Numerical Optimization Techniques for Secure Communications Over MIMO Channels

Maria Urlea

Thesis submitted to the
Faculty of Graduate and Postdoctoral Studies
in partial fulfillment of the requirements for the
Master of Applied Science
degree in Electrical and Computer Engineering

Ottawa-Carleton Institute for Electrical and Computer Engineering
University of Ottawa

Abstract

As multimedia applications become more popular, wireless communication systems are expected to reliably provide increased data rates. Multiple Input Multiple Output (MIMO) technologies can meet this demand without using additional bandwidth or transmit power. MIMO is part of modern wireless communication standards.

Another critical aspect of communications is to secure the confidentiality of data transmission. Cryptography accomplishes this at the upper layers of the protocol stack. At the physical layer, data travels unencrypted and can be secured by using the channel characteristics to "hide" data transmission from potential eavesdroppers.

We consider a Gaussian MIMO wiretap channel and are looking for the maximal rate at which data can be transmitted both reliably and securely to the intended receiver: the secrecy capacity. This quantity is difficult to find analytically and is known precisely in only a few cases. This thesis proposes several numerical optimization methods, both stochastic and deterministic, to evaluate the secrecy capacity and to find the optimal transmit covariance matrix. The stochastic approaches are based on Monte-Carlo and on Differential Evolution (a genetic algorithm). The deterministic approaches are based on successive linear approximation. The accuracy of the results obtained with these methods is, in general, better than the one offered by popular numerical optimization tools such as CVX or YALMIP.

Acknowledgment

Foremost, I would like to express my gratitude to my supervisor, Prof. Sergey Loyka. His knowledge, attention to detail and constructive feedback have helped me solidify and expand my understanding of communication systems. I would like to thank him for challenging me to be a more mindful researcher and a better technical writer. I would also like to thank Prof. Loyka for his helpful comments on earlier versions of this thesis.

Completing my Master's degree would have been much more difficult without the funding from OGS, NSERC and the University of Ottawa. I am really grateful for the opportunities that I have been given as they have allowed me to focus on things that mattered.

Writing my thesis has been a journey and many people deserve a big "Thank you!" for supporting me at one point or another. I thank my fellow labmate Kaiya Li for lending a supportive ear. I also thank my friends for helping me put things into perspective and for offering a different point of view when things seemed hard. I have gotten to know some amazing individuals in the past few of years. Their approach to life is continuously inspiring and has bettered me in ways that I barely begin to realize.

Finally, I thank my parents, my brothers, and my best friend & partner Stejărel Vereş for their unconditional love and support, for believing in me, and for brightening every day.

I owe an extra thanks to Leila Mouhsine for all the coffee and for proof-reading, and to Steju for the editing tips and for proof-reading the final version of the thesis.

Contents

Notation

List of Acronyms and Abbreviations

3GPP-LTE	3rd Generation Partnership Project - Long Term Evolution
AMPS	Advanced Mobile Phone System
AN	Artificial Noise
AWGN	Additive White Gaussian Noise
CDMA	Code Division Multiple Access
CRC	Cyclic Redundancy Check
CSI	Channel State Information
DE	Differential Evolution
Ev	Eavesdropper
Fig.	Figure
G	3rd, 4th or 5th Generation
GSM	Global System for Mobile Communications
IEEE	Institute of Electrical and Electronics Engineers
IS	Interim Standard
KKT	Karush-Khun-Tucker
LTE	Long Term Evolution
MAC	Message Authentication Checks
MC	Monte-Carlo
MIMO	Multiple Input Multiple Output

MISO	Multiple Input Single Output
mM	min-max
OFDM	Orthogonal Frequency Division Multiplexing
OSI	Open Systems Interconnection
PKI	Public Key Infrastructure
RaMC	Rank-adaptive Monte-Carlo
Rx	Receiver
SLA	Successive Linear Approximation
SNR	Signal to Noise Ratio
s.t.	subject to
Tx	Transmitter
UMTS	Universal Mobile Telecommunications System

List of Symbols

α	Acceptable decrease in linear approximation (backtracking)
β	Step length coefficient (backtracking)
$\varepsilon_{\mathbf{K}}$	Defines interval of validity for the log-determinant approximation with trace; defines the linear approximation step size
$\varepsilon_{\mathbf{R}}$	Defines interval of validity for the log-determinant approximation with trace; defines the linear approximation step size
$a_{\mathbf{K}}$	Modifies the value of $\varepsilon_{\mathbf{K}}$ in the SLA algorithm
$a_{\mathbf{R}}$	Modifies the value of $\varepsilon_{\mathbf{R}}$ in the SLA algorithm
$\mathbb{C}$	The set of complex numbers
C_{tr}	Optimal secrecy capacity value in the weak eavesdropper approximation
$C_{\mathrm{tr}}(\mathbf{R})$	The secrecy capacity value as a function of the transmitter covariance matrix $\mathbf{R}$ in the weak eavesdropper approximation
$C_{\mathrm{tr},CVX}$	Optimal secrecy capacity value in the weak eavesdropper approximation as evaluated by CVX
C_c	Constellation capacity

C_{CVX}	Channel capacity evaluated with CVX
C_{Ev}	Channel capacity for the eavesdropper's channel
C_M	Channel capacity when a M symbol constellation is used
C_{MC}	Channel capacity evaluated with a random blind search
C_{Rx}	Channel capacity for the main channel
C_s	The secrecy capacity
$C_s(\mathbf{R})$	Secrecy rate as a function of transmitter covariance matrix, $\mathbf{R}$
$C_{s,MC}$	The secrecy capacity evaluated with a random blind search
$C_{s,mM}$	The secrecy capacity evaluated with the min-max algorithm
$C_{s,RaMC}$	The secrecy capacity evaluated with the Rank-adaptive Monte-Carlo algorithm
$C_{s,SLA}$	The secrecy capacity evaluated with the Successive Linear Approximation algorithm
$C_s^u(\mathbf{R},\mathbf{K})$	Upper limit to the secrecy capacity as a function of the transmitter covariance matrix $\mathbf{R}$ and the matrix $\mathbf{K}$
CR	Crossover criteria (DE)
E	Equivocation
F	Mutation criteria coefficient (DE)
g_i	Gain on channel i
G_{max}	The maximal number of generations (DE)
$\mathbf{H}$	The channel gain matrix
$\mathbf{h}$	The channel gain vector
$\mathbf{H}_1$	The receiver channel gain matrix
$\mathbf{h}_1$	The receiver channel gain vector
$\mathbf{H}_2$	The eavesdropper channel gain matrix
$\mathbf{h}_2$	The eavesdropper channel gain vector
$\mathbf{K}(\mathbf{R})$	$\mathbf{K}$ as a function of the transmitter signal covariance matrix in the min-max expression
L	Leakage
M	Message at the transmitter
m	Number of antennas at the transmitter

N	Number of tests in a Monte-Carlo optimization
n	Number of antennas at the receiver when there is no eavesdropper
n_1	Number of antennas at the receiver
n_2	Number of antennas at the eavesdropper
$n_{\mathbf{K}}$	Number of iterations to perform when linearly approximating $\mathbf{K}$ in the min-max algorithm
n_{out}	Number of outer loop iterations in the min-max algorithm
$n_{\mathbf{R}}$	Number of iterations to perform when linearly approximating $\mathbf{R}$ in the min-max algorithm
NP	Number of members in the initial population (DE)
P_{error}	Probability of error
p_i	Power allocated to channel i
P_s	Saturation point in the weak eavesdropper approximation
P_T	Total available power at the transmitter
$\mathbf{R}$	Transmitter signal covariance matrix
$\mathbf{R}(\mathbf{K})$	Transmitter signal covariance matrix as a function of $\mathbf{K}$ in the min-max expression
$\mathbf{R}(P_T)$	Transmitter signal covariance matrix as a function of P_T
$\mathbf{R}^*$	Optimal transmitter signal covariance matrix
$\mathbf{R}^*_{\text{tr}}$	Optimal transmitter signal covariance matrix in the weak eavesdropper approximation
$\mathbf{R}^*_{\text{tr},CVX}$	Transmitter signal optimal covariance matrix in the weak eavesdropper approximation evaluated by CVX
t_{CVX}	Computation time for optimization with CVX
$t_{\mathbf{K}}$	Backtracking coefficients when approximating $\mathbf{K}^*$
t_{MC}	Computation time for optimization with a random blind search algorithm
$t_{\mathbf{R}}$	Backtracking coefficients when approximating $\mathbf{R}^*$
t_{SLA}	Computation time for optimization with SLA
$\mathbf{W}$	Channel (power) matrix
$\mathbf{W}_1$	Channel matrix for the receiver channel

$\mathbf{W}_1 - \mathbf{W}_2$	Difference of channel matrices / the difference matrix
$\mathbf{W}_2$	Channel matrix for the eavesdropper channel
X	Transmitter signal, information theoretic argument
$\mathbf{x}$	Transmitter signal
Y	Receiver signal, information theoretic argument
$\mathbf{y}$	Receiver signal
Z	Eavesdropper signal, information theoretic argument

List of Mathematical Operators

λ	Dual variable
$\Lambda(\mathbf{A})$	Eigenvalue matrix of matrix $\mathbf{A}$
$(a)_+$	Returns the larger value between a and 0
λ_1	The largest eigenvalue of a (given) matrix
$\lambda_i(\mathbf{A})$	Eigenvalue i of matrix $\mathbf{A}$
$\mathbf{A}$	Bold upper-case letters represent matrices
$\mathbf{a}$	Bold lower-case letters represent vectors
$a\mathrm{E}b$	Exponential notation: $a\mathrm{E}b = a \cdot 10^b, a \in [0, 1]$
$\mathbf{A}^\perp$	Null of matrix $\mathbf{A}$
$\mathbf{A}^*$	Optimal value of matrix $\mathbf{A}$
$\mathbf{A}^+$	Hermitian conjugate of matrix $\mathbf{A}$
$\mathbf{A}^{-1}$	Inverse of matrix $\mathbf{A}$
$\mathbf{A}^T$	Transpose of matrix $\mathbf{A}$
$\mathbf{A}_+$	$\sum_{i_+} \mathbf{u}_i \mathbf{u}_i^+ \lambda_i, \lambda_{i_+} > 0$; λ_i and $\mathbf{u}_i$ are the eigenvalues and the corresponding eigenvectors of matrix $\mathbf{A}$
$\|\mathbf{A}\|_2$	The 2-norm of matrix $\mathbf{A}$
$\mathbf{A} \preceq \mathbf{B}$	$\mathbf{B} - \mathbf{A}$ is a positive semidefnite matrix
$\mathbf{E}$	The all ones matrix
$\mathcal{E}\{\cdot\}$	Statistical expectation
$\mathbb{H}(X)$	Entropy
$\mathbb{H}(X\|Y)$	Conditional entropy
$\mathbf{I}$	The identity matrix

$\mathbb{I}(X;Y)$	Mutual information
Pr	Probability
$\mathrm{r}_+\mathbf{A}$	Number of positive eigenvalues of matrix $\mathbf{A}$
`randb`	Random number generator function; result follows a uniform distribution in the $[0,1]$ interval
`randn`	Random number generator function; result follows a Gaussian distribution with 0 mean and unit variance
`randnum`(a)	Random number generator function, chooses an integer number at random between 0 and a
rank$\mathbf{A}$	Rank of matrix $\mathbf{A}$
tr$\mathbf{A}$	Trace of matrix $\mathbf{A}$
$\mathbf{u}_1$	The eigenvector corresponding to the largest eigenvalue of a (given) matrix
$\mathbf{U}_\mathbf{A}$	The columns of this matrix are the eigenvectors of matrix $\mathbf{A}$

Chapter 1

Introduction

Wireless communication is very popular because it is convenient and supports user mobility. With the increase in the number of services offered, ensuring fast and secure communication is imperative.

A breakthrough in the amount of data that can be transmitted was reached with the advent of MIMO (Multiple Input Multiple Output) technologies. Both the transmitter and the receiver have multiple antennas and, through signal processing, space can be leveraged as an additional communication dimension. MIMO is an integral part of several wireless communication standards including 3GPP-LTE [1], LTE-Advanced [2] and IEEE 802.11n, IEEE 802.11ac [3]. Due to its ability to make efficient use of scarce resources such as spectrum, MIMO is likely to be part of future communication standards as well.

When transmitting, data bits are transformed into waveforms and sent toward their destination over the physical medium. Without additional precaution, anyone in the medium can hear the data being transmitted. Because the medium is shared, wiretapping in a wireless communication system does not involve anything in addition to being physically present within the range of the transmitter.

As a result of the inherently insecure communication medium, the use of data security proto-

cols is imperative. Algorithms dealing with data security are implemented at different layers of the communication system. Their main two functions are encryption and authentication.

When encryption is used, data (the plaintext) is not sent in clear across the channel, but instead is transformed into a ciphertext using an algorithm that has the plaintext and an encryption key as inputs. The eavesdropper has access to the ciphertext but cannot decrypt it without the correct decryption key. Encryption algorithms fall within one of two classes based on how encryption and decryption keys are generated and exchanged: asymmetric and symmetric.

Asymmetric encryption, also referred to as Public Key Infrastructure (PKI), uses a public-private key pair to protect the transmitted data. The two keys are different but related. The public key is available to everyone, including the eavesdropper. The transmitter encrypts the message with this key. The private key is known only to the intended receiver and he is the only one capable of decrypting the message. Public-private keys are computed ahead of time and distributed by a trusted third party. The security of the asymmetric encryption is based on the fact that it is computationally infeasible to infer what the private key is from the public key. Key validation is usually an asynchronous process (as opposed to a real-time one), which makes using PKI for real-time communication impractical [4].

Symmetric key encryption uses the same key for both encryption and decryption. As a result, the value of the key has to be known only by the communicating parties. Secret keys need to be distributed through a trusted distribution channel. However, they are more computationally efficient as compared to public keys [5]. Encryption algorithms that are used to secure network communication are based on symmetric encryption schemes. Secret key generation is usually accomplished through an algorithm of the Diffie-Hellman variety [6], and communicating parties authenticate mutually through PKI.

Therefore, in practice, securing the data communication employs both types of algorithms: a public key algorithm is used for authentication and to establish a secure channel for secret-

key distribution, and a symmetric algorithm for performing the actual real-time encryption and decryption.

Cryptographic protocols assume that the received data is error-free. The data-link layer receives frames and handles bit-wise error detection. Corrupted frames are discarded and eventually retransmitted. This provides largely error-free data to the upper layers. However, because the framework for error detection relies on cryptographically weak functions from the Cyclic Redundancy Check (CRC) family, error detection alone is not an adequate mechanism for ensuring the authenticity of data. Because of this, security mechanisms involve, in addition to encryption, algorithms that verify data integrity. These algorithms fall under the umbrella of Message Authentication Checks (MAC) [4].

In physical layer security, rather than relying on the fact that the eavesdropper has no access to the keys and it has limited processing power to perform attacks against the encryption algorithms themselves, the goal is to minimize the ability of the eavesdropper to retrieve information (encrypted or not) from the channel used by the transmitter and the receiver. This way, messages can be transmitted confidentially without using an encryption key. This can enhance the security of the secret keys by providing an additional layer of obfuscation to the Diffie-Hellman key exchange process. This and other system-related issues, such as key agreement, cross-layer design, authentication, etc. are discussed in [5].

The characteristics of the physical medium make it such that any receiver, legitimate or otherwise, hears a different version of the waveforms being transmitted. Confidentiality of data can be obtained by exploiting this difference to hide the transmission from unintended receivers. This is fundamentally different from the cryptographic assumption. In cryptography, the eavesdropper hears the same information as the receiver but it cannot decrypt it without a key.

The maximal rate at which information can be transmitted reliably subject to an information-theoretic secrecy requirement is called *secrecy capacity* [7]. Physical layer security, making

sure that no information is leaked to the eavesdropper, is an addition (not an alternative) to existing security schemes.

Whereas securing information at the physical layer is not so problematic in wired communication systems, since access to the physical cables is comparably hard to obtain inconspicuously, implementing physical layer security in wireless communication systems can provide increased confidence with respect to the securing of data. The research potential of this area has remained relatively untapped even though the fundamental idea dates back to a few decades ago [8]. In recent years, owing to an increased understanding of the complex theory involved and to the technological advances, there has been a surge in interest in physical layer security in the research community. Fig. 1.1[1] gives a rough estimate of the number of publications over the last decade.

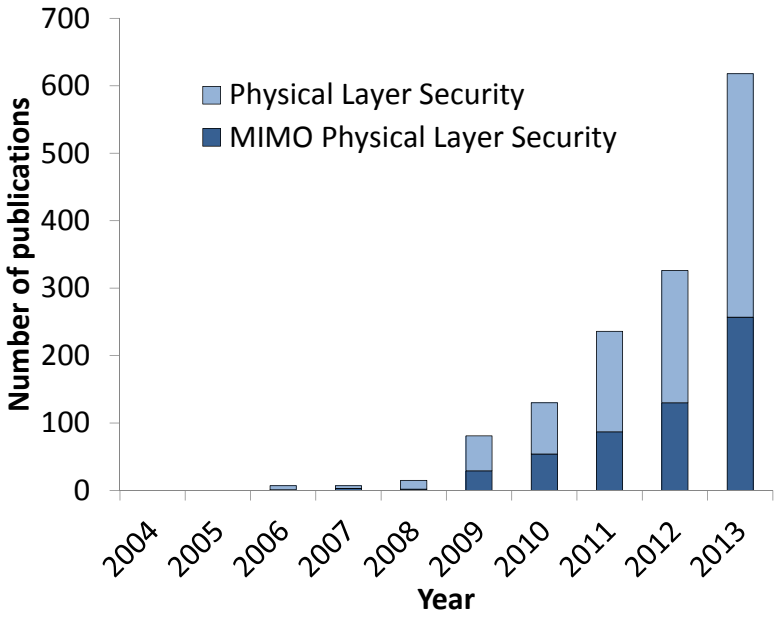

Figure 1.1. Number of research articles published on physical layer security.

[1]As counted by Google Scholar using all of the following key-words: wireless, wiretap, secrecy and "physical layer security" (light shaded bars); MIMO, wireless, wiretap, secrecy "physical layer security" (dark shaded bars).

When the channel noise, including interference, is modeled as Gaussian, the value of the secrecy capacity is dependent only on the receiver's and the eavesdropper's channel gains and on the transmit power. In a MIMO wireless channel, under the total power constraint, the secrecy capacity is determined by the transmitter covariance matrix[2], i.e. by the power allocated to each antenna signal and the correlation between them.

Finding the optimal transmit covariance matrix to maximize the secrecy capacity is not an easy task and closed-form solutions to this problem are known only in particular cases. Chapter 4: Known Solutions shows these solutions in more detail.

Three different approaches to secrecy capacity maximization are considered in this thesis:

1. *Convex optimization solvers*: Some numerical solutions can be obtained using readily-available tools or iterative tools. The two numerical optimization tools considered are CVX and YALMIP. CVX is a very popular[3] convex optimization tool. It encompasses several numerical solvers both publicly available and commercial. The solvers that are considered in this thesis are: SDP, SeDuMi, SDPT3, and Mosek. CVX accepts convex optimization problems that respect the disciplined convex optimization rules [10] and returns the optimum value of the objective function. YALMIP is also widely known solver[4]. It is structured as a MATLAB toolbox. It accepts convex and non-convex problems alike but does not guarantee convergence to the global optimum. Results obtained with these solvers have to be used with care because, in some cases, they are misleading.

2. *Monte-Carlo optimization*: The transmit covariance matrix is optimized by testing randomly-generated (valid) covariance matrices. To direct the search, additional constraints need to be imposed on the way the test matrices are constructed.

[2]In a MIMO Gaussian channel optimal secret signaling is Gaussian [9]. The mean and variance are sufficient to specify this distribution and hence, when the channel gains are known, the transmit covariance matrix is sufficient to determine the secrecy capacity.

[3]1747 citations on Oct. 23rd, 2014; as counted by Google Scholar.

[4]2842 citations on Oct. 23rd, 2014; as counted by Google Scholar.

3. *Approximations and analysis*: This approach is based on mathematical derivations. When the channel matrices are deterministic and known to all actors, approximations to the secrecy capacity value are obtained under some additional assumptions. The results help us build and validate numerical algorithms.

1.1 Main Results and Contributions

The main results of this thesis are as follows:

1. Numerical verification of some of the analytical solutions for the optimal transmit covariance of a MIMO wireless system under the total power constraint was performed.
2. Numerical optimization tools were used on the secrecy capacity expression. It was demonstrated by means of example that neither CVX or YALMIP tools are able to provide results with a good accuracy in the general case.
3. Four different custom-made numerical algorithms were developed to find the secrecy capacity.

1.2 Thesis Outline

The remainder of this thesis is organized as follows:

Chapter 2: Background

This chapter takes a closer look at wireless systems at the physical layer. It illustrates the importance of MIMO for the industry and explains how the MIMO architecture is used to increase data throughput. This chapter also introduces physical layer security and defines the secrecy capacity from an information-theoretic perspective. Different channel models and limiting assumptions are discussed with particular emphasis on the Gaussian MIMO case.

An overview of coding insights for wireless systems is also presented.

Chapter 3: System Model

Introduces the system model and notations for wireless MIMO systems. It also states two equivalent expressions for the secrecy capacity and a short physical interpretation of these formulas.

Chapter 4: Known Solutions

This chapter presents some of the known transmitter matrix characteristics, such as limiting behavior at low and high signal to noise ratio (SNR), special properties as well as its expression in a few particular cases. Some of the results are implemented in MATLAB. Short examples are also included.

Chapter 5: Notes on Numerical Solvers

Several examples are constructed to explore the application and limitations of CVX and YALMIP to MIMO secrecy capacity optimization. CVX cannot optimize the secrecy capacity expression directly and returns sub-optimal results for one of its approximations. YALMIP can handle any expression, but the accuracy of its results varies inconsistently with system parameters.

Chapter 6: Monte-Carlo Methods and Optimization

Two algorithms are developed. The first one, the Rank-adaptive Mote-Carlo algorithm, uses a rank property of the optimal covariance matrix. The second one is differential evolution, also known as genetic algorithm. It emulates natural evolution processes such as mutation, crossover, and selection to converge toward the best result.

Chapter 7: Numerical Optimization

The secrecy capacity function is optimized numerically using two custom-made algorithms. The first one is a successive linear approximation method. The second one is a min-max algorithm.

Chapter 2

Background

In this chapter, transmitting using MIMO systems is briefly reviewed. Also, some physical layer security aspects are discussed. Gaussian MIMO channels are presented. Finally, some aspects of transmitter signal optimization are made note of.

The internal functioning of communication systems can be understood using a conceptual model such as the Open Systems Interconnection (OSI) model [11] (see Fig. 2.1). In this model the communication functions are defined and divided in several logical layers. The top layer takes its input from the user (e.g. voice) via a particular software application. As information travels to lower layers it is no longer application specific, it is encapsulated, transformed into bits and eventually into signals that can be transported over the physical medium. Once on the other side, the signals are processed and information can be understood by the intended receiver (e.g. sound).

The functionality of each layer is well-defined. Each layer is supported by the layer below it [11]. This modular architecture makes technical solutions portable from one system to the next without complete system redesign. Wired and wireless communication systems function in the same way except at the two lower layers which deal with information transfer over the medium: data-link and physical. The data link layer is required to define the wireless-specific

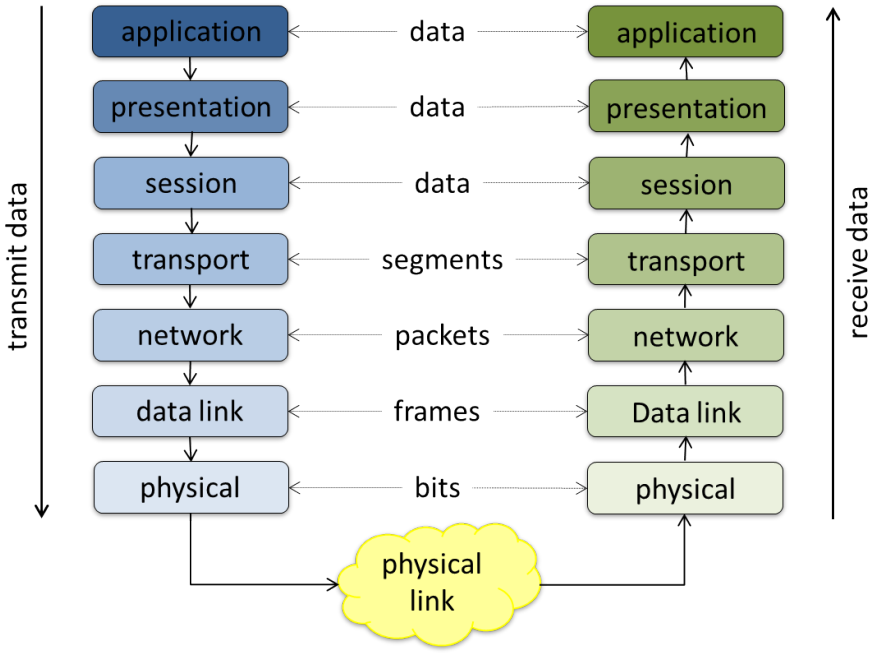

Figure 2.1. The OSI model layers.

functions such as authentication and base station association. Data link also specifies media access, framing and addressing, as well as cryptographic security parameters. The physical layer defines the way that information is being sent over the channel.

Aspects of communication at the physical layer are covered in several publications [12], [13], [14], [15]. Wireless specific issues are covered in depth in [9].

Wireless communications can refer to several systems: radio, TV, paging etc. The most evident modern example is cellular telephony. This system has a large number of users and serves large geographical areas. To provide coverage and ensure connectivity, the areas are divided in cells. Each cell has an antenna in its middle and is served by a base station (see Fig. 2.2).

Once a user is present in a network, it is associated to the base station that provides the best signal[1] [9]. When a call is placed, the base station collects the sent information and transmits it toward the receiver using the public telephone wired network [9]. The receiver

[1]This is not necessarily the closest base station. The base station that the user is associated to may change during the communication.

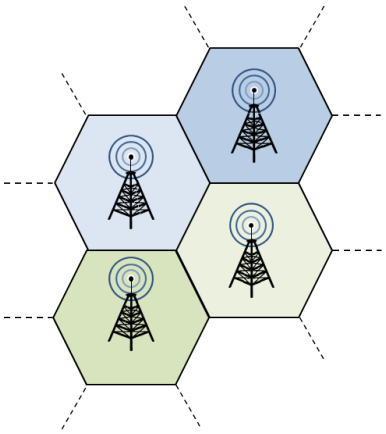

Figure 2.2. Base stations and cell arrangement.

may use a wired telephone line or may be another wireless client.

One of the advantages of having a wireless network is the support for user mobility. Changes in demand and the increase in the number of services offered in this medium resulted in a significant evolution of wireless communication technologies. The wireless communication schemes adopted over the years (AMPS–Advanced Mobile Phone System, GSM–Global System for Mobile Communications, UMTS–Universal Mobile Telecommunications System, IS-95–Interim Standard 95, CDMA2000, LTE–Long Term Evolution, IEEE's 802.11 set of standards for implementing Wireless Local Area Networks) attest to this development. The fifth generation could improve the user experience by providing lower latency, longer battery life, better coverage etc. In the research community, there are different visions for what 5G should be [16], [17], [18]. 5G has not yet been standardized [19].

The technology that is currently being deployed in North America is 4G-LTE and is based on OFDM-MIMO. MIMO, using multiple antennas at the transmitter and the receiver, makes it possible to transmit information using different spatial streams. It increases the data throughput and the cell coverage without increasing the average transmit power or the used bandwidth. Experimental results in [20] show that MIMO speeds can be much higher than the speeds achieved with a single transmit, single receive antenna system using the same bandwidth and the same total transmit power. These characteristics make it an important

part of wireless communication systems. Section 2.1 looks at MIMO in more detail.

2.1 MIMO Overview

The MIMO system is depicted in Fig. 2.3. The receiver signal is:

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

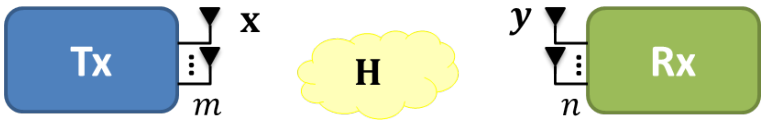

Figure 2.3. MIMO block diagram.

The Tx has m antennas and the Rx n antennas. $\mathbf{H} \in \mathbb{C}^{m \times n}$ is the channel gain matrix and $\mathbf{W} = \mathbf{H}^{+}\mathbf{H}$ the channel matrix. $\boldsymbol{\xi}$ is the channel noise and it is modeled as additive white Gaussian and normalized, $\boldsymbol{\xi} \sim \mathcal{N}(0, 1)$.

The channel capacity is equal to:

$$C = \max_{\mathbf{R} \succeq \mathbf{0}, \operatorname{tr} \mathbf{R} \leq P_T} \ln \det(\mathbf{I} + \mathbf{H}\mathbf{R}\mathbf{H}^{+}) \tag{2.2}$$

C is an increasing function in P_T [9]. The optimal transmitter covariance matrix solution is well-known [9], [14], [21] and can be understood following two steps. First, decorrelate the spatial streams. Then, optimize the power allocation over the set of parallel channels. These are explained in several publications [9], [22] and reviewed below.

Spatial Stream De-correlation

By singular value decomposition, the matrix $\mathbf{H}$ can be written as:

$$\mathbf{H} = \mathbf{V}_1 \mathbf{D} \mathbf{V}_2^+ \tag{2.3}$$

where both $\mathbf{V}_1$ and $\mathbf{V}_2$ are unitary matrices and invertible. $\mathbf{D}$ is a diagonal matrix and its diagonal entries are equal to the square root of the eigenvalues of $\mathbf{W} = \mathbf{H}^+\mathbf{H}$. Also, $\operatorname{rank}\mathbf{D} \leq \min(m, n)$.

Let $\widetilde{\mathbf{y}} = \mathbf{V}_1^+\mathbf{y}$, $\widetilde{\mathbf{x}} = \mathbf{V}_2^+\mathbf{x}$, and $\widetilde{\boldsymbol{\xi}} = \mathbf{V}_1^+\boldsymbol{\xi}$; $\widetilde{\boldsymbol{\xi}}$ and $\boldsymbol{\xi}$ have the same characteristics. Equation (2.1) is equivalent to:

$$\widetilde{\mathbf{y}} = \mathbf{D}\widetilde{\mathbf{x}} + \widetilde{\boldsymbol{\xi}} \tag{2.4}$$

alternately:

$$\widetilde{y}_i = d_i \widetilde{x}_i + \widetilde{\xi}_i, \quad 1 \leq i \leq \operatorname{rank}\mathbf{D} \tag{2.5}$$

The original channel is equivalent to $\operatorname{rank}\mathbf{D}$ parallel channels (see Fig. 2.4).

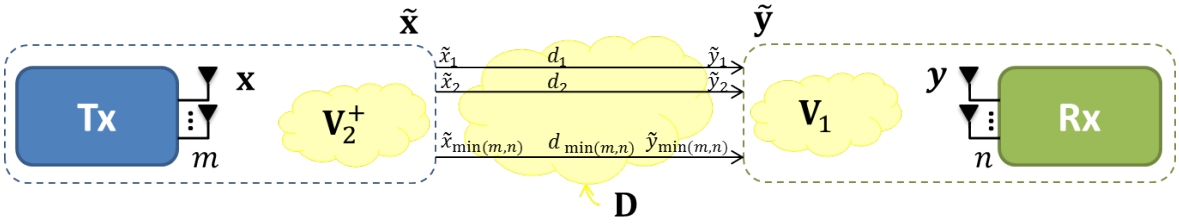

Figure 2.4. MIMO equivalent channel after decorrelation.

Power Allocation over Parallel Channels

The channel is equivalent to $k = \operatorname{rank}\mathbf{D}$ parallel scalar additive white Gaussian noise sub-channels, each having a channel gain of d_i. The noise power is normalized to 1 on all

sub-channels. The rate of reliable communication over these channels is:

$$R = \sum_{i=1}^{k} \ln\left(1 + d_i^2 p_i\right) \tag{2.6}$$

where p_i is the power allocated to sub-channel i, under the total power constraint $\sum_i p_i \leq P_T$. (2.6) can also be written in matrix form. Let $\mathbf{P}$ be a diagonal matrix and its diagonal elements:

$$\mathbf{P}(i,i) = \begin{cases} p_i, & 1 \leq i \leq k \\ 0, & k < i \leq m \end{cases} \tag{2.7}$$

and the channel capacity is equal to:

$$C = \max_{\mathbf{P}} \ln\det(\mathbf{I} + \mathbf{DPD}+) \tag{2.8}$$

The power allocation that maximizes the channel capacity is [9], [23]:

$$p_i^* = \begin{cases} \frac{1}{\lambda} - \frac{1}{d_i^2}, & \gamma_i > \lambda \\ 0, & \gamma_i \leq \lambda \end{cases} \tag{2.9}$$

where λ is such that $\sum_i p_i^* = P_T$. (2.9) is known as the water-filling solution. A graphical interpretation of this solution is presented in Fig. 2.5.

More power is allocated to the higher-gain sub-channels (streams) to take advantage of better channel conditions and less or even no power is allocated to the lower gain ones.

The value of C in (2.2) and in (2.8) is the same:

$$\max_{\mathbf{R}\preceq\mathbf{0},\mathrm{tr}\,\mathbf{R}\leq P_T} \ln\det(\mathbf{I} + \mathbf{HRH}^+) = \max_{\mathbf{P}\preceq\mathbf{0},\mathrm{tr}\,\mathbf{P}\leq P_T} \ln\det(\mathbf{I} + \mathbf{DPD}^+) \tag{2.10}$$

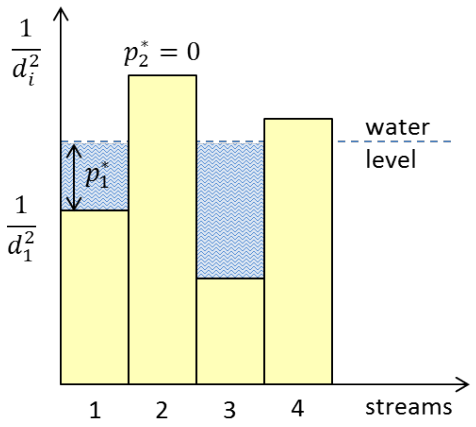

Figure 2.5. The water-filling solution.

The optimal transmitter covariance matrix for the MIMO channel is:

$$\mathbf{R}^* = \mathbf{V_2^+ P^* V_2} \tag{2.11}$$

The channel characteristics are summarized in the $\mathbf{W} = \mathbf{H}^+\mathbf{H}$ matrix. When there are several sub-channels, beamforming is done in multiple directions. $\mathbf{W}$'s eigenvectors are used to determine the beam pattern. The signals sent in the null of $\mathbf{W}$ do not reach the receiver, i.e. the channel gain in the null of $\mathbf{W}$ is zero. The eigenvalues of $\mathbf{W}$ indicate the stream strength[2]. The value of $\operatorname{rank}\mathbf{W} = \operatorname{rank}\mathbf{D} \leq \min(m, n)$ indicates the number of independent data streams between the transmitter and the receiver. $\mathbf{R}^*$'s eigenvectors are the same as $\mathbf{W}$'s. Note that when $\operatorname{rank}\mathbf{R}^* = 1$ the optimal transmit strategy is beamforming.

2.2 Security at the Physical Layer

The information being transmitted can be sensitive in nature and security is a major concern. This is usually addressed at the upper layers using cryptography. At the physical layer, information travels unencrypted as an analog signal.

[2] $\mathbf{W} = \mathbf{H}^+\mathbf{H}$; the eigenvalues of $\mathbf{W}$ are equal to the d_i^2 above.

In the context of securing transmitted data, three main actors need to be considered: the transmitter (Tx), the legitimate (or intended) receiver (Rx), and the eavesdropper (Ev).

Cryptographic protocols are based on encryption. The message at the transmitter (plaintext) is converted into ciphertext (unintelligible text) with the aid of a *key*. The chiphertext is then transmitted across the communication channel and can be heard by any user of the network, including eavesdroppers. Transmission of the ciphertext is assumed to be error-free. The legitimate receiver can recover the plaintext by using its own key to decrypt the ciphertext. Without access to the key, it is computationally infeasible for the eavesdropper to recover the original message [7].

The two main encryption methods are *secret-key* encryption and *pubic-key* encryption. In secret-key encryption, the transmitter and the receiver use the same key to encrypt and decrypt the information. This requires a key exchange mechanism which is capable of securely distributing secret keys to the transmitter and the intended receiver only. In public-key encryption, the encryption key is publicly known but the key that can be used to recover the message is known at the intended receiver only. Public key encryption is not usable in a practical fashion in real-time network communication because the key distribution and validation mechanisms are performed asynchronously and out-of-band.

In shared media such as wireless the data is broadcast at the physical layer and all the receivers within the transmit range can access it. Depending on where they are located, each receiver's signal is modified by environment characteristics such as interference from other users, path loss, scattering etc. Hence, each listener hears a different version of the signal.

Through physical layer security, this difference can be exploited. The channel randomness can be used to hide sensitive information from the eavesdroppers. References [5], [7] give an overview of physical layer security, including both the theoretical fundamentals and a discussion of system implementation issues.

Communication at the physical layer is described in Fig. 2.6. The transmitter encodes the

message M, a series of bits, and sends a codeword X, unencrypted, over a noisy communication channel. The information that reaches the receiver is Y_1, the transmitted signal as modified by the channel. Y_1 is decoded to extract the estimated information, $\hat{M}$. The communication path between Tx and Rx is called the *main channel*.

The eavesdropper is an unwelcome party to the communication. It can be a single individual or several (collaborating) terminals. Here, the eavesdropper is considered to be passive; it is just listening and does not try to disturb the communication between the Tx and the Rx. The eavesdropper also has access to the information being sent and hears Y_2, another version of the transmitted codeword. The communication path between the Tx and the Ev is called the *eavesdropper channel*.

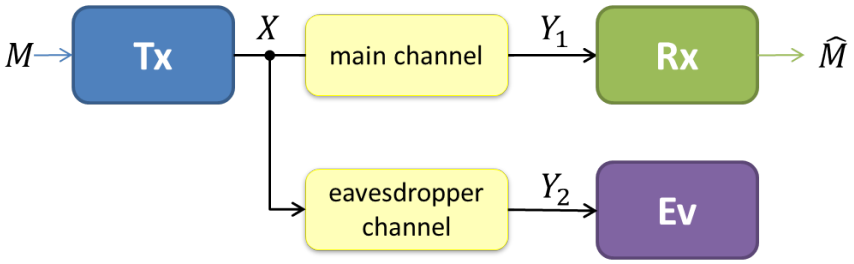

Figure 2.6. Communication in the presence of an eavesdropper.

The message transmission is confidential when the receiver is able to successfully interpret the information sent while the eavesdropper is unable to do so. Successful information transmission has two main characteristics: reliability and secrecy.

Reliability is defined in terms of the probability of error when decoding the message:

$$\mathrm{P}_{error} = \Pr(\hat{M} \neq M) \tag{2.12}$$

For reliable communication, the probability of error needs to be asymptotically zero:

$$\Pr(\hat{M} \neq M) \to 0 \tag{2.13}$$

Secrecy is defined relative to the amount of information that the eavesdropper can gather. The concept of secrecy was formalized by Shannon in [24]. Secrecy is defined in terms of *equivocation*: the amount of uncertainty about the message at the eavesdropper:

$$\mathtt{E} = \mathbb{H}(M|Y_2) \tag{2.14}$$

where $\mathbb{H}$ is *conditional entropy*[3]. The worst-case scenario is when the eavesdropper observes the codeword X without any degradation: $X = Y_2$. *Perfect secrecy* is then achieved when:

$$\mathtt{E} = \mathbb{H}(M) \tag{2.15}$$

i.e. when the codeword received at the eavesdropper is independent from the message and the message M cannot be recovered from the codeword Y_2 without any other further information.

Secrecy can be measured in terms of equivocation or in terms of *leakage*. Leakage measures the amount of information "leaked" to the eavesdropper and is defined in terms of mutual information, $\mathbb{I}$:

$$\mathtt{L} = \mathbb{I}(M; Y_2) = \mathbb{H}(M) - \mathbb{H}(M|Y_2) \tag{2.16}$$

Communication is considered to be secure when the eavesdropper observation is (asymptotically) statistically independent from the message being sent as the codeword length, n, approaches infinity:

$$\mathtt{L} \to 0 \tag{2.17}$$

The *weak* secrecy condition is:

$$\lim_{n\to\infty} \frac{1}{n}\mathbb{I}(M; Y_2) = 0 \tag{2.18}$$

where n is the length of the codeword being transmitted. When the weak secrecy condition

[3]Entropy is defined in any reference dealing with information theory such as [25]. $\mathbb{H}(Y|X) = -\sum_{x\in\mathcal{X}}\sum_{y\in\mathcal{Y}} p_{XY}(x,y)\log p_{Y|X}(y|x)$ where x and y are the symbols of the alphabet set $\mathcal{X}$ and $\mathcal{Y}$ and p_{XY} is their joint probability distribution.

holds, the rate of information leaked to the eavesdropper is asymptotically 0.

The *strong* secrecy condition is different:

$$\lim_{n\to\infty} \mathbb{I}(M; Y_2) = 0 \tag{2.19}$$

i.e. the amount of information that the eavesdropper is able to intercept is zero.

The two conditions are not equivalent and one can construct coding schemes that satisfy (2.18) but not (2.19). Example 3.2 in [7] is one such case.

The maximal rate at which information can be transmitted both reliably and securely over a communications channel is called *secrecy capacity*, C_s.

The secrecy capacity definition for the degraded broadcast wiretap channel (see Fig. 2.7) was introduced by Wyner [8]. In this model, the eavesdropper's channel is a degraded version of the main channel, i.e. $X \to Y_1 \to Y_2$ form a Markov chain. The difference in channel quality enables the legitimate receiver to hear more information than the eavesdropper. Confidential messages can be sent to the receiver without the eavesdropper's knowledge.

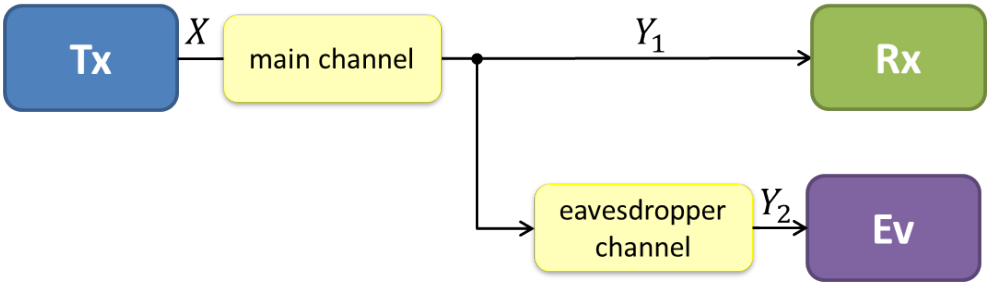

Figure 2.7. Wyner's wiretap channel (the degraded channel model).

On this channel, the secrecy capacity C_s is lower bounded [7] by :

$$C_s = \max_X(\mathbb{I}(X; Y_1) - \mathbb{I}(X; Y_2)) \geq \max_X \mathbb{I}(X; Y_1) - \max_X \mathbb{I}(X; Y_2) = C_{Rx} - C_{Ev} \tag{2.20}$$

where C_{Rx} and C_{Ev} are the channel capacity between the transmitter and the receiver, and the

transmitter and the eavesdropper respectively. If the eavesdropper sees the same information as the receiver: $Y_2 = Y_1$, then $\mathbb{I}(X;Y_2) = \mathbb{I}(X;Y_1)$ and information-theoretic secrecy cannot be achieved, $C_s = 0$.

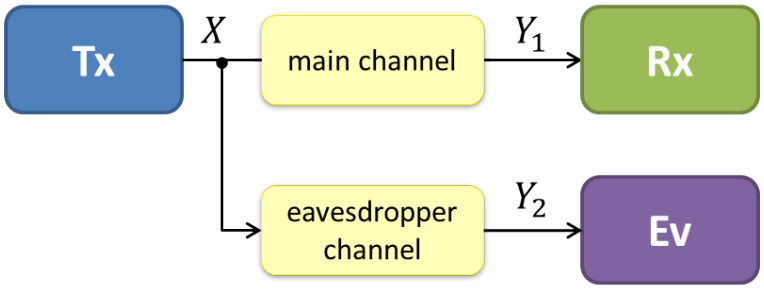

Figure 2.8. Non-degraded channel model.

The application of the secrecy capacity concept was expanded by Csiszár in [26]. There, an expression for secrecy is derived for non-degraded discrete memoryless channels (see Fig. 2.8). In this case, there is no clear advantage for the intended receiver. Secret communication is achieved when the eavesdropper's channel is *noisier*[4] than the receiver's. Information can still be delivered confidentially by using an additional random variable to randomize the encoder.

2.3 Physical Layer Security Over the Wireless Link

Assume that the channel noise can be modeled as additive white Gaussian noise[5]. For the scalar Gaussian wiretap channel (the transmitter, receiver and eavesdropper have on antenna each), either the eavesdropper's channel is degraded with respect to the receiver's or vice-

[4]Formally, a channel is noisier if a random variable V can be found such that $V \to X \to (Y_1, Y_2)$ and $\mathbb{I}(V;Y_1) \geq \mathbb{I}(V;Y_2)$. Note that a physically degraded channel, such as in the Wyner's model, is noisier.

[5]The noise model is based on the central limit theorem that indicates that the summation of many random processes (in this case noise sources) has a Gaussian distribution. The noise is additive because it is "added" to the signal being transmitted. White refers to the fact that the noise power is the same across all frequencies. Reference [27] discusses noise in receiving systems in more detail. The AWGN channel serves as a first step when studying the capacity of other wireless channels [9]. This model does not consider factors such as inter-cell interference or fading.

versa. An expression for the secrecy capacity in this case is shown in [28]:

$$C_s = \left(\frac{1}{2}\ln\frac{1+h_1^2\gamma}{1+h_2^2\gamma}\right)_+ = (C_{Rx} - C_{Ev})_+ \tag{2.21}$$

where h_1 and h_2 are the channel gain for the receiver and for the eavesdropper respectively, and γ is the signal power to noise ratio (SNR).

When the Tx, Rx and Ev have multiple antennas, recognizing the secrecy capacity characteristics is a bit more involved. The expression for the MIMO secrecy capacity, under the total power constraint, is given in [29], [30], [31]. An alternate characterization, in a form of an upper bound, is shown in [29], [31]. Both of these expressions are stated in Chapter 3: System Model. C_s is dependent on both the channel and the system characteristics as well as on the eavesdropper characteristics.

The value of the secrecy capacity is not known in the general case. However, some of its characteristics have been found.

In a system where the receiver has only one antenna, the requirement to have a positive value for secrecy capacity is for the eavesdropper to have less than twice the number of antennas that the transmitter has [29]. In MIMO, to prevent secure communication, Ev needs at least three times as many antennas as the Tx and Rx combined and the optimum distribution of antennas between Tx and Rx is 2:1 [32]. Adding more antennas to the Tx or the Rx can improve secrecy but only sub-linearly [33].

The noise over the wireless channel is modeled as additive white Gaussian. Under the total power constraint, optimal signaling follows a Gaussian distribution [28]. This distribution is fully characterized by its mean and its variance. Therefore, to characterize communication, the quantities that are required are the power allocation and the correlation between the signals at the transmitter antennas, i.e. the transmitter signal covariance matrix.

In reference [34] the value of the secrecy capacity is obtained under the transmit covariance

constraint. Some of the characteristics of the optimal power distribution at the transmitter antennas are obtained in [35]. The model presented there assumes that only the Ev channel power gain is known at the transmitter. An isotropic channel model is used to obtain tight upper and lower bounds for the secrecy capacity. The secrecy capacity value is monotonously increasing with the amount of the available power at the transmitter. The impact of Ev is more pronounced in the high SNR regime. The low SNR regime adapts better to the secrecy requirement; the effects of the eavesdropper are less pronounced when the channel gains are small. If the eavesdropper's channel is weaker than the receiver's, the effects of Ev are negligible.

Knowledge of the optimal transmitter covariance matrix can also be exploited opportunistically in fading channels. Bursts of information can be sent while the eavesdropper channel is weak.

Strategies for transmit signaling when the Ev characteristics are not known have been proposed in several references. Reference [29] comments on the artificial noise signaling in MISO (Multiple Input Single Output). Artificial noise (AN) is also studied using Monte-Carlo simulations in [33]. Transmitting is done in the direction of the Rx and artificial noise is sent in the Rx null. AN is almost optimal at high SNR where knowing the Ev channel does not provide significant gains [35].

However, knowing the eavesdropper channel is required in order to design a wiretap code and select the secure communication appropriately. The use of artificial noise can guarantee a minimum secrecy capacity regardless of the eavesdropper position [33].

Reference [36] discusses the optimality conditions for zero-forcing (or null forming) in the general case. Putting the nulls[6] in the direction of the eavesdropper makes sure that it receives no signal. Water-filling can be done on the remainder of the channel as if the Ev was not there. A low complexity on/off power allocation strategy that achieves near-optimal

[6]Nulls refer to the directions where the beam pattern has a gain of zero.

performance with only the main channel CSI (Channel State Information) is shown in [37].

Possible coding schemes for secure communication over the wiretap channel have been described in recent publications such as [38], [39], [40].

The results presented here are meaningful given a set of assumptions such as having Gaussian noise and known channel matrices. The fading channel case has been described in several publications (see Section 5.2 in [7] for an overview). If the transmitter can estimate the channel state accurately, then fluctuations in the received signal strength can be beneficial for security. Secret communication can take place during the periods when the eavesdropper channel has a lower instantaneous SNR.

Additional aspects of physical layer security are presented in [7] and [41]. These references also address system level concerns.

2.4 Summary

MIMO technologies (also included in the 4G-LTE and IEEE 802.11 standards) use spatial multiplexing to increase the communication system's throughput without using additional bandwidth or transmit power.

Security of data is established using cryptographic protocols at the upper layers of the communications system model. At the physical layer, security is addressed from an information-theoretic point of view. Confidentiality of the information being sent is ensured by imposing zero leakage of the transmitted data to the eavesdropper.

The secrecy capacity can be non-zero when the legitimate receiver's channel has some physical advantage over the eavesdropper's. In wireless channels, where noise is modeled as a Gaussian random variable, secrecy is possible when the main channel SNR is higher than the SNR perceived by the eavesdropper.

For MIMO wireless channels, the secrecy capacity can be expressed as a function of the transmitter covariance matrix under a total power constraint (this is a simplification from the information-theoretic expressions). The two equivalent expressions for the MIMO C_s assume that the Rx and Ev channel side information are known at the transmitter.

An expression for the optimal transmit covariance matrix is hard to find and the general case is still an open problem under the total power constraint. Explicit solutions are known only in a few special cases. They will be reviewed in Chapter 4.

Some coding rules of thumb were mentioned for the case where the eavesdropper's CSI is not readily available at the transmitter. In this case, artificial noise can improve the secrecy capacity by hindering the eavesdropper's ability to intercept the signal.

Chapter 3

System Model

The variables and the notations needed to define the secrecy capacity for a Gaussian MIMO channel are introduced in this chapter. Two expressions for the secrecy capacity are included and briefly explained. The assumptions made when defining the model are also discussed.

The regular MIMO channel is briefly discussed in Chapter 2. The channel capacity for MIMO is [9]:

$$C = \max_{\mathbf{R}\succeq\mathbf{0},\mathrm{tr}\,\mathbf{R}\leq P_T} \ln\det(\mathbf{I}+\mathbf{HRH}^{+}) \tag{3.1}$$

The optimal signaling on this channel is Gaussian and the optimal power allocation follows the water-filling solution [9].

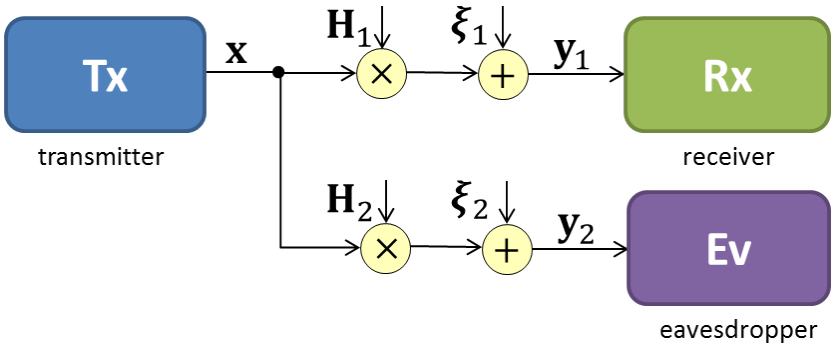

Figure 3.1. MIMO system in the presence of an eavesdropper - block diagram.

Figure 3.1 shows the standard Gaussian wiretap MIMO channel model. At baseband, the receiver and eavesdropper signals are expressed as:

$$\mathbf{y}_1 = \mathbf{H}_1\mathbf{x} + \boldsymbol{\xi}_1, \quad \mathbf{y}_2 = \mathbf{H}_2\mathbf{x} + \boldsymbol{\xi}_2 \tag{3.2}$$

where $\mathbf{x} = [x_1, x_2, \ldots, x_m]^T \in \mathbb{C}^{m\times 1}$ is the complex-valued transmitted signal; $\mathbf{y}_1 \in \mathbb{C}^{n_1\times 1}, \mathbf{y}_2 \in \mathbb{C}^{n_2\times 1}$ are the received signals at the receiver and the eavesdropper respectively; m is the number of transmitter antennas and n_1, n_2 is the number of receiver (Rx) and eavesdropper (Ev) antennas; $\boldsymbol{\xi}_1, \boldsymbol{\xi}_2$ are the circularly-symmetric additive white Gaussian noise (AWGN) at the Rx and the Ev; $\boldsymbol{\xi}_1, \boldsymbol{\xi}_2$ are normalized to unit variance in each dimension. $\mathbf{H}_1 \in \mathbb{C}^{n_1\times m}$ and $\mathbf{H}_2 \in \mathbb{C}^{n_2\times m}$ are the *channel gains matrix* between each Tx and each Rx and between Tx and each Ev antennas respectively (Fig. 3.2 is a graphical illustration). The channel is assumed to be quasistatic and frequency-flat and the channel gains are adjusted to account for noise normalization.

$$\mathbf{H}_1 = \begin{bmatrix} h_{11} & h_{12} \\ h_{21} & h_{22} \\ h_{31} & h_{32} \end{bmatrix}$$

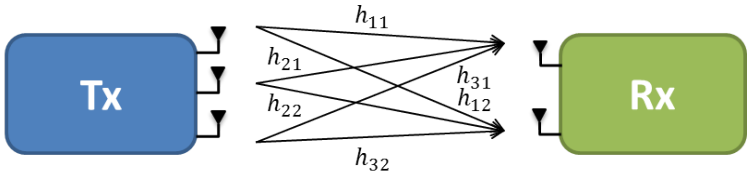

Figure 3.2. MIMO channel gain matrix.

The transmit signal satisfies the average power constraint: $\mathcal{E}\{\|\mathbf{x}\|^2\} = P_T$, where $\mathcal{E}\{\cdot\}$ is the statistical average. Since the noise power is normalized, $\sigma^2 = 1$, the signal to noise ratio (SNR) is:

$$\gamma = \frac{P_T}{\sigma^2} = P_T \tag{3.3}$$

$\mathbf{W}_1 = \mathbf{H}_1^+\mathbf{H}_1$ and $\mathbf{W}_2 = \mathbf{H}_2^+\mathbf{H}_2$ represent the receiver (main) and the eavesdropper *channel matrix* respectively. Both $\mathbf{W}_1$ and $\mathbf{W}_2$ are $m \times m$. $\mathbf{W}_1 - \mathbf{W}_2$ is referred to as the *difference matrix*.

For a given transmit signal covariance, $\mathcal{E}\{\mathbf{x}\mathbf{x}^+\} = \mathbf{R}, \operatorname{tr}\mathbf{R} \leq P_T$, the secrecy rate is [29], [30], [31]:

$$C_s(\mathbf{R}) = \ln \frac{\det(\mathbf{I} + \mathbf{H}_1\mathbf{R}\mathbf{H}_1^+)}{\det(\mathbf{I} + \mathbf{H}_1\mathbf{R}\mathbf{H}_1^+)} = \ln \frac{\det(\mathbf{I} + \mathbf{W}_1\mathbf{R})}{\det(\mathbf{I} + \mathbf{W}_2\mathbf{R})} \tag{3.4}$$

the secrecy capacity is:

$$C_s = \max_{\mathbf{R}\succeq\mathbf{0}, \operatorname{tr}\mathbf{R}\leq P_T} \ln \frac{\det(\mathbf{I} + \mathbf{W}_1\mathbf{R})}{\det(\mathbf{I} + \mathbf{W}_2\mathbf{R})} \tag{3.5}$$

and

$$\mathbf{R}^* = \arg \max_{\mathbf{R}\succeq\mathbf{0}, \operatorname{tr}\mathbf{R}\leq P_T} C_s(\mathbf{R}) \tag{3.6}$$

is the optimal covariance matrix, so that $C_s = C_s(\mathbf{R}^*)$.

An equivalent expression to (3.5) is given in [29], [32]:

$$C_s = \min_{\mathbf{K}} \max_{\mathbf{R}} \ln \frac{\det(\mathbf{I} + \mathbf{K}^{-1}\mathbf{H}\mathbf{R}\mathbf{H}^+)}{\det(\mathbf{I} + \mathbf{W}_2\mathbf{R})} \tag{3.7}$$

with

$$\mathbf{H} = \begin{bmatrix} \mathbf{H}_1 \\ \mathbf{H}_2 \end{bmatrix}, \mathbf{K} = \begin{bmatrix} \mathbf{I} & \mathbf{\Phi} \\ \mathbf{\Phi}^+ & \mathbf{I} \end{bmatrix}, \mathbf{R} \succeq \mathbf{0}, \operatorname{tr}\mathbf{R} \leq P_T \tag{3.8}$$

where $\mathbf{\Phi}$ is such that the $\mathbf{K}$ matrix in the noise covariance matrix between the receiver and the eavesdropper channels. The meaning of (3.7) is explained in [32] using an information-theoretic argument. It is briefly reviewed below.

Consider the system in Fig. 3.3. Since the channel is Gaussian, the secrecy capacity can be expressed as [7]:

$$C_s = \max_X (\mathbb{I}(X; Y_1) - \mathbb{I}(X; Y_2)) \tag{3.9}$$

(3.9) is upper bounded by the secrecy capacity that can be achieved when the receiver is aware of the eavesdropper's signal (assume that a genie informs the receiver about the value

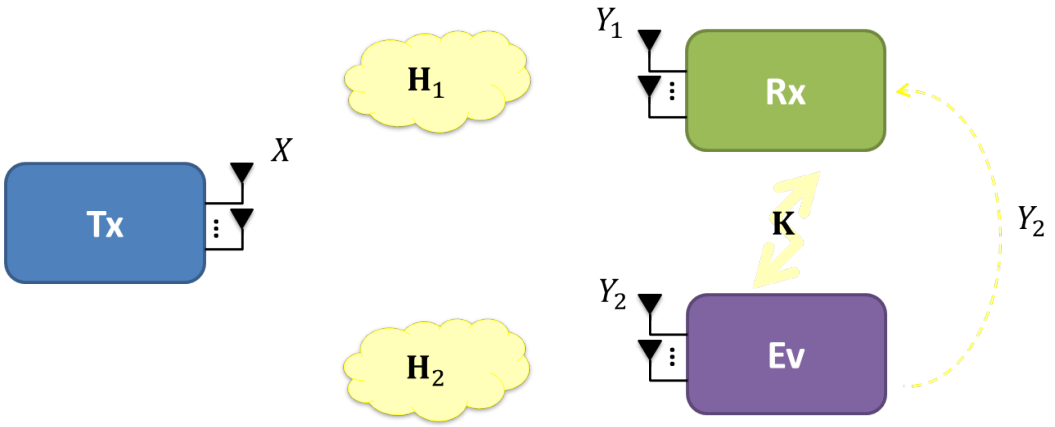

Figure 3.3. Information theoretic argument for the secrecy capacity upper bound.

of Y_2). Since $\mathbb{I}(X;Y_1) \leq \mathbb{I}(X;Y_1,Y_2)$ [25]:

$$C_s \leq \max_X (\mathbb{I}(X;Y_1,Y_2) - \mathbb{I}(X;Y_2)) \tag{3.10}$$

This is akin to having a degraded broadcast channel. Fig. 3.4 describes this situation. Since the receiver is made aware of Y_2, it is as if the system had a larger (equivalent) receiver to code for. In (3.7), $\mathbf{K}$ is the noise covariance between the receiver and eavesdropper communication channels[1]. Noise affects each of the Rx and Ev channels independently ($\mathbf{I}$ on the diagonal). However, the noise affecting the receiver may be correlated to the one affecting the eavesdropper ($\mathbf{\Phi}$).

The upper bound in (3.10) is similar to the lower bound in (2.20). Writing (3.10) in terms of $\mathbf{W}_1, \mathbf{W}_2$ and $\mathbf{R}$ results in a min-max expression, (3.7), or the secrecy capacity upper bound.

Unless otherwise specified, $C_s(\mathbf{R})$ is in [nat/s/Hz]. In the figures, the secrecy capacity is expressed in [bit/s/Hz].

[1]The maximum number of communication channels between Tx and Rx or the Tx and the Ev is $\min(m, n_1) \leq m$ or $\min(m, n_2) \leq m$ respectively.

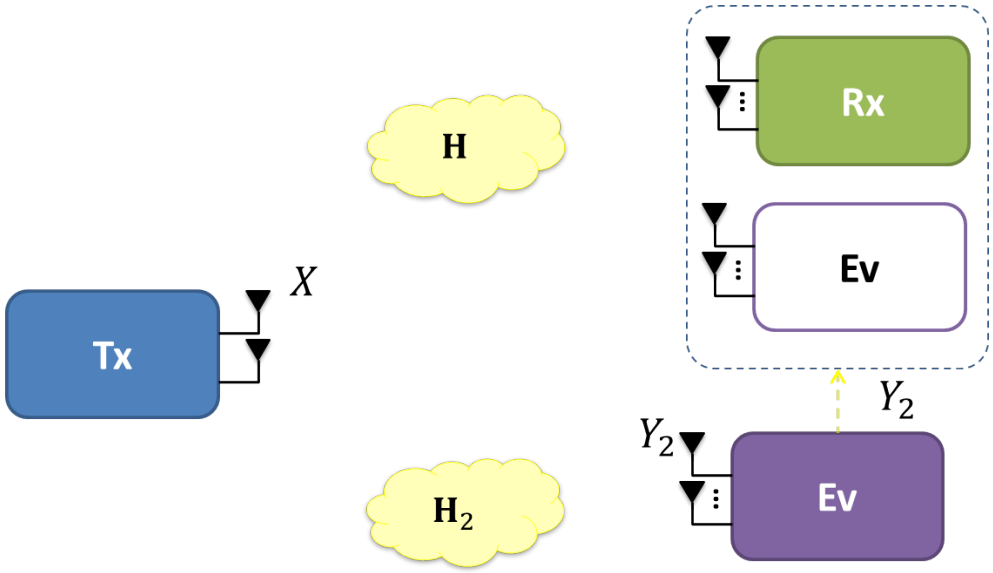

Figure 3.4. Secrecy capacity upper bound—equivalent degraded channel.

Assumptions and Their Implications

Equations (3.5) and (3.7) are accurate as long as the Rx and Ev channel state information (CSI) are known at all terminals. Because of limited feedback, the channel gains would be known with less accuracy and the secrecy capacity predicted by the model would not be achieved.

Noise is modeled as a Gaussian random variable and there are no restrictions on the transmitter modulation scheme. However, practical systems use discrete alphabets and $\mathbf{R}^*$ may not correspond to the best power allocation.

Chapter 4

Known Solutions

The non-convexity of the secrecy capacity expression makes it hard to optimize in general. Results are known in a few special cases and, although they rely on additional assumptions on the system model, they provide insight in how secrecy can be achieved. Transmit covariance matrices are specified by $\frac{1}{2}(m^2 + m)$ independent quantities: m of them define the antenna power distribution under the total power constraint and the rest specify the correlation coefficients between the antenna signals.

Some of the results reviewed in this chapter show the optimal covariance matrix in an explicit form. Others provide $\mathbf{R}^*$ as the solution to a set of linear equations which is easier to solve than the original maximization problem. Simple examples are used to describe some of the secrecy capacity curve behavior. This chapter completes the literature review.

4.1 Convexity of the Secrecy Capacity Expression

The attainable secrecy rate is a function of the transmit covariance matrix $\mathbf{R}$:

$$C_s(\mathbf{R}) = \ln \frac{\det(\mathbf{I} + \mathbf{W}_1\mathbf{R})}{\det(\mathbf{I} + \mathbf{W}_2\mathbf{R})} \tag{4.1}$$

Since $\ln\det(\mathbf{I}+\mathbf{HRH}^{+})$ is a concave function of $\mathbf{R}$ [9], as a difference of concave functions, $C_s(\mathbf{R})$ is neither concave nor convex, in general. When

- $\mathbf{W}_1 \succeq \mathbf{W}_2$: The secrecy capacity function, $C_s(\mathbf{R})$, is concave in $\mathbf{R}$ [42]. $\mathbf{R}^*$ is obtained in a closed form in some special cases such as when the channel gain matrices $\mathbf{H}_1$ and $\mathbf{H}_2$ have the same right-singular vectors (Section 4.6) or when the difference matrix, $\mathbf{W}_1-\mathbf{W}_2$, has only one positive eigenvalue (Section 4.5).
- $\mathbf{W}_1 \preceq \mathbf{W}_2$: The eavesdropper has a better channel than the receiver and the secrecy capacity is zero.
- $\mathbf{W}_1 \not\succeq \not\preceq \mathbf{W}_2$: The secrecy capacity expression is challenging to optimize.

Secure signaling is done in the directions where the receiver's channel is stronger than the eavesdropper's [29]. The rank of the optimal covariance matrix does not exceed the number of positive eigenvalues of $\mathbf{W}_1-\mathbf{W}_2$ [35], [43]. Moreover, corresponding eigenvectors span the same space as $\mathbf{R}^*$ [35].

4.2 Limiting Behavior of C_s with respect to P_T

Secrecy Capacity at Low SNR

At low SNR, signaling in the strongest direction of the difference matrix $\mathbf{W}_1-\mathbf{W}_2$ is optimal [35], [44]. An approximation for C_s in this regime is:

$$\begin{aligned} C_s(\mathbf{R}) = \ln\frac{\det(\mathbf{I}+\mathbf{W}_1\mathbf{R})}{\det(\mathbf{I}+\mathbf{W}_2\mathbf{R})} &\approx \operatorname{tr}\left((\mathbf{W}_1-\mathbf{W}_2)\mathbf{R}\right), \quad \mathbf{W}_1\mathbf{R}\ll\mathbf{I}, \mathbf{W}_2\mathbf{R}\ll\mathbf{I} \\ &\leq \lambda_1 P_T \end{aligned} \tag{4.2}$$

where λ_1 is the largest eigenvalue of $\mathbf{W}_1-\mathbf{W}_2=\mathbf{H}_1^{+}\mathbf{H}_1-\mathbf{H}_2^{+}\mathbf{H}_2$. Equality in (4.2) is achieved when:

$$\mathbf{R}^* = P_T\mathbf{u}_1\mathbf{u}_1^{+} \tag{4.3}$$

where $\mathbf{u}_1$ is the eigenvector corresponding to λ_1. If λ_1 has a multiplicity larger than 1 then transmitting in the positive direction of the corresponding eigen-space is optimal.

Secrecy Capacity at High SNR

At high values of SNR, the log-determinant can be approximated as:

$$C_s(\mathbf{R}) = \ln \frac{\det(\mathbf{I} + \mathbf{W}_1\mathbf{R})}{\det(\mathbf{I} + \mathbf{W}_2\mathbf{R})} \approx \ln \frac{\det \mathbf{W}_1}{\det \mathbf{W}_2}, \quad \mathbf{W}_1\mathbf{R} \gg \mathbf{I}, \mathbf{W}_2\mathbf{R} \gg \mathbf{I} \tag{4.4}$$

The conditions imply that both channel matrices need to be full rank. The secrecy capacity value does not depend on $\mathbf{R}$. This saturation effect is observed in [32], [35].

Example 4.1: *Secrecy capacity saturation effect at high SNR*

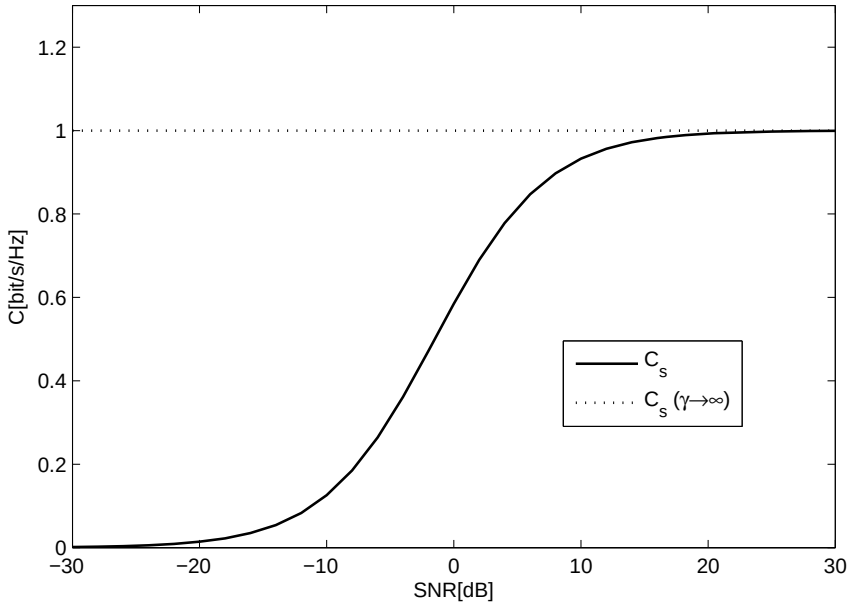

Figure 4.1. High SNR approximation for C_s.

Consider:

$$\mathbf{W}_1 = \begin{bmatrix} 2 & 0 \\ 0 & 1 \end{bmatrix}, \quad \mathbf{W}_2 = \mathbf{I}_2. \tag{4.5}$$

Fig. 4.1[1] shows the secrecy capacity C_s values over a wide SNR range.

[1]The C_s curve was obtained using a Monte-Carlo optimization with 10^5 iterations for each SNR value.

Both channel matrices are full rank and the secrecy-capacity curve saturates at:

$$\lim_{P_T \to \infty} C_s = \ln \frac{\det \mathbf{W}_1}{\det \mathbf{W}_2} = \ln \frac{2}{1} \rightarrow C_s = 1 \text{ bit/s/Hz} \tag{4.6}$$

When $\mathbf{W}_2$ is not full rank the value of the secrecy capacity can become very large at high SNR. The following example illustrates this behavior.

Example 4.2: *Secrecy capacity does not saturate at high SNR*

Consider:

$$\mathbf{W}_1 = \mathbf{I}_2, \quad \mathbf{W}_2 = \frac{1}{2}\mathbf{E}_2 \tag{4.7}$$

where $\mathbf{E}_2$ is the 2×2 all ones matrix. Fig. 4.2[2] shows the secrecy capacity value over a wide SNR range. The secrecy capacity C_s increases at a slower rate than C_{Rx}. Nonetheless, at high values of SNR, C_s does not saturate, i.e. it increases indefinitely.

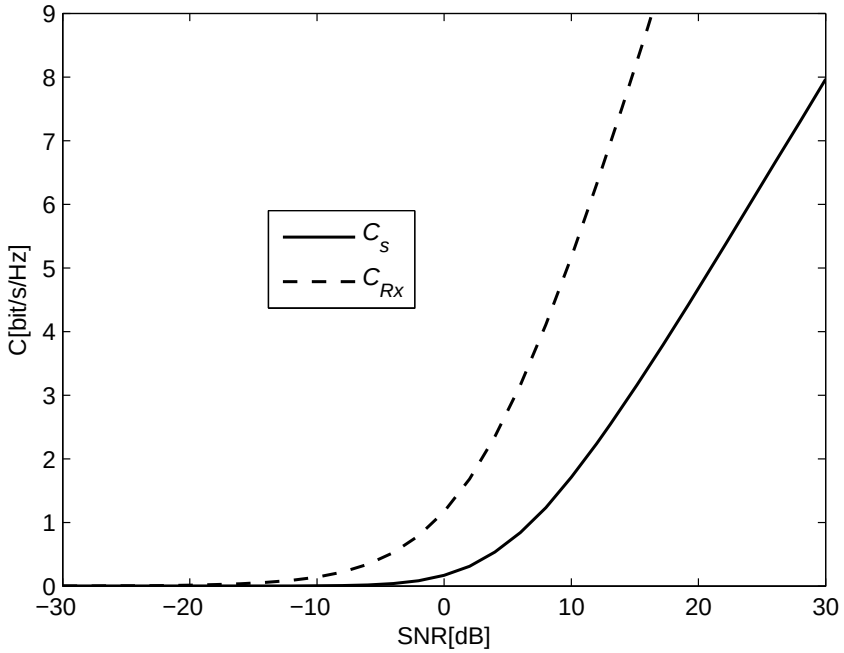

Figure 4.2. Secrecy capacity C_s compared to channel capacity C_{Rx}.

The $C_s(\gamma \to \infty)$ curve was plotted using (4.4).

[2]The C_s and C_{Rx} curves are obtained using Monte-Carlo optimizations with 10^5 iterations for each SNR value.

4.3 The Receiver and the Eavesdropper Have One or Two Antennas

Case 1: $n_1 = n_2 = 1$, m can be any

An analytical solution for the optimal covariance matrix when the receiver and the eavesdropper both have only one antenna is [45]:

$$\mathbf{R}^* = P_T \mathbf{u}_1 \mathbf{u}_1^+ \tag{4.8}$$

where $\mathbf{u}_1$ is the eigenvector corresponding to the largest eigenvalue of $\left(\mathbf{I} + P_T \mathbf{h}_2 \mathbf{h}_2^+\right)^{-1}$ $\left(\mathbf{I} + P_T \mathbf{h}_1 \mathbf{h}_1^+\right)$. $\mathbf{h}_1$ and $\mathbf{h}_2$ are the Rx and the Ev channel gain vectors.

The optimal solution uses all the available transmit power. Numerical evaluations in [45] show that the more correlated the channels are, the lower the secrecy rate when the Rx and Ev channel gains are kept constant. However, the chance of this happening is small when the number of transmit antennas is large [45].

Case 2: $n_1 = 1$, n_2, m can be any

The optimal signaling strategy scheme for a scalar Gaussian channel is discussed in [29], [46]. Optimal signaling requires beamforming and the direction depends on both $\mathbf{h}_1$ (the receiver channel gain vector) and $\mathbf{H}_2$:

- At low SNR, the optimal signaling strategy is beamforming on the eigenvector corresponding to the largest eigenvalue of $\mathbf{h}_1 \mathbf{h}_1^+ - \mathbf{H}_2^+ \mathbf{H}_2$. The result can be generalized to the case where the receiver has multiple antennas (see section 4.2).
- At high SNR, the power of the received signal has less impact on the value of the

secrecy capacity and it is optimal to transmit on the null space of $\mathbf{H}_2$ as long as $\mathbf{h}_1$ is not orthogonal to the null of $\mathbf{H}_2$.

Case 3: $m = n_1 = 2, n_2 = 1$

For this case, reference [47] shows that, when $\mathbf{W}_2$ is full rank, the rank of the optimal covariance matrix is 1 and beamforming is the optimal transmit strategy under the total power constraint:

$$\mathbf{R}^* = P_T \mathbf{r}\mathbf{r}^+ \tag{4.9}$$

where

$$\mathbf{r} = \frac{\mathbf{B}^{-1/2}\mathbf{u}_1}{\|\mathbf{B}^{-1/2}\mathbf{u}_1\|} \tag{4.10}$$

where $\mathbf{u}_1$ is the eigenvector corresponding to the largest eigenvalue of $\mathbf{B}^{-1/2}\mathbf{A}\mathbf{B}^{-1/2}$ and:

$$\begin{aligned} \mathbf{A} &= \mathbf{I} + P_T \mathbf{H}_1^+ \mathbf{H}_1 \\ \mathbf{B} &= \mathbf{I} + P_T \mathbf{h}_2^+ \mathbf{h}_2 \end{aligned} \tag{4.11}$$

where $\mathbf{h}_2$ is the eavesdropper channel gain vector. In [47] the result is shown to be optimal by constructing a tight upper bound that meets the same achievable secrecy rate as $\mathbf{R}^*$.

4.4 The Degraded Full Rank Channel

When $\mathbf{W}_1 \succ \mathbf{W}_2 \succeq \mathbf{0}$ the eavesdropper channel is degraded with respect to the main channel[3]. When the optimal covariance matrix is full rank and an explicit closed-form solution for $\mathbf{R}^*$ is obtained in [35, (10)-(12)].

[3]There are two types of degraded channels: physically degraded or stochastically degraded. In the first case $X \to Y_1 \to Y_2$ form a Markov chain. In the second case, the channel is statistically equivalent to a physically degraded channel.

4.5 The Difference Matrix has One Positive Eigenvalue

When $\operatorname{rank}(\mathbf{W}_1 - \mathbf{W}_2)_+ = 1$, $\operatorname{rank}\mathbf{R}^* = 1$ [35]:

$$\mathbf{R}^* = P_T \mathbf{u}_1 \mathbf{u}_1^+ \quad \text{and} \quad C_s = \ln \lambda_1 \tag{4.12}$$

where $\mathbf{u}_1$ is the eigenvector corresponding to λ_1, the largest eigenvalue of $(\mathbf{I} + P_T\mathbf{W}_2)^{-1}(\mathbf{I} + P_T\mathbf{W}_1)$. The optimal signaling strategy is beamforming along $\mathbf{u}_1$.

4.6 The Channel Gain Matrices have Identical Right Singular Vectors

The right singular vectors characterize the scattering environment around the transmitter [9], [36]. When $\mathbf{H}_1$ and $\mathbf{H}_2$ have the same right singular vectors, the optimal covariance matrix is known and its expression is provided in [36]:

$$\mathbf{R}^* = \mathbf{V}\mathbf{\Lambda}^*\mathbf{V}^+ \tag{4.13}$$

where $\mathbf{V}$ is the right singular vector matrix for $\mathbf{H}_{1(2)}$ and $\mathbf{\Lambda}^*$ is a diagonal matrix with diagonal entries:

$$\mathbf{\Lambda}(i,i) = \frac{\lambda_{2i} + \lambda_{1i}}{2\lambda_{2i}\lambda_{1i}} \left(\sqrt{1 + \frac{4\lambda_{2i}\lambda_{1i}}{(\lambda_{2i} + \lambda_{1i})^2} \left(\frac{\lambda_{1i} - \lambda_{2i}}{\lambda} - 1 \right)_+} - 1 \right) \tag{4.14}$$

where λ_{1i} and λ_{2i} are the eigenvalues of $\mathbf{W}_1$ and those of $\mathbf{W}_2$ and λ is obtained from the total power constraint $\operatorname{tr}\mathbf{R}^* = P_T$.

Optimal signaling for secrecy resembles MIMO beamforming: the difference matrix eigenvalues indicate the quality of the communication paths; its eigenvectors indicate the direction

of the communication paths. However, the optimal covariance matrix solution takes into account the eavesdropper's presence and the secrecy requirement and differs from regular water-filling [35]:

- When the goal is to maximize the secrecy rate, the stronger eigen-modes of the difference matrix (and not of $\mathbf{W}_1$) receive the larger power.
- At high SNR strong eigen-modes get more power. Unlike in water-filling, uniform power allocation is not optimal.
- When the eavesdropper is weak, $\mathbf{W}_2\mathbf{R} \ll \mathbf{I}$, the modes that spill over into the eavesdropper channel get less power than they would have if water-filling was used between Tx and Rx.

4.7 The Weak Eavesdropper Case

When the eavesdropper's channel is weak (for example when it is situated far from the base-station or when the Rx needs to be close to the Tx for communication to happen), the secrecy rate given a transmit covariance $\mathbf{R}, \operatorname{tr}\mathbf{R} \leq P_T$, can be approximated with:

$$C_{\text{tr}}(\mathbf{R}) = \ln\det(\mathbf{I} + \mathbf{W}_1\mathbf{R}) - \operatorname{tr}(\mathbf{W}_2\mathbf{R}), \quad \mathbf{W}_2\mathbf{R} \ll \mathbf{I} \tag{4.15}$$

(4.15) is a concave expression in $\mathbf{R}, \forall\mathbf{W}_{1,2}$ and its optimum be evaluated by solving the necessary Karush-Khum-Tucker (KKT) conditions[4] as done in [36].

Let $\mathbf{R}^*_{\text{tr}}$ be the covariance matrix that maximizes (4.15) and let $C_{\text{tr}} = C_{\text{tr}}(\mathbf{R}^*_{\text{tr}})$. (4.15) underestimates the secrecy capacity[5]:

$$C_{\text{tr}}(\mathbf{R}^*_{\text{tr}}) \leq C_s(\mathbf{R}^*_{\text{tr}}) \tag{4.16}$$

[4]Many of the results here rely on convex optimization techniques. These are explained in depth in [23].
[5]Since since $\ln(1+x) \leq x, \forall x > -1$, $\sum \lambda_{\mathbf{A}} \geq \sum \ln(1+\lambda_{\mathbf{A}})$ and $\ln\det(\mathbf{I}+\mathbf{A}) \leq \operatorname{tr}\mathbf{A}, \mathbf{A} \succeq \mathbf{0}$.

The tr term increases with power much faster than the ln det and at high SNR the optimal covariance matrix, $\mathbf{R}_{\text{tr}}^*$, does not make use of the full available power. The maximal power P_s where full transmit power is used is:

$$P_s = \text{tr}\left(\mathbf{W}_2^{-1}(\mathbf{I} - \mathbf{W}_2^{-\frac{1}{2}}\mathbf{W}_1^{-1}\mathbf{W}_2^{-\frac{1}{2}})\right) \leq P_T \tag{4.17}$$

When $P_T \geq P_s$, $\mathbf{R}_{\text{tr}}^*$ no longer depends on P_T.

$$\mathbf{R}_{\text{tr}}^* = \mathbf{W}_2^{-1}\left(\mathbf{I} - \mathbf{W}_2^{-1/2}\mathbf{W}_1^{-1}\mathbf{W}_2^{-1/2}\right)_+ \tag{4.18}$$

When $\mathbf{R}_{\text{tr}}^*$ is full rank:

$$\mathbf{R}_{\text{tr}}^* = \mathbf{W}_2^{-1} - \mathbf{W}_1^{-1} \tag{4.19}$$

When $P_T < P_s$:

$$\begin{aligned} \mathbf{R}_{\text{tr}}^* &= \mathbf{A}^{-1/2}(\mathbf{I} - \widetilde{\mathbf{W}}_1^{-1})_+\mathbf{A}^{-1/2} \\ \widetilde{\mathbf{W}}_1 &= \mathbf{A}^{-1/2}\mathbf{W}_1\mathbf{A}^{-1/2}, \mathbf{A} = \lambda\mathbf{I} + \mathbf{W}_2 \end{aligned} \tag{4.20}$$

where λ is the Lagrange multiplier corresponding to the power constraint in the KKT conditions and is chosen such that $\text{tr}\,\mathbf{R}_{\text{tr}}^* = P_T$. λ is a monotonically decreasing function of P_T. If $P_T \geq P_s$, $\lambda = 0$. An upper bound for λ is $\frac{1}{P_T}$ which follows from the fact that the maximal amount of power allocated in any direction is P_T: $\lambda_i(\mathbf{R}) \leq P_T$ i.e. all of $\mathbf{R}$'s eigenvalues are less than P_T. The flowchart in Fig. 4.3 explains how to find $\mathbf{R}_{\text{tr}}^*$.

Example 4.3: *Capacity approximation in the weak eavesdropper case*

Let:

$$\mathbf{W}_1 = \begin{bmatrix} 2 & 0 \\ 0 & 1 \end{bmatrix}, \quad \mathbf{W}_2 = \begin{bmatrix} 2 & 1 \\ 1 & 1 \end{bmatrix}. \tag{4.21}$$

Fig. 4.4[6] shows the optimal value of C_{tr} and C_s at different values of SNR. At all SNR

[6]The $C_{\text{tr}}(\mathbf{R}_{\text{tr}}^*)$ curve was obtained following the flow-chart in Fig.4.3. The C_s and *Approximation(MC)*

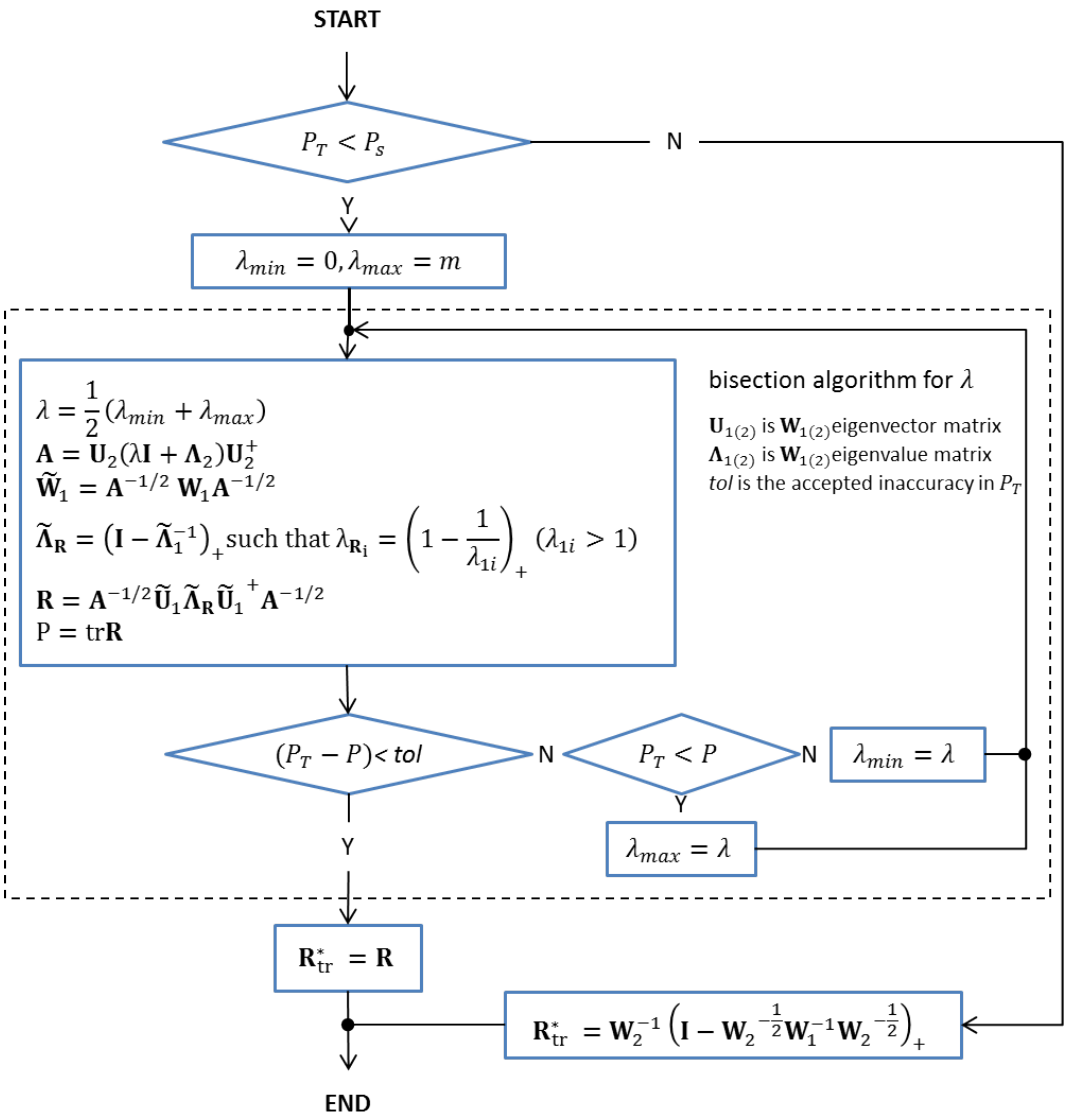

Figure 4.3. Covariance matrix optimization in the case of weak eavesdropper.

values, the Monte-Carlo solution is below or equal to the analytical solution (see Fig. 4.4). This is clear at SNR = 20dB.

When $P_T \leq P_s$, C_{tr} is very close to the value of C_s. P_s is a threshold indicating where the approximation in (4.15) is no longer sufficiently accurate. When $P_T \geq P_s$ the optimal covariance matrix is not unique[7]. The analytical solution in [36] gives an optimal covariance matrix such that $\text{tr}\,\mathbf{R}_{\text{tr}}^* = P_s$. Hence, the $\mathbf{R}_{\text{tr}}^*$ obtained with

curves were obtained using a Monte-Carlo optimization using 10^5 iterations for each SNR value.

[7]The power constraint is respected with inequality and the solution is not on the boundary.

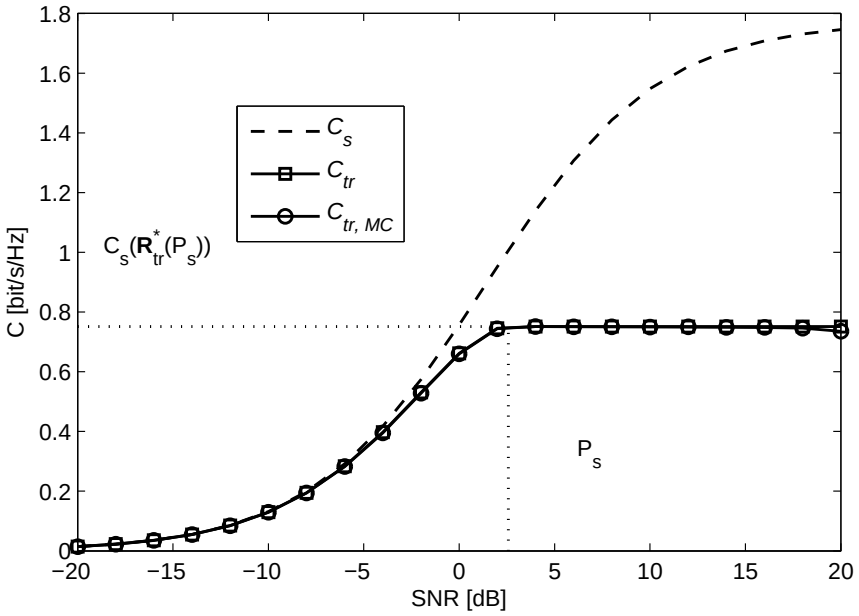

Figure 4.4. Weak eavesdropper approximation C_{tr}.

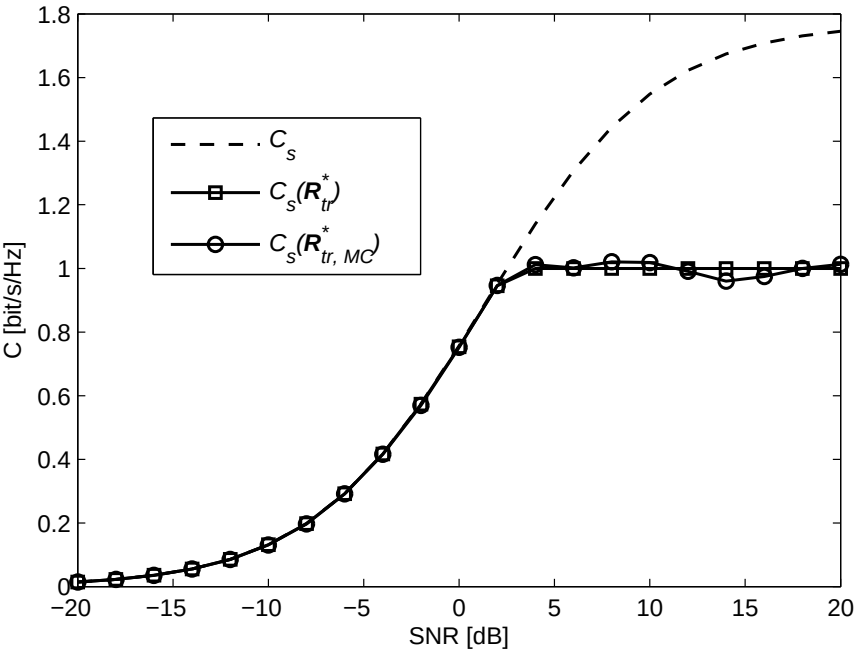

Figure 4.5. Weak eavesdropper approximation $C_s(\mathbf{R}^*_{\mathrm{tr}})$.

the analytical solution is the solution having the smallest trace among all the optimal covariance matrices, i.e. the solution that uses the least amount of power. Using less

than P_s would be sub-optimal and using more than P_s means that power is allocated in directions that do not increase the value of C_{tr}.

Using $\mathbf{R}_{\text{tr}}^*$ as the transmitter covariance matrix in the C_s expression (see (3.5)) results in the secrecy rates in Fig. 4.5[8]. Let $\mathbf{R}_{\text{tr},MC}^*$ be the optimal covariance matrix returned by the Monte-Carlo algorithm. The oscillations in the $C_s(\mathbf{R}_{\text{tr},MC}^*)$ curve are due to the non-uniqueness of $\mathbf{R}_{\text{tr}}^*$. Sometimes, $C_s(\mathbf{R}_{\text{tr},MC}^*) \geq C_s(\mathbf{R}_{\text{tr}}^*)$. This is to be expected since the optimization is performed on the approximate function in (4.15).

Example 4.4: *Capacity approximation in the weak eavesdropper case,* $\mathbf{W}_2$ *is not full rank*

Let:

$$\mathbf{W}_1 = \begin{bmatrix} 2 & 1 \\ 1 & 1 \end{bmatrix}, \quad \mathbf{W}_2 = \frac{1}{10}\mathbf{E}_2. \tag{4.22}$$

In this case, $P_s = \infty$ and there is no saturation effect. Hence, $\operatorname{tr}\mathbf{R}^* = P_T, \forall P_T$. Fig. 4.6 and 4.7 illustrate this.

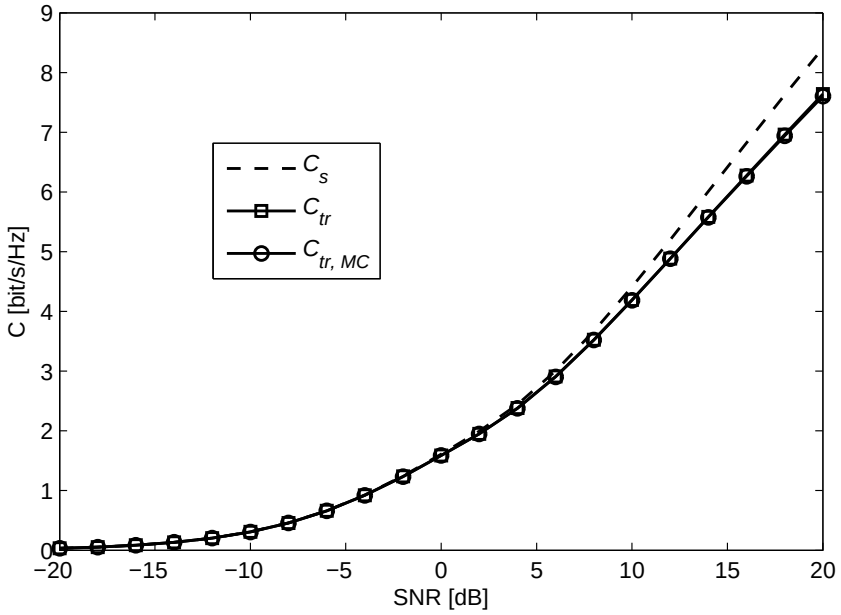

Figure 4.6. Weak eavesdropper approximation C_{tr}.

[8]The C_s curve is the same as in Fig.4.4 The other two curves were obtained by using $\mathbf{R}_{\text{tr}}^*$ in the secrecy capacity expression.

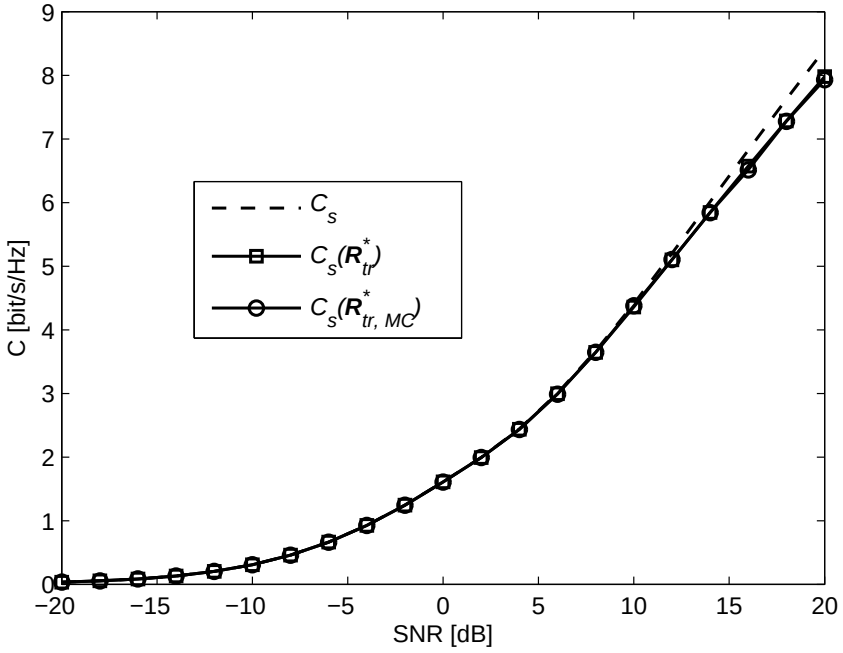

Figure 4.7. Weak eavesdropper approximation $C_s(\mathbf{R}^*_{\mathrm{tr}})$.

The approximate solution is a lower bound to the secrecy capacity since $\ln\det(\mathbf{I}+\mathbf{A}) \geq \operatorname{tr}\mathbf{A}, \mathbf{A} \succeq \mathbf{0}$.

4.8 Summary

The secrecy capacity is not concave in $\mathbf{R}$ and finding the optimal covariance matrix is hard in general. Under the total power constraint, some of its characteristics are known.

The optimal signaling strategy is discussed at all SNR values when the eavesdropper has only one antenna and at low SNR in the general case. When the channel matrices are full rank, the secrecy capacity curve saturates at high SNR.

When maximizing (3.5), $\mathbf{R}^*$ makes use of all the available power. An important property of $\mathbf{R}^*$ is that its rank is not larger than the number of positive eigenvalues of $\mathbf{W}_1 - \mathbf{W}_2$ and that its range is included in the positive eigen-space of the difference matrix. From this,

obtaining $\mathbf{R}^*$ when $\text{rank}(\mathbf{W}_1 - \mathbf{W}_2)_+ = 1$ is immediate.

$\mathbf{R}^*$ is known when $\mathbf{H}_1$ and $\mathbf{H}_2$ have the same right singular vectors and also in the degraded channel case ($\mathbf{W}_1 \succeq \mathbf{W}_2$) when $\mathbf{R}^*$ is full rank. In both of these cases $\mathbf{R}^*$ is found by solving (different) systems of linear equations. When $\mathbf{W}_2 = \mathbf{0}$, the solutions reduce to regular water-filling.

The weak eavesdropper channel is another special case. The secrecy capacity optimization problem is simplified by linearizing the eavesdropper part of the equation. The approximation under-estimates the true secrecy capacity and is accurate when $\mathbf{W}_2\mathbf{R} \ll \mathbf{I}$.

Some of the analytical results obtained in [35], [36] were implemented numerically and tested in a few simple scenarios thus confirming the conclusions in the references. The MATLAB code used to generate the plots is provided in the Appendix.

Chapter 5

Notes on Numerical Solvers

A simple approach to finding the secrecy capacity is to use readily-available numerical solvers. This does not require analytical manipulation or in-depth mathematical knowledge of numerical optimization algorithms. Accurate results are returned as long as the problem is specified in a suitable programming language. All optimization steps happen "behind the scenes". The two solvers considered are CVX and YALMIP. They were chosen because of their popularity[1], ease of use and cost (free).

CVX is widely used to solve convex problems. CVX is an umbrella to several optimization tools[2]: SeDuMi, SDP3, Mosek, Gurobi, Glpk. CVX provides the user with a unified way of accessing these solvers without requiring an advanced knowledge of the particular solver. CVX can be deployed as a MATLAB toolbox. CVX accepts problems that respect the disciplined convex optimization rules outlined in the CVX user manual [10].

YALMIP is a modeling language and is implemented as a free toolbox for MATLAB. It accepts the data provided, but it does not return accurate results in general.

[1] CVX has been cited 1747 times while YALMIP has been cited 2842 times (as counted by Google Scholar on Oct 23rd, 2014).

[2] Mosek is free for academia, Gurobi requires a commercial CVX+Gurobi package and Glpk is not supported in MATLAB.

When optimizing the secrecy capacity expression, some limitations appear in the numerical solvers due to the $\ln\det$ function. These affect the way that the secrecy capacity function is handled by the two numerical solvers. This chapter shows, by means of example, that the results to the secrecy capacity optimization problem are not reliable when obtained with CVX or YALMIP.

5.1 CVX

CVX is capable of solving complex optimization problems as long as they respect a strict set of rules. CVX is continuously evolving and the software and user guide can be found at [10]. When it solves an optimization problem, CVX returns what it considers the optimal function value and it also keeps the arguments that generated it stored as a variable.

CVX cannot optimize the secrecy capacity expression directly. However, it is being used to optimize objective functions that are linear in $\mathbf{R}$ (see Chapter 7).

Optimizing with CVX

First, CVX is used to solve an optimization problem for which the results are well known: finding the regular MIMO channel capacity. This is done in order to understand how to chose the CVX programming parameters and the type of results that this solver provides.

The MIMO channel capacity is written as:

$$C = \max_{\mathbf{R}\succeq\mathbf{0},\operatorname{tr}\mathbf{R}\leq P_T} \ln\det(\mathbf{I}+\mathbf{WR}) \tag{5.1}$$

The optimal transmitter covariance matrix in this case is obtained analytically using the water-filling algorithm [9].

The objective function in the maximization problem is concave and respects the rules of disciplined convex programming, therefore the problem can be solved using CVX. A possible implementation of the CVX solution in MATLAB is illustrated in Fig. 5.1.

CVX code	Optimization problem
cvx_begin variable R(m, m) symmetric R == semidefinite(m); C = log_det(I+W*R); maximize C 0 <= trace(R) <= snr_lin; cvx_end	maximize $C(\mathbf{R}) = \ln\det(\mathbf{I} + \mathbf{WR})$ subject to $\mathbf{R} \succcurlyeq \mathbf{0}$, $\mathrm{tr}\mathbf{R} \geq 0$, $\mathrm{tr}\mathbf{R} \leq P_T$, $\mathbf{R} \in \mathbb{R}^{m\times m}$

Figure 5.1. CVX code for finding the optimal covariance matrix over the MIMO channel.

Example 5.1: *Using CVX to solve the power allocation problem over the MIMO channel*

Consider a MIMO system having the Tx-Rx channel matrix:

$$\mathbf{W} = \frac{1}{m}\mathbf{E}_m, \quad m \in \{2, 16\} \tag{5.2}$$

Since $\mathrm{rank}(\mathbf{W}) = 1$, $\mathrm{rank}(\mathbf{R}^*) = 1$ and $\mathbf{R}^*$ has the same eigenvector as $\mathbf{W}$:

$$\mathbf{R}^* = \frac{\gamma}{m}\mathbf{E}_m \tag{5.3}$$

where $\mathbf{E}_m$ is the all-ones matrix. The solution obtained with CVX is compared with the analytical solution in Fig. 5.2. C is the channel capacity obtained analytically, as explained in Chapter 2. $C_{m=2}$ and $C_{m=16}$ are the channel capacity values returned by CVX. The two solutions are numerically identical. The $m = 2$ and $m = 16$ curves are the same because the channel gains are adjusted to the number of antennas.

When compiling the program, CVX issues a warning relative to the use of functions in the exponential family[3]. When $\ln\det$ is the only term in the objective function, such as in the

[3]CVX issues the following warning: "CVX Warning: Models involving "log" or other functions in the log,

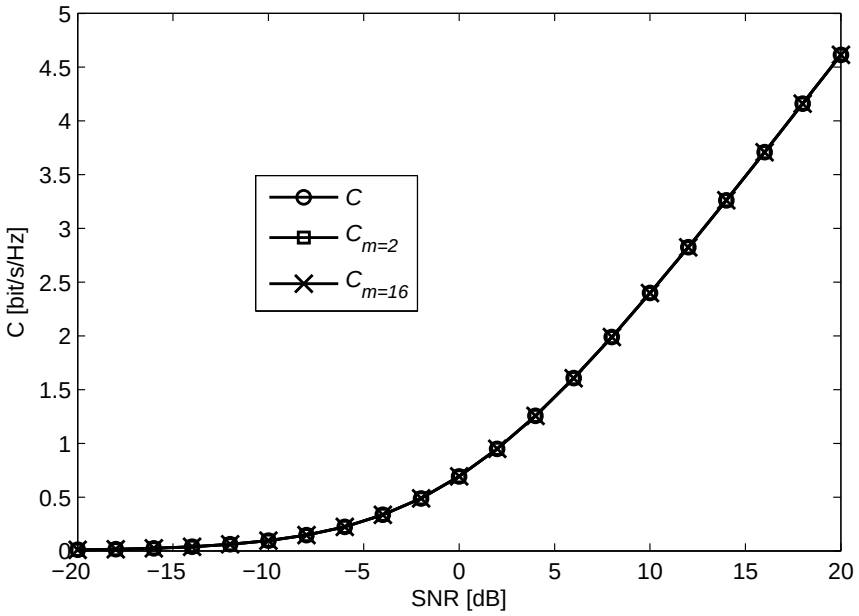

Figure 5.2. CVX optimization for MIMO.

example above, the problem formulation can be modified such that the software can handle it properly:

$$\begin{aligned} \mathbf{R}^* &= \arg\max_{\mathbf{R}} \ln\det(1+\mathbf{WR}) = \arg\max_{\mathbf{R}} \det(1+\mathbf{WR}), \quad \mathbf{R}\succeq\mathbf{0}, \operatorname{tr}\mathbf{R}\leq P_T \\ C &= \ln\det(1+\mathbf{WR}^*) \end{aligned} \tag{5.4}$$

In this formulation the function to maximize is determinant and it no longer has a log component. The equivalent CVX code is presented in Fig. 5.3. The results obtained with this solution are (numerically) identical to those that were obtained with the code in Fig. 5.1. Moreover, the running time is shorter. However, this work around is applicable only if the objective function consists solely of a ln det term.

exp, and entropy family are solved using an experimental successive approximation method. This method is slower and less reliable than the method CVX employs for other models. Please see the section of the user's guide entitled The successive approximation method for more details about the approach, and for instructions on how to suppress this warning message in the future."

CVX code	Optimization problem
cvx_begin variable R(m, m) symmetric R == semidefinite(m); F = det(I+W*R); maximize F 0 <= trace(R) <= snr_lin; cvx_end	maximize $F(\mathbf{R}) = \det(\mathbf{I} + \mathbf{WR})$ subject to $\mathbf{R} \succcurlyeq \mathbf{0}$, $\mathrm{tr}\mathbf{R} \geq 0$, $\mathrm{tr}\mathbf{R} \leq P_T$, $\mathbf{R} \in \mathbb{R}^{m \times m}$

Figure 5.3. CVX alternate code for finding the optimal covariance matrix over the MIMO channel ($C = \ln F$).

The CVX Precision Parameter

In CVX, the precision parameter can be specified using a pre-defined value or a numerical value. The meaning of the precision variable depends on the transformations that CVX applies to the model before delivering it to the solver and is linked to the tolerance level used by the solvers [10].

Example 5.2: *CVX precision parameter and timing considerations*

Consider the Tx-Rx channel in the previous example.

$$\mathbf{W} = \frac{1}{m}\mathbf{E}_m, \quad m \in \{2, 16\} \tag{5.5}$$

Table 5.1 shows the optimal channel capacity value and the running time required to obtain it. The analytical value is the one obtained following the water-filling solution presented in Chapter 2. A negative optimal value is interpreted as 0. Choosing a lower value for the precision parameter returns faster results at the expense of accuracy (in this case, when precision is set to 10^{-2}, the optimal value is zero).

There is a trade-off between the accuracy of the optimization results and the running time. The optimization result also depends on the way that the maximization problem is written.

For example:

$$C_1 = \max_{\mathbf{R}\succeq\mathbf{0},\mathrm{tr}\,\mathbf{R}\leq 1} \ln\det(\mathbf{I}+\gamma\mathbf{WR}) \quad \text{and} \tag{5.6}$$

$$C_2 = \max_{\mathbf{R}\succeq\mathbf{0},\mathrm{tr}\,\mathbf{R}\leq P_T} \ln\det(\mathbf{I}+\mathbf{WR}) \tag{5.7}$$

are mathematically equivalent but the numerical solver returns different values for the channel capacity.

Example 5.3: *CVX sensitivity to problem formulation*

Let the CVX precission parameter be set to 10^{-4}. The channel matrix is the same as in the previous example. The channel capacity values returned at SNR = -20dB in the two cases are shown in Table 5.2. The results are slightly different depending on the problem formulation.

The minimum precision level used for other applications of CVX in this thesis is 10^{-4} and the optimization problems are programmed similarly to C_2, i.e. P_T appears explicitly in the constraints and not in the objective function.

Table 5.1. CVX optimization sensitivity to the precision parameters.

SNR = -20dB	$m = 2$		$m = 16$	
precision	C_{CVX}	t [s]	C_{CVX}	t [s]
low	9.897E-3	0.84	9.277E-3	12.49
medium	9.949E-3	1.16	9.937E-3	15.48
default	9.950E-3	1.45	9.950E-3	20.70
high	9.950E-3	2.53	9.050E-3	33.79
best	9.950E-3	2.43	9.950E-3	27.99
10^{-2}	0	0.53	0	9.61
10^{-4}	9.895E-3	0.80	9.204E-3	12.81
10^{-8}	9.950E-3	1.39	9.950E-3	21.14
10^{-16}	9.950E-3	1.72	9.950E-3	26.47
analytical	9.950E-3			

Table 5.2. CVX optimization sensitivity to problem formulation.

SNR = -20 dB	$m = 2$	$m = 16$
C_1	9.8958E-3	9.2480E-3
C_2	9.9850E-3	9.2037E-3

CVX and the Weak Eavesdropper Case

The secrecy capacity expression in (3.5) does not follow the disciplined convex optimization rules (it is concave only when $\mathbf{W}_1 \preceq \mathbf{W}_2$) and CVX cannot be used to solve it directly.

Consider the weak eavesdropper approximation to the secrecy capacity in (4.15) [36]:

$$C_{\text{tr}}(\mathbf{R}) = \ln\det(\mathbf{I} + \mathbf{W}_1\mathbf{R}) - \text{tr}(\mathbf{W}_2\mathbf{R}), \quad \text{tr}\,\mathbf{R} \leq P_T \tag{5.8}$$

(5.8) is concave in $\mathbf{R}$ and CVX accepts it as an objective function. The lines of code in Fig. 5.4 reproduce the CVX part of the algorithm. The value of C_{tr} can be obtained analytically

CVX code	Optimization problem

`cvx_begin`
`  variable R(m, m) symmetric`
`  R == semidefinite(m);`
`  C_tr = log_det(I+W1*R)-trace(W2*R);`
`  maximize C`
`  0 <= trace(R) <= snr_lin;`
`cvx_end`
	maximize $C_{\text{tr}}(\mathbf{R}) = \ln\det(\mathbf{I} + \mathbf{W}_1\mathbf{R}) - \text{tr}(\mathbf{W}_2\mathbf{R})$
	subject to $\mathbf{R} \succcurlyeq \mathbf{0}$,
	$\text{tr}\mathbf{R} \geq 0$,
	$\text{tr}\mathbf{R} \leq P_T$,
	$\mathbf{R} \in \mathbb{R}^{m\times m}$

Figure 5.4. CVX code for the optimization of C_{tr}.

following the results in [36]. The numerical implementation of the algorithm is discussed in Chapter 4. The example below shows the optimization results in one simple case.

Example 5.4: *Optimizing the weak eavesdropper expression with CVX*

The CVX optimization is run for the following case:

$$\mathbf{W}_1 = \begin{bmatrix} 2 & 1 \\ 1 & 1 \end{bmatrix}, \quad \mathbf{W}_2 = \begin{bmatrix} 2 & 0 \\ 0 & 1 \end{bmatrix} \tag{5.9}$$

The optimization results are shown in Fig. 5.5[4]. The optimal values returned by CVX

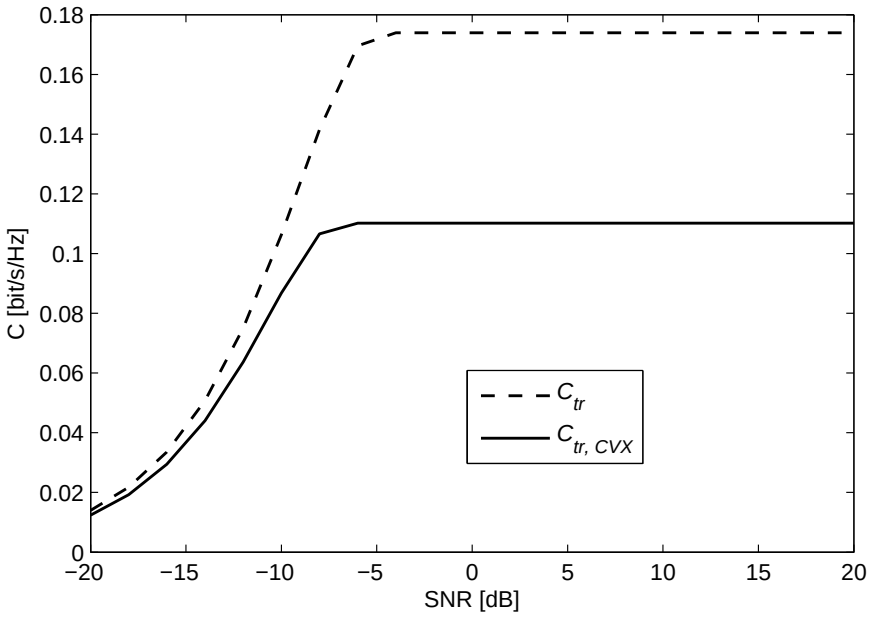

Figure 5.5. CVX optimization for (5.8).

are below the ones obtained with the analytical solution.

This type of result was observed in most of the cases tested. This behavior is discussed in more detail in the next section.

CVX and Optimization Inaccuracies

When optimizing (5.8) for the channel matrices in the previous example, CVX does not print an error message and reports the problem as "solved". However, it issues a warning (see footnote 3 in page 49) about possible inaccuracies in the ln det function. The user manual [10] specifies that, as part of the exponential family, the ln det function is solved using an experimental successive approximation method which is slower and less reliable than other methods used by CVX. This does not fully explain why the optimization result is

[4]The CVX precision parameter was set to "best". The analytical solution curve was obtained following the solution in [36] as explained in Chapter 4.

so far from the true value. The discrepancies observed in the previous example are deemed significant enough to justify further investigation of the problem.

A possible solution was to test the different solvers that CVX encompasses and are available through MATLAB: SDP, SDPT3, SeDuMi and Mosek. As the example below shows, solving the optimization problem with either of these solvers does not change the result.

Example 5.5: *CVX optimization with different solvers*

Consider:

$$\mathbf{W}_1 = \begin{bmatrix} 2 & 1 \\ 1 & 1 \end{bmatrix}, \quad \mathbf{W}_2 = \begin{bmatrix} 2 & 0 \\ 0 & 1 \end{bmatrix} \tag{5.10}$$

The optimization results with the different solvers are shown in Fig. 5.6. There is no significant difference between the four curves presented on this plot.

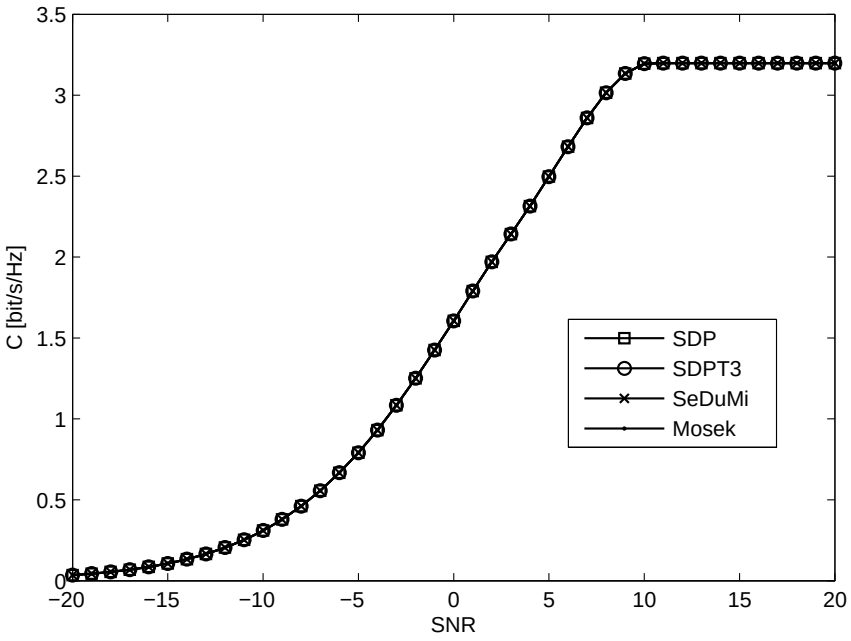

Figure 5.6. CVX optimization with different numerical solvers.

Another possible solution was to reformulate the problem in a way that makes use of expo-

nential functions differently. Two additional formulations are considered:

$$\max(t_1 - \mathrm{tr}(\mathbf{W}_2\mathbf{R})) \qquad \text{s.t.} \quad 0 \leq \mathrm{tr}\,\mathbf{R} \leq P_T, t_1 - \ln\det(\mathbf{I} + \mathbf{W}_1) \leq 0 \tag{5.11}$$

and

$$\max(t_1 - \mathrm{tr}(\mathbf{W}_2\mathbf{R})) \qquad \text{s.t.} \quad 0 \leq \mathrm{tr}\,\mathbf{R} \leq P_T, e^{t_1} - \det(\mathbf{I} + \mathbf{W}_1) \leq 0 \tag{5.12}$$

The first option eliminates $\ln\det$ from the objective function. The second option eliminates the $\ln\det$ from the problem formulation and makes use of determinant and an exponential function. As the example below shows, writing the code for the optimization problem in either one of these three mathematically equivalent forms does not change the CVX result.

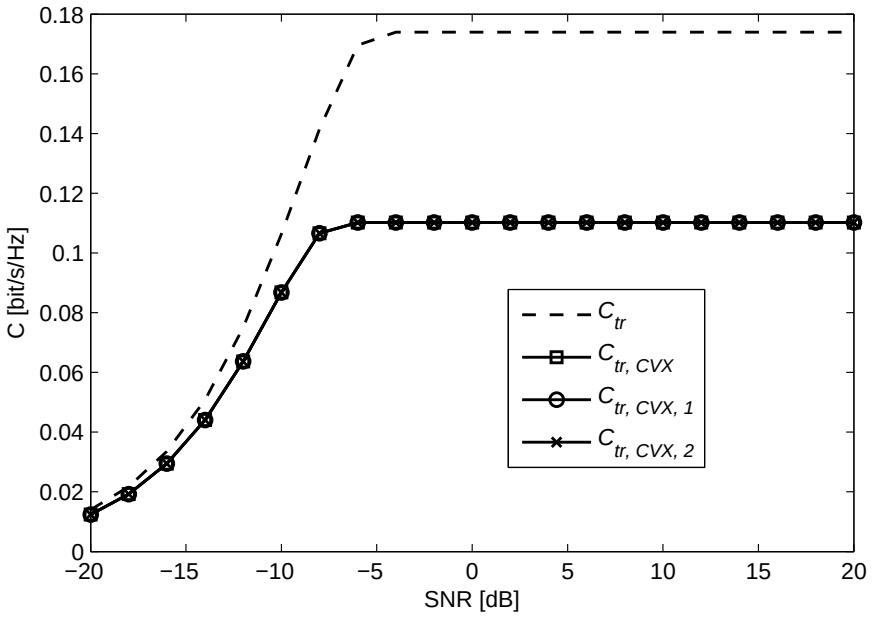

Figure 5.7. CVX optimization with different problem formulation.

Example 5.6: *CVX optimization sensitivity to the problem formulation*

Consider the same channel matrices as in the previous example. The optimization results when different problem formulations are used are plotted in Fig. 5.7. The C_{tr} curve is obtained using the solution in [36]. The $C_{\mathrm{tr},CVX}$ curve is the same as in Fig.

5.5. The $C_{\text{tr},CVX,1.}$ and $C_{\text{tr},CVX,2.}$ correspond to the formulations in (5.11) and (5.12). The three formulations return equivalent results.

The issue is further investigated by looking at the covariance matrix that CVX stores at the end of the optimization. The example below illustrates the characteristics of this matrix.

Example 5.7: *Charecteristics of the optimal covariance matrix as returned by CVX*

Consider:

$$\mathbf{W}_1 = \begin{bmatrix} 2 & 1 \\ 1 & 1 \end{bmatrix}, \quad \mathbf{W}_2 = \mathbf{E}_2 \tag{5.13}$$

where $\mathbf{E}_2$ is a 2×2 all ones matrix. The optimization results are plotted in Fig. 5.8[5]. CVX stores the covariance matrix that generated $C_{\text{tr},CVX}$ as a variable[6]. For special values of SNR, the optimal covariance matrix and its corresponding eigenvectors is presented in table 5.3. $\mathbf{R}^*_{\text{tr},CVX}$ gives $C_{\text{tr},CVX}$ and $\mathbf{R}^*_{\text{tr}}$ is the optimal covariance matrix as obtained using [36]. $\mathbf{U}_{\text{tr},CVX}$ and $\mathbf{U}_{\text{tr}}$ are the corresponding eigenvector matrices.

Table 5.3. Optimal covariance matrix with CVX.

SNR[dB]	$\frac{1}{\gamma}\mathbf{R}^*_{\text{tr},CVX}$	$\mathbf{U}_{\text{tr},CVX}$	$\frac{1}{\gamma}\mathbf{R}^*_{\text{tr}}$	$\mathbf{U}_{\text{tr}}$
-20	$\begin{bmatrix} 0.7236 & 0.4472 \\ 0.4472 & 0.2764 \end{bmatrix}$	$\begin{bmatrix} 0.5257 & -0.8507 \\ -0.8507 & -0.5257 \end{bmatrix}$	$\begin{bmatrix} 0.9996 & -0.0196 \\ -0.0196 & 0.2526 \end{bmatrix}$	$\begin{bmatrix} 0.9998 & 0.0196 \\ -0.0196 & 0.9998 \end{bmatrix}$
0	$\begin{bmatrix} 0.3213 & -0.3574 \\ -0.3574 & 0.6787 \end{bmatrix}$	$\begin{bmatrix} 0.5257 & -0.8507 \\ -0.8507 & -0.5257 \end{bmatrix}$	$\begin{bmatrix} 0.7473 & -0.4345 \\ -0.4345 & 0.2526 \end{bmatrix}$	$\begin{bmatrix} 0.8645 & 0.5026 \\ -0.5026 & 0.8645 \end{bmatrix}$
20	$\begin{bmatrix} 0.0200 & -0.0300 \\ -0.0300 & 0.0500 \end{bmatrix}$	$\begin{bmatrix} 0.5257 & -0.8507 \\ -0.8507 & -0.5257 \end{bmatrix}$	$\begin{bmatrix} 0.5045 & -0.4995 \\ -0.4995 & 0.4996 \end{bmatrix}$	$\begin{bmatrix} 0.7106 & 0.7036 \\ -0.7036 & 0.7106 \end{bmatrix}$

The analytical solution recognizes that $\mathbf{W}_2$ is rank 1 and signals in the direction corresponding to the null of the eavesdropper channel matrix. CVX does not. Note that

[5]The C_{tr} curve is the maximal value of (5.8) obtained with the analytical solution. The $C_{\text{tr},CVX}$ curve shows the CVX optimization results

[6]Using $\mathbf{R}^*$ as returned by CVX gives a slightly higher optimum value than what is stored in $\texttt{cvx}_{optval}$.

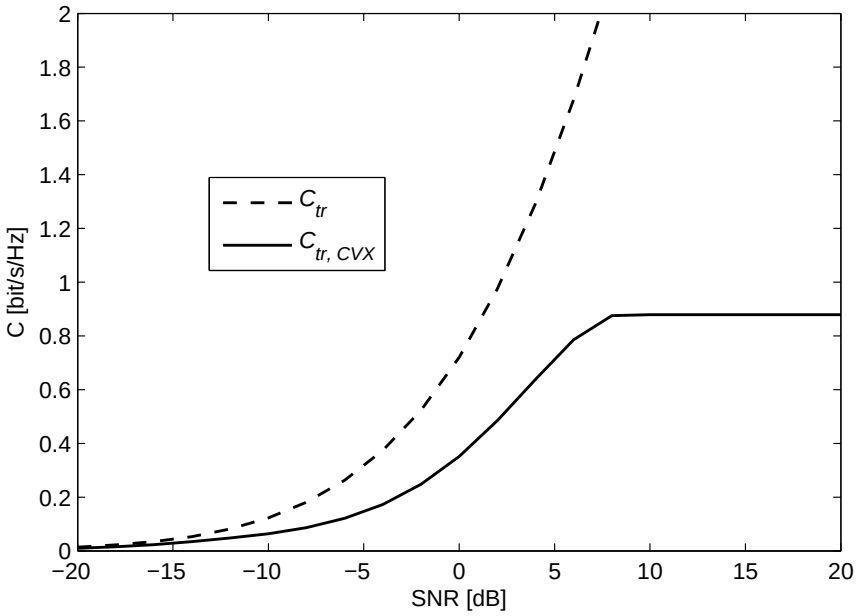

Figure 5.8. CVX optimization for (5.8).

the eigenvectors of $\mathbf{R}^*_{\mathrm{tr},CVX}$ are always the same as those of $\mathbf{W}_1$:

$$\mathbf{U}_1 = \begin{bmatrix} 0.5257 & -0.8507 \\ -0.8507 & -0.5257 \end{bmatrix} \tag{5.14}$$

Fig. 5.9 shows the eigenvalues of the optimal covariance matrix and the eigenvalues normalized to the total available transmit power, i.e. the eigenvalues of $\frac{1}{\gamma}\mathbf{R}^*_{\mathrm{tr}}$ and $\frac{1}{\gamma}\mathbf{R}^*_{\mathrm{tr},CVX}$. A possible explanation for the CVX solution is that it uses the same communication directions as if the eavesdropper was not present and then adjusts the transmit power to minimize the leakage toward the eavesdropper, i.e. as if the optimization is done for the $\ln\det(\mathbf{I}+\mathbf{W}_1\mathbf{R})$ term first and then the optimal covariance matrix is scaled to find the maximum of the expression

CVX performs the optimization as long as the problem is convex[7] and respects the disciplined convex programming rules. The implementation of this software struggles with functions in

[7]The objective function in concave in a maximization problem and the constraints are convex or linear [23].

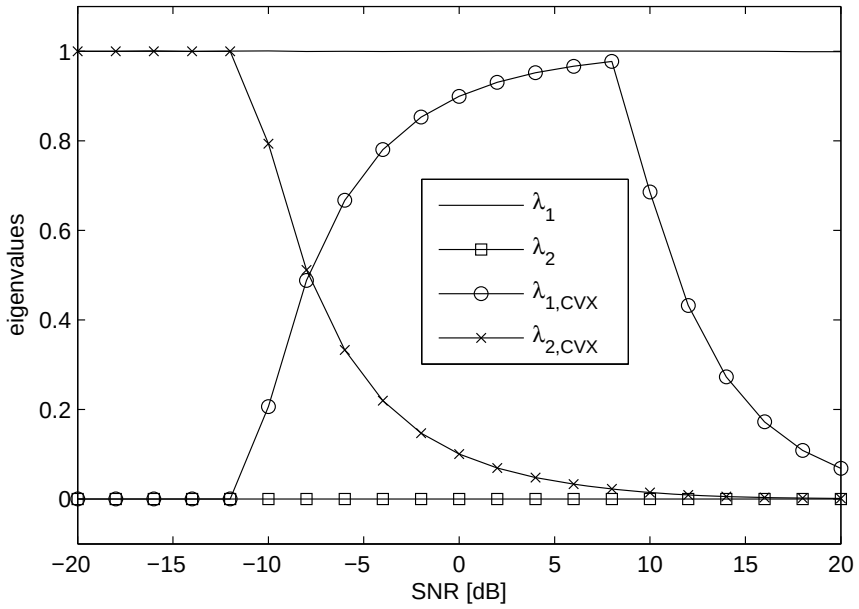

Figure 5.9. Eigenvalues of optimal covariance matrix.

the exponential family such as $\ln\det$. Because of this, the results are unreliable even when a convex function such as the weak eavesdropper approximation to the secrecy capacity is set as an objective function.

5.2 YALMIP

YALMIP is a modeling language used for optimizing both convex and non-convex problems. It is implemented as a free MATLAB toolbox [48]. Reference [49] is a basic beginner's guide. YALMIP also encompases several solvers both free and commercial—a complete list is provided in [48]. YALMIP can return the minimum value of an expression. To obtain the maximum value of an expression one needs to multiply the expression by -1, find the minimum and then multiply the result by -1.

Optimizing with YALMIP

When its input is a convex problem,YALMIP searches for the global minumum (in a minimization problem). A possible implementation of a YALMIP program optimizing the secrecy capacity value in (3.4) is shown in Fig. 5.10. This optimization gave satisfactory results in a few cases. However, as illustrated in the following example, the accuracy of these results is not consistent for different channel matrices.

YALMIP code	Optimization problem

`R = sdpvar(m, m,'hermitian', 'real')`
`F = set(sum(diag(R))<=snr_lin);`
`F = F + set(sum(diag(R))>=0);`
`F = F + set(R>=0);`
`objective = -log(det(I+W1*R))…`
`+log(det(I+W2*R));`
`solution = solvesdp(F, objective);`
`R_star = double(R);`
`C_s = log(det(eye(m)+W1*R_star))…`
`-log(det(eye(m)+W2*R_star));`
	maximize $C_s(\mathbf{R}) = \ln\det(\mathbf{I}+\mathbf{W}_1\mathbf{R}) - \ln\det(\mathbf{I}+\mathbf{W}_2\mathbf{R})$
	subject to $\mathbf{R} \succcurlyeq \mathbf{0}$,
	$\mathrm{tr}\mathbf{R} \geq 0$,
	$\mathrm{tr}\mathbf{R} \leq P_T$,
	$\mathbf{R} \in \mathbb{R}^{m\times m}$

Figure 5.10. YALMIP code for the optimization of C_s.

Example 5.8: *Secrecy capacity evaluation with YALMIP*

Consider:

$$\mathbf{W}_1 = \frac{1}{2}\mathbf{E}_2, \quad \mathbf{W}_2 = 0.1\mathbf{W}_1 \tag{5.15}$$

When defining the positive semi-definite constraint explicitly, the program in Fig. 5.10 is not able to solve the problem. An equivalent condition is used requiring that all the eigenvalues and determinants of the covariance matrix be positive. This condition is shown in Fig. 5.11.

```
F = set(R>=0);   →   F = set(det(K)>=0);
                     F = F + set(eig(K)>=0);
```

Figure 5.11. YALMIP equivalent semidefinite constraint.

The results returned by YALMIP are compared with the known analytical solution in

Fig. 5.12[8].

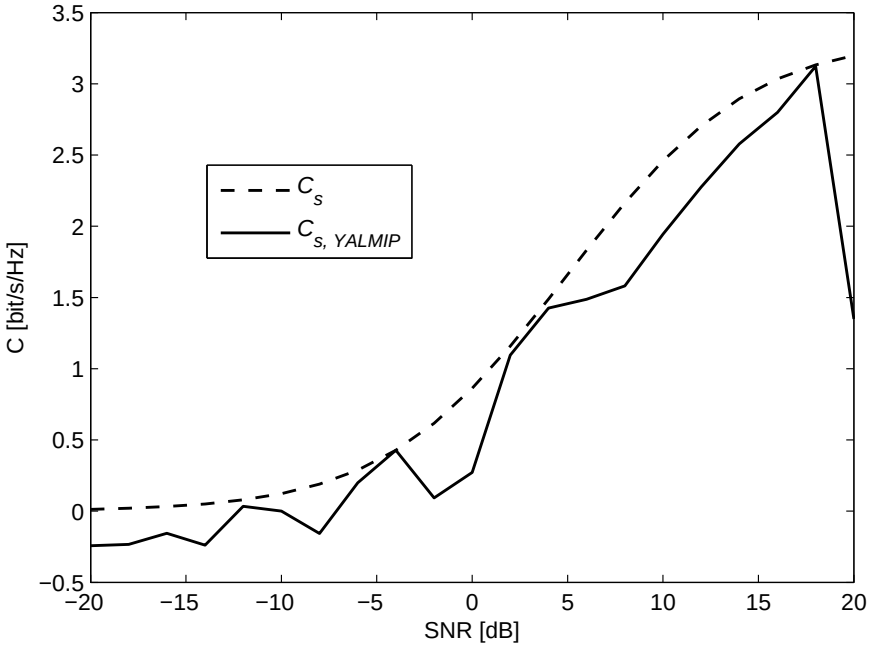

Figure 5.12. YALMIP optimization.

The optimal secrecy capacity value that YALMIP outputs oscillate significantly with respect to SNR.

YALMIP returns one of three warnings. In the first case, a minimum is found and the constraints are satisfied. Since the objective function expression is convex, the result is equal to the global minimum. In the second case, the warning indicates that the solver was stopped prematurely because the maximum number of iterations was achieved. Finally, in the third case, no feasible solution is found even when the step size is less than the default value of the step tolerance.

For similar problems (same $\mathbf{W}_1, \mathbf{W}_2$, similar SNR values), the optimization results are quite different. Because of results such as this, optimization of C_s with YALMIP is deemed inaccurate.

[8]The complete MATLAB code written for the optimization with YALMIP is given in Appendix 9.

5.3 Summary

Numerical optimization tools are relatively easy to use and can return an approximate optimal value for a function without requiring mathematical optimization background. Two popular optimization software tools were considered in this section: CVX and YALMIP. Both of them can be used as a MATLAB toolbox and they offer a unified way to perform optimization and to use several numerical solvers.

In CVX, the secrecy capacity optimization problem cannot be used directly since it does not follow the disciplined convex programming rules. Furthermore, when attempting to optimize the weak eavesdropper approximation, the results obtained by CVX are significantly lower than the optimal value that was obtained with other methods. This is possibly due to optimization inaccuracies that appear in an expression having a $\ln\det$ term and other terms. Nonetheless, CVX is powerful and simple to use. Note that with proper precautions, CVX can be used to solve optimization problems. This is done in Chapter 7.

YALMIP optimizes convex and non-convex problems alike. When the problem solved is convex, YALMIP returns the global minimum. Otherwise, a local minimum[9] is returned on a best-effort basis. In some of the cases tested, the results obtained with YALMIP were below the expected values. Moreover, the accuracy of the results obtained varied considerably for similar types of problems. For this reason, YALMIP is not used any further.

[9]Note that the objective function, when optimizing with YALMIP, was the negative of the secrecy capacity expression.

Chapter 6

Monte-Carlo Methods and Optimization

Due to the non-convexity of the secrecy capacity expression, it is difficult to find an optimal transmit covariance matrix either analytically or numerically using readily-available optimization tools. The stochastic optimization methods presented in this chapter optimize the maximal value of $C_s(\mathbf{R})$ by performing a random search over all possible transmit covariance matrices. The algorithms used are iterative, the result at each step is closer to $\mathbf{R}^*$ than the results at the previous steps. The first algorithm is a random blind search, a direct Monte-Carlo Optimization. The second one uses the covariance matrix rank as an optimization parameter. The third method is a variant of the genetic algorithm presented in [50] and adapted to the secrecy capacity optimization.

Since the methods presented in this chapter include randomness, simulations produce results that change at each run. To capture the algorithm behavior on average, the simulation is run multiple times and the results at each trail number are averaged to eliminate variations. Consider the following variables:

- x: the quantity to optimize.
- n: the trail number. n indicates how many trials were done up to the current one. The total number of trials considered is N; $n = 1, \ldots, N$.
- K is the total number of simulation runs; $k = 1, \ldots, K$. Each run includes N trials.
- $x_k(n)$ is the result after the n^{th} trial in the k^{th} run.

For example, if 5 trials are done and the algorithm is run twice ($N = 5$ and $K = 2$), the output of a maximization problem can be written as:

$$x_1(1) \leq x_1(2) \leq x_1(3) \leq x_1(4) \leq x_1(5)$$
$$x_2(1) \leq x_2(2) \leq x_2(3) \leq x_2(4) \leq x_2(5)$$

After K runs of the algorithm, the value of x at each iteration is estimated to:

$$x(n, K) = \frac{1}{K} \sum_{k=1}^{K} x_k(n) \tag{6.1}$$

On average, this empirical mean is the result that the Monte-Carlo optimization returns after n trials. Another quantity of interest is the standard deviation of the empirical mean:

$$\sigma_x(n, K) = \sqrt{\frac{1}{K-1} \sum_{k=1}^{K} \left(x(n, K) - x_k(n)\right)^2} \tag{6.2}$$

$\sigma_x(n, K)$ is a measure of the variability in the results. In this chapter, the Monte-Carlo optimization results presented are averaged over 100 runs of the algorithm, i.e. $K = 100$.

6.1 Random Blind Search

The simplest way to (randomly) search for the optimal $C_s(\mathbf{R})$ value is to repeatedly sample over all possible transmit covariance matrices and retaining the one that returns the best

result. The quality of the results increases with the number of repetitions N. If N is large enough, the algorithm converges to the optimal value[1]. However, the rate of convergence decreases significantly as N increases [51].

A flowchart detailing the random blind search algorithm for the secrecy capacity is presented in Fig. 6.1.

Any test covariance $\mathbf{R}$ needs to satisfy the following two conditions:

$$\mathbf{R} \succeq \mathbf{0}, \quad \operatorname{tr} \mathbf{R} = P_T \tag{6.3}$$

The first condition comes from the fact that $\mathbf{R}$ is a covariance matrix. The second one ensures that all available power at the transmitter is used—since for proportional $\mathbf{R}$, the largest $C_s(\mathbf{R})$ corresponds to the matrix having the largest trace, covariance matrices using less power are sub-optimal. The test matrices are built making use of MATLAB's `randn` function[2] as follows:

$$\mathbf{A} = \texttt{randn}(m, m) \rightarrow \mathbf{B} = \mathbf{A}\mathbf{A}^T \rightarrow \mathbf{R} = \frac{\mathbf{B}}{\operatorname{tr} \mathbf{B}} \cdot P_T \tag{6.4}$$

First, matrix $\mathbf{A}$, having randomly-generated independent elements is created; $\mathbf{A}$ is not a covariance matrix. Matrix $\mathbf{B} = \mathbf{A}\mathbf{A}^T$ is positive-semidefinite. Covariance matrices constructed in this fashion are (almost surely) full rank. To obtain the test matrix $\mathbf{R}$, $\mathbf{B}$ is normalized and scaled such that $\operatorname{tr} \mathbf{R} = P_T$.

For a system having m transmit antennas, there are $\frac{1}{2}m(m+1)$ variables to optimize: m of them specify the power allocation and the rest of them the covariance between the transmit antenna signals. The dimensionality of the optimization problem increases as m^2. The volume

[1] The proof of convergence for the random blind search used in a minimization algorithm is provided in [51]. The proof can be adapted to a maximization algorithm by inverting the sign of the objective function.

[2] This function generates random numbers that are normally distributed with mean 0 and variance 1. In this thesis, the values generated by this pseudo-random number generator are assumed to be truly random and independent from each other.

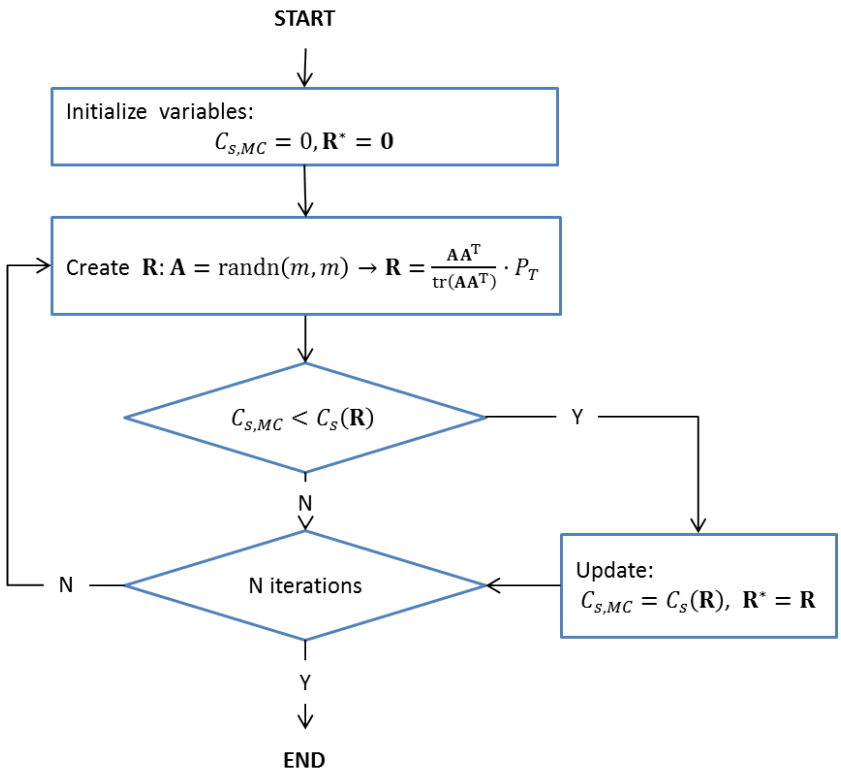

Figure 6.1. Random Blind Search algorithm.

of the search region increases exponentially with the dimensionality [51]. For larger systems it is increasingly difficult to find the optimal covariance matrix and, without exploiting other knowledge about the possible solution, an optimal result is hard to attain.

Example 6.1: *Secrecy capacity—random blind search*

Consider the following channel matrices:

$$\mathbf{W}_1 = \frac{1}{m}\mathbf{E}_m, \quad \mathbf{W}_2 = 0.1\mathbf{W}_1 \quad m \in \{2, 4, 8, 16\} \tag{6.5}$$

where $\mathbf{E}_m$ is an $m \times m$ all ones matrix. Since $\mathbf{W}_1 - \mathbf{W}_2$ has only one (positive) eigenvalue, the optimal covariance matrix is known [35]:

$$\mathbf{R}^* = \frac{1}{m}\mathbf{E}_m \tag{6.6}$$

The secrecy capacity is:

$$C_s = \ln\det(\mathbf{I} + \mathbf{W}_1\mathbf{R}^*) - \ln\det(\mathbf{I} + \mathbf{W}_2\mathbf{R}^*) = \ln\frac{1+\gamma}{1+0.1\gamma} \tag{6.7}$$

because of the way that the channel matrices are chosen, the value of C_s is independent of the number of transmit antennas. This allows us to compare optimization results for systems of different sizes. The output of the Monte-Carlo algorithm is illustrated in Fig.

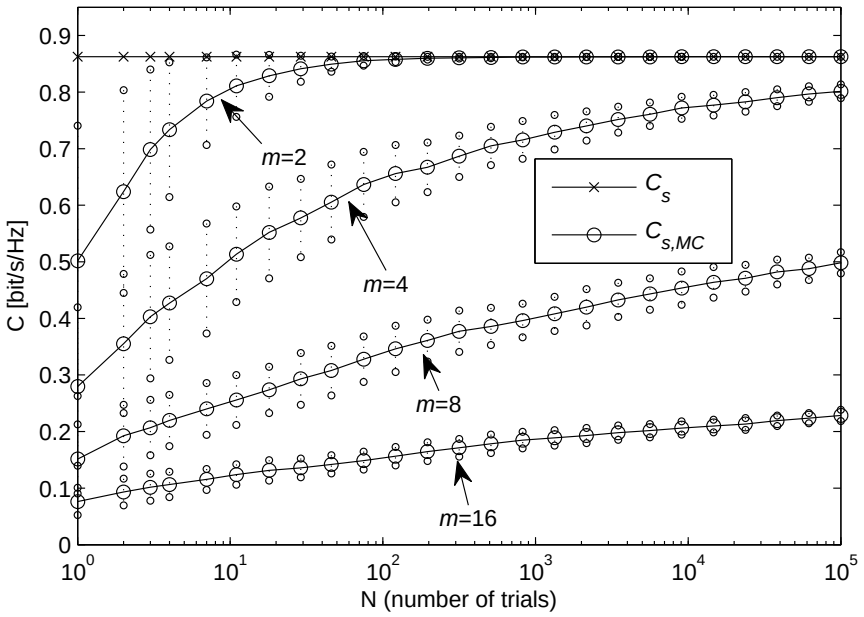

Figure 6.2. Random Blind Search optimization, SNR = 0dB.

6.2. As the number of trials N increases, the convergence rate decreases. This is more visible for large systems ($m = 16$). The fact that a reasonable value for $C_{s,MC}$ is more difficult to find for larger systems can also be explained from a physical point of view. According to the analytical solution $\operatorname{rank}\mathbf{R}^* = 1$. Hence, the optimal transmit strategy is beamforming. In this case, finding the optimal transmit covariance is equivalent to finding the optimal beamforming direction. For larger values of m, the beam can be made narrower. Therefore, more of the transmit power can be directed to the receiver, as shown in Fig. 6.3. A narrower beam needs to be directed more precisely toward

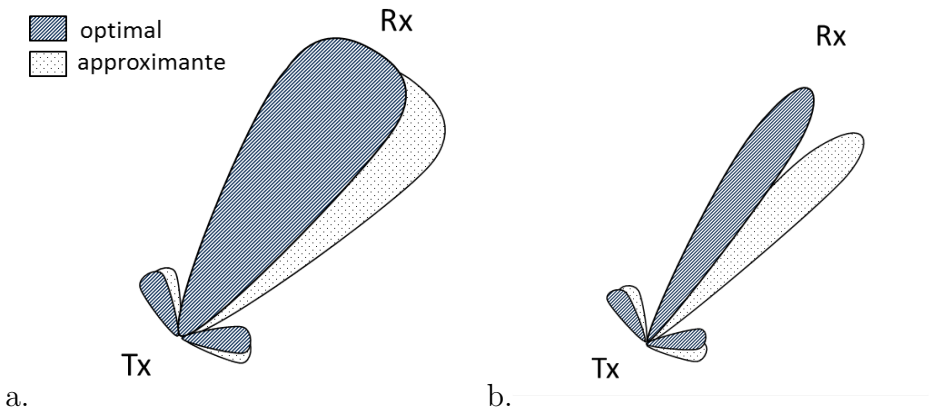

Figure 6.3. Beamwidth: a. small m; b. large m.

the receiver and the beamforming direction needs to be detected with more accuracy. When the beam is wide, some of it points toward the receiver even though the direction might not be as precise.

Fig. 6.2 shows that there is a larger variation in results for the case where $m = 2$. For smaller systems the dimensionality of the search area is smaller and the chances of getting high values for $C_{s,MC}$ within the first few trials are larger. The values of $C_{s,MC}$ vary more for these smaller systems.

The variation in the results is shown by plotting some points at a standard deviation away from the mean value. This is done in order to illustrate the variation in the results reached after a given number of trials. Note that the distribution of values for $C_{s,MC}$ is not uniform between 0 and C_s. In fact, if $C_s(\mathbf{R}) \leq 0$, it is interpreted as 0. Moreover, there are fewer $\mathbf{R}$ matrices that give a secrecy rate close to C_s than there are covariance matrices that do not approach the optimum.

The blind search algorithm generates a new sample value at each iteration without taking into account where the previous samples have been. This makes it simple to implement but very inefficient for larger values of m. The random search algorithm can also incorporate known characteristic of the objective function or of the optimal covariance matrix. This is done in Section 6.2.

Random Blind search vs CVX

For small systems, the random blind search algorithm can return the secrecy rate values close to C_s within the first few trials. In Fig. 6.2, for example, for $m = 2$, after approximately 100 trials the value of $C_{s,MC}$ is higher that 95% of C_s. For the purpose of evaluating the random blind search with respect to other numerical optimization methods its performance is compared with that of CVX. Since CVX only accepts objective functions that follow the disciplined convex optimization rules and because CVX does not return reliable results in the weak eavesdropper case (see Chapter 5) this comparison is done for a system where $\mathbf{W}_2 = \mathbf{0}$, i.e. there is no eavesdropper.

Example 6.2: *Secrecy capacity—random blind search vs CVX*

Consider the following channel matrix:

$$\mathbf{W} = \text{diag}\{1, 2, \ldots, m\}, \quad m \in \{2, 4\} \tag{6.8}$$

The function to optimize is the channel capacity:

$$\max \ln \det(\mathbf{I} + \mathbf{WR}), \quad \text{s.t. } \mathbf{R} \succeq \mathbf{0}, \text{tr}\,\mathbf{R} \leq P_T \tag{6.9}$$

Table 6.1 shows the computational time and number of iterations that the random blind search requires to reach 95% of the channel capacity value. C_{CVX} is the channel capacity returned by CVX, C_{MC} is the channel capacity returned by the random blind search; t_{CVX} and t_{MC} are the corresponding computational times. In this example, C_{CVX} is equal to the channel capacity, within the numerical precision that CVX can provide. This table shows that when m is small, the random blind search can return satisfactory values for the channel capacity with less computational effort than CVX.

CVX limits itself to convex problems. Furthermore, CVX only accepts convex problems

Table 6.1. Optimization results and computational time for MC and CVX.

SNR [dB]	m	C_{CVX}	C_{MC}	t_{CVX}	t_{MC} [s]	N
-20	2	0.0198	0.0193	1.4545	0.0002	9
	4	0.0392	0.0378	1.4321	1.1894	54,853
0	2	1.1394	1.1064	1.1748	0.0001	4
	4	2.0841	2.0002	1.2566	0.0067	311
20	2	8.5470	8.3494	1.0987	0.0001	6
	4	16.1360	15.4888	1.2105	0.0029	133

that respect the disciplined convex programming rules. The secrecy capacity optimization problem is not accepted by CVX. Unlike CVX, Monte-Carlo methods are universal in nature. No constraint is imposed on the structure of the problem e.g. convexity and Monte-Carlo methods can be used find the secrecy capacity.

6.2 Rank-adaptive Monte-Carlo

The set of covariance matrices is constrained following [35]:

$$1 \leq \mathrm{r}_+(\mathbf{R}^*) \leq \mathrm{r}_+(\mathbf{W}_1 - \mathbf{W}_2) \tag{6.10}$$

where $\mathrm{r}_+(\mathbf{A})$ is the number of positive eigen-values of $\mathbf{A}$. Since the exact rank value varies with SNR and is not known a-priori, all possible values in this range need to be tested. One way to do this is shown in Fig. 6.4. The rank of the sample matrix changes from one iteration to the next and no one value is preferred over the others. Covariance matrices are generated as follows:

$$\text{select } r = 1...r_{max} \rightarrow \mathbf{A} = \texttt{randn}(m, r) \rightarrow \mathbf{R} = \frac{\mathbf{A}\mathbf{A}^T}{\mathrm{tr}(\mathbf{A}\mathbf{A}^T)} \cdot P_T \tag{6.11}$$

This approach adjusts the covariance matrix according to its rank and hence it is called the Rank-adaptive Monte-Carlo (RaMC) algorithm. Note that RaMC makes use of less random numbers when generating the test covariance matrices than the random blind search. Because

of this, the computational time that it requires to reach the optimal covariance matrix is, on average, less than the time required by the random blind search.

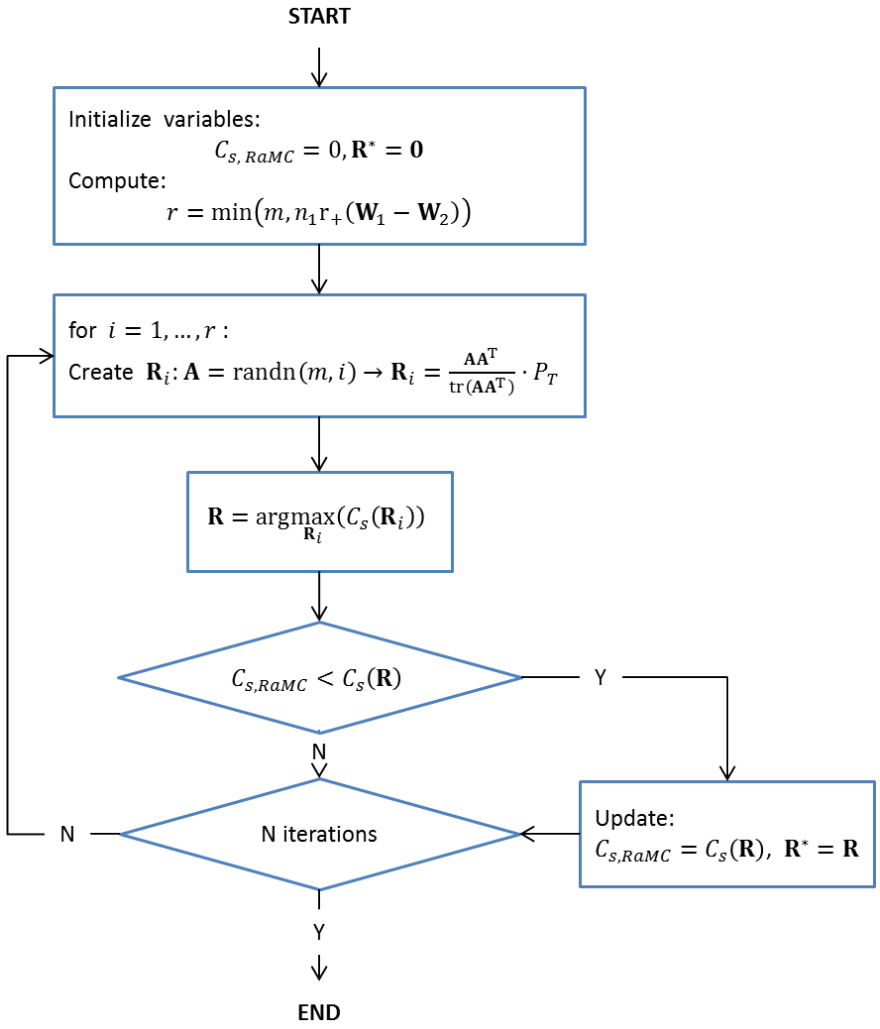

Figure 6.4. Rank-adaptive Monte-Carlo algorithm.

Example 6.3: *Rank-adaptive Monte-Carlo and SNR*

This example shows how the choice of rank for $\mathbf{R}^*$ can impact the accuracy and convergence speed of the Monte-Carlo estimation. Consider:

$$\mathbf{W}_1 = \begin{bmatrix} 2 & 0 \\ 0 & 1 \end{bmatrix}, \quad \mathbf{W}_2 = \frac{1}{10} \begin{bmatrix} 1 & 0 \\ 0 & 2 \end{bmatrix} \tag{6.12}$$

Fig. 6.5 and 6.6 show the simulation results when the rank of the sample matrix is 1 or 2 (curves $C_{s,MC}(\text{rank} = 1)$ and $C_{s,MC}(\text{rank} = 2)$ respectively), and when the rank-adaptive algorithm is used $C_{s,RaMC}$.

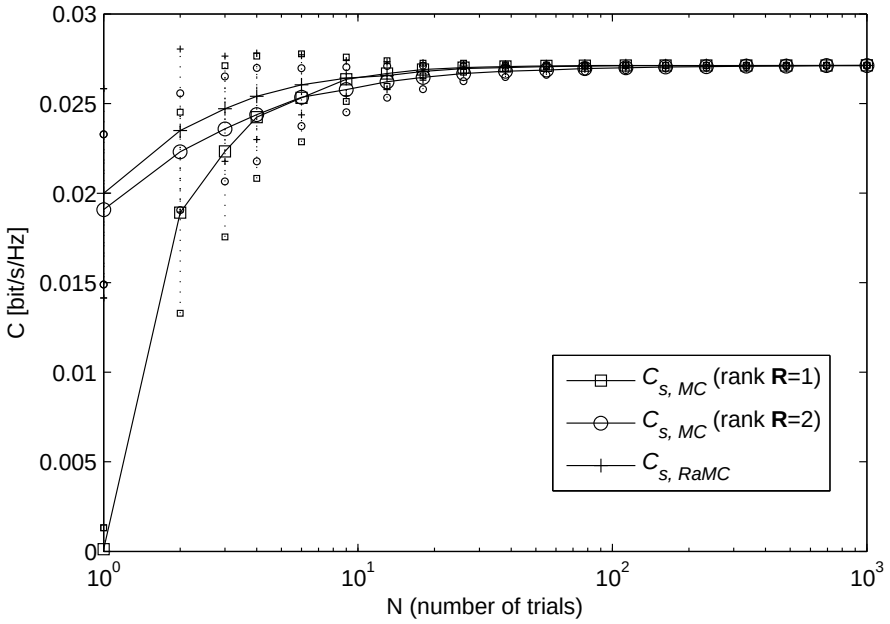

Figure 6.5. RaMC, low SNR.

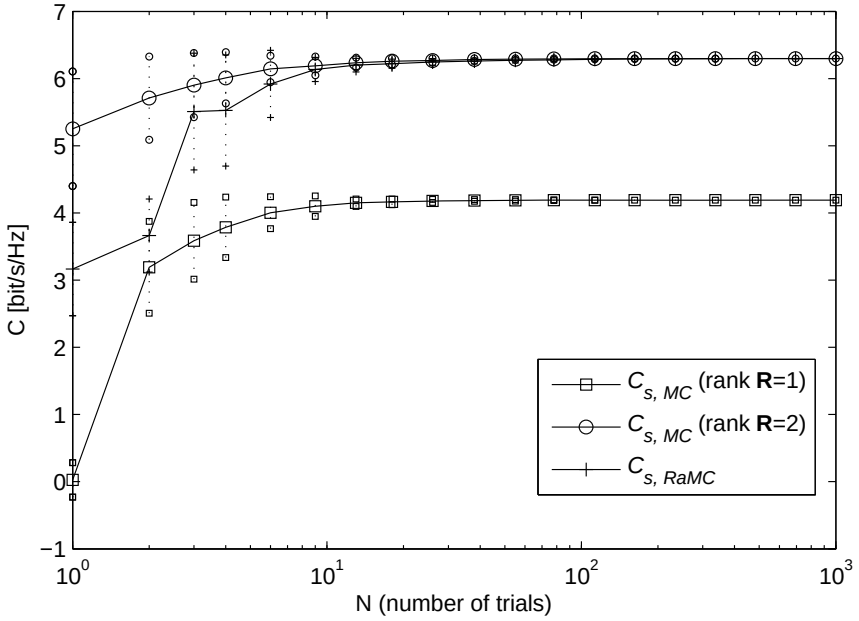

Figure 6.6. RaMC, high SNR.

At SNR = -20dB, the rank-adaptive algorithm is faster than an algorithm where the test covariance matrices would be rank 2 and almost as good as when the test matrices are rank 1 only. At SNR = 20dB, using a lower rank is sub-optimal and the rank-adaptive algorithm result is almost as good as when only full rank matrices are tested. The observations made in this case hold true for RaMC, in general.

At low SNR, choosing the appropriate rank speeds convergence. When $\mathbf{R}^*$ is found, strong communication directions receive all the available power and the weak ones are ignored (at very low SNR the optimal covariance matrix rank is 1 [44]). When the selected rank is higher than needed, there is at least one communication direction that has power assigned to it but it does not contribute to the secrecy rate. Since C_s is monotonous in P_T, having a higher rank for $\mathbf{R}^*$ is sub-optimal. Since the secrecy capacity can saturate at high SNR and power can be re-assigned to channels where this has not happened yet.

At high SNR, not assigning power to strong communication directions (by choosing a lower covariance matrix rank) causes significant losses in the secrecy capacity.

An (almost) optimal covariance matrix is found even if the search is done on matrices having a higher rank: in this case the solution would be a matrix whose eigenvalues are close to zero in the directions that do not increase the secrecy capacity and are (approximately) lower rank. However, this approach requires a longer search than if the rank had been selected correctly. On one hand, the search is done on a much larger set. On the other hand, generating higher rank random matrices makes use of more random variables.

Example 6.4: *Rank-adaptive Monte-Carlo and system size*

Consider:

$$\mathbf{W}_1 = \frac{1}{m}\mathbf{E}_m, \quad \mathbf{W}_2 = 0.1\mathbf{W}_1, \quad m \in \{2, 4, 8, 16\} \tag{6.13}$$

and let SNR = 0dB. Fig. 6.7 and 6.8 show the results of the Rank-adaptive Monte-Carlo algorithm and compare them with the ones returned by the direct Monte-Carlo

optimization. The analytical solution is also plotted for reference (see (6.7)).

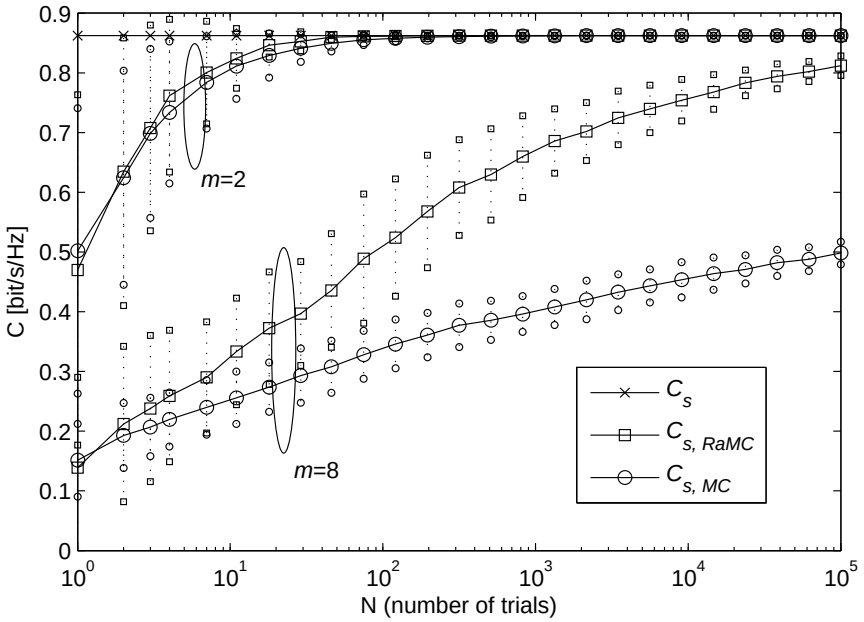

Figure 6.7. RaMC, $m \in \{2, 8\}$.

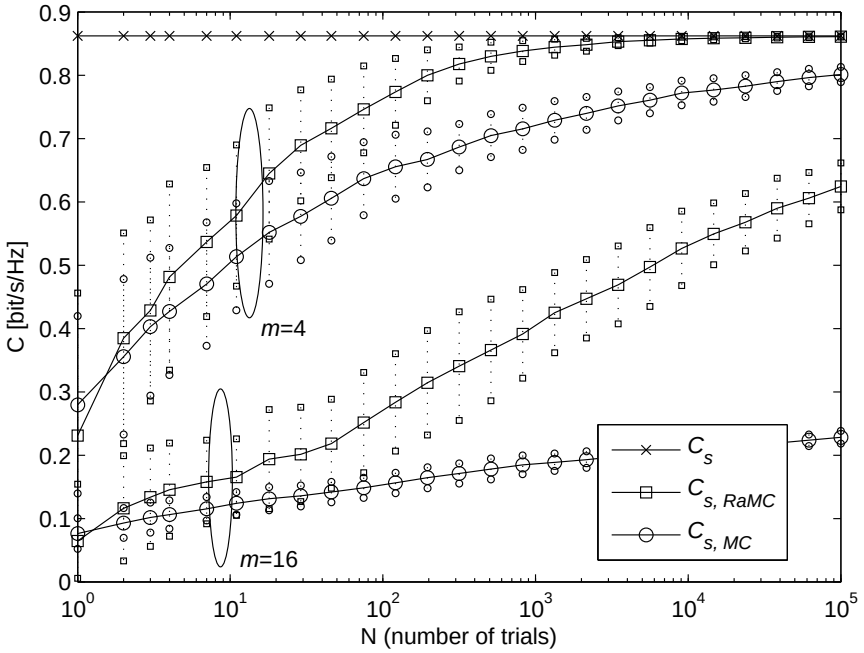

Figure 6.8. RaMC, $m \in \{4, 16\}$.

In this case, RaMC can recognize that $\operatorname{rank} \mathbf{R} = 1$ (see (6.10)) and has only to find

the right communication direction. For m transmit antennas optimization is done for $m+1$ independent variables only (direction angle and power allocation).

Beamforming in the right direction is easier for smaller systems, in this case the beam width is larger. Inaccuracies in the communication direction coordinates have a smaller effect on the signal gain at the receiver.

Choosing suitable values for the rank of the covariance matrix reduces the number of parameters that need to be optimized. This is important for larger systems where finding the optimal covariance matrix using a Monte-Carlo simulation requires more test matrices.

The Rank-adaptive Monte-Carlo algorithm returns a suitable transmit covariance matrix after several trials. The performance of this algorithm serves as a comparison point for the other solutions presented in this document.

6.3 Differential Evolution

Differential evolution (DE) is a very popular[3] stochastic optimization approach. It is an iterative process in which a population of test variables is directed toward the optimal argument using algorithms that emulates the natural evolution process: *mutation*, *crossover* and *selection*. Since it does not require the underlying problem to be convex nor continuous, DE adapts well to the covariance optimization problem. The algorithm is well explained in Storn's popular article [50] and is briefly reviewed here. The vectors forming the original population are interpreted as covariance matrices as follows:

$$\begin{aligned} x_{k,1} &= \texttt{randn}(m^2, 1) \\ &\rightarrow \mathbf{A} = [[x_k(1), \dots x_k(m)]^T, \dots [x_k(m^2 - m + 1), \dots x_m(m^2)]^T] \\ &\rightarrow \mathbf{R} = \frac{\mathbf{A}\mathbf{A}^T}{\operatorname{tr} \mathbf{A}\mathbf{A}^T} \cdot P_T \end{aligned} \tag{6.14}$$

[3]Reference [50] has been cited over 8174 times (as counted by Google Scholar on Oct. 23rd, 2014).

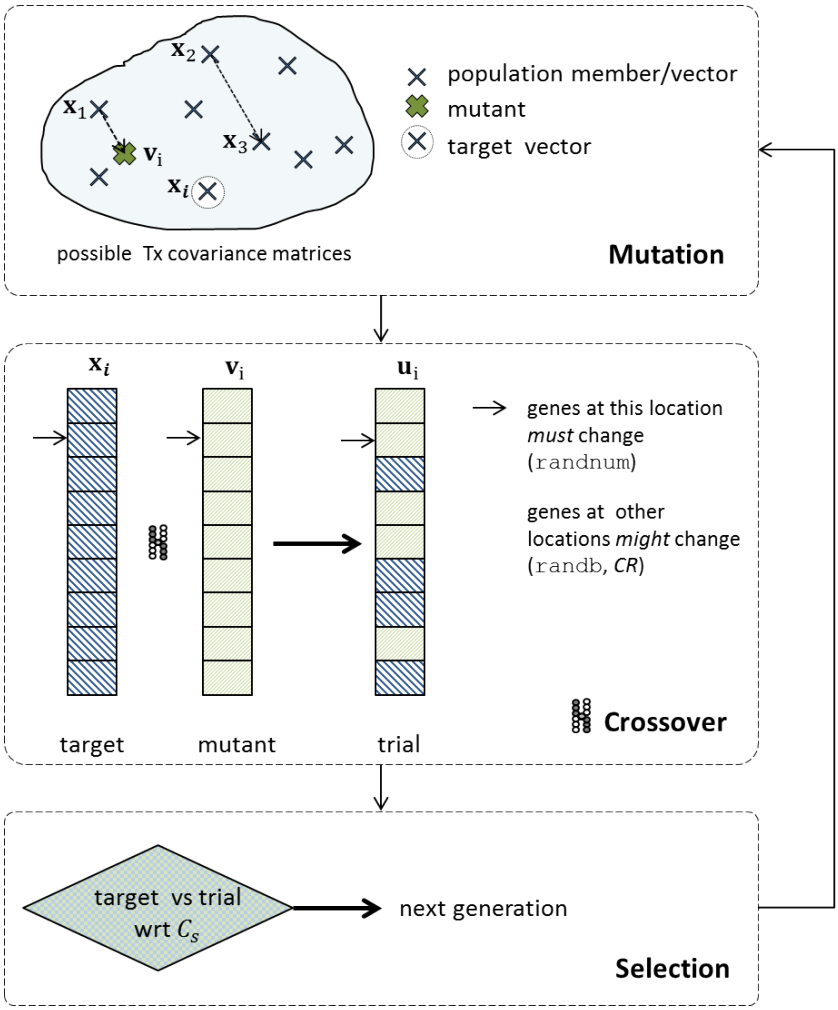

Figure 6.9. DE algorithm overview.

At every iteration of the algorithm (called *generation*), population members undergo three operations (Fig. 6.9 serves as a visual aid):

1. *Mutation*: a new parameter vector, a mutant vector, is created by adding the weighted difference of two population members to a third one.

$$v_{i,g+1} = x_{r_1,g} + F(x_{r_2,g} - x_{r_3,g}) \tag{6.15}$$

i, r_1, r_2, r_3 are all different from one another and r_1, r_2, r_3 are chosen at random. $F \in$

$[0, 2]$ is the maximal evolution distance allowed between generations.

2. *Crossover*: A vector is selected from the current generation termed *target* vector and some of its elements are swapped with those of the parameter vector creating a *trial* vector as follows:

$$index = \texttt{randnum}(m^2), \text{ for each } j : sel(j) = \texttt{randb}$$
$$u_{i,g+1}(j) = \begin{cases} v_{i,g+1}(j), & \text{if } sel(j) \leq CR \text{ or } j = index \\ x_{i,g}(j), & \text{if } sel(j) > CR \text{ and } j \neq index \end{cases} \tag{6.16}$$

`randnum` chooses an index at random between 1 and m^2; `randb` assigns a value between 0 and 1, at random, to $sel(j)$. CR determines the number of genes allowed to change between the new and the existing population members. The second condition ensures that each trial vector keeps at least one of the target vector characteristics. Every population member is chosen as target during one iteration.

3. *Selection*: the performance of trial vector $u_{i,g+1}$ is compared to that of $x_{i,g}$ and the one that generates the larger secrecy rate is kept for the next generation.

The algorithm is implemented in MATLAB and the code is provided in the Appendix.

The behavior of DE and the accuracy of its results is determined both by the algorithm parameters and by the input channel matrices. Reference [50] and the accompanying web page [52] give an indication about what these optimal values are in the general case:

- F—mutation parameter, distance: If the objective function is noisy, the value of F that improves convergence is $F \in [0.5, 1]$.
- CR—crossover parameter, gene swapping: A low value of CR encourages search along the coordinate axes and helps to optimize separable functions. If there is a dependence between the parameters, such as is the case for transmit antennas signals that are correlated, $CR = 0.9$ is more suitable.

- *NP*—population size: The article indicates that a value of *NP* should be between $5 \times D$ and $10 \times D$ where D is the number of variables to optimize. The associated webpage indicates that a *NP* value higher than 40 does not seem to improve convergence, irrespective of the number of parameters to be optimized.

When optimizing the secrecy capacity expression, the following observations were made:

- *NP*: Convergence is faster when the population size is small. In this case there are less matrices to test at each generation and the population can move toward an optimal point faster. For example 100 tests allow each member to evolve once when *NP*=100 while if *NP*=20 each member can evolve 5 times with the same computational effort. However, if the original population is too small then not enough test cases are covered, there is not enough room to evolve and the results converge to a value that is not optimal.

- F: Larger values of F result in a larger variation/distance within the population members. This means that a larger part of the solution pool is covered within fewer trials. In general, when F is large, results are obtained quicker.

- *CR* : A large value of *CR* encourages crossover and creates diverse population members. Therefore, convergence is faster.

The best choice for the algorithm parameters depends on the channel matrices. With the exception of *NP*, no direct link with the channel parameters was observed.

Example 6.5: *Algorithm parameters in DE (F, CR)*

Consider:

$$\mathbf{W}_1 = \frac{1}{m}\mathbf{E}_m, \quad \mathbf{W}_2 = 0.1\mathbf{W}_1, \quad m \in \{4, 8\} \tag{6.17}$$

and let SNR = 0dB, and the DE parameters:

a. $NP = 100, F \in \{0.5, 1, 2\}, CR = 0.9, G_{max} = 100$. The results of the DE opti-

mization is illustrated in Fig. 6.11.a

b. $NP = 100, F = 1, CR \in \{0.1, 0.5, 0.9\}, CR = 0.9, G_{max} = 100$. The results of the DE optimization algorithm can be seen in Fig. 6.11.b.

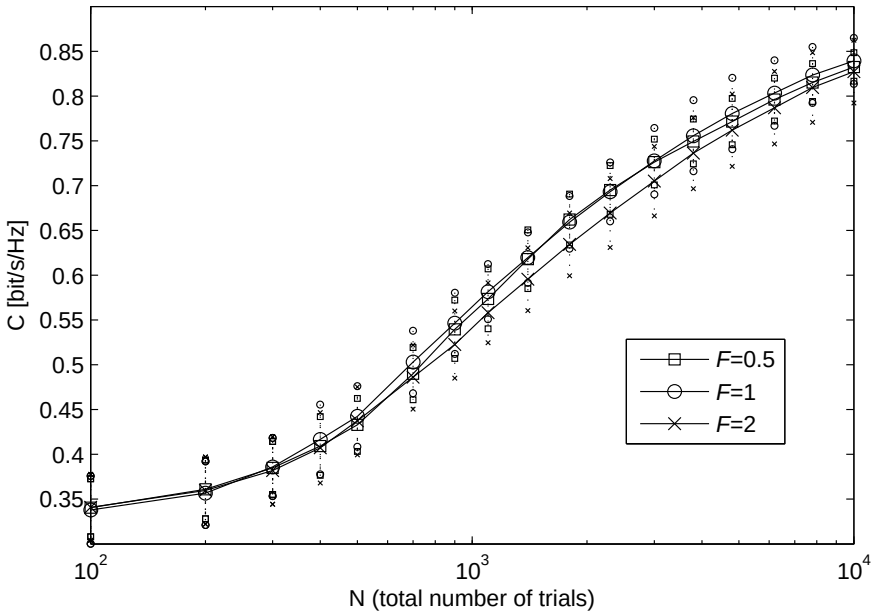

Figure 6.10. DE variation with F.

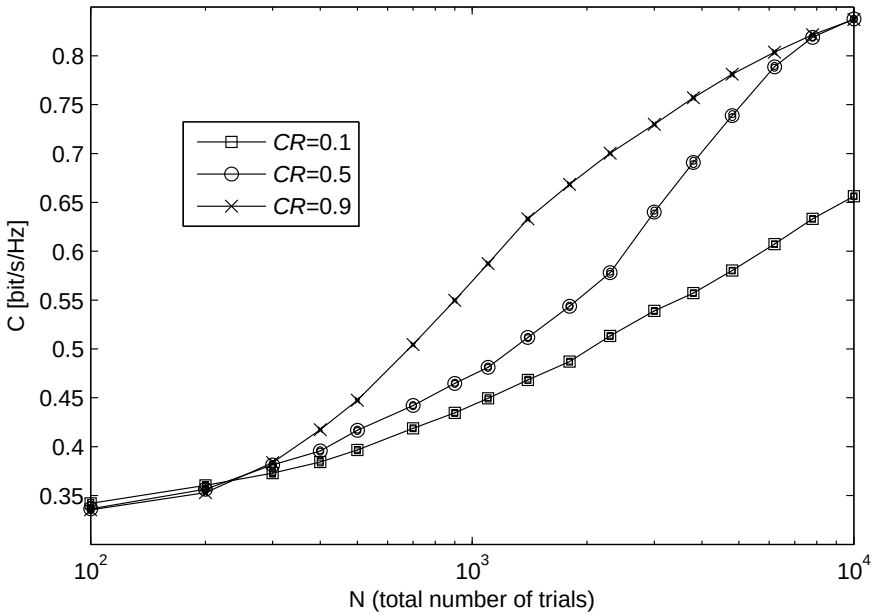

Figure 6.11. DE variation with CR.

In general, changing the value of F or that of CR does not affect the algorithm output consistently and significantly. In further testing, F is set to 1 and CR to 0.9.

Rank constraints could be taken into account in this algorithm by running it separately for different classes of matrix rank. This direction was not explored.

Example 6.6: *Covariance matrix optimization using DE*

Consider:

$$\mathbf{W}_1 = \frac{1}{m}\mathbf{E}_m, \quad \mathbf{W}_2 = 0.1\mathbf{W}_1, \quad m \in \{2, 4, 8, 16\} \tag{6.18}$$

and let SNR = 0dB. The DE parameters are: $NP = 200, CR = 0.9, F = 1, G_{max} = 500$. Fig. 6.12 and 6.13 show the output of the DE algorithm and compare it to the Rank-adaptive Monte-Carlo optimization. The total number of different covariance matrices tested is $N = NP \cdot G_{max}$. For DE, the horizontal axis starts at $N = NP$—after the members of the first generation were tested. The secrecy rate that DE returns after the first generation is smaller than the Rank-adaptive Monte-Carlo result.

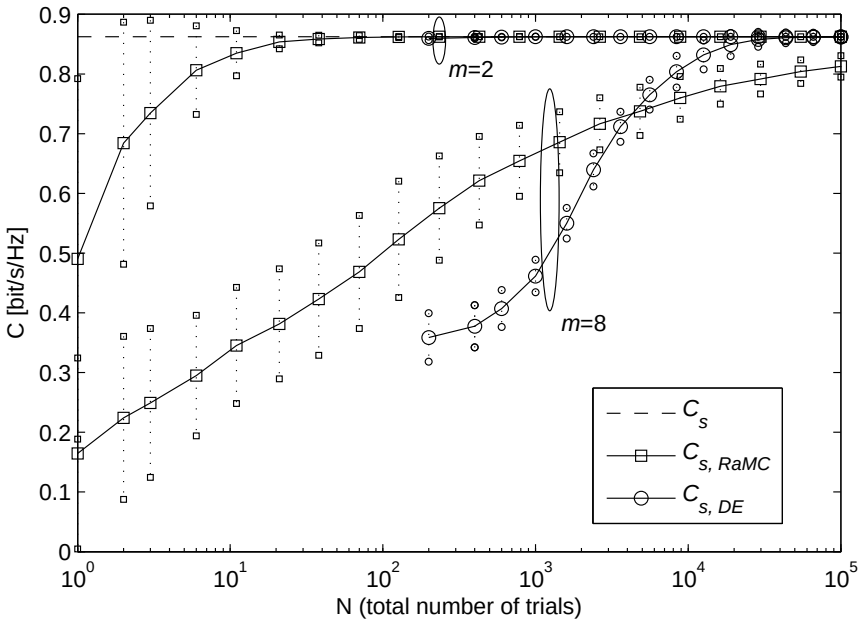

Figure 6.12. DE compared with RaMC, $m \in \{2, 8\}$.

For small systems ($m = 2$), the rank-adaptive algorithm approaches the secrecy capacity

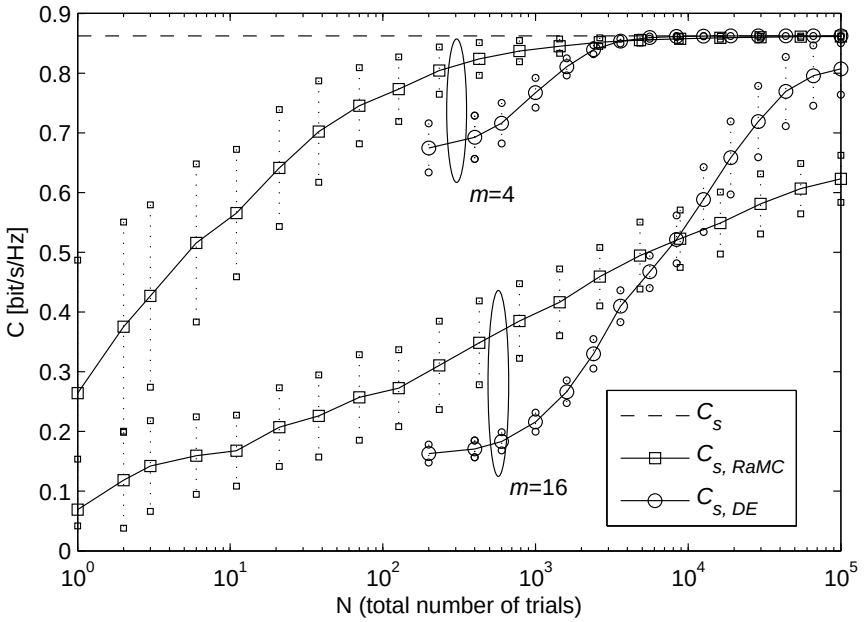

Figure 6.13. DE compared with RaMC, $m \in \{4, 16\}$.

in less than NP tests. In this case, for DE to be efficient, the value of NP should be somewhat smaller. For larger systems, the DE solution outperforms the RaMC algorithm in the long run. For $m = 4$ and $m = 8$ this occurs after approximately 10^3 tests. For $m = 16$, the DE algorithm curve flattens before the optimal value is reached. This is because the chosen population size considered for optimization is not large enough for the system, it does not offer enough randomness. For large values of m, the value of NP needs to be larger.

The error bars in Fig. 6.12 have values that are larger than C_s when $m = 2$. This is because of how the error bars are defined. Note that the values returned by RaMC are never larger than the secrecy capacity. The values of $C_{s,RaMC}$ are not uniformly distributed between 0 and C_s. The large variation in the values lower than the mean leads to a large variation on average. This average is not representative of the distribution of values higher than the mean value.

When its parameters are well chosen, although slow in the beginning, DE can return covari-

ance matrices that approach channel capacity faster than the Rank-adaptive Monte-Carlo method. However, when the allowed run-time for the algorithms is limited, the amount of testing required makes DE ineffective and this approach is not practical.

6.4 Summary

In this chapter, two Monte-Carlo based algorithms are used to search for the optimal covariance matrix. Neither of them require convexity of the underlying problem or any other specific structure. What is more, these random search algorithms produce results much faster than deterministic techniques when m is small.

The first approach takes into account the fact that the rank of the optimal transmitter covariance matrix can take only certain values as in (6.10). Searching among matrices of lower rank ignores possible sub-channels and limits the maximal secrecy rate. Searching among matrices of higher rank is redundant and requires more trials before converging. A compromise is to cycle through the possible ranks of the covariance matrix. This is the Rank-adaptive Monte-Carlo algorithm.

The second approach is differential evolution. This solution emulates natural evolution techniques such as mutation, crossover and selection. It is a random iterative process in which the candidate solutions are used to generate new candidate solutions using simple nature-like mechanisms. New and existing solutions are compared to a threshold and the best candidates are kept for the next iteration. For optimal results, the parameters of this algorithm need to be chosen carefully.

If the goal is to find the optimal covariance matrix, in the long run, DE returns better results faster than the Rank-adaptive Monte-Carlo algorithm. However, if the computational time and the resources are limited, DE is not the best choice.

Chapter 7

Numerical Optimization

Using a Monte-Carlo optimization to search for the optimal transmit covariance matrix requires a lot of time, especially for large systems. In the third example in Chapter 6, for $m = 16$, Rank-adaptive Monte-Carlo (RaMC) optimization can only reach 75% of the secrecy capacity value after 10^5 trials. Obtaining an analytical solution is difficult. Furthermore, explicit results are known only in a few special cases. To achieve better results, a different approach is needed. This chapter presents two possible solutions. They combine the knowledge acquired from analysis with the power of optimization tools and search for the optimal transmit covariance matrix in iterative steps.

7.1 Successive Linear Approximation

The successive linear approximation (SLA) method starts at an estimate of the optimal variable and uses the first-order approximation of the objective function to find a maximization direction. It then updates the starting point accordingly. The process is repeated until the stopping criteria are met e.g. when the starting point no longer changes significantly or after a set number of iterations. When the objective function is concave, this algorithm converges

to the maximal value; when it is not, it may stop at a local optimum.

The function to maximize is the secrecy rate $C_s(\mathbf{R})$:

$$C_s(\mathbf{R}) = \ln \frac{\det(\mathbf{I} + \mathbf{W}_1\mathbf{R})}{\det(\mathbf{I} + \mathbf{W}_2\mathbf{R})} \tag{7.1}$$

When the value of the expression is negative, it is interpreted as 0.

Let $\mathbf{R}_0$ be a starting point; (7.1) can be re-written as:

$$\begin{aligned} C_s(\mathbf{R}) &= \ln \frac{\det(\mathbf{I} + \mathbf{W}_1\mathbf{R})}{\det(\mathbf{I} + \mathbf{W}_2\mathbf{R})} \\ &= \ln \frac{\det(\mathbf{I} + \gamma\mathbf{W}_1(\mathbf{R}_0 + \Delta\mathbf{R}))}{\det(\mathbf{I} + \gamma\mathbf{W}_2(\mathbf{R}_0 + \Delta\mathbf{R}))} \\ &= \ln \frac{\det(\mathbf{I} + \gamma\mathbf{W}_1\mathbf{R}_0)}{\det(\mathbf{I} + \gamma\mathbf{W}_2\mathbf{R}_0)} + \ln \frac{\det(\mathbf{I} + (\mathbf{I} + \gamma\mathbf{W}_1\mathbf{R}_0)^{-1}\mathbf{W}_1\Delta\mathbf{R})}{\det(\mathbf{I} + (\mathbf{I} + \gamma\mathbf{W}_2\mathbf{R}_0)^{-1}\mathbf{W}_2\Delta\mathbf{R})} \\ &= C_s(\mathbf{R}_0) + \Delta C_s(\Delta\mathbf{R}) \end{aligned} \tag{7.2}$$

The first term is equal to the secrecy rate when $\mathbf{R} = \mathbf{R}_0$. The second one represents the difference between this rate and the secrecy capacity. Both $\mathbf{R}$ and $\mathbf{R}_0$ are covariance matrices under the total power constraint. Hence, $\Delta\mathbf{R} = \mathbf{R} - \mathbf{R}_0$ needs to be such that:

$$\operatorname{tr}\Delta\mathbf{R} = 0 \text{ and } \mathbf{R}_0 + \Delta\mathbf{R} \succeq \mathbf{0} \tag{7.3}$$

The expansion changes the optimization variable from $\mathbf{R}$ to $\Delta\mathbf{R}$ and the objective function from $C_s(\mathbf{R})$ to $\Delta C_s(\Delta\mathbf{R})$.

For $\Delta\mathbf{R}$ in the vicinity of $\mathbf{R}_0$, the Taylor series expansion can be used to approximate $\Delta C_s(\Delta\mathbf{R})$:

$$\Delta C_s(\Delta\mathbf{R}) = \ln \frac{\det(\mathbf{I} + (\mathbf{I} + \mathbf{W}_1\mathbf{R}_0)^{-1}\mathbf{W}_1\Delta\mathbf{R})}{\det(\mathbf{I} + (\mathbf{I} + \mathbf{W}_2\mathbf{R}_0)^{-1}\mathbf{W}_2\Delta\mathbf{R})} \approx \operatorname{tr}\left((\mathbf{Z}_1 - \mathbf{Z}_2)\Delta\mathbf{R}\right) \tag{7.4}$$

where $\mathbf{Z}_{1(2)} = (\mathbf{I}+\mathbf{W}_{1(2)}\mathbf{R}_0)^{-1}\mathbf{W}_{1(2)}$. For the approximation to hold, $(\mathbf{I}+\mathbf{W}_{1,(2)}\mathbf{R}_0)^{-1}\Delta\mathbf{R} \ll \mathbf{I}$

or, equivalently, $\Delta\mathbf{R} \preceq \varepsilon_{\mathbf{R}}\mathbf{I}$, where $\varepsilon_{\mathbf{R}} \simeq \max\left(\|\mathbf{Z}_1\|_2, \|\mathbf{Z}_2\|_2\right)^{-1}$.

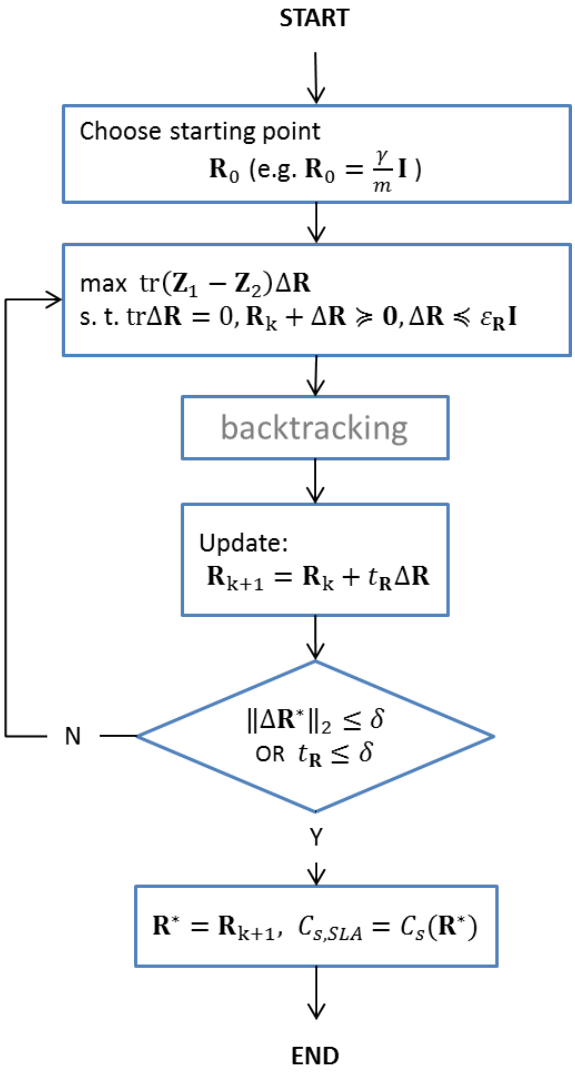

Figure 7.1. Optimization via Successive Linear Approximation.

This is a linear function in $\Delta\mathbf{R}$. Maximizing the approximation returns $\Delta\mathbf{R}^*$ which effectively maximizes the value of $C_s(\mathbf{R})$ in a small region around $\mathbf{R}_0$. As a result, the matrix $\mathbf{R}_1 = \mathbf{R}_0 + \Delta\mathbf{R}^*$ gives a communication rate $C_s(\mathbf{R}_1) \geq C_s(\mathbf{R}_0)$. Replacing $\mathbf{R}_0$ with $\mathbf{R}_1$ and repeating the process increases the value of $C_s(\mathbf{R})$ until a (local) maximum is found. The optimization problem is transformed into an iterative linear problem in $\Delta\mathbf{R}$. The linear problem solved

at each step is:

$$\max_{\Delta\mathbf{R}} \operatorname{tr}\left((\mathbf{Z}_1 - \mathbf{Z}_2)\Delta\mathbf{R}\right), \quad \text{s.t. } \operatorname{tr}\Delta\mathbf{R} = 0, \mathbf{R}_0 + \Delta\mathbf{R} \succeq \mathbf{0}, \Delta\mathbf{R} \preceq \varepsilon_{\mathbf{R}}\mathbf{I} \tag{7.5}$$

where $\varepsilon_{\mathbf{R}} \leq \max\left(\|\mathbf{Z}_1\|_2, \|\mathbf{Z}_2\|_2\right)^{-1}$ is selected such that the approximation in (7.4) holds. $\varepsilon_{\mathbf{R}}$ defines the linear approximation step size. $\mathbf{Z}_1$ and $\mathbf{Z}_2$ are updated at every iteration. After several iterations, $\Delta\mathbf{R}^*$ should be approaching $\mathbf{0}$. The flowchart for the successive linear approximation algorithm is shown in Fig. 7.1. In this figure, δ is a small constant which represents numerical zero. The corresponding MATLAB code is included in the Appendix. The meaning of *backtracking* and that of $t_{\mathbf{R}}$ are explained below.

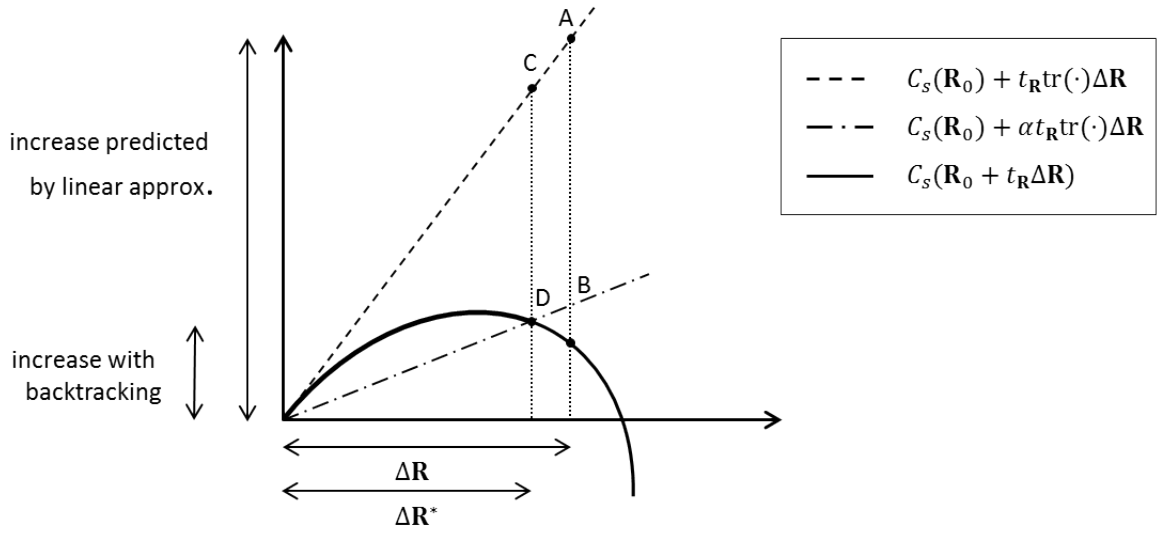

Figure 7.2. Backtracking line search in a maximization problem. This figure illustrates the backtracking concept; it is understood that $\Delta\mathbf{R}$ is not a scalar.

The optimization in (7.5) returns $\Delta\mathbf{R}^*$. This indicates the maximization direction for $\Delta C_s(\Delta\mathbf{R})$. The increase in $C_s(\mathbf{R})$, predicted by the linear approximation, overestimates the value of the function at $\mathbf{R}_0 + \Delta\mathbf{R}$. In Fig. 7.2, this is indicated by point A. Point B is the actual value of $C_s(\mathbf{R}_0 + \Delta\mathbf{R})$. To find the step size, a line search along $C_s(\mathbf{R}_0 + t_{\mathbf{R}}\Delta\mathbf{R})$ is required.

The line search is made computationally efficient by backtracking[1]. Backtracking decreases the step size and makes it such that the predicted value deviates less from the function to optimize. After backtracking, the linear approximation has the value indicated by point C while the objective function value is indicated by D—a higher value than the one before backtracking. Fig. 7.3 shows a flow-chart of the backtracking portion of the algorithm. α and β are two constants. α represents the acceptable decrease from what the linear extrapolation predicted. β determines the length of the step at every iteration. The value of these constants affects the precision of the final function objective value and the speed of convergence [23]. In this thesis, both α and β are set to 0.5.

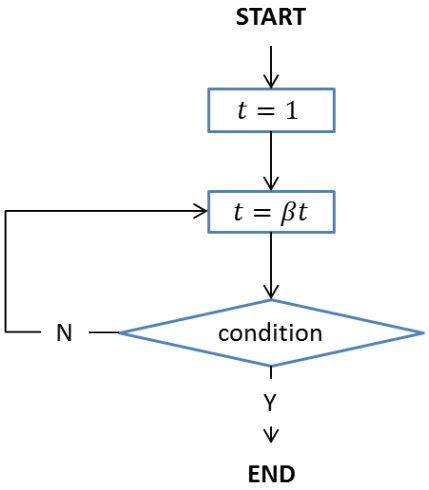

Figure 7.3. Backtracking.

The initial step size is set to a unit of $\Delta\mathbf{R}$: $t_{\mathbf{R}} = 1$. Changing the value of $t_{\mathbf{R}}$ ($t_{\mathbf{R}} := \beta t_{\mathbf{R}}$) brings the step closer to the starting point. When:

$$C_s(\mathbf{R}_k + \Delta\mathbf{R}) \geq C_s(\mathbf{R}_k) + \alpha t_{\mathbf{R}} \cdot \mathrm{tr}\left((\mathbf{Z}_1 - \mathbf{Z}_2)\Delta\mathbf{R}\right) \tag{7.6}$$

[1] An exact linear search requires evaluating the objective function and one or more of its derivatives along the search line [23]. Although less precise, backtracking makes use of the search direction and only evaluates the function. A detailed explanation for backtracking, including convergence considerations is provided in [23].

backtracking ends and a new iteration can begin. The $\mathbf{R}_k + t_{\mathbf{R}}\Delta\mathbf{R} \succeq \mathbf{0}$ condition ensures that the line search is done in the domain of $C_s(\mathbf{R})$. Backtracking guarantees monotonicity, i.e. the value of the estimated optimal point increases at every step of the algorithm, and avoids oscillating behavior close to the optimal point.

Impact of the Starting-point Matrix

The choice of $\mathbf{R}_0$ can change the convergence time and affect the optimal secrecy capacity value returned by SLA when $C_s(\mathbf{R})$ is not concave. A natural choice for $\mathbf{R}_0$ is a scaled identity matrix. This makes no assumption about the channel matrices and treats all communication directions equally. However, other choices might be optimal depending on the channel matrices.

Example 7.1: *The starting point can affect the algorithm output*

Consider:

$$\mathbf{W}_1 = \begin{bmatrix} 4 & 0 \\ 0 & 1 \end{bmatrix}, \quad \mathbf{W}_2 = \begin{bmatrix} 1 & 0 \\ 0 & 4 \end{bmatrix} \tag{7.7}$$

and let SNR = 20dB. Since $\mathbf{W}_1 - \mathbf{W}_2$ has only one positive eigenvalue, $\operatorname{rank}\mathbf{R}^* = 1$ and the eigenvector of $\mathbf{R}^*$ is eigenvector corresponding to the positive eigenvalue of $\mathbf{W}_1 - \mathbf{W}_2$ [36]:

$$\mathbf{R}^* = \begin{bmatrix} \gamma & 0 \\ 0 & 0 \end{bmatrix} \tag{7.8}$$

Table 7.1 shows how choosing the starting point can affect convergence. $n_{\mathbf{R}}$ is the number of iterations. The closer the starting point is to the optimal covariance matrix, the faster the convergence. In this case, $\mathbf{R}_3$ requires the least number of iterations. In fact, this is the optimal covariance matrix.

Table 7.1. Optimization with different starting-points.

$n_{\mathbf{R}}$	$\mathbf{R}_1 = \mathbf{I}$	$\mathbf{R}_2 = \begin{bmatrix} 0 & 0 \\ 0 & \gamma \end{bmatrix}$	$\mathbf{R}_3 = \begin{bmatrix} \gamma & 0 \\ 0 & 0 \end{bmatrix}$
	$C_{s,SLA}$	$C_{s,SLA}$	$C_{s,SLA}$
5	0	0.3829	1.3788
10	1.3788	1.3788	

At every step, the role of $\Delta\mathbf{R}$ is to reallocate power between the transmitter antennas and to change the correlation of the signals being sent.

Approximation Accuracy and Speed of Convergence

The variable $\varepsilon_{\mathbf{R}}$ controls how large $\Delta\mathbf{R}$ is allowed to be; $t_{\mathbf{R}}$, the backtracking variable, modifies the step size in a linear manner. Let:

$$\varepsilon_{\mathbf{R}} = a_{\mathbf{R}} \cdot \max\left(\|(\mathbf{I} + \mathbf{W}_1\mathbf{R})^{-1}\mathbf{W}_1\|_2, \|(\mathbf{I} + \mathbf{W}_2\mathbf{R})^{-1}\mathbf{W}_2\|_2\right)^{-1} \tag{7.9}$$

Tables 7.2 and 7.3 show how different values of $a_{\mathbf{R}}$ change the accuracy of results, $C_{s,SLA}$ after 20 iterations. Two additional columns show the optimization results when the constraint in $\varepsilon_{\mathbf{R}}$ is not used and when the value of the secrecy capacity after 10^5 Monte-Carlo trials. The test channel matrices are generated at random[2]. Each line of the table represents a different channel realization.

In general, choosing an $\varepsilon_{\mathbf{R}}$ that is too small, increases the number of iterations needed to achieve the optimal value. Allowing a large value of $\varepsilon_{\mathbf{R}}$ is often beneficial for both small and large systems. However, the best choice depends on the channel matrices and cannot be known a-priori. Unless otherwise specified, for simplicity, $a_{\mathbf{R}}$ is set to 1.

[2]Many of the test-cases presented in this chapter make use of systems for which the channel matrix is randomly generated: the channel matrix elements are i.i.d and follow a Gaussian distribution $\mathcal{N}(0, 1)$.

Table 7.2. Impact of the step length, $m = 2$.

SNR [dB]	$C_{s,SLA}$ $\mathrm{a}_{\mathbf{R}} = 0.1$	$C_{s,SLA}$ $\mathrm{a}_{\mathbf{R}} = 1$	$C_{s,SLA}$ $\mathrm{a}_{\mathbf{R}} = 10$	$C_{s,SLA}$ $\varepsilon_{\mathbf{R}} = \infty$	$C_{s,RaMC}$
-20	0.0590	0.0590	0.0590	0.0590	0.0590
	0.0030	0.0030	0.0030	0.0030	0.0030
	0.0431	0.0431	0.0431	0.0431	0.0431
0	0.9098	0.9099	0.9102	0.9098	0.9098
	0.8038	0.8038	0.8042	0.8038	0.8038
	1.4657	1.4658	1.4658	1.4656	1.4657
20	2.0325	3.5672	3.5673	3.5673	3.5671
	4.3858	4.3858	4.3858	4.3858	4.3858
	0	1.2477	1.2524	1.2519	1.2838

Table 7.3. Impact of the step length, $m = 16$.

SNR [dB]	$C_{s,SLA}$ $\mathrm{a}_{\mathbf{R}} = 0.1$	$C_{s,SLA}$ $\mathrm{a}_{\mathbf{R}} = 1$	$C_{s,SLA}$ $\mathrm{a}_{\mathbf{R}} = 10$	$C_{s,SLA}$ $\varepsilon_{\mathbf{R}} = \infty$	$C_{s,RaMC}$
-20	0.3745	0.3741	0.3743	0.3740	0.1010
	0.3881	0.3881	0.3882	0.3881	0.0907
	0.3786	0.3781	0.3785	0.3782	0.1060
0	7.4177	6.0495	5.0676	5.0677	2.1035
	7.2910	6.4071	4.9747	4.9747	3.0412
	7.0038	5.3833	4.9046	4.9047	2.1091
20	2.1511	4.0337	5.3532	8.0536	2.4022
	7.4348	9.4751	10.8007	7.3101	6.2896
	7.6820	10.2781	9.9156	5.8744	5.0307

Optimum Value versus the Number of Iterations

There is a trade-off between the computational time and the accuracy. Tables 7.4 and 7.5[3] compare the algorithm output to the result that is obtained after 10^5 tests in a Monte-Carlo optimization. They show the optimal value of $C_s(\mathbf{R})$ after 1, 10 and 20 iterations. The last two columns show the $C_s(\mathbf{R})$ value that is achieved when the constraint in $\varepsilon_{\mathbf{R}}$ is not considered.

At SNR = 0dB, the successive linear approximation requires a small number of iterations

[3]The test channel matrices used to generate the results in the tables were generated randomly. The channel matrix elements are i.i.d and follow a Gaussian distribution $\mathcal{N}(0, 1)$.

Table 7.4. SLA iterations and $C_{s,SLA}$, $m = 2$ as a function of the coefficient $a_{\mathbf{R}}$ and of $n_{\mathbf{R}}$.

SNR [dB]	$C_{s,RaMC}$	$C_{s,SLA}$ $n_{\mathbf{R}} = 1$	$C_{s,SLA}$ $n_{\mathbf{R}} = 10$	$C_{s,SLA}$ $n_{\mathbf{R}} = 20$	$C_{s,SLA}$ $\varepsilon_{\mathbf{R}} = \infty$, $n_{\mathbf{R}} = 1$	$C_{s,SLA}$ $\varepsilon_{\mathbf{R}} = \infty$, $n_{\mathbf{R}} = 20$
-20	0.0176	0.0176	0.0176	0.0176	0.0175	0.0176
	0.0043	0.0043	0.0043	0.0043	0.0043	0.0043
	0.0142	0.0142	0.0142	0.0142	0.0142	0.0142
0	1.3153	1.3114	1.3153	1.3153	1.3114	1.3153
	0.4211	0.3095	0.4213	0.4213	0.3095	0.4211
	0.3392	0.3304	0.3392	0.3392	0.3304	0.3392
20	1.8340	0.8548	1.2232	1.5081	0.8551	1.5162
	1.6790	1.0274	1.1406	1.2489	1.0289	1.2475
	1.3438	1.3287	1.3377	1.3392	1.3350	1.3395

Table 7.5. SLA iterations and $C_{s,SLA}$, $m = 16$ as a function of the coefficient $a_{\mathbf{R}}$ and of $n_{\mathbf{R}}$.

SNR [dB]	$C_{s,RaMC}$	$C_{s,SLA}$ $n_{\mathbf{R}} = 1$	$C_{s,SLA}$ $n_{\mathbf{R}} = 10$	$C_{s,SLA}$ $n_{\mathbf{R}} = 20$	$C_{s,SLA}$ $\varepsilon_{\mathbf{R}} = \infty$, $n_{\mathbf{R}} = 1$	$C_{s,SLA}$ $\varepsilon = \infty$, $n_{\mathbf{R}} = 20$
-20	0.0995	0.3593	0.3788	0.3788	0.3593	0.3788
	0.1033	0.3772	0.3829	0.3830	0.3772	0.3830
	0.0941	0.4093	0.4185	0.4185	0.4088	0.4183
0	2.7477	1.1727	3.7156	5.0494	1.1728	5.0495
	1.6998	0.7489	3.5872	4.7505	0.7489	4.7504
	3.3847	1.7785	3.8102	5.2627	1.7785	5.2627
20	4.8573	1.6439	7.3693	9.0866	1.6440	10.7527
	7.2371	3.0312	8.3256	10.9064	3.0312	8.0428
	1.6776	5.9839	7.6526	8.4706	5.9782	10.5921

before surpassing the value returned by Monte-Carlo, this is especially visible when the system size is large. At lower SNR values, the approximation in (7.4) is accurate even when no constraint is imposed on the values in $\Delta\mathbf{R}$. In this SNR range, the result returned by SLA after one iteration is better than the result obtained with a RaMC estimation after 10^5 tests.

During the random testing performed for the successive linear algorithm some other cases where convergence occurs in one step were observed. In these cases, the constraint $\Delta\mathbf{R} \ll \varepsilon_{\mathbf{R}}\mathbf{I}$ is removed. The occurrence of these cases is higher at low SNR and when the rank of the difference matrix is 1. Both of these situations correspond to systems where the optimal covariance matrix rank is 1.

Accuracy and Timing Considerations, Comparison with the Monte-Carlo Algorithm

Tables 7.6 and 7.7 show the secrecy capacity values achieved with the successive linear approximation and the Rank-adaptive Monte-Carlo algorithm in roughly the same computational time[4], the channel matrix entries are chosen randomly. For smaller systems, the results are

Table 7.6. Time measurements, $m \in \{2, 16\}$, SNR = 0dB.

iterations/tests	$C_{s,RaMC}$	$C_{s,SLA}$	t_{RaMC} [s]	t_{SLA} [s]
$n_{\mathbf{R}} = 1$	0.8603	0.8336	0.1375	1.3139
$5 \cdot 10^3$ tests	1.5499	1.5476	0.1350	0.1776
	0.5607	0.2602	0.1341	0.1501
$n_{\mathbf{R}} = 5$	0.1124	0.1126	0.6728	0.7143
$25 \cdot 10^3$ tests	1.6055	1.6053	0.6735	0.7504
	0.3500	0.3504	0.6839	0.7466
$n_{\mathbf{R}} = 10$	0.9612	0.9611	1.3403	1.4302
$5 \cdot 10^4$ tests	1.2558	1.2560	1.3476	1.4409
	0.1577	0.1577	1.3445	1.4170

Table 7.7. Time measurements, $m = 16$, SNR = 0dB.

iterations/tests	$C_{s,RaMC}$	$C_{s,SLA}$	t_{RaMC} [s]	t_{SLA} [s]
$n_{\mathbf{R}} = 1$;	2.1387	1.0139	0.2972	0.3013
$5 \cdot 10^3$ tests	1.5867	0.4588	0.2939	0.3177
	2.4078	1.2040	0.2954	0.3207
$n_{\mathbf{R}} = 5$	1.8880	2.8492	1.4373	1.5607
$25 \cdot 10^3$ tests	1.8030	5.2370	1.4606	1.5526
	1.4329	2.1587	1.4360	1.6292
$n_{\mathbf{R}} = 10$;	0.8330	4.5460	2.9574	3.0853
$5 \cdot 10^4$ tests	1.7920	5.8095	2.8891	3.0379
	1.2616	4.9602	2.9141	3.2581

comparable. When the system is larger, the successive linear algorithm is performing better. For roughly the same computational time, the SLA algorithm returns a higher value for the secrecy rate.

[4]The measurements were made on a laboratory computer with the code presented in Appendix 9. The measurements are made for informative purpose only; the codes were not specifically optimized for efficiency.

In general, the secrecy capacity values returned by SLA are higher than those returned by RaMC. SLA is relatively easy to implement and can be stopped after a pre-determined number of iterations. Unfortunately, there is no indication of how far the secrecy rate obtained is from the secrecy capacity.

7.2 min-max

An equivalent expression for the secrecy capacity is given in [29] and subsequently in [32]:

$$C_s = \min_{\mathbf{K}} \max_{\mathbf{R}} \ln \frac{\det(\mathbf{I} + \mathbf{K}^{-1}\mathbf{H}\mathbf{R}\mathbf{H}^{+})}{\det(\mathbf{I} + \mathbf{W}_2\mathbf{R})}, \mathbf{H} = \begin{bmatrix} \mathbf{H}_1 \\ \mathbf{H}_2 \end{bmatrix}, \mathbf{K} = \begin{bmatrix} \mathbf{I} & \mathbf{\Phi} \\ \mathbf{\Phi}^{+} & \mathbf{I} \end{bmatrix} \tag{7.10}$$

Let:

$$C_s^u(\mathbf{R}, \mathbf{K}) = \ln \frac{\det(\mathbf{I} + \mathbf{K}^{-1}\mathbf{H}\mathbf{R}\mathbf{H}^{+})}{\det(\mathbf{I} + \mathbf{W}_2\mathbf{R})} \tag{7.11}$$

This expression is convex in $\mathbf{K}$ for a fixed $\mathbf{R}$ and concave in $\mathbf{R}$ for a fixed $\mathbf{K}$ and has a saddle-point optimum [32]. The optimization order can be reversed.

Optimization over R (K is fixed)

Let $\mathbf{W} = \mathbf{H}^{+}\mathbf{K}^{-1}\mathbf{H}$. The min-max problem is reduced to:

$$\begin{aligned} \mathbf{R}^*(\mathbf{K}) &= \arg\max_{\mathbf{R}} \ln \frac{\det(\mathbf{I} + \mathbf{K}^{-1}\mathbf{H}\mathbf{R}\mathbf{H}^{+})}{\det(\mathbf{I} + \mathbf{W}_2\mathbf{R})} \\ &= \arg\max_{\mathbf{R}} \ln \frac{\det(\mathbf{I} + \mathbf{W}\mathbf{R})}{\det(\mathbf{I} + \mathbf{W}_2\mathbf{R})} \end{aligned} \tag{7.12}$$

where $\operatorname{tr}\mathbf{R} \leq P_T$. This is the same problem as in section 7.1, where $\mathbf{W}_1$ has been replaced by $\mathbf{W}$. Since the expression to optimize is concave in $\mathbf{R}$ [32], an optimal value is found after a sufficient number of iterations.

Optimization over K (R is fixed)

Let $\mathbf{Q} = \mathbf{H}\mathbf{R}\mathbf{H}^{+}$. The min-max problem becomes:

$$\begin{aligned}\mathbf{K}^{*}(\mathbf{R}) &= \arg\min_{\mathbf{K}} \ln \frac{\det(\mathbf{I}+\mathbf{K}^{-1}\mathbf{H}\mathbf{R}\mathbf{H}^{+})}{\det(\mathbf{I}+\mathbf{W}_2\mathbf{R})} \\ &= \arg\min_{\mathbf{K}} \ln\det(\mathbf{I}+\gamma\mathbf{K}^{-1}\mathbf{H}\mathbf{R}\mathbf{H}^{+}) \\ &= \arg\min_{\mathbf{K}} \ln\det(\mathbf{I}+\mathbf{K}^{-1}\mathbf{Q}) \\ &= \arg\max_{\mathbf{K}} \big(-\ln\det(\mathbf{I}+\mathbf{K}^{-1}\mathbf{Q})\big)\end{aligned} \tag{7.13}$$

where $\mathbf{K}$ has the form in (7.10). $\mathbf{K}^{*}(\mathbf{R})$ is optimal only for a given $\mathbf{R}$.Writing $\mathbf{K} = \mathbf{K}_0 + \Delta\mathbf{K}$ and using the Taylor series expansion, similar to what is done for $\mathbf{R}$ in (7.2), optimizing around $\mathbf{K}_0$ is equivalent to:

$$\max_{\mathbf{K}} \big(-\ln\det(\mathbf{I}+\mathbf{K}^{-1}\mathbf{Q})\big) = \max_{\Delta\mathbf{K}} \ln\frac{\det\mathbf{K}_0}{\det(\mathbf{K}_0+\mathbf{Q})} + \mathrm{tr}(\mathbf{K}_0^{-1} - (\mathbf{K}_0+\mathbf{Q})^{-1})\Delta\mathbf{K} \tag{7.14}$$

and the problem in (7.13) becomes:

$$\max_{\Delta\mathbf{K}} \mathrm{tr}(\mathbf{K}_0^{-1} - (\mathbf{K}_0+\mathbf{Q})^{-1})\Delta\mathbf{K} \quad \text{s.t.} \quad \mathbf{K}_0 + \Delta\mathbf{K} \succeq \mathbf{0}, \mathrm{tr}\,\Delta\mathbf{K} = 0, \Delta\mathbf{K} \preceq \varepsilon_{\mathbf{K}}\mathbf{I} \tag{7.15}$$

The constraints are the same as in (7.10) and the fact that the trace approximation is valid around $\mathbf{K}_0$.

For a given covariance matrix $\mathbf{R}$:

$$\ln\frac{\det(\mathbf{I}+\mathbf{W}_1\mathbf{R})}{\det(\mathbf{I}+\mathbf{W}_2\mathbf{R})} \leq \ln\frac{\det(\mathbf{I}+\mathbf{K}^{-1}\mathbf{H}\mathbf{R}\mathbf{H}^{+})}{\det(\mathbf{I}+\mathbf{W}_2\mathbf{R})} \tag{7.16}$$

The right-hand side of (7.16) is an upper bound to the secrecy capacity. Optimizing it on either $\mathbf{R}$ or $\mathbf{K}$ requires finding an extremum for $\ln\det$.

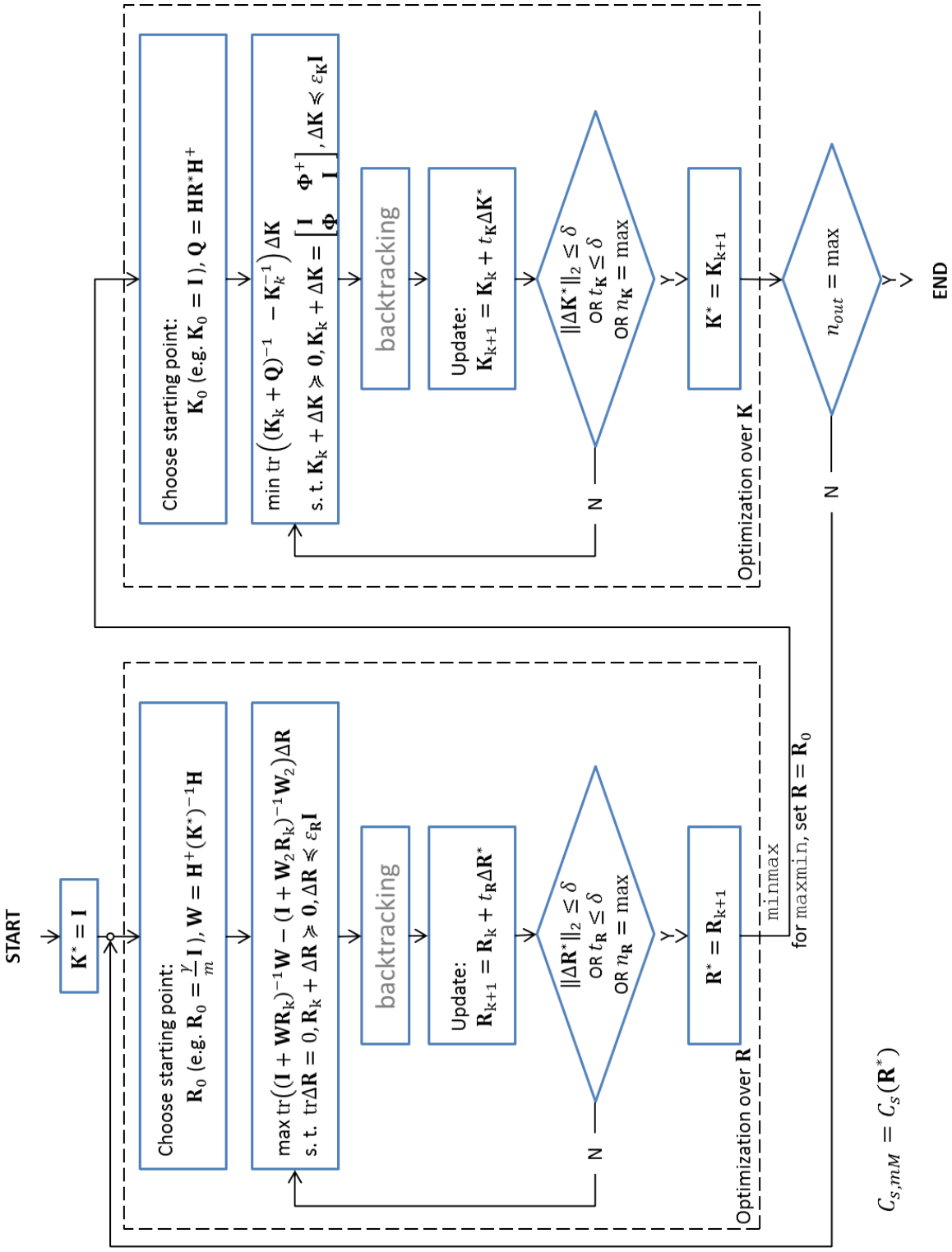

Figure 7.4. min-max algorithm (start with optimization over $\mathbf{R}$).

Optimization of $C_s^u(\mathbf{R}, \mathbf{K})$ is done separately over $\mathbf{R}$ and over $\mathbf{K}$. Fig. 7.4 shows the min-max algorithm flowchart. The variables are:

(7.17)

$\mathbf{R}_0 = \frac{\gamma}{m}\mathbf{I}$ (as is the case in the previous section). $\mathbf{K}_0 = \mathbf{I}$. This represents the case when the noise at Rx and Ev are not correlated. The dimensionality of $\mathbf{K}$ is $2m \times 2m$.

If optimization starts over $\mathbf{R}$ then it is in the form min-max. Starting with $\mathbf{K}$ effectively changes the problem to max-min. The min-max method requires multiple iterations to complete the optimization. However, since the optimization is done separately, neither $\mathbf{R}^*(\mathbf{K})$ or $\mathbf{K}^*(\mathbf{R})$ give the upper bound value immediately. The min-max (mM) algorithm consists of three loops: 2 inner loops: optimization over $\mathbf{R}$ when $\mathbf{K}$ is constant ($n_{\mathbf{R}}$ iterations) and optimization over $\mathbf{K}$ when $\mathbf{R}$ is constant ($n_{\mathbf{K}}$ iterations), and an outer loop (n_{out} indicates the amount of times that the algorithm should alternate between optimization over $\mathbf{R}$ and optimization over $\mathbf{K}$).

Optimum Value versus the Number of Iterations

The values assigned to $n_{\mathbf{R}}$, $n_{\mathbf{K}}$ and n_{out} determine the optimization precision. The operations in $\Delta\mathbf{R}$ and $\Delta\mathbf{K}$ optimization loops are similar to SLA. The computational time requirement is also similar.

Example 7.1: *Selecting the number of iterations in the min-max algorithm*

Consider:

$$\mathbf{H}_1 = \begin{bmatrix} 2 & 1 \\ 1 & 1 \end{bmatrix}, \quad \mathbf{H}_2 = \begin{bmatrix} 2 & 0 \\ 0 & 1 \end{bmatrix} \tag{7.18}$$

and SNR = 20dB.

Fig. 7.5–7.12 show the algorithm results when the total number of iterations $(n_{\mathbf{R}} + n_{\mathbf{K}})n_{out} = 50$. The horizontal axis shows the total number of approximations for $\Delta\mathbf{R}$

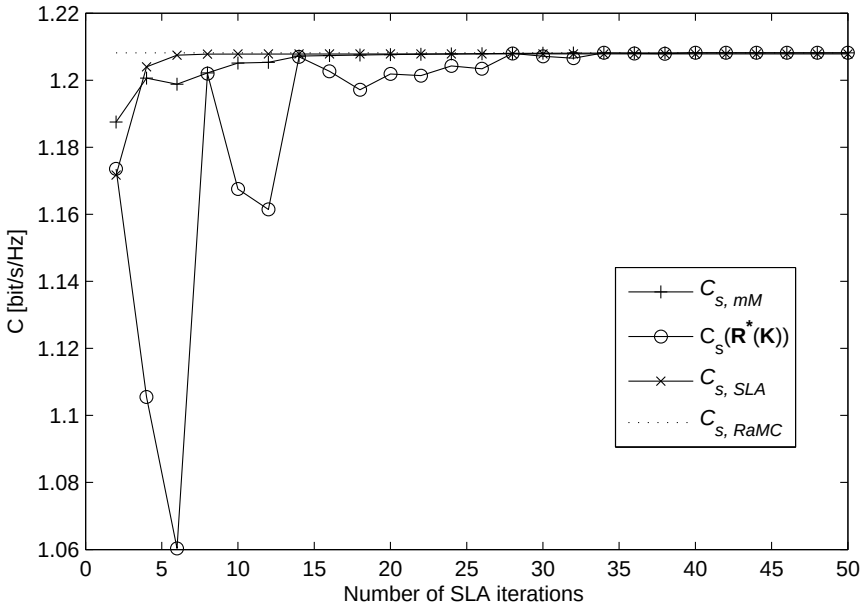

Figure 7.5. minmax, $n_{\mathbf{R}} = 1, n_{\mathbf{K}} = 1, n_{out} = 25$.

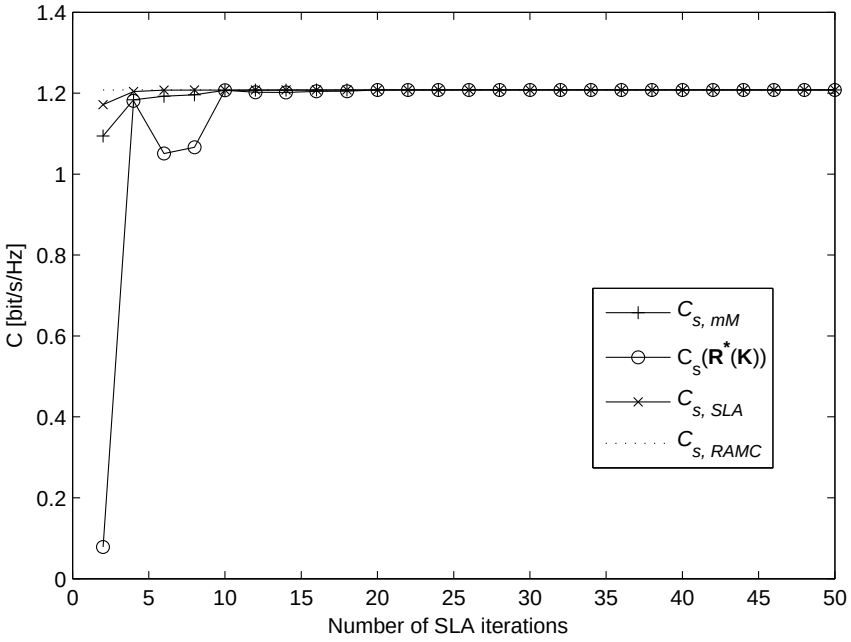

Figure 7.6. maxmin, $n_{\mathbf{R}} = 1, n_{\mathbf{K}} = 1, n_{out} = 25$.

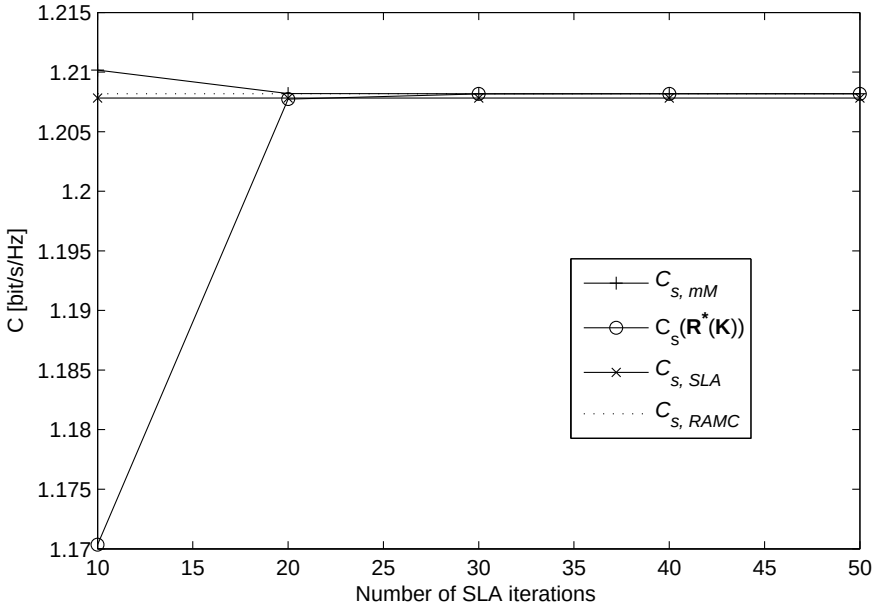

Figure 7.7. minmax, $n_{\mathbf{R}} = 9, n_{\mathbf{K}} = 1, n_{out} = 5$.

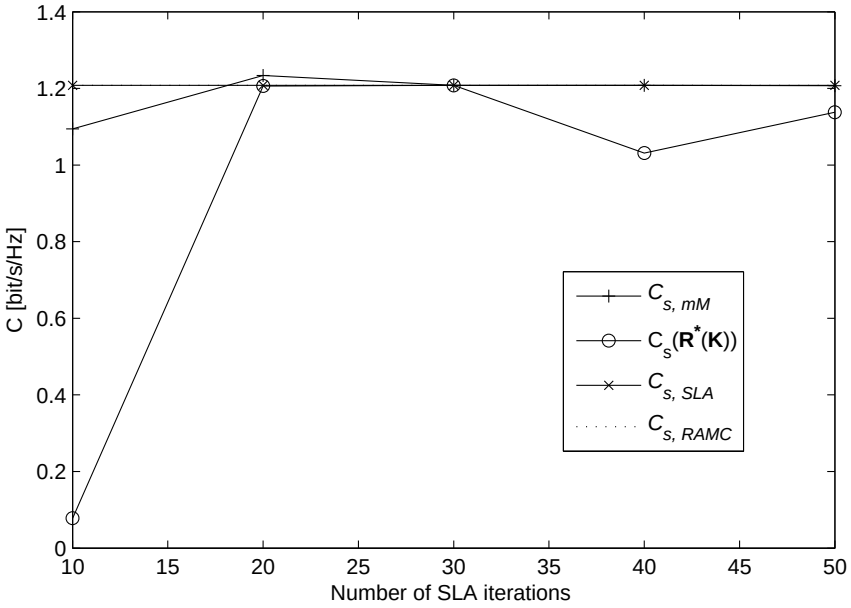

Figure 7.8. maxmin, $n_{\mathbf{R}} = 9, n_{\mathbf{K}} = 1, n_{out} = 5$.

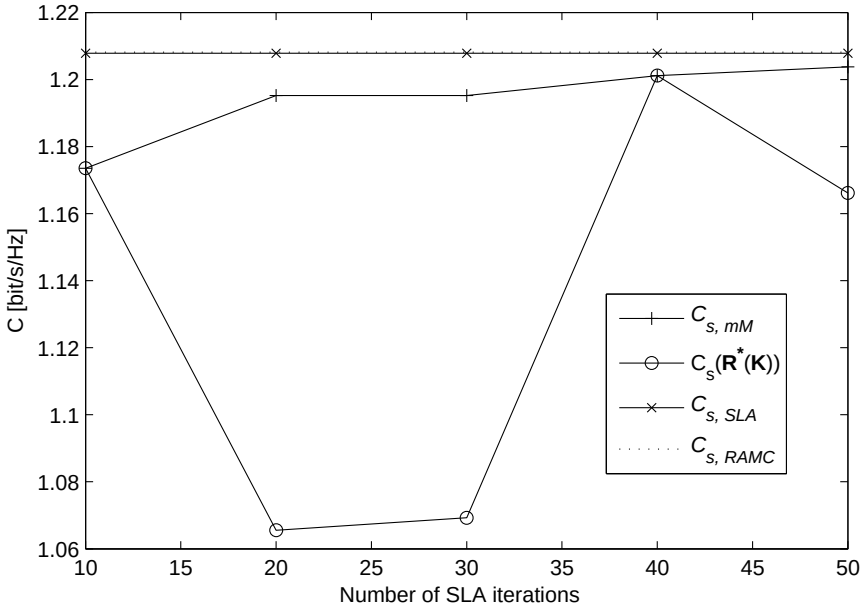

Figure 7.9. minmax, $n_{\mathbf{R}} = 1, n_{\mathbf{K}} = 9, n_{out} = 5$.

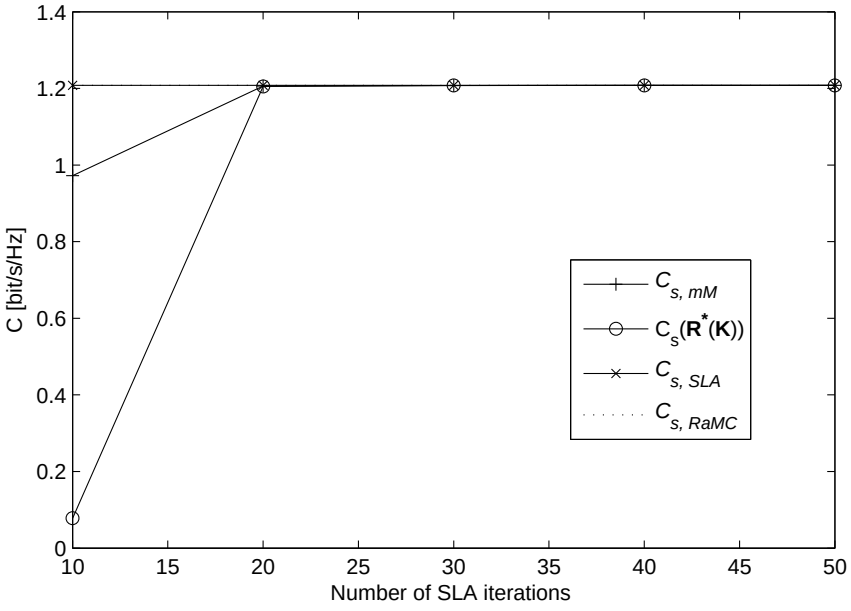

Figure 7.10. maxmin, $n_{\mathbf{R}} = 1, n_{\mathbf{K}} = 9, n_{out} = 5$.

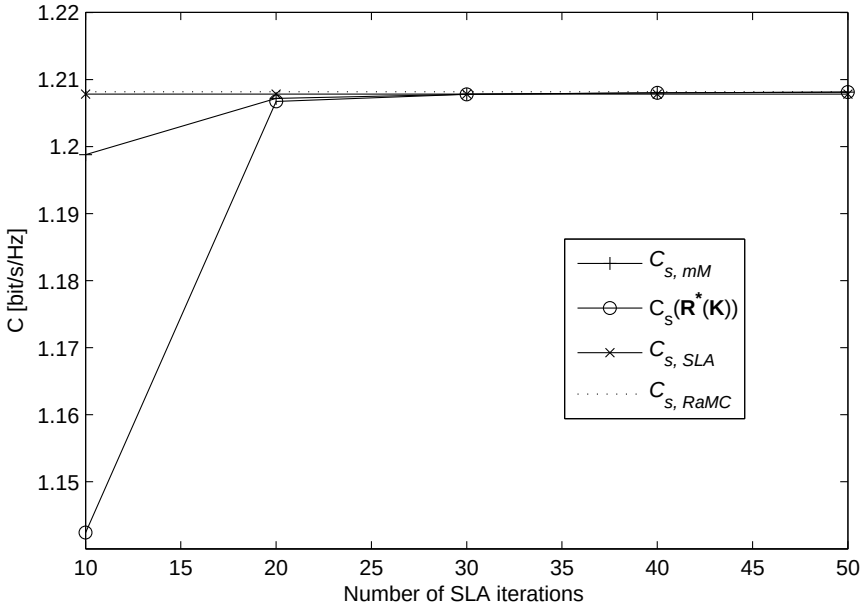

Figure 7.11. minmax, $n_{\mathbf{R}} = 5, n_{\mathbf{K}} = 5, n_{out} = 5$.

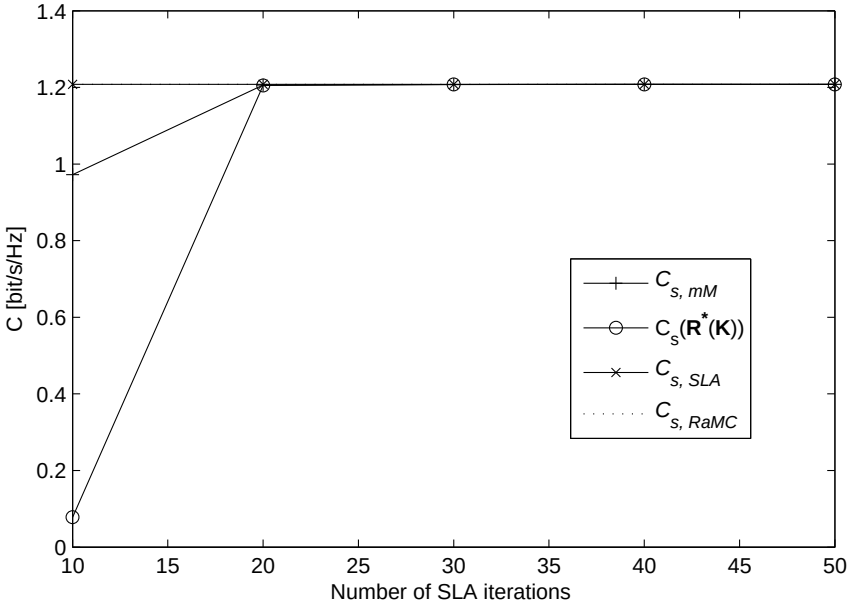

Figure 7.12. maxmin, $n_{\mathbf{R}} = 5, n_{\mathbf{K}} = 5, n_{out} = 5$.

and $\Delta\mathbf{K}$. There are four curves shown in each of the plots: $C_{s,mM} = C_s^u(\mathbf{R}^*, \mathbf{K}^*)$ is the value of (7.11) with the current $\mathbf{R}^*(\mathbf{K})$ and $\mathbf{K}^*(\mathbf{R})$, $C_s(\mathbf{R}^*(\mathbf{K}))$ is the secrecy capacity that could be attained with the current transmitter covariance $\mathbf{R}^*(\mathbf{K})$ in (7.1), $C_{s,SLA}$ is the secrecy capacity that is obtained with the successive linear approximation after the same number of iterations, and $C_{s,RaMC}$ is the secrecy capacity value returned by the RaMC algorithm after 10^5 tests.

The oscillations in $C_s(\mathbf{R}^*(\mathbf{K}))$ appear because $\mathbf{R}$ and $\mathbf{K}$ are not optimized simultaneously, i.e. once the $\mathbf{K}$ matrix changes, the optimal transmitter covariance needs to be adjusted significantly for optimal results. The $C_{s,mM}$ curve, corresponding to the current value of the min-max expression is smoother. This indicates that changes in $\mathbf{K}$ do not affect C_s in a significant way. If optimization starts over $\mathbf{K}$ first, the algorithm converges faster since $\mathbf{R}$ is optimized taking into account the worst situation.

Since optimization is done separately in $\mathbf{R}$ and $\mathbf{K}$, instability close to the convergence point may occur if the step size is not small enough. This is illustrated in the following example.

Example 7.2: *min-max instability around the convergence point*

Consider:

$$\mathbf{H}_1 = \begin{bmatrix} 0.7765 & -0.3039 \\ -0.3280 & -0.6409 \end{bmatrix}, \quad \mathbf{H}_2 = \begin{bmatrix} 0.5443 & -0.1055 \\ -0.9261 & -1.7141 \end{bmatrix}, \quad \text{SNR} = -20\text{dB} \tag{7.19}$$

The min-max results are shown in Fig. 7.13 and 7.14. $\varepsilon_{\mathbf{R}}$ and $\varepsilon_{\mathbf{K}}$, defined below, control how large the changes in $\mathbf{R}$ and $\mathbf{K}$ are allowed to be.

$$\begin{aligned} \varepsilon_{\mathbf{R}} &= a_{\mathbf{R}} \cdot \max\left(\|(\mathbf{I}+\mathbf{W}\mathbf{R})^{-1}\mathbf{W}\|_2, \|(\mathbf{I}+\mathbf{W}_2\mathbf{R})^{-1}\mathbf{W}_2\|_2\right)^{-1} \\ \varepsilon_{\mathbf{K}} &= a_{\mathbf{K}} \cdot \max\left(\|(\mathbf{K}+\mathbf{Q})^{-1}\|_2, \|\mathbf{K}^{-1}\|_2\right)^{-1} \end{aligned} \tag{7.20}$$

Oscillations in the optimal value appear because optimization is not done simultane-

ously for both $\mathbf{R}$ and $\mathbf{K}$. Precision is improved when $\varepsilon_{\mathbf{R}}$ and $\varepsilon_{\mathbf{K}}$ are assigned smaller values. However, this requires more iterations before the optimal value is approached.

Since the upper bound is only optimal for the current value of $\mathbf{R}^*(\mathbf{K})$ and $\mathbf{K}^*(\mathbf{R})$, it is not always monotonous. The value of the min-max expression oscillates around the saddle-point of $C_s^u(\mathbf{R}, \mathbf{K})$. The amplitude of the oscillations is smaller when smaller step-sizes are allowed. The oscillations in the optimal value vanish unless there is an accumulation of errors in the computation (see Fig. 7.14).

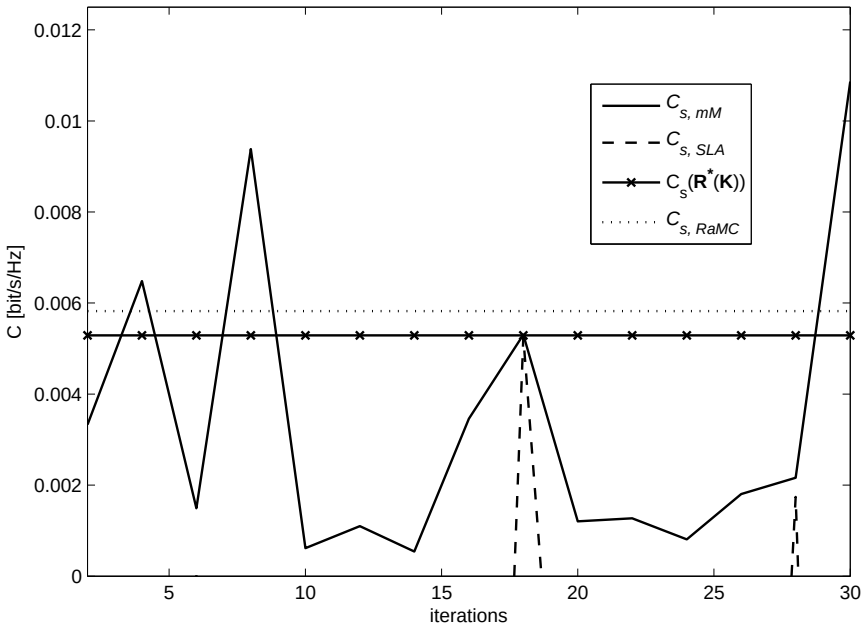

Figure 7.13. minmax oscillations, $a_{\mathbf{R}} = a_{\mathbf{K}} = 1$.

Nonetheless, the min-max algorithm returns an optimal value that is larger than the one returned by the successive approximation solution, which may be stuck at a local maximum. Moreover, this algorithm also gives $\mathbf{R}^*(\mathbf{K})$ which can be used in the secrecy capacity expression.

As shown on the plots, min-max does not always return C_s. In this case

$$C_s(\mathbf{R}^*(\mathbf{K})) \leq C_{s,RaMC} \leq C_s \tag{7.21}$$

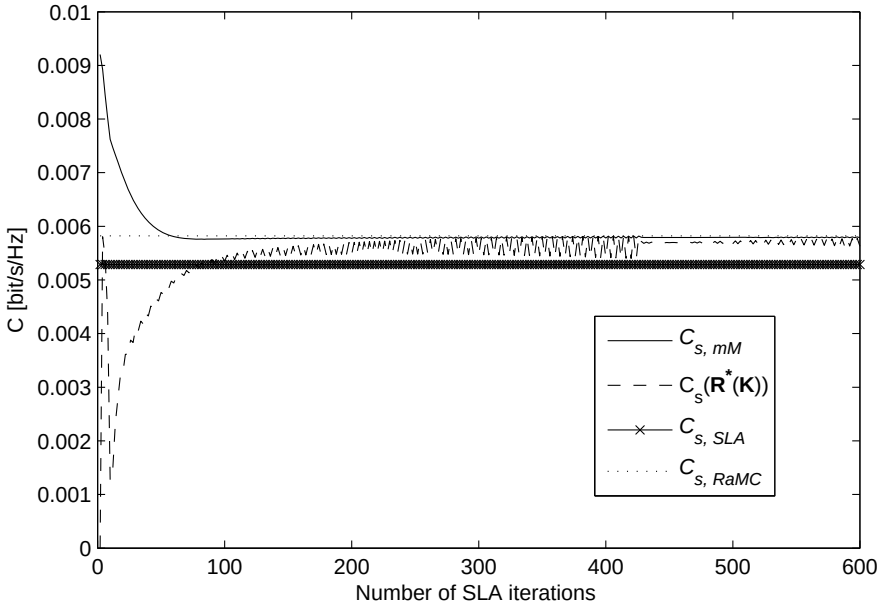

Figure 7.14. minmax oscillations, $a_{\mathbf{R}} = a_{\mathbf{K}} = 0.05$.

This holds even when the min-max algorithm has run for a sufficiently large number of iterations. The min-max result can be improved by choosing a smaller step size.

Testing with Random Channel Matrices

To explore the oscillatory behavior further, the min-max algorithm is tested on a set of randomly generated channel matrices. Table 7.8 compares the results obtained with the min-max algorithm with those obtained using SLA or using Monte-Carlo for different values of $\varepsilon_{\mathbf{R}}$ and $\varepsilon_{\mathbf{K}}$ as defined in (7.20).

Although $C_s^u(\mathbf{R}, \mathbf{K})$ and $C_s(\mathbf{R}^*(\mathbf{K}))$ approach each other, it does not mean that the algorithm has converged. This is because the upper limit obtained with the min-max algorithm is valid for the current value of $\mathbf{R}$ only. $\mathbf{R}^*(\mathbf{K})$ is not necessarily the optimal transmitter covariance. Furthermore, in some of these cases, the optimal value returned by the Monte-Carlo algorithm is higher.

It seems that for a certain class of channel matrices the maximal secrecy rate value is ap-

Table 7.8. Impact of $\varepsilon_{\mathbf{R}}$ and $\varepsilon_{\mathbf{K}}$, $m = 2$, SNR = 20dB, $n_{\mathbf{R}} = 1, n_{\mathbf{K}} = 1$.

	$C_s^u(\mathbf{R}^*, \mathbf{K}^*)$	$C_s(\mathbf{R}^*(\mathbf{K}))$	$C_{s,SLA}$	$C_{s,RaMC}$	$\lambda_1(\mathbf{W}_1 - \mathbf{W}_2)$	$\lambda_2(\mathbf{W}_1 - \mathbf{W}_2)$
	4.8586	4.7422	4.8612	6.1674	-1.5269	9.3692
	1.5288	1.5286	1.5281	1.5882	-0.6551	3.8476
	2.1573	1.9924	2.1642	2.6934	-0.6734	0.7171
$a_{\mathbf{R}} = 1$	2.8818	2.8818	2.8818	3.5464	0.0999	2.2369
$a_{\mathbf{K}} = 1$	3.5936	3.5916	3.5936	4.9592	-0.9698	1.8959
$n_{out} = 10$	0.7172	0.3285	0.3040	0.7530	-3.5255	0.8920
	3.4063	3.3352	3.4067	4.9780	-3.0122	2.0388
	0.7363	0.4099	0.6450	0.7569	-1.1600	1.5286
	0.0002	0	0	0	-3.5904	-0.1139
	0.4052	0	0.1656	0.4108	-3.5567	1.4760
	0.9861	0.9861	0.9861	1.0264	0.4767	1.3386
	1.1158	1.1157	1.0961	1.1405	-2.8417	2.3650
	1.9075	1.9075	1.9075	2.0138	0.8636	2.8822
$a_{\mathbf{R}} = 0.05$	2.5754	2.5754	2.5754	2.6972	0.1480	4.5320
$a_{\mathbf{K}} = 0.05$	0.8771	0.8771	0.8767	1.0473	-1.5992	0.2601
$n_{out} = 100$	0	0	0	0	-1.9773	-0.0852
	1.9448	1.9448	1.9444	2.0588	-3.8324	1.9536
	1.2698	1.2698	1.2698	1.2902	-0.6314	3.7346
	0	0	0	0	-2.6080	-0.7195
	1.1556	1.1556	1.1555	1.1930	-0.3513	1.6836

proached in less iterations than for others. This is linked to the sign of the difference matrix eigenvalues as illustrated by table 7.9.

7.3 Summary

Two iterative numerical optimization methods for the secrecy capacity value were presented in this chapter: the successive linear approximation (SLA) and min-max (mM).

The successive linear approximation method starts at an estimate of the optimal covariance matrix and uses first-order approximations to the secrecy capacity curve to approach its maximum value in an iterative way. At every iteration, a linear optimization problem is solved with CVX and the result is adjusted with backtracking. The value returned by SLA is a lower bound to the secrecy capacity. When C_s is concave, the algorithm approximates

Table 7.9. Variation with difference matrix sign.

Eigenvalue sign	Behavior
+ +	Equivalent to $\mathbf{W}_1 \succeq \mathbf{W}_2$. In this case, the problem is convex and successive approximation methods eventually reaches the global optimal
+ -	Does not converge every time / the apparent convergence point is below the optimal capacity value (as attested by MC). If the negative eigenvalue is larger, the numerical problem is, harder i.e. more iterations are needed to obtain a decent result
- -	Equivalent to $\mathbf{W}_1 \preceq \mathbf{W}_2$. This is the trivial case: $C_s = 0$

the global optimum. The successive linear approximation is adequate at all SNR values and the most efficient at low SNR. Testing with randomly selected channel matrices showed that this algorithm can converge as fast as in one step.

The successive approximation algorithm does not indicate how far the result is from the secrecy capacity. This justifies the practical value of the upper bound expression and the evaluation using a min-max algorithm.

The upper bound expression for the secrecy capacity, given in [29], [32], depends on two variables: $\mathbf{R}$, the transmitter covariance matrix and $\mathbf{K}$ the covariance matrix between the noise in the receiver and the eavesdropper. The algorithm presented here optimizes the expression independently over the two variables and approaches the secrecy capacity value in several iterations. The objective function is concave in $\mathbf{R}$ and convex in $\mathbf{K}$. The secrecy capacity is at the saddle-point. Instability in the final result occurs because optimization is not done simultaneously over $\mathbf{R}$ and $\mathbf{K}$.

Chapter 8

Conclusion

8.1 Thesis Summary

This thesis looks at the transmitter covariance matrix optimization for secure communications over Gaussian MIMO wiretap channels. Under the total power constraint, the secrecy capacity expression is written in terms of the channel gain matrices and the transmitter covariance matrix. It is not a concave function in $\mathbf{R}$ and it is hard to optimize in the general case.

Some insight about the properties of the optimal transmit covariance matrix can be gathered by looking at the few known solutions. In the literature, it has been shown that $\mathbf{R}^*$ makes use of all the available transmit power and its range is included in the positive directions of the difference matrix. It has also been shown that power allocation for secrecy is done only in the directions where the receiver is stronger than the eavesdropper. The optimal power allocation solution is somewhat similar to the well-known water-filling solution for MIMO but has also a number of differences.

Some of the known analytical results were implemented numerically and tested in a few special cases. This gave us confidence in the conclusions presented in the references. Also,

the results of these implementations served as standard by which the optimal values returned by the algorithms developed in this thesis were evaluated.

An attempt was made to solve the secrecy capacity optimization problem by using numerical optimization tools. These approximate the optimal value for the function without requiring mathematical optimization background.

Two optimization tools were used: YALMIP and CVX. Unfortunately, the results obtained with these tools were not accurate and, in some cases, grossly misleading. YALMIP returns notably different results for problems that are similar in nature and fails to find the optimal value for the secrecy capacity even in cases where it is convex. Due to the disciplined convex optimization rules, CVX does not accept the secrecy capacity optimization problem directly. Moreover, when a concave expression containing a $\ln\det$ term and some other terms is being optimized with CVX, the optimal value returned by the solver is lower than what was obtained using other methods.

Four different numerical optimization methods were developed to find the secrecy capacity: the Rank-adaptive Monte-Carlo algorithm, a genetic algorithm, successive linear approximation and $\min\max$.

The first two approaches are relying on an "ordered" random search among the possible solutions. Neither of them requires a special structure for the underlying optimization problem. Using Monte-Carlo like solutions was motivated in part by the fact that, in certain situations, a random blind search returns satisfactory results faster than generic numerical optimization tools. In particular, for systems having two transmit antennas ($m = 2$), finding the regular channel capacity[1] a random blind search can return a channel capacity value within 95% of the optimal, orders of magnitude faster than CVX.

The first algorithm is a Monte-Carlo optimization that takes into account the fact that

[1] In this case, there is no eavesdropper and the channel capacity expression in concave. The optimization problem can be solved correctly by CVX.

the rank of the optimal covariance matrix cannot be larger than the number of positive eigenvalues of the difference between the receiver and the eavesdropper channel matrices, $\mathbf{W}_1 - \mathbf{W}_2$. This focuses the search on a smaller set than the one used by the random blind search. Furthermore, since less random numbers are required, the algorithm is faster than the random blind search. The Rank-adaptive Monte-Carlo algorithm works particularly well in cases where the covariance matrix is expected to have a low rank e.g. when $\mathbf{W}_1 - \mathbf{W}_2$ has only one positive eigenvalue and when the size of the systems is relatively small, $m \in \{2, 4\}$.

The second algorithm developed was based on differential evolution. This is a random iterative process which emulates natural evolution techniques such as mutation, crossover and selection. The DE performance in different scenarios depends on the algorithm parameters. For example, larger systems require a larger population size to evolve from in order to be able to reach the optimal secrecy capacity value. If the goal is to find the value of the optimal secrecy capacity, DE can be more efficient than RaMC. However, if the computational time and resources are limited DE is ineffective.

Both the RaMC and the DE algorithms perform a random search among the possible transmit covariance matrices. Also, they rely on random number generation to build possible solutions. Because of this, these two methods return timely and accurate results when the size of the communication system is not too large, $m \in \{2, 4\}$.

The other optimization methods developed in this thesis are two iterative approaches. Both of them make use of an approximation to the secrecy capacity curve based on a Taylor series expansion. Rather than directly optimizing the secrecy capacity, these two methods solve a series of linear optimization problems.

The third optimization method developed in this thesis was a successive linear approximation (SLA) algorithm. This approach is not unlike Euler's method for solving differential equations. It starts at an estimate of the optimal covariance matrix and uses first-order approximations to the secrecy capacity curve to maximize it in an iterative way. Each iteration

consists in a linear optimization problem. The step size is adjusted using backtracking. In cases where the optimal covariance matrix is known to be rank 1 e.g. at low SNR or when $\mathbf{W}_1 - \mathbf{W}_2$ has only one positive eigenvalue, the SLA converges in one step.

The fourth numerical algorithm developed in this thesis, the min max algorithm, uses an alternate expression for the secrecy capacity and searches for a saddle point. The min max algorithm returns both an approximate value for the secrecy capacity and an approximate upper bound to C_s. The algorithm optimizes two variables: $\mathbf{R}$, the transmitter covariance matrix, and $\mathbf{K}$, the covariance matrix between the noise at the receiver and the eavesdropper. The objective function is concave in $\mathbf{R}$ and convex in $\mathbf{K}$ and SLA is used to optimize these matrices in turns. Some instability in the final result occurs because optimization is not done simultaneously. Finding a way to address this issue would be the subject of further research.

The four numerical optimization algorithms developed here can approximate the value of the secrecy capacity and that of the corresponding covariance matrix in the general case. Also they return timely results for smaller systems. Furthermore, these methods are efficient at finding the optimal covariance matrix at low SNR and in cases when the $\mathbf{R}$ is rank 1 even when the system size is large.

8.2 Further Research

The work started in this thesis could be expanded in several ways:

- The code written to implement the numerical optimization algorithms could be optimized for performance. A lower level coding language would perhaps be more appropriate if this code were to be used in a real-time system.
- Other numerical methods may be considered when optimizing the secrecy capacity expression. For instance, if C_s^u could be simultaneously optimized in $\mathbf{R}$ and $\mathbf{K}$ then some of the oscillations observed in the min max algorithm output may be eliminated.

- The results presented in this thesis were derived under ideal conditions. A possible research direction would be to investigate the impact of practical conditions such as uncertainty in the channel CSI or fading in the communications channels.

Chapter 9

Appendix – MATLAB Code

The computer ans software specs can be seen in Table .1.

Table .1. Computer and software specs.

Processor	Intel(R) Core(TM)2 QUad CPU Q6600 @ 2.40GHz 900 MHz
Installed memory(RAM)	5.00GB
System type	64-bit Operating System
Windows edition	Windows 7 Professional, Service Pack 1
MATLAB	Version 7.11.1.866 (R2010b) Service Pack 1
CVX	Version 2.1, Build 1058 (77726e8)

A The Weak Eavesdropper Solution

```
%% The weak eavesdropper case
% This a MATLAB implementation of the solution in "Rank-Defficient
% Solutions for Optimal Signaling over Secure MIMO Channels", S. Loyka and
% C.D. Charalambous"

% Function input:
% SNR : signal to noise ratio [dB];
% W1  : the main channel matrix (Tx-Rx)
% W2  : the eavesdropper channel matrix (Tx-Ex)
% tol : the allowed error for the total power constraint (%)

```

```
% Function output:
% C      : the approximate secrecy capacity
% R      : the covariance matrix (obtained analitycally)
% lambda : the value of the lagrange multiplier corresponding to the power
%          constraint
% p_star : power level where the trace approximation breaks

function [C R lambda p_star] = fun_weakW2(SNR, W1, W2, tol)
%% Function variables
snr = 10^(SNR/10);   % linear value of SNR
m   = max(size(W1)); % number of transmitter antennas
I   = eye(m);

%% Analytical approximation optimization
% Find the saturation point -  when the trace approximation is no longer
% valid. This point exists when the eavesdropper channel matrix is full rank.
if rank(W2)==m
   P_star_mat = I-W2^(1/2)*W1^(-1)*W2^(1/2);
   [U_P_star_mat lam_P_star_mat] = eig(P_star_mat);
   lam_P_star_mat = lam_P_star_mat.*(lam_P_star_mat>0);
   B = U_P_star_mat*lam_P_star_mat*U_P_star_mat';
   p_star = trace(W2^(-1)*B);
else
    p_star = Inf;
end

%%
% Use bisection to find λ
if snr≤p_star
   % defining min and max values for lambda
   lam_min = 0;
   lam_max = m/snr;
   %%
   % bisection starts
   for j=1:10^4
       lambda = (lam_max+lam_min)/2;
      [U_W2 lam_W2 ]= eig(W2);
       A = U_W2*(lambda*eye(2)+lam_W2)*U_W2';
       W1_til = A^(-1/2)*W1*A^(-1/2);
      [U_W1_til lam_W1_til] = eig(W1_til);
       lam_R = zeros(m);
       for i=1:m
           if lam_W1_til(i,i)>1
              lam_R(i,i) = 1-1/lam_W1_til(i,i);
           end
       end
       R_star = A^(-1/2)*U_W1_til*lam_R*U_W1_til'*A^(-1/2);
       pow = trace(R_star);
       if pow < snr*(1-tol)
          lam_max = lambda;
       else if pow > snr*(1+tol)
                lam_min = lambda;
            else
               break
```

```
            end
        end
    end
else
    lambda = 0;
   [U_W2 lam_W2] = eig(W2);
    A = U_W2*(lambda*eye(2)+lam_W2)*U_W2';
    W1_til = A^(-1/2)*W1*A^(-1/2);
   [U_W1_til lam_W1_til] = eig(W1_til);
    lam_R = zeros(m);
    for i=1:m
        if lam_W1_til(i,i)>1
           lam_R(i,i) = 1-1/lam_W1_til(i,i);
        end
    end
    R_star = A^(-1/2)*U_W1_til*lam_R*U_W1_til'*A^(-1/2);
end
% Using the value of the optimal covariance in the approximate formula
C = log(det(I+W1*R_star))-trace(W2*R_star);
R = R_star;
end
```

B Proportional Channel Matrices

```
%% Channel Matrices have the same eigenvectors
% This a MATLAB implementation of the solution in "Rank-Defficient
% Solutions for Optimal Signaling over Secure MIMO Channels", S. Loyka and
% C.D. Charalambous
% When the channel matrices have the same eigenvectors, the optimal
% covariance matrix has them too. To find the value of secrecy capacity all
% is left is to find the power distribution between the two eigenvalues

% Function input:
% SNR : signal to noise ratio [dB];
% W1  : the main channel matrix (Tx-Rx)
% W2  : the eavesdropper channel matrix (Tx-Ex)
% tol : the allowed error for the total power constraint (%)

% Function output:
% C      : the approximate secrecy capacity
% R      : the covariance matrix (obtained analitycally)
% lambda : the value of the lagrange multiplier corresponding to the power
%          constraint

function [C R lambda] = fun_sameEig(SNR, W1, W2, tol)
%% Function variables
snr  = 10^(SNR/10);   % linear value of SNR
m    = max(size(W1)); % number of transmitter antennas
I    = eye(m);
[U Lam1] = eig(W1);
Lam2     = diag(eig(W2));
```

```

%% Finding the power allocation
% defining min and max values for λ
lam_max = 1/snr;
lam_min = 0;
%%
% bisection starts
for j=1:10^4
    lambda = (lam_max+lam_min)/2;
    % Finding eigenvalues of covariance matrix
    for i=1:m
        LamR(i,i) = (Lam2(i,i)+Lam1(i,i))/(2*Lam2(i,i)*Lam1(i,i))*...
                    (sqrt(1+4*Lam2(i,i)*Lam1(i,i)/(Lam2(i,i)+Lam1(i,i))^2*...
                     max(0, (Lam1(i,i)-Lam2(i,i))/lambda-1))-1);
    end
    % forming the covarinace matrix
    R = U*LamR*U';
    % verifying the power constraint
    pow = trace(R);
    if abs((pow-snr)/snr)≥tol
        if pow < snr
           lam_max = lambda;
        else
           lam_min = lambda;
        end
    else
        break
    end
end

C = log(det(eye(m)+W1*R))-log(det(eye(m)+W2*R));
```

C Rank-adaptive Monte-Carlo

```
%% Rank-adaptive Monte-Carlo Algorithm
% Tests covariance matrices formed at random and only keeps the one that
% returns the highest secrecy rate. Takes into account that the covariance
% matrix is not always full rank: the random matrices  have any of the
% possible ranks of R

% Function input
% SNR    : signal to noise ratio [dB]
% W1     : main channel matrix (Tx-Rx)
% W2     : eavesdropper channel matrix (Tx-Ex)
% trials : number of tests do be done

% Function output:
% C   : estimated secrecy capacity at every trial
% R_s : estimated optimal covariance matrix

function [C R_s] = fun_MC_rank(SNR, W1, W2, trials)
```

```
%% Function variables
snr = 10^(SNR/10); % linear value of SNR
m   = max(size(W1)); % number of transmitter antennas
I   = eye(m);

%% Output
C = zeros(trials, 1);

%% Rank-Adaptive Monte-Carlo Algorithm
% maximal value for rankR
max_r = min(rank(W1),rank(W1-W2));

% first test
% construct a random covariance matrix
A = randn(m, 1);
B = A*A';
R = B/sum(diag(B));
% find the secrcy rate that it can give (0 if negative)
C(1) = max((log(det(eye(m)+snr*R*W1))-...
            log(det(eye(m)+snr*R*W2))), 0);
% the rest of the tests
for t=2:trials
    % decide the value of rankR for current trail
    r = mod(t, max_r)+1;
    % build a random covariance matrix
    A = randn(m, r);
    B = A*A';
    R = B/sum(diag(B));
    % compare its performance to the best up to this point and store the
    % result in C(trial)
    C(t) = max((log(det(eye(m)+snr*R*W1))-...
                log(det(eye(m)+snr*R*W2))), C(t-1));
    if C(t)>C(t-1)
       R_s = R; % keeps the value of the best covariance matrix so far
    end
end
end

%% Note
% If only the secrecy capacity is of interest, the if statement should be
% removed. This speeds-up the algorithm.
```

D Differential Evolution

```
%% Differential Evolution (DE)
% Uses differential evolution to find the optimal cvariance matrix. The
% search starts with NP random covariance matrices and, through mutation,
% crossover and selection, evolves toward the optimal.

% Function input
% SNR : signal to noise ratio [dB]
```

```
% W1  : main channel matrix (Tx-Rx)
% W2  : eavesdropper channel matrix (Tx-Ex)
% DE variables
% NP : number of members in initial population
% CR : crossover criteria (how many genes are mutated)
% F  : mutation criteria (difference between old and new population members)
% Gmax : how many generations before algorithm stop

% Function output:
% C : estimated secrecy capacity at every generation

function C = fun_DE(SNR, W1, W2, NP, CR, F, Gmax)
%% Function variables
snr = 10^(SNR/10);   % linear value of SNR
m   = max(size(W1)); % number of transmitter antennas
len = m^2;           % size of population members

%% Algorithm auxiliary variables
X = zeros(Gmax, NP, len); % current population members
V = zeros(Gmax, NP, len); % mutants
T = zeros(Gmax, NP, len); % trial entities (to be compared w/ current gen.)

%% Output
C = zeros(Gmax, 1);       % record best capacity at given generation

%% Differential Evolution
% form initial population
for np=1:NP
    X(1, np, 1:len) = randn(1,len);
end
% compute covariance matrices corresponding to each of the population
% members and find the best secrecy rate that can be acheived in the first
% generation
for np=1:NP
    A = reshape(X(1, np, 1:len), m, m);
    B = A*A';
    R = B/sum(diag(B));
    % maximum secrecy rate
    C(1) = max((log(det(eye(m)+snr*R*W1))-...
                log(det(eye(m)+snr*R*W2))), C(1));
end
%%
% loop through generations
for gen=2:Gmax
    % loop through all population members
    for np=1:NP
        %%
        % Mutation
        % choose 3 different members
        i = floor(rand()*NP)+1;
        j = floor(rand()*(NP-1))+1;
        j = (j≥i)+j;
        k = floor(rand()*(NP-2))+1;
        k = (k≥min(i,j))+k;
```

```
62          k = (k≥max(i,j))+k;
63          % form the mutant
64          V(gen, np, 1:len) = X(gen-1, k, 1:len)+...
65                              F*(X(gen-1, j, 1:len)-X(gen-1, i, 1:len));
66
67          %%
68          % Crossover
69          % decide which part of the vector stays
70          same = floor(rand()*len)+1;
71          for t=1:m*m
72              if rand()≤CR || t==same
73                 T(gen, np, t) = V(gen, np, t);
74              else
75                 T(gen, np, t) = X(gen-1, np, t);
76              end
77          end
78
79          %%
80          % Selection
81          % compare with target. Selection is done based on who can acheive
82          % the higher secrecy rate.
83          % target cost (performance of current population member)
84            A = reshape(X(gen-1, np, 1:len), m, m);
85            B = A*A';
86            R = B/sum(diag(B));
87            cost_target = log(det(eye(m)+snr*R*W1))-...
88                          log(det(eye(m)+snr*R*W2));
89          % trial cost (performance of the potential member)
90            A = reshape(T(gen, np, 1:len), m, m);
91            B = A*A';
92            R = B/sum(diag(B));
93            cost_trial = log(det(eye(m)+snr*R*W1))-...
94                              log(det(eye(m)+snr*R*W2));
95           % keep the better-performing vector
96            if cost_target<cost_trial
97               X(gen, np, 1:len) = T(gen, np, 1:len);
98            else
99               X(gen, np, 1:len) = X(gen-1, np, 1:len);
100           end
101     end
102     %%
103     % Best secrecy capacity in this generation
104     for np=1:NP
105         A = reshape(X(gen, np, 1:len), m, m);
106         B = A*A';
107         R = B/sum(diag(B));
108         C(gen) = max((log(det(eye(m)+snr*R*W1))-...
109                         log(det(eye(m)+snr*R*W2))), C(gen));
110     end
111 end
112
113 %% Note:
114 % The population members were written as vectors to speed up the function.
115 % Having matrices in the original populations works equally well.
```

E Successive Linear Approximation

```
%% Successive Linear Approximation Solution
% Starts at an estimate of the optimal covariance matrix and
% uses first-order approximations to the secrecy capacity curve to approach
% its maximum value in an iterative way.

% Function input:
% SNR : the signal to noise ratio [dB]
% W1  : the main channel matrix (Tx-Rx)
% W2  : the eavesdropper channel matrix (Tx-Ex)
% it  : the number of iterations for the algorithm to run

% Function output:
% C   : an estimate of the secrecy capacity
% R   : an estimate of the optimum covariance matrix

% Internal variables:
% cvx_precision  : determines the precision of every linear approximation
% alpha, beta    : constants relative to the backtracking portion
% a_eps          : determines the search area around starting-point

function [C, R] = fun_linApx(SNR, W1, W2, it)
%% Algorithm Variables
cvx_precision('best');
alpha = 0.5;
beta  = 0.5;
a_eps = 1;

%% Function variables
snr = 10^(SNR/10); % linear value of SNR
m   = max(size(W1)); % number of transmitter antennas
I   = eye(m);

%% Successive Linear Approximation Algorithm
% At each iteration, the secrecy capacity expression is linearized around
% the. starting-point and the linearization is then maximized.
%%
% The starting-point: a normalized identity matirx
R = I/m*snr;

for i = 1:it
    %%
    % Defining the search area for the current step
    Z = I/(I+W1*R)*W1 - I/(I+W2*R)*W2;
    eps = a_eps/max(norm(I/(I+W1*R)*W1), norm(I/(I+W2*R)*W2));

    %%
    % Begin linear approximation (optimal value for ΔR is
    % obtained through CVX). This indicates the maximization direction.
    cvx_begin quiet SDP
```

```
        variable del_R(m, m) symmetric
        del_C = trace(Z*del_R);
        maximize del_C
        subject to
                del_R + eps*eye(m) == semidefinite(m);
                R + del_R == semidefinite(m);
                trace(del_R) == 0;
    cvx_end
    %%
    % Begin backtracking. Backtracking is also an itarative algorithm. It
    % will indicate how much to move in the ΔR direction -
    % the step size.
    t = 1;
    % First backtracking iteration
    R_new = R+t*del_R;
    func_R_delR      = log(det(I+W1*R_new))-log(det(I+W2*R_new));
    fun_R_and_delR   = log(det(I+W1*R))-log(det(I+W2*R))+ ...
                       alpha*t*trace(Z*del_R);
    %%
    % The first condition is:
    % f(x+tΔx) < f(x)+αt∇fΔx
    % The second contition limits the number of iterations alowed for
    % backtracking.
    while (func_R_delR < fun_R_and_delR || t>10^-4)
           R_new = R+t*del_R;
           func_R_delR      = log(det(I+W1*R_new))-log(det(I+W2*R_new));
           fun_R_and_delR   = log(det(I+W1*R))-log(det(I+W2*R))+ ...
                              alpha*t*trace(Z*del_R);
           t = t*beta;
    end
    %%
    % The updated value for the starting point
    % R_k = R_{k-1} + tΔR
    R = R+t*del_R;
    % C(i) = log(det(I+W1*R))-log(det(I+W2*R));
end
% The secrrecy capacity
C = log(det(I+W1*R))-log(det(I+W2*R));
end

%% Note:
% The function can be modified to return the secrecy capacity approximate
% value at each iteration:
% Uncomment C(i) = log(det(I+W1*R))-log(det(I+W2*R));
% Comment   C = log(det(I+W1*R))-log(det(I+W2*R));
```

F min-max

```
%% The minmax algorithm
% Uses the upper-bound expression for secrecy capacity and estimates its
% optimum value by successivly optimizing over the R and
```

```
% K matrices.

% Function input
% SNR    : the signal to noise ratio
% H1     : the main channel gain matrix (Tx-Rx)
% H2     : the eavesdropper channel gain matrix (Tx-Ex)
% i_R    : number of iterations to perform over R
% i_K    : number of iterations to perfrm over K
% i_out  : number of iterations for the external loop
% maxmin : decides the order of the algorithm. 0 for minmax, 1 for maxmin

% Function output
% C_up  : an estimate of the secrecy capacity upper limit
% C_low : an extimate of the secrecy capacity lower limit
% R     : an estimate of the optimum covariance matrix
% K     : an estimate of the optimum K matrix

% Internal variables:
% cvx_precision  : determines the precision of every linear approximation
% alpha, beta    : constants relative to the backtracking portion
% a_eps_R          : determines the search area around starting-point for R
% a_eps_K          : determines the search area around starting-point for K

function [C_up C_low R K] =fun_minMax(SNR, H1, H2, i_R, i_K, i_out, maxmin)
%% Algorithm variables
cvx_precision('best');
alpha   = 0.5;
beta    = 0.5;
a_eps_R = 1;
a_eps_K = 1;

%% Function variables
snr = 10^(SNR/10); % linear value of SNR
H   = [H1; H2];
W1  = H1'*H1;      % the main channel matrix
W2  = H2'*H2;      % the eavesdropper channel matrix

m         = max(size(W1)); % number of transmitter antennas
[row col] = size(H);
I         = eye(m);
I2m       = eye(2*m);

%% minmax algorithm
% At each outter iteration the objective function is linearized and
% optimized over R and K successively.
%%
%The starting-point
K = eye(row);
R = I/m*snr;

for i = 1:i_out
    % each of the inner loopa is a successive approximation algorithm
    %%
    % Maximization over R
```

```
58      W = H'*(K\H); % Auxiliary variable used in the successive approximation
59      for i_max = 1:i_R
60          %%
61          % Defining the search area for the current step
62          ZR = (I+W*R)\W - (I+W2*R)\W2;
63          eps = a_eps_R/max(norm((I+W*R)\W), norm((I+W2*R)\W2));
64          %%
65          % Linear approximation over R begins
66          cvx_begin quiet SDP
67              variable del_R(m, m) symmetric
68              del_C = trace(ZR*del_R);
69              maximize del_C
70              subject to
71                  del_R + eps*I ≥ 0;
72                  R + del_R ≥0;
73                  trace(del_R) == 0;
74          cvx_end
75          %%
76          % First backtracking step.
77          t = 1;
78          R_new = R+t*del_R;
79          func_R_delR    = log(det(I+W*R_new))-log(det(I+W2*R_new));
80          fun_R_and_delR = log(det(I+W*R))-log(det(I+W2*R))+...
81                           alpha*t*trace(ZR*del_R);
82          %%
83          % The first condition is:
84          % f(x+tΔx) < f(x)+αt∇fΔx
85          % The second contition limits the number of iterations alowed for
86          % backtracking.
87          while func_R_delR < fun_R_and_delR
88                t = t*beta;
89                R_new = R+t*del_R;
90                func_R_delR    = log(det(I+W*R_new))-log(det(I+W2*R_new));
91                fun_R_and_delR = log(det(I+W*R))-log(det(I+W2*R))+ ...
92                                 alpha*t*trace(ZR*del_R);
93          end
94          %%
95          % This is the current optimal R to be used in the
96          % optimization over K
97          R = R+t*del_R;
98      end
99
100     %%
101     % if desired, reverse the minmax order
102     if (i==1 && maxmin==1)
103         R = I/m*snr;
104     end
105     % it is as if the first optimization over R did not happen
106
107     %%
108     % Minimization over K. Same comments apply for the
109     % successive approximation part.
110     Q = H*R*H'; % auxiliary variable used in the successive approximation
111     for i_min = 1:i_K
```

```
        ZK = I2m/K-I2m/(K+Q);
        eps = a_eps_K/max(norm(I2m/(K+Q)), norm(I2m/K));
        cvx_begin quiet SDP
            variable del_K(row, row) symmetric
            del_C = trace(ZK*del_K);
            maximize del_C
            subject to
                K+del_K ≥ (10^-3)*I2m
                del_K + eps*eye(row) ≥ 0
                del_K(1:row/2, 1:row/2)==zeros(m, m)
                del_K((row/2+1):row, (row/2+1):row)==zeros(m,m)
        cvx_end
        t = 1;
        K_new = K+t*del_K;
        func_K_delK    = log(det(K_new))-log(det(K_new+Q));
        fun_K_and_delK = log(det(K))-log(det(K+Q))+alpha*t*trace(ZK*del_K);
        while func_K_delK < fun_K_and_delK
              t = t*beta;
              K_new = K+t*del_K;
              func_K_delK    = log(det(K_new))-log(det(K_new+Q));
              fun_K_and_delK = log(det(K))-log(det(K+Q))+ ...
                               alpha*t*trace(ZK*del_K);
        end
        K = K+t*del_K;
    end
end
% The upper and lower bound estimations
C_up  = log(det(eye(row)+K\H*R*H'))- log(det(eye(m)+W2*R));
C_low = log(det(eye(m)+W1*R))- log(det(eye(m)+W2*R));
end
```

Bibliography

[1] M. Rumney, "LTE and MIMO Test Challenges," http://www.3gpp.org/news-events/press-clippings/1323-LTE-and-MIMO-test-challenges.

[2] A. J. Paulraj, D. A. Gore, R. U. Nabar, and H. Bölcskei, "An Overview of MIMO Communications—A Key to Gigabit Wireless," *Proceedings of the IEEE*, vol. 92, no. 2, pp. 198–218, Nov. 2004.

[3] D. Hucaby, *CCNA Wireless 640-722 Official Cert Guide*. Cisco Press, 2014.

[4] A. Carlisle and S. Lloyd, *Understanding PKI: Concepts, Standards, and Deployment Considerations*, 2nd ed. Addison-Wesley, 2002.

[5] Y. Liang and V. H. Poor, "Information Theoretic Security," *Foundations and Trends in Communication and Information Theory*, vol. 5, no. 4–5, pp. 355–580, Apr. 2008.

[6] W. Diffie and M. E. Hellman, "New Directions in Cryptography," *IEEE Transactions on Information Theory*, vol. 22, no. 6, pp. 644–654, Nov. 1976.

[7] M. Bloch and J. Barros, *Physical-Layer Security: From Information Theory to Security Engineering*. Cambridge University Press, 2011.

[8] A. D. Wyner, "The Wiretap Channel," *The Bell System Technical Journal*, vol. 24, no. 8, pp. 1355–1387, Oct. 1975.

[9] D. Tse and P. Viswanath, *Fundamentals of Wireless Communication*. Cambridge University Press, 2005.

[10] M. Grant and S. Boyd, "CVX: Matlab Software for Disciplined Convex Programming, version 2.1," http://cvxr.com/cvx, Mar. 2014.

[11] I. Recommendation, "200 (1994)— ISO/IEC 7498-1: 1994."

[12] J. M. Wozencraft and I. M. Jacobs, *Principles of Communication Engineering*. John Wiley & Sons, 1965.

[13] J. G. Proakis and M. Salehi, *Fundamentals of Communication Systems*. Pearson Prentice Hall, 2005.

[14] J. R. Barry, E. A. Lee, and D. G. Messerschmitt, *Digital Communication*, 3rd ed. Springer, 2003.

[15] E. Biglieri, *Coding for Wireless Channels*. Springer, 2005.

[16] J. G. Andrews, S. Buzzi, W. Choi, S. Hanly, A. Lozano, A. C. Soong, and J. C. Zhang, "What Will 5G Be?" *IEEE Journal on Selected Areas in Communications*, vol. 32, no. 6, pp. 1065–1082, Jun. 2014.

[17] T. Rappaport, S. Sun, R. Mayzus, and H. Zhao, "Millimeter Wave Mobile Communications for 5G Cellular: It Will Work!" *IEEE Access*, vol. 1, pp. 335–349, May 2013.

[18] T. Janevski, "5G Mobile Phone Concept," in *IEEE Consumer Communications and Networking Conference, CCNC 2009*, Las Vegas, NV, 10–13 Jan., pp. 1–2.

[19] GSM History: History of GSM, Mobile Networks, Vintage Mobiles, "5G," http://www.gsmhistory.com/5G/.

[20] P. W. Wolniansky, G. J. Foschini, G. Golden, and R. Valenzuela, "V-BLAST: An Architecture for Realizing Very High Data Rates over the Rich-scattering Wireless Channel,"

in *International Symposium on Signals, Systems, and Electronics ISSSE 1998*, 2 Oct., pp. 295–300.

[21] H. L. Van Trees, *Optimum Array Processing.* Wiley, 2002.

[22] D. Gesbert, M. Shafim, D.-s. Shiu, P. J. Smith, and A. Naguib, "From Theory to Practice: An Overview of MIMO Space-Time Coded Wireless Systems," *IEEE Journal on Selected Areas in Communications*, vol. 21, no. 3, pp. 281–302, Apr. 2003.

[23] S. Boyd and L. Vandenberghe, *Convex Optimization.* Cambridge University Press, 2004.

[24] C. Shannon, "Communication Theory of Secrecy Systems," *Bell System Technical Journal*, vol. 28, no. 4, pp. 656–715, Oct. 1949.

[25] T. M. Cover and J. A. Thomas, *Elements of Information Theory*, 2nd ed. Wiley-Interscience, 2005.

[26] I. Csiszár and J. Körner, "Broadcast Channels with Confidential Messages," *IEEE Transactions on Information Theory*, vol. 24, no. 3, pp. 339–348, May 1978.

[27] R. Pettai, *Noise in Receiving Systems.* Wiley, 1984.

[28] S. Leung-Yan-Cheong and M. Hellman, "The Gaussian Wiretap Channel," *IEEE Transactions on Information Theory*, vol. 24, no. 4, pp. 451–456, July 1978.

[29] A. Khisti, G. Wornell, A. Wiesel, and Y. Eldar, "On the Gaussian MIMO Wiretap Channel," in *IEEE International Symposium on Information Theory, ISIT 2007*, Nice, France, 24–29 Jun., pp. 2471–2475.

[30] F. Oggier and B. Hassibi, "The Secrecy Capacity of the MIMO Wiretap Channel," *IEEE Transactions on Information Theory*, vol. 57, no. 8, pp. 4961–4972, Aug. 2011.

[31] T. Liu and S. Shamai(Shitz), "A Note on the Secrecy Capacity of the Multiple-Antenna Wiretap Channel," *IEEE Transactions on Information Theory*, vol. 55, no. 6, pp. 2547–2553, June 2009.

[32] A. Khisti and G. W. Wornell, "Secure Transmission With Multiple Antennas-Part II: The MIMOME Wiretap Channel," *IEEE Transactions on Information Theory*, vol. 56, no. 11, pp. 5515–5532, Nov. 2010.

[33] R. Negi and S. Goel, "Secret Communication Using Artificial Noise," in *IEEE Vehicular Technology Conference, 2005*, vol. 62, no. 3, Dallas, TX, 25–28 Sept., p. 1906.

[34] R. Bustin, R. Liu, and H. Poor, "A MMSE Approach to the Secrecy Capacity of the MIMO Gaussian Wiretap Channel," *EURASIP Journal on Wireless Communications and Networking*, vol. 2009, article ID: 370970.

[35] S. Loyka and C. D. Charalambous, "On Optimal Signaling over Secure MIMO Channels," in *IEEE International Symposium on Information Theory, ISIT 2012*, Boston, MA, 1–6 July, pp. 443–447.

[36] S. Loyka and C. Charalambous, "Rank-Deficient Solutions for Optimal Signaling over Secure MIMO Channels," in *IEEE International Symposium on Information Theory, ISIT 2014*, Honolulu, HI, 29 June–4 July, pp. 201–205.

[37] P. K. Gopala, L. Lai, and H. El Gamal, "On the Secrecy Capacity of Fading Channels," *IEEE Transactions on Information Theory*, vol. 54, no. 10, pp. 4687–4698, Oct. 2008.

[38] H. Mahdavifar and A. Vardy, "Acheiving the Secrecy Capacity of Wiretap Channels Using Polar Codes," *IEEE Transactions on Information Theory*, vol. 57, no. 10, pp. 6428–6443, Oct. 2011.

[39] D. Klinc, J. Ha, S. W. McLaughlin, J. Barros, and B.-J. Kwak, "LDPC Codes for the Gaussian Wiretap Channel," *IEEE Transactions on Information Forensics and Security*, vol. 6, no. 3, pp. 532–540, Sept. 2011.

[40] H. Tyagi and A. Vardy, "Explicit Capacity-Achieving Coding Scheme For The Gaussian Wiretap Channel," *IEEE International Symposium on Information Theory, ISIT 2014*, pp. 956–960, 29 June–4 July.

[41] R. Liu and W. Trappe, Eds., *Securing Wireless Communications at the Physical Layer*. Springer, 2010.

[42] M. Van Dijk, "On a Special Class of Broadcast Channels with Confidential Messages," *IEEE Transactions on Information Theory*, vol. 43, no. 2, pp. 712–714, Mar. 1997.

[43] S. A. A. Fakoorian and A. L. Swindlehurst, "Full Rank Solutions for the MIMO Gaussian Wiretap Channel With an Average Power Constraint," *IEEE Transactions on Signal Processing*, vol. 61, no. 10, pp. 2620–2631, May 2013.

[44] M. C. Gursoy, "Secure Communication in the Low-SNR Regime: A Characterization of the Energy-secrecy Tradeoff," in *IEEE International Symposium on Information Theory, ISIT 2009*, Seoul, Korea, 28 June–3 July, pp. 2291–2295.

[45] Z. Li, W. Trappe, and R. Yates, "Secret Communication via Multi-antenna Transmission," in *Conference on Information Sciences and Systems, CISS 2007*, Baltimore, MD, 14–16 Mar., pp. 905–910.

[46] A. Khisti and G. W. Wornell, "Secure Transmission With Multiple Antennas-Part I: The MISOME Wiretap Channel," *IEEE Transactions on Information Theory*, vol. 56, no. 7, pp. 3088–3104, July 2010.

[47] S. Shafiee, N. Liu, and S. Ulukus, "Towards the Secrecy Capacity of the Gaussian MIMO Wire-Tap Channel: The 2-2-1 Channel," *IEEE Transactions on Information Theory*, vol. 55, no. 9, pp. 4033–4039, Sept. 2009.

[48] J. Löfberg, "YALMIP Wiki," http://users.isy.liu.se/johanl/yalmip/.

[49] ——, "YALMIP : A Toolbox for Modeling and Optimization in MATLAB." in *IEEE Computer Aided control Systems Design Conference, CACSD 2004*, Taipei, Taiwan, 2–4 Sept., pp. 284–289.

[50] R. Storn and K. Price, "Differential Evolution-A Simple and Efficient Heuristic for Global Optimization over Continuous Spaces," *Journal of Global Optimization*, vol. 11, no. 4, pp. 341–359, Dec. 1997.

[51] J. C. Spall, *Introduction to Stochastic Search and Optimization: Estimation, Simulation and Control*. John Wiley & Sons, 2003.

[52] R. Storn and Contributors, "Differential Evolution (DE)," http://www1.icsi.berkeley.edu/~storn/code.html.