University of Alberta

Reduced-Complexity User and Data-Stream Scheduling for Multiuser MIMO Downlink

by

Marcin Marek Misiewicz

A thesis submitted to the Faculty of Graduate Studies and Research
in partial fulfillment of the requirements for the degree of

Master of Science
in
Communications

Electrical and Computer Engineering

Fall 2013
Edmonton, Alberta

Abstract

We consider the scheduling problem in multiuser multiple-input multiple-output (MIMO) systems with successive zero-forcing precoding and coordinated transmit-receive processing. Focusing specifically on the problem of user and data stream scheduling, we allow variable number of data streams per user. Optimal scheduling employing exhaustive search for a constant number of streams per user is already combinatorially complex, and is further compounded by allowing for a variable number of streams per user. We consider a suboptimal metaheuristic scheduling algorithm based on simulated annealing, an algorithm which has shown to provide quick and relatively good solutions to combinatorial optimization problems. We introduce various parameters and examine their impact on the performance of the algorithm. Simulated annealing is shown to offer comparable results at a fraction of the computational cost of the exhaustive search. Our contributions to this area are emphasized in the conclusion and future areas of work are discussed.

Acknowledgements

I would like to thank my supervisor, Dr. Krzymień for his mentorship and financial support, without which I would not have had the opportunity to complete my studies.

This work was made possible by the generous financial support of Dr. Witold Krzymień, TRTech (formerly known as Telecommunications Research Laboratories (TRLabs)), Telus Communications, and MITACS. Invaluable technical support from Kamil Marcinkowski, Masao Fujinaga, Walter Lysz, Roman Baranowski at Westgrid, and computing resources provided by WestGrid, Compute/Calcul Canada, and lastly the help of the Research Support Group in Academic Information & Communications Technologies (AICT) at the University of Alberta are also highly appreciated.

I would also like to acknowledge the extraordinary backing from my parents Bogumila and Mark Misiewicz, and grandparents Jadwiga Buko, and Leonard and Bożena Misiewicz, who provided me with constant support, understanding, encouragement, and fostered my inquisitive nature throughout my childhood. Without their unwavering support the completion of my schooling would not have been possible. The last part of the work for this thesis would not have been completed without the help of Ola Żebrowska who provided me with unmeasurable amounts of support.

Last but not least, I need to emphasize the invaluable support of Dr. Robert Elliott and Dr. Kevin Jacobson. They generously offered precious technical advice throughout my program and tirelessly entertained my never ending stream of questions during our long winter afternoons and evenings at the lab.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Introduction and Motivation

Ever since the development and deployment of the first cellular systems by NTT Docomo in Japan in the late 70's, and subsequently the Advanced Mobile Phone System (AMPS) in North America in the early 80's, the mobile phone has cemented itself as a ubiquitous part of our lives. While the original analog systems were originally created to handle primarily voice traffic, the shift to digital networks and the introduction of packet data in 2G and 2.5G systems spurred the beginning of an increase in demand for wireless data transfer capabilities. Specifically, the advent of 3G services and smartphones in the late 2000's heralded the shift from networks carrying primarily voice traffic, to networks carrying increasing amounts of data. Usage of mobile phones shifted from just making phone calls and sending short messages to checking emails, watching videos, sharing pictures, and using media-rich social media platforms. The decline in popularity of feature-phones and the proliferation of smartphones such as Android devices, Apple's iPhone, and Research in Motion's Blackberry devices in the subsequent years helped data usage surge, and finally surpassed voice traffic in late 2009 (Fig 1.1) .

Faced with the steadily increasing demand for bandwidth intensive mobile applications on limited spectral resources of the existing cellular systems, new cellular standards are designed for high data rates and greater spectral efficiencies. Specifically, due to downlink-heavy load of mobile data traffic, in which the majority of the data is sent to the users from a source on the forward, or downlink channel, much of the focus in research has been on

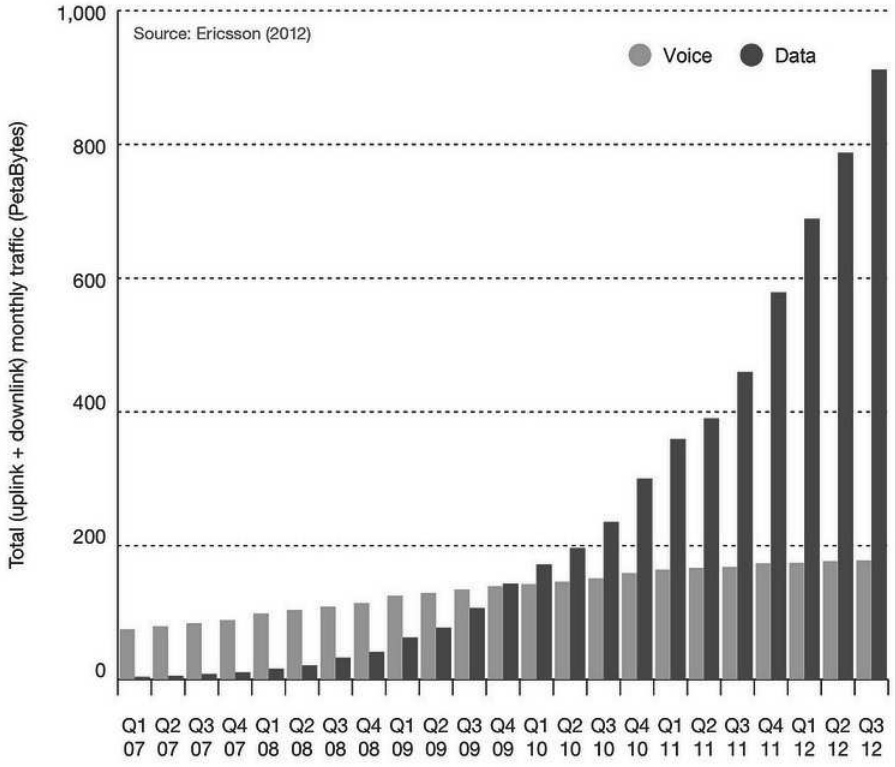

Figure 1.1: Mobile data monthly traffic [1]

increasing the spectral efficiencies and capacities on the downlink of cellular systems.

With the onset of voice-data convergence and to fulfill demands for increased throughput, the evolution and development of modern beyond 3G, 4G, and beyond 4G cellular systems has required a fundamental shift from first generation cellular designs: including a shift from circuit switched networks in 1G, to hybrid circuit-packet switched designs in 2-3.5G, to pure packet-switched networks in 4G, and the inclusion of multiple antennas at both transmit and receive terminals in 4G.

Unlike legacy circuit-switched 1G, 2G, and 3G systems which allocated dedicated links for voice traffic for the entire duration of a call, true packet-switched voice was introduced in 4G and has subsequently become a fundamental feature of modern wireless telecommunication systems. Packet-switching systems are characterized by the fact that they split the data to be transmitted into packets which are sent over a shared network link with no dedicated paths.

1.2 MIMO Wireless Communications

The concept of exploiting multiple antennas in wireless systems can be attributed to the pioneering work done by researchers in the 80's [2, 3] on multiple antenna systems that set the stage for the development of MIMO systems. Later in the 90's, research into the capabilities of single-user MIMO channels [4, 5] showed that performance far outpacing single antenna links is possible especially in highly dispersive channels with independent fading. Further investigation by Telatar [6] showed that given a wireless system with n_T antennas at the transmitter, and n_R antennas at the receiver, the maximum capacity possible increases with n_T and n_R, and was proportional to $\min(n_R, n_T)$ given a rich, independent scattering environment. In general, multi-antenna systems provide us with a capacity that scales linearly with $\min(n_T, n_R)$ and with extra spatial degrees of freedom, which can be leveraged to increase data rate by multiplexing or increase the reliability of the transmission through diversity [5, 7].

In a MIMO spatial multiplexing scheme, several independently encoded data streams or symbols are transmitted from each of the multiple transmit antennas, while in the diversity exploiting scheme redundancy is introduced by transmitting several copies of the same data. Though both diversity and multiplexing can be leveraged to better exploit wireless transmission, [8] showed that a diversity multiplexing trade-off exists and dictates an upper limit on how much improvement any scheme can realistically achieve.

Wireless transmission using multiple-input multiple-output (MIMO) techniques are capable of higher capacity and throughput without the need for additional power and bandwidth[9], and therefore they are perfect candidates for the next generation of wireless transmission standards. The latest 4G and future (beyond 4G) systems, are characterized by multiple antennas at both the receiver and transmitter [10] that allow them to exploit the extra spatial resources of MIMO configurations.

1.3 Multiuser MIMO Wireless

Consider a generalization of the single-user downlink MIMO vector channel, the downlink multi-user MIMO (MU-MIMO) vector broadcast channel formed by considering K users simultaneously. MU-MIMO is an extension of single user MIMO with the added restriction

that while coordination can occur amongst the individual antennas of a given user, there is no cooperation in decoding between the respective users. Multiuser diversity is a form selection diversity amongst users (and not specific to MIMO, or MU-MIMO) where the base station schedules transmission to those users with favourable channel conditions at transmission time to improve the overall system performance, and it is a resource that can be exploited by MU-MIMO systems. On the multiuser broadcast channel on the downlink between the base station and mobile users, different data streams are now simultaneously transmitted to several users. This causes multiuser interference (MUI) and necessitates the application of interference mitigation techniques via precoding at the base station if the gains of the MU-MIMO are to be exploited.

In most cases precoding can be thought of as generalized beamforming (or taking advantage of interference by selective weighting of signals to change the directionality of the array) to support multi-layer transmissions in MIMO systems. While precoding is not strictly necessary in the single-user case (meaning the receiver can be tasked with interference mitigation), adapting the transmitted signal at the transmitter via precoding with the aforementioned CSIT is absolutely necessary for MU-MIMO in order to exploit multiuser diversity and other MU-MIMO gains [11]. This is because the user terminals cannot cooperate in the decoding of their received signals, thus the transmitter *must* precode each users data streams as to enable each user to spatially separate their data from the aggregate transmitted signal.

MU-MIMO provides many advantages, including multi-user diversity, user and stream multiplexing, decorrelation of spatial signatures, and the ability to take advantage of low-rank channels. However, these additional diversity and multiplexing gains come at a cost of requiring channel state information at the transmitter (CSIT)[11] and the necessity of employing interference mitigation or cancelation techniques. It is important to note that in contrast to single-user point-to-point MIMO systems in which exploiting some MIMO gains without CSIT is possible (albeit with reduced performance), the absence of accurate and timely CSIT in the MU-MIMO case not allow the system to extract MU-MIMO diversity and multiplexing gains [12, 13].

1.4 Multiuser MIMO Scheduling

As described in the previous section, spatial stream (or layer) separation via precoding is necessary in order to secure MU-MIMO gains, however the implementation of precoding (transmitter side) will usually place constraints on the maximum number of users that can be served simultaneously. In modern cellular deployments it is also common to have data queued for transmission at the transmitter to a large pool of users at any given instant, as well as finite bandwidth and power available for transmission. This resource-limited environment coupled with the dimensionality constraints imposed by the implementation of precoding techniques gives rise to an upper bound to the number of users that can be served simultaneously (usually related to the number of transmit antennas at the base station and the transmit power). These constraints give rise to the need of some sort of resource distribution algorithms, or scheduling, to effectively divide the limited resources among the group of users requesting service to maximize the utility of the system based on a relevant performance metric (such as maximizing sum-rate or minimizing delay times).

During this distribution of these limited resources, fairness of the scheduler can also be taken into account. For example, if the system schedules the resources in such a way as to maximize the total throughput of our system, we will achieve the best rates. However, in this case users with chronically bad channel conditions (at the cell edge for instance) face the possibility of being subject to long delays and low throughput or being denied service outright. To prevent the starvation of these users and to maintain a minimum level of service for all users, schedulers can be designed with Quality of Service (QoS) metrics defining minimum rate or maximal delay times to ensure acceptable service is provided for all users. In addition, the selection of users with spatially uncorrelated channels in the same transmission interval can help to mitigate the necessity of wasting power on the suppression of MUI.

The design of MU-MIMO systems and scheduling algorithms is inherently complex, as we must consider many factors and are often forced to make compromises between complexity and performance. These problems of user scheduling and many other difficulties must be dealt with in the design of MU-MIMO systems, including: multiple, (as opposed to single), per user or per antenna power constraints, and the limited or non-existent cooperation between multiple users in the decoding process.

1.5 Thesis Objectives and Summary of Contributions

The principal objective of this thesis is design of new reduced-complexity scheduling algorithms for MU-MIMO in systems employing successive zero-forcing precoding and investigation of their performance. While the problem of scheduling for cases in which a full allocation of n_R spatial data streams (implying that each user has n_R receive antennas) per user (and up to n_T data streams for all users) has been examined in the past, relatively little has been done in exploring the added diversity created by allowing also fewer than n_R data streams per user. The addition of an extra layer of flexibility in scheduling by allowing variable number of data streams per user drastically increases the number of possible scheduling configurations to search through. The optimal solution would be to search over all possible combinations of users, orders (as the precoding method in question is order dependent), and data streams per user, but this proves intractable for larger antenna arrays even for a moderate number of users. The complexity of a scheduling algorithm must therefore be significantly lower than the exhaustive search, while still offering performance relatively close to that of the exhaustive search. The resultant factorial time complexity (see Section 2.2.5 for discussion of complexity) coupled with the high cost of sending necessary user channel feedback to the transmitter, dictates that the short decision time for a potential scheduler will be a limiting factor in performance. As a result, a scheduler that wishes to optimize some metric, be it sum-rate, sum-rate incorporating fairness, or finite delay times, should be able to extract the most throughput due to spatial stream scheduling while keeping execution time low and bounding complexity to ensure that it will be practical to implement.

The aims of this research are firstly the further investigation of the effects of variable data stream allocation under low-complexity linear precoding algorithms and secondly, the implementation and investigation of low-complexity data stream and user scheduling algorithms.

The main contributions of this research are as follows:

- The impact of data stream allocation under successive zero-forcing precoding with coordinated transmit-receive processing on achievable sum-rates is investigated in greater detail. While existing work has focused primarily on users with 1-2 receive antennas

and 2-4 transmit antennas, our analysis is expanded to the case of 4 receive antennas and up to 8 transmit antennas. The exhaustive search for the optimal stream scheduling and the subsequent ordering and search over a large user pool is shown to be intractable, thus simplifications are introduced in order to reduce the effective search space. Finally, the negligible impact of these simplifications on the achievable throughput performance in comparison to the exhaustive search are shown.

- This thesis proposes simulated annealing as a user and data stream scheduling algorithm in a system employing linear transmitter precoding, and investigates the performance of the algorithm under a sum-rate maximization metric. Specifically, the use and performance of our scheduling algorithm under reduced complexity successive zero-forcing precoding with coordinated transmit-receive processing is examined. Simulated annealing algorithms are metaheuristic approaches to solving non-deterministic polynomial-time hard problems and tend to offer fast convergence and relatively good solutions. Implementations of simulated annealing tend to be (for the most part) problem invariant apart from a few parameter choices and tend to work with a wide variety of utility functions.

- The application of data stream scheduling is shown to offer better performance than that of user scheduling alone. Furthermore, the use of simulated annealing as a suboptimal data stream scheduling algorithm is shown to offer performance close to that of the exhaustive search at greatly reduced computational complexity. Data stream scheduling with simulated annealing in certain antenna configurations is also shown to offer better performance than exhaustive search user scheduling at a lower complexity.

1.6 Organization of Subsequent Chapters

- Chapter 2 introduces some relevant background information on MIMO concepts in wireless communications. We examine optimal dirty paper coding (DPC) and suboptimal linear precoding and multiuser interference mitigation methods. We also present the concept of stream allocation via coordinated transmit-receive processing and discuss in detail the scheduling problem created by stream allocation. It is shown that the optimal exhaustive search is intractable by deriving an upper bound on its com-

plexity. Furthermore, we examine some simplifications that can be made to the search space and demonstrate the negligible effect these simplifications have on finding the near-optimal solution while decreasing complexity by several orders of magnitude.

- Chapter 3 presents the simulation environment and the assumptions we make concerning the channel and user environment. We discuss in detail the system model we use for simulations, and clarify our assumptions concerning the finer details of our simulation environment. A brief overview of the simulation software and hardware provided by Westgrid used to run our simulations is also provided.

- In Chapter 4 we present an overview of optimization problems, metaheuristic methods, and general simulated annealing approaches. We then discuss in detail our use of simulated annealing (SA) as a MU-MIMO data stream and user scheduling algorithm used under successive zero-forcing, a linear precoding method. Design specifics of our algorithm are given and justified.

- In Chapter 5 we present the results of our simulations and explain their relevance to modern wireless communication system design. The effects of parameter selection on the sum-rate maximization and convergence time performance of SA are also examined.

- Chapter 6 concludes our work with a few closing remarks and gives a summary of the key points and contributions of this thesis. The remaining questions and potential areas for future investigation are also discussed.

- Lastly, appendices are included with additional results and figures as well as more detailed explanations of any relevant concepts not covered in depth in the earlier chapters.

Chapter 2

Background on Precoding, Receiver Processing, and Data Stream Scheduling

We examine optimal dirty paper coding (DPC) [14, 15] and suboptimal linear precoding and multiuser interference mitigation methods. We also review the concept of stream allocation via coordinated transmit-receive processing and detail the scheduling problem created by stream allocation. By deriving an upper bound on the exhaustive search complexity, it is shown that this method, although optimal, presents an intractable problem. Furthermore, we examine some simplifications that can be made to the search space and show the negligible effects they have on finding the near-optimal solution while decreasing complexity by several orders of magnitude.

2.1 MIMO in Wireless Transmission Systems

Wireless transmissions in modern communications systems are affected primarily by multipath fading, or the arrival of many copies of the signal at the receiver at varying angles, delays, and frequencies due to the scattering of the electromagnetic signal. This random superposition of the received signals creates large variations in the received signal level known as small-scale fading, and presents a significant, performance-limiting problem for conventional wireless systems by affecting the reliability and efficacy of older wireless approaches.

However, the presence of this small-scale fading is actually key to achieving the capacity gains that can be provided by spatial multiplexing in MIMO system, to the degree that the absence of multipath fading will prevent the realization of spatial multiplexing gains in MIMO.

It is for this reason that MIMO techniques involving the use of multiple antennas at the transmitter (in) and receiver (out) have attracted much attention in wireless communications by offering the possibility of increased data rates, and reliability without necessarily increasing bandwidth or power. While providing a layer of diversity that can be exploited using a Rake receiver, the introduction of these multipath components requires the use of additional processing to exploit these gains and adds an additional layer of complexity. With the extra spatial degrees of freedom, single and multi-user MIMO offer powerful potential benefits (array, diversity, multiplexing, and interference reduction gains) to help address the challenges posed in wireless transmission, but at the same time introducing unique new challenges that must be faced in order to exploit these advantages.

2.1.1 Single User MIMO

Consider the single user MIMO channel created by a terminal with n_R receive antennas and a transmitter with n_T transmit antennas (see fig 2.1).

We make the following assumptions:

- the time-variability of the channel characterized by the coherence time of the channel (or the time interval over which channel gains are strongly correlated) is longer than the symbol duration (in other words, the channel gain remains approximately constant over the duration of the symbol)

- the frequency-variability of the channel characterized by the frequency response of the channel is relatively flat, meaning that the coherence bandwidth (or the frequency interval in which frequencies of a signal are likely to exhibit correlated amplitude fading) is larger than the signal bandwidth.

Under these conditions we can model the system with a matrix channel frequency-flat block-fading model where the channel is assumed to be stationary during the transmission period

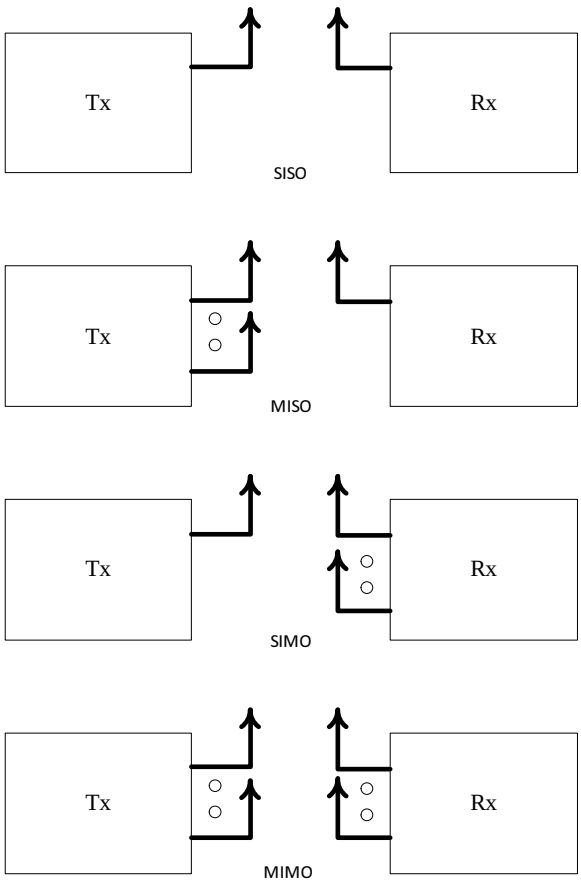

Figure 2.1: Antenna configurations

for one symbol, and all of the frequency components of the transmitted signal are assumed to experience similar fading characteristics across the frequency band.

While this model holds for most narrowband environments, modern communication systems and next-generation wireless standards operate in wideband, or frequency selective fading channels, the gains of which vary significantly across the frequency band and cause intersymbol interference. To cope with the severe channel conditions associated with wideband communications, frequency-division multiplexing schemes such as orthogonal frequency division multiplexing (OFDM) (encoding with several orthogonal subcarriers created through the careful selection of subcarrier separation and symbol duration) can be used to subdivide a wideband channel into several narrowband channels. These resulting narrowband channels can then be considered to experience frequency flat fading as opposed to frequency selective fading. This enables the transmission of many parallel data streams,

and in concert with channel (or error control) coding, allows us to cope with the severe channel conditions associated with wideband communications such as frequency selective fading and narrowband interference. In systems where OFDM is considered, it is possible to implement MIMO processing techniques on each individual subcarrier or sub-band in which the fading characteristics can be considered narrowband.

We define $\mathbf{H}$ to denote the $n_R \times n_T$ channel matrix, with the constituent elements h_{ij} ($i = 1, 2, \ldots, n_R$; $j = 1, 2, \ldots, n_T$), denoting the complex channel gains between the j-th transmit antenna and the i-th receive antenna during one transmission interval. In the single user case all of the n_R receive antennas in the system can cooperate in processing the received data signal. The n-th column of $\mathbf{H}$ is often called the spatial signature of the n-th transmit antenna on the receive array, and the relative distribution or orthogonality of the n_T spatial signatures determines our ability to distinguish signals projected from the different transmit antennas onto the receiver. Assuming the transmitted signal vector is $\mathbf{x} = [x_1\ x_2\ \ldots\ x_{n_T}]^T \in \mathcal{C}^{n_T \times 1}$ the received complex signal vector $\mathbf{y} = [y_1\ y_2\ \ldots\ y_{n_R}]^T$ can be written as

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

with $\mathbf{n}$ indicating the $n_R \times 1$ additive white Gaussian noise (AWGN) vector. The elements of $\mathbf{n}$ are independent identically distributed (i.i.d) circularly symmetric complex Gaussian random variables with zero mean and a variance σ_n^2 per dimension, which implies that $E[\mathbf{n}\mathbf{n}^H] = \sigma_n^2 \mathbf{I}_{n_R}$.

Given the power constraint $\mathrm{Tr}((E\{\mathbf{x}\mathbf{x}^H\})) \leq P$ (with $\mathrm{Tr}(\cdot)$ denoting the matrix trace operation $Tr(A) \equiv \sum_{i=1}^{n} a_{ii}$) and perfect channel state information at the transmitter (CSIT) and receiver, the capacity of the channel can be given by[6]

$$C = \log_2 \prod_{i=1}^{r} \left(1 + \frac{p_i \alpha_i}{\sigma_n^2}\right) \tag{2.2}$$

where α_i denote the squared singular values of $\mathbf{H}$ and r =rank($\mathbf{H}$) . The variable μ and the individual power weights p_i are determined by waterfilling[16] over the eigenmodes of $\mathbf{H}$ such that $p_i = max\{(\mu - \sigma^2/\alpha_i), 0\}$ given the power constraint $\sum_{i=1}^{r} p_i = P$.

As reported previously in [4, 6], the capacity of the MIMO channel will scale with $\min(n_T, n_R)$ providing that the channel $\mathbf{H}$ is full rank. However, the degree of correlation between the channel gains is usually dependent on the scattering environment and antenna

spacing at the receiver and transmitter. With adequate spacing at the receiver (i.e. a distance greater than half the wavelength of the signals received) and a rich, omni-directional, isotropic scattering environment will help to result in a full rank channel with a high degree of decorrelation [16, 17, 18].

A common simulation environment that achieves this is where the individual channel gains are modeled as zero-mean circularly symmetric complex Gaussian random variables. Thus the magnitudes of the gains $|H_{i,j}|$ are Rayleigh distributed random variables with corresponding exponentially distributed powers $|H_{i,n}|^2$.

2.1.2 Multi-User MIMO

Unlike single user MIMO where all of the receiver antennas can cooperate in the decoding of the received signal, in MU-MIMO it is usually assumed that no coordination exists between antennas of different user terminals (though coordination between antennas co-located on the same user terminal naturally exists). If full coordination existed between all of the user terminals and their corresponding antennas, the system would reduce to an equivalent single user MIMO case. Assuming K active mobile terminals and one base station, two basic MU-MIMO transmission links exist:

1. the downlink (forward channel), or broadcast channel (BC) - the link from the base station (network node) to multiple users (fig. 2.2a)

2. the uplink, or multiple-access channel (MAC) - the link from multiple user terminals to the base station (fig. 2.2b).

In the context of cellular architectures, the MAC models the uplink between the K user terminals and the base station, and the BC models the downlink from the base station to the mobile users. In both cases due to the simultaneous nature of the data transmission, the problem of cross-talk or multi-user interference (MUI) arises and can cause undesired performance degradation unless interference mitigation techniques are employed to counter the effects of the unwanted signals and separate the received data streams. Time, frequency, or space division multiplexing, with suitable signal processing (precoding at the transmitter, or processing at the receiver) can be implemented to help mitigate MUI, but will introduce upper bounds as to the maximum number of users that can be simultaneously served. Thus,

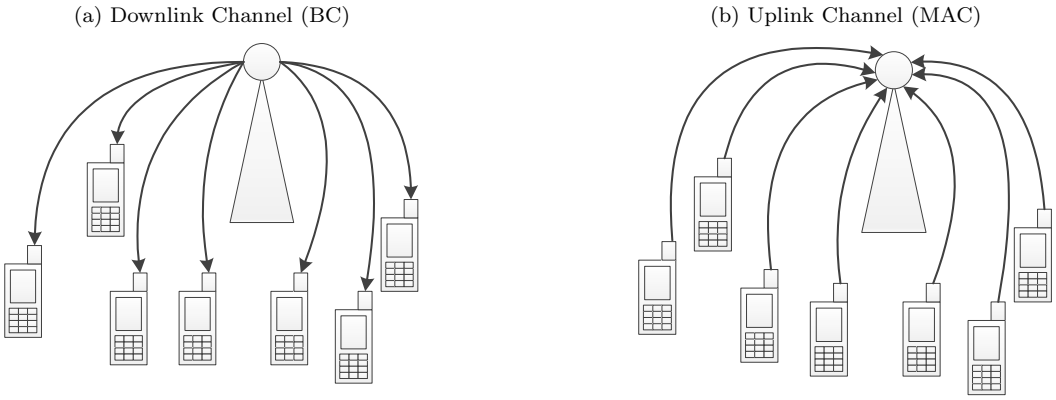

Figure 2.2: MU-MIMO transmission links

the large number of users requesting service and a subsequent limit on the maximum number of users that can be served simultaneously necessitates a scheduler to select which users to give service to and will be further discussed in section Chapter 4, which also discusses optimization (including optimization of user and data stream scheduling).

2.1.3 Uplink Multiple Access Channel and Downlink Broadcast Channel Capacities

On the MAC, or uplink channel, all of the K users with n_R antennas per terminal simultaneously transmit a signal to the base station. Let $\mathbf{x}_k \in \mathcal{C}^{n_R \times 1}$ be the data vector transmitted by user k, with $\mathbf{H}_k^{MAC} \in \mathcal{C}^{n_T \times n_R}$ denoting the uplink channel matrix of user k, and $\mathbf{n} \in \mathcal{C}^{n_T \times 1}$ designating the complex additive white Gaussian noise vector with zero mean and a variance 0.5 per dimension.

The received signal vector $\mathbf{y}_{MAC}$ at the base station can be then written as

$$\mathbf{y}_{MAC} = \sum_{k=1}^{K} \mathbf{H}_k^{MAC} \mathbf{x}_k + \mathbf{n} \tag{2.3}$$

where each user is subject to an individual power constraint of $P_k \geq Tr(\mathbf{\Sigma}_k)$ with $\mathbf{\Sigma}_k$ defining the transmit covariance matrix of each user $\mathbf{\Sigma}_k \triangleq E[\mathbf{x}_k \mathbf{x}_k^H]$.

With per-user power constraints, the capacity region of the MAC can be defined as the convex hull enclosing the set of user rates that can be achieved with an arbitrarily small probability of error. The sum-capacity is the highest possible sum of these user rates. It is known that we can achieve this capacity region if the base station employs successive

interference cancelation to decode the individual signals [6, 19]. Assuming the order vector of the K users is given by vector $\pi = [\pi(K)\,\pi(K-1)\,\ldots\,\pi(2)\,\pi(1)]$ where $\pi(K)$ denotes the *first* encoded user, the receiver first decodes the signal of user $\pi(K)$ treating the signal from user $\pi(K-1)$ (and each subsequently encoded user all the way to $\pi(1)$) as an extra source of additive Gaussian noise. The receiver (or base station in the case of the MAC channel) then re-encodes the decoded signal and subtracts it from the combined received signal. The signals for the subsequent users are similarly and successively decoded, re-encoded, and subtracted, until all users signals are decoded. Due to the successive nature of this process, the order the users are decoded in will affect the

The individual rate achievable for user $\pi(k)$ is given by

$$R_{\pi(k)}^{MAC} = \log_2 \frac{\left|\mathbf{I} + \sum_{i=1}^{k} \mathbf{H}_{\pi(i)}^{MAC} \mathbf{P}_{\pi(i)} \left(H_{\pi(i)}^{MAC}\right)^H\right|}{\left|\mathbf{I} + \sum_{i=1}^{k-1} \mathbf{H}_{\pi(i)}^{MAC} \mathbf{P}_{\pi(i)} \left(H_{\pi(i)}^{MAC}\right)^H\right|} \tag{2.4}$$

and subsequently the MAC sum rate becomes

$$R_{MAC} = \sum_{k=1}^{K} R_{\pi(k)}^{MAC} = \log_2 \left|\mathbf{I} + \sum_{i=1}^{K} \mathbf{H}_{\pi(i)}^{MAC} \mathbf{P}_{\pi(i)} (\mathbf{H}_{\pi(i)}^{MAC})^H\right|. \tag{2.5}$$

Now, given the downlink channel matrix of the k-th user on the broadcast channel as $\mathbf{H}_{\pi(k)}^{BC} \in \mathcal{C}^{n_R \times n_T}$, the received signal $\mathbf{y}_k \in \mathcal{C}^{n_r \times 1}$ at each user k is given by

$$\mathbf{y}_k = \mathbf{H}_{\pi(k)}^{BC} \sum_{i=1}^{K} \mathbf{x}_i + \mathbf{n}_k. \tag{2.6}$$

The vector $\mathbf{x}_k \in \mathcal{C}^{n_T \times 1}$ denotes the data vector transmitted by the base station designated for user k, with $\mathbf{n}_k \in \mathcal{C}^{n_R \times 1}$ representing the complex AWGN received by the user k in question. As opposed to the single user MIMO case where all of the receive antennas (on the user in question) are assumed to cooperate in the decoding of the received signal, no cooperation is possible in the downlink channel between users in MU-MIMO, although cooperation occurs for co-located antennas on the same user. It is therefore the task of the transmitter to implement the interference mitigation or precoding algorithms, thereby enabling the separation of the desired received signal vector at each desired terminal. Also in contrast to the single user case, a base station transmitting to multiple co-channel users requires CSIT [11] to extract multiuser gains. However due to estimation errors, quantization, and limited resources for the feedback information, obtaining full CSIT for all users is usually not possible

in practice. Nevertheless, MU-MIMO will benefit even from statistical, limited, partially incorrect, or outdated channel information[12, 20, 21]. This is because the transmitter can use the CSIT to reduce the intra-cell interference produced by the desired user's signal at the other user terminals via beamforming, enabling high signal-to-interference-plus-noise ratios (SINRs) at the receiver[18].

Given perfect channel knowledge and non-causal knowledge of the interference source at the transmitter, it is possible to encode the transmitted signals for many users at the transmitter in a way that will achieve capacity by using a technique known as Dirty Paper Coding (DPC) [14, 22]. DPC precoding for MU-MIMO is based on the idea of idea that if the transmitter has advanced, non-causal knowledge of the interference introduced by transmission for each user, the signals for each user can be structured in a way that compensates and effectively removes the effect of this interference.

In the case of the MU-MIMO BC, the signal intended for any user k is the interference for any of the other users $\{1, 2, \ldots k-1, k+1, \ldots K\}$. Given that the base station inherently knows all of the data destined for each user, it can successively encode the data using DPC to effectively remove the interference on user k from any other previously encoded interfering user j (for all $j < k$). Due to the successive nature of the encoding and similar to the MAC case, the rates of each user will be affected by their encoding order $[\pi(1)\ \pi(2)\ \ldots\ \pi(K)]$ (respectively denoting the first, second, and the K'th encoded users).

From [9] the achievable DPC sumrate on the BC for each encoded user can then be given by

$$R^{BC}_{\pi(k)} = \log_2 \frac{\left| \mathbf{I} + \mathbf{H}^{BC}_{\pi(k)} \left(\sum_{j \geq k} \mathbf{\Sigma}_{\pi(j)} \right) \left(\mathbf{H}^{BC}_{\pi(k)} \right)^H \right|}{\left| \mathbf{I} + \mathbf{H}^{BC}_{\pi(k)} \left(\sum_{j > k} \mathbf{\Sigma}_{\pi(j)} \right) \left(\mathbf{H}^{BC}_{\pi(k)} \right)^H \right|} \tag{2.7}$$

with the covariance matrix of the k-th user's data given by $\mathbf{\Sigma}_k = E\{\mathbf{x}_k \mathbf{x}_k^H\} \in \mathcal{C}^{n_T \times n_T}$. The DPC achievable rate region is then defined as the set of all achievable rate-tuples for all users given all possible orders and respective covariance matrices $\mathbf{\Sigma}_k$ subject to the transmit sum-power constraint $\sum_{\forall k} Tr(\mathbf{\Sigma}_k) \leq P$. Research has shown [9, 14, 15, 23] that DPC can achieve the capacity of the MU-MIMO BC and is an optimal encoding method, however obtaining the optimal transmit covariance structure is a computationally complex non-convex problem. In contrast, the problem of finding the MIMO MAC capacity region is shown [24] to be a simpler, well-structured convex problem. Work [23, 15, 24] on characterizing the capacity

region of the BC has shown the existence of a MAC/BC duality stating that the capacity rate region of the BC is equivalent to the case of the MAC. Specifically, given a BC with a set of downlink channels $\mathbf{H}_k$ and sum-power constraint $\sum_{\forall k} Tr(\mathbf{\Sigma}_k) \leq P$ encoded with order $\pi = \{\pi(K)\,\pi(K-1)\,\ldots\pi(1)\}$, the achievable rate region on the BC is equivalent to the rate region achievable on the dual MAC channels $\mathbf{H}_k^H$ with sum-power constraint $\sum_{\forall k} Tr(\mathbf{P}_k) \leq P$ and reversed (in comparison to the MAC) user ordering order π such that $\pi(1)$ is encoded first.

Although DPC can achieve the capacity of the BC, employing it requires a very complex transmitter and receiver structure, nonlinear processing, and noncausal knowledge of interferers. These factors preclude its adoption in real world applications, and as such much work in research has been dedicated to finding practical, although suboptimal, non-linear and linear precoding methods.

Various non-linear approaches allowing varying degrees of tradeoff between complexity and performance have since been proposed including: Tomlinson-Harashima precoding [25], vector perturbation [26, 27, 28], lattice encoding [29], Trellis encoding [30], and superposition coding. One approach is vector-perturbation precoding[26] in which the transmit data is perturbed in such a way that the largest signal component is placed along the smallest channel inverse and vice versa. This minimizes the required transmit power and has been shown [31] to offer a reasonable performance-complexity tradeoff, and outperform Tomlinson-Harashima precoding[25, 32] (which is itself a more constrained version of vector perturbation) when users are deployed with single-antenna receivers. An extension of vector perturbation presented in [33] is multiple-user multi-stream vector perturbation (MUMS-VP). This extended VP to multi-antenna users, and assumed coordination in decoding between co-located antennas on each user, and showed the benefits of multiple data stream allocation and the relatively robust performance of a VP-based precoding method. While these non-linear approaches provide us with slightly simpler alternatives to DPC while offering quite good performance, the non-linear nature of these methods leaves room for simpler yet adequately performing techniques.

Linear precoding techniques can fill this gap, as they are relatively simple yet tend to offer reasonable performance. Although linear precoding methods tend to perform worse than their non-linear counterparts, various linear techniques have been shown [34, 35, 36, 37, 38]

to asymptotically approach capacity at high SNR, at large numbers of users K, or when the system is interference-limited. Therefore, we choose to focus our work on linear precoding methods, and discuss the details of linear precoding methods in the section below.

2.2 Suboptimal Linear Precoding and Interference Mitigation Techniques for MIMO

2.2.1 Linear Beamforming Techniques

As mentioned previously, the complexity of DPC and other nonlinear methods leaves a gap to be filled in terms of simpler yet reasonably performing solutions. Linear beamforming methods are of particular interest due to their relatively low complexity and relatively good performance. The data vectors for each user $\mathbf{s}_k \in \mathcal{C}^{n_R \times 1}$ are processed individually by a beamforming matrix $\mathbf{W}_k \in \mathcal{C}^{n_T \times n_R}$, giving us the aggregate signal $\mathbf{x} = \sum_{\forall k} \mathbf{W}_k \mathbf{s}_k$. The beamforming matrix $\mathbf{W}_k$ predistorts the phase and relative amplitude of each user's target signal at each transmitter in order to create a pattern of constructive and destructive interference in the wavefront at each prospective receiver. These matrices can be designed to remove some or all of the interference between the users on the transmitter side, and can be weighted individually to yield different SINR's for each user depending on any given rate requirements.

In terms of the optimality of beamforming, as the number of users K increases and with it the subsequent likelihood of finding users with near-orthogonal channels, the sum rate achievable with certain beamforming techniques asymptotically approaches the capacity achievable with DPC[36, 34]. Therefore when used in environments with little preexisting interference (orthogonal user channels with little cross-talk or MUI), linear beamforming has the capacity to achieve results similar to those achievable with DPC.

Let us consider a linear transmit precoding method called zero-forcing beamforming[39], in which multiple single antenna users are placed in a system with a multi-antenna transmitter that employs a beamforming matrix that completely cancels MUI by inverting the channel gains. Zero-forcing beamforming[34, 39] is a case of channel inversion which creates a set of orthogonal, non-interfering channels and completely countering the effects of the interfering users. Assume a multi-user MISO system where there exists a set of K single antenna

users with corresponding channels $\mathbf{H}_k$, where $K \leq n_T$. Defining the concatenated aggregate channel matrix $\mathbf{H} = \left[\mathbf{H}_1^T\ \mathbf{H}_2^T\ \ldots, \mathbf{H}_K^T\ \right]^T$ we can construct a Moore-Penrose pseudo-inverse $\mathbf{H}^\dagger$ of the resulting $\mathbf{H}$, defined as $\mathbf{H}^\dagger = \mathbf{H}^H(\mathbf{H}\mathbf{H}^H)^{-1}$. The resulting transmitted signal can then be given by $\mathbf{x} = \mathbf{H}^\dagger\mathbf{s}$, where $\mathbf{s} \in \mathcal{C}^{K\times 1}$ is the data vector of symbols s_k for each user k. Thus, the beamforming vector for each user is the k-th column of $\mathbf{H}^\dagger$, which is structured to cancel either all of the interference from other users. The use of channel inversion however will not result in linear capacity increases with $\min(n_T, n_R)$ as would be expected with a multi-user channel[14, 36, 34].

While it is possible to structure the beamforming matrices in a way to completely cancel all of the interference for all of the single antenna users, this will not necessarily maximize the sum rate. This is highlighted in the case of zero-forcing for any users suffering from deep fades or generally poorly conditioned channels. In addition to the large amount of power required to counter the effects of the weak channels, channel inversion results in noise-enhancement at the receiver terminal. While any linear precoding (as well as equalization) results in noise enhancement, it is particularly severe for straight channel inversion for MU-MIMO DL with multiple-antenna users. The performance of this method[21] (as well as that of any form of precoding) is also dependent on accurate CSIT and suffers when CSI is inaccurate or delayed as residual MUI remains uncanceled due to imperfect beamforming vector construction. Furthermore, if complete MUI cancelation via zero-forcing beamforming is extended defacto to the multi-antenna receiver case it requires the consideration of each antenna as a separate user. This fails to consider that the antennas co-located on a receiver can cooperate in the processing of the received signals, and results in sub-par performance compared to DPC especially at higher SNR [36].

2.2.2 Block Diagonalization

An extension of zero-forcing to multiple antenna receivers in a MU-MIMO system, a technique called block diagonalization (BD) [37] completely nulls interference between users while accommodating coordination between antennas at the receivers. Recall the aggregate channel matrix $\mathbf{H} = \left[\mathbf{H}_1^T\ \mathbf{H}_2^T\ \ldots\ \mathbf{H}_K^T\right]^T$, and define the aggregate precoding matrix of all K users $\mathbf{W} = [\mathbf{W}_1\ \mathbf{W}_2\ \ldots\ \mathbf{W}_K]$. BD designs each $\mathbf{W}_k$ to eliminate multiuser interference such that $\mathbf{H}_k\mathbf{W}_j = 0$, $\forall k \neq j$. This decomposes the multiuser channels into single user

channels, resulting in a block diagonal structure of the matrix product $\mathbf{HW}$.

Consider the concatenated aggregate channel matrix

$$\breve{\mathbf{H}}_k = \left[\mathbf{H}_1^T, \mathbf{H}_2^T, \ldots, \mathbf{H}_{k-1}^T, \mathbf{H}_{k+1}^T, \ldots, \mathbf{H}_K^T\right]^T \tag{2.8}$$

containing the channels for all users except the kth. In BD the design of the precoding matrices $\mathbf{W}_k$ is done such that they lie in the null space of $\breve{\mathbf{H}}_k$ for each $k = 1 \ldots K$. This requires a nullspace of $\breve{\mathbf{H}}_k$ with nonzero dimension, consequently imposing constraints on the rank and therefore the total number of receive antennas that can be supported for a given number of n_T. To elaborate on this antenna dimensionality constraint, we denote the singular value decomposition of $\breve{\mathbf{H}}_k$ as $\breve{\mathbf{H}}_k = \breve{\mathbf{U}}_k \breve{\mathbf{\Sigma}}_k (\breve{\mathbf{V}}_k^1 \breve{\mathbf{V}}_k^0)$, where $\breve{\mathbf{U}}_k$ are the left singular vectors, $\breve{\mathbf{\Sigma}}_k$ is the $n_R \times n_R$ diagonal matrix containing the $\breve{r}_k = \text{rank}(\breve{\mathbf{H}}_k)$ nonzero singular values of $\breve{\mathbf{H}}_k$, $\breve{\mathbf{V}}_k^1$ are the first $\breve{r}_k$ right singular vectors, and $\breve{\mathbf{V}}_k^0$ are the $n_T - \breve{r}_k$ right singular vectors. The zero multiuser interference condition can then be satisfied by constructing the precoding matrix $\mathbf{W}_k$ for each of the users out of n_R column vectors of $\breve{\mathbf{V}}_k^0$. Assuming full rank channel matrices $\mathbf{H}_k$, sending n_R data streams under BD for each of the K users is possible if $\max(\breve{r}_1, \breve{r}_2, \ldots, \breve{r}_K) < n_T$, and $n_T \geq \sum n_R$ if no coordinated transmit receiver processing (which is discussed below in section 2.2.4) is employed.

Under BD the multiuser MIMO channel is then decoupled into K parallel non-interfering single-user MIMO channels, and the achievable throughput of the system under a sum power constraint is given by

$$R_{BD} = \max_{\mathbf{Q}_k : \mathbf{Q}_k \succeq 0} \sum_{k=1}^{K} \log_2 \det \left(\mathbf{I} + \frac{1}{\sigma_n^2} \mathbf{H_k} \breve{\mathbf{V}}_k^0 \mathbf{Q}_k (\mathbf{H_k} \breve{\mathbf{V}}_k^0)^H \right) \tag{2.9}$$

where the transmit covariance matrices of each user $\mathbf{Q}_k = E\{\mathbf{s}_k \mathbf{s}_k^H\}$ are structured such that $\sum_{\forall k} \text{Tr}(\mathbf{Q}_k) \leq P$ with the individual powers achieved via waterfilling. While BD performs better than full zero-forcing channel-inversion directly extended to multiple antenna users, its performance still leaves room for improvement to the rates achievable with DPC. Motivated by the example of DPC, which only cancels interference successively, we can infer that complete removal of MUI may not necessarily be optimal for maximizing sum rates at the receivers. Furthermore, it can be shown that by relaxing the zero-interference condition we can improve the performance of linear beamforming techniques.

2.2.3 Successive Zero-Forcing (SZF)

Motivated by the incomplete interference cancelation approach in DPC, and the sub-optimality of BD, we look at an incomplete interference cancelation scheme called successive zero-forcing (SZF) [38], which can be considered an extension of BD via the use of a relaxed nullspace constraint. Unlike BD in which the precoding matrix for each user lies in the nullspace of all encoded users, SZF restricts the precoding matrix for user i to lie in the nullspace of only the previously $i-1$ encoded users, implying that an encoding order π must be specified. Consider a set of K users, with the encoding order is defined as $\pi = [\pi(1)\,\pi(2)\,\ldots\,\pi_K]$, π_i denoting the index of the i-th encoded user. The i-th encoded user is therefore only subject to interference from the $i-1$ previously encoded users.

We define

$$\mathbf{L}^m = [l_1 l_2 \cdots l_K] : \{l_i \leq n_R, \sum_{i=1}^{K} l_i = n_T\} \tag{2.10}$$

as the data stream vector with each element l_i corresponding to the number of data streams sent respectively to each encoded user. Consider the set of users $\mathcal{S}$ simultaneously requesting service, and the subset of users $\mathcal{S}_T$ (where $K = |\mathcal{S}_T|$) with the particular order π created by the projection of $\mathbf{L}^m$ onto a particular set of users. Defining the previously $i-1$ encoded user's channels as $\bar{\mathbf{H}}_{i-1} = [\mathbf{H}^T_{\pi(1)}\mathbf{H}^T_{\pi(2)}\ldots\mathbf{H}^T_{\pi(i-1)}]$, we can define the SVD of $\bar{\mathbf{H}}_{i-1}$ as $\bar{\mathbf{H}}_{i-1} = \bar{\mathbf{U}}_{i-1}\bar{\mathbf{\Sigma}}_{i-1}[\bar{\mathbf{V}}^1_{i-1}\bar{\mathbf{V}}^0_{i-1}]^H$. The precoding matrix for user $\pi(i)$, $W_{\pi(i)}$ is then only constrained to lie in space composed by the basis formed by $\bar{\mathbf{V}}^0_{i-1}$, the $n_T - \text{rank}(\bar{\mathbf{H}}_{i-1})$ nullspace vectors of $\bar{\mathbf{H}}_{i-1}$.

Given a set of K users with order π the received signal for each user k can be expressed as

$$y_{\pi(k)} = \mathbf{H}_{\pi(\mathbf{k})}\left(\mathbf{W}_{\pi(k)}\mathbf{x}_{\pi(k)} + \sum_{i<k}\mathbf{W}_{\pi(i)}\mathbf{x}_{\pi(i)} + \sum_{i>k}\mathbf{W}_{\pi(i)}\mathbf{x}_{\pi(i)}\right) + \mathbf{n}_{\pi(k)} \tag{2.11}$$

with the third term $\sum_{i>k}\mathbf{W}_{\pi}(i)\mathbf{x}_{\pi(i)}$ cancelled, since by definition $\mathbf{H}_{\pi(k)}\mathbf{W}_{\pi(i)} = 0, \forall i > k$. Following the forward interference constraint, the received signal vector for each user can then be reduced to

$$y_{\pi(k)} = \mathbf{H}_{\pi}(k)\left(\mathbf{W}_{\pi}(k)\mathbf{x}_{\pi(k)} + \sum_{i<k}\mathbf{W}_{\pi(k)}\mathbf{x}_{\pi(k)}\right) + \mathbf{n}_{\pi(k)}. \tag{2.12}$$

Assuming K users, precoding with SZF is possible if

$$\text{rank}(\bar{\mathbf{H}}_{K-1}) < n_T. \tag{2.13}$$

Assuming no coordinated transmit-receive processing (which can selectively choose to transmit on less than the maximal number of modes, and is described in detail in the subsequent section) and full rank users each with n_R antennas and independent channels, this condition reduces to $n_R \times K < n_T$ (assuming each user must receive n_R data streams). The number of users is then restricted by the dimensions of the nullspaces. Thus, the upper bound on the number of full rank users that can be scheduled (as with BD) is $K_{max} = \lceil n_T/n_R \rceil$, with each users data stream allocation $l_i = n_R$, $i = \{1, \ldots, K_{max}\}$.

The maximum achievable rate for user $\pi(i)$ given an ordering π of a set of users $\mathcal{S}_T$ in the case of Gaussian-distributed transmit signal vectors $\mathbf{x}$ can then be written as [38]

$$R_{\pi(i)} = \log_2 \frac{\left| \mathbf{I} + \mathbf{H}_{\pi(i)} \left(\sum_{j=1}^{i} \bar{\mathbf{V}}_{j-1}^0 \mathbf{Q}_{\pi(j)} (\bar{\mathbf{V}}_{j-1}^0)^H \right) \mathbf{H}_{\pi(i)}^H \right|}{\left| \mathbf{I} + \mathbf{H}_{\pi(i)} \left(\sum_{j=1}^{i-1} \bar{\mathbf{V}}_{j-1}^0 \mathbf{Q}_{\pi(j)} (\bar{\mathbf{V}}_{j-1}^0)^H \right) \mathbf{H}_{\pi(i)}^H \right|} \tag{2.14}$$

with the input covariance matrices $\mathbf{Q}_{\pi(i)}$ defined as follows:

$$\mathbf{W}_{\pi(i)} \mathbf{W}_{\pi(i)}^H = \bar{\mathbf{V}}_{i-1}^0 \mathbf{Q}_{\pi(i)} (\bar{\mathbf{V}}_{i-1}^0)^H. \tag{2.15}$$

The maximum achievable sum-rate under SZF for that particular set of users $\mathcal{S}_T$ with order π_c given a transmit power constraint P is then

$$R_\pi = \max_{\{\mathbf{Q}_j\}_{j \in \{1,2 \cdots K\}} : \mathbf{Q}_j \succeq 0, \sum_j \mathrm{Tr}(\mathbf{Q}_j) \leq P} \sum_{j=1}^{K} R_{\pi(j)}. \tag{2.16}$$

Maximizing (2.16) over $|\bar{\pi}|$ possible user orders gives us the best achievable sum-rate for the user set $\mathcal{S}_T$,

$$R = \max_{\pi \in \bar{\pi}} R_\pi. \tag{2.17}$$

To obtain the maximum possible rate for the entire user set $\mathcal{S}$, we must search over all possible subsets $\mathcal{S}_T$ of K_o users, and subsequent ordering possibilities $\bar{\pi}$ of the users in these subsets, for a total of $\frac{|\mathcal{S}|!}{(|\mathcal{S}|-K_o)!}$ possible permutations. Due to this combinatorial nature of the complexity, exhaustive searches become prohibitively complex for large user populations $|\mathcal{S}| >> K_o$.

Determining the optimal covariance matrices and computing the solutions to Eqn. (2.14)-(2.17) is a non-convex problem, and can be quite computationally expensive. In lieu of the prohibitive optimal calculations, suboptimal numerical techniques were proposed in [38]

based on DPC covariance optimization to solve for the covariance matrices for a given user ordering. The authors find the optimal covariance matrices for the MAC using sum-power iterative waterfilling method in [24], and then obtain the covariance matrices for the broadcast channel under DPC employing the MAC to BC transformation from [9] (using the principle of the MAC-BC duality). Finally, the projection of these BC covariance matrices onto the SZF nullspaces is used to obtain each $\mathbf{Q}_{\pi(k)}$. We use this method for determining our covariance matrices in the rest of our work.

2.2.4 Improved Covariance Optimization Methods for SZF

Although the method mentioned in section 2.2.3 for numerically determining the transmit covariance matrices (and thus the achievable rates) works reasonably well, the authors in [38] acknowledged that it was suboptimal. Further investigation in [40] revealed that when using this method in the case of larger transmit antenna arrays at higher SNR and with increased number of users, the achievable throughput with SZF fell below that of BD. This is counterintuitive, since by nature BD is a more constrained version of the optimization problem of SZF. To briefly elaborate, any solution that satisfies the BD constraints of no interference automatically satisfies the forward interference constraints for SZF. The performance of SZF should then be theoretically lower bounded by the performance BD. It was found that the problem lay with the covariance optimization (particularly the SZF nullspace projection). Additional problems arise in cases where weighted sum-rate maximization is used, as the original method assumed unweighted sum rate maximization for the generation of the subsequent matrices.

Since our work assumes unweighted (or equivalently equally weighted) sum rates, it does not suffer from that latter deficiency, however it still suffers from other problems in the covariance optimization method that were mentioned in [40]. A better performing but still suboptimal method was subsequently proposed by the authors in [40] at the cost of additional complexity which remedied these shortcomings. However, to help reduce the complexity of our problem, we use the simpler method proposed in [38] and acknowledge its shortcomings.

2.2.5 Variable Data Stream Scheduling via Coordinated Transmit-Receive Processing

As shown above, the dimensionality constraints of SZF given in 2.13 constrain the number of users that can be supported at any given time. Furthermore, as noted in [41], even though it was implicitly allocated the full n_R modes, the waterfilling operation will sometimes result in transmission on less than the n_R possible modes per user thereby resulting in an overall higher throughput. Spurred by previous research [35, 42, 43, 44, 45, 46, 47, 48] in variable resource allocation intuitive benefits of additional degrees of freedom to leverage during transmission, it lends credit to the idea that explicitly limiting the transmission to certain modes of certain users can yield improved performance.

The explanations of BD and SZF given in the previous subsection assume that $n_T \geq n_R$, and assume full rank channels $H_{\pi(i)}$ thus giving us the maximum number of supported users $K_o = \lceil n_T/n_R \rceil$. However, by coordinating the signal processing between the transmitter and the receivers as proposed in [37] (which uses the results of the work on the power control problem in [49]), the transmitter can transmit up to n_T interference free data streams regardless of the number of users.

Coordinated transmit-receiver processing requires joint design of transmit precoding matrices and receiver coefficients, unlike cases above in which the transmit precoders and the receive filters are designed independently. Assuming minimum mean square error (MMSE) receivers, we know that since the transmitter has advanced knowledge of the channel and the data vectors being sent it can predict what MMSE coefficients for each of the receivers will be. Assuming one data stream is allocated per user ($l_i = 1 \ \forall i$) thus allowing up to n_T users, a better choice for the transmit vectors can be found by choosing an initial set of receive vectors, and alternatively recomputing the corresponding transmit vectors in an iterative fashion until they converge to the lowest power solution. However this method only works for single data streams per user, is inherently complex with no bounded finite-time performance, and has no guarantee on convergence as it depends on the choice of initial transmit and receive vectors.

Due to the potentially unbounded nature of the iterative method and because no iterative extension has been proposed for the case of more than one data stream per user, it has been proposed to choose a reasonable receiver estimate and then apply BD or SZF, reducing the

complexity and allowing a blockwise optimization of the transmit vectors for multiple data users. Assuming the receiver coefficients to be the first l_k left singular values of the SVD of the channel matrices $\mathbf{H}_k$, the effective channels seen at the transmitter can be written as

$$\hat{\mathbf{H}}_k = \mathbf{U}_{l_k}^H \mathbf{H}_k. \tag{2.18}$$

Let us define the aggregate effective block channel matrix of the user set $\mathcal{S}_T$ as

$$\tilde{\mathbf{H}}_{\mathcal{S}_T} = \begin{bmatrix} \hat{\mathbf{H}}_1 \\ \hat{\mathbf{H}}_2 \\ \cdots \\ \hat{\mathbf{H}}_K \end{bmatrix} = \begin{bmatrix} \mathbf{U}_1^H \mathbf{H}_1 \\ \mathbf{U}_2^H \mathbf{H}_2 \\ \cdots \\ \mathbf{U}_K^H \mathbf{H}_K \end{bmatrix} \tag{2.19}$$

which we can now substitute for $\bar{\mathbf{H}}_k$. Forcing $\sum l_i \leq n_T$ and $l_i < n_R$ to satisfy (2.13), we can then apply either BD or SZF to the effective channels $\tilde{\mathbf{H}}_{\mathcal{S}_T}$ and serve up to n_T users.

It is worthwhile to note that a more accurate yet computationally expensive approach for the receiver structure would include the effect of the SZF precoding matrices $\mathbf{W}_k$ on the effective channels $\hat{\mathbf{H}}_k$, and these matrices would be iteratively recomputed until a solution with a resulting maximal sum-rate was found. However, again due to the iterative nature of this approach with no bounds on the finite-time performance we choose to ignore the effects of the precoding matrix allowing for a reasonably performing, computationally simple solution.

Variable Data Stream Scheduling Complexity

Variable data stream allocation via coordinated transmit-receive processing presents an additional degree of freedom, which we refer to as "multi-stream diversity" that can now be exploited. While allowing greater flexibility in transmission and coding, it severely compounds the scheduling problem as now the scheduler must choose which of the n_R modes provided by each user to transmit on, adding an additional layer of combinatorial complexity.

Recall from 2.10 where we defined $\mathbf{L}^m = [l_1 l_2 \cdots l_K]$ as the m-th unique integer partition of the maximum possible number of data streams n_T, with l_j denoting the number of data streams allocated to user j. To simplify the problem of selecting modes for transmission, we follow [38] where the authors choose to transmit only on the l_j dominant modes of each

n_T	n_R	Valid stream allocations
4	2	[2,2] [2,1,1] [1 1 1 1]
4	4	[4] [3,1] [2,2] [2,1,1] [1,1,1,1]
8	2	[2,2,2,2] [2,2,2,1,1] [2,2,1,1,1,1] [2,1,1,1,1,1,1] [1,1,1,1,1,1,1,1]

Table 2.1: Valid data stream allocation for common antenna configurations

user's channel, which does away with the additional $\binom{n_R}{l_j}$ possible combinations over all possible transmission modes for each user with negligible loss in performance.

This idea of transmitting on the dominant eigenmodes follows the single user MIMO approach for throughput maximization, in which the majority of the power is allocated to the dominant modes of the channel. While this transmission mode limitation may result in potential losses in cases where the dominant eigenmodes from a limited user pool are correlated due to correlated channels, this effect has negligible impact on performance in environments with large number of users and thus a greater chance of finding orthogonal subchannels, in which we consider our scheduling problem.

To obtain the maximum possible rate for the entire user set $\mathcal{S}$ searching only over the dominant modes of each user's channel, the optimal solution can be obtained by searching over all of the unique projections of the integer partitions of n_T streams (satisfying $\sum l_k = n_T$, $l_k \leq n_R$) (see Table 2.1 for examples of valid allocations), and all of the orders π of the resultant user subsets $\mathcal{S}_T$.

The exhaustive search must therefore calculate the rate for each of the

$$\sum_{j=1}^{b_{n_T}} \frac{\binom{|\mathcal{S}|}{|\mathbf{L}^j|}(|\mathbf{L}^j|!)^2}{\prod_{\forall i}\phi_i!} \tag{2.20}$$

unique solutions, where b_{n_T} is the number of integer partitions of n_T data streams satisfying $l_i \leq n_R, \forall i$, and ϕ_j denotes the number of times the value of j appears in $\mathbf{L}^m$. This is in contrast to the case of user allocation, in which the scheduler allocates a full n_R streams per user scheduling a maximum of $\lfloor \frac{n_T}{n_R} \rfloor$ at a time. In this case the number of unique solutions necessary to be searched is given by

$$\frac{|\mathcal{S}|!}{(|\mathcal{S}| - \frac{n_T}{n_R})!} \tag{2.21}$$

(for the case where n_T is an integer multiple of n_R). A plot of the number of unique solutions vs. the number of users requesting service for maximal eigenmode scheduling in different

antenna environments is presented in fig. 2.3. Table 2.2 presents numerical complexity comparisons for both exhaustive maximal eigenmode and user allocation search space sizes.

(n_T, n_R)	$\|\mathcal{S}\| = 10$	$\|\mathcal{S}\| = 20$	$\|\mathcal{S}\| = 30$	$\|\mathcal{S}\| = 40$
Exhaustive Maximal Eigenmode Allocation Search Complexity (number of unique solutions)				
(4,2)	7290	137 180	731 670	2 372 760
(4,4)	7470	137 941	733 411	2 375 882
(8,2)	8 623 440	8 251 345 080	314 397 394 920	3 800 788 002 960
(8,4)	10 446 570	8 468 871 140	617 410 045 710	3 819 410 342 280
Exhaustive User Allocation Complexity (number of unique solutions)				
(4,2)	90	380	870	1560
(4,4)	10	20	30	40
(8,2)	5040	116 280	657 720	2 193 360
(8,4)	90	380	870	1560

Table 2.2: Solution space size for exhaustive maximal eigenmode and exhaustive user allocation scheduling

Specifically for larger values of n_T and $|\mathcal{S}_T|$, where many allocations of one or two data streams per-user occur ($\mathbf{L}^m = [2\,,1\,,1\,,1\,,1\,,1\,,1]$ for example), the exhaustive search over all possible orders causes a particular problem as up to $n_T!$ (7! in this case) calculations of (2.16) occur for a given $\mathcal{S}_T$ and $\mathbf{L}^m$. The computational complexity of the exhaustive search in terms of FLOPS and operations required for all of the necessary SVD and waterfilling operations is discussed in greater detail in section 5.5.3. This layered, combinatorially explosive problem posed by the stream projection and the iterative nature of (2.16) in environments with realistic $|\mathcal{S}|$, n_T, and n_R creates an intractable exhaustive search and necessitates the use of reduced-complexity search algorithms.

2.3 Related Work

In [44] the authors showed that in the single user MIMO case employing linear receivers, multimode selection (allowing any number of substreams to be dynamically selected) dramatically increased diversity gain even in the presence of limited feedback. Similar research of multiuser variable resource allocation in nonlinear precoding methods like vector perturbation has shown similar results, particularly in [42] where the authors demonstrate that

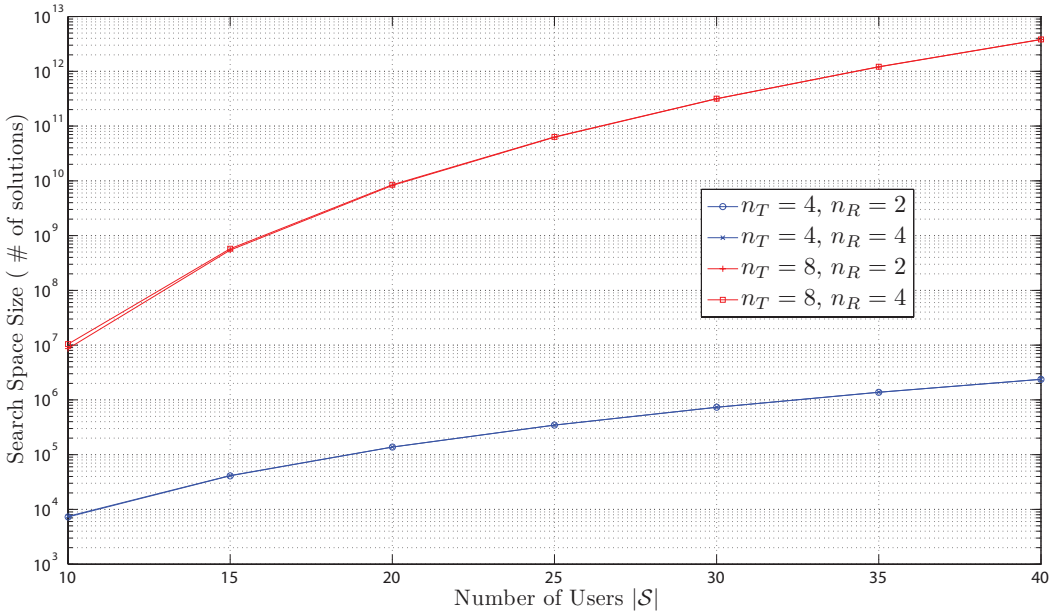

Figure 2.3: Size of solution space for exhaustive maximal eigenmode data stream allocation under SZF

the effective SNR can be increased (significantly increasing the throughput or decreasing bit error rates) by allowing for asymmetrical resource allocation between users being serviced. In research extending coordinated beamforming with BD by Chae et. al. [50], the authors show that even in the presence of quantization error in the CSIT due to limited feedback, variable allocation of resources to users via coordinated processing or receive antenna selection is able significantly outperform schemes not exploiting multi-stream diversity.

An optimal encoding scheme for MIMO OFDM channels was discussed in [43], where the authors developed an optimal transmit covariance matrix computation method that decomposes the vector channels of all users into orthogonal scalar ones without loss of capacity. Furthermore, they proposed a suboptimal iterative successive method of constructing the transmit covariance matrices such that they were projected onto the nullspace dimension with the largest singular value achieving near optimum sum-capacity performance.

Simplified heuristic and greedy scheduling algorithms for BD and SZF were investigated in [51, 52, 41], however no coordinated transmitter-receiver processing was assumed and only a fixed n_R data streams per user were assumed to be transmitted. The authors noticed that in circumstances with low transmit power, the waterfilling approach used to allocate

power would sometimes allocate power to a fraction of the possible modes for a given user, leading them to conclude that in some cases the allocation of fewer than the maximum possible data modes for each user can result in better sum rate performance.

In [35] the authors investigate variable data stream transmission using a technique termed "multiuser eigenmode transmission" or MET under BD and show that selective transmission of variable number of data streams to a user via coordinated processing not only outperforms transmitting on all possible eigenmodes for each user, but also selective transmission via receive antenna selection. Similarly, in [45] a low complexity scheduling algorithm for multiuser eigenmode transmission schemes for users precoded with BD was discussed, and subsequent results showed performance scaling matching that of DPC. Of particular interest related to our research was the result that showed little performance loss in scheduling only on the dominant modes of the channel as opposed to searching over all possible modes of each user. To the best of our knowledge however, no work on heuristic schedulers for variable data stream allocation with coordinated transmitter receiver processing under SZF has been published.

Chapter 3

System Model and Simulation Environment

3.1 System and Channel Models used for simulation

In our research we are interested in the performance of the scheduling algorithm and not system specific implementation such as channel coding or modulation. We therefore assume the data rates for each user are upper bounded by the channel capacity and that each user requesting service has data queued for transmission. This is a realistic assumption, as channel coding with turbo codes or Low-density parity check (Gallager) codes are known to approach within 0.5 dB of capacity.

For all of our following simulations we consider a single cell MIMO system consisting of a transmitter with n_T antennas and a set of users each with n_R receive antennas requesting service $\mathcal{S}$. We assume rich, high scattering environments that result in Rayleigh fading with no line of sight signal component, resulting in channel gains on each transmit-receive antenna pair that can be modelled as i.i.d. circularly-symmetric Gaussian random processes with unit variance. The coherence times of the channels are assumed to be much larger than the symbol duration, resulting in channel gains that are relatively constant over one transmission interval, and allowing us to assume block-fading channel model. This is a reasonable assumption as the current specifications for Long Term Evolution advanced (LTE-A) and 3GPP Release 12 E-UTRA [53] support bitrates with transmission intervals on the order of milliseconds It is also assumed that the channels fade independently between

each transmission interval. For simplicity, we do not assume any path-loss effects, or large scale fading or shadowing, instead assuming a ring of users distributed at equal distance from the transmitter in a high scattering Rayleigh fading environment with no direct line of sight signal component. It is assumed that the base station has full and perfect knowledge of all channels (full CSIT) for all users at each transmission interval. In time division duplex systems (TDD) this can be achieved by deriving the downlink channel based from the users received signals on the uplink by using channel reciprocity, or in in frequency division duplex (FDD) systems through the use of sounding or pilot tones broadcast from the base station which are then sent to the base via a feedback channel. Instead of a per-antenna power constraint which is motivated by transmission systems where arrays are powered by separate amplifiers, for simplicity we consider only the typical sum-power constraint (SPC) also used in other simulations of these types.

3.2 Cluster Computing Simulation Environment

The majority of the simulations were carried out using the cluster computing resources provided by Westgrid and Compute/Calcul Canada. Simulations were run on the Checkers cluster located at the University of Alberta. The Checkers cluster is an SGI Altix XE320-based machine with 160 nodes featuring 2 x LE-5420 (2.50GHz) 4-core processors (1280 cores total) and 16 GB of memory, 240 nodes featuring Xeon X5675 2 x (3.06GHz) 6 core processors (2880 cores total) nodes and 24 GB of memory, with all of the nods running the Scientific Linux operating system. A Lustre parallel distributed file system is attached to the Jasper nodes via the InfiniBand interconnect. It consists of an SGI IS16000 disk array with 250 drives, and provides a single 356 TB filesystem which was used for the storage of simulation data. The simulations were submitted using the portable batch system (PBS) scheduling software. Compiled MATLAB binaries were run with the random seeds of each simulation being generated based on current system time and unique identifiers associated with each node. Additionally, these seeds were checked for duplicates against a database of previously used seeds before runtime as to provide a *unique* pseudorandom (yet repeatable if the need arose to verify results) user and channel generation scheme.

As an interesting footnote, due to the large number and time-consuming nature of the simulations necessary for this thesis it is worth to mention that an adaptive job submission

script was developed for the PBS system. In this adaptive scheduling script, the N_{samp} total Monte Carlo iterations would be split on-the-fly to fit into currently available and upcoming free nodes and time blocks. This system allowed us to make maximal use of the cluster and avoid the 1-3 day waiting periods that the jobs would wait to run when submitted without adapting the run times to the current cluster load, as the short simulations could be used to fill gaps in the scheduler when the cluster load was high and therefore ran faster.

Chapter 4

Optimization Techniques

In this chapter we first examine the general optimization problem, and metaheuristic algorithms and their applications for solving non-convex problems. We then review the general simulated annealing algorithm, detailing the different components of this approach, as well as several variations on the traditional techniques. Our simulated annealing algorithm for user and data stream scheduling under SZF is then provided, and we detail the implementation and specifics of our approach.

4.1 Introduction to Optimization

Optimization in the practical sense can be described as finding the best available solution to some objective function given a defined domain.

Optimization problems can then be described as the task of finding an optimal configuration ω_{opt} from a finite or infinite countable configuration space Ω given a quality metric $R(\omega)$. The general discrete optimization problem can be presented by the following:

- given a function $f : \Omega \rightarrow \mathbf{R}$, mapping some discrete set Ω to the set of real numbers
- find an optimal solution ω_{opt} such that
  - $f(\omega_{opt}) > f(\omega) \forall \omega \in \Omega$ for maximization problems
  - $f(\omega_{opt}) < f(\omega) \forall \omega \in \Omega$ for minimization problems.

where f is an objective function. In contrast to continuous optimization problems where the constituent variables compromising each solution ω can take any valid real values, the

variables x_i used in the case of discrete optimization problems, where $\omega \triangleq (x_1, x_2, , \ldots, x_n)$, are discrete integer values each representing different subcomponents that compromise a solution.

One of the most basic optimization techniques is hill-climbing, or local search. A random starting solution ω_0 is chosen, and adjacent solutions ω' defined by a neighbourhood $\mathbf{N}(\omega)$ are searched for solutions that offer an improvement in the objective function $R(\omega)$. Only improving solutions with greater utility (or lesser cost, depending on whether maximizing or minimizing) are chosen, and the algorithm eventually hones into a locally optimal solution where none of the neighbours defined by $\mathbf{N}(\omega)$ offer an improvement in solution quality. It should be clear that the topology, or statistical distribution of the utility function values, of the solution space, the initial solution ω_0, and the scope or number of solutions reachable via the neighbourhood function $\mathbf{N}()$ in one jump, will determine the effectiveness of the algorithm. While very simple to implement, the primary drawback of hill-climbing algorithms is that they routinely suffer from getting trapped in locally optimal solutions. Increasing the search scope of the neighbourhood function can be used to combat this, however this increases the computational complexity and only guarantees convergence if a neighbourhood scope is equal to of the total search space, effectively transforming hill-climbing into an exhaustive search.

It is noted for future reference that while in general the objective function can be a utility function (for maximization problems) or a cost or energy function (for minimization), we will be referring to optimization in the sense of *maximizing* a utility function for the remainder of this thesis.

4.2 Metaheuristic Optimization Methods

Metaheuristics are methods for iteratively finding solutions to discrete, combinatorial, non-deterministic polynomial-time (NP) hard problems, where the search space of candidate solutions grows more than exponentially as the size of the problem increases. To avoid the problem of getting stuck in local optima that is usually associated with hill-climbing approaches, metaheuristic optimization methods can be viewed as modified hill-climbing algorithms that offer a mechanism(or mechanisms) to escape local optima in the intermediate stages of their execution. Leveraging local improvement procedures and higher level

strategies, they have been designed to create processes that escape local optima and can perform robust searches of the solution space. While not guaranteeing optimal performance, they generally manage to achieve better results and faster convergence performance than their classic optimization counterparts. The inspiration for many optimization methods often comes from biological or physical systems, which tend to offer peculiar insight into finding relatively good solutions to complex problems quite quickly with seemingly simple structures. Some methods with roots in biological systems (Biologically inspired computing) include: ant colony optimization, swarm intelligence, and genetic algorithms.

Ant colony optimization is a technique based on swarm intelligence where many simple agents interact locally with each other and the environment. These agents have no centralized control strategy dictating their individual behaviour, but instead follow relatively simple rules which lead to the emergence of intelligent global behaviour from the agents seemingly random individual movements and interactions. It has been used with success in various problems in other areas of optimization, as well as in communications in areas of routing and scheduling in wireless systems [54].

Genetic algorithms (and their parent class of Evolutionary Algorithms) mimic natural evolution by maintaining a pool of solutions and create new solutions by mutating and recombining existing solutions from the gene (or candidate solution) pool. Through successive mutations and evolutions over several generations, the genetic algorithms tend to evolve near optimal solutions relatively quickly. Genetic algorithms are fairly popular and have been used extensively in communications to help solve optimizations problems in everything from hardware and network design[55, 56], to user scheduling [52, 57, 58].

In general, metaheuristic approaches tend to perform well due to their stochastic nature and robustness against the objective function being maximized. They work well for problems with large solution spaces because few or no assumptions are made about the problem being optimized, and the stochastic nature helps ensure that they are not trapped in local optima and converge towards the optimal solutions. While not guaranteed to achieve the globally optimal solution, with proper implementation most metaheuristic optimization algorithms tend to converge quite quickly to solutions close to the global optima [59].

4.3 Overview of Simulated Annealing

In our research, we examine a particular metaheuristic algorithm that models itself after the physical process of metallurgical annealing. Simulated annealing (SA)[60] can be classified as a general probabilistic metaheuristic approach for the optimization of combinatorial (non-deterministic polynomial-time NP hard) problems. The inspiration and name behind SA comes from the metallurgical process of annealing, in which a substance is heated and then cooled in a controlled manner, allowing the crystal structure of the material to reconfigure into a lower, more stable energy state. Whereas metallurgical annealing tends to find the lowest energy configuration of a material, SA can be easily applied to both minimization and maximization problems. During the cooling process the atoms of the substance drift through higher energy states and are given time to find a more optimal, lower energy state than the initial one thereby reducing defects in the material. Furthermore, if the cooling process is sufficiently slow the final configuration is one that has is in a minimum lattice energy state and has superior structural integrity [61], and can converge asymptotically to the globally optimal solution as the number of iterations is increased[61].

Similar to other metaheuristic approaches such as Tabu search [62](a method similar to SA incorporating a blacklist of previously traversed solutions to avoid revisiting old solutions) or genetic algorithms [59, 63] (in which populations of solutions are 'bred' for successive generations), SA relies on a stochastic approach to optimizing a problem by iteratively improving candidate solutions with respect to a quality metric. Like other metaheuristic algorithms, it implements a mechanism to escape locally optimal solutions. To this end, SA employs a Metropolis criteria [64] which accepts suboptimal solutions with decreasing probability as the algorithms execution progresses. Therefore, by decreasing the probability of it becoming stuck in local optima, the acceptance of worse solutions during the course of the execution can be considered the fundamental strength of SA (and metaheuristics in general), and allows the algorithms to achieve a more extensive search over the solution space.

The basic execution flow of SA can be summarized by the following points:

- At each iteration of the algorithm given a current solution, a candidate solutions based on the current solution is generated via a neighbourhood function.

- The objective function values for the newly selected candidate are calculated and compared to the current solutions objective function value.

- Improving solutions are always accepted, whereas a fraction of the non-improving solutions are also accepted in the hopes of escaping local optima. For non-improving solutions, the probability of accepting inferior solutions is dependant on the temperature parameter and the relative difference in solution quality between the current and candidate solutions.

- The temperature is decreased in a controlled manner as the execution of the algorithm progresses (mimicking the cooling in the metallurgical annealing process), until eventually only improving solutions are accepted.

- Execution progresses until a predetermined number of iterations are reached, or a stop criterion based on relative change in solution is reached.

While improving solutions are always accepted regardless of the temperature, as the system cools and temperature decays, the probability of disregarding worse candidates increases. The resulting behaviour of the algorithm is such that when the temperature is high at the beginning of the annealing process, almost any solution regardless of relative solution quality (improving or non-improving) will be accepted. This leads to the selection of primarily improving solutions towards the end of the execution of the algorithm. The implementation of this structure allows the wide exploration of the search space with a coarse search at the start of the execution, transitioning towards exploitation of local solutions (hill-climbing) with a finer search.

SA can be thought to bridge the thermodynamic behaviour of annealing with the discrete optimization problem, providing a way to exploit the physical process of annealing for algorithmic optimization. We note that in reference to optimization, the goal of minimizing a cost metric is analogous to maximizing a utility metric. While we examine SA for the purposes of maximization, it is very easily be extended to minimization.

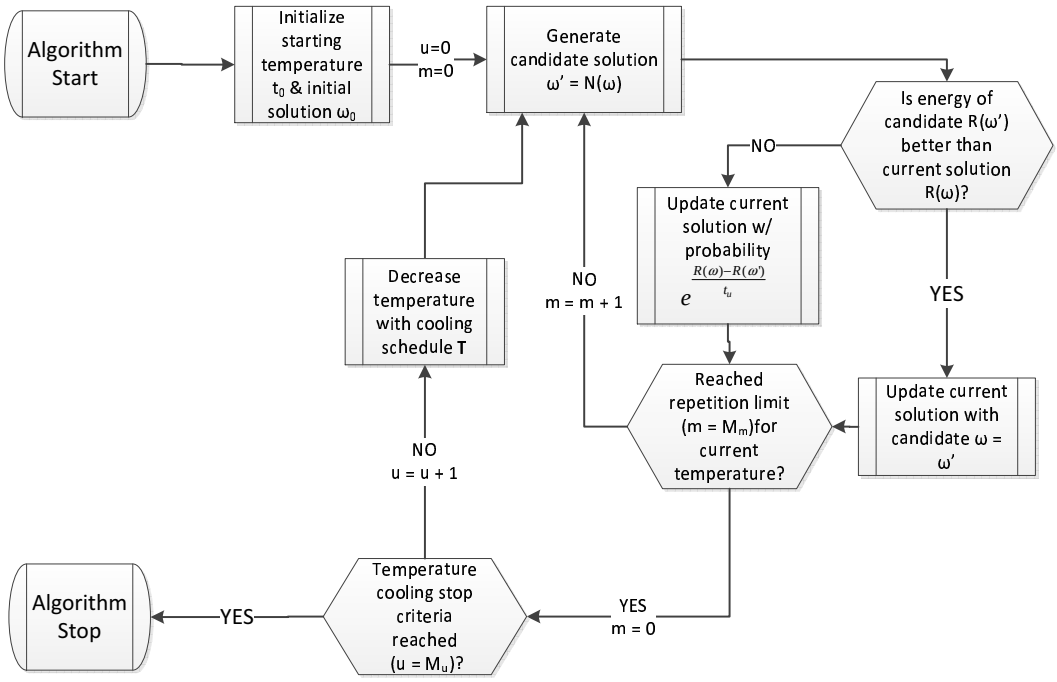

Figure 4.1: General SA algorithm flowchart for maximization

4.4 Functional Description of Simulated Annealing

In SA, each possible solution ω in the set of all possible solutions Ω can be equated to the state of a physical system, with the corresponding energy function $R(\omega)$ being the metric we wish to optimize. The goal of the SA process is then to bring the system from some arbitrary initial state ω_0 and a hot temperature t_0, to a final temperature t_f and the optimal configuration ω_{opt}, where $R(\omega_{opt}) > R(\omega) \forall \omega \in \Omega$. This is achieved via the interaction of several component parts of the algorithm including: an acceptance function P_A, a cooling schedule **T** (including initial and final temperatures, (t_0 and t_f respectively), a neighbourhood function $N()$ which controls the algorithms traversal through the search space, and the definition of a global search space Ω which defines all possible valid solutions ω.

The flowchart in Fig. 4.1 details the execution flow of the algorithm. At each outer step or iteration u of the algorithm, from the current solution ω, a candidate solution $\omega' = N(\omega)$ is generated using a neighbourhood function $N()$. This neighbourhood function generates new solutions by manipulating the current solution in some predefined way. The energy function for the candidate solution $R(\omega')$ is then compared to the energy of the current

solution $R(\omega)$ and accepted with probability given by the function $P_A()$. This acceptance function, given by

$$P_A(\Delta R(\omega, \omega'), t_u) = \begin{cases} e^{\frac{\Delta R(\omega, \omega')}{t_u}} & \Delta R(\omega, \omega') < 0 \\ 1 & \Delta R(\omega, \omega') \geq 0 \end{cases} \tag{4.1}$$

employs an adaptation of the Metropolis-Hastings algorithm acceptance criterion [64], and is a function of the current temperature (t_u) at outer iteration u and the difference in energy

$$\Delta R(\omega, \omega') = R(\omega') - R(\omega) \tag{4.2}$$

between the candidate solution and the current solution. It should be noted that improving solutions ($\Delta R(\omega, \omega') = R(\omega') - R(\omega) \geq 0$) are always accepted, whereas the non-improving candidates with $\Delta R < 0$ are subject to an exponential acceptance rate. It can be easily seen for non-improving solutions that because of the exponential nature of Eqn. 4.1, the higher the relative difference in utility functions $\Delta R()$ and the lower the temperature, the lower the resultant probability and the more unlikely the transition. Therefore as the temperature decreases during the course of the execution of the algorithm, the likelihood of accepting suboptimal candidate solutions (and jumping out of local optima) decreases, and transitions the algorithm into a hill-climbing phase that tends to only accept improving solutions.

These m inner steps (m denoting the m'th inner repetition) of the algorithm are repeated until an inner loop repetition criterion M_m is reached, upon which the outer loop parameter u is incremented and the temperature value is updated according to the cooling schedule defined by

$$\mathbf{T} = [t_0,\ t_1,\ \ldots,\ t_u,\ t_{u+1},\ \ldots,\ t_f]. \tag{4.3}$$

These inner loops are then repeated for each u until the outer loop stop criterion M_u is reached, or a stop criteria is satisfied (such as the incremental change in utility function in the last several iterations not increasing by a prespecified amount, or any other problem-specific stop criteria [59]). The system is then said to be in a "cooled" state and the execution is terminated.

While the acceptance function $P_A()$ is relatively robust and problem invariant, the choice of the cooling schedule $\mathbf{T}$ and corresponding initial and final temperatures (t_0 and t_f), the inner and outer loop repetition criteria M_m and M_u, and neighbourhood function $N()$

present many problem-specific issues which must be dealt with in the design of the algorithm. Specifically, due to the sizeable influence of these parameters on performance and their problem specific nature, improper selection can have a significant impact on the algorithms performance in terms of both solution quality and convergence. While there exists no good choice of above parameters that will perform well for all problems, there are some guidelines which can be followed that can help facilitate better performance.

4.4.1 Neighbourhood Function $N()$

Given any arbitrary starting point ω_0, the optimal solution ω_{opt} should be reachable by any number of subsequent manipulations, the sum of which correspond to a path through Ω. While the performance of the algorithm no doubt depends also on the other components, the neighbourhood function $N()$ should be considered one of the integral parts as its primary function of manipulating the current solutionω has a very large influence on the behaviour of the algorithm during its exploration of the solution space Ω. Regardless of the problem-specific details of any individual implementation, the general idea behind $N()$ is that it randomly generates candidate solutions via a probabilistic, structured manipulation of a current solution.

We define the set of solutions

$$\Omega_N = \{\omega_N | \omega_N \in N(\omega)\} \tag{4.4}$$

to be the set of all neighbours of ω noting $\omega \notin \Omega_N$. The probability of generating a particular solution ω' from ω is then given by $g_k(\omega, \omega')/g(\omega)$, where $g_k(\omega, \omega')$ is the weighting given to the transition at iteration k and $g(\omega)$ is the normalization function such that

$$\frac{1}{g_k(\omega)} \sum_{\omega \in \Omega_N} g(\omega, \omega') = 1. \tag{4.5}$$

In many traditional implementations of SA, the probabilities of generating any particular neighbouring solution are equal, given simply by $g_k(\omega, \omega') = \frac{1}{|\Omega_N|}$, $|\cdot|$ denoting the cardinality of the encapsulated set. This approach works well, but there no doubt exist certain problems in which benefits can be gained by biasing the values of $g_k()$ with advanced statistical knowledge of the solution topography, steering the algorithm towards solutions with characteristics that are known to offer good performance [65, 66].

The neighbourhood function $N()$ must also balance the need for exploring the global search space and exploiting local solution areas to obtain the best solutions possible. If the one-hop neighbourhood Ω_N (or the number of solutions reachable in one neighbourhood transition) is too large ($|\Omega_N|$ on the order of $|\Omega|$), the function $N()$ essentially reverts to a random search with the next state chosen uniformly over Ω. This case may require impractical execution times and cause the algorithm to miss the opportunity to exploit the local characteristics of the search space. Conversely, a neighbourhood that is too small will not allow the algorithm to sufficiently explore the search space in a reasonable amount of iterations, and will impact the convergence and overall performance. In practice it has been shown in [59, 65] that in order to facilitate relatively consistent performance, $N()$ should be chosen in such a way that the distribution of the energy's $R(\omega)$ in the neighbouring solutions Ω_N do not exhibit particularly large variances.

4.4.2 Temperature and Cooling Schedules Overview

Recall from Eqn. 4.3 where we define a cooling schedule $\mathbf{T}$ by a starting temperature (t_0), final temperature (t_f), and a cooling profile that dictates the way the temperature decreases. In SA, as with the physical process of annealing, it is assumed that the cooling rate is slow enough as to allow the current state to settle into a thermodynamic equilibrium (exhibiting relatively few state changes) at each temperature. This requires maintaining an adequate relaxation time at each temperature, or the amount of time or inner repetitions m required at each temperature t_u for the system to reach a steady state (or relatively low energy level transitions). The design of the cooling schedule should also be done in concert with the other component parts of the algorithm. For example, if the cooling is done quickly and the steps between subsequent temperatures are relatively large ($t_{u+1} << t_u$), the neighbourhood function $N()$ should be structured in such a way as to offer a more variation in solution 'distance', to compensate.

Many different methods and schools of thought exist for choosing cooling schedules and temperature profiles for SA[59]. One approach commonly used is where the initial and final temperatures are obtained from pre-known statistical information of the solutions and then cooled using a fixed, monotonically decreasing schedule. Simple, monotonically decreasing cooling profiles that have worked in practice [63, 67] include linear (Fig. 4.3a), geometric

(Fig. 4.3b), and exponential (Fig. 4.3c).

More complex, adaptive methods [68] also exist in which the temperature profiles are adjusted on-the-fly based on decision-time information gathered during the execution of the algorithm. In contrast to the monotonically decreasing cooling schedules adapted from the physical process of annealing, research into techniques [69, 70, 71] that do not follow a simple decreasing temperature profile has shown promise. The logic behind a reheating

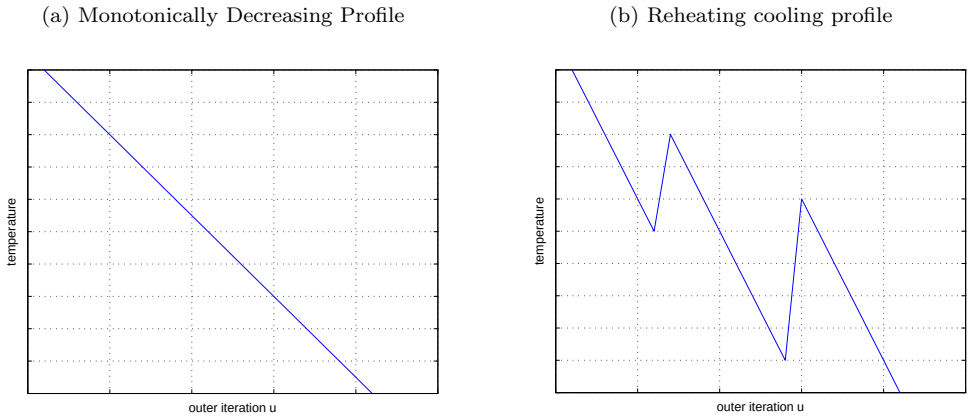

Figure 4.2: Example of monotonically decreasing and reheating cooling profiles

profile such as one shown in figure 4.2 is that the intermediate reheating of the temperature can allow the algorithm more freedom to explore the search space and jump out of solution areas in which it may have otherwise gotten stuck in, thereby allowing the selection of a better solution in the long run. However, these methods are inherently more complex than their monotonic cooling counterparts as reheating profiles must also be tuned in addition to the 'hot' and 'cold' temperatures and cooling profiles.

While adaptive temperature control approaches like those used in [68, 72, 73] may offer better performance in some cases at the cost of additional complexity and structure, many simpler implementations [67, 74, 75] of SA employing simple, monotonically decreasing cooling schedules yield similar results. Simple, reduced complexity cooling schedules based on theoretical analysis, and requiring little or no tuning, were suggested in [67, 76], and showed that choosing the correct t_0, t_f and a proper cooling 'shape' allows for faster convergence.

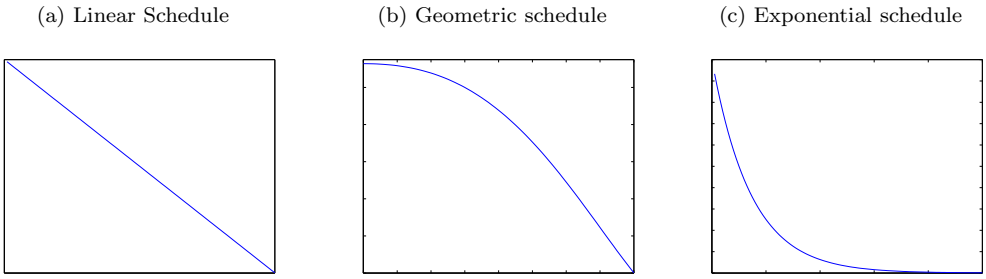

Figure 4.3: Various types of cooling schedules for SA

4.4.3 Tuning Initial and Final Temperature Values

Regardless of the chosen temperature profile, it should be obvious from the structure of the acceptance function $P_A()$ given in (4.1) that the value of the initial and final temperatures (t_0, and t_f) will depend on the distribution of the rate differences ΔR, as well as the neighbourhood function. The cooling schedule $\mathbf{T}$ will therefore be problem-specific, and will also depend on the design choices made in the other parts of the algorithm.

Given a particular distribution of ΔR, t_0 is usually chosen to allow the majority of *all* (improving, and non-improving) transitions generated by the neighbourhood function $N(\omega)$. It is generally accepted to t_0 such that $P_A()$ will accept 80-90% of all possible transitions at the start of the algorithm execution, although more theoretical approaches to setting the initial acceptance criteria based on thermodynamic equilibrium of the system exist and are discussed by White et. al in [77]. Adaptive methods that do not require any preset values but guess and adjust the temperatures using data obtained throughout the execution have also been examined in [68]. Barring the implementation of an adaptive temperature control methods mentioned above, some advanced knowledge information on the topology of the solution space must be known in order to choose an effective t_0.

One approach [59] to determine the distributions is to take 'walks' using the neighbourhood function and randomized starting points and record all of the resulting absolute differences in utility function, ΔR. These empirically obtained values are used to form distributions of the possible ΔR's that occur from neighbourhood transitions generated by $N()$ in a solution space Ω. The distributions of these ΔR's are then used to set the initial temperature accordingly to accept a certain percentile of the possible transitions(a more detailed description of the selection is offered in section 5.2.2). Similar yet simpler methods

for finding initial temperatures exist and are also used, offering good performance in some cases; for example in [78] where the authors proposed an alternative to recording distributions in which only the single maximal absolute value of the difference is recorded, scaled, and subsequently used for the initial temperature calculation.

The final temperature t_f or the "cooled state" of the system is set to accept only improving solutions and to force the algorithm to seek out the best local solution, shifting the algorithm from an exploration phase to an exploitation or hill-climbing one. While choosing $t_f = 0$ would be a logical choice, the temperature at which the acceptance probability will, for all practical purposes, stop accepting suboptimal solutions, be some nonzero value and a function of the ΔR distributions. Therefore the choice of final temperature is usually done in similar fashion as the method for finding the starting temperatures, where the distribution of ΔR is gathered via random walks through the solution space and the final temperature set by choosing a low acceptance probability ($< 1\%$).

4.4.4 Inner and Outer Loop Stop Parameters

The repetition limits M_m at each temperature t_u (or equivalently outer repetition u) are usually selected to establish its steady state at each temperature. The outer loop repetition limit on the other hand is usually set in concert with the cooling schedule as to adequately allow the temperature to reach its final state. However, it is not unusual for it to be subject to additional stop criteria. In order to help bound the finite-execution time of the algorithm, a watchdog can placed on the solution to terminate the execution of the algorithm if no improved solutions have been found over some number of iterations, or if the level of improvement in the solutions has plateaued or not increased by a pre-specified amount (usually over some last number of iterations).

4.5 An Approach to Data Stream and User Scheduling with Simulated Annealing

For the purposes of our algorithm and simulations, we assume the following: a single cell MIMO broadcast channel formed by a base station with n_T antennas is serving a set of $\mathcal{S}_T$ (where $\mathrm{K} = |\mathcal{S}_T|$) out of a total set of $\mathcal{S}$ users each with n_R antennas. All of the users in $\mathcal{S}$

are assumed to have data queued at the transmitter. It is assumed that the channel state information $\mathbf{H}_k$, of each user k, is known perfectly at the transmitter. SZF precoding along with coordinated transmit-receive processing under a sum-power constraint is employed at the transmitter, allowing a maximum of n_T data streams to be transmitted at once, with n_R maximum possible data streams per user. Recalling $\mathbf{L}$ as the set of integer partitions of n_T, we denote $\mathbf{L}^m = [l_1, l_2, \ldots, l_K]$ as the m-th unique integer partition of the n_T data streams, with l_k denoting the number of data streams for the k-th encoded user, satisfying $\sum l_k = n_t$ and $l_k < n_R, \forall k$. The sets of $\mathbf{L}$ for the antenna cases considered in our simulations are listed in Table 2.1. $\pi = [\pi_1, \pi_2, \ldots, \pi_K]$ is defined to denote the encoding order, with $\pi(i) = \pi_i$ representing the i-th encoded user and $\bar{\pi}$ representing the set of all $K!$ possible orderings π of a given K users and their corresponding data stream allocations $\mathbf{L}^m$. We define an additional user subset, the candidate pool $\mathcal{S}_C$ as a subset of the global user pool $\mathcal{S}$ with users that can be considered by the algorithm for transmission, $\mathcal{S}_T$, noting

$$\mathcal{S}_T \subseteq \mathcal{S}_C \subseteq \mathcal{S}. \tag{4.6}$$

We also define a candidate pool update function $N_C()$, and detail its function, and that of the candidate pool, in greater detail in section 4.5.6 below.

4.5.1 Solution Definition and the Global Solution Space

A solution ω is defined as a particular stream allocation $\mathbf{L}^m$, user subset $\mathcal{S}_T$ and order π. Consider the case illustrated in Figure 4.4 where users 1,7, & 8 have been scheduled for transmission with 2 data streams being allocated to user 7, followed by 1 data stream each to users 8 and 1. This particular solution ω consists of the order vector $\pi = [7\,8\,1]$, and the stream allocation vector $\mathbf{L}^m = [2\,1\,1]$.

The complete solution space Ω is defined then as all unique combinations of b_{n_T} possible stream allocations $\mathbf{L}$ for that combination of n_T and n_R, projected on to all possible user subsets $\mathcal{S}_T$ of $\mathcal{S}$ and all possible subsequent orderings $\bar{\pi}$.

We define the utility function $R(\omega)$ as the achievable sum-rate (Eqn. 2.16) with that particular set of streams $\mathbf{L}^m$, users $\mathcal{S}_T$, and order π, although simpler metrics could also be used. Our use of the aggregate sum-rate can be described as the use of a weighted sum-rate $\sum_{\forall k} W_k R_k$ with all of the K users having an equal nonzero weighting $W_k = \frac{1}{K}$, and can be

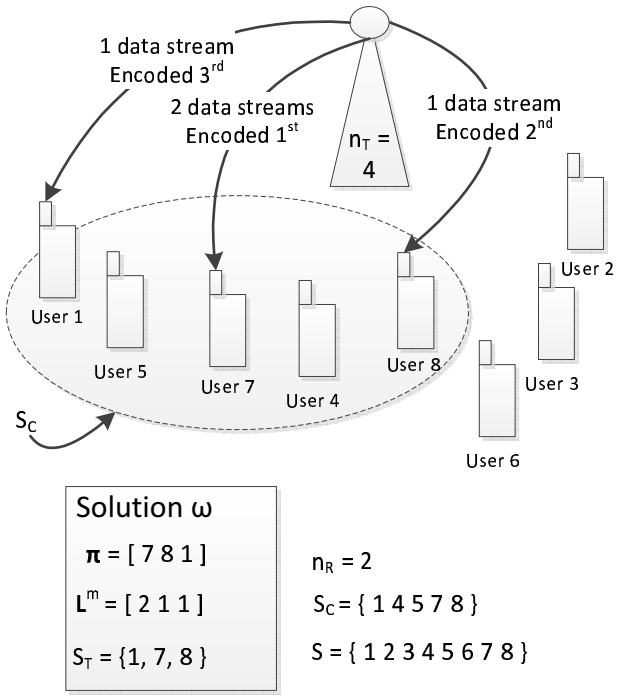

Figure 4.4: Solution structure for data stream allocation

extended to include fairness by weighting the sum-rate by the appropriate factors.

4.5.2 Algorithm Description and Pseudo-code

The following pseudo-code algorithm describes our approach to user and data stream scheduling with SA.

Initialize: random initial solution $\omega = \omega_0$, $\mathcal{S}_C, \mathcal{S}_T, \mathbf{T}$

temperature change count $u = 0$

define repetition limits M_u, M_m

$\omega_{\text{best}} = \omega$

while $u \leq M_u$

inner loop repetition counter $m = 0$

update $\mathcal{S}_C = N_C(\mathcal{S}_C, u)$

while $(m \leq M_m)$

$\omega' = N(\omega)$

$\omega \leftarrow \omega'$ updated with probability $= P_A(\omega, \omega')$

update $\omega_{\text{best}} \leftarrow \omega$ iff $R(\omega) > R(\omega_{\text{best}})$

$m = m + 1$

end

$u = u + 1$

update t_u

end

final (limited) ordering step on the solution ω_{best} for final user selection

The final limited ordering step performs a search of up to 50 possible permutations on the best solution ω_{best}. In the case of $n_T = 4$, because the maximum number of scheduled users is $|\mathbf{L}^m| \leq 4$, $\forall m$ there exist a maximum of $4! = 24$ possible ordering, resulting in an exhaustive search over all possible orders of the users and their respective stream allocations in solution ω_{best}. However, in the case of $n_t = 8$ there can be up to 8 scheduled users in the allocation $\mathbf{L}^m = [1, 1, 1, 1, 1, 1, 1, 1]$ (and subsequently up to $8! = 40320$ possible orderings). In many cases this ordering step would exceed the total number of algorithm steps ($M_u \times M_m$) used in our simulations. Therefore in allocations where $|\mathbf{L}^m| \geq 5$ (120 possible orderings or more) only a random 50 of the total possible orderings are searched, with priority given to ordering the users with the largest singular values of their respective effective channels first.

4.5.3 Neighbourhood Search Function

Recall in Eqn. 4.4 where we define the immediate (one-hop) neighbourhood of the solution ω, Ω_C, as all possible solutions which can be reached by one operation of the neighbourhood function $N(\omega)$ on the solution ω. The neighbourhood function $N()$ used in our algorithm is comprised by the four following operations illustrated in figure 4.5, with each operation detailed below. Each time the function is called, one of these four neighbourhood transitions is randomly selected with probabilities p_u, p_a, p_o, chosen such that $p_o + p_u + p_a = 1$. The guidelines for selection, and effect on performance of these parameters, is discussed further in Section 5.2.2.

1. **Order Swap** p_o - randomly swapping the encoding order $\pi(a)$ and $\pi(b)$ of any two scheduled users (a and b) with probability p_o

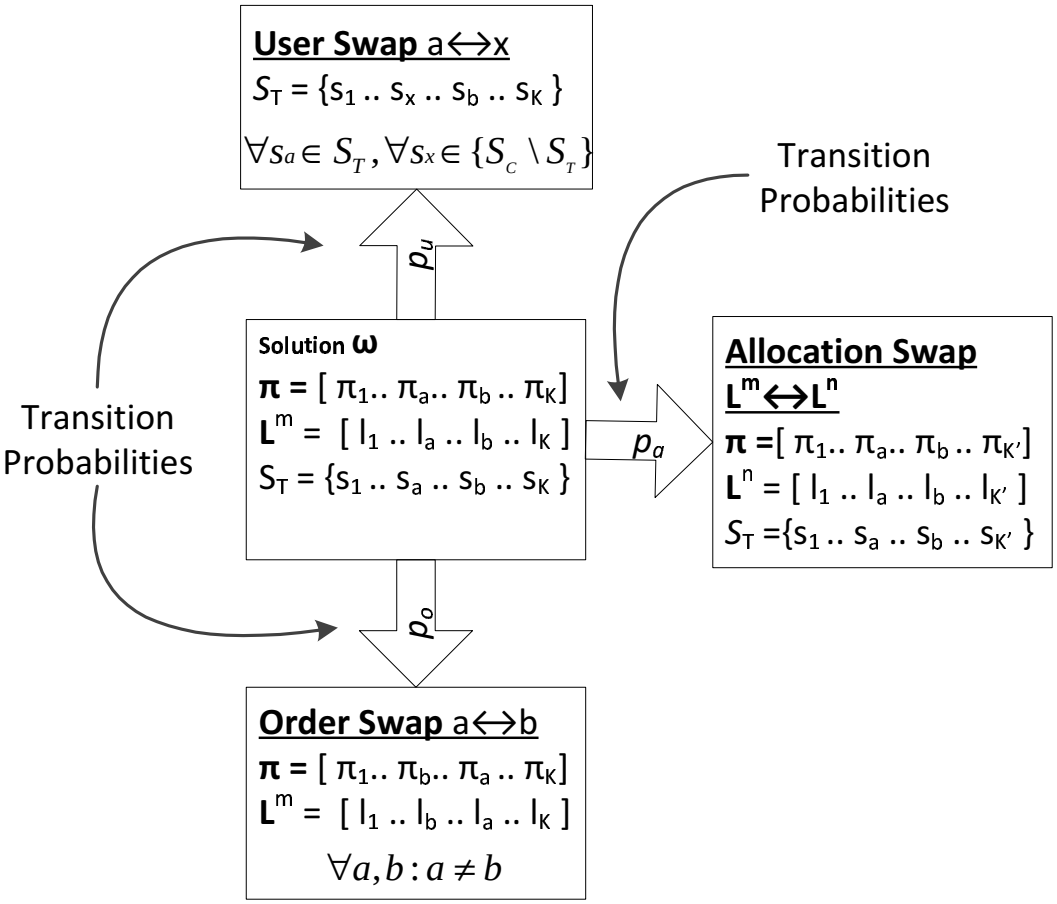

Figure 4.5: Neighbourhood transition definitions

2. **User Swap** p_u- randomly swapping a non-scheduled user from the candidate pool $\mathcal{S}_C$ with a currently scheduled user in $\mathcal{S}_T$ with probability p_u

3. **Allocation Swap** p_a - randomly swapping the stream allocation from $\mathbf{L}^m$ to $\mathbf{L}^n$ with probability p_a, (choosing one of the $|\mathbf{L}| - 1$ valid non-selected stream allocations with probability $\frac{1}{|\mathbf{L}|-1}$).

To clarify the Allocation Swap operation, it swaps the stream allocation $\mathbf{L}^m$ to $\mathbf{L}^n$ while retaining the encoding order based on the users with the maximal number of streams, modifying the new allocation in a way that requires the least number of data streams to be added or removed for each user. For example, in the case of $n_T = 4$ and $n_R = 4$, assume $\mathbf{L}^m = [1, 2, 1]$ and $\mathcal{S}_T = \{6, 3, 4\}$ and we wish to transition to an allocation $\mathbf{L}^n = [3, 1]$. Since the current solution encodes 1 data stream to the first user(user index 6), 2 to the

second (user index 3), and 1 to the last one (user index 1), the new solution is modified such that user with index 2 receives the largest number of data streams, and the resulting stream allocation is $\mathbf{L}^n = [1\,3]$ with $\mathcal{S}_T = \{6, 3\}$. This structure on the allocation swaps limits the 'distance' of each solution from the previous solution allowing us to search through the solutions in a more methodical manner, hopefully mitigating the possibility of losing a particularly well-performing group of users and stream allocations.

It should clear that given any two arbitrary solutions $\omega_1, \omega_2 \in \Omega$, our neighbourhood function $N()$ allows the solution ω_2 to be reached from ω_1 in a finite number of transitions defining a connected path from ω_1 to ω_2. This helps satisfy the suggested guidelines given in [59, 79] on structuring neighbourhood functions as to help ensure asymptotical convergence to the globally optimal solution in the limit as the inner (M_u) and outer (M_m) repetition limits increase.

4.5.4 Delta-rate Distribution Estimates

As mentioned in Section 4.4.2, the selection of the majority of the parameters in SA is related in some way to the distributions of the delta-rates (with $\Theta_{|\Delta R|}\,(|\Delta R|)$ denoting the probability density function, and $\Phi_{|\Delta R|}\,(|\Delta R|)$ denoting the cumulative distribution function). We do not take an adaptive approach to temperature, decay, and repetition loop parameter selection, rather employing a simpler technique with static temperature, decay, and repetition criteria parameters that are changed only during the initialization of the algorithm. However because of this, some advanced knowledge of the delta-rate distributions becomes a necessity. The behaviour of the algorithm in the sense of how it traverses the solution space is controlled by the neighbourhood function $N()$ (compromised of the neighbourhood transition probabilities) and the acceptance function $P_A()$ (and therefore the repetition schedules M_u and M_m, and the cooling schedule T). Theoretically, for a given user environment n_T, n_R, $|\mathcal{S}|$, the delta-rate distributions would also be a function of $N()$ and $P_A()$.

One possible approach to obtain distributions for a given set of SA parameters would be to take an educated guess at an initial estimate of the delta-rates, derive the appropriate parameters, run the algorithm, and then update the delta-rate guess based on the actual delta-rates observed during the execution. This process would be repeated in an iterative

fashion, updating the guess of the delta-rates until they converged to the actual distributions. While this would characterize the execution of the algorithm quite well, the problem with this iterative approach is that we would not be able to guarantee the convergence or the finite-time performance of this sort of method due to the already complex nature of our problem. An analytical approach to check for performance and convergence of an iterative approach would be impractical and is out of scope of this research. Barring an iterative approach, ideally we would like to have an accurate representation of all of the possible solution paths the algorithm could encounter to base the derivation of the other parameters on, and this could conceivably be achieved by modeling each possible transition path from all possible initial solutions and all subsequent solution branches. However, calculating all of these possible transition paths would be more complex than the scheduling problem, or the exhaustive search for that matter, as the utility function of each possible combination of paths would have to be evaluated and stored, requiring a large amount of computational power and enormous amounts of memory.)

To help simplify our algorithm and because a complete characterization of the solution space would be infeasible, we instead chose a simplified method in which we choose to generate many short paths and to record the value of the transitions, thereby forming an *estimate* of the actual distribution. The approach used is to take many random walks while accepting all transitions, and record the absolute value of all the resulting transitions $|\Delta R| = |R(\omega') - R(\omega)|$ as opposed to the actual value of $R(\omega) - R(\omega')$ (as one possible method mentioned in section 4.4.3 did). To clarify, we start from a randomly generated solution and generate neighbourhood transitions using $N()$, always accepting the transitions (analogous to having a 'super-heated' very high temperature t_0 and very low decay rate). For each new candidate $\omega' = N(\omega)$ the $|\Delta R|$ is recorded, and a path length of $300 - 500$ traversals are taken, forming an estimate of the ΔR distributions. This process is repeated until we have an acceptable estimate of the distributions (4000 samples of paths on independently generated random user channels were used in each case). The delta-rate distributions are given by $\Theta_{|\Delta R|}(|\Delta R|)$ and $\Phi_{|\Delta R|}(|\Delta R|)$, denoting the probability density function (PDF) and cumulative distribution function (CDF) respectively. We define

$$P_{a,u} = e^{(\frac{\Delta R_{0.9}}{t_u})} = e^{(\frac{\Delta R_{0.9}}{t_0 \alpha^{(u-1)}})}. \tag{4.7}$$

as denoting the probability of accepting a suboptimal solution for a given $\Delta R_{0.9}$ and starting

temperature t_0 after u repetitions. Through simulation, the factors that affected the distributions of $\Phi_{\Delta R}$ the most were found to be (in decreasing order): SNR, n_T, n_R, p_o, p_u, p_s, and $|\mathcal{S}|$. More analysis regarding the effects of the parameters on the delta-rate are given in section 5.2.2.

Therefore, at each n_T and n_R, a group of users $|\mathcal{S}| = 25$ with randomly i.i.d generated channels is generated and the following method is followed to obtain the estimate of the delta-rate distributions and obtain the appropriate SA parameters:

1. transition probabilities p_a p_o p_u are chosen

2. from a random initial solution ω_0 a path created by ≈ 300 subsequent neighbourhood transitions is taken. The candidates are generated via the neighbourhood function $N()$ with the transition probabilities p_a p_o p_u, and are carried out at an arbitrarily high temperature and low decay rate (effectively accepting all solutions). These delta-rate characteristics can be saved and do not need to be recomputed as long as the primary characteristics (n_T,n_R, and SNR of the simulation environment remain the same.

3. $\Phi_{\Delta R}$, the CDF of the distribution, is then used to obtain the upper bound ΔR_x which contains x-th percentile of the $|\Delta R|$ values. A working range of ΔR_x was determined experimentally to be $x = 0.85$ to $x = 0.90$.

4. Given a target initial acceptance probability $P_{a,1} = 0.90$ (set using guidelines from [59, 65, 66, 79, 80]), we then find the initial temperature

$$t_0 = \frac{-\Delta R_{0.9}}{\ln P_{a,1}} \tag{4.8}$$

5. Given a final acceptance probability threshold $P_{a,f} \approx 0.01$ (on guidelines from [79]) The required number of outer iterations $M_{u\,min}$ to sufficiently cool the system is then

$$M_{u\,min} = \frac{\ln(t_f)}{\ln(\alpha t_0)} \tag{4.9}$$

where

$$t_f = \frac{-\Delta R_{0.9}}{\ln P_{a,f}}. \tag{4.10}$$

To briefly clarify step 3 above, assume an arbitrary path from solution $\omega_1 \rightarrow \omega_2$ where each solution ω_{i+1} is generated from the previous ω_i, using $\omega_{i+1} = N(\omega_i)$ (with an arbitrarily

high initial temperature t_0 and decay factor $\alpha = 1$, thereby accepting all solutions). The CDF distribution of $\Delta R(\omega, \omega')$ for this path is $\Psi_{\Delta R,1-2}(\Delta R)$ (note: not $|\Delta R|$), is created by all of the transitions of the solutions from ω_1 to ω_2, $\Delta R(\omega_i, \omega_{i+1})$. As the path $\omega_1 \rightarrow \omega_2$ was generated using the neighbourhood function $N()$, it is also possible to, by definition, to generate the path $\omega_2 \rightarrow \omega_1$ (with corresponding CDF $\Psi_{\Delta R,2-1}(\Delta R)$) using the reverse (also valid) transitions. The values of $\Psi_{\Delta R,1-2}$ for $\Delta R > 0$ which represent the *improving* solutions observed on path $1-2$, are now the *non-improving* solutions observed on path $2-1$. The resulting $|\Delta R|$ CDF distribution for the path $1-2$ $\Phi_{|\Delta R|,1-2}$ is now also equal to the CDF of the reversed path $2-1$, $\Phi_{|\Delta R|,2-1}$. Therefore, the x-th percentile ΔR_x of $\Phi_{|\Delta R|,1-2} = \Phi_{|\Delta R|,2-1}$ now characterizes an estimate of the distribution of both non-improving and improving solutions of both paths. Through experimentation, we found the values of x for $|\Delta R_x|$ values which performed well in our simulations to be in the range $x = 0.85$ to $x = 0.90$.

By choosing $M_u > M_{u\,min}$ for a given α and T_0, we can allow the algorithm to run *past* the minimum required temperature or outer loop iterations u and force it to run in a 'super-cooled' state. Since in this super-cooled state the acceptance function $P_A()$ is accepting only better solutions, SA is said to run as a hill-climbing algorithm in this state. In [79] where the authors discuss application guidelines for simulated annealing, they argued that in certain cases it may be beneficial to allow the algorithm to run in this super-cooled state for some fraction of the total execution time ($c \times M_u$ where $c < 1$ outer repetitions executed with $P_{a,cM_{u\,min}} < P_{a,f}$). We experiment with the effects of super-cooled temperatures and detail our results in Chapter 5.

4.5.5 Cooling Schedule Parameter Selection

The cooling schedule $\mathbf{T}$ (see eq. 4.3) that we use is modelled after geometric schedules discussed in [63, 67] and in Section 4.4.2. In these types of schedules, an initial temperature t_0 is decayed by a constant α over the course of M_u iterations until reaching a final temperature t_f. The temperature at the u-th outer loop repetition can thus be given by

$$t_u = t_{u-1}\alpha = t_0\alpha^{u-1} \tag{4.11}$$

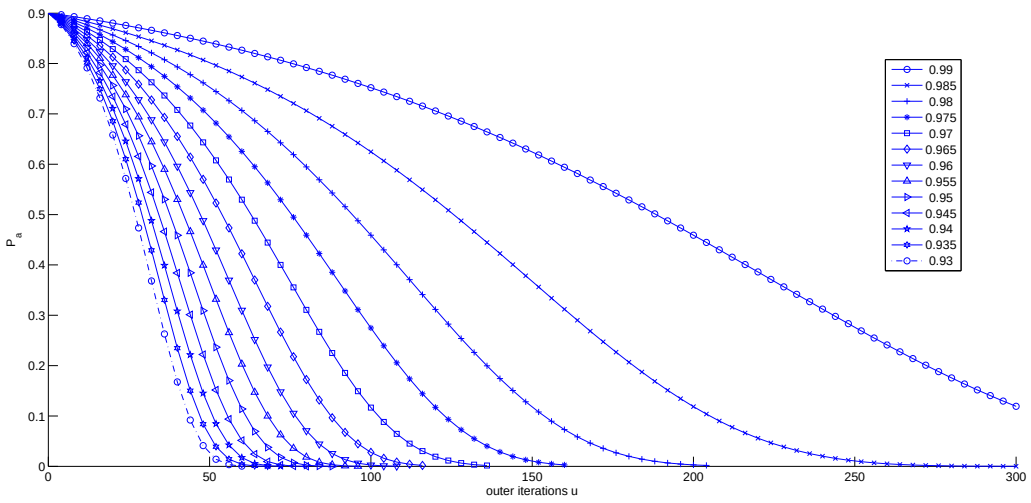

Figure 4.6: Acceptance probability for various decay rates

The effects of different decay rates on the $P_{a,u}$ are shown in figure 4.6, with the range of interest in our simulations being $0.93 < \alpha < 0.98$. The decay rates were chosen in this particular range because at these α's, the number of outer loop repetitions $M_{u,min}$ given by (4.9) required to achieve $P_{a,u} < 0.01$ gave us the flexibility to vary the inner repetitions M_m while still staying below a desired level of total complexity $M_u \times M_m \leq 10000$ (see table 5.1).

Based on the above design considerations, the outer repetitions chosen were $M_u = 100\,200\,300]$. It can be seen that at $M_u = 100, 200\,, 300$ for $\alpha < 0.95$, and $M_u = 200, 300$ for $\alpha > 0.98$, $P_{a,u}$ falls below the aforementioned 1% acceptance threshold . This was mentioned in the Section 4.5.4 as supercooling, and allows the algorithm to run as a hill-climbing algorithm, accepting only improving solutions for a small duration of its runtime. Table 4.1 shows the proportion of time spent in the super-cooled or hill-climbing state for various decay rates and outer loop repetition limits used in our simulations.

4.5.6 Candidate and Transmission User pools

At the beginning of each outer loop u, the candidate user pool $\mathcal{S}_C$ is updated using a candidate pool update function $N_C(\mathcal{S}_C, u)$, via the following steps:

1. the users not currently scheduled are placed into the set $\mathcal{S}_n = \mathcal{S}_C \backslash \mathcal{S}_T$ (where $\backslash$ denotes

		Proportion of execution in hill-climbing		
α	$M_{u,min}$ for $\Delta R_{0.9}$	$M_u = 100$	$M_u = 200$	$M_u = 300$
93	60	0.40	0.70	0.80
94	68	0.32	0.66	0.773
95	80	0.20	0.60	0.733
96	100	0	0.50	0.667
97	132	0	0.34	0.56
98	192	0	0.04	0.36
99	384	0	0	0

Table 4.1: Proportion of execution time in super-cooled state for different α and M_u

the relative complement, or users belonging to $\mathcal{S}_C$ and not $\mathcal{S}_T$)

2. $\lceil d_u \times \mathcal{S}_n \rceil$ randomly selected users in $\mathcal{S}_n$ are replaced by users randomly selected from $\mathcal{S} \backslash \mathcal{S}_n$ where the candidate pool update ratio d_u is defined by $\mathbf{d}_{upd} = [d_1\, d_2 \cdots d_u \cdots\, d_{M_u}]$, with $d_u \leq 1$, $\forall u$

3. the number of users in the candidate pool $\mathcal{S}_C$, $c_u = |\mathcal{S}_C|$, is updated according to a schedule $\mathbf{c}_{upd} = [c_1\, c_2 \cdots c_u \cdots c_{M_u}]$, with $c_i \geq c_{i+1}$. If the candidate pool is scheduled to be decreased at iteration u,$(c_u < c_{u-1})$, the appropriate number of users are randomly removed from $\mathcal{S}_n$ to satisfy $|\{\mathcal{S}_T \cup \mathcal{S}_n\}| = c_u$.

4. the new candidate pool for the current iteration u is updated, $\mathcal{S}_C = \{\mathcal{S}_T \cup \mathcal{S}_n\}$

The update ratio schedule $\mathbf{d}_{upd}$ is constructed in a similar manner to the temperature schedule $\mathbf{T}$, monotonically decreasing as $u \to M_u$ and facilitates an increasingly thorough search over solutions from a particular user subset as the algorithm progresses. This additional structure is added in the hopes of constraining the neighbourhood size during the M_m inner loops as to facilitate the adequate exploration of the local search space, yet at the same time allowing for the coarse exploration of Ω by switching off users.

However, in the antenna environments simulated (n_T, n_R); $(4,2)$ $(4,4)$ $(8,2)$, it was found that this additional structure was in fact detrimental to performance except for a few select cases where (n_T, n_R) (4,2) (4,4), $|\mathcal{S}| \leq 10$ and the total SA complexity ($M_u \times M_m$) was close to that of the exhaustive search (7000 iterations). Preliminary simulations done in uncorrelated channel environments with candidate pools and candidate pool update

schedules for larger numbers of SA iterations and $|\mathcal{S}| > 20$ yielded solutions with sum-rate that were 5-10% below the sum-rates achievable when no candidate pool was used ($\mathcal{S}_C = \mathcal{S}$). This is most likely due to the gains from multi-user diversity that are achievable by allowing the unfettered selection of users for transmission from the total user pool $\mathcal{S}$ outweigh the advantages created by performing a more complete search over the possible stream allocations and orders over a small pool of users. While we used uncorrelated channels in all of our other simulations, in the case of spatially correlated user channels a candidate pool with an update criteria that favours more orthogonal, uncorrelated users could conceivably offer better performance, similar to the intermediate grouping acceptance threshold used in the greedy algorithm in [52]. Since our research did not include the effects of correlated channels, further work on this concept would be an interesting area for future work.

Chapter 5

Simulation Results

In this section we evaluate the performance of SA as a data stream scheduling algorithm for maximum-throughput scheduling in a system employing SZF with coordinated transmit receive processing, where we explicitly schedule up to n_R data stream per user on each users maximal channel modes. The first section briefly details the negligible performance loss experienced when we choose to allocate data streams only to the maximal eigenmodes of each user, and justifies the use of maximal eigenmode transmission. The second section details the derivation of the SA parameters from the delta-rates distribution estimates, and details the effect of parameter variation on the actual delta-rates distributions and the convergence of the algorithm.

An evaluation of the performance of the SA algorithm in terms of sum-rate is given, and performance of SA is compared to exhaustive search data stream scheduling, exhaustive user allocation (no data stream scheduling), and the channel capacity given by DPC. The exhaustive search data stream scheduling serves as the upper bound to the performance achievable by SZF with coordinated transmit receiver processing, whereas the channel capacity given by DPC gives the maximum theoretically achievable ergodic limit. We note that all n_T data streams are assumed to be scheduled ($\sum l_i = n_T$) for all of the subsequent SA simulations. Performance of the algorithm in terms of convergence is then discussed in the subsequent section. Finally, the complexity of the algorithm and the exhaustive search are examined in the last section.

5.1 Performance of Maximal Eigenmode Allocation

As described in section 2.2.5, in order to simplify our scheduling problem we chose to allocate data streams only to the maximal eigenmodes of each user. This removes an additional layer of combinatorial complexity that would otherwise exist by having to search over $\frac{n_R!}{l_i!(n_R-l_i)!}$ possible combinations of eigenmodes allocations for each user. Instead, the l_i data streams allocated for each user i are sent along the l_i dominant eigenmodes of that users channel. For small to medium amount of users and antennas ($|\mathcal{S}| \leq 10$, and $n_T = 4$, $n_R = 2$ for example), we expect to see a performance loss for the maximal eigenmode search in comparison to searching over all possible eigenmodes. This is because the scheduler is more likely to find orthogonal (less correlated) eigenmodes when searching over all possible modes for each user when there are limited numbers of users to transmit to. However, as we are dealing with designing a user scheduling algorithm to work in an environment in which large number of users requesting service at a given instant, we show that the performance gap associated with maximal eigenmode transmission can be considered negligible when $|\mathcal{S}|$ is large, (the case of interest to us), and is an acceptable performance-complexity tradeoff.

This effect is visible in figure 5.1, where "exh eigenmode" denotes an exhaustive search over all possible eigenmodes of all users channels, while "exh max eigenmode" is the case of an exhaustive search over the maximal eigenmodes only. The "user allocation only" case assumes that an exhaustive search using only user allocation is performed by allocating the maximum number of streams (such that $l_i = n_R$) per user. The complexity of the exhaustive eigenmode search does not scale well past 6-10 users, for $n_T = 4$ $n_T = 2$, (with over 6 hours for $N_{samp} = 1$ Monte Carlo iterations required at $n_T = 4$ $n_T = 2$ and $|\mathcal{S}| = 10$), and is completely impractical for the $n_T = 4$ $n_R = 4$ $|\mathcal{S}| > 4$, and $n_T = 8$ $n_R = 2$ $|\mathcal{S}| > 8$ cases, where benchmarks for calculating the time for $N_{samp} = 1$ Monte Carlo iterations ran over the maximum execution time limits on the Jasper Cluster. Due to impracticality of simulating even small numbers of N_{samp} for the larger transmit and receive antenna array cases, we present only results for $n_T = 4$ and $n_R = 2$ in Figure 5.1.

Figures 5.2 and 5.3 (for the $n_T = 4$,$n_r = 2$ and $n_T = 4$,$n_R = 4$ cases respectively) show the sum-rate performance increase over user allocation achievable with variable data stream allocation when using SZF precoding, and allow us to visualize the sum-rate distributions possible for each of the various stream allocations. These results were generated by running

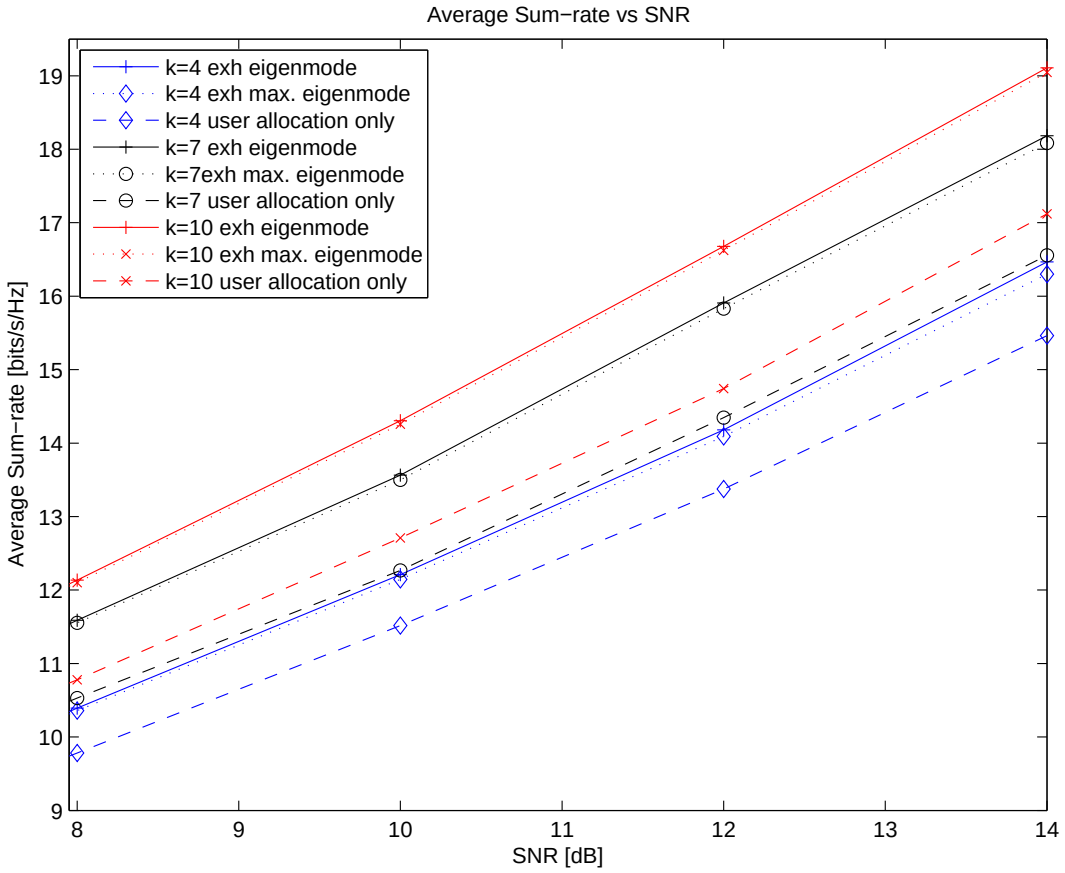

Figure 5.1: Exhaustive and maximal eigenmode transmission with coordinated transmit-receiver processing on average sum-rate performance for $n_T = 4$, $n_R = 2$

a random search at each $\mathbf{L}^m$ and finding the best achievable rates with random users and order. The total number of random solutions generated for each $\mathbf{L}^m$ was 10 000. The legend given for each allocation $\mathbf{L}^m$ denotes the results generated for any possible permutations of the stream allocation vector in question. For example, the results obtained by allocating $\mathbf{L}^m = [1, 2, 1]$, or 1 stream to the first user in $\mathcal{S}_T$, 2 to the second, and 1 to the third is still denoted by [2,1,1]. In the n_R=2 cases, choosing any stream allocation will on average result in better performance than sending a full n_R streams to each user. Transmitting a single data stream to n_T users allows on-par (low SNR $\approx 0dB$) or better sum-rates (as SNR increases) than those achievable with the [2,1,1] and [2,2] allocations. In general at all SNR's, stream allocation for $n_T = 4$, $n_R = 2$ allows higher achievable rates than just user allocation.

The benefits of multi-stream diversity are more pronounced when n_R and $|\mathcal{S}|$ are in-

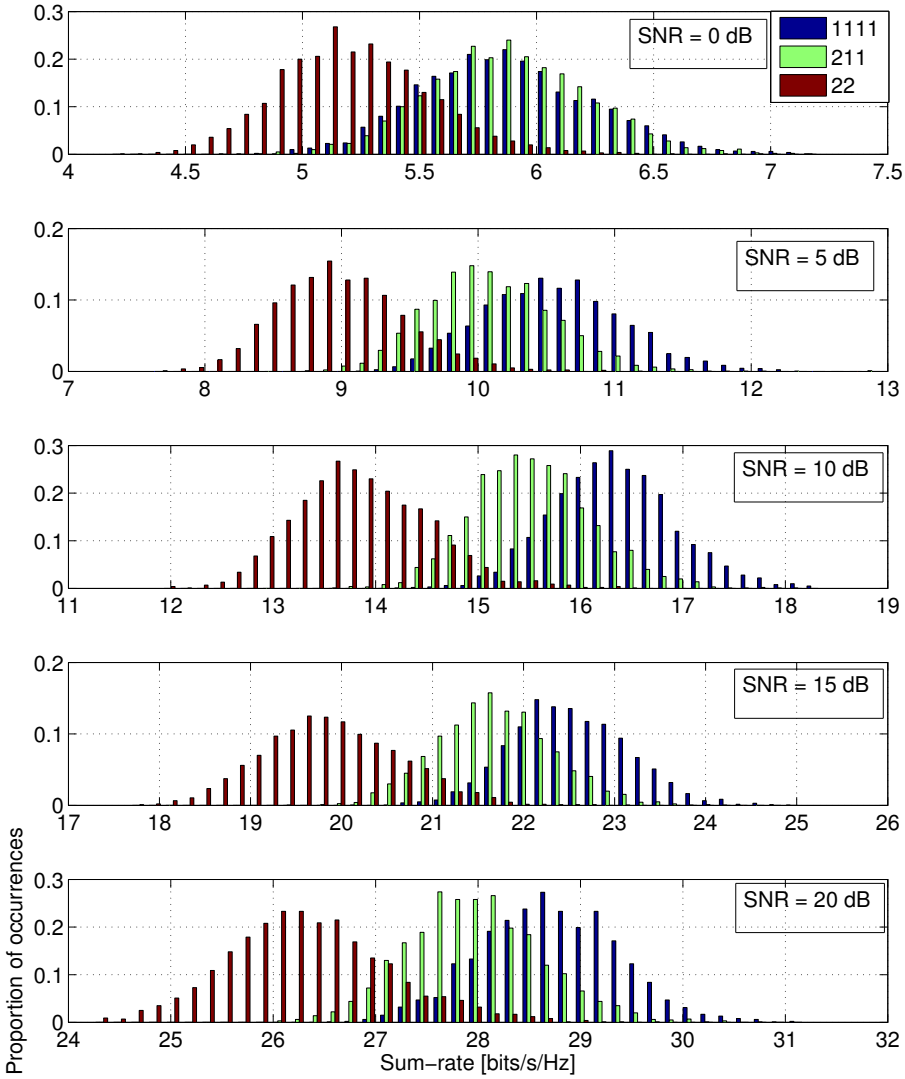

Figure 5.2: Proportion of occurrences of the best achievable rates under SZF with variable data stream allocation for $n_T = 4$, $n_R = 2$,$|\mathcal{S}| = 30$ at various SNR

creased, and similar results to those obtained for the $n_R = 4$ case are obtained when we increase the number of antennas at each user's terminal to $n_R = 4$ and the number of users $|\mathcal{S}|$. In the $n_T = 4$ $n_R = 4$ case, it should first be noted that in the stream allocation $\mathbf{L}^m = [4]$, there is no SZF precoding and we only waterfill the power over the users channel. This is because no other users to interfere with, thus making the projection onto

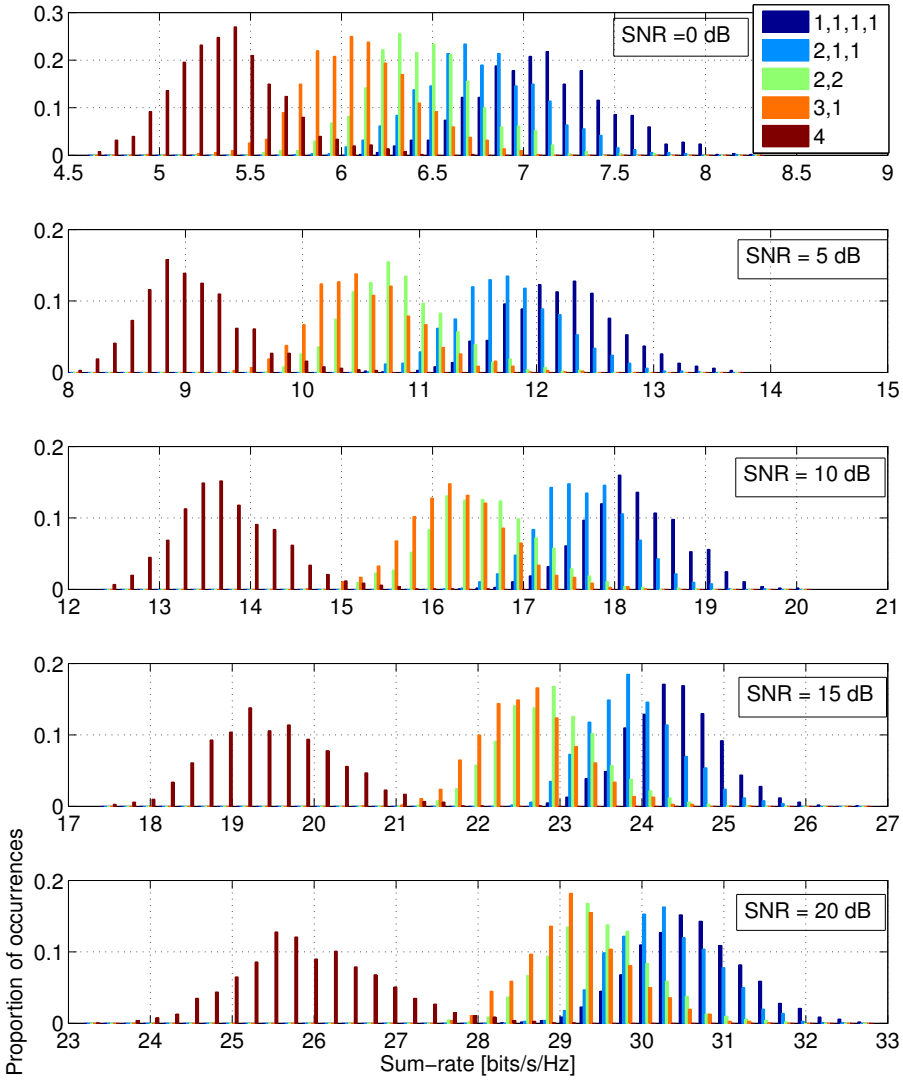

Figure 5.3: Proportion of occurrences of the best achievable rates under SZF with variable data stream allocation for $n_T = 4$, $n_R = 4$,$|\mathcal{S}| = 25$ at various SNR

the nullspace of the non-existent previous users unnecessary. At low SNR, sending a single data stream to n_T users still affords the greatest average rate and outperforms the other allocations. Any stream allocation other than $\mathbf{L}^m = 4$ at low SNR offers markedly improved performance over the single user case, with the effects becoming more pronounced at higher SNR. Even scheduling just two users, with either $\mathbf{L}^m$ =[2,2] or [3,1] streams offers tangible

gains in achievable sum-rate sum-rate, but in general allocations with fewer data streams per user offer the best performance.

Overall, it can be seen that by increasing the flexibility of the SZF algorithm by allowing variable numbers of data streams per user (thus scheduling up to a maximum of $K_{max} = n_T$ users offers noticeable increases in achievable rates over scheduling a full n_R data streams per user (allowing only $K_{max} = \lfloor \frac{n_T}{n_R} \rfloor$ users). In particular at higher SNR, because certain types of data stream allocations afford better performance, priority could be given to using these types of allocations in a scheduling algorithm for faster convergence or better overall performance.

5.2 SA Algorithm Parameters

5.2.1 Repetition Limits, and Decay Rates

In order to compare the performance of the algorithm while varying the repetition limits M_u and M_m, we chose a set of complexity targets (in terms of the number of unique sum-rate calculations of $R(\omega)$) listed in the first column of Table 5.1 and chose the repetition limits to fit those targets. These outer repetition limits M_u are set based on practical decay rates parameters and initial acceptance probabilities (see section 4.5.5) and were chosen to be $M_u = 100, 200, 300$.

Given these outer repetition limits and the chosen complexity targets, the target complexity was not achievable with an integer multiple of M_u in some cases. Therefore, in these cases the lowest M_m (listed in Table 5.2) satisfying $M_u \times M_m \geq$ (**target complexity**) was chosen. Since the majority of our interest was in the performance of simulations in the range of 2000-7500 sum-rate calculations, only overshoots of 101.3% and 102% and 105% of the target complexities were observed, and their effects can be considered negligible. Table 5.1 lists the outer and inner repetition limits M_u and $M_{m,c_{\text{cmplx}}}$ and decay rates used for simulations (with c_{cmplx} denoting the complexity index that will be used in the subsequent evaluation sections).

index c_{cmplx}	Target	(M_u=100) $\times M_{m,c}$	% Target	(M_u=200) $\times M_{m,c}$	% Target	(M_u=300) $\times M_{m,c}$	% Target
1	500	500	100	600	120	600	120
2	1000	1000	100	1000	100	1200	120
3	2000	2000	100	2000	100	2100	105
4	3000	3000	100	3000	100	3000	100
5	4000	4000	100	4000	100	4200	105
6	5000	5000	100	5000	100	5100	102
7	7500	75000	100	7600	101.3	7500	100
8	10000	10000	100	10000	100	10200	102

Table 5.1: Solution complexity for different repetition criteria

M_u limits	$M_{m,c}$ limits (c_{cmplx}=1, c_{cmplx}=2, ..., c_{cmplx}=8)	α	$\lvert\mathcal{S}\rvert$
100	5, 10, 20, 30, 40, 50, 75,100	93, 94, 95, 96	10, 20, 30, 40
200	3, 5, 10, 15, 20, 25, 38, 50	94, 95, 96, 97	10, 20, 30, 40
300	2, 4, 7, 10, 14, 17, 25, 34	95, 96, 97, 98	10, 20, 30, 40

Table 5.2: Parameter sweep ranges for SA simulations

	Simulated transition probabilities		
transition index c_{trans}	p_a - Allocation swap	p_u - User swap	p_o - Order swap
1	0.25	0.70	0.05
2	0.35	0.60	0.05
3	0.45	0.50	0.05
4	0.55	0.40	0.05
5	0.65	0.30	0.05

Table 5.3: Neighbourhood transition probabilities

5.2.2 Delta-Rate and Temperatures

To generate the approximations of $|\Delta R|$ distributions, we used the method detailed in section 4.5.4. It was found through initial experimentation that the greatest effect on the $\Phi_{|\Delta R|}$ distributions was SNR. This was more or less expected, as in our simulations the transmit power was varied from 0-20 dB, and the effect of power on the variance of the achievable rates for a given user pool had a greater effect than the gains achieved by doubling the number of transmit and receive antennas. Varying the number of users from $|\mathcal{S}| = 10$ to

$|\mathcal{S}| = 40$ had a negligible effect on the $\Delta R_{0.9}$, ($\sim 2\%$ variance) that increased slightly at higher SNR's to $(3-4\%)$.

For a given SNR, n_T, and n_R, it was originally thought that the neighbourhood transition probabilities p_a, p_o, p_u change the characteristics of the neighbourhood function $N()$ enough to cause a statistically significant shift in $\Phi_{|\Delta R|}$, and consequently $\Delta R_{0.9}$. The initial $\Phi_{|\Delta R|}$ distributions were then made for each: SNR, n_T, n_R, $|\mathcal{S}|$, and neighbourhood transition probabilities listed in Table 5.3, with corresponding transition indexes c_{trans} which will be used in the subsequent evaluation sections. Upon sweeping over $c_{\text{trans}} = 1$ to $c_{\text{trans}} = 5$, it was found that these parameters also had a marginal effect on the $\Delta R_{0.9}$. Figure 5.4 shows the effect of the transition probabilities at high SNR at which the effects were the most visible for $n_T = 4$ $n_R = 2$. $\Delta R_{0.9}$ values ranged from approximately $6.65 \sim 6.85$ corresponding to initial temperatures ranging from $t_0 = 63 \sim 65$, a 3% change. Similar trends were observed for $n_T = 4$, $n_R = 4$ and $n_T = 8$, $n_R = 2$ (not shown) and therefore we chose $c_{\text{trans}} = 3$ for our neighbourhood function transition parameters for the simulation of $\Phi_{|\Delta R|}$ estimates. Figure 5.5, 5.6, and 5.7 show the $\Phi_{|\Delta R|}$ for (n_T,n_R); (4,2),(4,4), and

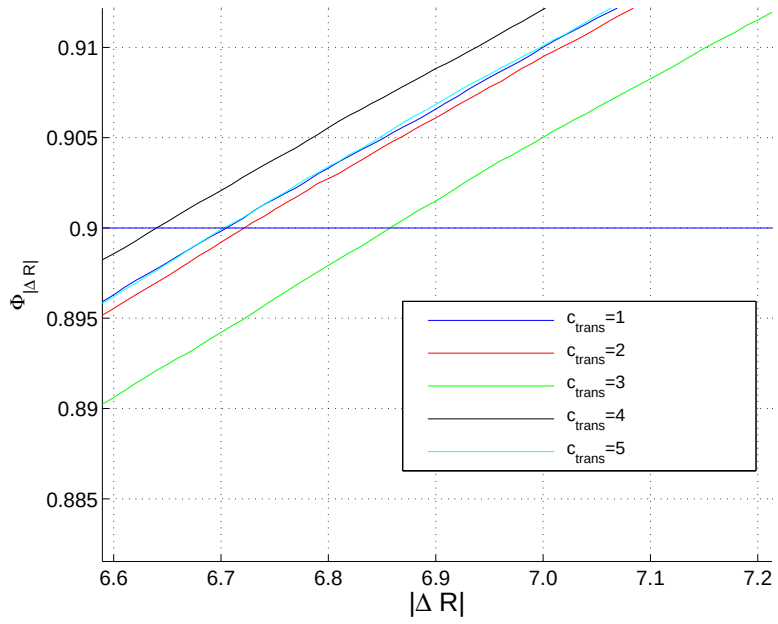

Figure 5.4: $\Delta R_{0.90}$ for various c_{trans} at $n_T = 4$, $n_R = 2$, $|\mathcal{S}| = 40$, SNR=20 dB

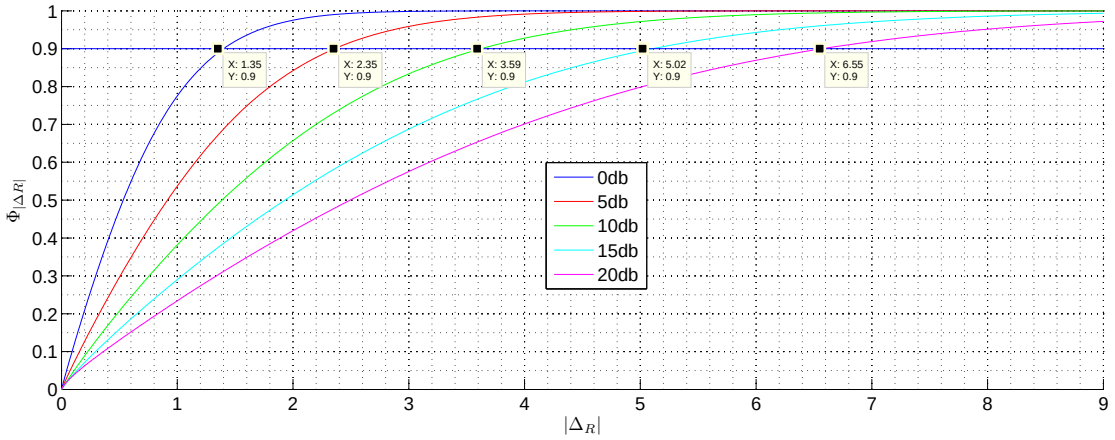

Figure 5.5: $\Phi_{|\Delta R|}$ distributions for $n_T = 4$, $n_R = 2$ showing $\Delta R_{0.90}$ for 0-20 dB

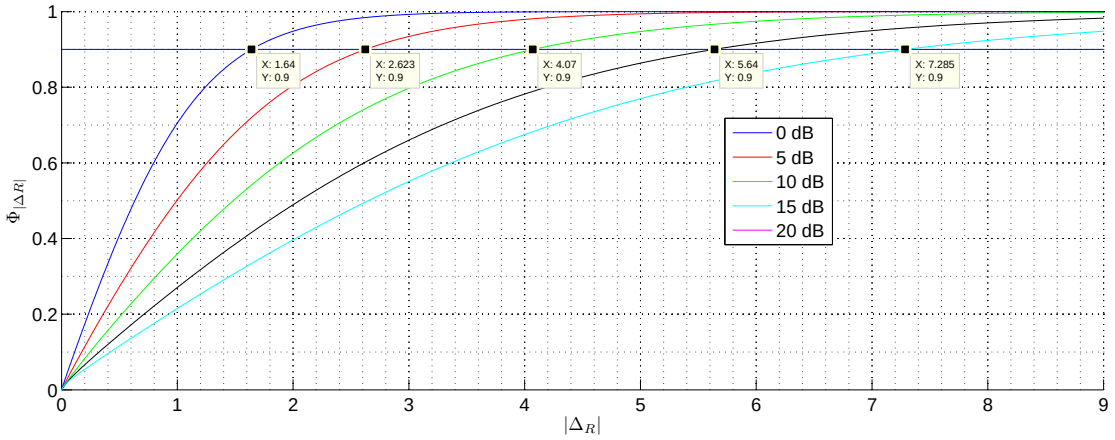

Figure 5.6: $\Phi_{|\Delta R|}$ distributions for $n_T = 4$, $n_R = 4$ showing $\Delta R_{0.90}$ for 0-20 dB

(8,2), at SNR$= 0-20dB$ with corresponding $\Delta R_{0.9}$ values for a particular user set $|\mathcal{S}| = 40$, averaged over 4000 iterations. After obtaining $\Delta R_{0.9}$ and setting $P_{a,o} = 0.90$, we used eq 4.8 to get the initial temperature values t_0 for each SNR. The range over SNR$= 0-20$ dB of initial temperatures t_0 from their corresponding $\Delta R_{0.90}$ values are listed in table 5.4. It is evident that as the n_T (and hence the possible data streams) and increases, the variance of achievable rates also increases, while similarly increasing n_R and allowing greater numbers of data streams per user also has a similar effect.

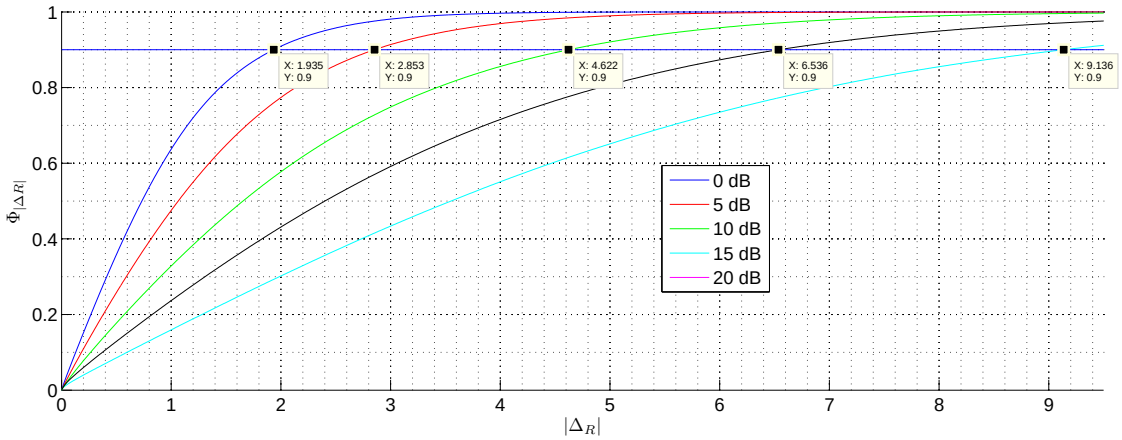

Figure 5.7: $\Phi_{|\Delta R|}$ distributions for $n_T = 8$, $n_R = 2$ showing $\Delta R_{0.90}$ for 0-20 dB

n_T	n_R	SNR=0dB	SNR=5dB	SNR=10dB	SNR=15dB	SNR=20dB
4	2	12-14	21-23	34-36	46-48	62-64
4	4	15-17	24-27	37-40	53-56	70-73
8	2	13-15	25-28	38-41	57-60	85-89

Table 5.4: Temperatures ranges for each n_T and n_R per each SNR, derived from delta-rates distribution estimates to satisfy $P_{a,1} = 0.90$

5.3 Sum-Rate results

In all of the cases regardless of the number of antennas, users, and SNR's, the achievable rates with exhaustive variable data stream allocation over the maximal eigenmodes is able to match or outperform the case of allocating all possible data streams per scheduled user, (or just user allocation with no data stream scheduling). This is in part because the user allocation case is a constrained version of variable data stream processing where only one allocation (allocating a maximum number of data streams (n_R) per user is possible. Therefor for a given set of users with given channel realizations, the performance of exhaustive variable data stream scheduling under SZF will never be less than that of only user allocation (no data stream scheduling). In fig. 5.8 we compare the performance of our SA scheduler

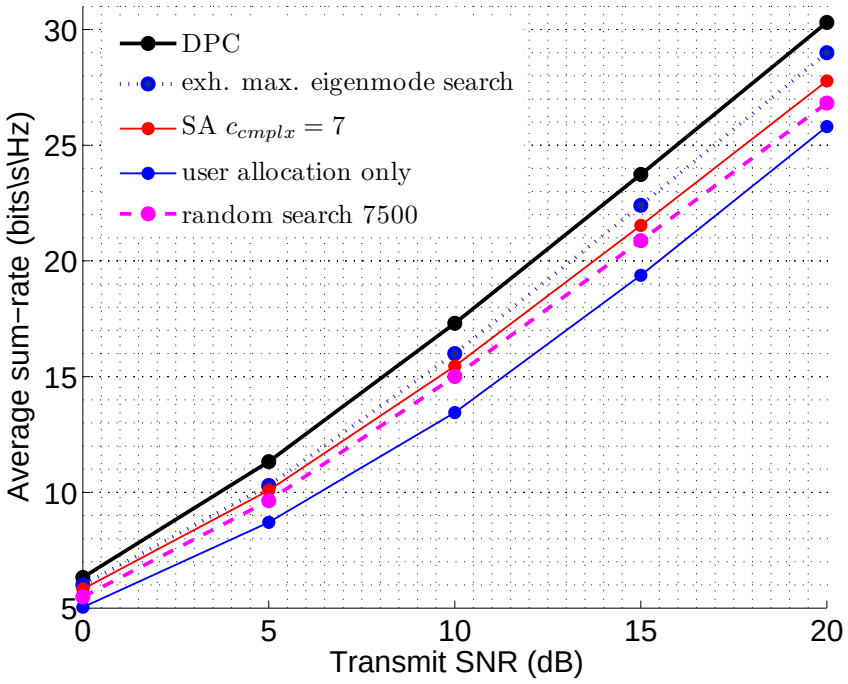

Figure 5.8: Comparison of the performance of maximum throughput scheduling vs. SNR for $n_T = 4$, $n_R = 2$ $|\mathcal{S}| = 15$ for DPC, SA, exhaustive search, random search, and user allocation

in terms of system throughput versus SNR to the exhaustive maximal eigenmode search, DPC, user allocation only, and a random search at $n_T = 4$, $n_R = 2$, and $|\mathcal{S}| = 15$. In the graph, "exh. max. eigenmode" is the case of an exhaustive search over the maximal

eigenmodes only. The "user allocation only" case assumes that an exhaustive search using only user allocation is performed, allocating the maximum number of streams (n_R) per user. Finally, DPC refers to the maximal theoretical possible capacity of the given users when dirty paper precoding is implemented. Regarding the portions referring to DPC, it is not an SA scheduler for DPC. Instead, DPC refers to the ergodic channel capacity results (eq.2.7), Thus no scheduler is used nor strictly required when looking at the theoretical capacity. For the case of throughput maximization, power is allocated using a waterfilling method to all available users in DPC, and only those users with non-zero power are actually transmitted to.

It can be seen that in this case SA provides 93% of the exhaustive maximal eigenmode search in this case while being significantly less complex in terms of number of utility function calculations. SA also outperforms (at all SNR) both the exhaustive user allocation search case and a random search with the same number of utility function calculations. It can be seen that both the SA and random search diverge from the exhaustive maximal eigenmode search at higher SNR, but that SA suffers less divergence as SNR increases. In fig. 5.8 the best overall average results for the SA parameters were used and were $c_{\text{trans}} = 1$, $M_u = 300$, $c_{\text{cmplx}} = 8$ $\alpha = 4$. For clarity in some of the graphs, we omit the plotting of the average rate results for different SA parameters at $M_u = 200$ and instead focus on showing the greatest effect of varying M_u by showing the results for $M_u = 100$ to $M_u = 300$. We justify this by mentioning that keeping all other parameters constant, M_u displays the same trends as the ones observed going from $M_u = 100, 300$. In all cases simulated, the performance of the algorithm increased as $c_{cmplx} \to 8$, which allowed a greater exploration of the search space by increasing the number of inner iterations M_m at each temperature.

For the effect of the c_{trans} SA parameter on achievable throughput, fig. 5.9 shows the average throughput results of parameter sweeps for the case of $n_T = 4$,$n_R = 2$, and $c_{\text{cmplx}} = 7$, at ∼SNR=20 dB where the effects of the parameter variation are most significant. It can be seen from fig. 5.9 that the best average sum-rate is offered by $c_{\text{trans}} = 1$ in most of the cases, and decreases as $c_{\text{trans}} \to 5$. The significance of these results is that the algorithm performs best at $n_T = 4$ and $n_R = 2$ when the neighbourhood function has the highest probability of swapping users instead of checking other stream allocations. The better performance is achieved by more heavily exploiting multiuser diversity instead of exploring

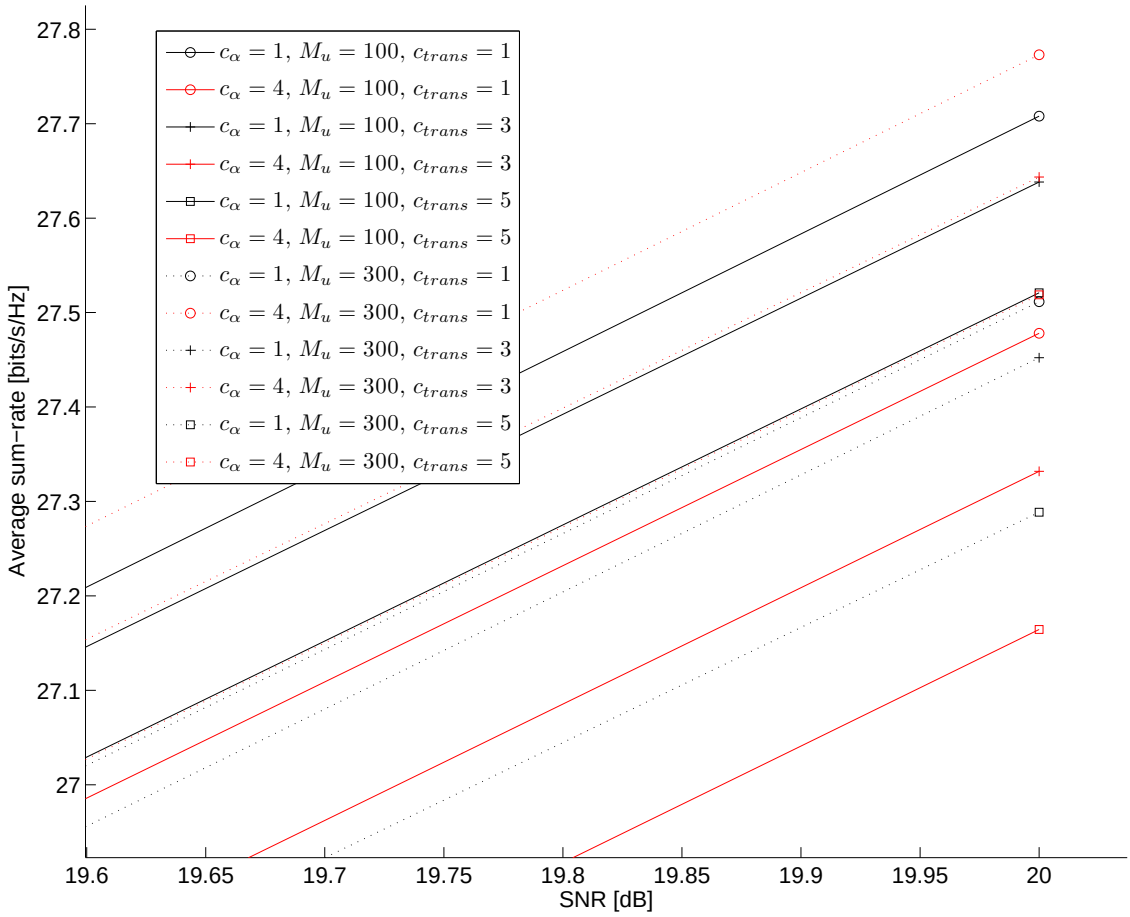

Figure 5.9: Effect of SA parameter variation on average throughput for the case of $n_T = 4$ $n_R = 2$ $|\mathcal{S}| = 40$ and SNR=20 dB

a limited number of stream allocations available at $n_T = 4$ $n_R = 2$. As we can see, for the same c_{trans}, choosing the higher outer loop parameter $M_u = 300$ will always yield the better performance when the decay rate $c_{alpha} = 4$, whereas choosing $M_u = 100$ provides the best performance when $c_{alpha} = 1$. In general however, choosing a faster decay rate ($c_\alpha = 1$) and larger outer number of iterations ($M_u = 300$) tends to provide on average better on achievable sum-rate performance. Similar results for the effects of the parameters on the average throughput were obtained for $n_T = 4$, $n_R = 4$.

The average throughput results for SA for $|\mathcal{S}| = 20$ and $|\mathcal{S}| = 40$ are shown in fig.5.10,

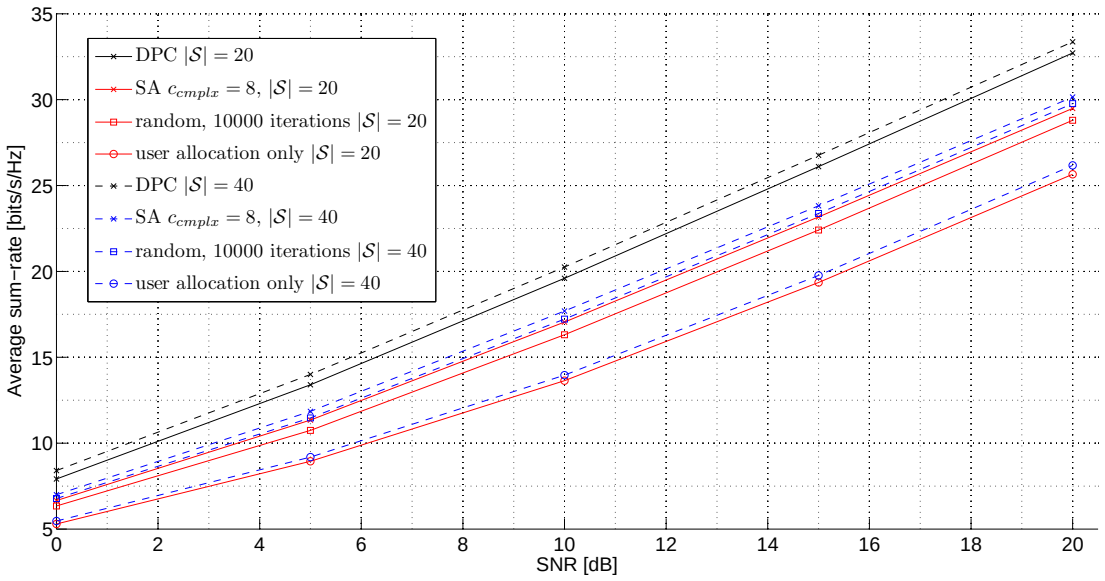

Figure 5.10: Average sum-rate with DPC, simulated annealing (SA), exhaustive user allocation, and random search for $n_T = 4$, $n_R = 4$, $|\mathcal{S}| = 20, 40$, SNR $= 0 - 20$ dB

and it can be seen that SA is able to outperform both the exhaustive user allocation search and the random search in both cases. The gap between SA and the random search narrows slightly as we go from $|\mathcal{S}| = 20$ and $|\mathcal{S}| = 40$, possibly due to the fact that the unstructured nature of the random search is able to better able to explore the search space as opposed to the comparatively structured nature of the SA.

For the case of $n_T = 8$, $n_R = 2$, it was found that regardless of the complexity c_{cmplx} and other parameters chosen, we could *at best* only match the performance of exhaustive user allocation. In fig.5.11 we can see that only at the highest complexity level are we able to match the performance of exhaustive user allocation. Recall from table 2.2 that the total number of utility function calculations required for the exhaustive user allocation search for $n_T = 8$ $n_R = 2$ and $|\mathcal{S}| = 10$ is 5040. For this case, it is clear that stream allocation with simulated annealing is impractical and should not be used, as it takes approximately twice the number of iterations to simply match the performance of the exhaustive user allocation search. A random search with the same complexity actually outperforms SA, most likely due to the fact that the solution space (8.2×10^6 solutions) is so large that that SA would require many more iterations to actually exhibit some convergence like in the $n_T = 4$ $n_R = 2$

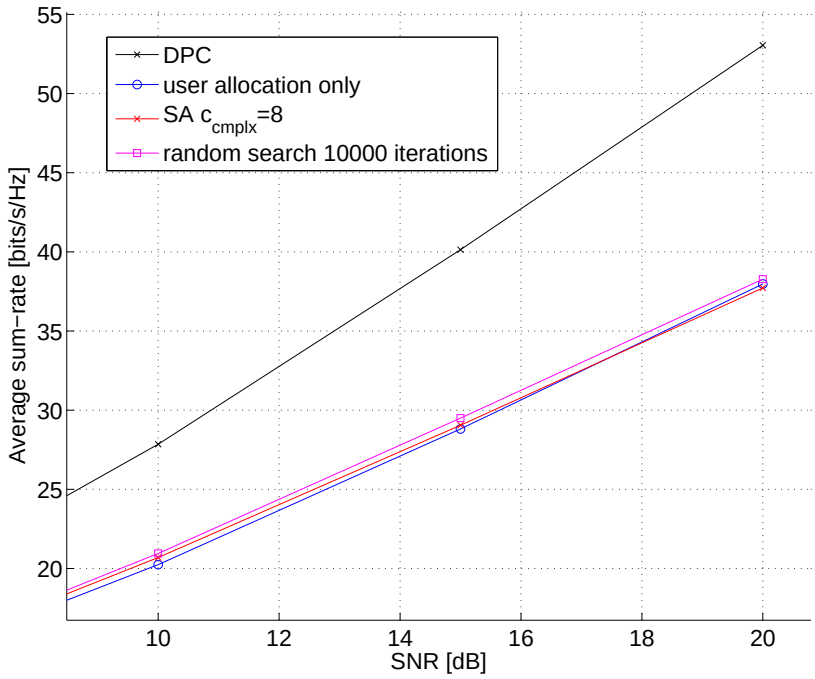

Figure 5.11: Average sum-rate with DPC, exhaustive user allocation, random search and simulated annealing (SA) for $n_T = 8$, $n_R = 2$, $|\mathcal{S}| = 10$, SNR $= 0 - 20$ dB

and $n_T = 4$ $n_R = 4$ case. The behaviour SA in this case in terms of the total number of iterations required to reach the best achievable utility function value was much like the first graph of fig. 5.16, and therefore exhibited no fast convergence. Given the current structure of the algorithm, for $n_T = 8$ it is simpler and faster to use the exhaustive user allocation search, although future work could include an extension of the algorithm to investigate the effects of larger antenna arrays. Due to the relatively poor overall performance of the SA algorithm for the structure and parameters for the case of $n_T = 8$, we chose to exclude the analysis for the delta-rate graphs and include only figure 5.11 as an example.

5.4 Convergence of SA

Due to the random nature of the SA algorithm in which the neighbourhood function randomly chooses candidate solutions and then from those candidates suboptimal solutions are randomly accepted, the convergence of the algorithm can arguably be said to be a stochastic process. While the convergence of the algorithm in the limit for iterations is theoretically possible if careful consideration is taken in the structuring of the algorithm, there exists a distinct possibility that SA will not converge to the globally optimal value in a finite number of iterations. However, while SA may not necessarily converge to the globally optimal value it has been shown to find relatively good solutions quite quickly. If the solutions in the intermediate stages of the execution are sufficiently good then further execution is not necessary and the algorithm can terminate. This warrants the inclusion of the additional stop criteria that can be placed at the end of each outer loop iteration u to halt the execution of the algorithm when the solutions is "good enough". Additionally, while the true SA approach is memoryless and can potentially move away from the optimal solution by choosing a locally suboptimal solution and 'forget' the best solution found thus far, we employ a elitism mechanism by which the best solution encountered is always saved. By employing elitism, the utility function can now be considered strictly increasing as the execution progresses.

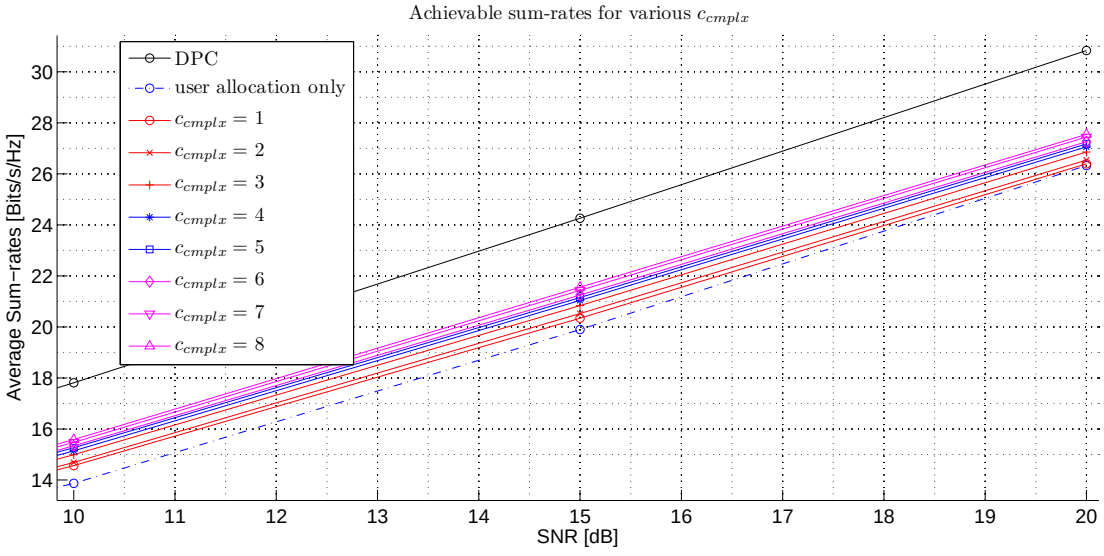

Figure 5.12: Achievable sum-rate performance of SA over various c_{cmplx} for $n_T = 4$, $n_R = 2$, $|\mathcal{S}| = 30$, SNR $= 0 - 20$ dB

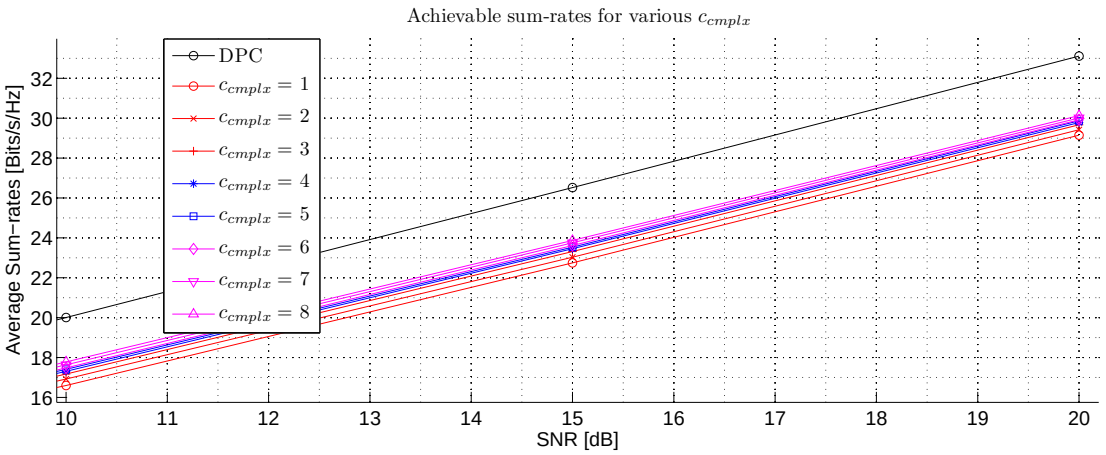

Figure 5.13: Achievable sum-rate performance of SA over various c_{cmplx} for $n_T = 4$, $n_R = 4$, $|\mathcal{S}| = 30$, SNR $= 0 - 20$ dB

Figures 5.12 and 5.13 for the $(4, 2, 30)$ and $(4, 4, 30)$ cases respectively, highlight the average best-achievable sum-rate performance of SA when c_{cmplx} is varied from 1 to 8, with the total number of total iterations increasing from 1000-10000. It can be seen that after approximately 5000 iterations, increasing the complexity of the algorithm results in diminishing marginal returns for both cases. This result also correlates our findings on the convergence of the algorithm covered in the subsequent section, where the algorithm is shown to converge close to the best achievable value after approximately 5000-6000 total iterations $(M_u \times M_m)$. $|\mathcal{S}| = 30$ was chosen to demonstrate the achievable sum-rate convergence as the total number of unique solutions for the exhaustive eigenmode search are much greater than the number of SA iterations which gives us a more realistic view on how well the algorithm is able to extract multi-stream diversity from a very large solution space. The simulation parameters resulting in these best achievable average rates tended to be $M_u = 300$ with $c_\alpha \approx 1$ or 2, with $c_{\text{trans}} \approx 1$ or 2.

The next several figures show the probability that the algorithm has achieved the best solution it will reach during execution by the given number of total iterations $(max((u - 1) \times M_m, 0) + m)$ for the (4,2,10), (4,2,40), (4,4,10),(4,4,40),(8,2,10) cases (respectively denoting (n_T,n_R,$|\mathcal{S}|$)). The decay rates for the various outer repetition limits used in these simulations are listed in Table 5.4, and the transition probabilities in table 5.3. To clearly show the impact of the decay rates on convergence, we choose to show the results of the two extremes of possible decay rates $c_\alpha = 1$ and $c_\alpha = 4$, noting that the results for $c_\alpha = 1 - 4$

follow progression of general convergence trends as c_α varies from 1 to 4. The graphs on each page are for the same $|\mathcal{S}|$, and show the results for $c_\alpha = 1$ on top, and $c_\alpha = 4$ on bottom, with SA complexity $M_u \times M_m$ given by the discretized levels c_α in increasing total complexity of 2000, 4000, and 10 000 iterations from left to right (see Figure 5.1 for values of M_m and M_u). Figure 5.14 shows the different symbols and line types used to denote the various parameters used in the subsequent graphs in fig (5.15 - 5.22).

	Decay indexes c_α			
M_u	$c_\alpha = 1$	$c_\alpha = 2$	$c_\alpha = 3$	$c_\alpha = 4$
100	93	94	95	96
200	94	95	96	97
300	95	96	97	98

Table 5.5: Outer repetition limits and their corresponding decay rates used in convergence simulations

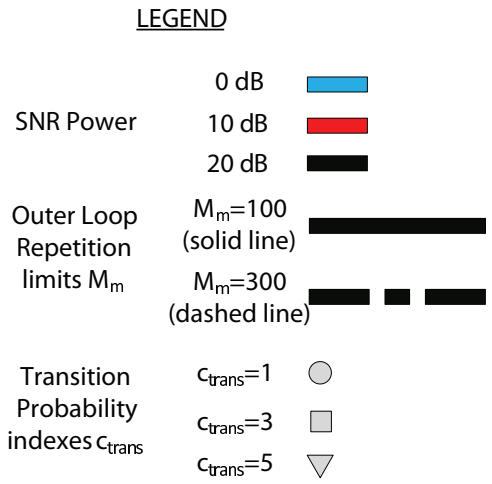

Figure 5.14: Legend denoting variables and their corresponding symbols used in fig (5.15 - 5.22

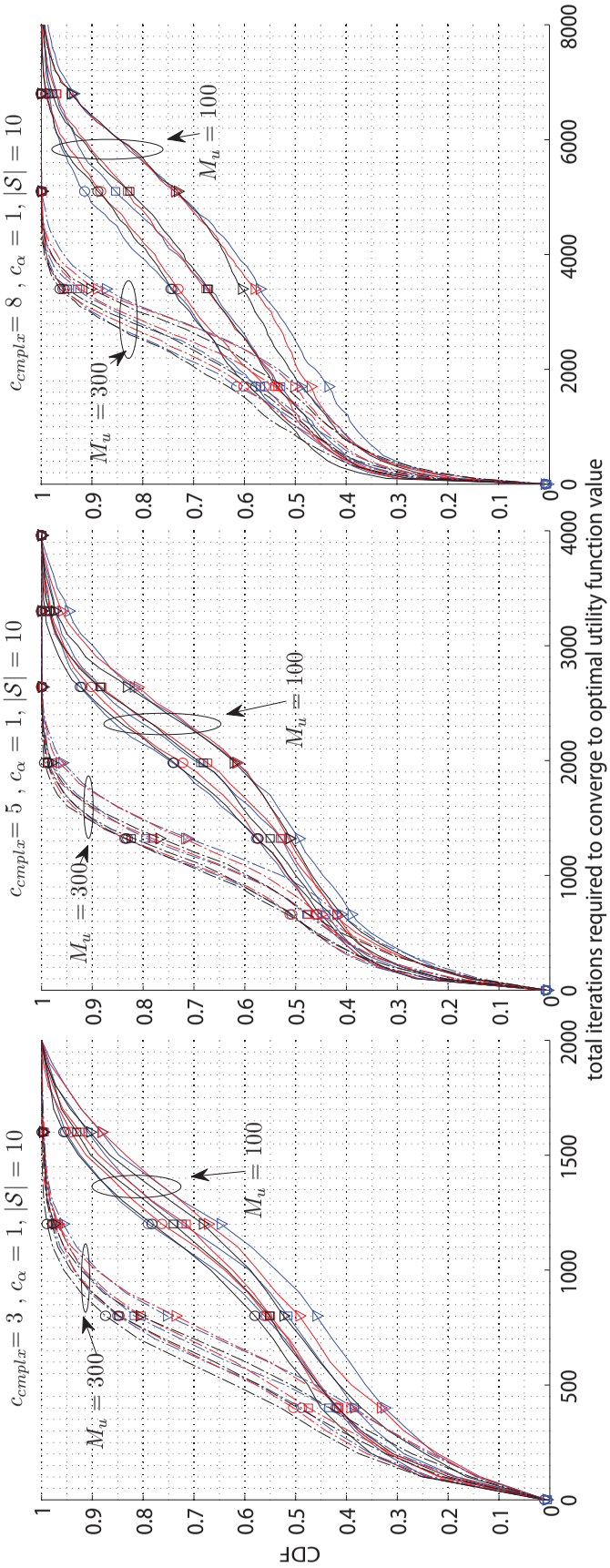

Figure 5.15: Convergence distribution for SA at (4,2,10) for $c_\alpha = 1$ showing good convergence

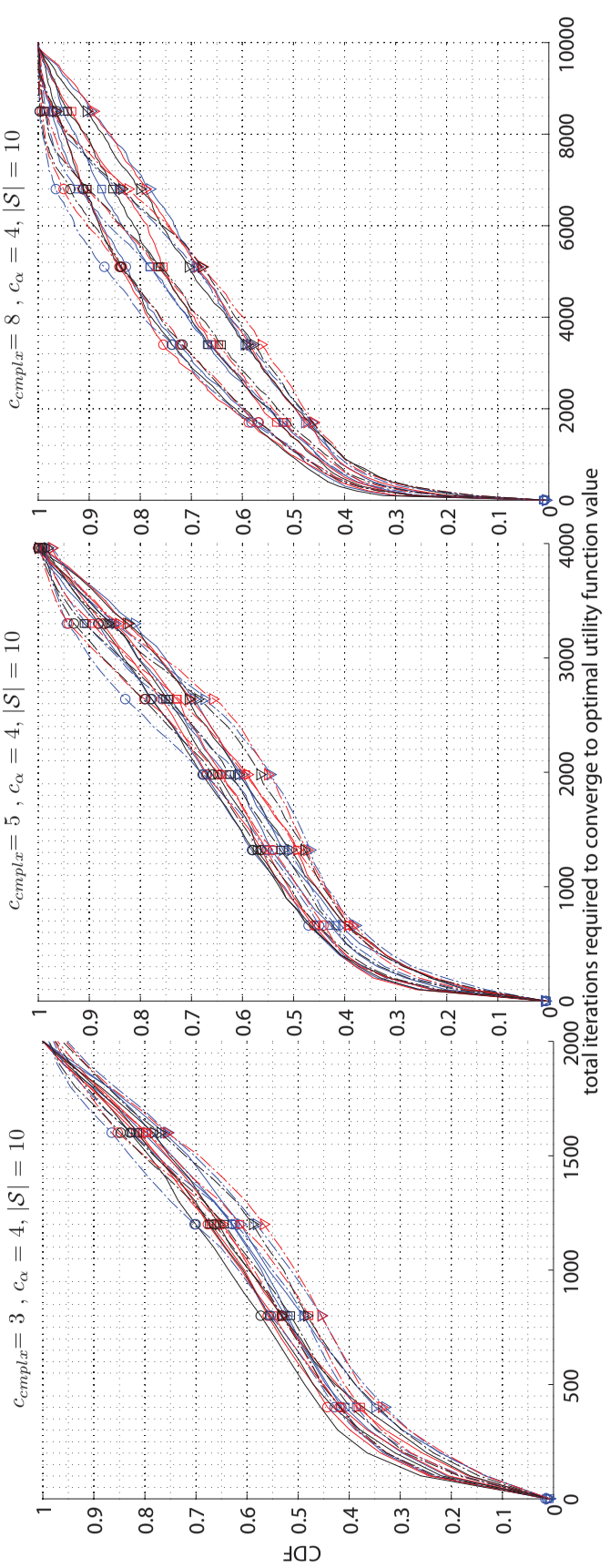

Figure 5.16: Convergence distribution for SA at (4,2,10) for $c_\alpha = 4$ showing poor convergence trends

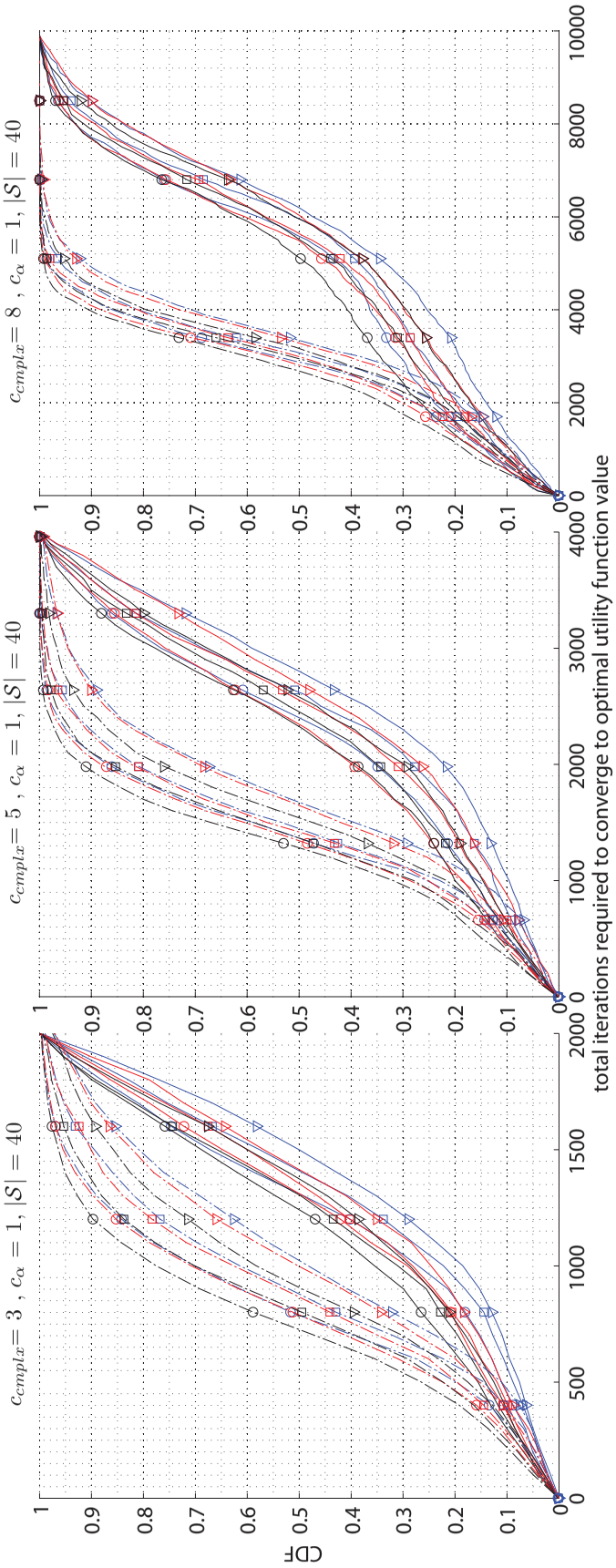

Figure 5.17: Convergence distribution for SA at (4,2,40) for $c_\alpha = 1$ showing good convergence

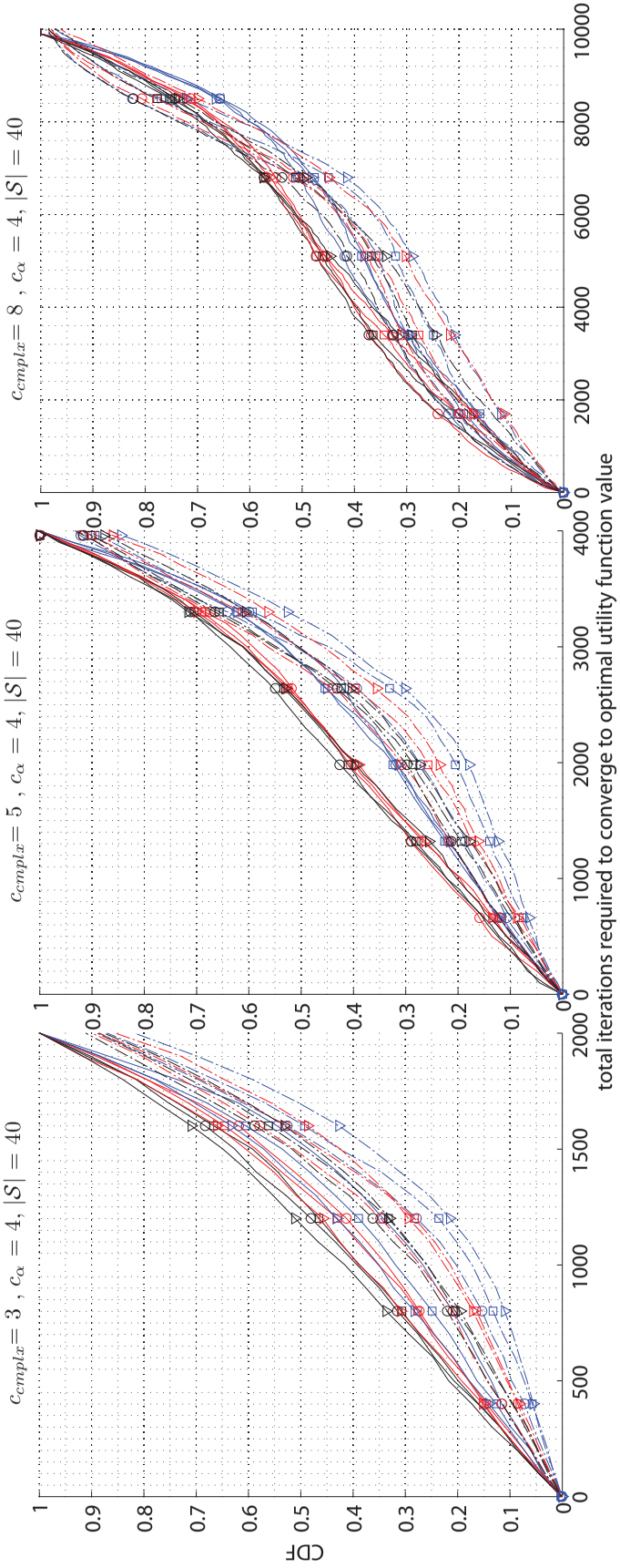

Figure 5.18: Convergence distribution for SA at (4,2,40) for $c_\alpha = 4$ showing poor convergence trends

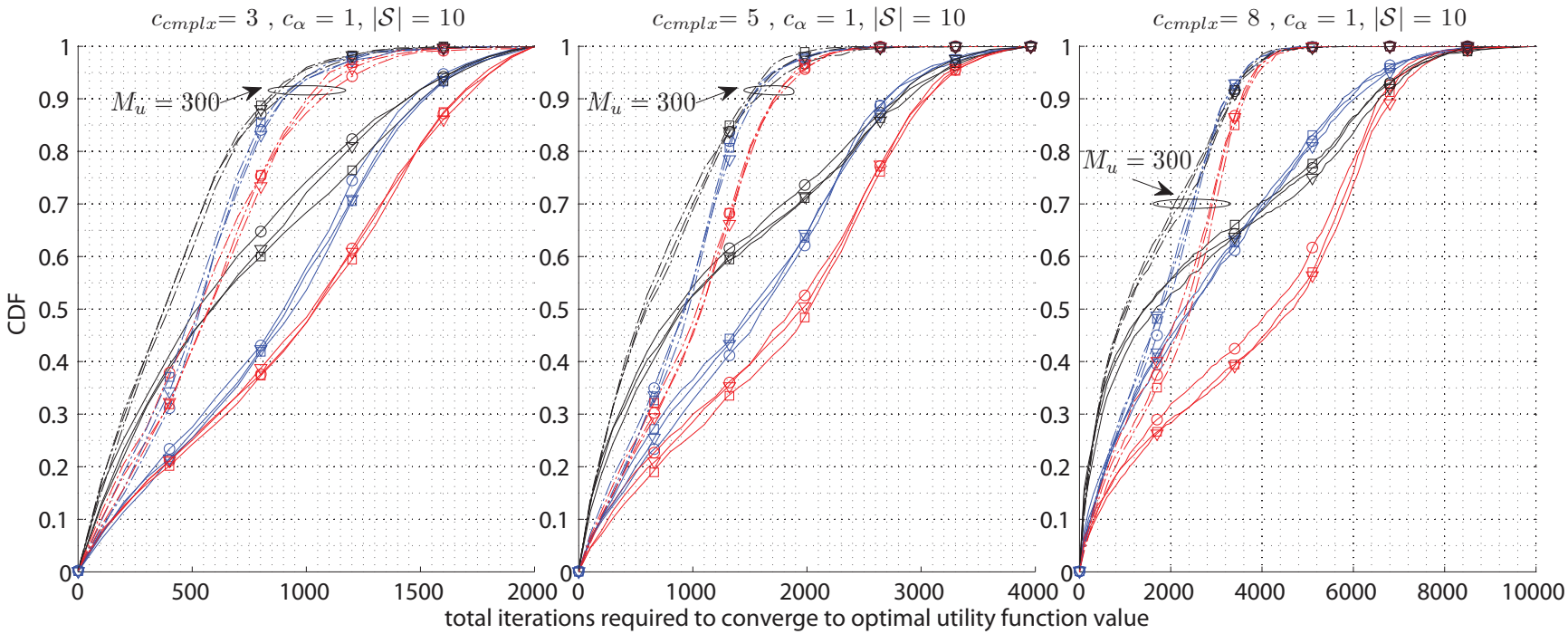

Figure 5.19: Convergence distribution for SA at (4,4,10) for $c_\alpha = 1$ showing good convergence

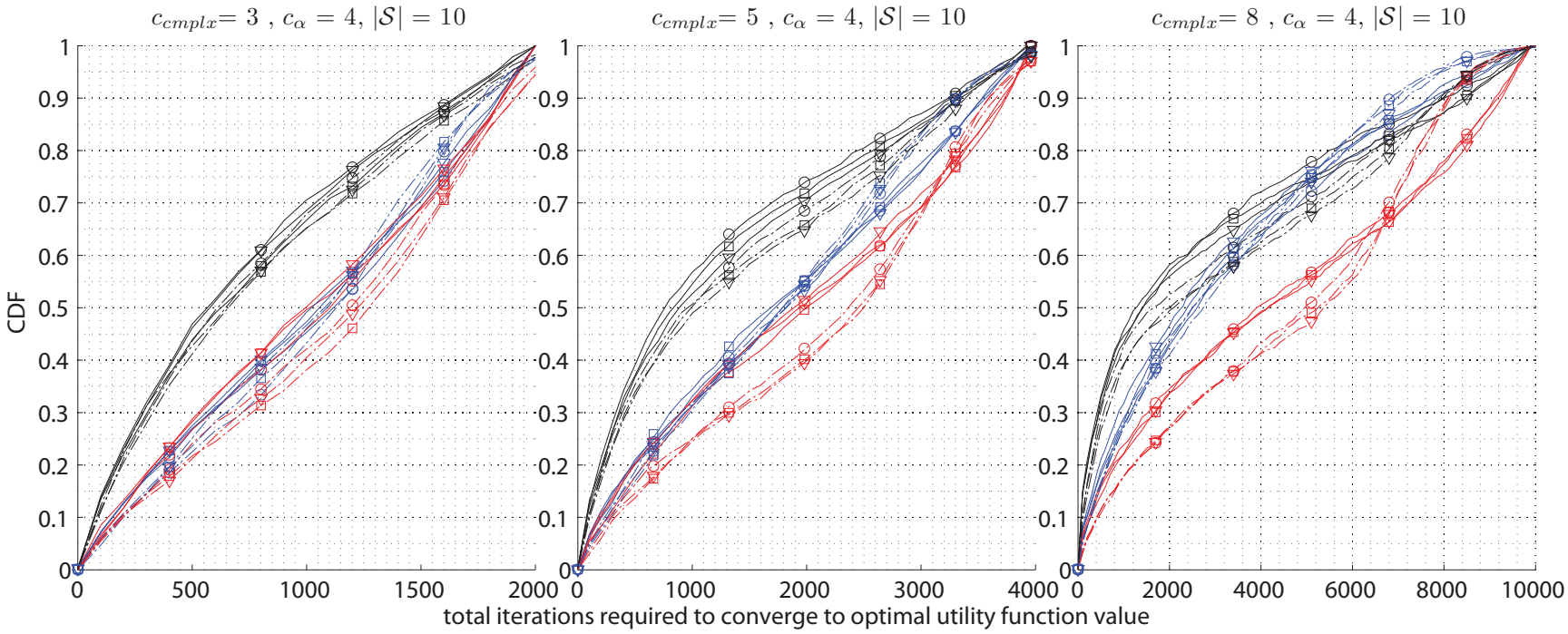

Figure 5.20: Convergence distribution for SA at (4,4,10) for $c_\alpha = 4$ showing poor convergence trends

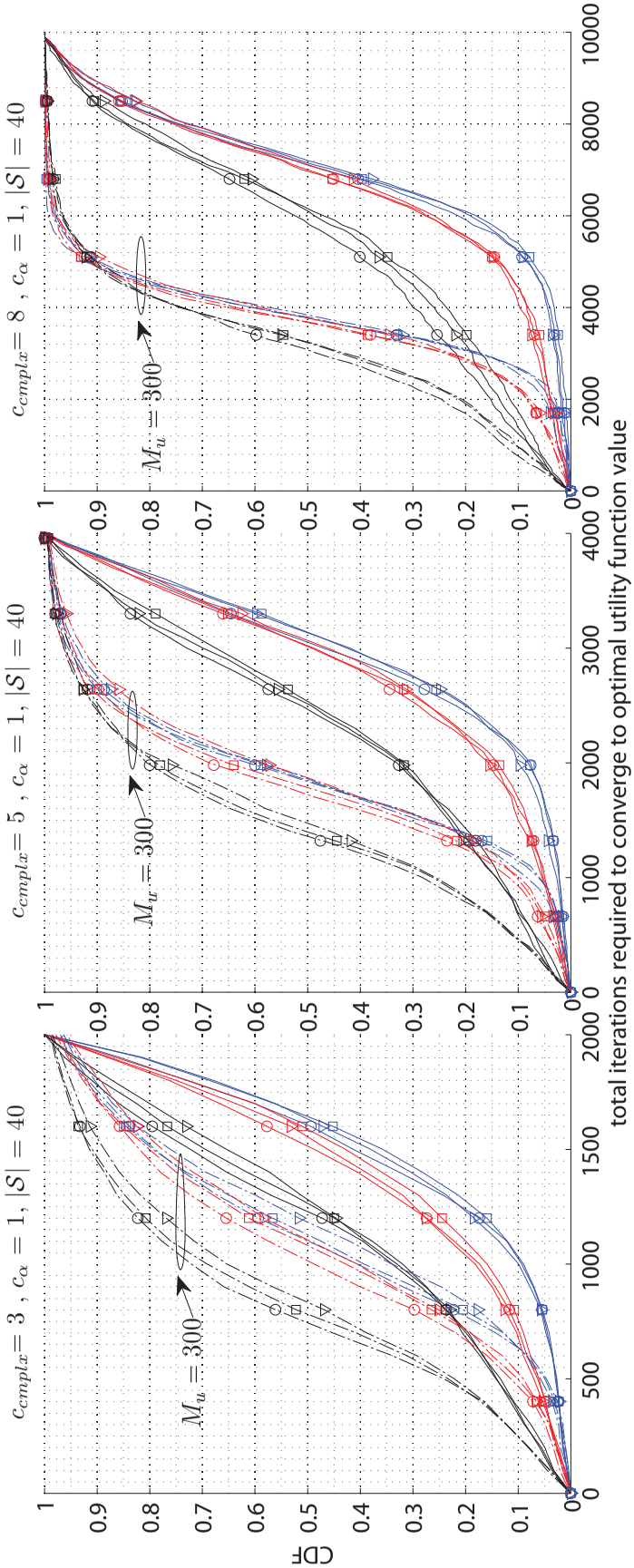

Figure 5.21: Convergence distribution for SA at (4,4,40) for $c_\alpha = 1$ showing good convergence

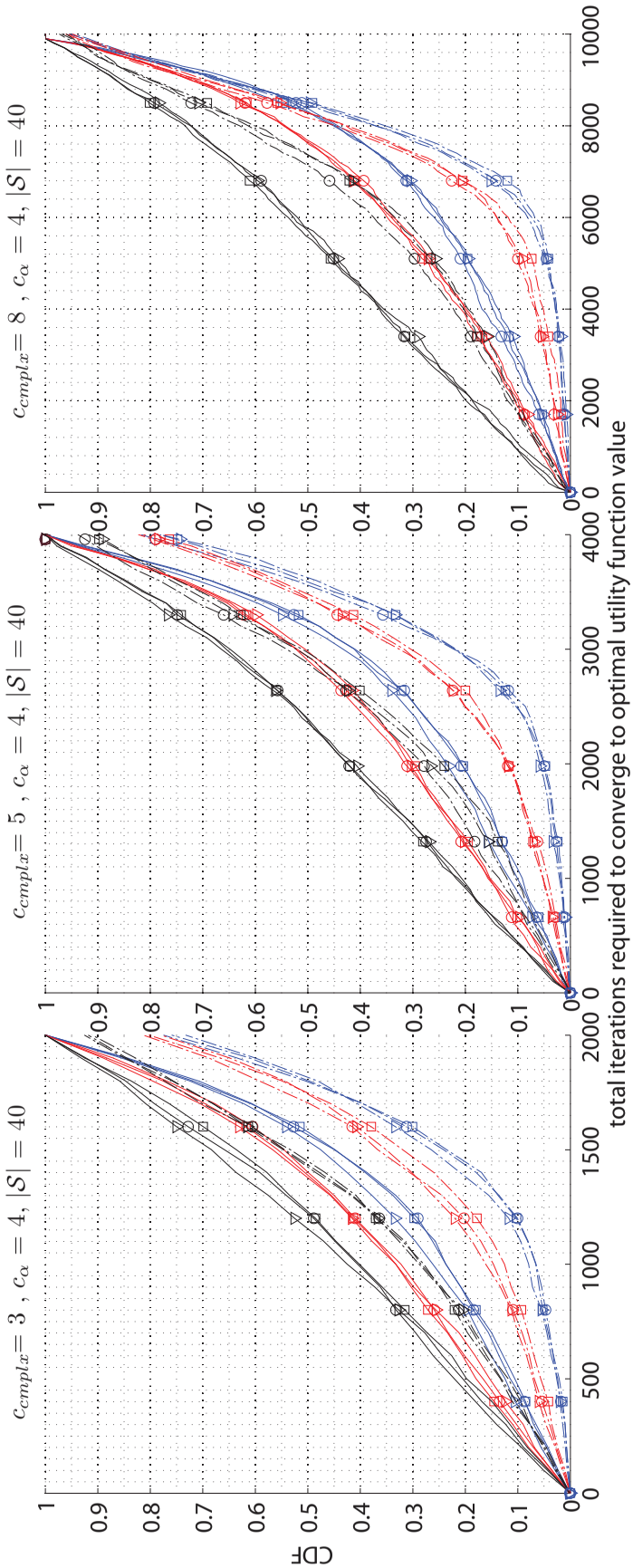

Figure 5.22: Convergence distribution for SA at (4,4,40) for $c_\alpha = 4$ showing poor convergence trends

Figures 5.16 and 5.18 of the (4,2,10) and (4,2,40) cases ($c_\alpha = 4$), as well as figures 5.20 and 5.22 for (4,4,10) and (4,4,40) both show extremely poor convergence, and while the data points on the graphs may be crowded and hard to read, it is not the individual values but the overall trend which is important and hence all the plots are included for comparison. It can be easily seen that $c_\alpha = 1$ and $M_u = 300$ for cases of both (4,2,10) (4,2,40) offers the best convergence performance regardless of decay rates c_α chosen. For (4,2,10) in figure 5.17 at $c_\alpha = 4$, there is some initial convergence for approximately the first quarter of the total iterations for each complexity, and we begin to see similar trends to the ones clearly visible at $c_\alpha = 1$ (with the improved convergence of the dotted lines representing $M_u = 300$) but any increase in convergence quickly drops off and it continues linearly until the end of the execution. For (4,2,40), where the exhaustive search solution complexity is on the order of 10^6, only at the highest $c_\alpha = 8$ do we begin to see any convergence better than linear or sub-linear seen at the lower c_{cmplx}. However, particularly at higher $|\mathcal{S}|$ there the solution space complexity is on the order of 10^6, regardless of the other parameter choices given $c_\alpha = 4$, the algorithm fails show the aggressive convergence seen in the $c_\alpha = 1$ and lower $|\mathcal{S}|$ cases. For $n_T = 4$ $n_R = 4$ and $c_\alpha = 4$ (poor convergence cases) $M_u = 100$ (solid line) gives slightly better convergence than the $M_u = 300$ (dotted lines). However, the convergence of the algorithm in these cases is very poor and was included for the sake of completeness to show trends, and should not be used in practice.

The choice of outer loop parameter as $M_u = 300$ offers noticeably better performance in all cases when it is paired with a low decay rate ($c_\alpha = 1$). This means is that for $n_T = 4$ $n_R = 2$ and $n_T = 4$ $n_R = 4$ SA tends to converge much faster with more aggressive decay rates (lower α) and higher outer loop repetitions M_u (for the same total complexity), with the effect even more prevalent at lower $|\mathcal{S}|$. The general structure of this cooling profile stemming from smaller α and larger M_u is one that features a sharper overall decay profile but smaller discrete steps in temperatures between each subsequent outer iteration (t_u and t_{u+1}). Referring back to Table 4.1 we see that the algorithm is (theoretically) running approximately 73.3% of its execution time with $P_{a,u} < 0.01$. While the actual acceptance probabilities are initially quite higher than 90% and the algorithm actually only runs less than 40% of its total iterations with $P_{a,u} < 0.10$, the trends indicating the benefits of running in a super-cooled state (at least for $n_T = 4$ $n_R = 2$) are evident nonetheless.

Regarding the effect of transition probabilities on convergence, it can be seen that the best convergence is offered by $c_{\text{trans}} = 1$ (curves denoted with circles) in almost all (good convergence) cases, and decreases as $c_{\text{trans}} \to 5$ (curves denoted with triangles). The significance of these results is that for the (4,2) case at all SNR's and other parameter choices offers increasing performance as (see Figure 5.23). While this trend did not hold for the case $n_T = 4$ for both $n_R = 2$ and $n_R = 4$, $|\mathcal{S}| = 40$ and low c_{cmplx}, proper convergence does not occur in these cases and those results are shown only for completeness.

The best performance case of $c_{cmplx} = 1$ indicates corresponding transition probabilities $p_a = 0.20$, $p_u = 0.70$, $p_o = 0.05$, meaning that for the $n_T = 4$, $n_R = 2$ case the algorithm converges to the best solution the fastest when preference is given to swapping currently non-scheduled users as opposed to trying different stream allocations. This makes sense, as for $n_T = 4$, $n_R = 2$ only 3 valid stream allocations $\mathbf{L}^m$ exist, $[2, 2]$, $[2, 1, 1]$, $[1, 1, 1, 1]$ (from Table 2.1). In general, it seems that a more aggressive search to exploit multiuser diversity rather than exploring particular stream allocations fairs better in all cases. This is because while the stream swap operation inherently must schedule users off and on, it appears that directly exploiting multiuser diversity by directly swapping scheduled for non-scheduled users offers the best performance overall.

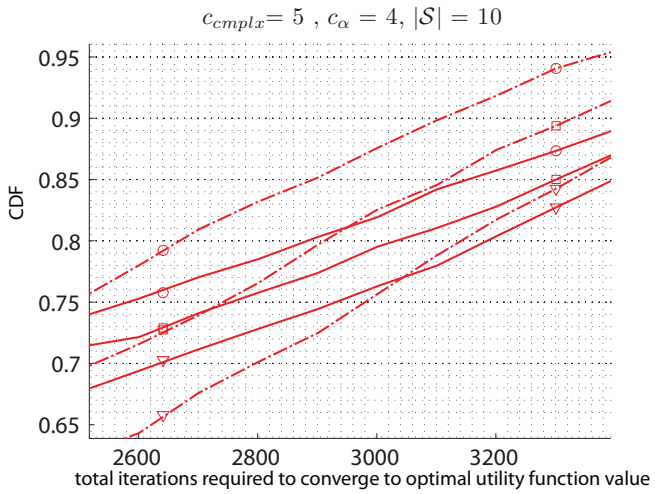

Figure 5.23: Convergence graph showing faster convergence rates as $c_{\text{trans}} = 4 \to 1$

In all cases the simulations at higher SNR converge faster, and this is clearly evident with the black representing SNR=20dB, converging the fastest, followed by red denoting

SNR=10dB, and lastly blue showing SNR = 0dB. It is interesting to note that in fig. 5.17 for $n_T = 4$ $n_R = 2$, $c_{\text{cmplx}} = 5$, $c_{\alpha=1}$, $M_u = 300$ converges at approximately 2000 iterations, which is about the same as the complexity of $c_{\text{cmplx}} = 2$. Similarly, for $c_{\text{cmplx}} = 8$, we see convergence at approximately 6000 iterations. Similar results are seen in fig. 5.21 for $n_T = 4$ $n_R = 4$ for $c_{\text{cmplx}} = 5$ (3000 iterations), and for $c_{\text{cmplx}} = 8$ (5000 iterations). Since in general as $c_{\text{cmplx}} \to 8$ performs better, slightly better performance might be achieved if a higher complexity level is chosen and the execution is terminated when the results converge.

5.5 Complexity Analysis

5.5.1 Complexity Analysis of Matrix operations

Given an $m \times n$ complex-valued matrix $\mathbf{H} \in \mathcal{C}^{m \times n}$, the complexities of various matrix operations are listed below:

- Multiplying an $m \times n$ matrix by an $n \times p$ matrix requires $8mnp$ flops [81].
- The inverse square root of a square $n \times n$ matrix will require $\frac{340}{3}n^3$ flops [81]
- Waterfilling over j eigenmodes requires a maximum of $2j^2 + 6j$ flops [82]
- The full SVD of a matrix $\mathbf{H} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^H$ of an $m \times n$ matrix ($m \geq n$) requires $16m^2n + 32mn^2 + 36n^3$ flops [81]
- An SVD requiring only the left singular values $\mathbf{U}$ and singular values $\mathbf{\Sigma}$ requires $4m^2n - 8mn^2$ flops [81]

5.5.2 Complexity Analysis of Exhaustive search

In this section, we determine the complexity of the exhaustive search for user and data stream scheduling under successive zero-forcing precoding with coordinated transmit-receiver processing, allowing up to n_R data streams per user and assuming that n_T total data streams are always scheduled. We assume $n_R \leq n_T$ and $|\mathbf{L}^m| > 1$ so therefore $l_j < n_T \forall j$. The work done in this section was done with the extensive help of Dr. Robert Elliott and uses partial results from his previously published work on successive zero-forcing [52].

Given a data stream allocation of $\mathbf{L}^m$, a set of users $\mathcal{S}_T$, and an order $\pi(j)$ (for simplicity denoting $\pi(j) = j$) denoting the jth encoded user, we calculate the sum-rate according to

the following steps: (i) the effective channels (2.18) for each of the j users are calculated; (ii) the covariance matrices $\mathbf{P}_j$ for the dual MAC are then calculated using the iterative method in [24]; (iii) a MAC to BC conversion detailed in [9] is used to obtain the DPC matrices Σ_j; (iv) the DPC matrices are converted to the SZF matrices using the method presented in [38]; (v) the rate calculation (2.14) is then calculated for each user j.

The complexity of the steps is illustrated below:

- For the first step (i), left singular vectors $\mathbf{U}$ are calculated for each of the $|\mathbf{L}^m|$ scheduled users, requiring $16n_T n_R^2 + 32n_R^3$ flops. Note: this differs from the last bullet in 5.6.1 because we are calculating the $\mathbf{V}$ of $\mathbf{H}^H$, since m (i.e n_r) is less than n (n_T), rather than calculating $\mathbf{U}$ of $\mathbf{H}$. The effective channels for each user j are created by multiplying each user's channel by the first l_i left singular vectors for each user, requiring $\sum_{n=1}^{|L^m|} 8l_n n_R n_T$ flops. For the worst case complexity, it can be shown that the allocation of $|\mathbf{L}|^m| = n_T$ users with a single data stream each ($l_i = 1\ \forall i$) dominates the complexity, and is $\mathcal{O}(n_r n_T^2)$

- Steps (ii) through (iv) were analyzed in [52] and the complexity of (ii) and (iii) was shown to be dominated by the matrix inverse square root operation. Extending that analysis, we can see that the complexity of the MAC covariance calculation and MAC to BC conversion for each user, is $\mathcal{O}(n_T^3)$ as it is dominated by the *columns* (and therefore n_T) and not the rows (or l_j) of the effective channel matrices. The biggest difference is the calculation of the SVDs in step (iv) to find the nullspace basis vectors, which depends on the size of the effective channel matrices and is dictated by the stream allocation $\mathbf{L}^m$. For the worst general case of any $\mathbf{L}^m$ in any order, the dominant term in the complexity of the SVD operations can then be given by $\sum_{G=1}^{|\mathbf{L}^m|-1} n_T^2(\sum_{n=1}^{G-1} l_n)$. Therefore for step (iv) and the nullspace calculation and projection for each of the $|\mathbf{L}^m| - 1$ users encoded after the first user, knowing that $\sum_{n-1}^{G-1} l_n = n_T - \sum_{n=G}^{|\mathbf{L}^m|} l_n$ and because by definition $\sum_{r=1}^{|\mathbf{L}^m|-1} l_r < n_T$, we can simplify, showing that the dominant term in the complexity is $n_T^2(n_T - \sum_{n=1}^{n_T} l_n)$ for SVD's. Further simplification yields $\mathcal{O}(|\mathbf{L}^m| n_T^3)$. Thus, steps (ii) through (iv) each have complexity $\mathcal{O}(|\mathbf{L}^m| n_T^3)$.

- The rate calculation for step (v) is unchanged from the complexity given in [52], as the dimensions of the aggregate effective channel matrices for the coordinated transmit

and receiver processing method (2.19) must be by definition equal in order to satisfy the dimensionality constraints of SZF (Eq. 2.13). This step is of lower complexity than (ii)-(iv).

- Therefore, for one particular $\mathbf{L}^m$, $\mathcal{S}_T$, π, the complexity is $\mathcal{O}(|\mathbf{L}^m|n_T^3)$. For the exhaustive search, for a set of $\mathcal{S}$ users, all possible allocations $\mathbf{L}^m$, and all possible subsequent orders π must be attempted on all possible user selections, with the total number of possible solutions given by (2.20). Since the number of possible permutations ($|\mathbf{L}^m|!$) of an allocation $\mathbf{L}^m$ varies with the number of scheduled users, from the exhaustive search in the worst case complexity for the exhaustive search occurs when $|\mathbf{L}^m| = n_T$ users are scheduled with $l_j = 1$ streams each. Using this most dominant term, the exhaustive search complexity can be extended from the analysis in [52] to $\mathcal{O}(|\mathbf{L}^m|!\binom{|\mathcal{S}}{|\mathbf{L}^m|})|\mathbf{L}^m|n_T^3) \cong \mathcal{O}(|\mathcal{S}|^{n_T}n_T^4)$.

5.5.3 Complexity Analysis of Simulated Annealing

In this section,we determine the complexity of our Simulated Annealing algorithm. The various operations performed by the SA algorithm such as the neighbourhood function generation and cooling are simple integer arithmetic operations $\mathcal{O}(\infty)$ operating on the solutions ω and $\mathbf{T}$ and have negligible complexity. The computational complexity of SA lies primarily in the $M_u \times M_m$ total loop iterations and subsequent utility function or rate calculations the SZF sum-rate in each iteration. We determined in the exhaustive search complexity section that the complexity of one utility rate calculation to be $\mathcal{O}(|\mathbf{L}^j|n_T^3)$ flops. Therefore the worst case complexity of the SA algorithm is from the $M_u \times M_m$ times the rate is calculated and thus can be given by $\mathcal{O}(M_u M_m|\mathbf{L}^j|n_T^3)$.

5.5.4 Simulation Times for Simulated Annealing

Average simulation times for the compiled MATLAB code (compiled using the *mcc* MATLAB compiler with the *–singlethread* flag) running on a single core of a 2.50GHz processor on the Checkers node as described in Section 3.2 for $N_{samp} = 1$ at various transmit and receive antenna cases (n_R,n_T) = $(4,2)$, $(4,4)$, and $(8,2)$ for each complexity index (from Table 5.1) are presented in table 5.5.4. As we can see from the simulation times, the com-

plexity of the code scaled linearly with little or no overhead for practically all antenna cases and complexity indexes.

	Time per iteration (minutes)		
complexity index c	$n_T = 4$, $n_r = 2$	$n_T = 4$, $n_r = 4$	$n_T = 8$, $n_r = 2$
1	0.10	0.12	0.20
2	0.20	0.24	0.40
3	0.36	0.48	0.8
4	0.62	0.86	1.2
5	0.84	1.00	1.6
6	1.04	1.26	2.0
7	1.50	1.80	2.8
8	2.16	2.46	4.06

Table 5.6: Worst-case simulation times for ($N_{samp} = 1$) Monte Carlo iterations for different n_T, n_R

Chapter 6

Conclusion and Future Work

6.1 Summary of Work and Its Contributions

With MIMO techniques currently being employed in current 4G systems and being proposed for the next generation beyond 4G systems, investigation of scheduling algorithms presents a very active area for research.

In this thesis, we investigated the problem of data stream scheduling under SZF precoding, while allowing a variable number of data streams per user by also employing coordinated transmitter-receiver processing. Although it is shown that the extra degrees of freedom available by allowing for a variable number of data streams per user offer substantial gains in achievable throughput, these potential improvements come at the cost of an additional layer of complexity in choosing the allocation of data streams for each user. The optimal, exhaustive search of all possible data stream allocations per user and all possible eigenmodes of each target user is shown to be intractable even for relatively small numbers of users. We showed that transmitting on the maximal eigenmodes helps remove a combinatorial layer of complexity from the problem, while still giving us practically matching performance to that of the exhaustive eigenmodes search, particularly in environments where a large pool of users requesting service is present (as is the case in the latter part of our simulations).

However, the exhaustive search for the maximal eigenmode variable data stream allocation scheduling problem in systems employing SZF with coordinated transmitter-receiver processing is still impractical, particularly due to the order sensitive nature of the precoding method. To this end we have proposed and investigated the application of simulated

annealing (SA) as a simplified heuristic approach to the maximal eigenmode data stream and user scheduling problem. In order to select parameters for simulated annealing which would perform well, we first derived a simplified tuning mechanism that could be used to derive parameters for any number of transmitter or receiver antennas and any number of users.

We chose to investigate maximal throughput scheduling criteria using an unweighted rate (or equivalently a weighted sum-rate with equal weight per user), although the flexible structure of SA and the utility function allows for the maximization of any arbitrary utility function. The complexity of SA increases linearly with the number of antennas and users, and only depends on the complexity of the utility function calculation. We investigate the algorithm for the optimization of maximizing sum-rate in a single cell MIMO system and show that the proposed algorithm offers a radically simpler solution than the intractable exhaustive search, while providing sum-performance of up to 93% that of the exhaustive search in the case of $n_T = 4$ $n_R = 2$. Similar performance is demonstrated for the $n_T = 4$ $n_R = 4$ case. As the complexity of the exhaustive search scheduling increases as the number of transmitter antennas is increased to $n_T = 8$, for $n_R = 2$ and low numbers of users, our SA scheduling algorithm is able to perform the random search until a crossover at approximately 15 dB. At larger numbers of users however, the sheer complexity of the solution space causes the SA to under-perform the random search at the same complexity.

In all cases, our reduced-complexity variable data stream scheduling approach has been shown to offer superior sum-rate performance over that of exhaustive user scheduling approaches, where only user scheduling is performed (no data stream scheduling is used and each user receives the maximal n_R possible streams). The performance of SA was shown to be less sensitive to parameter selection at lower SNR, but displayed greater sensitivity to the effect of parameter selection on achievable throughput at high SNR. Since most of the work done for scheduling algorithms under SZF precoding assumes n_R data streams per user, we are able to outperform existing schedulers operating with SZF precoding. While the complexity of the exhaustive user scheduling search in most $n_T = 4, 8$ and $n_R = 2$ antenna cases for $K > 10$ is below that of our data stream scheduling algorithm, our SA approach provides a reasonable performance-complexity tradeoff. It was also found that SA did not provide the best solution in terms of complexity and performance for all of the

cases we simulated. It was found that for the case of $n_T = 8$ $n_R = 2$ and $K = 10$, SA at the highest complexity we simulated (10,000 total iterations) barely managed to match the performance of exhaustive user allocation (5020 total iterations required).

The convergence of SA was examined, and it was seen that the careful selection of parameters could drastically shorten time necessary for SA to converge. We also investigated the effects of different parameters on the convergence rates of the algorithm and developed basic guidelines for the selection of parameters to help facilitate good performance. It was shown that in some cases the best and fastest results could be obtained by choosing larger initial complexity targets $M_u \times M_m$ and prematurely terminating the execution.

A detailed analysis of the complexity of the exhaustive algorithm was carried out, with that analysis then extended to find the complexity of the SA algorithm. Simulation times for our SA algorithm were presented, and it was shown that our code scales linearly with the number of Monte Carlo iterations required for each of the $M_u \times M_m$ utility rate calculations required for each iteration. In summary, this work demonstrated that simulated annealing algorithms can, when properly structured, be a viable method of scheduling of both data streams and users for multiuser MIMO systems.

6.2 Future Work

One particular attribute of our SA algorithm is the need for an initial training phase, in which the characteristics (delta-rates for our case of using a sum-rate metric) of the solution space are gathered for use in the subsequent derivation of the operational parameters. While these delta-distributions are saved and can be reused for similar simulation environments, there undeniably is an appeal to an approach that could be applicable to a broader problem without requiring the initial tuning phase for each different problem environment. To this end, future work may include investigation into adaptive selection techniques for the simulated annealing parameter, which have shown [68] to offer promising performance in a variety of settings.

Concerning the utility function and possible alternatives, the complexity of our SA approach is primarily in the evaluation of the sum-rate utility function that was chosen. An avenue for further research could therefore lie in simpler utility function metrics, such as orthogonality or channel gains in the null-space of the aggregate user channels, and could

be investigated for a performance complexity tradeoff. Since we do not look at fairness and instead choose an unweighted sum-rate as the utility function for optimization, a weighted sum-rate could also be chosen and the fairness of SA investigated using more advanced channel models with path loss and shadowing components.

While SA does give good performance in cases with smaller number of transmit antennas, the failure of the algorithm for $n_T = 8$ and large number of users shows the potential for the development and evaluation of a simpler greedy data stream scheduler that could possible perform better. The extension of existing greedy algorithms for SZF (such as the greedy algorithm proposed in [52]) designed for full data stream per user allocation to the variable data stream allocation is an open problem that may be an interesting area of research.

The assumption of full CSIT in our research presents a potentially unrealistic feedback problem for large numbers of users in practical systems. It is known that the effects of imperfect or limited channel knowledge severely impact the ability of precoding methods to support multiple users without severe losses on the achievable rate, and thus the investigation of limited channel information on our SA scheduler would be a good subject for future work. The investigation of the effects of the imperfect channel knowledge on the coordinated transmitter and receiver processing with SZF also have not been investigated and remains an open problem of significant interest, particularly because it is the multi-stream diversity that allows us to achieve performance above that of existing full allocation SZF algorithms.

A particularly interesting topic in wireless research is the idea of network coordination, or coordinated multi-point (CoMP), which tackles the problem of inter-cell interference by which the base stations in several cells can be thought of as a distributed MIMO array. In network coordination schemes, the base stations for several cells mitigate inter-cell interference by coordination of transmissions such that data sent to users in one cell does not cause interference with the reception of data of other users in adjacent cells. Our SA approach to data stream scheduling for a single-cell MIMO downlink can be expanded to the case of multi-cell network MIMO downlink, and presents a promising direction for future work.

Bibliography

[1] "Ericsson mobility report," Ericsson, Tech. Rep., Nov. 2012.

[2] J. Winters, "Optimum combining in digital mobile radio with cochannel interference," *IEEE J. Sel. Areas Commun.*, vol. 2, no. 4, pp. 528 – 539, July 1984.

[3] J. Salz, "Digital transmission over cross-coupled linear channels," *AT&T Technical Journal*, vol. 64, no. 6, pp. 1147 – 1159, July 1985.

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

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

[6] I. E. Telatar, "Capacity of multi-antenna Gaussian channels," *European Trans. Telecommun.*, vol. 10, no. 6, pp. 585–595, Nov.-Dec 1999.

[7] S. Alamouti, "A simple transmit diversity technique for wireless communications," *IEEE J. Sel. Areas Commun.*, vol. 16, no. 8, pp. 1451 – 1458, Oct. 1998.

[8] L. Zheng and D. Tse, "Diversity and multiplexing: a fundamental tradeoff in multiple-antenna channels," *IEEE Trans. Inf. Theory*, vol. 49, no. 5, pp. 1073 – 1096, May 2003.

[9] S. Vishwanath, N. Jindal, and A. Goldsmith, "Duality, achievable rates, and sum-rate capacity of Gaussian MIMO broadcast channels," *IEEE Trans. Inf. Theory*, vol. 49, no. 10, pp. 2658–2668, Oct. 2003.

[10] Q. Li, G. Li, W. Lee, M. il Lee, D. Mazzarese, B. Clerckx, and Z. Li, "MIMO techniques in WiMAX and LTE: a feature overview," *IEEE Commun. Mag.*, vol. 48, no. 5, pp. 86–92, May 2010.

[11] D. Gesbert, M. Kountouris, R. W. Heath, C. Chae, and T. S. Sälzer, "From single user to multiuser communications: Shifting the MIMO paradigm," in *IEEE Sig. Proc. Magazine*, vol. 24, no. 5, Sept. 2007, pp. 36–46.

[12] D. Love, J. Heath, R.W., W. Santipach, and M. Honig, "What is the value of limited feedback for MIMO channels?" *IEEE Commun. Mag.*, vol. 42, no. 10, pp. 54 – 59, Oct. 2004.

[13] M. Sharif and B. Hassibi, "On the capacity of MIMO broadcast channels with partial side information," *IEEE Trans. Inf. Theory*, vol. 51, no. 2, pp. 506 – 522, Feb. 2005.

[14] G. Caire and S. Shamai, "On the achievable throughput of a multiantenna Gaussian broadcast channel," *IEEE Trans. Inf. Theory*, vol. 49, no. 7, pp. 1691 – 1706, July 2003.

[15] H. Weingarten, Y. Steinberg, and S. Shamai, "The capacity region of the Gaussian multiple-input multiple-output broadcast channel," *IEEE Trans. Inf. Theory*, vol. 52, no. 9, pp. 3936–3964, Sept. 2006.

[16] D. Tse and P. Viswanath, *Fundamentals of Wireless Communication*, ser. Wiley Series in Telecommunications. Cambridge University Press, 2005.

[17] W. Yadum and N. Nakajima, "The correlation of diversity MIMO antenna for portable terminals," *Wirel. Commun. Mob. Comput.*, vol. 7, no. 8, pp. 995–1002, Oct. 2007.

[18] E. Biglieri, *MIMO Wireless Communications*. Cambridge University Press, 2007.

[19] W. Yu, W. Rhee, S. Boyd, and J. Cioffi, "Iterative water-filling for gaussian vector multiple-access channels," *IEEE Trans. Inf. Theory*, vol. 50, no. 1, pp. 145–152, Jan. 2004.

[20] D. Love, R. Heath, V. Lau, D. Gesbert, B. Rao, and M. Andrews, "An overview of limited feedback in wireless communication systems," *IEEE J. Sel. Areas Commun.*, vol. 26, no. 8, pp. 1341 – 1365, Oct. 2008.

[21] L. Li, W. Jing, and W. Xiaoyun, "ZF beamforming performance analysis for multiuser spatial multiplexing with imperfect channel feedback," in *Proc. Int. Conf. on Wireless Commun., Netw. and Mobile Computing*, Sept. 2007.

[22] M. Costa, "Writing on dirty paper," *IEEE Trans. Inf. Theory*, vol. 29, no. 3, pp. 439 – 441, May 1983.

[23] N. Jindal, S. Vishwanath, and A. Goldsmith, "On the duality of Gaussian multiple-access and broadcast channels," *IEEE Trans. Inf. Theory*, vol. 50, no. 5, pp. 768–783, May 2004.

[24] N. Jindal, S. Vishwanath, S. Jafar, and A. Goldsmith, "Sum power iterative water-filling for multi-antenna Gaussian broadcast channels," *IEEE Trans. Inf. Theory*, vol. 51, no. 4, pp. 1570–1580, Apr. 2005.

[25] M. Tomlinson, "New automatic equaliser employing modulo arithmetic," *Electronics Letters*, vol. 7, no. 5, pp. 138–139, Mar. 1971.

[26] C. Peel, B. Hochwald, and A. Swindlehurst, "A vector-perturbation technique for near-capacity multiantenna multiuser communication-part i: channel inversion and regularization," *IEEE Trans. Commun.*, vol. 53, no. 1, pp. 195–202, Jan. 2005.

[27] M. Mazrouei-Sebdani and W. Krzymień, "Vector perturbation precoding for network MIMO: Sum rate, fair user scheduling, and impact of backhaul delay," *IEEE Trans. Veh. Tech.*, vol. 61, no. 9, pp. 3946–3957, Nov. 2012.

[28] ——, "On MMSE vector-perturbation precoding for MIMO broadcast channels with per-antenna-group power constraints," *IEEE Trans. Signal Process.*, vol. 61, no. 15, pp. 3745–3751, June 2013.

[29] C. Windpassinger, R. F. H. Fischer, and J. Huber, "Lattice-reduction-aided broadcast precoding," *IEEE Trans. Commun.*, vol. 52, no. 12, pp. 2057–2060, Dec. 2004.

[30] W. Yu and J. Cioffi, "Trellis precoding for the broadcast channel," in *Proc. 2001 IEEE Global Telecommun. Conf. (GLOBECOM '01)*, vol. 2, Nov. 2001, pp. 1344–1348.

[31] D. Ryan, I. Collings, I. V. L. Clarkson, and R. Heath, "Performance of vector perturbation multiuser mimo systems with limited feedback," *IEEE Trans. Commun.*, vol. 57, no. 9, pp. 2633–2644, Sept. 2009.

[32] H. Harashima and H. Miyakawa, "Matched-transmission technique for channels with intersymbol interference," *IEEE Trans. Commun.*, vol. 20, no. 4, pp. 774–780, Aug. 1972.

[33] R. Chen, J. Li, C. Li, and W. Liu, "Multi-user multi-stream vector perturbation precoding," *Wireless Pers. Commun.*, vol. 69, no. 1, pp. 335–355, Mar. 2013.

[34] M. Joham, W. Utschick, and J. Nossek, "Linear transmit processing in MIMO communications systems," *IEEE Trans. Signal Process.*, vol. 53, no. 8, pp. 2700–2712, Aug. 2005.

[35] F. Boccardi and H. Huang, "A near-optimum technique using linear precoding for the MIMO broadcast channel," in *Proc. IEEE Acoust. Speech and Signal Process. (ICASSP 2007)*, vol. 3, Apr. 2007, pp. III–17–III–20.

[36] Q. Spencer, A. Swindlehurst, and M. Haardt, "Zero-forcing methods for downlink spatial multiplexing in multiuser MIMO channels," *IEEE Trans. Sigal Proc.*, vol. 52, no. 2, pp. 461–471, Feb.

[37] ——, "Zero-forcing methods for downlink spatial multiplexing in multiuser MIMO channels," *IEEE Trans. Signal Process.*, vol. 52, no. 2, pp. 461 – 471, Feb. 2004.

[38] A. Dabbagh and D. Love, "Precoding for multiple antenna Gaussian broadcast channels with successive zero-forcing," *IEEE Trans. Signal Process.*, vol. 55, no. 7, pp. 3837 – 3850, July 2007.

[39] A. Wiesel, Y. Eldar, and S. Shamai, "Zero-forcing precoding and generalized inverses," *IEEE Trans. Signal Process.*, vol. 56, no. 9, pp. 4409–4418, Sept. 2008.

[40] R. Elliott and W. Krzymien, "Improved and weighted sum rate maximization for successive zero-forcing in multiuser MIMO systems," *EURASIP J. Wireless Commun. and Netw.*, vol. 2011, no. 133, pp. 1–16, Oct. 2011.

[41] R. Elliott, S. Sigdel, W. A. Krzymień, M. Al-Shalash, and A. Soong, "Genetic and greedy user scheduling for multiuser MIMO systems with successive zero-forcing," in *Proc. 5th IEEE Broadband Wireless Access Workshop (co-located with the IEEE GLOBECOM 2009)*, Nov-Dec. 2009, pp. 6 IEEE–format pages.

[42] B. Lee and B. Shim, "A vector perturbation with virtual users for multiuser MIMO downlink," in *Proc. IEEE Acoust. Speech and Signal Process. (ICASSP 2010)*, Mar. 2010, pp. 3406–3409.

[43] P. Tejera, W. Utschick, G. Bauch, and J. Nossek, "Efficient implementation of successive encoding schemes for the MIMO OFDM broadcast channel," in *Proc. IEEE Int. Conf. on Commun. (ICC '06)*, vol. 12, June 2006, pp. 5354–5359.

[44] R. Heath and D. Love, "Multimode antenna selection for spatial multiplexing systems with linear receivers," *IEEE Trans. Signal Process.*, vol. 53, no. 8, pp. 3042–3056, Aug. 2005.

[45] Z. Shi, C. Zhao, and Z. Ding, "Low complexity eigenmode selection for MIMO broadcast systems with block diagonalization," in *Proc. IEEE Int. Conf. on Commun., (ICC '08)*, May 2008, pp. 3976–3981.

[46] Z. Shi, W. Xu, S. Jin, C. Zhao, and Z. Ding, "On wireless downlink scheduling of MIMO systems with homogeneous users," *IEEE Trans. Inf. Theory*, vol. 56, no. 7, pp. 3369–3377, July 2010.

[47] M. Aydin, R. Kwan, J. Wu, and J. Zhang, "Multiuser scheduling on the LTE downlink with simulated annealing," in *Proc. IEEE Veh. Technol. Conf. (VTC 2011-Spring)*, May 2011, pp. 1–5.

[48] R. Kwan, M. Aydin, C. Leung, and J. Zhang, "Multiuser scheduling in HSDPA using simulated annealing," in *Proc. Int. Wireless Commun. and Mobile Comput. Conf. 2008 (IWCMC'08)*, Aug. 2008, pp. 236–241.

[49] J. Chang, L. Tassiulas, and R.-F. F, "Joint transmitter receiver diversity for efficient space division multiple access," *IEEE Trans. Wireless Commun.*, vol. 1, pp. 16–27, Jan. 2002.

[50] C.-B. Chae, D. Mazzarese, T. Inoue, and R. Heath, "Coordinated beamforming for the multiuser MIMO broadcast channel with limited feedforward," *IEEE Trans. Sign. Process.*, vol. 56, no. 12, pp. 6044–6056, Dec. 2008.

[51] S. Sigdel and W. Krzymień, "Efficient user selection and ordering algorithms for successive zero-forcing precoding for multiuser MIMO downlink," in *Proc. IEEE Veh. Technol. Conf. (VTC 2009-Spring)*, April 2009, pp. 1–6.

[52] R. C. Elliott, S. Sigdel, and W. A. Krzymień, "Low complexity greedy, genetic and hybrid user scheduling algorithms for multiuser MIMO systems with successive zero-forcing," *Trans. Emerging Telecommun. Technol.*, vol. 23, no. 7, pp. 604–617, Nov. 2012.

[53] "High speed downlink packet access (HSDPA)," in *3rd Generation Partnership Project (3GPP) Standard V12.x (2013-03)*, 2010, pp. 3406–3409.

[54] S. Dhurandher, S. Misra, H. Mittal, A. Agarwal, and I. Woungang, "Ant colony optimization-based congestion control in ad-hoc wireless sensor networks," in *Proc. International Conference on Computer Systems and Applications, (AICCSA 2009)*, May 2009, pp. 492–497.

[55] T.-Y. Lin, K.-C. Hsieh, and H.-C. Huang, "Applying genetic algorithms for multiradio wireless mesh network planning," *IEEE Trans. Veh. Tech.*, vol. 61, no. 5, pp. 2256–2270, June 2012.

[56] C.-K. Ting, C.-N. Lee, H.-C. Chang, and J.-S. Wu, "Wireless heterogeneous transmitter placement using multiobjective variable-length genetic algorithm," *IEEE Trans. Syst. Man, Cybern. B, Cybernetics*, vol. 39, no. 4, pp. 945–958, Aug. 2009.

[57] R. Elliott and W. Krzymien, "Downlink scheduling via genetic algorithms for multiuser single-carrier and multicarrier MIMO systems with dirty paper coding," *IEEE Trans. Veh. Tech.*, vol. 58, no. 7, pp. 3247–3262, Sept. 2009.

[58] ——, "On the convergence of genetic scheduling algorithms for downlink transmission in multi-user mimo systems," *Wireless Pers. Commun.*, vol. 58, no. 3, pp. 469–481, June 2011.

[59] M. Gendrea and J.-Y. Potvin, *Handbook of Metaheuristics(2nd ed.)*. London, UK: Springer-Verlag, 2010.

[60] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi, "Optimization by simulated annealing," *Science*, vol. 220, no. 4598, pp. 671 – 680, May 1983.

[61] D. Mitra, F. Romeo, and A. Sangiovanni-Vincentelli, "Convergence and finite-time behavior of simulated annealing," *Advances in Applied Probability*, vol. 18, no. 3, pp. 747–771, Sept. 1986.

[62] F. Glover, "Tabu search, part I," *ORSA Journal on Computing*, vol. 1, no. 3, pp. 190 – 206, Summer 1989.

[63] Z. Michalewicz, *Genetic Algorithms + Data Structures = Evolution Programs (3rd ed.)*. London, UK: Springer-Verlag, 1996.

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

[65] L. Goldstein and M. Waterman, "Neighborhood size in the simulated annealing algorithm," *Am. J. Math. Manage. Sci.*, vol. 8, no. 3-4, pp. 409–423, Jan. 1988.

[66] K. M. Cheh, J. B. Goldberg, and R. G. Askin, "A note on the effect of neighborhood structure in simulated annealing," *Computers & Operations Research*, vol. 18, no. 6, pp. 537 – 547, 1991.

[67] H. Sanvicente-Sánchez and J. Frausto-Solís, "A method to establish the cooling scheme in simulated annealing like algorithms," in *Proc. 2004 Int. Conf. Comput. Sci. and Its Applicat. (ICCSA 2004)*, vol. 3045. Springer, May 2004, pp. 755–763.

[68] N. Azizi and S. Zolfaghari, "Adaptive temperature control for simulated annealing: a comparative study," *Computers & Operations Research*, vol. 31, no. 14, pp. 2439 – 2451, Dec. 2004.

[69] K. Boese and A. Kahng, "Simulated annealing of neural networks: the "cooling" strategy reconsidered," in *Proc. IEEE International Symposium on Circuits and Systems, (ISCAS '93)*, May 1993, pp. 2572–2575.

[70] A. Anagnostopoulos, L. Michel, P. V. Hentenryck, and Y. Vergados, "A simulated annealing approach to the traveling tournament problem," *J. of Scheduling*, vol. 9, no. 2, pp. 177–193, Apr. 2006.

[71] I. Osman, "Metastrategy simulated annealing and tabu search algorithms for the vehicle routing problem," *Annals of Operations Research*, vol. 41, no. 4, pp. 421–451, 1993.

[72] D. Stefankovic, S. Vempala, and E. Vigoda, "Adaptive simulated annealing: A near-optimal connection between sampling and counting," in *Proc. IEEE Symposium on Foundations of Computer Science, (FOCS '07)*, Oct. 2007, pp. 183–193.

[73] N. Prajapati, R. R. Agravat, and M. Hasan, "Comparative study of various cooling schedules for location area planning in cellular networks using simulated annealing," in *Proc. Int. Conf. on Networks and Communications, (NETCOM '09)*, Dec. 2009, pp. 146–150.

[74] B. Hajek, "Cooling schedules for optimal annealing," *Mathematics of Operations Research*, vol. 13, no. 2, pp. 311–329, May 1988.

[75] D. E. Jeffcoat and R. L. Bulfin, "Simulated annealing for resource-constrained scheduling," *European Journal of Operational Research*, vol. 70, no. 1, pp. 43 – 51, Oct. 1993.

[76] V. Rayward-Smith, *Modern heuristic search methods*. John Wiley & Sons, Inc., 1996.

[77] S. White, "Concepts of scale in simulated annealing," in *Proc. IEEE Int. Conf. on Computer Design*, 1984, pp. 646–651.

[78] J. Schneider and S. Kirkpatrick, *Stochastic Optimization*, ser. Scientific Computation. Springer-Verlag Berlin / Heidelberg, 2006.

[79] Z. Michalewicz and D. Fogel, *How to Solve It: Modern Heuristics*. Springer, 2000.

[80] K. A. Dowsland, "Modern heuristic techniques for combinatorial problems," C. R. Reeves, Ed. New York, NY, USA: John Wiley & Sons, Inc., 1993, pp. 20–69.

[81] G. Golub and C. Van Loan, *Matrix Computations*, ser. Johns Hopkins Studies in the Mathematical Sciences. Johns Hopkins University Press, 1996.

[82] Z. Shen, R. Chen, J. Andrews, R. Heath, and B. Evans, "Low complexity user selection algorithms for multiuser mimo systems with block diagonalization," *IEEE Trans. Signal Process.*, vol. 54, no. 9, pp. 3658–3663, Aug. 2006.

[83] J. Proakis and M. Salehi, *Digital Communications, 5th ed.* McGraw-Hill Higher Education, 2008.

Appendix A

Validation of Simulation Model and Results

A.1 Validation of Simulation Model

In order to check the validity of our simulations, we examine the following aspects of work:

- As the transmission environment in this work assumed a Rayleigh fading channel model, for the purposes of confirming the statistical validity of our simulations we first examine the complex channel gains distribution to ensure the actual distribution of the values used matches the assumed theoretical channel model.
- Secondly, due to the nature of the Monte Carlo simulations in which many independent realizations are required, and due the distributed computing environment used to simulate the results, we briefly go over methodology used to generate ensure random and independent pseudorandom sequences for the thousands of simulated used simultaneously.
- The errors in the Monte Carlo results are subsequently presented and examined.
- Finally, we offer a comparison to published results in literature.

A.1.1 Channel Model Verification

In all of the results provided in this thesis we assumed a Rayleigh fading channel model, which represents a rich scattering transmission environment where the received signal has no

Gaussian random variable x		Rayleigh random variable r	
Theoretical	MATLAB generated	Theoretical	MATLAB generated
$\mu_x = 0$	$\mu_{x,samp} = -0.0014$	$\mu_r = 0.8862$	$\mu_{r,samp} = 0.8870$
$\sigma_x = 1$	$\sigma_{x,samp} = 0.9980$	$\sigma_r = 0.2146$	$\sigma_{r,samp} = 0.2143$

Table A.1: Theoretical and generated (200,000 samples) mean and variance for Gaussian and Rayleigh distributions

significant line-of-sight component. The resulting channel gains should therefore be Gaussian distributed values with zero mean and unit variance [18, 83]. Then AWGN model assumed for noise also assumes i.i.d circularly symmetric complex Gaussian random variables with 0 mean and σ_n^2 variance per dimension. The importance of random variables in our modeling then highlights the necessity of obtaining accurate representations of the random variables.

The Gaussian random variables were generated in MATLAB using the *randn(⋯)* function, which generates pseudorandom values (see appendix A.1.2 for details relating to the random number generation and seeding) drawn from the standard normal distribution. A Gaussian random variable x with mean μ_x and variance σ_x^2 has a PDF [18, 83]

$$p_x(x) = \frac{1}{\sigma_x\sqrt{2\pi}}\, e^{-(x-\mu_x)^2/2\sigma_x^2} \tag{A.1}$$

and cumulative distribution function (CDF) [18, 83]

$$F_x(x) = \frac{1}{\sqrt{\pi}} \int_{-\infty}^{x-\mu_x} e^{-t^2}\, dt. \tag{A.2}$$

Figure A.1.1 displays the approximation of the Gaussian distribution obtained using MATLAB and 200,000 samples (coloured lines) and the theoretical Gaussian distribution (dashed line). For this set of samples the resulting mean and variance were (see Table A.1.1) $\mu_{x,samp} = -0.0014$ and $\sigma_{x,samp} = 0.9980$ respectively, very close to the expected value of $\mu_x = 0$ and $\sigma_x = 1$. The PDF and CDF of the MATLAB generated distribution also closely follow the theoretical Gaussian distribution. Although generating a Rayleigh channel with fading envelope magnitudes that are Rayleigh distributed requires complex-valued Gaussian random variables with $\mu = 0$ and $\sigma = 1$, these can be generated from i.i.d. real valued Gaussian random variables x_R and x_I (with $\mu_R = \mu_I = 0$ and $\sigma_R = \sigma_I = 1$) by

$$r = \frac{1}{\sqrt{2}}(x_R + j\, x_I). \tag{A.3}$$

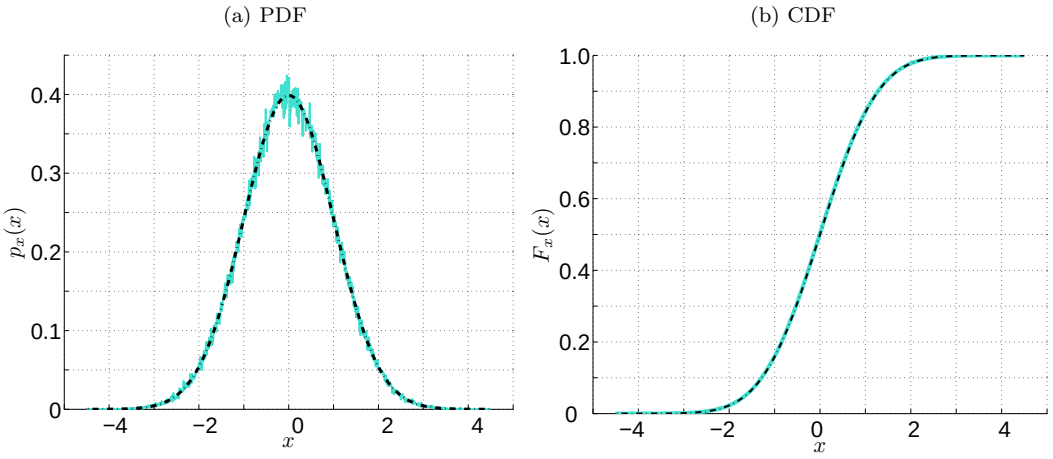

Figure A.1: Theoretical Gaussian distribution (dashed lines) compared with Gaussian distribution generated by MATLAB (coloured) with 200 000 samples

Similarly, a Rayleigh random variable r with a scale parameter σ_x the has a probability distribution function (PDF) [18, 83]

$$p_r(r) = \frac{r}{\sigma_x^2} e^{-r^2/2\sigma_x^2} \tag{A.4}$$

and cumulative distribution function (CDF) [18, 83]

$$F_r(r) = 1 - e^{-r^2/2\sigma_x^2} \tag{A.5}$$

where $\sigma_x^2 = 1/2$ denotes the variance of the Gaussian real and imaginary parts of the complex random variables.

Likewise, Figure A.1.1 shows the theoretical Rayleigh distribution (dashed line) closely tailed by the distribution generated by MATLAB (200,000 samples). The magnitude of the generated Rayleigh random variables had a mean (see Table A.1.1) $\mu_{r,samp} = 0.2143$ and variance $\sigma_{r,samp} = 0.8870$, very closely matching the theoretically expected mean and variance of $\mu_r = \sigma_x\sqrt{\pi/2} = 0.8862$ and $\sigma_r^2 = \sigma_x^2(2 - \pi/2) = 0.2146$. It can therefore be seen from our closely matching means, variances, and overall distributions for our generated random variables that our methodology for modeling the channels is accurate.

A.1.2 Pseudorandom Generator Seeding

All of the simulation results in this thesis were determined using the Monte Carlo method using the MATLAB software package on the distributed computing resources provided by

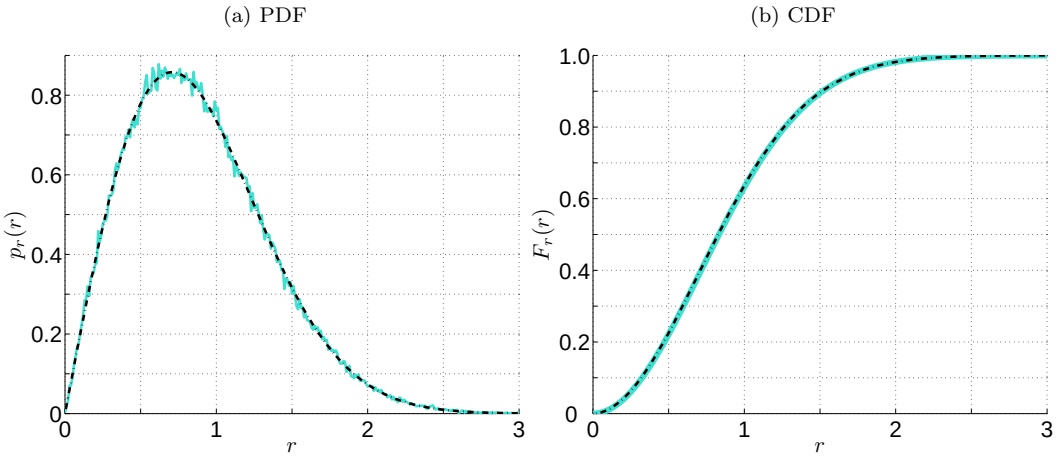

Figure A.2: Theoretical Rayleigh (a)PDF and (b)CDF distribution (dashed lines) compared with Rayleigh distribution generated by MATLAB (coloured) with 200 000 samples

WestGrid/Compute Calcule Canada and a desktop machine provided by TRTech (formerly Telecommunications Research Laboratories, TRLABS). Due to the excessive number of CPU hours required to simulate all of the N_{samp} Monte Carlo samples used to estimate the ergodic capacity on one machine, we chose to distribute the N_{samp} samples over the distributed computing environment (described in section 3.2) provided by WestGrid. Dependent on the availability of free computing nodes (n_{nodes}), the total number of samples would be split between the n_{free} nodes and run in parallel. However, an issue arose in that MATLAB by default initializes the pseudorandom sequence generator used by all of the random functions to a default seed value that was the same across all of the nodes. Due to the parallel nature of our computations, a possible solution of initializing the pseudorandom generator seed based on the system time also proved infeasible due to the fact that depending on the cluster load simulations would sometimes start at the exact same time. When this occurred, the simulations would share the same random seed, resulting in the same channels being generated for the users and identical execution of the algorithm, and result in repeated data. This would result in wasted computing resources and a smaller pool of useable data.

Our solution was to initialize the pseudorandom seed based on the unique job identifier given by the Portable batch system (PBS) scheduler used by the cluster, *$PBS_JOB_ID* multiplied by the current system time, and checked against a database of previously used seeds. If by chance the seed was a duplicate of one already used for that batch of

simulations, another value was generated in the same manner until a unique value was found. Additionally, the storage of the random seeds also allowed us to, if necessary, replicate the channels and environment of our simulations to facilitate direct comparisons of parameters under the same runtime conditions, as well as enabling us to re-run results.

A.1.3 Monte Carlo Error

Recall from sections 2.1.3 and 2.2.3, where for different encoding methods (DPC, SZF) the achievable instantaneous rates for specific channel realizations are given. The aforementioned formulas give the instantaneous rates achievable in a particular channel realization whereas the results presented in this thesis are typically an average, or an ergodic capacity, which is the expected value obtained by averaging over a large number of channel realizations. While the true ergodic capacity could be calculated by averaging over the infinite number of possible realizations, since it is impossible to simulate over all possible channel realizations for all users (of which there would be an infinite amount of unique realizations), it is accepted in practice to sample and average a large number of independent simulations instead of attempting to obtain the true ergodic mean. Although this will result in an inevitable intrinsic error between the reported sample mean and the true ergodic mean, this error can be made arbitrarily small by simply choosing an appropriately large sample size. This error in the sampled mean can then be visualized through the use of error bars on the data which can indicate the uncertainty (one standard error, a specific confidence (xx% interval), or one standard deviation of uncertainty) in the displayed data and can be used to visually compare two quantities and determine whether differences are statistically significant. For completeness, all simulation results should include error bars, however their practiced omission in this thesis mirrors convention followed in other published works for simulations of these types. Their omission is justified however, if the chosen sample size is large enough that the resulting error and error bars would not be clearly visible in the figures.

The standard error of the mean is the standard deviation of the sampling distribution of the sample mean's estimate of a population ($\sigma_{\bar{x}}$) mean. It is usually estimated via the sample's estimate of the standard deviation of the population mean divided by the square

P	x
0.800	1.28155 $\sigma_{\bar{x}}$
0.900	1.64485 $\sigma_{\bar{x}}$
0.950	1.95996 $\sigma_{\bar{x}}$
0.990	2.57583 $\sigma_{\bar{x}}$
0.995	2.80703 $\sigma_{\bar{x}}$
0.999	3.29053 $\sigma_{\bar{x}}$

Table A.2: Confidence interval probabilities and spreads

root of the sample size,

$$\sigma_{\bar{x}} = \frac{\sigma}{\sqrt{n}}. \tag{A.6}$$

The confidence interval and the associated probability for the sample mean (the ergodic throughput of our system), is defined as the interval in which the true mean measurement falls with the respective probability, can then be found from the standard error of the mean. If the sample size is sufficiently large, by the central limit theorem the distribution of the sample means will approach a normal distribution. Therefor, for a normal (Gaussian) distribution the probability that a measurement falls within n standard deviations ($n\sigma$) of the population mean μ can be given by

$$P(\mu - n\sigma < x < \mu + n\sigma) = erf(n/\sqrt{2}) \tag{A.7}$$

where erf is the error function for the normal distribution. The sigma distance for several confidence intervals are given in Table A.1.3.

The confidence interval (or standard deviation of the sampled mean, $\sigma_{\bar{x}}$) can then be narrowed simply by simply increasing the number of samples n, thereby affording a higher probability over the same interval than a sample with lower n, or the same probability but over a much narrower interval. In our simulations, we ran Monte Carlo simulations for the SA algorithm simulations, for the exhaustive search, and for the random searches. Looking back on our results, we illustrate the effects of varying sample sizes on the confidence intervals in small to large user environments, and low to high SNR. Specifically, the cases of (4,4) (where (n_T, n_R)) for $|\mathcal{S}| = 10 - 40$, and SNR $= 0, 20$ dB are examined in greater detail in figures A.3 and A.4 Since the majority of our simulations were done at 2000 iterations, the figures denoting the nsamp = 2000 case with $\pm 0.0507 bits/s/Hz$ at 20 dB and $\pm 0.0177 bits/s/Hz$

at 0dB are fairly representative of our work. Overall, the average 95% confidence interval encountered in our work was about $\pm 0.04 bits/s/Hz$.

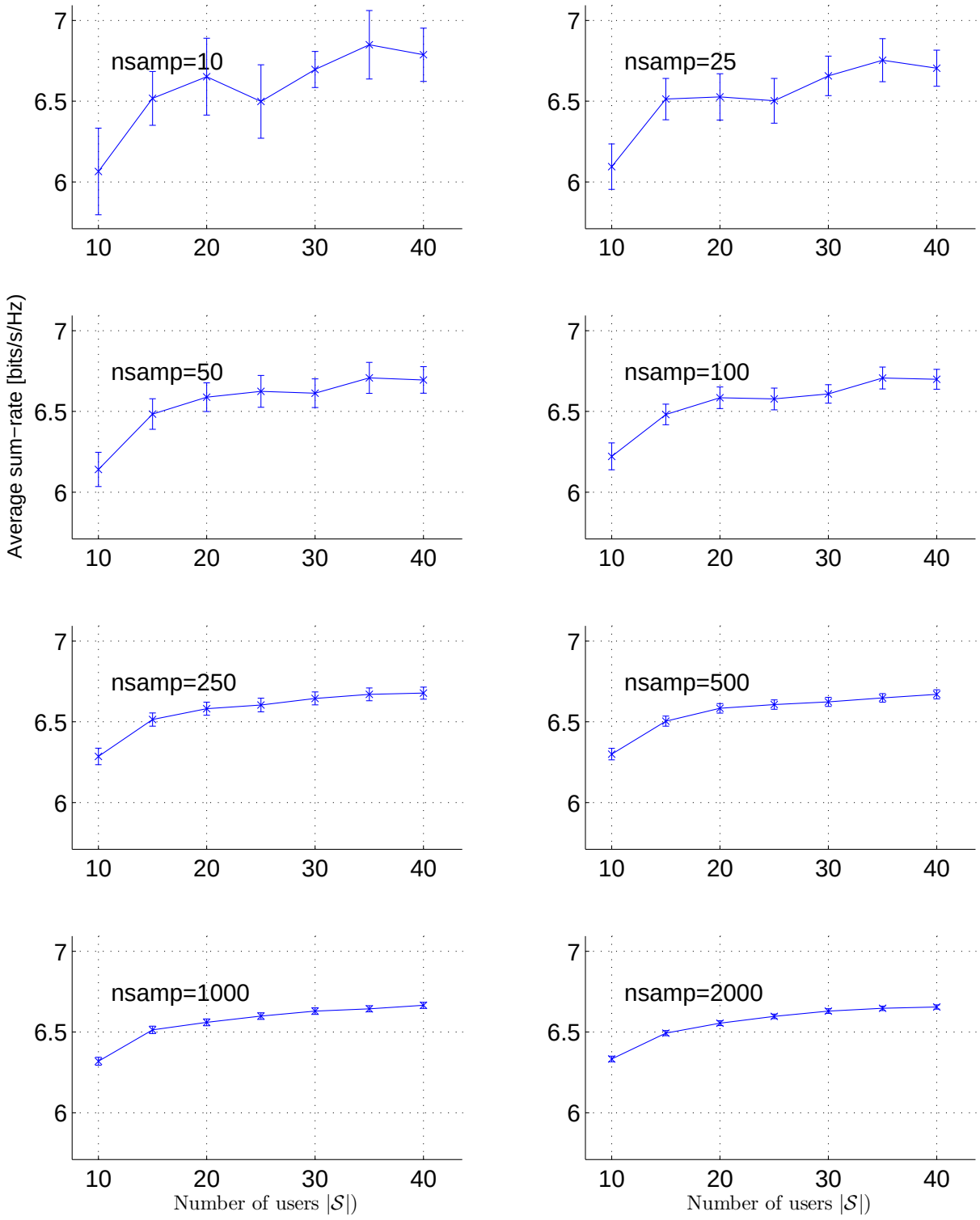

Figure A.3: Average SA sum rate and 95% confidence intervals vs $|\mathcal{S}|$ for $n_T = 4$, $n_R = 4$, SNR = 0dB

A.1.4 Comparison of Results to Published Works

After verifying the random variable generation, channel models, and statistical reliability of our results using confidence intervals and error bars on the data, a comparison to results previously published in literature can offer a final check of the validity of our data.

The results given in [52] for sum-rate maximization under SZF were compared against our results for exhaustive user allocation. In Figure 3 in [52], the authors presented the results of the exhaustive search for $n_T = 4$ $n_R = 2$, at 5 dB, 10 dB, and 15 dB, assuming that a full n_R data streams were sent to each user. Our results for exhaustive user allocation at K =20, 30, and 40 mirrored the results presented in that figure. Similarly, in Figure 4 in [52] results for exhaustive search under SZF for the case of and $n_T = 8$ $n_R = 2$ at 5 dB, 10 dB, and 15 dB were presented, again assuming that n_R data streams were sent to each user. Again, our results for exhaustive user allocation at K =20, 30, and 40 matched the results presented in that figure. We also compared our results for DPC precoding (used to provide an upper bound to the maximum achievable capacity) to the results provided in Figures 4 and 5 in the paper above, at K =20, 30, and 40 users, and found that they matched.

Figures 1 and 2 in [51] also provided results for $n_T = 4$ $n_R = 2$ at 5 dB and 10 dB respectively, assuming $n_T/n_R = 2$ users were scheduled with n_R data streams per user, matching our results for exhaustive user allocation at K =10, 20, and 40 users.

As there is very little literature for SZF scheduling with coordinated transmit-receive processing, this makes a more thorough comparison difficult. For verifying our results on coordinated transmit-receive processing, we looked at the results provided in [37], Particularly, Figure 8 provided probability densities of capacities for different channel geometries and channel decomposition algorithms at 10 dB. Of interest to us were the $\{4, 4\} \times 4$, $2SC$ denoting $n_T = 4$, $n_R = 4$, $K = 2$ users and a fixed allocation of 2 subchannels per user. Restricting our stream allocations to $\mathbf{L}^m = [2, 2]$ and simulating for $K = 2$ users (not shown) yielded a distribution closely matching that provided in Figure 8.

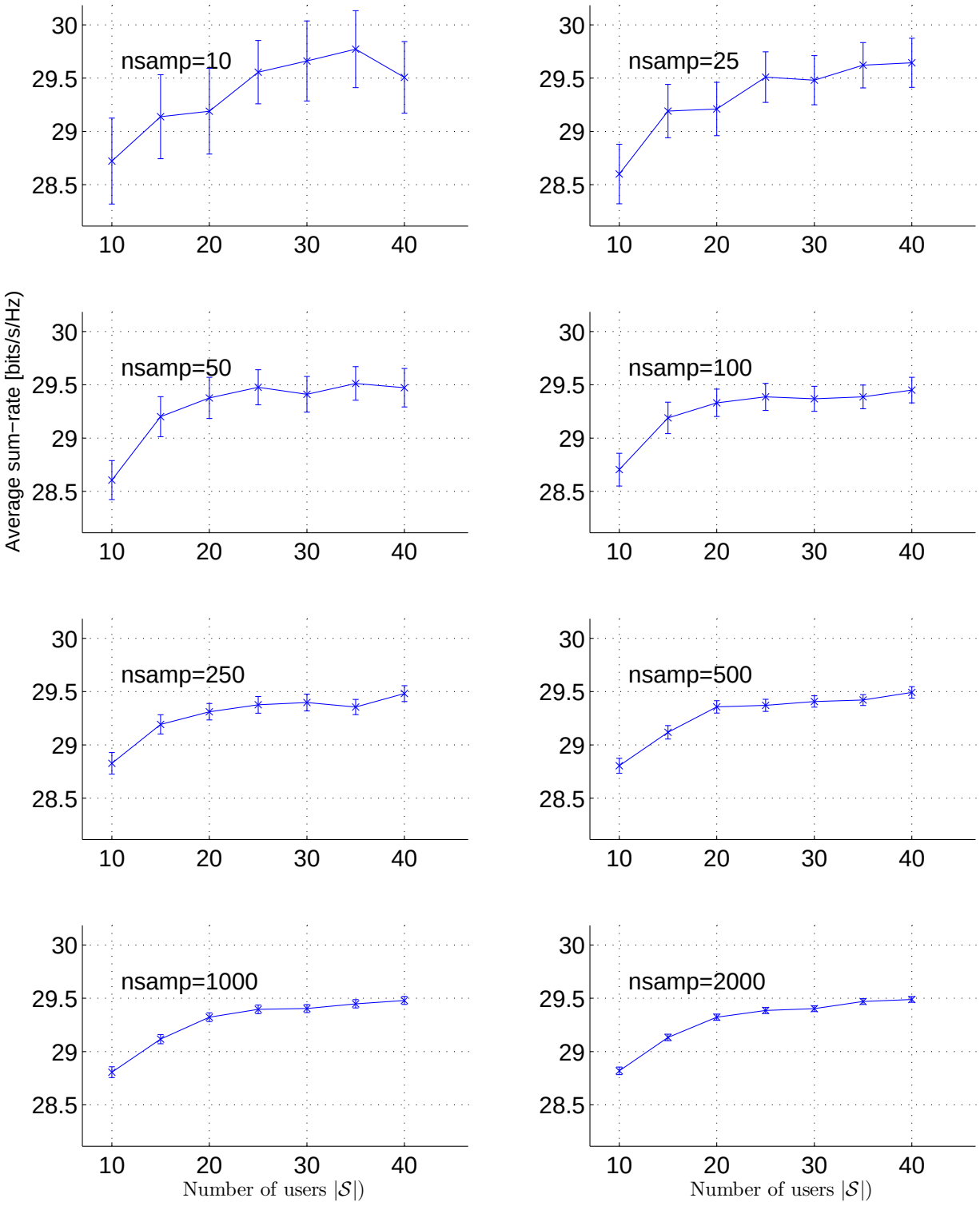

Figure A.4: Average SA sum rate and 95% confidence intervals vs $|\mathcal{S}|$ for $n_T = 4$, $n_R = 4$, SNR = 20dB