation of an ultra-wide-band propagation channel," *IEEE Trans. Antennas Propag.*, vol. 50, no. 5, pp. 561–570, 2002.

[94] A. S. Poon and M. Ho, "Indoor multiple-antenna channel characterization from 2 to 8 GHz," in *IEEE Int. Conf. Commun. (ICC)*, vol. 5, 2003, pp. 3519–3523.

[95] G. German, Q. Spencer, L. Swindlehurst, and R. Valenzuela, "Wireless indoor channel modeling: Statistical agreement of ray tracing simulations and channel sounding measurements," in *IEEE Int. Conf. on Acoust., Speech, and Signal Process (ICASSP)*, vol. 4, 2001, pp. 2501–2504.

[96] J.-G. Wang, A. S. Mohan, and T. A. Aubrey, "Angles-of-arrival of multipath signals in indoor environments," in *IEEE Veh. Technol. Conf. (VTC)*, vol. 1, 1996, pp. 155–159.

[97] C.-C. Chong, D. I. Laurenson, and S. McLaughlin, "Statistical characterization of the 5.2 GHz wideband directional indoor propagation channels with clustering and correlation properties," in *IEEE Veh. Technol. Conf. (VTC)*, vol. 1, 2002, pp. 629–633.

[98] J. P. Kermoal, L. Schumacher, P. E. Mogensen, and K. I. Pedersen, "Experimental investigation of correlation properties of MIMO radio channels for indoor picocell scenarios," in *IEEE Veh. Technol. Conf. (VTC)*, vol. 1, 2000, pp. 14–21.

[99] L. Schumacher, K. I. Pedersen, and P. E. Mogensen, "From antenna spacings to theoretical capacities-guidelines for simulating MIMO systems," in *IEEE Int. Symp. on Personal, Indoor and Mobile Radio Commun.*, vol. 2, 2002, pp. 587–592.

[100] P. Soma, D. S. Baum, V. Erceg, R. Krishnamoorthy, and A. Paulraj, "Analysis and modeling of multiple-input multiple-output (MIMO) radio channel based on outdoor measurements conducted at 2.5 GHz for fixed BWA applications," in *IEEE Int. Conf. Commun. (ICC)*, vol. 1, 2002, pp. 272–276.

[101] H. Artés, D. Seethaler, and F. Hlawatsch, "Efficient detection algorithms for MIMO channels: A geometrical approach to approximate ML detection," *IEEE Trans. Signal Process.*, vol. 51, no. 11, pp. 2808–2820, 2003.

[102] A. K. Cline, C. B. Moler, G. W. Stewart, and J. H. Wilkinson, "An estimate for the condition number of a matrix," *SIAM J. on Numerical Anal.*, vol. 16, no. 2, pp. 368–375, 1979.

[103] A. Edelman, "Eigenvalues and condition numbers of random matrices," *SIAM J. on Matrix Anal. and Appl.*, vol. 9, no. 4, pp. 543–560, 1988.

[104] S. Roger, A. Gonzalez, V. Almenar, and A. Vidal, "Combined K-best sphere decoder based on the channel matrix condition number," in *IEEE Int. Symp. on Commun., Control and Signal Process. (ISCCSP)*, 2008, pp. 1058–1061.

[105] J. Maurer, G. Matz, and D. Seethaler, "Low-complexity and full-diversity MIMO detection based on condition number thresholding," in *IEEE Int. Conf. on Acoust., Speech, and Signal Process (ICASSP)*, vol. 3, 2007, pp. III–61.

[106] M. D. McKinley, K. A. Remley, M. Myslinski, J. S. Kenney, D. Schreurs, and B. Nauwelaers, "EVM calculation for broadband modulated signals," in *ARFTG Conf. Dig.*, 2004, pp. 45–52.

[107] R. A. Shafik, M. S. Rahman, A. R. Islam, and N. S. Ashraf, "On the error vector magnitude as a performance metric and comparative analysis," in *IEEE Int. Conf. on Emerging Technol.*, 2006, pp. 27–31.

[108] P. S. Bullen, *Handbook of Means and Their Inequalities*. Berlin, Germany: Springer Science & Business Media, 2013, vol. 560, ch. The Arithmetic, Geometric, and Harmonic Means.

[109] D. Fowler and J. E. Smith, "An accurate, high speed implementation of division by reciprocal approximation," in *IEEE Proc. Symp. on Comput. Arithmetic*, 1989, pp. 60–67.

[110] J. E. Stine, "Oklahoma State University (OSU) cell library," 2007, http://www.vlsitechnology.org/html/libraries04.html.

[111] U. Deidersen, D. Auras, and G. Ascheid, "A parallel VLSI architecture for Markov chain Monte Carlo based MIMO detection," in *Proc. of the ACM Int. Conf. on Great Lakes Symp. on VLSI*. ACM, 2013, pp. 167–172.

[112] M. Wenk, M. Zellweger, A. Burg, N. Felber, and W. Fichtner, "K-best MIMO detection VLSI architectures achieving up to 424 Mbps," in *IEEE Int. Symp. on Circuits and Syst. (ISCAS)*, 2006, pp. 4–pp.

[113] S. Chen, T. Zhang, and Y. Xin, "Relaxed K-best MIMO signal detector design and VLSI implementation," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 15, no. 3, pp. 328–337, 2007.

[114] H.-L. Lin, R. C. Chang, and H.-L. Chen, "A high-speed sdm-mimo decoder using efficient candidate searching for wireless communication," *IEEE Trans. Circuits Syst. II, Exp. Briefs*, vol. 55, no. 3, pp. 289–293, 2008.

[115] M. Shabany and P. G. Gulak, "A 675 Mbps, 4×4 64-QAM K-best MIMO detector in 0.13μm CMOS," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 20, no. 1, pp. 135–147, 2012.

[116] K. Amiri, P. Radosavljevic, and J. R. Cavallaro, "Architecture and algorithm for a stochastic soft-output mimo detector," in *Asilomar Conf. on Signals, Syst. and Comput. (ACSSC)*. IEEE, 2007, pp. 1034–1038.

[117] J. Chen, J. Hu, and G. E. Sobelman, "Stochastic MIMO detector based on the Markov chain Monte Carlo algorithm," *IEEE Trans. Signal Processing*, vol. 62, no. 6, pp. 1454–1463, 2014.

[118] ——, "Stochastic iterative MIMO detection system: Algorithm and hardware design," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 62, no. 4, pp. 1205–1214, 2015.

[119] A. Alaghi and J. P. Hayes, "Survey of stochastic computing," *ACM Trans. on Embedded Computing Syst. (TECS)*, vol. 12, no. 2s, p. 92, 2013.

[120] A. Burg, M. Borgmann, M. Wenk, M. Zellweger, W. Fichtner, and H. Bolcskei, "VLSI implementation of MIMO detection using the sphere decoding algorithm," *IEEE J. of Solid-State Circuitss*, vol. 40, no. 7, pp. 1566–1577, 2005.

[121] S. Chen and T. Zhang, "Low power soft-output signal detector design for wireless mimo communication systems," in *ACM/IEEE Int. Symp. on Low Power Electron. and Design (ISLPED)*, 2007, pp. 232–237.