Technische Universität München
TUM School of Computation, Information and Technology

Computable or Not Computable: Algorithmic Aspects of Channel Capacity

Andrea Leticia Grigorescu Vlass

Vollständiger Abdruck der von der TUM School of Computation, Information and Technology der Technischen Universität München zur Erlangung einer

Doktorin der Ingenieurwissenschaften (Dr.-Ing.)

genehmigten Dissertation.

Vorsitz: Prof. Dr.-Ing. Wolfgang Kellerer

Prüfende der Dissertation:

1. Prof. Dr.-Ing. Dr. rer. nat. Holger Boche
2. Prof. H. Vincent Poor, Ph.D.

Die Dissertation wurde am 15.04.2024 bei der Technischen Universität München eingereicht und durch die TUM School of Computation, Information and Technology am 26.06.2024 angenommen.

Zusammenfassung

Die Einführung der 6G-Technologie soll die Zugänglichkeit von Internet of Things (IoT) und Taktiles-Internet-Anwendungen erweitern, was eine hochzuverlässige, sichere und effiziente Infrastruktur erfordert, einschließlich der automatisierten Berechnung von Verbindungskapazitäten und Ressourcenzuweisungen. Diese Studie verwendet das Konzept der Turing-Maschinen, um die algorithmische Bewertung von Kommunikationsraten-Benchmarks für verschiedene Kanalmodelle zu untersuchen und die grundlegenden Leistungsgrenzen der digitalen Datenverarbeitung in diesem Kontext hervorzuheben.

Die Studie beginnt mit dem Compound Broadcast Channel with Confidential Messages. Sie untersucht die Stetigkeit der Secrecy-Kapazität in Bezug auf Systemparameter und bestätigt die Robustheit des Modells.

Bei der Betrachtung von Finite State Channels (FSCs) mit Feedback wird jedoch festgestellt, dass ihre Feedback-Kapazitätsfunktion nicht Banach-Mazur-berechenbar und folglich nicht Borel-Turing-berechenbar ist. Dies deutet darauf hin, dass entweder Erreichbarkeits- oder Umkehrergebnisse—oder möglicherweise beide—nicht algorithmisch berechnet werden können. Dies bedeutet, dass es für diese FSCs unmöglich ist, algorithmisch berechenbare scharfe obere und untere Grenzen für ihre Feedback-Kapazitäten zu bestimmen.

Dieses negative Ergebnis wirft eine grundlegende Frage auf: Was ist der einfachste Kommunikationskanal, dessen Kapazität nicht numerisch berechnet werden kann? Diese Frage führt zum Band-limited Additive Colored Gaussian Noise (ACGN) Kanal, einem Modell mit einer einfachen Struktur, aber komplexen Berechnungseigenschaften. Es wird gezeigt, dass einige ACGN-Kanäle mit berechenbaren Rauschleistungsdichtespektren nicht-berechenbare Kapazitäten haben und das Bestimmen berechenbarer scharfer oberer Grenzen für diese Kapazitäten unmöglich ist.

Ein wichtiges Ergebnis ist, dass, wenn das Rauschleistungsdichtespektrum streng positiv und berechenbar ist, die Kapazität von ACGN-Kanälen berechenbar wird und zu einem $\#\mathrm{P}_1$-vollständigen Problem wird, was auf ein höheres Komplexitätsniveau als NP_1-vollständige Probleme hinweist. Diese Komplexität betrifft ebenso die Ermittlung des kapazitätserreichenden Leistungsdichtespektrums, welches ebenfalls als $\#\mathrm{P}_1$-vollständig nachgewiesen wird.

Darüber hinaus untersucht diese Studie den Einfluss von berechenbaren konvexen Nebenbedingungen auf die Berechenbarkeit optimaler Lösungen in konvexen Optimierungsszenarien und zeigt, dass bestimmte konvexe Nebenbedingungen die genaue Berechnung optimaler Punkte verhindern, selbst bei streng konvexen Zielfunktionen.

Diese Arbeit hebt bedeutende rechnerische Herausforderungen und Komplexitäten beim Entwurf und der Bewertung von Kommunikationssystemen innerhalb der

zukünftigen 6G-Technologie hervor und schlägt zukünftige Forschungen zu alternativen Rechentechnologien für die Berechnung von Benchmarks für Kommunikationssysteme vor, wie beispielsweise Analog Computing.

Abstract

The advent of 6G technology is set to expand the accessibility of Internet of Things (IoT) and Tactile Internet applications, necessitating a highly reliable, secure, and efficient infrastructure, requiring the automated calculation of link capacities and resource allocations. This study employs the concept of Turing machines to delve into the algorithmic evaluation of communication rate benchmarks across various channel models, highlighting the fundamental performance limits of digital computing in this context.

The study begins with the compound broadcast channel with confidential messages (BCC), examining the continuity of its secrecy capacity region with respect to system parameters and affirming the model's robustness.

However, when considering finite state channels (FSCs) with feedback, it is revealed that their feedback capacity function is not Banach-Mazur computable, and consequently, not Borel-Turing computable. This indicates that either achievability or converse results—or possibly both—cannot be algorithmically computed. This implies that for these FSCs, it is impossible to algorithmically determine computable tight upper and lower bounds for their feedback capacities.

This negative result raises a fundamental question: What is the simplest communication channel whose capacity cannot be numerically computed? This inquiry leads to the band-limited additive colored Gaussian noise (ACGN) channel, a model with a straightforward structure but intricate computational properties. It is shown that some ACGN channels with computable noise power spectral densities (psd) have non-computable capacities, and deriving computable tight upper bounds for these capacities is unfeasible.

A significant finding is that when the noise psd is strictly positive and computable, the capacity of ACGN channels becomes computable and it is shown to be a $\#\mathrm{P}_1$-complete problem, indicating a higher complexity level than NP_1-complete problems. This complexity extends to finding the capacity-achieving distribution, also shown to be $\#\mathrm{P}_1$-complete.

Additionally, the study examines the impact of computable convex constraints on the computability of optimal solutions in convex optimization scenarios, revealing that certain constraints hinder the precise computation of optimal points, even with strictly convex objective functions.

This work highlights significant computational challenges and complexities in the design and evaluation of communication systems within the emerging 6G landscape, suggesting future research into alternative computing technologies for computing communication systems benchmarks, such as analog computing.

Contents

Nomenclature

Number sets

$\mathbb{N}$	set of natural numbers
$\mathbb{Q}$	set of rational numbers
$\mathbb{R}$	set of real numbers
$\mathbb{R}_{\geq 0}$	set of non-negative real numbers
$\mathcal{A}, \mathcal{B}, \ldots, \mathcal{X}$	finite sets unless stated otherwise
$\mathcal{X} \times \mathcal{Y}$	Cartesian product of the sets $\mathcal{X}$ and $\mathcal{Y}$

Other symbols

ln	natural logarithm
log	logarithm base 2

Probability

$\mathcal{P}(\mathcal{X})$	set of all probability distributions on a finite set $\mathcal{X}$
$\mathcal{P}(\mathcal{X} \times \mathcal{Y})$	set of all joint probability distributions on the set $\mathcal{X} \times \mathcal{Y}$
$\mathcal{P}(\mathcal{Y}\|\mathcal{X})$	set of stochastic matrices
P_X	probability distribution of the random variable X
$P_{Y\|X}$	conditional probability distribution of the random variable Y given X
X, Y, Z, U, V	random variables over finite sets

1 Introduction

1.1 Motivation

5G networks have laid the groundwork for Internet of Things (IoT) in industrial settings. However, the next generation 6G networks are expected to provide the basis to massively expand IoT and Tactile Internet usage among individual consumers in daily life. This expansion promises to improve the quality of life through new and innovative applications. However, this will also lead to a significant rise in data traffic and introduce the need for managing both physical and virtual objects over the network. As a result, 6G infrastructure will have to provide reliable, scalable, and secure communication with substantially higher throughput than 5G, and with more stringent latency requirements. The advances in wireless network infrastructure for 6G will have to enable sensing and coordination of control functions [1, 2]. Yet, as we push the boundaries of these technologies, we need to account for potential risks. For instance, without adequate security protection, enhanced sensory capabilities could be misused. Thus, 6G needs to be designed with robust security measures that also comply with legal and social standards, such as the General Data Protection Regulation (GDPR). Therefore, it is crucial to build a native, trustworthy architecture for 6G networks, addressing concerns that go beyond traditional security and reliability.

Trustworthiness in 6G is built on following key pillars: privacy, security, integrity, resilience, reliability, availability, accountability, authenticity, and device independence. These principles are essential for establishing a 6G ecosystem where users can rely on network services without fearing data breaches, service interruptions, or unauthorized surveillance. This work focuses on the theoretical foundations of reliability, integrity, and accountability.

Information theory provides the theoretical tools for achieving both high data throughput and reliable communication over wireless channels. It offers a mathematical framework to analyze communication scenarios, evaluate their characteristics, and establish benchmarks for reliable transmission rates by taking into account the specific attributes of the channel, as well as noise and power limitations in transceiver devices. These benchmarks are key to designing communication systems that employ close to optimal coding strategies, allowing for high transmission rates with minimal errors.

Evaluating a code's performance, i.e., how closely it aligns with the theoretical benchmark or channel capacity, is essential in determining its effectiveness. This will become important for 6G networks, which are expected to support critical applications in fields such as mobile robotics and autonomous driving. This includes developing methods for the automated evaluation of system performance. It is therefore essential to formulate

standards in a machine-readable format, and to guarantee that performance metrics can be calculated by digital computers.

It is often assumed that performance functions, especially those involving entropic quantities such as capacity expressions, are computable. In 1967, methods for constructing both upper and lower bounds for channel reliability functions were introduced in [3]. These techniques were specifically designed to enable the computation of these bounds using digital computers. In 1972, an algorithm to compute the capacity of arbitrary discrete memoryless channels (DMCs) was independently presented in [4] and [5]. In [5], an analogous algorithm was proposed to compute the rate distortion trade-off of lossy source compression. Typically, channel capacity is expressed through mutual information formulas. It is worth noting that even for the binary symmetric channel (BSC) with a rational crossover probability, the capacity turns out to be a transcendental number [6]. This implies that a precise calculation is not possible, since the computation has to stop after a finite number of steps. Only a suitable approximation of it can be calculated.

Surprisingly, whether capacity functions can be calculated by digital computers, especially for channels modeling complex communication scenarios such as multi-user environments or channels with memory, remains a relatively unexplored area in information theory. When channels are modeled under such complex conditions, defining their capacity becomes difficult, even for discrete channels with finite input and output alphabets. Take finite state channels (FSCs), for example, which incorporate memory effects, allowing the current output to be influenced by the channel's state and indirectly by past inputs and outputs. This characteristic of FSCs is particularly relevant for modeling intersymbol interference (ISI). In [7], the capacity of indecomposable FSC has been determined, showing that it is described by a multi-letter expression, which represents the limit of a series of optimization problems. This complexity persists even for channels with binary inputs, binary outputs, and binary states, making the task of computing capacity particularly challenging.

The question of whether it is possible to compute such multi-letter expressions using today's technologies has only recently been raised. The algorithmic computability properties of channel capacities have been studied for channel models, including FSCs [8], correlation-assisted DMCs [9], and compound channels [10]. For all of these channels, it has been demonstrated that their capacities are not generally computable functions, due to their complicated descriptions.

The primary focus of this work is on examining the analytical attributes, computational feasibility, and complexity of multi-letter capacity expressions in various communication settings. Furthermore, the study extends to the exploration of capacities expressed by integrals, investigating their algorithmic computability and computational complexity. In addition to these subjects, this research addresses the challenge of algorithmically solving convex optimization problems, which are pivotal in numerous capacity computation efforts.

To address algorithmic computability of channel capacities, we use the concept of a *Turing machine* [11, 12], which is a mathematical model of an abstract machine that manipulates symbols on a strip of tape according to certain given rules. Any algorithm

can be translated into a sequence of steps that can be executed by a Turing machine and therefore, Turing machines provide a simple and very powerful model of computation. Turing machines have no limitations on computational complexity, computing capacity or storage, and execute programs completely error-free. Accordingly, they provide fundamental performance limits for today's digital computers. Turing machines account for all those problems and tasks that are algorithmically computable on a classical (i.e., non-quantum) machine. They are equivalent to the von Neumann-architecture without hardware limitations and the theory of recursive functions [13, 14, 15, 16].

1.2 Contributions and Outline

In Chapter 2, we introduce fundamental concepts in information theory, convex optimization, and computability theory.

In Chapter 3, we delve into how the capacity region of the Broadcast Channel with Confidential Messages (BCC) responds to changes in its parameters, focusing on the uncertainty set. Our objective is to determine how variations in the uncertainty set influence the capacity region, an essential consideration for designing secure and efficient communication systems. We demonstrate that the BCC capacity region is a continuous function of the uncertainty set. Parts of this chapter were published in [17] and [18].

In Chapter 4, we address questions regarding the computability of the feedback capacity of FSCs. Our study reveals that the feedback capacity of such channels is not Banach-Mazur computability [12], which is the weakest form of computability and it further implies that it is not Borel-Turing computable. In other words, it is not possible to find a universal algorithm that takes the parameters describing an FSC as input and returns its feedback capacity.

Moreover, we show the impossibility of algorithmically approximating the feedback capacity of FSCs with any computable function within any desired margin of error. As a consequence, we show that it is impossible to find computable arbitrarily tight upper and lower bounds on the feedback capacity of FSCs. Furthermore, we show that it is not possible to express the feedback capacity of FSCs by a finite-letter entropic expression. Parts of this chapter were published in [19] and [20].

In Chapter 5, we address the task of computationally determining the capacity of band-limited Additive Colored Gaussian Noise (ACGN) channels. We show that the capacity of certain band-limited ACGN channels turns out to be a non-computable number, indicating that there is no universal algorithm capable of accurately estimating the capacity based on the channel's parameters. Furthermore, we show that it is impossible to find algorithmically computable sharp upper bounds for the capacity of these channels. Lastly, we illustrate that even when the power constraints are loosened, the inherent computational challenges in determining the capacity of band-limited ACGN channels do not diminish. Parts of this chapter were published in [21].

In Chapter 6, we demonstrate that for band-limited ACGN channels with strictly positive noise power spectral densities (psd), the channel capacity becomes computable. We further establish that computing this capacity as a specific numerical value falls

within the $\#\mathrm{P}_1$ complexity class—a class that consists of problems counting the number of solutions of a problem that can be verified by a Turing machine in polynomial time. Additionally, we explore the computational complexity involved in determining the optimal input psd that maximizes the channel capacity. Our analysis reveals that both approximating the capacity and computing the optimal psd are $\#\mathrm{P}_1$-complete tasks. Parts of this chapter were published in [22] and [23].

In Chapter 7, we explore the computability of optimal points in constrained convex optimization problems. We demonstrate that for certain constraint functions, the optimal points are non-computable numbers across all strictly convex and continuously computable objective functions. Additionally, we reveal that for any such strictly convex function, the optimal point of its associated Lagrangian dual problem, given these constraints, is also non-computable. Parts of this chapter will be published in [24].

2 Preliminaries

This chapter is designed to lay the groundwork essential for a comprehensive understanding of the analyses and contributions presented in this dissertation.

Subsequently, Section 2.1 explores the core principles of information theory, focusing on the properties of information theory measures used in Chapter 3 and Chapter 4.

Next, Section 2.2 introduces the convex optimization framework. It provides the foundational concepts that will be further analyzed in Chapter 7.

Lastly, Section 2.3 explores computability theory by detailing fundamental concepts of computable numbers and functions. This section also introduces computational complexity and complexity classes, and examines the computational complexity attributes of integration. The concepts explained here will be instrumental in assessing the computational aspects studied in Chapters 4, 5, 6, and 7.

2.1 Information Theory

In this section, we introduce the foundational tools of information theory, with a particular focus on the concepts of causal conditioning and directed information. These concepts are crucial for characterizing the capacity of discrete channels and were introduced and applied in [25], [26], [27], and [28].

Definition 1. *The* entropy *of the random variable X, taking values in a discrete and finite set $\mathcal{X}$ with probability distribution $P_X \in \mathcal{P}(\mathcal{X})$, is defined by*

$$H(X) = \sum_{x \in supp(P_X)} -P_X(x) \log P_X(x)$$

with $supp(P_X) \coloneqq \{x \in \mathcal{X} \colon P_X(x) > 0\}$.

Consider a binary alphabet $\mathcal{X} = \{0, 1\}$. The entropy of the random variable X taking values in $\mathcal{X}$ with probability distribution $P_X(0) = p$ is called the *binary entropy function* and is denoted by

$$H_2(p) = -p \log p - (1-p) \log(1-p).$$

Definition 2. *Consider a joint distribution $P_{XY}(\cdot)$ where the random variable Y takes values in a discrete and finite alphabet $\mathcal{Y}$ and X takes values in a discrete and finite alphabet $\mathcal{X}$. The* conditional entropy *of X given Y is defined by*

$$H(X|Y) = \sum_{(x,y) \in supp(P_{XY})} -P_{XY}(x,y) \log P_{X|Y}(x|y).$$

Definition 3. *The* mutual information *between two random variable X and Y with respective discrete and finite alphabets $\mathcal{X}$ and $\mathcal{Y}$ is defined as*

$$I(X;Y) = \sum_{(x,y)\in supp(P_{XY})} P_{XY}(x,y) \log \frac{P_{XY}(x,y)}{P_X(x)P_Y(y)}.$$

Next we introduce the concepts of causally conditioned probability distributions and directed information.

Definition 4. *The probability distribution of the sequences $x^n \in \mathcal{X}^n$* causally conditioned *on the sequence $y^n \in \mathcal{Y}^n$ is given by*

$$p(x^N \| y^N) = \prod_{n=1}^{N} p(x_n | y^n, x^{n-1}). \tag{2.1}$$

A special case of Definition 4 used in the context of the FSC with feedback is

$$p(x^N \| y^{N-1}) = \prod_{n=1}^{N} p(x_n | x^{n-1}, y^{n-1}). \tag{2.2}$$

Definition 5. *The* directed information *from a sequence X^N to a sequence Y^N is defined by*

$$\begin{aligned} I(X^N \to Y^N) &= \sum_{n=1}^{N} I(X^n; Y_n | Y^{n-1}) \\ &= \sum_{n=1}^{N} H(Y^n | Y^{n-1}) - H(Y_n | X^n Y^{n-1}). \end{aligned}$$

An important property of the directed information that we will use in our work, is that it can be upper bounded. The upper bound of the directed information from X^n to Y^n is presented in the following lemma.

Lemma 1. *[26, Theorem 2] If X^N and Y^N are the input and output sequences respectively of a DMC, then*

$$I(X^N \to Y^N) \leq \sum_{n=1}^{N} I(X_n; Y_n) \tag{2.3}$$

with equality if and only if $Y_1, Y_2, \ldots, Y_n$ are statistically independent.

2.2 Fundamentals of Convex Optimization

Optimization problems are part of the core problems in engineering, economics and other fields. These problems involve the task of finding the best possible solution, often under certain constraints and resource limitations. Whether it is designing efficient structures, allocating resources effectively, or maximizing profits, optimization plays an indispensable role in building today's infrastructure.

In this section, we introduce the main concepts and properties of optimization problems, with a particular focus on convex optimization problems. The definitions of this section are taken from [29].

2.2.1 Convex and Concave Functions

In this subsection we introduce some basic definitions for mathematical optimization. Mathematical optimization deals with finding the best solution (maximum or minimum) to a problem from a set of possible solutions. In particular, we focus on convex optimization. A convex optimization problem has a convex objective function and is subject to convex constraint functions on the function's domain. The goal is to minimize the convex objective function within the constraints. The constraints on the function's domain build a convex set. For this we first introduce the notion of convex set.

Definition 6. *A set $\mathcal{C} \subset \mathbb{R}^n$ is* convex *if every segment between two points in $\mathcal{C}$ lies in $\mathcal{C}$, i.e., if for any $x_1, x_2 \in \mathcal{C}$ and $0 \leq \lambda \leq 1$, we have*

$$\lambda x_1 + (1 - \lambda)x_2 \in \mathcal{C}. \tag{2.4}$$

In other words, $\mathcal{C}$ is convex if it contains the convex combination of any two points in $\mathcal{C}$.

Definition 7. *Let $\mathcal{C} \subset \mathbb{R}^n$. A function $f \in \mathcal{C} \to \mathbb{R}$ is* convex *if $\mathcal{C}$ is a convex set and if for all $x, y \in \mathcal{C}$ and λ with $0 \geq \lambda \geq 1$, we have*

$$f(\lambda x + (1 - \lambda)y) \leq \lambda f(x) + (1 - \lambda)f(y). \tag{2.5}$$

Minimization problems, such as cost minimization and and loss minimization can be ideally modeled by convex optimization problems where the objective function is a convex function. Convex functions have some key important properties, which makes them easier to solve, i.e., to minimize, compared to non-convex functions. One important property is that any local minimum of a convex function is also a global minimum. While the global minimum of a convex function may not be unique, if a function is strictly convex, it will always have a unique minimum. Next, we introduce the concept of strictly convex functions.

Definition 8. *Let $\mathcal{C} \subset \mathbb{R}^n$. A function $f \in \mathcal{C} \to \mathbb{R}$ is* strictly convex *if $\mathcal{C}$ is a convex set and if for all $x, y \in \mathcal{C}$ and λ with $0 \leq \lambda \leq 1$, we have*

$$f(\lambda x + (1 - \lambda)y) < \lambda f(x) + (1 - \lambda)f(y). \tag{2.6}$$

Maximization problems, such as profit maximization and efficiency maximization on the other hand, can be ideally modeled by optimization problems whose objective functions are concave. A significant property of concave functions is that any local maximum of the function is also a global maximum.

Definition 9. *Let $\mathcal{C} \subset \mathbb{R}^n$. A function $f \in \mathcal{C} \to \mathbb{R}$ is* concave *if $-f$ is convex. Furthermore, f is said to be* strictly concave *if $-f$ is strictly convex.*

2.2.2 Convex Optimization Problems

A *convex optimization problem* is of the form

$$\begin{aligned} \text{minimize} \quad & f(x) \\ \text{subject to} \quad & \varphi_i(x) \leq 0, \quad i = 1, \ldots, m, \\ & \gamma_i(x) = 0, \quad i = 1, \ldots, p. \end{aligned} \tag{2.7}$$

It describes the problem of finding an x that minimizes $f(x)$ among all x that satisfy the conditions $\varphi_i(x) \leq 0$, $i = \{1, \ldots, m\}$ and $\gamma_j(x) = 0$, $j = \{1, \ldots, p\}$.

The *objective function* $f\colon \mathbb{R}^n \to \mathbb{R}$ is convex. The *optimization variable* is $x \in \mathbb{R}^n$. The *inequality constraints* $\varphi_i(x) \leq 0$, for convex functions $\varphi_i\colon \mathbb{R}^n \to \mathbb{R}$ with $i = \{1, \ldots, m\}$, along with the *equality constraints* $\gamma_j(x) = 0$, for affine functions $\gamma_j\colon \mathbb{R}^n \to \mathbb{R}$ with $j = \{1, \ldots, p\}$, build the *feasible set* of the problem;

$$\mathcal{D} = \bigcap_{i=1}^{m} \mathbf{dom}\varphi_i \cap \bigcap_{j=1}^{p} \mathbf{dom}\gamma_j$$

.

We say $x \in \mathbb{R}^n$ is a *feasible point* if $x \in \mathcal{D}$.

The *optimal value* of the problem is denoted by $\mathrm{OptV}(f, \varphi_1, \ldots, \varphi_m, \gamma_1, \ldots, \gamma_p)$: $\mathrm{OptV}(f, \varphi_1, \ldots, \varphi_m, \gamma_1, \ldots, \gamma_p) = \inf\{f(x) | \quad \varphi(x) \leq 0, \quad i = \{1, \ldots, m\}, \quad \gamma_j(x) = 0, \quad j = \{1, \ldots, p\}\}$ and the point x^* is called the optimal point, if x^* is feasible and $f(x^*) = \mathrm{OptV}(f, \varphi_1, \ldots, \varphi_m, \gamma_1, \ldots, \gamma_p)$.

2.2.3 Duality

An optimization problem can be viewed from a different perspective. For this, we use the duality principle. A constrained optimization problem can be reformulated into a dual problem. If the original optimization problem consists of a minimization problem, then the dual problem is formulated as a maximization problem.

Both problems are closely related since the optimal solution of one of the problems gives a bound on the optimal solution of the other. The relationship between the original and the dual problem can be categorized into two categories: strong and weak duality. We have strong duality when the optimal solution of the dual problem equals the optimal solution of the original problem. Weak duality holds when the optimal solution of the dual problem differs from the optimal solution of the original problem.

Formulating an optimization problem into its dual problem is often very attractive since the dual problem can be easier to solve. Consequently, it might be easier to find the solution or a bound on the solution of the original optimization problem.

Lagrangian duality consists of reformulating a constrained optimization problem by augmenting its objective function f with a weighted sum of the constraint functions φ_i for $i \in \{1, \ldots, m\}$ and γ_j for $j \in \{1, \ldots, p\}$. For this we introduce the Lagrangian function.

Definition 10. *The* Lagrangian function $\mathcal{L}\colon \mathbb{R}^n \times \mathbb{R}^m \times \mathbb{R}^p \to \mathbb{R}$ *associated with the problem* (2.7) *is given by*

$$\mathcal{L}(x,u,v) = f(x) + \sum_{i=1}^{m} u_i \varphi_i(x) + \sum_{j=1}^{p} v_j \gamma_j(x).$$

where $u_1,\ldots,u_m$ *and* $v_1,\ldots,v_p$ *are real scalars. We refer to* u_i *as the* Lagrange multiplier *associated with the* i*th inequality constraint* $\varphi_i(x) \leq 0$ *and to* v_j *associated with the* j*th equality* γ_j*. The vectors* u *and* v *are called the* dual variables *or* Lagrange multiplier vectors *associated with the problem* (2.7)*.*

Next we introduce the Lagrangian dual function in order to formulate the Lagrangian dual problem.

Definition 11. *The* Lagrangian dual function *is a function of the dual variables* u *and* v *defined as the minimum value of the Lagrangian function over* x *for* $u \in \mathbb{R}^m$ *and* $v \in \mathbb{R}^p$*,*

$$g(u,v) = \inf_{x \in \mathcal{D}} \mathcal{L}(x,u,v).$$

Remark 1. *Note that the Lagrangian dual function is* $-\infty$*, when the Lagrangian is unbounded from below in* x*.*

Remark 2. *Note that the Lagrangian dual function is the pointwise infimum of a family of affine functions of* (u,v) *and hence, the Lagrangian dual function is concave, even when the objective function of the original problem is not convex.*

Next we introduce the Lagrangian dual problem.

Definition 12. *The* Lagrangian dual problem *associated with the problem* (2.7) *is defined as*

$$\max g(u,v) \qquad \textit{subject to } u \geq 0 \tag{2.8}$$

where $g(u,v)$ *is the Lagrangian dual function associated with* (2.7)*.*

Remark 3. *The Lagrangian dual problem is a convex optimization problem, since its objective function is the Lagrangian dual function, which is a convex function.*

Let $\mathrm{OptV}(f,\varphi_1,\ldots,\varphi_m,\gamma_1,\ldots,\gamma_p)$ be the optimal value of a primal problem and $\mathrm{OptV}(g,u_1,\ldots,u_m,v_1,\ldots,v_p)$ be the optimal value of its corresponding Lagrangian dual problem. The difference $\mathrm{OptV}(f,\varphi_1,\ldots,\varphi_m,\gamma_1,\ldots,\gamma_p) - \mathrm{OptV}(g,u_1,\ldots,u_m,v_1,\ldots,v_p)$ is called the *duality gap*. The duality gap is always non-negative, i.e., $\mathrm{OptV}(f,\varphi_1,\ldots,\varphi_m,\gamma_1,\ldots,\gamma_p) \geq \mathrm{OptV}(g,u_1,\ldots,u_m,v_1,\ldots,v_p)$. This property is called *weak duality*. If the duality gap is zero, i.e., $\mathrm{OptV}(f,\varphi_1,\ldots,\varphi_m,\gamma_1,\ldots,\gamma_p) = \mathrm{OptV}(g,u_1,\ldots,u_m,v_1,\ldots,v_p)$, we say that we have *strong duality*. In this case, the solution of the dual problem is also the solution of the primal problem.

2.2.4 Optimality conditions

In this subsection we introduce the necessary and sufficient Karush-Kuhn-Tucker (KKT) conditions for the optimality of a solution to a constrained optimization problem in the form of (2.7).

Assume that the functions $f, \varphi_1, \ldots, \varphi_m, \gamma_1, \ldots, \gamma_p$ are differentiable. The *KKT conditions* are

- Stationarity:
$$0 \in \partial f(x) + \sum_{i=1}^{m} u_i \partial \varphi_i(x) + \sum_{j=1}^{p} v_j \partial \gamma_j(x).$$
- Complementary slackness:
$$u_i \varphi_i(x) = 0 \quad \forall i \in \{1, \ldots, m\}.$$
- Primal feasibility:
$$\begin{aligned} \varphi_i(x) &\leq 0 \quad \forall i \in \{1, \ldots, m\} \\ \gamma_j(x) &= 0 \quad \forall j \in \{1, \ldots, p\}. \end{aligned}$$
- Dual feasibility:
$$u_i \geq 0 \quad \forall i \in \{1, \ldots, m\}.$$

Theorem 1 ([30]). *[Necessary Condition] If the optimization problem has strong duality, i.e., zero duality gap, then for any primal solution x^* and any dual solution u^* and v^*, the pair (x^*, u^*, v^*) must satisfy the KKT conditions.*

Theorem 2 ([31]). *[Sufficient Condition] If there exists a solution x^* to the primal problem and a solution (u^*, v^*) to the dual problem, such that (x^*, u^*, v^*) satisfy the KKT conditions, then the problem pair has strong duality and (x^*, u^*, v^*) is a solution pair to the primal and dual problems.*

In Chapter 7, we explore algorithmically solving convex optimization problems that have strictly convex objective functions. For this purpose, we introduce an optimization problem and two lemma that provides insights into the Lagrangian dual function associated with this problem.

We consider the following optimization problem:

$$\begin{aligned} \text{minimize} \quad & f(x) \\ \text{subject to} \quad & \varphi(x) \leq \lambda \end{aligned}$$

with $f \colon \mathcal{D} \to \mathbb{R}$ and $\varphi \colon \mathcal{D} \to \mathbb{R}$ where $\mathcal{D} = [a, b]$ with $a, b \in \mathbb{R}$.

Lemma 2. *Let $f\colon \mathcal{D} \to \mathbb{R}$ be a strictly convex function, $\varphi\colon \mathcal{D} \to \mathbb{R}$ be a convex function and $\lambda \in \mathbb{R}$. Let $\mathcal{L}\colon \mathcal{D} \times \mathbb{R} \to \mathbb{R}$ be the Lagrangian function associated with the optimization problem f, φ, λ and $g\colon \mathbb{R} \to \mathbb{R}$ be the Lagrangian dual function. Let $\hat{x}(u) \in \mathcal{D}$ be such that*

$$g(u) = \mathcal{L}(\hat{x}(u), u).$$

Then for every $u_1 \neq u_2$ with $u_1, u_2 > 0$ it holds that

$$\hat{x}(u_1) \neq \hat{x}(u_2).$$

Proof. Let f and φ be fixed. We consider the Largangian function

$$\mathcal{L}(x, u) = f(x) + u(\varphi(x) - 1).$$

For $x \geq x_*$ and $x \in [x_*, \tilde{x}]$, we have

$$\mathcal{L}(x, u) = f(x).$$

This implies that $\mathcal{L}(x, u)$ is monotonically increasing on the interval $[x, \tilde{x}]$.

In the interval $[\tilde{x}, b]$ we have that $\varphi(\cdot) - 1$ is monotonically increasing. Since $u \geq 0$, we have that $\mathcal{L}(x, u)$ is monotonically increasing on the interval $[x_*, b]$.

Since $\mathcal{L}(x, u)$ is strictly convex with respect to x, then there exists exactly one $x(u) \in [0, x_*]$ with

$$\mathcal{L}(x(u), u) = \min_{x \in [0,b]} \mathcal{L}(x, u) = g(u)$$

for $u \geq 0$. Furthermore, f and φ are continuously differentiable functions. We then have

$$\frac{\partial}{\partial x}\mathcal{L}(x, u) = f'(x) + u\varphi'(x).$$

φ is monotonically decreasing on the interval $[0, x_*]$. Thus $\varphi'(0) \leq 0$. Furthermore, we have $f'(0) \geq 0$. For the case where $f'(0) = 0$, we have for all $u > 0$ that $\frac{d}{dx}\mathcal{L}(x, u)\Big|_{x=0} < 0$. This implies that there is an interval $[0, \widehat{x}]$ where $\mathcal{L}(x, u)$ is monotonically decreasing.

For the case where $f'(0) > 0$, for every $u > \widehat{u} = -\frac{f'(0)}{\varphi'(0)}$ there is an interval where $\mathcal{L}(x, u)$ is monotonically decreasing. This way we have that for all $u > \hat{u}$ $x(u) \in (0, x_*]$.

Let $u_1, u_2 > \hat{u}$ be arbitrary. Then, since $\mathcal{L}(x, u_l)$ with $l = 1, 2$ is strictly convex with respect to x, we have that

$$\frac{\partial}{\partial x}\mathcal{L}(x, u_l) = 0 \iff x = x(u_l), l = 1, 2.$$

This result holds because the minimum points lie within the interior of the interval $[0, b]$ and are also global minima.

It holds that

$$\mathcal{L}(x, u_2) = \mathcal{L}(x, u_1) + (u_2 - u_1)(\varphi(x) - 1).$$

Furthermore, since $\varphi'(x_*) = 0$, and f is monotonically increasing on $[0, b]$, we have $x(u_l) < x_*$. It then holds that

$$\begin{aligned}\frac{\partial}{\partial x}\mathcal{L}(x, u_2)|_{x=x(u_1)} &= \frac{\partial}{\partial x}\mathcal{L}(x, u_2)|_{x=x(u_1)} + (u_2 - u_1)\varphi'(x)|_{x=x(u_1)} \\ &= (u_2 - u_1)\varphi'(x)|_{x=x(u_1)} < 0\end{aligned}$$

since φ is strictly monotonically decreasing on $[0, x_*]$.

We then have that $x(u_2) \neq x(u_1)$. □

Lemma 3. *Let $f\colon \mathcal{D} \to \mathbb{R}$ be a strictly convex function, $\varphi\colon \mathcal{D} \to \mathbb{R}$ be a convex function and $\lambda \in \mathbb{R}$. Let $\mathcal{L}\colon \mathcal{D} \times \mathbb{R} \to \mathbb{R}$ be the Lagrangian function associated with the optimization problem f, φ, λ and $g\colon \mathbb{R} \to \mathbb{R}$ be the Lagrangian dual function. Then it holds that*

$$C_{\mathcal{D}} = \sup_{u \geq 0} g(u) = OptV(f, \varphi, \lambda).$$

Proof. Since for every $u_2 > u_1$ we have that $x(u_2) > x(u_1)$, then there exists a $\bar{x} \in [0, x_*]$ with

$$\lim_{u \to \infty} x(u) = \bar{x}.$$

Assume that $\bar{x} < x_*$. Then we have that $f'(\bar{x}) > 0$ and $\varphi'(x) < 0$. Hence, there is a $\bar{u}$, such that for all $u \geq \bar{u}$ the following holds:

$$f'(\bar{x}) + u\varphi'(\bar{x}) < -1.$$

Since f' and φ' are continuous functions, there exists a $\delta > 0$, such that for all $u \geq \bar{u}$ and for all $x \in [\bar{x} - \delta, \bar{x}]$ it holds that

$$f'(x) + u\varphi'(x) < -\frac{1}{2}. \tag{2.9}$$

Furthermore, there is a u_0, such that for all $u \geq u_0$

$$x(u) \in [\bar{x} - \delta, \bar{x}]$$

holds. However, we now have

$$(f'(x) + u\varphi'(x))|_{x=x(u)} = 0,$$

which contradicts (2.9). Hence, $\lim_{u\to\infty} x(u) = x_*$ must hold.

With this we have

$$\begin{aligned}f(x) &= \lim_{u \to \infty} f(x(u)) \\ &\leq \lim_{u \to \infty} \mathcal{L}(x(u), u) \\ &= \liminf_{u \to \infty} g(u) \\ &\leq C_{\mathcal{D}} \leq \mathrm{OptV}(f, \varphi, \lambda).\end{aligned}$$

Since $f(x_*) = \mathrm{OptV}(f, \varphi, \lambda)$, it holds that $C_{\mathcal{D}} = \mathrm{OptV}(f, \varphi, \lambda)$.

□

2.3 Computability Framework

In this section, we cover the basics of computability theory, a field foundational to understanding the limits and capabilities of algorithmic processes. The concept of computability, along with the notion of computable real numbers, were initially proposed by Turing in [32] and [11], where he introduced the idea of numbers that can be precisely determined by Turing machines. For our definitions and notation, we draw upon the comprehensive treatments found in [33, 34] and [35].

2.3.1 Computable Numbers

A sequence of rational numbers $\{r_n\}_{n\in\mathbb{N}}$ is called a *computable sequence* if there exist recursive functions $a, b, s : \mathbb{N} \to \mathbb{N}$ with $b(n) \neq 0$ for all $n \in \mathbb{N}$ and

$$r_n = (-1)^{s(n)} \frac{a(n)}{b(n)}, \qquad n \in \mathbb{N}.$$

A real number x is said to be computable if there exists a computable sequence of rational numbers $\{r_n\}_{n\in\mathbb{N}}$, such that

$$|x - r_n| < 2^{-n} \tag{2.10}$$

for all $n \in \mathbb{N}$. This means that the computable real number x is completely characterized by the recursive functions $a, b, s : \mathbb{N} \to \mathbb{N}$. It has the representation (a, b, s), which we also write as $x \sim (a, b, s)$. It is clear that this representation must not be unique and that there might be other recursive functions $a', b', s' : \mathbb{N} \to \mathbb{N}$ which characterize x, i.e., $x \sim (a', b', s')$.

We denote the set of computable real numbers by $\mathbb{R}_c$, which encompasses numbers that can be computed to any desired precision by a Turing machine in a finite number of steps.

2.3.2 Computable Sequences

In this dissertation, we extensively work with the concepts of computable sequences of computable numbers and effective convergence. These fundamental notions provide a framework for understanding how sequences can be computed and how they converge in a manner that is algorithmically verifiable.

Definition 13. *A sequence of real numbers $\{x_n\}_{n\in\mathbb{N}}$ is* computable *(as a sequence) if there is a computable double sequence of rationals $\{r_{m,n}\}_{m,n\in\mathbb{N}^2}$ such that*

$$|r_{m,n} - x_n| \leq 2^{-m}$$

for all $m \in \mathbb{N}$ and $n \in \mathbb{N}$.

Definition 14. *A sequence* $\{r_n\}_{n\in\mathbb{N}}$ *of rational numbers converges* effectively *to a real number* x *if there exists a recursive function* $e\colon \mathbb{N}\to\mathbb{N}$ *such that for all* $N\in\mathbb{N}$ *it holds that*

$$k \geq e(N) \quad \textit{implies} \quad |r_k - x| \leq 2^k.$$

In what follows, we introduce the concept of Cauchy sequences and their subset, effectively convergent Cauchy sequences, which are essential for the computability analysis in Chapter 5.

Definition 15. *A sequence* $\{x_n\}_{n\in\mathbb{N}}$ *is called a* Cauchy sequence *if for every* $\epsilon > 0$*, there is a* $n_0\in\mathbb{N}$ *such that for every* $m, n > n_0$ *it holds that*

$$|x_n - x_m| < \epsilon.$$

Definition 16. *A Cauchy sequence* $\{x_n\}_{n\in\mathbb{N}}$ *is said to converge* effectively *if there is a recursive function* $e\colon \mathbb{N}\times\mathbb{N}\to\mathbb{N}$ *such that for all* $n, N\in\mathbb{N}$ *it holds that*

$$k \geq e(n, N) \quad \textit{implies } |x_k - x_n| \leq 2^{-N}$$

Remark 4. *Let* $\{x_n\}_{n\in\mathbb{N}}$ *and* $\{y_n\}_{n\in\mathbb{N}}$ *be computable sequences of real number. Then the following sequence are also computable:*

$$x_n \pm y_n,\ x_n y_n,\ x_n/y_n\ (y_n \neq 0 \textit{ for all } n),\ \exp x_n,\ \log x_n\ (x_n > 0 \textit{ for all } n).$$

Next, we introduce a new class of numbers which encompasses also the computable numbers, Σ_1, following the definition provided by Zheng in [36].

Definition 17. *The set* Σ_1 *is the set of numbers* $x\in\mathbb{R}$*, such that there is a computable sequence of rational numbers* $\{\mu_n\}_{n\in\mathbb{N}}$ *with* $\mu_n \leq \mu_{n+1}$ *and*

$$\lim_{n\to\infty} \mu_n = x.$$

It is noteworthy that numbers in Σ_1 may extend beyond computable numbers; in fact, the set of computable numbers, $\mathbb{R}_c$, is a subset of Σ_1.

Remark 5. *In Definition 17, we can also require the sequence* $\{\mu_n\}_{n\in\mathbb{N}}$ *can also be considered as a sequence of computable numbers, not limited to rational numbers. This implies that each* μ_n *must satisfy the condition* $\mu_n \leq \mu_{n+1}$ *for* $n\in\mathbb{N}$*.*

2.3.3 Computable Functions

In this subsection, we explore the concept of computable functions, delving into various classes of computability and examining how they interrelate.

Definition 18. *A function* $f_c : \mathbb{R}_c \to \mathbb{R}_c$ *is called* Borel-Turing computable *if there is an algorithm (or Turing machine) that transforms each given representation* (a, b, s) *of a computable real number* x *into a corresponding representation for the computable real number* $f_c(x)$*.*

To address the questions from Chapter 4 and 5, we need the concept of *computable continuous functions* [33, Def. A]. For this, let $\mathbb{I}_c$ denote a computable interval, i.e., $\mathbb{I}_c = [a, b]$ with $a, b \in \mathbb{R}_c$.

Definition 19 ([33])**.** *Let $\mathbb{I}_c \subset \mathbb{R}_c$ be a computable interval. A function $f_c : \mathbb{I}_c \to \mathbb{R}_c$ is called* computable continuous *if:*

1. *f_c is* sequentially computable*, i.e., f_c maps every computable sequence $\{x_n\}_{n\in\mathbb{N}}$ of points $x_n \in \mathbb{I}_c$ into a computable sequence $\{f_c(x_n)\}_{n\in\mathbb{N}}$ of real numbers,*
2. *f_c is* effectively uniformly continuous*, i.e., there is a recursive function $d : \mathbb{N} \to \mathbb{N}$ such that for all $x, y \in \mathbb{I}_c$ and all $N \in \mathbb{N}$ with*

$$\|x - y\| \leq \frac{1}{d(N)}$$

it holds that

$$|f_c(x) - f_c(y)| \leq \frac{1}{2^N}.$$

Remark 6. *The notion of computable continuous functions is stronger than that of Borel-Turing computable functions. Functions that are computable continuous are also Borel-Turing computable.*

There are other forms of computability including *Banach-Mazur computability*, which is the weakest form of computability.

Definition 20. *A function $f : \mathbb{R}_c \to \mathbb{R}_c$ is called* Banach-Mazur computable *if f maps any given computable sequence $\{x_n\}_{n\in\mathbb{N}}$ of computable real numbers into a computable sequence $\{f(x_n)\}_{n\in\mathbb{N}}$ of computable real numbers.*

In particular, Borel-Turing computability and computable continuous functions imply Banach-Mazur computability, but not vice versa.

For an overview of the logical relations between different notions of computability we refer to [37].

We further need the concepts of a recursive set and a recursively enumerable set as defined in [38]. These are used with the purpose of constructing sequences of computable channels used to study the computability of the feedback capacity function.

Definition 21. *A set $\mathcal{A} \subset \mathbb{N}$ is called* recursive *if there exists a computable function f such that $f(x) = 1$ if $x \in \mathcal{A}$ and $f(x) = 0$ if $x \notin \mathcal{A}$.*

Definition 22. *A set $\mathcal{A} \subset \mathbb{N}$ is* recursively enumerable *if there exists a recursive function whose domain is exactly $\mathcal{A}$.*

We have the following properties [38]:

- $\mathcal{A}$ is recursive is equivalent to: $\mathcal{A}$ is recursively enumerable and $\mathcal{A}^c$ is recursively enumerable.
- There exist recursively enumerable sets $\mathcal{A} \subset \mathbb{N}$ that are not recursive, i.e., $\mathcal{A}^c$ is not recursively enumerable. This means there are no computable, i.e., recursive, functions $f : \mathbb{N} \to \mathcal{A}^c$ with $[f(\mathbb{N})] = \{m \in \mathbb{N} \colon \exists n \in \mathbb{N} \text{ with } f(n) = m\} = \mathcal{A}^c$.

2.3.4 Dyadic Representation of Computable Numbers and Oracle Turing Machines

In this section, we introduce the fundamental concepts of computability and complexity theory, which are essential for understanding the material presented in Chapter 6.

First, we introduce the dyadic representation of rational numbers.

The set $\mathcal{D}$ of *dyadic rational numbers* are rational numbers with finite binary expansion. Each dyadic rational number d is naturally represented by a binary string $s = \pm s_n s_{n-1} \cdots s_0.t_1 t_2 \cdots t_m$ satisfying

$$d = \pm \sum_{i=1}^{n} s_i 2^i \pm \sum_{j=1}^{m} t_j 2^{-j}.$$

A representation s of a dyadic rational d has the precision m ($\mathrm{prec}(s) = m$), if it has m bits to the right of the binary point.

Next, we will use a notation different from the one in (2.10) to represent computable numbers, which will be utilized to define the subsequent concept of function-oracle Turing machines.

A number $t \in \mathbb{R}_c$ is said to be computable if there exists a Turing machine M with input $n \in \mathbb{N}$ and output $\varphi(n) = M(n) \in \mathbb{Q}$ such that

$$|t - \varphi(n)| \leq 2^{-n}.$$

A *function-oracle Turing machine* is an ordinary Turing machine M equipped with an additional query tape and two additional states: the query state and the answer state. When the machine enters the query state, the oracle function φ replaces the current string s in the query tape by the string $\varphi(s)$, moves the tape head back to the first cell of the query tape and puts the machine M in the answer state. When the time complexity is considered, the entire process of querying for the value $\varphi(s)$ costs only one time unit to the machine.

Next we introduce the notion of computable functions using the concept of oracle Turing machines.

Definition 23. *A real function $f\colon \mathbb{R} \to \mathbb{R}$ is* computable *if there is a function-oracle Turing machine M such that for each $x \in \mathbb{R}$ and each φ that binary converges to x, the function ψ computed by M with oracle φ (i.e., $\psi(n) = M_\varphi(n)$) binary converges to $f(x)$.*

Intuitively, a function f is computable, if for a given x and the oracle φ, the oracle Turing machine M_φ takes n as input and computes a dyadic rational $M_\varphi(n)$ that binary converges to $f(x)$, i.e., $|M_\varphi(n) - f(x)| \leq 2^{-n}$. During the computation, the information about x can be obtained from the oracle φ in one time step.

2.3.5 Complexity of Real Functions

For any $t \in \mathbb{R}_c$, the Turing machine will need several iterations to compute $\varphi(n)$. The number of iterations needed will increase when n increases. The quantitative relation between the number of iterations for computing $\varphi(n)$ and n determines the computational complexity of $t \in \mathbb{R}_c$.

Definition 24. *Let t be an integer function. The* time complexity *of a computable real number x is bounded by t if there exists a Turing machine which computes, on each input $n \in \mathbb{N}$, a dyadic rational number d in $t(n)$ moves such that $|d - x| \leq 2^{-n}$.*

Definition 25. *A real number x is* polynomial time computable *if its time complexity is bounded by a polynomial function* P*.*

Definition 26. *Let $f\colon [0,1] \to \mathbb{R}$ be a computable function. The* complexity *of f on $[0,1]$ is bounded by a function $q\colon \mathbb{N} \to \mathbb{N}$ if there exists an oracle Turing machine M which computes f such that for all φ that binary converge to a real number $x \in [0,1]$ and for all $n > 0$, $M_\varphi(n)$ halts in at most time $q(n)$.*

Definition 27. *A real function $f\colon [0,1] \to \mathbb{R}$ is* polynomial time computable *if its time complexity is bounded by a polynomial function* P*.*

Definition 28. *Let $\{\alpha_n\}_{n\in\mathbb{N}}$ be a computable sequence of computable numbers. This sequence is computable in polynomial time if there exists a polynomial $P\colon \mathbb{N} \times \mathbb{N} \to \mathbb{N}$, such that for all $n \in \mathbb{N}$ for each $M \in \mathbb{N}$ a number $\alpha_n \in \mathbb{Q}$ is computed in at most $P(n, M)$ steps such that*

$$|\alpha_n - \alpha_{n,M}| \leq \frac{1}{2^M}$$

holds.

2.3.6 Complexity Classes

In this subsection, we introduce some complexity classes that characterize the complexity of solving certain problems. We introduce and discuss the complexity classes related to decision and counting problems. The fundamental concepts discussed here are based on the comprehensive analysis found in [35].

For decision problems, the best known complexity classes are P and NP. These are problems that for a given input string only have two possible solutions, "0" = "no" or "1" = "yes".

Consider the language $\Sigma \subset \{0,1\}^*$[1]which is an infinite subset. The Entscheidungsproblem is described as follows: Find a Turing machine M, such that for $x \in \{0,1\}^*$ it holds that $M(x) = 1$ if and only if $x \in \Sigma$, otherwise it holds that $M(x) = 0$, i.e., the output of TM is either 0 or 1. The class P is the set of all decision problems that can be solved by a deterministic Turing machine in polynomial time, i.e., in a computation time that grows polynomial in the input size. The class NP is the set of all problems that can be solved by a non-deterministic Turing machine in polynomial time. It is clear from the definition that P $\subset$ NP but it remains an open question whether P = NP or P $\subsetneq$ NP. It is widely assumed that P is a proper subset of NP.

In order to formally define the complexity classes, we consider the concept non-deterministic Turing machines (NDTMs). The difference between a deterministic Turing

[1] *Notation:* $\{0,1\}^*$ denotes the set of all finite words in $\{0,1\}$, $|x|$ denotes the length of the sequence x.

machine and a NDTM is that the latter has more than one possible move from a given configuration, and also a special state q_{accept}. Hence, for a NDTM M, M outputs 1 on a given input x if there is at least one sequence of moves for x that makes M reach q_{accept}, otherwise, if x makes M stop, then it reaches the halting state and $M(x) = 0$. A NDTM is polynomial, if there is a $p\colon \mathbb{N} \to \mathbb{N}$ such that for every x the NDTM reaches the halting state or q_{accept} in at most $p(|x|)$ steps.

Definition 29 (Classes P and NP)**.** *Let $\Sigma \subset \{0,1\}^*$ be a language. Then Σ is in* P *if there exists a Turing machine M that solves the Entscheidungsproblem for Σ in polynomial time.*

A language $\Sigma \subset \{0,1\}^*$ is in NP *, if there exists a polynomial time NDTM M such that $x \in \Sigma \Leftrightarrow M(x) = 1$.*

The name NP comes originally from **N**on-deterministic **P**olynomial Turing machines. Intuitively, the class NP is the class of all problems, which are verifiable in polynomial time. This can be ilustrated in the following way: Let M be a polynomial NDTM. Let y be a sequence of moves describing a non-deterministic path that makes M reach q_{accept} on input x, then y is a *certificate* for x. This certificate has length $p(|x|)$ and can be then verified by a polynomial time Turing machine, which checks that M would have entered q_{accept} after using the non-deterministic path of M. In other words, if a set $\Sigma \subset \{0,1\}^*$ is in NP, then there exists a Turing machine M_v that takes inputs from $\{0,1\}^* \times \{0,1\}^*$ and outputs values in $\{0,1\}$ such that for any $x \in \Sigma$, there exists a certificate $y \in \{0,1\}^{p(|x|)}$ satisfying $M_v(x,y) = 1$.

Next, we consider functions that are defined on a different domain. More precisely, we consider a complexity class similar to P and NP that contains functions whose input belong to a singleton alphabet, i.e., functions whose domain is $\{0\}^*$.

Definition 30 (Classes P_1 and NP_1)**.** *Let $\Sigma_1 \subset \{0\}^*$ be a language. Then Σ_1 is in* P_1 *if there exists a Turing machine M that solves the Entscheidungsproblem for Σ_1 in polynomial time. A language $\Sigma_1 \subset \{0\}^*$ is in* NP_1 *, if there exists a non-deterministic polynomial time Turing machine M such that $x \in \Sigma_1 \Leftrightarrow M(x) = 1$.*

Let us introduce the characteristic functions $\chi_\Sigma\colon \{0,1\}^* \to \{0,1\}$ and $\chi_{\Sigma_1}\colon \{0\}^* \to \{0,1\}$ for languages $\Sigma \subset \{0,1\}^*$ and $\Sigma_1 \subset \{0\}^*$, respectively. These characteristic functions evaluate as follows: $\chi_\Sigma(x) = 1 \Leftrightarrow x \in \Sigma$.

We can then consider Definitions 30 and 29 as complexity requirements for computing the characteristic function χ_Σ. In other words, we can determine whether Σ is in the complexity class P and whether Σ_1 is in the complexity class P_1 by investigating whether the characteristic function χ_Σ is polynomial time computable.

We want to illustrate the same notions for the complexity classes for general functions. To represent such problems, we use functions denoted as $f : \{0,1\}^* \to \mathbb{N}$, defined on the set of all finite words in the binary alphabet $\{0,1\}$. For any given word $x \in \{0,1\}^*$, the value of $f(x)$ represents the count of solutions for that particular instance. The classes analog to the classes P and NP for counting problems are denoted by FP and #P respectively.

Definition 31 (Classes FP and #P)**.** *A function* $f\colon \{0,1\}^* \to \mathbb{N}$ *is in* FP *if it can be computed by a deterministic Turing machine in polynomial time.*

A function $f\colon \{0,1\}^* \to \mathbb{N}$ *is in* #P *if there exists a polynomial* $p\colon \mathbb{N} \to \mathbb{N}$ *and a polynomial time Turing machine* M*, such that for every string* $x \in \{0,1\}^*$*,*

$$f(x) = |\{y \in \{0,1\}^{p(|x|)} \colon M(x,y) = 1\}|.$$

Remark 7. *Definition 31 can also be described using NDTMs.* #P *consists of all functions* f*, such that* $f(x)$ *equals the number of certificates that describes the sequence of moves of an accepting path of a polynomial time NDTM* M *on input* x*.*

Remark 8. *By Definition 31, it is evident that* FP $\subseteq \#P$*. However, similar to* P *vs.* NP *it is an open question whether* FP $= \#P$*, i.e., whether any problem in* #P *can be efficiently (in polynomial time) solved by a Turing machine. It is commonly assumed that* FP $\subsetneq \#P$*. Moreover, if* FP $= \#P$*, then this would imply that* $P = NP$*. Conversely,* $P \neq NP$ *implies* $FP \neq \#P$*.*

With this, we can argue that problems in #P and $\#\mathrm{P}_1$ are considerably more complex than problems in NP and NP_1, respectively. In NP, it is generally challenging to find a y_* for x such that

$$f(x, y_*) = 1 \tag{2.11}$$

assuming the commonly accepted complexity assumption that $P \neq NP$. If $\hat{y}$ is considered a potential solution for (2.11), it is a straightforward task to determine whether $f(x,\hat{y}) = 1$ in polynomial time. The verification of whether $\hat{y}$ is a solution for the problem or not can be performed in a simple manner. It should be noted that this asymmetry between finding the solution and verifying if a given value is a solution or not forms the foundation of cryptography as a whole.

For #P, such a behavior is unknown. Even if $\{y_1, \ldots, y_r\}$ is considered as a solution set, the verification of whether $f(x, y_l) = 1$ for $1 \leq l \leq r$ is easy to implement. However, it is not clear if this approach is useful for #P since there may exist additional solutions not included in the considered solution set. In the case of #P, we require the set of all possible solutions. These arguments also apply to the sets NP_1.

In this work, we are interested in studying functions that are defined on the singleton alphabet, i.e., $\{0\}^* \subset \{0,1\}^*$. In other words, these functions are defined solely on the set of finite words composed of the symbol 0. The classes analog to FP and #P defined on singleton sets are denoted by FP_1 and $\#\mathrm{P}_1$ respectively.

Definition 32 (Classes FP_1 and $\#\mathrm{P}_1$)**.** *A function* $f\colon \{0\}^* \to \mathbb{N}$ *is said to be in* FP_1 *if it can be computed by a deterministic Turing machine in polynomial time.*

A function $f\colon \{0\}^* \to \mathbb{N}$ *is said to be in* $\#\mathrm{P}_1$ *if there exists a polynomial* $p\colon \mathbb{N} \to \mathbb{N}$ *and a polynomial time Turing machine* M *so that for every string* $x \in \{0\}^*$

$$f(x) = |\{y \in \{0\}^{p(|x|)} \colon M(x,y) = 1\}|.$$

Remark 9. *As in the previous cases, we have* $\mathrm{FP}_1 \subseteq \#\mathrm{P}_1$ *but it is open whether* $\mathrm{FP}_1 = \#\mathrm{P}_1$*. However, it is widely assumed that* $\mathrm{FP}_1 \subsetneq \#\mathrm{P}_1$*. Similarly as above,*

an $f \in \#\mathrm{P}_1$ is said to be complete in $\#\mathrm{P}_1$ if any other $g \in \#\mathrm{P}_1$ can be reduced to f by a polynomial time Turing machine.

Remark 10. *Similar to the relation between the classes* NP *and* #P*, the class $\#\mathrm{P}_1$ is more general than NP_1 since it not only contains problems for which a certificate can be verified in polynomial time but also counts the number of certificates verifiable in polynomial time. $\#\mathrm{P}_1$ is therefore more difficult and more complex than NP_1.*

For the next definition we need the notion of reduction. A reduction is an algorithm for transforming one problem into another problem.

Definition 33. *A function $f \in \#\mathrm{P}_1$ is said to be complete in $\#\mathrm{P}_1$ if any other $g \in \#P_1$ can be reduced to f by a polynomial time Turing machine.*

Suppose there is a deterministic Turing machine capable of solving a problem f in polynomial time. In such a scenario, it implies that any other problem g belonging to the complexity class $\#\mathrm{P}_1$ can also be solved in polynomial time by a deterministic Turing machine. In simpler terms, if f is considered as a complete problem within the class $\#\mathrm{P}_1$, then f is at least as challenging as the most difficult problem in $\#\mathrm{P}_1$.

2.3.7 Computation and Complexity of the Logarithm Function

In this subsection, we examine the binary convergence and computational behavior of the logarithm function. Additionally, we study the computational complexity arising from applying the logarithm function to another computable function. These properties are essential for showing the desired results in Chapter 6.

Lemma 4. *Let $\underline{\alpha}, \bar{\alpha} \in \mathbb{Q}$ with $0 < \underline{\alpha} < \bar{\alpha} < \infty$. Let $x_* = \frac{\underline{\alpha}+\bar{\alpha}}{2}$ and $\beta = \frac{2\underline{\alpha}}{\bar{\alpha}+\underline{\alpha}}$. Let $\frac{r}{2^s}$ be a dyadic number with $s, r \in \mathbb{N}$ such that $\beta \leq \frac{r}{2^s} < 1$ and let $m_1 \in \mathbb{N}$ be such that $\frac{r^{m_1}}{2^{sm_1}} < \frac{1}{2}$. Then for all $x \in [\underline{\alpha}, \bar{\alpha}]$ and for all $m \geq m_1$ we have*

$$\left| \ln x - \ln(x_*) - \sum_{\ell=1}^{m^2} \frac{(-1)^{\ell-1}}{\ell x_*^\ell} (x - x_*)^\ell \right| < \gamma \frac{1}{2^m}$$

with $\gamma = \frac{2\underline{\alpha}}{\bar{\alpha}-\underline{\alpha}} \in \mathbb{Q}$.

Proof. $\ln(\cdot)$ is an absolute convergent Taylor-series in $x \in [\underline{\alpha}, \bar{\alpha}]$. For $\Psi(x) = \ln(x)$ we have $\Psi^{(\ell)}(x) = \frac{(-1)^{\ell-1}}{\ell x_*^\ell}$, for $\ell \geq 1$ and $x > 0$.

We then have

$$\begin{aligned}
&\Big|\ln x - \ln(x_*) - \sum_{\ell=1}^{m^2} \frac{(-1)^{\ell-1}}{\ell x_*^\ell}(x-x_*)^\ell\Big| \\
&= \Big|\sum_{\ell=m^2+1}^{\infty} \frac{(-1)^{\ell-1}}{\ell x_*^\ell}(x-x_*)^\ell\Big| \leq \sum_{\ell=m^2+1}^{\infty} \frac{|x-x_*|^\ell}{|x_*|^\ell} \\
&= \sum_{\ell=m^2+1}^{\infty} \Big|1-\frac{x}{x_*}\Big|^\ell \leq \sum_{\ell=m^2+1}^{\infty} \Big|\frac{\underline{\alpha}+\bar{\alpha}-(\bar{\alpha}-\underline{\alpha})}{\underline{\alpha}+\bar{\alpha}}\Big|^\ell \\
&= \sum_{\ell=m^2+1}^{\infty} \Big(\frac{2\underline{\alpha}}{\underline{\alpha}+\bar{\alpha}}\Big)^\ell = \sum_{\ell=m^2+1}^{\infty} \beta^\ell \\
&= \beta^{m^2+1}\frac{1}{1-\beta} = \frac{\beta}{1-\beta}\beta^{M^2} \\
&= \frac{\frac{2\underline{\alpha}}{\underline{\alpha}+\bar{\alpha}}}{1-\frac{2\underline{\alpha}}{\underline{\alpha}+\bar{\alpha}}}\Big(\frac{2\underline{\alpha}}{\underline{\alpha}+\bar{\alpha}}\Big)^{m^2} \\
&= \frac{2\underline{\alpha}}{\bar{\alpha}-\underline{\alpha}}\Big(\frac{2\underline{\alpha}}{\underline{\alpha}+\bar{\alpha}}\Big)^{m^2} \\
&\leq \frac{2\underline{\alpha}}{\bar{\alpha}-\underline{\alpha}}\frac{r^{m^2}}{2^{sm^2}} = \gamma\Big(\frac{r^m}{2^{sm}}\Big)^m < \gamma\frac{1}{2^m}.
\end{aligned}$$

□

The constant γ, does not influence the binary convergence of the power series. So the power series binary converges to the logarithm function. This is visualized in the next lemma.

Lemma 5. *Let $m_1, m_2 \in \mathbb{N}$ be arbitrary such that $m_2 \geq m_1$. Let $\gamma < 2^{m_2}$. Then for all $m \in \mathbb{N}$ and all x as in Lemma 4 we have*

$$\Big|\ln x - \ln(x_*) - \sum_{\ell=1}^{(m_2+m)^2} \frac{(-1)^{\ell-1}}{\ell x_*^\ell}(x-x_*)^\ell\Big| < \frac{1}{2^m}.$$

Proof. The proof follows from Lemma 4. We have

$$\Big|\sum_{\ell=(m_2+m)^2}^{\infty} \frac{(-1)^{\ell-1}}{\ell x_*^\ell}(x-x_*)^\ell\Big| < \gamma\frac{1}{2^{m_2+m}} < \frac{1}{2^m}.$$

□

Lemma 6. *For $x \in [\underline{\alpha}, \bar{\alpha}]$, $\beta = \frac{\bar{\alpha}-\underline{\alpha}}{\bar{\alpha}+\underline{\alpha}}$ and $m \in \mathbb{N}$ we consider the polynomial*

$$Q_m(x) = \ln(x_*) + \sum_{\ell=1}^{m^2} \frac{(-1)^{\ell-1}}{\ell x_*^\ell}(x-x_*)^\ell.$$

For $x_1, x_2 \in [\underline{\alpha}, \bar{\alpha}]$ we have

$$|Q_m(x_1) - Q_m(x_2)| \leq \frac{2}{\beta(1-\beta)^2}|x_1 - x_2|.$$

Proof. Using the mean value theorem we get for $f_\ell(x) = (x - x_*)^\ell$ for $1 \leq \ell \leq m^2$

$$\begin{aligned}|f_\ell(x_1) - f_\ell(x_2)| &= |f'_\ell(x_{1,2})||x_1 - x_2| \\ &\leq \ell|(\bar{\alpha} - x_*)^{\ell-1}||x_1 - x_2| \\ &= \ell|(\frac{\bar{\alpha} - \underline{\alpha}}{2})^{\ell-1}||x_1 - x_2|.\end{aligned}$$

We then have that

$$\begin{aligned}|Q_m(x_1) - Q_m(x_2)| &\leq \sum_{\ell=1}^{m^2} \frac{(-1)^{\ell+1}}{\ell x_*^\ell} \ell(\frac{\bar{\alpha} - \underline{\alpha}}{2})^{\ell-1}||x_1 - x_2| \\ &\leq \frac{|x_1 - x_2|}{x_*} \sum_{\ell=1}^{m^2} \left(\frac{\frac{\bar{\alpha}-\underline{\alpha}}{2}}{\frac{\bar{\alpha}+\underline{\alpha}}{2}}\right)^{\ell-1} \\ &< \frac{|x_1 - x_2|}{x_*} \sum_{\ell=1}^{\infty} \beta^{\ell-1} \\ &\leq \frac{|x_1 - x_2|}{x_*} \sum_{\ell=0}^{\infty} \beta^{\ell} \\ &= \frac{|x_1 - x_2|}{x_*} \frac{1}{1-\beta} \\ &= \frac{2}{\beta(1-\beta)^2}|x_1 - x_2|.\end{aligned}$$

□

Next we look at the computability and complexity properties of the logarithm function. More precisely, we show that the composition of a computable continuous periodic function g with the logarithm function, i.e. $\ln(g(\omega))$, results in a computable continuous function. Moreover, if the function g has low complexity, then the composition $\ln(g(\omega))$ has also low complexity.

Lemma 7. *Let g be a computable continuous 2π-periodic function with $\min_{\omega\in[-\pi,\pi]} g(\omega) = \underline{c} > 0$.*

1. *Then $\ln \circ g$ is also a computable continuous 2π-periodic function.*
2. *Let g polynomial time computable, then $\ln \circ g$ is also polynomial time computable.*

Proof. Let g be a fixed function. Let

$$\underline{c} = \min_{\omega\in[-\pi,\pi]} g(\omega) > 0 \text{ and } \bar{C} = \max_{\omega\in[-\pi,\pi]} g(\omega) < \infty.$$

We choose a $\underline{\alpha} \in \mathbb{Q}$ with $\underline{\alpha} < \frac{\underline{c}}{2}$ and $\bar{\alpha} \in \mathbb{Q}$ with $\bar{\alpha} > \bar{C} + 1$. In this proof, we work with the oracle Turing model. We start with an algorithm for computing g. Based on this algorithm, we construct a new algorithm for computing $\ln \circ g$. If the algorithm computing

g computes the approximation of g with an approximation error of $\frac{1}{2^M}$ in polynomial time, then the new constructed algorithm for $\ln \circ g$ also computes an approximation of $\ln \circ g$ with an approximation error of $\frac{1}{2^M}$ in polynomial time.

We choose $m_3 \in \mathbb{N}$ such that $\frac{2}{\beta(1-\beta)^2} < 2^{m_3}$ holds. Let $M \in \mathbb{N}$ be arbitrary. Let $C(\omega, M)$ be the approximation of g with precision M, i.e.,

$$|g(\omega) - C(\omega, M)| < \frac{1}{2^M}.$$

Let the number $C(\omega, \tilde{M})$ with $\tilde{M} = M + m_3 + 1 + m_2$ be the approximation for $g(\omega)$ with approximation error $\frac{1}{2^M}$ computed by the algorithm to calculate $g(\omega)$ for input M and Oracle input ω. We then have

$$|g(\omega) - C(\omega, \tilde{M})| < \frac{1}{2^{M+1}}.$$

We use the number $C(\omega, \tilde{M})$ and calculate $d(\omega, M) \coloneqq Q_{\tilde{M}}(C(\omega, \tilde{M}))$.

The polynomial

$$Q_{\tilde{M}}(x) = \sum_{\ell=0}^{M^2} \frac{(-1)^{\ell+1}}{x_*^\ell}(x - x_*)^\ell.$$

has only polynomial many coefficients that are different from 0. All of them are rational numbers and are computed depending on the precision $\tilde{M}$ in polynomial time. This way, the approximation $d(\omega, M)$ of the number $C(\omega, \tilde{M})$ can be computed in polynomial time depending on M. Now we have

$$\begin{aligned} |\ln(g(\omega)) - d(\omega, M)| &= |\ln(g(\omega)) - Q_{\tilde{M}}(g(\omega)) + Q_{\tilde{M}}(g(\omega)) - Q_{\tilde{M}}(C(\omega, \tilde{M}))| \\ &\leq |\ln(g(\omega)) - Q_{\tilde{M}}(g(\omega))| + |Q_{\tilde{M}}(g(\omega)) - Q_{\tilde{M}}(C(\omega, \tilde{M}))|. \end{aligned}$$

From Lemma 5 we have that

$$|\ln(g(\omega)) - Q_{\tilde{M}}(g(\omega))| < \frac{1}{2^{M+1}}.$$

Further, from Lemma 6 and the definition of m_3 we have

$$|Q_{\tilde{M}}(g(\omega)) - Q_{\tilde{M}}(C(\omega, \tilde{M}))| \leq |g(\omega) - C(\omega, \tilde{M})| < \frac{1}{2^{M+1}}.$$

This way we have

$$|\ln(g(\omega)) - d(\omega, M)| < \frac{1}{2^{M+1}} + \frac{1}{2^{M+1}}.$$

□

2.3.8 Computation and Complexity of Integration

In this subsection, we explore the computational aspects and complexity properties of integration, which are important for the analysis in Chapters 5 and 6.

Theorem 3 ([33, p.37])**.** *Let $[a, b] \subset \mathbb{R}$ be a compact interval. If $f\colon [a, b] \to \mathbb{R}$ is a computable function then $\int_a^b f(t)\,dt$ is a computable number.*

The first results regarding the computational complexity of integrals were derived in [39]. The following theorem states that the integral of a polynomial time computable function over an interval gives a polynomial time computable number if $\mathrm{FP}_1 = \#\mathrm{P}_1$.

Theorem 4 ([34, p.184])**.** *The computation of $\int_0^1 f(t)\,dt$ lies in $\#\mathrm{P}_1$ for all polynomial time computable functions $f\colon [0, 1] \to \mathbb{R}$. Moreover, there exists a polynomial-time computable function g which is infinitely differentiable and such that the computation of its integral is $\#\mathrm{P}_1$-complete.*

3 Capacity Region Continuity of Compound Broadcast Channels with Confidential Messages

In this chapter, we delve into the study of the compound BCC. Compound channels provide a framework for modeling a realistic Channel State Information (CSI) scenario, where legitimate users lack precise knowledge of the actual channel realization. Instead, they are aware that the current channel belongs to a known uncertainty set and that the channel remains constant throughout the entire transmission. This model applies, for example, to the down-link of a cellular system, in which the base station transmits information to a user. The base station obtains limited CSI, for example via the up-link from pilot signal estimation at the receiver. Compound channels model the channel uncertainty based on a finite number of estimates.

The discrete memoryless compound BCC consists of one sender and two receivers. The sender aims to transmit two messages: a common message for both receivers and a confidential message intended solely for receiver 1, while keeping receiver 2 unaware of the confidential message.

In [40], the compound BCC capacity region using the strong secrecy criterion was characterized. This characterization involves a multi-letter expression, which represents the limit of a sequence of optimization problems, despite the channel being described only by discrete and finite parameters.

Multi-letter expressions present challenges in practical applications. One such challenge is the difficulty in extracting important insights, such as robustness and computability, from these expressions. Understanding the continuity properties of capacity with multi-letter expressions is crucial for addressing these challenges.

Here, we aim to determine whether the capacity region of the compound BCC depends continuously on the uncertainty set. If small changes in the uncertainty set lead to significant changes in the corresponding capacity region, the compound BCC is considered fragile, complicating the design of practical communication systems. Therefore, a continuous behavior of the capacity region is desired.

3.1 Compound Broadcast Channel with Confidential Messages

The transmitter and the receiver of a compound channel know an uncertainty set of channels to which the channel belongs; however, they do not know the actual channel realization. The channel remains constant during the entire transmission. We consider a two receiver compound BCC. The transmitter sends simultaneously a common message to both receivers and a confidential message to receiver 1, which must be kept secret from receiver 2. Let $\mathcal{X}$ be the finite input alphabet, $\mathcal{Y}$ and $\mathcal{Z}$ the finite output alphabets of receivers 1 and 2, respectively, and let $\mathcal{S}$ be a finite set of channel states. For each channel state $s \in \mathcal{S}$, input sequence $x^n \in \mathcal{X}^n$ and output sequences $y^n \in \mathcal{Y}^n$ and $z^n \in \mathcal{Z}^n$, the discrete memoryless broadcast channel is given by

$$Q_s^n(y^n, z^n|x^n) \coloneqq \prod_{i=1}^{n} Q_s(y_i, z_i|x_i)$$

with marginal channels $W_s^n(y^n|x^n)$ and $V_s^n(z^n|x^n)$.

Definition 34. *The discrete memoryless compound broadcast channel $\mathfrak{W}$ is given by the channel pair family with common input*

$$\mathfrak{W} \coloneqq \{(W_s, V_s) \colon s \in \mathcal{S}\}.$$

3.1.1 Codes for Compound Broadcast Channels

We consider a block-code of arbitrary but fixed length n. Let $\mathcal{M}_0 := \{1, \dots, M_{0,n}\}$ be the common message set and $\mathcal{M}_1 := \{1, \dots, M_{1,n}\}$ the confidential message set. We use the abbreviation $\mathcal{M} \coloneqq \mathcal{M}_0 \times \mathcal{M}_1$.

Definition 35. *An* $(n, M_{0,n}, M_{1,n})$-code *for the compound BCC consists of a stochastic encoder*

$$E \colon \mathcal{M}_0 \times \mathcal{M}_1 \to \mathcal{P}(\mathcal{X}^n)$$

i.e., a stochastic matrix, and decoders at receivers 1 *and* 2

$$\begin{aligned} \varphi_1 &\colon \mathcal{Y}^n \to \mathcal{M}_0 \times \mathcal{M}_1 \\ \varphi_2 &\colon \mathcal{Z}^n \to \mathcal{M}_0. \end{aligned}$$

The average error probability for receivers 1 and 2 and the channel realization $s \in \mathcal{S}$ are

$$\begin{aligned} \overline{e}_{1,n}(s) &:= \frac{1}{|\mathcal{M}|} \sum_{m \in \mathcal{M}} \sum_{x^n \in \mathcal{X}^n} \sum_{y^n : \varphi_1(y^n) \neq m} W_s^n(y^n|x^n) E(x^n|m) \\ \overline{e}_{2,n}(s) &:= \frac{1}{|\mathcal{M}|} \sum_{m \in \mathcal{M}} \sum_{x^n \in \mathcal{X}^n} \sum_{z^n : \varphi_2(z^n) \neq m_0} V_s^n(z^n|x^n) E(x^n|m). \end{aligned}$$

Since reliable communication is required for all $s \in \mathcal{S}$, we consider the maximum average error probabilities, i.e. $\overline{e}_{1,n} = \max_{s\in\mathcal{S}} \overline{e}_{1,n}(s)$ and $\overline{e}_{2,n} = \max_{s\in\mathcal{S}} \overline{e}_{2,n}(s)$.

The confidential message has to be kept secret from the non-legitimate receiver for all channel realizations. Therefore, we require $\max_{s\in\mathcal{S}} I(M_1; Z_s^n) \leq \epsilon_n$ for some $\epsilon_n > 0$ with M_1 uniformly distributed over the set $\mathcal{M}_1$ and Z_s^n the output at the non-legitimate receiver for the channel realization $s \in \mathcal{S}$. This criterion is known as *strong secrecy* [41, 42].

Definition 36. *A rate pair $(R_0, R_1) \in \mathbb{R}_+^2$ is said to be achievable for the compound BCC if for any $\tau > 0$ there is an $n(\tau) \in \mathbb{N}$ and a sequence of $(n, M_{0,n}, M_{1,n})$-codes such that for all $n \geq n(\tau)$ we have $\frac{1}{n}\log M_{0,n} \geq R_0 - \tau, \frac{1}{n}\log M_{1,n} \geq R_1 - \tau$, and*

$$\max_{s\in\mathcal{S}} I(M_1; Z_s^n) \leq \epsilon_n \tag{3.1}$$

with $\overline{e}_{1,n}, \overline{e}_{2,n}, \epsilon_n \to 0$ as $n \to \infty$.

Definition 37. *The set closure of all achievable rate pairs is the* capacity region $\mathcal{C}(\mathfrak{W})$ *of the compound BCC $\mathfrak{W}$.*

3.1.2 Capacity Results

In this section we present an achievable rate region and a multi-letter characterization of the compound BCC capacity region [40].

Lemma 8 ([40])**.** *An achievable secrecy rate region for the compound BCC is given by the set of all rate pairs $(R_0, R_1) \in \mathbb{R}_+^2$ satisfying*

$$\begin{aligned} R_0 &\leq \min_{s\in\mathcal{S}} \min\{I(U; Y_s), I(U; Z_s)\} \\ R_1 &\leq \min_{s\in\mathcal{S}} I(V; Y_s|U) - \max_{s\in\mathcal{S}} I(V; Z_s|U) \end{aligned}$$

for some random variables U, V, X where $U - V - X - (Y_s, Z_s)$ forms a Markov chain. Furthermore, the strong secrecy criterion goes exponentially fast to zero and the decoding error at the non-legitimate receiver goes exponentially fast to one.

We next present a multi-letter description of $\mathcal{C}(\mathfrak{W})$ of the compound BCC $\mathfrak{W}$. Let $n \in \mathbb{N}$ be arbitrary but fixed. We define the rate region $\mathcal{R}_n(\mathfrak{W}, U, V, X^n)$ as the set of all rate pairs $(R_0, R_1) \in \mathbb{R}_+^2$ satisfying

$$R_0 \leq \frac{1}{n} \inf_{s\in\mathcal{S}} \min\{I(U; Y_s^n), I(U; Z_s^n)\} \tag{3.2}$$

$$R_1 \leq \frac{1}{n}(\inf_{s\in\mathcal{S}} I(V; Y_s^n|U) - \sup_{s\in\mathcal{S}} I(V; Z_s^n|U)) \tag{3.3}$$

for the random variables satisfying the Markov chain relationship $U - V - X^n - (Y_s^n, Z_s^n)$. For a given $n \in \mathbb{N}$ we define the region

$$\mathcal{M}_n(\mathfrak{W}) = \bigcup_{U-V-X^n} \mathcal{R}_n(\mathfrak{W}, U, V, X^n)$$

that is, $\mathcal{M}_n(\mathfrak{W})$ is the union of the regions $\mathcal{R}_n(\mathfrak{W}, U, V, X^n)$ over all random variables satisfying the Markov chain relationship $U - V - X^n$.

Theorem 5. *The strong secrecy capacity region $\mathcal{C}(\mathfrak{W})$ of the compound BCC $\mathfrak{W}$ is the convex hull closure of the union of the regions $\mathcal{M}_n(\mathfrak{W})$ over all $n \in \mathbb{N}$, i.e.*

$$\mathcal{C}(\mathfrak{W}) = \overline{\text{conv}}(\bigcup_{n\in\mathbb{N}} \mathcal{M}_n(\mathfrak{W})).$$

Remark 11. *To the best of our knowledge, there is still no single-letter characterization of $\mathcal{C}(\mathfrak{W})$ known.*

Remark 12. *The union of the rate regions $\bigcup_{n\in\mathbb{N}} \mathcal{M}_n(\mathfrak{W})$ may itself not be convex. However, all rate pairs in the convex hull can be achieved by time sharing between the points in the rate regions $\mathcal{M}_n(\mathfrak{W})$.*

3.2 Continuity of the Compound BCC Capacity Region

In this section we first define the distance between two compound BCCs and the distance between rate regions. We then analyze the continuity of the compound BCC capacity region.

3.2.1 Distance between Compound Broadcast Channels and Sets

Let (W, V) and $(\widetilde{W}, \widetilde{V})$ be two broadcast channels. We define the distance between channels as

$$d(W, \widetilde{W}) \coloneqq \max_{x\in\mathcal{X}} \sum_{y\in\mathcal{Y}} |W(y|x) - \widetilde{W}(y|x)|$$
$$d(V, \widetilde{V}) \coloneqq \max_{x\in\mathcal{X}} \sum_{z\in\mathcal{Z}} |V(z|x) - \widetilde{V}(z|x)|$$

and the distance between two broadcast channels as

$$d((W, V), (\widetilde{W}, \widetilde{V})) \coloneqq \max(d(W, \widetilde{W}), d(V, \widetilde{V})).$$

Let $\mathfrak{W}_1 = \{(W_{s_1}, V_{s_1})\colon s_1 \in \mathcal{S}_1\}$ and $\mathfrak{W}_2 = \{(W_{s_2}, V_{s_2})\colon s_2 \in \mathcal{S}_2\}$ be two finite compound broadcast channels with marginal compound channels $\mathcal{W}_i = \{W_{s_i}\colon s_i \in \mathcal{S}_i\}$ and $\mathcal{V}_i = \{V_{s_i}\colon s_i \in \mathcal{S}_i\}$ for $i \in \{1, 2\}$. We define the distance between two marginal compound channels as

$$d_1(\mathcal{W}_1, \mathcal{W}_2) = \max_{s_2\in\mathcal{S}_2} \min_{s_1\in\mathcal{S}_1} d(W_{s_1}, W_{s_2})$$
$$d_2(\mathcal{W}_1, \mathcal{W}_2) = \max_{s_1\in\mathcal{S}_1} \min_{s_2\in\mathcal{S}_2} d(W_{s_1}, W_{s_2})$$

$$d_1(\mathcal{V}_1, \mathcal{V}_2) = \max_{s_2\in\mathcal{S}_2} \min_{s_1\in\mathcal{S}_1} d(V_{s_1}, V_{s_2})$$
$$d_2(\mathcal{V}_1, \mathcal{V}_2) = \max_{s_1\in\mathcal{S}_1} \min_{s_2\in\mathcal{S}_2} d(V_{s_1}, V_{s_2}).$$

Definition 38. *Let $\mathfrak{W}_1$ and $\mathfrak{W}_2$ be two compound broadcast channels. The distance $D(\mathfrak{W}_1, \mathfrak{W}_2)$ between $\mathfrak{W}_1$ and $\mathfrak{W}_2$ is defined as*

$$D(\mathfrak{W}_1, \mathfrak{W}_2) = \max\Big\{ d_1(\mathcal{W}_1, \mathcal{W}_2), d_2(\mathcal{W}_1, \mathcal{W}_2), \\ d_1(\mathcal{V}_1, \mathcal{V}_2), d_2(\mathcal{V}_1, \mathcal{V}_2) \Big\}.$$

To compare different rate regions, we define the following distance of sets.

Definition 39. *Let $\mathcal{R}_1$ and $\mathcal{R}_2$ be two non-empty compact subsets of the metric space $(\mathbb{R}^2_+, d)$ with $d(x,y) = \sum_{i=1}^2 |x_i - y_i|$ for all $x, y \in \mathbb{R}^2_+$. We define the distance between two sets as*

$$D_R(\mathcal{R}_1, \mathcal{R}_2) = \max\Big\{ \max_{r_1 \in \mathcal{R}_1} \min_{r_2 \in \mathcal{R}_2} d(r_1, r_2), \\ \max_{r_2 \in \mathcal{R}_2} \min_{r_1 \in \mathcal{R}_2} d(r_1, r_2) \Big\}.$$

3.2.2 Continuity of the Capacity Region of the Compound BCC

We use the following technical result, which is an extension of [43, Lem. 2,].

Lemma 9 ([43])**.** *Let $\epsilon \in (0,1)$ be arbitrary. For all (X,Y) and $(\widetilde{X}, \widetilde{Y})$ be two pairs of random variables with finite range $\mathcal{X} \times \mathcal{Y}$ and joint probabilities distributions $P_{X,Y}, P_{\widetilde{X},\widetilde{Y}} \in \mathcal{P}(\mathcal{X} \times \mathcal{Y})$. If $||P_{X,Y} - P_{\widetilde{X},\widetilde{Y}}|| \leq \epsilon$, then it holds*

$$|H(Y|X) - H(\widetilde{Y}|\widetilde{X})| \leq \delta_1(\epsilon, |\mathcal{Y}|) \tag{3.4}$$

with $\delta_1(\epsilon, |\mathcal{Y}|) := 2\epsilon \log |\mathcal{Y}| + 2H_2(\epsilon)$.

Remark 13. *Note that the right-hand side of (3.4) depends only on the size of the alphabet $\mathcal{Y}$, but it is independent of $\mathcal{X}$.*

Lemma 10. *Let $\mathcal{X}$ and $\mathcal{Y}$ be finite alphabets and $W, \widetilde{W} \colon \mathcal{X} \to \mathcal{P}(\mathcal{Y})$ be arbitrary channels with*

$$d(W, \widetilde{W}) \leq \epsilon$$

for some $\epsilon > 0$. For an arbitrary $n \in \mathbb{N}$, let $\mathcal{U}$ and $\mathcal{V}$ be two finite sets, $P_U \in \mathcal{P}(\mathcal{U})$ be the uniform distribution on $\mathcal{U}$, $P_{V|U}(\cdot|u)$ be the conditional distribution of the random variable V over $\mathcal{V}$ given $U = u$, and $E(x^n|v)$ with $x^n \in \mathcal{X}^n$ conditioned on $u \in \mathcal{U}$ be an arbitrary stochastic encoder. We consider the probability distributions

$$P_{UVY^n}(u,v,y^n) = \sum_{x^n \in \mathcal{X}^n} W^n(y^n|x^n) E(x^n|v) P_{V|U}(v|u) P_U(u)$$
$$P_{UV\widetilde{Y}^n}(u,v,y^n) = \sum_{x^n \in \mathcal{X}^n} \widetilde{W}^n(y^n|x^n) E(x^n|v) P_{V|U}(v|u) P_U(u).$$

Then it holds that

$$|I(V;Y^n|U) - I(V;\widetilde{Y}^n|U)| \leq n\delta_2(\epsilon, |\mathcal{Y}|) \tag{3.5}$$

with $\delta_2(\epsilon, |\mathcal{Y}|) := 4\epsilon \log |\mathcal{Y}| + 4H_2(\epsilon)$.

Proof. Let $0 \leq k \leq n$ be arbitrary. We define

$$P_{UVY_1^k\widetilde{Y}_{k+1}^n}(u,v,y_1^k,y_{k+1}^n) := \sum_{x^n\in\mathcal{X}^n}\prod_{l=1}^{k}W(y_l|x_l)\prod_{l=k+1}^{n}\widetilde{W}(y_l|x_l)E(x^n|v)P_{V|U}(v|u)P_U(u).$$

So we have

$$I(V;Y^n|U)-I(V;\widetilde{Y}^n|U)=\sum_{k=0}^{n-1}\Big(I(V;Y_1^{k+1}\widetilde{Y}_{k+2}^n|U)-I(V;Y_1^k\widetilde{Y}_{k+1}^n|U)\Big). \tag{3.6}$$

For all $0 \leq k \leq n-1$ it holds

$$\begin{aligned}
I(V;Y_1^{k+1}\widetilde{Y}_{k+2}^n|U)-I(V;Y_1^k\widetilde{Y}_{k+1}^n|U) &= I(V;Y_1^k|U)+I(V;Y_{k+1}\widetilde{Y}_{k+2}^n|Y_1^kU)\\
&\quad -I(V;Y_1^k|U)-I(V;\widetilde{Y}_{k+1}^n|Y_1^kU)\\
&= I(V;Y_{k+1}\widetilde{Y}_{k+2}^n|Y_1^kU)-I(V;\widetilde{Y}_{k+1}^n|Y_1^kU)\\
&= I(V;\widetilde{Y}_{k+2}^n|Y_1^kU)+I(V;Y_{k+1}|\widetilde{Y}_{k+2}^nY_1^kU)\\
&\quad -I(V;\widetilde{Y}_{k+2}^n|Y_1^kU)-I(V;\widetilde{Y}_{k+1}|\widetilde{Y}_{k+2}^nY_1^kU)\\
&= I(V;Y_{k+1}|\widetilde{Y}_{k+2}^nY_1^kU)-I(V;\widetilde{Y}_{k+1}|\widetilde{Y}_{k+2}^nY_1^kU)\\
&= H(Y_{k+1}|\widetilde{Y}_{k+2}^nY_1^kU)-H(\widetilde{Y}_{k+1}|\widetilde{Y}_{k+2}^nY_1^kU)\\
&\quad -H(VY_{k+1}|\widetilde{Y}_{k+2}^nY_1^kU)+H(V\widetilde{Y}_{k+1}|\widetilde{Y}_{k+2}^nY_1^kU).
\end{aligned} \tag{3.7}$$

We want to analyze the right-hand side of (3.7). For $0 \leq k \leq n-1$, it holds

$$\begin{aligned}
&\|P_{UVY_1^{k+1}\widetilde{Y}_{k+2}^n}-P_{UVY_1^k\widetilde{Y}_{k+1}^n}\|\\
&\quad=\sum_{v\in\mathcal{V}}\sum_{u\in\mathcal{U}}\sum_{y^n\in\mathcal{Y}^n}\Big|P_{UVY_1^{k+1}\widetilde{Y}_{k+2}^n}(u,v,y_1^{k+1}y_{k+2}^n)-P_{UVY_1^k\widetilde{Y}_{k+1}^n}(u,v,y_1^ky_{k+1}^n)\Big|\\
&\quad=\sum_{v\in\mathcal{V}}\sum_{u\in\mathcal{U}}\sum_{y^n\in\mathcal{Y}^n}\Big|\sum_{x^n\in\mathcal{X}^n}\Big(\prod_{l=1}^{k+1}W(y_l|x_l)\prod_{l=k+2}^{n}\widetilde{W}(y_l|x_l)-\prod_{l=1}^{k+1}W(y_l|x_l)\prod_{l=k+2}^{n}\widetilde{W}(y_l|x_l)\Big)\\
&\qquad\times E(x^n|v)P_{V|U}(v|u)P_U(u)\Big|\\
&\quad=\sum_{v\in\mathcal{V}}\sum_{u\in\mathcal{U}}\sum_{y^n\in\mathcal{Y}^n}\Big|\sum_{x^n\in\mathcal{X}^n}\prod_{l=1}^{k}W(y_l|x_l)\prod_{l=k+2}^{n}\widetilde{W}(y_l|x_l)\Big(W(y_{k+1}|x_{k+1})-\widetilde{W}(y_{k+1}|x_{k+1})\Big)\\
&\qquad\times E(x^n|v)P_{V|U}(v|u)P_U(u)\Big|\\
&\quad\leq\sum_{v\in\mathcal{V}}\sum_{u\in\mathcal{U}}\sum_{y^n\in\mathcal{Y}^n}\sum_{x^n\in\mathcal{X}^n}\prod_{l=1}^{k}W(y_l|x_l)\prod_{l=k+2}^{n}\widetilde{W}(y_l|x_l)\Big|W(y_{k+1}|x_{k+1})-\widetilde{W}(y_{k+1}|x_{k+1})\Big|\\
&\qquad\times E(x^n|v)P_{V|U}(v|u)P_U(u)\\
&\quad=\sum_{v\in\mathcal{V}}\sum_{u\in\mathcal{U}}\sum_{x^n\in\mathcal{X}^n}\Big(\sum_{y^n\in\mathcal{Y}^n}\prod_{l=1}^{k}W(y_l|x_l)\prod_{l=k+2}^{n}\widetilde{W}(y_l|x_l)\Big|W(y_{k+1}|x_{k+1})-\widetilde{W}(y_{k+1}|x_{k+1})\Big|\Big)\\
&\qquad\times E(x^n|v)P_{V|U}(v|u)P_U(u)
\end{aligned}$$

$$
\begin{aligned}
&= \sum_{u\in\mathcal{U}} \sum_{x^n\in\mathcal{X}^n} \sum_{y_{k+1}\in\mathcal{Y}} \left| W(y_{k+1}|x_{k+1}) - \widetilde{W}(y_{k+1}|x_{k+1}) \right| \\
&\quad \times E(x^n|v) P_{V|U}(v|u) P_U(u) \\
&< \epsilon \sum_{v\in\mathcal{V}} \sum_{u\in\mathcal{U}} \sum_{x^n\in\mathcal{X}^n} E(x^n|v) P_{V|U}(v|u) P_U(u) = \epsilon.
\end{aligned}
$$

Which shows that the total variation between the joint probability distribution $P_{UVY^k\widetilde{Y}^n_{k+1}}$ and $P_{UVY^{k+1}\widetilde{Y}^n_{k+2}}$ is smaller than ϵ. Then by Lemma 9 it holds

$$
\left| H(Y_{k+1}|\widetilde{Y}^n_{k+2}Y_1^kU) - H(\widetilde{Y}_{k+1}|\widetilde{Y}^n_{k+2}Y_1^kU) \right| < 2\epsilon \log|\mathcal{Y}| + 2H_2(\epsilon) \tag{3.8}
$$

and

$$
\begin{aligned}
&\left| H(VY_{k+1}|\widetilde{Y}^n_{k+2}Y_1^kU) - H(V\widetilde{Y}_{k+1}|\widetilde{Y}^n_{k+2}Y_1^kU) \right| \\
&\qquad = \Big| H(V|\widetilde{Y}^n_{k+2}Y_1^kU) + H(Y_{k+1}|V\widetilde{Y}^n_{k+2}Y_1^kU) \\
&\qquad\quad - H(V|\widetilde{Y}^n_{k+2}Y_1^kU) - H(\widetilde{Y}_{k+1}|V\widetilde{Y}^n_{k+2}Y_1^kU) \Big| \\
&\qquad = \left| H(Y_{k+1}|V\widetilde{Y}^n_{k+2}Y_1^kU) - H(\widetilde{Y}_{k+1}|V\widetilde{Y}^n_{k+2}Y_1^kU) \right| \\
&\qquad < 2\epsilon \log|\mathcal{Y}| + 2H_2(\epsilon)
\end{aligned} \tag{3.9}
$$

Inserting (3.8) and (3.9) into (3.7) we obtain

$$
\left| I(V; Y_1^{k+1}\widetilde{Y}^n_{k+2}|U) - I(V; Y_1^k\widetilde{Y}^n_{k+1}|U) \right| \leq 4\epsilon \log|\mathcal{Y}| + 4H_2(\epsilon) =: \delta_2(\epsilon, |\mathcal{Y}|) \tag{3.10}
$$

This gives in particular the following upper bound for the difference between $I(V;Y^n|U)$ and $I(V;\widetilde{Y}^n|U)$

$$
\begin{aligned}
\left| I(V;Y^n|U) - I(V;\widetilde{Y}^n|U) \right| &\leq \sum_{k=0}^{n-1} \left| I(V; Y_1^{k+1}\widetilde{Y}^n_{k+2}|U) - I(V; Y_1^k\widetilde{Y}^n_{k+1}|U) \right| \\
&\leq n\delta_2(\epsilon, |\mathcal{Y}|)
\end{aligned}
$$

proving the lemma. □

Remark 14. *Note that the right-hand side of (3.5) depends only on the size of the output alphabet $\mathcal{Y}$, and is independent of the size of the auxiliary alphabets $\mathcal{U}$ and $\mathcal{V}$, the conditional distribution $P_{V|U}$ and the chosen stochastic encoder E.*

Lemma 11. *Let $\epsilon \in (0,1)$ and $n \in \mathbb{N}$. Let $\mathfrak{W}_1$ and $\mathfrak{W}_2$ be two compound BCCs and consider random variables satisfying the Markov chain relationship $U - V - X^n$. If*

$$
D(\mathfrak{W}_1, \mathfrak{W}_2) \leq \epsilon
$$

then it holds that

$$D_R(\mathcal{R}_n(\mathfrak{W}_1, U, V, X^n), \mathcal{R}_n(\mathfrak{W}_2, U, V, X^n)) \leq \delta(\epsilon, |\mathcal{Y}|, |\mathcal{Z}|)$$

with $\delta(\epsilon, |\mathcal{Y}|, |\mathcal{Z}|) = \delta'(\epsilon, |\mathcal{Y}|, |\mathcal{Z}|) + \delta''(\epsilon, |\mathcal{Y}|, |\mathcal{Z}|)$, $\delta'(\epsilon, |\mathcal{Y}|, |\mathcal{Z}|) := 4H_2(\epsilon) + 4\epsilon \max\{\log|\mathcal{Y}|, \log|\mathcal{Z}|\}$ *and* $\delta''(\epsilon, |\mathcal{Y}|, |\mathcal{Z}|) := 4\epsilon \log|\mathcal{Y}||\mathcal{Z}| + 8H_2(\epsilon)$.

Proof. The regions $\mathcal{R}_n(\mathfrak{W}_1, U, V, X^n) \in \mathbb{R}_+^2$ and $\mathcal{R}_n(\mathfrak{W}_2, U, V, X^n) \in \mathbb{R}_+^2$ are rectangles described by the rates $(R_{0,\mathcal{S}_1}, R_{1,\mathcal{S}_1})$ and $(R_{0,\mathcal{S}_2}, R_{1,\mathcal{S}_2})$ satisfying (3.2) and (3.3) respectively. For $i = 1, 2$, we define $A_{0_{\mathcal{S}_i}}$ and $A_{1_{\mathcal{S}_i}}$ as

$$\begin{aligned} A_{0_{\mathcal{S}_i}} &= \max_{(R_{0,\mathcal{S}_i}, R_{1,\mathcal{S}_i}) \in \mathcal{R}_n(\mathfrak{W}_i, U, V, X^n)} R_{0,\mathcal{S}_i} \\ A_{1_{\mathcal{S}_i}} &= \max_{(R_{0,\mathcal{S}_i}, R_{1,\mathcal{S}_i}) \in \mathcal{R}_n(\mathfrak{W}_i, U, V, X^n)} R_{1,\mathcal{S}_i}. \end{aligned}$$

Note that both regions are rectangles sharing the corner point $(0, 0)$. Therefore, the longest distance between these two sets is given by the corner points $(A_{0_{\mathcal{S}_1}}, A_{1_{\mathcal{S}_1}})$ and $(A_{0_{\mathcal{S}_2}}, A_{1_{\mathcal{S}_2}})$, i.e.,

$$\begin{aligned} &D_R(\mathcal{R}_n(\mathfrak{W}_1, U, V, X^n), \mathcal{R}_n(\mathfrak{W}_2, U, V, X^n)) \\ &\quad = |A_{0_{\mathcal{S}_1}} - A_{0_{\mathcal{S}_2}}| + |A_{1_{\mathcal{S}_1}} - A_{1_{\mathcal{S}_2}}|. \end{aligned}$$

We first analyze the difference between the maximum achievable common rates, i.e., $|A_{0_{\mathcal{S}_1}} - A_{0_{\mathcal{S}_2}}|$ and then the difference between the maximum achievable confidential rates, i.e., $|A_{1_{\mathcal{S}_1}} - A_{1_{\mathcal{S}_2}}|$.

Common Message Rate

There are four cases that may occur:

1) $A_{0_{\mathcal{S}_1}} = \frac{1}{n} \inf_{s_1 \in \mathcal{S}_1} I(U; Y_{s_1}^n)$
$A_{0_{\mathcal{S}_2}} = \frac{1}{n} \inf_{s_2 \in \mathcal{S}_2} I(U; Y_{s_2}^n)$

2) $A_{0_{\mathcal{S}_1}} = \frac{1}{n} \inf_{s_1 \in \mathcal{S}_1} I(U; Z_{s_1}^n)$
$A_{0_{\mathcal{S}_2}} = \frac{1}{n} \inf_{s_2 \in \mathcal{S}_2} I(U; Z_{s_2}^n)$

3) $A_{0_{\mathcal{S}_1}} = \frac{1}{n} \inf_{s_1 \in \mathcal{S}_1} I(U; Y_{s_1}^n)$
$A_{0_{\mathcal{S}_2}} = \frac{1}{n} \inf_{s_2 \in \mathcal{S}_2} I(U; Z_{s_2}^n)$

4) $A_{0_{\mathcal{S}_1}} = \frac{1}{n} \inf_{s_1 \in \mathcal{S}_1} I(U; Z_{s_1}^n)$
$A_{0_{\mathcal{S}_2}} = \frac{1}{n} \inf_{s_2 \in \mathcal{S}_2} I(U; Y_{s_2}^n)$

For Case 1), we have

$$\begin{aligned}&\left|A_{0_{\mathcal{S}_1}} - A_{0_{\mathcal{S}_2}}\right| \\ &\quad = \left|\frac{1}{n}\inf_{s_1\in\mathcal{S}_1} I(U;Y_{s_1}^n) - \frac{1}{n}\inf_{s_2\in\mathcal{S}_2} I(U;Y_{s_2}^n)\right|. \end{aligned} \tag{3.11}$$

Let $\eta > 0$ be arbitrary. There exists an $\hat{s}_1 = \hat{s}_1(\eta)$ such that

$$\inf_{s_1\in\mathcal{S}_1} I(U;Y_{s_1}^n) \geq I(U;Y_{\hat{s}_1}^n) - \eta. \tag{3.12}$$

Since $D(\mathfrak{W}_1, \mathfrak{W}_2) < \epsilon$, there is an $\hat{s}_2 = \hat{s}_2(\hat{s}_1)$ such that

$$d(W_{\hat{s}_1}, W_{\hat{s}_2}) < \epsilon. \tag{3.13}$$

We can now apply Lemma 10. (We let U in (3.5) be a constant and we let U in (3.11) take the role of V in (3.5).) By (3.13), we have

$$\left|I(U;Y_{\hat{s}_1}^n) - I(U;Y_{\hat{s}_2}^n)\right| \leq n\delta_2(\epsilon, |\mathcal{Y}|). \tag{3.14}$$

Combining (3.12) and (3.14) we obtain

$$\begin{aligned}\inf_{s_1\in\mathcal{S}_1} I(U;Y_{s_1}^n) &\geq I(U;Y_{\hat{s}_2}^n) - n\delta_2(\epsilon, |\mathcal{Y}|) - \eta \\ &\geq \inf_{s_2\in\mathcal{S}_2} I(U;Y_{s_2}^n) - n\delta_2(\epsilon, |\mathcal{Y}|) - \eta.\end{aligned}$$

Since this inequality holds for all $\eta > 0$, we then obtain

$$\inf_{s_1\in\mathcal{S}_1} I(U;Y_{s_1}^n) > \inf_{s_2\in\mathcal{S}_2} I(U;Y_{s_2}^n) - n\delta_2(\epsilon, |\mathcal{Y}|).$$

By changing the roles of $\mathcal{S}_1$ and $\mathcal{S}_2$ in the previous derivation, we get

$$\left|\inf_{s_1\in\mathcal{S}_1} I(U;Y_{s_1}^n) - \inf_{s_2\in\mathcal{S}_2} I(U;Y_{s_2}^n)\right| \leq n\delta_2(\epsilon, |\mathcal{Y}|).$$

Using the same line of argument as for Case 1), for Case 2), we have

$$\left|\inf_{s_1\in\mathcal{S}_1} I(U;Z_{s_1}^n) - \inf_{s_2\in\mathcal{S}_2} I(U;Z_{s_2}^n)\right| \leq n\delta_2(\epsilon, |\mathcal{Z}|).$$

In Case 3) and Case 4) we have that for one compound BCC the maximum achievable common rate depends on the random variable Y and for the other, the maximum achievable common rate depends on the random variable Z. We first study Case 3). We have

$$\begin{aligned}B_{0_{\mathcal{S}_1}} &= \frac{1}{n}\inf_{s_1\in\mathcal{S}_1} I(U;Z_{s_1}^n) \geq \frac{1}{n}\inf_{s_1\in\mathcal{S}_1} I(U;Y_{s_1}^n) = A_{0_{\mathcal{S}_1}} \\ B_{0_{\mathcal{S}_2}} &= \frac{1}{n}\inf_{s_2\in\mathcal{S}_2} I(U;Y_{s_2}^n) \geq \frac{1}{n}\inf_{s_2\in\mathcal{S}_2} I(U;Z_{s_2}^n) = A_{0_{\mathcal{S}_2}}.\end{aligned}$$

We have six possibilities to relate the two previous inequalities:

I) $B_{0_{\mathcal{S}_1}} \geq A_{0_{\mathcal{S}_1}} \geq B_{0_{\mathcal{S}_2}} \geq A_{0_{\mathcal{S}_2}}$ and Lemma 10 implies

$$\left|A_{0_{\mathcal{S}_1}} - A_{0_{\mathcal{S}_2}}\right| \leq \left|B_{0_{\mathcal{S}_1}} - A_{0_{\mathcal{S}_2}}\right| \leq \delta_2(\epsilon, |\mathcal{Z}|)$$

II) $B_{0_{\mathcal{S}_1}} \geq B_{0_{\mathcal{S}_2}} \geq A_{0_{\mathcal{S}_1}} \geq A_{0_{\mathcal{S}_2}}$ implying

$$|A_{0_{\mathcal{S}_1}} - A_{0_{\mathcal{S}_2}}| \leq |B_{0_{\mathcal{S}_1}} - A_{0_{\mathcal{S}_2}}| \leq \delta_2(\epsilon, |\mathcal{Z}|)$$

III) $B_{0_{\mathcal{S}_1}} \geq B_{0_{\mathcal{S}_2}} \geq A_{0_{\mathcal{S}_2}} \geq A_{0_{\mathcal{S}_1}}$ implying

$$|A_{0_{\mathcal{S}_1}} - A_{0_{\mathcal{S}_2}}| \leq |A_{0_{\mathcal{S}_1}} - B_{0_{\mathcal{S}_2}}| \leq \delta_2(\epsilon, |\mathcal{Y}|)$$

IV) $B_{0_{\mathcal{S}_2}} \geq A_{0_{\mathcal{S}_2}} \geq B_{0_{\mathcal{S}_1}} \geq A_{0_{\mathcal{S}_1}}$ implying

$$|A_{0_{\mathcal{S}_1}} - A_{0_{\mathcal{S}_2}}| \leq |A_{0_{\mathcal{S}_1}} - B_{0_{\mathcal{S}_2}}| \leq \delta_2(\epsilon, |\mathcal{Y}|)$$

V) $B_{0_{\mathcal{S}_2}} \geq B_{0_{\mathcal{S}_1}} \geq A_{0_{\mathcal{S}_2}} \geq A_{0_{\mathcal{S}_1}}$ implying

$$|A_{0_{\mathcal{S}_1}} - A_{0_{\mathcal{S}_2}}| \leq |A_{0_{\mathcal{S}_1}} - B_{0_{\mathcal{S}_2}}| \leq \delta_2(\epsilon, |\mathcal{Y}|)$$

VI) $B_{0_{\mathcal{S}_2}} \geq B_{0_{\mathcal{S}_1}} \geq A_{0_{\mathcal{S}_1}} \geq A_{0_{\mathcal{S}_2}}$ implying

$$|A_{0_{\mathcal{S}_1}} - A_{0_{\mathcal{S}_2}}| \leq |A_{0_{\mathcal{S}_2}} - B_{0_{\mathcal{S}_1}}| \leq \delta_2(\epsilon, |\mathcal{Z}|)$$

We use the same line of argument for Case 4) as for Case 3) to bound the distance between the two maximum achievable common rates. It then holds for all cases that

$$\begin{aligned} |A_{0_{\mathcal{S}_1}} - A_{0_{\mathcal{S}_2}}| &\leq \max\{\delta_2(\epsilon, |\mathcal{Y}|), \delta_2(\epsilon, |\mathcal{Y}|)\} \\ &= 4H_2(\epsilon) + 4\epsilon \max\{\log|\mathcal{Y}|, \log|\mathcal{Z}|\}. \end{aligned}$$

Confidential Message Rate

Using the same line of argument as in Case 1) for the common-message rate, we obtain

$$\begin{aligned} |A_{1_{\mathcal{S}_1}} - A_{1_{\mathcal{S}_2}}| = \Big|&\frac{1}{n}\inf_{s_1\in\mathcal{S}_1} I(V;Y_{s_1}^n|U) - \frac{1}{n}\sup_{s_1\in\mathcal{S}_1} I(V;Z_{s_1}^n|U) \\ &- \frac{1}{n}\inf_{s_2\in\mathcal{S}_2} I(V;Y_{s_2}^n|U) + \frac{1}{n}\sup_{s_2\in\mathcal{S}_2} I(V;Z_{s_2}^n|U)\Big| \\ \leq\ &\frac{1}{n}\Big|\inf_{s_1\in\mathcal{S}_1} I(V;Y_{s_1}^n|U) - \inf_{s_2\in\mathcal{S}_2} I(V;Y_{s_2}^n|U)\Big| \\ &+ \frac{1}{n}\Big|\inf_{s_2\in\mathcal{S}_2} I(V;Z_{s_2}^n|U) - \inf_{s_1\in\mathcal{S}_1} I(V;Z_{s_1}^n|U)\Big| \\ \leq\ &\delta_2(\epsilon, |\mathcal{Y}|) + \delta_2(\epsilon, |\mathcal{Z}|) \\ \leq\ &4\epsilon \log|\mathcal{Y}||\mathcal{Z}| + 8H_2(\epsilon). \end{aligned}$$

□

Theorem 6. *Let $\epsilon \in (0,1)$. Let $\mathfrak{W}_1$ and $\mathfrak{W}_2$ be two compound BCCs. If*

$$D(\mathfrak{W}_1, \mathfrak{W}_2) \leq \epsilon \tag{3.15}$$

then it holds that

$$D_R(\mathcal{C}(\mathfrak{W}_1), \mathcal{C}(\mathfrak{W}_2)) \leq \delta(\epsilon, |\mathcal{Y}|, |\mathcal{Z}|).$$

Proof. We define the sets $\mathcal{D}_1, \mathcal{B}_1 \subset \mathbb{R}_+^2$ and

$$\begin{aligned}\mathcal{D}_1 &= \bigcup_{n\in\mathbb{N}} \bigcup_{U-V-X^n} \mathcal{R}_n(\mathfrak{W}_1, U, V, X^n)\\ \mathcal{B}_1 &= \mathcal{C}(\mathfrak{W}_1) \backslash \bigcup_{n\in\mathbb{N}} \bigcup_{U-V-X^n} \mathcal{R}_n(\mathfrak{W}_1, U, V, X^n)\end{aligned}$$

with random variables $U-V-X^n$ forming a Markov chain. Let $(R_{0_{\mathcal{S}_1}}, R_{1_{\mathcal{S}_1}}) \in \mathcal{D}_1$. Then there exists an $n \in \mathbb{N}$ and random variables satisfying the Markov chain relationship $\hat{U} - \hat{V} - \hat{X}^n$ such that $(R_{0_{\mathcal{S}_1}}, R_{1_{\mathcal{S}_1}}) \in \mathcal{R}_n(\mathfrak{W}_1, \hat{U}, \hat{V}, \hat{X}^n)$. From Lemma 11 and (3.15) we have that

$$D_R(\mathcal{R}_n(\mathfrak{W}_1, \hat{U}, \hat{V}, \hat{X}^n), \mathcal{R}_n(\mathfrak{W}_2, \hat{U}, \hat{V}, \hat{X}^n)) \leq \delta(\epsilon, |\mathcal{Y}|, |\mathcal{Z}|).$$

This means that there exists a rate pair $(R_{0_{\mathcal{S}_2}}(R_{0_{\mathcal{S}_1}}), R_{1_{\mathcal{S}_2}}(R_{1_{\mathcal{S}_1}})) \in \mathcal{R}_n(\mathfrak{W}_2, \hat{U}, \hat{V}, \hat{X}^n)$ such that

$$|R_{0_{\mathcal{S}_1}} - R_{0_{\mathcal{S}_2}}| + |R_{1_{\mathcal{S}_1}} - R_{1_{\mathcal{S}_2}}| \leq \delta(\epsilon, |\mathcal{Y}|, |\mathcal{Z}|).$$

Let $(\hat{R}_{0_{\mathcal{S}_1}}, \hat{R}_{1_{\mathcal{S}_1}}) \in \mathcal{B}_1$. Then there exist two rate pairs $(\dot{R}_{0_{\mathcal{S}_1}}, \dot{R}_{1_{\mathcal{S}_1}}), (\tilde{R}_{0_{\mathcal{S}_1}}, \tilde{R}_{1_{\mathcal{S}_1}}) \in \mathcal{D}_1$ such that

$$\begin{aligned}\hat{R}_{0_{\mathcal{S}_1}} &= \lambda \dot{R}_{0_{\mathcal{S}_1}} + (1-\lambda)\tilde{R}_{0_{\mathcal{S}_1}}\\ \hat{R}_{1_{\mathcal{S}_1}} &= \lambda \dot{R}_{1_{\mathcal{S}_1}} + (1-\lambda)\tilde{R}_{1_{\mathcal{S}_1}}\end{aligned}$$

for some $\lambda \in (0,1)$. For each $(\dot{R}_{0_{\mathcal{S}_1}}, \dot{R}_{1_{\mathcal{S}_1}})$ and $(\tilde{R}_{0_{\mathcal{S}_1}}, \tilde{R}_{1_{\mathcal{S}_1}})$ there exist random variables satisfying the Markov chain relation $\dot{U} - \dot{V} - \dot{X}^n$ and $\tilde{U} - \tilde{V} - \tilde{X}^n$ such that $(\dot{R}_{0_{\mathcal{S}_1}}, \dot{R}_{1_{\mathcal{S}_1}}) \in \mathcal{R}_n(\mathfrak{W}_1, \dot{U}, \dot{V}, \dot{X}^n)$ and $(\tilde{R}_{0_{\mathcal{S}_1}}, \tilde{R}_{1_{\mathcal{S}_1}}) \in \mathcal{R}_n(\mathfrak{W}_1, \tilde{U}, \tilde{V}, \tilde{X}^n)$. Then from Lemma 11 and (3.15) we have that there exist rate pairs $(\dot{R}_{0_{\mathcal{S}_2}}(\dot{R}_{0_{\mathcal{S}_1}}), \dot{R}_{1_{\mathcal{S}_2}}(\dot{R}_{1_{\mathcal{S}_1}})) \in \mathcal{R}_n(\mathfrak{W}_2, \dot{U}, \dot{V}, \dot{X}^n)$ and $(\tilde{R}_{0_{\mathcal{S}_2}}(\tilde{R}_{0_{\mathcal{S}_1}}), \tilde{R}_{1_{\mathcal{S}_2}}(\tilde{R}_{1_{\mathcal{S}_1}})) \in \mathcal{R}_n(\mathfrak{W}_2, \tilde{U}, \tilde{V}, \tilde{X}^n)$ such that

$$\begin{aligned}|\dot{R}_{0_{\mathcal{S}_1}} - \dot{R}_{0_{\mathcal{S}_2}}| + |\dot{R}_{1_{\mathcal{S}_1}} - \dot{R}_{1_{\mathcal{S}_2}}| &\leq \delta(\epsilon, |\mathcal{Y}|, |\mathcal{Z}|)\\ |\tilde{R}_{0_{\mathcal{S}_1}} - \tilde{R}_{0_{\mathcal{S}_2}}| + |\tilde{R}_{1_{\mathcal{S}_1}} - \tilde{R}_{1_{\mathcal{S}_2}}| &\leq \delta(\epsilon, |\mathcal{Y}|, |\mathcal{Z}|).\end{aligned}$$

Then there is a rate pair $(\hat{R}_{0_{\mathcal{S}_2}}, \hat{R}_{1_{\mathcal{S}_2}}) \in \mathcal{C}(\mathfrak{W}_2)$ with

$$\begin{aligned}\hat{R}_{0_{\mathcal{S}_2}} &= \lambda \dot{R}_{0_{\mathcal{S}_2}} + (1-\lambda)\tilde{R}_{0_{\mathcal{S}_2}}\\ \hat{R}_{1_{\mathcal{S}_2}} &= \lambda \dot{R}_{1_{\mathcal{S}_2}} + (1-\lambda)\tilde{R}_{1_{\mathcal{S}_2}}.\end{aligned}$$

Further we have

$$\begin{aligned}|\hat{R}_{0_{\mathcal{S}_1}} - \hat{R}_{0_{\mathcal{S}_2}}| &= |\lambda \dot{R}_{0_{\mathcal{S}_2}} + (1-\lambda)\tilde{R}_{0_{\mathcal{S}_2}} \\ &\quad - \lambda \dot{R}_{0_{\mathcal{S}_1}} + (1-\lambda)\tilde{R}_{0_{\mathcal{S}_1}}| \\ &\leq \lambda |\dot{R}_{0_{\mathcal{S}_1}} - \dot{R}_{0_{\mathcal{S}_2}}| + (1-\lambda)|\tilde{R}_{0_{\mathcal{S}_1}} - \tilde{R}_{0_{\mathcal{S}_2}}| \\ &\leq \delta'(\epsilon, |\mathcal{Y}|, |\mathcal{Z}|)\end{aligned}$$

and using the same line of argument

$$|\hat{R}_{1_{\mathcal{S}_1}} - \hat{R}_{1_{\mathcal{S}_2}}| \leq \delta''(\epsilon, |\mathcal{Y}|, |\mathcal{Z}|).$$

This leads us to the following result:

$$|\hat{R}_{0_{\mathcal{S}_1}} - \hat{R}_{0_{\mathcal{S}_2}}| + |\hat{R}_{1_{\mathcal{S}_1}} - \hat{R}_{1_{\mathcal{S}_2}}| \leq \delta(\epsilon, |\mathcal{Y}|, |\mathcal{Z}|).$$

We can conclude that for every rate pair $(R_{0_{\mathcal{S}_1}}, R_{1_{\mathcal{S}_1}}) \in \mathcal{C}(\mathfrak{W}_1)$ we can find a rate pair $(R_{0_{\mathcal{S}_2}}(R_{0_{\mathcal{S}_1}}), R_{1_{\mathcal{S}_2}}(R_{1_{\mathcal{S}_1}})) \in \mathcal{C}(\mathfrak{W}_2)$ such that

$$|R_{0_{\mathcal{S}_1}} - R_{0_{\mathcal{S}_2}}| + |R_{1_{\mathcal{S}_1}} - R_{1_{\mathcal{S}_2}}| \leq \delta(\epsilon, |\mathcal{Y}|, |\mathcal{Z}|). \tag{3.16}$$

We use the same line of argument to show that for every rate pair $(R_{0_{\mathcal{S}_2}}, R_{1_{\mathcal{S}_2}}) \in \mathcal{C}(\mathfrak{W}_2)$ there is a rate pair $(R_{0_{\mathcal{S}_1}}(R_{0_{\mathcal{S}_2}}), R_{1_{\mathcal{S}_1}}(R_{1_{\mathcal{S}_2}})) \in \mathcal{C}(\mathfrak{W}_1)$ such that (3.16) holds. This completes the proof. □

3.3 Conclusions

We have shown that the compound BCC model is robust, i.e., small changes in the uncertainty set lead to small changes in the capacity region, which is desirable.

Let us see what happens when the user's CSI is reduced further. For example, the arbitrarily varying BCC is described by the same uncertainty set as the compound BCC, but in addition, the actual channel realization varies from channel use to channel use in an arbitrary fashion. The arbitrarily varying BCC can be used for example to model the presence of jamming; see [43]. This may lead the channel to "emulate" a valid input, impeding the legitimate receiver to decide on the correct codeword. This property is known as symmetrizability; see [43, Sec. III, Def. 5].

We adapt the AVC example from [43, Sec. V] to the channel of receiver 1 of the arbitrarily varying BCC, where the input and the output alphabets are of size $|\mathcal{X}| = 2$ and $|\mathcal{Y}| = 3$, respectively, and the uncertainty set consists of only two elements, i.e., $|\mathcal{S}| = 2$. The AVC to receiver 1 is given by $\mathcal{W}(\lambda) = \{W_1(\lambda), W_2(\lambda)\}$ with

$$W_1(\lambda) = \begin{pmatrix} 1 & 0 & 0 \\ 1 & \lambda & 1-\lambda \end{pmatrix} \text{ and } W_2(\lambda) = \begin{pmatrix} \lambda & 0 & 1-\lambda \\ 0 & 1 & 0 \end{pmatrix}$$

where $\lambda \in [0, 1]$. The AVC $\mathcal{V}$ to receiver 2 has an output alphabet of size $|\mathcal{Z}| = 2$ and is defined as $\mathcal{V} = \{V, V\}$ with

$$V = \begin{pmatrix} \frac{1}{2} & \frac{1}{2} \\ \frac{1}{2} & \frac{1}{2} \end{pmatrix}.$$

In [43, Sec. V], it is shown that the arbitrarily varying channel $\mathcal{W}(\lambda)$ is non-symmetrizable for all $\lambda \in (0, 1]$, and symmetrizable for $\lambda = 0$, in which case the capacity region collapses to the point $(0, 0) \in \mathbb{R}_+^2$. Following the argumentation in [43, Sec. V], it can be shown that capacity region is indeed discontinuous at $\lambda = 0$.

4 Computability of the Finite State Channel Capacity with Feedback

FSCs model channels with memory where the channel output depends not only on the current channel input but also on the underlying channel state. The channel state allows the channel output to implicitly depend on previous channel inputs and outputs. FSCs are of significant interest as they allow one to model certain types of channel variations appearing in wireless communications including, e.g., flat fading and ISI [7].

Determining the capacity of FSCs is a very difficult task. For instance, the trapdoor channel is relatively simple to describe: it comprises binary input and output alphabets, with the channel having two states, $0, 1$. When the channel is in state 0, it operates noiselessly; otherwise, when in state 1, it behaves as a BSC with a crossover probability of $\epsilon = 0.5$. Despite its straightforward description, only a lower bound [44] and an upper bound [45] for the capacity are known. For general FSCs, only a general formula based on the inf-information rate has been established in [46]. Moreover, in [8] it was shown that the FSC capacity is not a computable function.

In [47], it was shown that feedback does not increase the capacity of a DMC. However, the zero error capacity for a channel with feedback might be greater in some cases, while there is still no closed-form formula for the zero error capacity without feedback so far. The feedback capacity for Markov channels without ISI was studied in[48] and general FSCs in [49, 50, 51]. Only a multi-letter characterization of the capacity is known in these cases to date.

In recent years, there has been a growing interest in computing the capacity function for FSCs with feedback. The feedback capacity of FSCs was first formulated as a dynamic program for Markov channels without ISI in [48] and [52]. This formulation has also been used to compute the feedback capacity of the trapdoor channel [53], the binary Ising channel [54], the input-constrained BSC [55] and the input-constrained binary erasure channel [56]. In [57], reinforcement learning algorithms have been proposed to estimate the feedback capacity of a class of unifilar FSCs.

4.1 Finite State Channels with Feedback

In this section we introduce the concept of discrete FSCs and present the capacity results with feedback known to date.

Let $\mathcal{X}$, $\mathcal{Y}$, and $\mathcal{S}$ be finite input, output, and state sets, respectively. FSCs are described by the following probability law:

$$P(y_n, s_n | x_n, s_{n-1}) \in \mathcal{P}(\mathcal{Y} \times \mathcal{S} | \mathcal{X} \times \mathcal{S}), \tag{4.1}$$

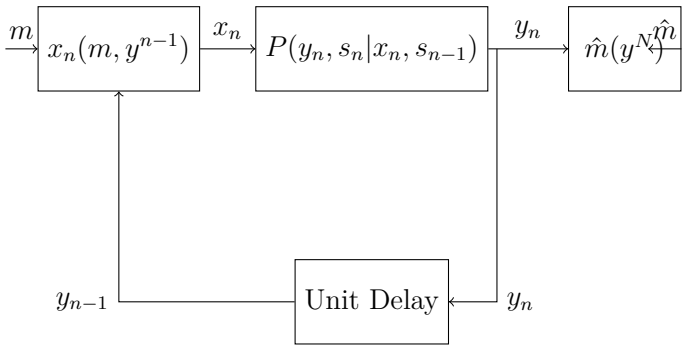

Figure 4.1: Finite state channel with deterministic feedback y_{n-1}.

where $y_n \in \mathcal{Y}$ and $s_n \in \mathcal{S}$ are the output and state of the channel at time instant n whose probability distribution depends on the input $x_n \in \mathcal{X}$ at time instant n and on the previous state $s_{n-1} \in \mathcal{S}$ at time instant $n-1$. We consider the transmission in the presence of feedback. The feedback at time instant $n \in \mathbb{N}$ is the last output symbol of the channel, i.e., y_{n-1}; see Fig. 4.1.

For a fixed blocklength n, the probability of the output sequence y^n and the final state s_n at time instant n given an input sequence x^n and an initial state s_0 is given by

$$P^n(y^n, s_n|x^n, s_0) = \sum_{s_{n-1}\in\mathcal{S}} P(y_n, s_n|x_n, s_{n-1}P^{n-1}(y^{n-1}, s_{n-1}|x^{n-1}, s_0). \tag{4.2}$$

In this work we focus on unifilar FSCs.

Definition 40. *An FSC is called* unifilar *if there exists a time-invariant function* $f\colon \mathcal{S}\times\mathcal{X}\times\mathcal{Y}\to\mathcal{S}$ *such that the state evolves according to the equation*

$$s_n = f(s_{n-1}, x_n, y_n).$$

Remark 15. *The probability law of a unifilar FSC is described by*

$$\begin{aligned} P(y_n, s_n|x_n, s_{n-1}) &= W(y_n|x_n, s_{n-1})p(s_n|y_n, x_n, s_{n-1}) \\ &= W(y_n|x_n, s_{n-1})\mathbb{1}(s_n = f(s_{n-1}, x_n, y_n)). \end{aligned} \tag{4.3}$$

From (4.3), we see that we only need the channel $W \in \mathcal{P}(\mathcal{Y}|\mathcal{X}\times\mathcal{S})$ *and the state transition function* f *to fully describe a unifilar FSC.*

The capacity of general FSCs with deterministic feedback was derived in [50]. Here, we study the algorithmic behavior of the capacity depending on the parameters $\{W, f, s_0\}$. To this aim, we express the FSC feedback capacity of unifilar FSCs as a function of $\{W, f, s_0\}$, i.e., $C_{FB}(\{W, f, s_0\})$.

To describe the feedback capacity function, we introduce the upper and lower capacity as follows:

$$\begin{aligned} \underline{C}_{FB}(\{W, f\}) &= \lim_{N\to\infty}\frac{1}{N}\max_{p(x^N\|y^{N-1})}\min_{s_0} I(X^N\to Y^N|s_0), \\ \overline{C}_{FB}(\{W, f\}) &= \lim_{N\to\infty}\frac{1}{N}\max_{p(x^N\|y^{N-1})}\max_{s_0} I(X^N\to Y^N|s_0). \end{aligned}$$

Theorem 7. *[50] For any unifilar FSC with deterministic feedback, the capacity is bounded as*

$$\underline{C}_{FB}(\{W,f\}) \leq C_{FB}(\{W,f,s_0\}) \leq \overline{C}_{FB}(\{W,f\}).$$

Indecomposable FSCs are channels for which the initial state effect on the capacity vanishes with time. To define indecomposable FSCs we set

$$q^n(s_n|x^n,s_0) = \sum_{y^n \in \mathcal{Y}^n} P^n(y^n,s_n|x^n,s_0).$$

Definition 41. *An FSC is called* indecomposable *if for every $\epsilon > 0$ there exists an $n_0 \in \mathbb{N}$ such that for all $n \geq n_0$ we have $|q^n(s_n|x^n,s_0) - q^n(s_n|x^n,s_0')| \leq \epsilon$ for all $s_n \in \mathcal{S}$, $x^n \in \mathcal{X}^n$, $s_0 \in \mathcal{S}$, and $s_0' \in \mathcal{S}$.*

For a strongly connected unifilar channel, the feedback capacity has a simpler expression. We next introduce the definition of strongly connected FSCs.

Definition 42. *A finite state channel is said to be* strongly connected *if for any state s there exists an integer T and an input distribution of the form $\{p(x_n|s_{n-1})\}_{n=1}^T$ such that the probability that the channel reaches state s for any starting state s' in less than T time-steps is positive, i.e.*

$$\sum_{n=1}^{T} \Pr(S_n = s|S_0 = s') > 0, \quad \forall s \in \mathcal{S}, \forall s' \in \mathcal{S}.$$

Remark 16. *Strongly connected FSCs are also indecomposable FSCs. However, not every indecomposable FSC is a strongly connected FSC.*

If a unifilar FSC is strongly connected, and therefore indecomposable, then the lower and upper capacity coincide, and are equal to the capacity, i.e., $\overline{C}_{FB}(\{W,f,s_0\}) = \underline{C}_{FB}(\{W,f,s_0\}) = C_{FB}(\{W,f,s_0\})$. The capacity of indecomposable unifilar FSCs with feedback is presented in the following theorem.

Theorem 8 ([50])**.** *The capacity of an indecomposable unifilar FSC with deterministic feedback is*

$$C_{FB}(\{W,f,s_0\}) = \lim_{N\to\infty} \max_{p(x^N\|y^{N-1})} \frac{1}{N} I(X^N \to Y^N). \tag{4.4}$$

for all $s_0 \in \mathcal{S}$.

Remark 17. *The relationship in (4.4) represents the capacity of an indecomposable unifilar FSC with deterministic feedback as the limit of the sequence $\{\max_{p(x^N\|y^{N-1})} \frac{1}{N} I(X^N \to Y^N)\}_{N\in\mathbb{N}}$. This representation cannot be used to compute the number $C_{FB}(\{W,f,s_0\})$, since the speed of convergence cannot be characterized algorithmically, i.e., there is no known computable stopping criterion for the sequence $\{\max_{p(x^N\|y^{N-1})} \frac{1}{N} I(X^N \to Y^N)\}_{N\in\mathbb{N}}$. If we had a computable function $\phi : \mathbb{N} \to \mathbb{N}$, so that for every $M \in \mathbb{N}$ it computes the index $\hat{N} = \phi(M)$, such that*

$|C_{FB}(\{W,f,s_0\}) - \max_{p(x^{\hat{N}}\|y^{\hat{N}-1})} \frac{1}{\hat{N}} I(X^{\hat{N}} \to Y^{\hat{N}})| < \frac{1}{2^M}$, *then a stopping criterion would exist and we could actually use* $\max_{p(x^{\hat{N}}\|y^{\hat{N}-1})} \frac{1}{\hat{N}} I(X^{\hat{N}} \to Y^{\hat{N}})$ *as an approximation for* $C_{FB}(\{W,f,s_0\})$. *Note that no generality is lost by measuring the approximation error in the following form:* $\frac{1}{2^M}$. *One could also use many other monotonically decreasing sequences of rational numbers.*

Remark 18. *It is interesting to note that in general the capacity function is expected to have computable behavior as a function of the approximation error. However, in [10], a computable compound channel* $\{W_n\}_{n\in\mathbb{N}}$ *, i.e.* $\{W_n\}_{n\in\mathbb{N}}$*, a computable sequence of computable channels, with binary input and output alphabets, was found whose capacity* $C(\{W_n\}_{n\in\mathbb{N}})$ *was shown to be an uncomputable number. In [10], it was shown that for the compound capacity* $C(\{W_n\}_{n\in\mathbb{N}})$*, a monotonically decreasing sequence of computable numbers* $\{z_n\}_{n\in\mathbb{N}}$ *can be found that converges to the compound capacity, i.e.,* $\lim_{n\in\mathbb{N}}\{z_n\} = C(\{W_n\}_{n\in\mathbb{N}})$. *However, a stopping criterion cannot be found for these sequences, otherwise* $C(\{W_n\}_{n\in\mathbb{N}})$ *would be a computable number.*

4.2 Problem Formulation

Capacity functions arising in communication scenarios have entropic formulations. For example, for finite input and output alphabets $|\mathcal{X}| < \infty$ and $|\mathcal{Y}| < \infty$, the capacity of a DMC $W \in \mathcal{P}(\mathcal{Y}|\mathcal{X})$ is $\max_{p\in\mathcal{P}(\mathcal{X})} I(p,W)$, i.e., the maximization of an entropic function over the input probabilities [58]. Let us consider a BSC with a rational crossover probability $\epsilon \in [0,1) \cap \mathbb{Q}$. Such a channel is clearly computable, since every rational number can be exactly expressed by a digital computer. Interestingly, the capacity of a BSC with a rational crossover probability, except for $\epsilon = \{0, \frac{1}{2}\}$, is a transcendental number.

Subsets of the transcendental numbers are non-computable. Intuitively, a number $x \in \mathbb{R}$ is computable if there exists an algorithm for x, that given a desired precision, returns an approximation of the number to that precision in finitely many steps. A function $g\colon \mathbb{R} \to \mathbb{R}$ is Turing computable if there exists an algorithm that returns a computable number for every possible computable input parameter. The Blahut-Arimoto algorithm is an algorithm that takes any computable DMC as input and computes the capacity, see [4, 5]. Hence, the capacity of DMCs is a computable function.

Coming back to the FSC with feedback, as stated in Theorem 7, the capacity of general FSCs with feedback can be bounded from above and from below. Both bounds are given by multi-letter expressions. If we restrict the class of FSCs to be indecomposable and unifilar, then a mathematical expression of the capacity is known.

In Theorem 8, the capacity of indecomposable unifilar FSCs with feedback is given by a multi-letter expression. This expression is the limit of a sequence of optimization problems. At first glance, this expression looks complicated to compute. It would be desirable to have a universal algorithm for indecomposable and unifilar FSCs that takes the channel $W \in \mathcal{P}(\mathcal{Y}|\mathcal{X}\times\mathcal{S})$, the state transition function $f \in \mathcal{S}^{\mathcal{S}\times\mathcal{X}\times\mathcal{Y}}$, and the initial state $s_0 \in \mathcal{S}$ and computes the capacity in the presence of feedback. This is visualized

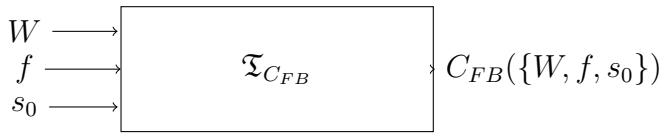

Figure 4.2: Turing machine $\mathfrak{T}_{C_{FB}}$ for capacity computation for fixed and finite alphabets $\mathcal{X}$, $\mathcal{Y}$, and $\mathcal{S}$. It takes the channel $W \in \mathcal{P}(\mathcal{Y}|\mathcal{X} \times \mathcal{S})$, the state transition function $f \in \mathcal{S}^{\mathcal{S}\times\mathcal{X}\times\mathcal{Y}}$, and the initial state $s_0 \in \mathcal{S}$ and computes the capacity in the presence of feedback $C_{FB}(\{W, f, s_0\}$.

in Fig. 4.2.

In [59], it was shown that there are computable functions whose optimal values are uncomputable numbers. In [10], as discussed previously in Remark 18, it was shown that although the compound capacity $C(\{W_n\}_{n\in\mathbb{N}})$ has a single-letter expression, i.e. $C(\{W_n\}_{n\in\mathbb{N}}) = \sup_{p\in\mathcal{P}(\mathcal{X})} \inf_{n\in\mathbb{N}} I(p, W_n)$[1], there are computable compound channels whose capacities are uncomputable numbers. In other words, the capacities of compound channels are not computable. It is of interest to know whether such a behavior can occur for the function C_{FB}. This evokes the following question:

Question 1: *For fixed and finite alphabets $\mathcal{X}$, $\mathcal{Y}$, and $\mathcal{S}$, is there an algorithm that takes a channel W, a state transition function f and an initial state s_0 as inputs and computes the feedback capacity function $C_{FB}(\{W, f, s_0\})$?*

In case the FSC feedback capacity is not a computable function, one could aim to approximate the capacity function by a computable function. We are also interested in studying whether or not it is possible to algorithmically approximate the capacity of FSCs in the presence of feedback. This leads us to the next question:

Question 2: *For fixed and finite alphabets $\mathcal{X}$, $\mathcal{Y}$, and $\mathcal{S}$, is it possible to approximate the capacity function of unifilar FSCs with feedback $C_{FB}(\{W, f, s_0\})$ by a computable function within a desired margin of error?*

Coding schemes and general achievability results provide us with lower bounds. Upper bounds are established via converse arguments. In [3], techniques to derive lower and upper bounds on the reliability function of DMCs were presented. These techniques provide an approach for finding tight upper and lower bounds that are computable on digital computers.

In practical communication scenarios (such as described in [60]) and in standard approaches, the design, optimization and standardization of communication networks simulate the behavior of coding procedures and complex protocols and thus provide achievable performance lower bounds for optimal performance. In practice, the behavior of coding procedures is always compared with upper bounds or, if possible, optimal performance. Such upper and lower bounds should be computable to enable a numerical evaluation on digital computers. This brings us to our next question:

Question 3: *For fixed and finite alphabets $\mathcal{X}$, $\mathcal{Y}$, and $\mathcal{S}$, is it possible to find arbitrarily tight computable lower and upper bounds* that depend on the parameters $\{W, f, s_0\}$ for

[1]Here the mutual information is presented as a function of the input distribution $p \in \mathcal{P}(\mathcal{X})$ and the channel $W \in \mathcal{P}(\mathcal{Y}|\mathcal{X})$.

the feedback capacity of unifilar FSCs?

In [58], the well know capacity characterization of the DMC is formulated as a maximization problem over the input distribution set $\mathcal{P}(\mathcal{X})$. The determination of the capacities of communication channels as optimization problems is frequently used, e.g., the wiretap channel [61]. There is significant interest in formulating the FSC feedback capacity, among other communication scenarios, as such an optimization problem. This leads us to the following question:

Question 4: *For fixed and finite alphabets $\mathcal{X}$, $\mathcal{Y}$, and $\mathcal{S}$, is it possible to characterize the capacity function of unifilar FSCs with feedback $C_{FB}(\{W, f, s_0\})$ as an optimization of a finite letter function?*

4.3 Computability Analysis

In this section, we explore the ability of a Turing machine to compute the feedback capacity $C_{FB}(\{W, f, s_0\})$ for any given computable tuple $\{W, f, s_0\}$, where W represents the channel, f the feedback function, and s_0 the initial state. Our focus is particularly on FSCs that are computable and serve as input parameters.

To support this investigation, we first define the set of computable probability distributions $\mathcal{P}_c(\mathcal{X})$ as all distributions p within $\mathcal{P}(\mathcal{X})$ such that $p(x)$ is a computable real number $\mathbb{R}_c$ for every x in $\mathcal{X}$. Similarly, the set of computable conditional probability distributions $\mathcal{P}_c(\mathcal{Y}|\mathcal{X})$ includes those stochastic matrices $W : \mathcal{X} \to \mathcal{P}(\mathcal{Y})$ for which $W(\cdot|x)$ is computable for each x in $\mathcal{X}$. This is essential since a Turing machine can only process computable real numbers.

Next, we introduce the concept of computable channels.

Definition 43. *A* computable channel *is a stochastic matrix $W : \mathcal{X} \to \mathcal{P}(\mathcal{Y})$ whose elements are all computable numbers, i.e., $W(y|x) \in \mathbb{R}_c$ for every $y \in \mathcal{Y}$ and $x \in \mathcal{X}$.*

In [8], it was shown that the capacity of a general class FSC is not Banach-Mazur computable. The authors used a class of channels for which the current output and current state are statistically independent, given the current input and previous state. For this particular channel class, the capacity is not affected by the presence of feedback. Here we consider the feedback capacity function and restrict the class of channels to the unifilar channels. For this particular class of channel, feedback increases the capacity.

We consider sequences of rational unifilar channels. We show that even restricting the channel set to the set of unifilar channels, the FSC feedback capacity $C_{FB}(\{W, f, s_0\})$ is not even computable according to the weakest form of computability, i.e., Banach-Mazur computability. Unfortunately, this result provides Question 1 with a negative answer.

Theorem 9. *For all $|\mathcal{X}| \geq 2$, $|\mathcal{Y}| \geq 2$, and $|\mathcal{S}| \geq 2$, the feedback capacity function $C_{FB}(\{W, f, s_0\}) \colon \mathcal{W}_c \times \mathcal{S}^{\mathcal{S}\times\mathcal{X}\times\mathcal{Y}} \times \mathcal{S} \to \mathbb{R}$ of unifilar FSCs feedback with parameters $\{W, f, s_0\}$ is not Banach-Mazur computable.*

Proof. We consider the set of computable FSCs. The capacity is a function $C_{FB} \colon \mathcal{W}_c \times \mathcal{S}^{\mathcal{S}\times\mathcal{X}\times\mathcal{Y}} \times \mathcal{S} \to \mathbb{R}_{\geq 0}$. To prove the computability we use an indirect proof. We assume

that the feedback capacity C_{FB} is Borel-Turing computable and we prove the opposite by contradiction. The proof is organized as follows:

- We design a suitable class of rational unifilar FSCs $\{W_\lambda, f\}_{\lambda\in[0,\frac{1}{2}]\cap\mathbb{Q}}$ characterized by a parameter λ.
- We consider a recursively enumerable non-recursive set $\mathcal{A}$. The elements of the recursive enumerable set are listed by a unique recursive function $\varphi_A\colon \mathbb{N}\to\mathbb{N}$. There is a Turing machine $\mathfrak{T}_\mathcal{A}$ that stops for input n if and only if $n\in\mathcal{A}$. Otherwise $\mathfrak{T}_\mathcal{A}$ runs forever.
- We generate a computable double sequence of rational numbers $\{\lambda_{n,m}\}_{n,m\in\mathbb{N}}$ using the Turing machine $\mathfrak{T}_\mathcal{A}$. We use $\{\lambda_{n,m}\}_{n,m\in\mathbb{N}}$ to construct a computable double sequence of rational unifilar FSCs $\{W_{\lambda_{n,m}}, f\}_{n,m\in\mathbb{N}}$ from the class of unifilar FSCs $\{W_\lambda, f\}_{\lambda\in[0,\frac{1}{2}]\cap\mathbb{Q}}$. This sequence of rational unifilar FSCs converges effectively to the computable sequence of computable FSCs $\{W_{\lambda_n^*}, f\}_{n\in\mathbb{N}}$. Hence, the set $\mathcal{A}$ is encoded in the sequence $\{W_{\lambda_n^*}, f\}_{n\in\mathbb{N}}$.
- We define the function $\phi(\{W,f\}) = C_{FB}(\{W,f,0\}) - C_{FB}(\{W,f,1\})$. Since C_{FB} is assumed to be a computable function, ϕ is also Borel-Turing computable. This would mean that the sequence $\{\phi(\{W_{\lambda_n^*}, f\})\}_{n\in\mathbb{N}}$ is a computable sequence of computable reals. With this computable sequence of computable reals we can build a Turing machine $\mathfrak{T}_*$ that stops for input n if and only if $\phi(\{W_{\lambda_n^*}, f\}) > 0$. Thus, $\mathfrak{T}_*$ stops if $n\in\mathcal{A}^c$, which is a contradiction, since it would mean that $\mathcal{A}$ is a recursive set. Hence, the assumption that $C_{FB}(\{W,f,s_0\})$ is computable is wrong. Even if C_{FB} would be Banach-Mazur computable, then it would solve the halting problem, which is known to be unsolvable.

To begin the proof, we first introduce a notion of distance between FSCs. In particular, for two FSCs $P_1, P_2 \in \mathcal{P}(\mathcal{Y}\times\mathcal{S}|\mathcal{X}\times\mathcal{S})$ we define the distance between P_1 and P_2 based on the total variation distance as

$$D(P_1,P_2) = \max_{s'\in\mathcal{S}}\max_{x\in\mathcal{X}}\sum_{y\in\mathcal{Y}}\sum_{s\in\mathcal{S}}|P_1(y,s|x,s') - P_2(y,s|x,s')|.$$

We begin by proving the result for $|\mathcal{X}| = |\mathcal{Y}| = |\mathcal{S}| = 2$. Then we extend it to $|\mathcal{X}|\geq 2$, $|\mathcal{Y}|\geq 2$, and $|\mathcal{S}|\geq 2$.

We consider the channel

$$W(y_n|x_n,0) = \begin{pmatrix}1 & 0\\ 0 & 1\end{pmatrix}, W(y_n|x_n,1) = \begin{pmatrix}1-\epsilon & \epsilon\\ 0 & 1\end{pmatrix} \tag{4.5}$$

for some $\epsilon\in(0,\frac{1}{2})\cap\mathbb{Q}$, i.e., for state $s_{n-1}=0$ the channel is noiseless; for $s_{n-1}=1$ it is noisy. Further, we consider the state transition function $f\colon \mathcal{S}\times\mathcal{X}\times\mathcal{Y}\to\mathcal{S}$ described by the state diagram in Fig. 4.3. The nodes represent the states and the tuple of the edges represent the input and output symbols (x_n, y_n) of the channel.

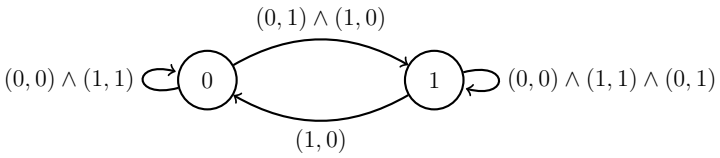

Figure 4.3: Diagram of the state transition function f.

Table 4.1: The state s_n given x_n, y_n and s_{n-1}.

x_n	0	0	0	0	1	1	1	1
y_n	0	0	1	1	0	0	1	1
s_{n-1}	0	1	0	1	0	1	0	1
s_n	0	1	1	1	1	0	0	1

The state of the channel s_n given the input x_n, output y_n and previous state s_{n-1} is also shown in Table 4.1.

The channel $\{W, f, 0\}$ corresponds to a discrete memoryless channel. Since the initial state is $s_0 = 0$, i.e. the channel $W(y_n|x_n, 0)$ is noiseless, the only two possible input-output tuples are $(0, 0)$ and $(1, 1)$. Applying the state transition function f to the tuples $(x_1, y_1, s_0) \in \{(0, 0, 0), (1, 1, 0)\}$, we get that for both tuples the next state is $s_1 = 0$. This implies that if the initial state is 0, then the state stays 0 forever. Applying Lemma 1 to the directed information for this channel, we get

$$I(X^N \to Y^N|s_0 = 0) = I(X^N; Y^N|s_0 = 0).$$

This and the fact that $W(y|x, 0)$ is a binary noiseless channel imply that the FSC feedback capacity and initial state $s_0 = 0$ is

$$C_{FB}(\{W, f, 0\}) = 1.$$

The channel $\{W, f, 1\}$ corresponds to the discrete memoryless channel $W(y|x, 1)$ with $x \in \mathcal{X}, y \in \mathcal{Y}$. Similar to the line of arguments for $\{W, f, 0\}$, if the initial state is $s_0 = 1$, then the channel has only three possible input output tuples, $(0, 0), (0, 1)$ and $(1, 1)$. Applying the state transition function f to the tuples $(x_1, y_1, s_0) \in \{(0, 0, 1), (0, 1, 1), (1, 1, 1)\}$, we get that for all three tuples the next state is $s_1 = 1$. Meaning that if the initial channel state is 1, then the channels stays in state 1 forever. Applying Lemma 1 we have that

$$I(X^N \to Y^N|s_0 = 1) = I(X^N; Y^N|s_0 = 1). \tag{4.6}$$

Note that the channel $\{W, f, 1\}$ is a Z-channel. Due to (4.6) we see that the FSC feedback capacity with initial state $s_0 = 1$ is

$$\begin{aligned} C_{FB}(\{W, f, 1\}) &= \max_{p \in \mathcal{P}(\mathcal{X})} H_2(p(1-\epsilon)) - pH_2(\epsilon) \\ &= \log_2\left(1 + 2^{-g(\epsilon)}\right) \end{aligned}$$

with $g(\epsilon) = \frac{H_2(\epsilon)}{1-\epsilon}$. The optimal input distribution is $p(0) = \left[(1-\epsilon)\left(1+2^{\frac{H_2(\epsilon)}{(1-\epsilon)}}\right)\right]^{-1}$ and $p(1) = 1 - \left[(1-\epsilon)\left(1+2^{\frac{H_2(\epsilon)}{(1-\epsilon)}}\right)\right]^{-1}$.

Next we show that $C_{FB}(\{W, f, 0\})$ and $C_{FB}(\{W, f, 1\})$ cannot be simultaneously Banach-Mazur computable. Let $\mathcal{A} \in \mathbb{N}$ be an arbitrary recursively enumerable but not recursive set. Let $\mathfrak{T}_{\mathcal{A}}$ be a Turing machine that stops for input n if and only if $n \in \mathcal{A}$. Otherwise $\mathfrak{T}_{\mathcal{A}}$ runs forever. Such a Turing machine can easily be found as argued next: Let $\varphi_{\mathcal{A}} : \mathbb{N} \to \mathbb{N}$ be a recursive function that lists all elements of the set $\mathcal{A}$ and for which $\varphi_{\mathcal{A}} : \mathbb{N} \to \mathcal{A}$ is a unique function.

Let $n \in \mathbb{N}$ be arbitrary. The Turing machine $\mathfrak{T}_{\mathcal{A}}$ with input n is defined as follows: We start with $l = 1$ and compute $\varphi_{\mathcal{A}}(1)$. If $n = \varphi_{\mathcal{A}}(1)$, then the Turing machine stops. In the other case, the Turing machine computes $\varphi_{\mathcal{A}}(2)$. Similarly, if $n = \varphi_{\mathcal{A}}(2)$, then the Turing machine stops and otherwise it continues computing the next element. It is clear that this Turing machine stops if and only if $n \in \mathcal{A}$.

Assume $C_{FB}(\{W, f, 0\})$ and $C_{FB}(\{W, f, 1\})$ are both Borel-Turing computable. For $\lambda \in (0, \frac{1}{2}] \cap \mathbb{R}_c$ we consider the channel $W_\lambda \in \mathcal{W}_c$ with

$$W_\lambda(y_n|x_n, 0) = \begin{pmatrix} 1-\lambda & \lambda \\ \lambda & 1-\lambda \end{pmatrix},$$
$$W_\lambda(y_n|x_n, 1) = \begin{pmatrix} 1-\epsilon & \epsilon \\ \lambda & 1-\lambda \end{pmatrix}. \tag{4.7}$$

For $\lambda \in [0, \frac{1}{2}] \cap \mathbb{R}_c$, both $W_\lambda(y_n|x_n, 0)$ and $W_\lambda(y_n|x_n, 1)$ are computable probability distributions.

For $\lambda = 0$, we have

$$\begin{aligned} C_{FB}(\{W_0, f, 0\}) - C_{FB}(\{W_0, f, 1\}) \\ = 1 - \log_2\left(1 + 2^{-g(\epsilon)}\right) > 0. \end{aligned}$$

Note that for $\lambda \in (0, \frac{1}{2}] \cap \mathbb{R}_c$ and f as described in Fig. 4.3 and Table 4.1, the FSC $\{W_\lambda, f, s_0\}$ is strongly connected. For $s_0 = 0$, the FSC achieves the state $s_1 = 1$ if the input output tuple (x_1, y_1) of the channel is either $(0, 1)$ or $(1, 0)$. Since $W_\lambda(1|0, 0) = W_\lambda(0|1, 0) = \lambda > 0$, the channel can reach the state $s_1 = 1$ in a one time-step for any input distribution $p(x_1|0) > 0$. Similarly, for $s_0 = 1$, the FSC achieves the state $s_1 = 0$ if the input output tuple (x_1, y_1) of the channel is $(1, 0)$. Since $W_\lambda(0|1, 1) = \lambda > 0$, the channel can reach the state $s_1 = 0$ in a one time-step for any input distribution with $p(0|0) > 0$. This implies that the FSC $\{W_\lambda, f, s_0\}$ with $s_0 \in \mathcal{S}$ is also indecomposable.

Hence, for every $\lambda \in (0, \frac{1}{2}]$, we have

$$C_{FB}(\{W_\lambda, f, 0\}) = C_{FB}(\{W_\lambda, f, 1\}). \tag{4.8}$$

Next we generate the indirect proof. First, we build a Turing machine that encodes the recursively enumerable non-recursive set $\mathcal{A}$ in a sequence of unifilar FSCs. For the

construction of the Turing machine, we rely on the construction introduced in [62][Case I, page 336]. This plays an important role in emphasizing the properties of the FSC feedback capacity. Similar constructions have been developed in [63] and [64].

Let $n \in \mathbb{N}$ be arbitrary. Now, for every $n \in \mathbb{N}$ and $m \in \mathbb{N}$, let

$$\lambda_{n,m} = \begin{cases} \frac{1}{2^l} & \mathfrak{T}_{\mathcal{A}} \text{ stops for input } n \text{ after } l \leq m \text{ steps} \\ \frac{1}{2^m} & \mathfrak{T}_{\mathcal{A}} \text{ does not stop for input } n \text{ after } m \text{ steps.} \end{cases}$$

Then the sequence $\{\lambda_{n,m}\}_{n,m\in\mathbb{N}}$ is a computable double sequence of rationals. For arbitrary $n \in \mathbb{N}$ and for all $m \geq M$, $m, M \in \mathbb{N}$, we have

$$|\lambda_{n,m} - \lambda_{n,M}| < \frac{1}{2^M}. \tag{4.9}$$

To prove (4.9) we will consider both cases: $\mathfrak{T}_{\mathcal{A}}$ stops for input n and $\mathfrak{T}_{\mathcal{A}}$ does not stop for input n.

- *$\mathfrak{T}_{\mathcal{A}}$ stops for input n after $l \leq M$ iterations*: In this case $\lambda_{n,M} = \lambda_{n,m}$, so $|\lambda_{n,m} - \lambda_{n,M}| = 0$.

- *$\mathfrak{T}_{\mathcal{A}}$ has not stopped for input n in M iterations*: For every $m \geq M$ it holds that $\lambda_{n,M} \geq \lambda_{n,m}$, meaning that $|\lambda_{n,m} - \lambda_{n,M}| = \lambda_{n,M} - \lambda_{n,m} = \frac{1}{2^M} - \lambda_{n,m} < \frac{1}{2^M}$.

The sequence $\{\lambda_{n,m}\}_{n,m\in\mathbb{N}}$ is a computable double sequence of rationals that converges effectively in m. This implies that for every $n \in \mathbb{N}$ the sequence $\{\lambda_{n,m}\}_{m\in\mathbb{N}}$ converges effectively to its limit λ_n^* and the limit is a computable real number $\lambda_n^* \in \mathbb{R}_c$. Since $\{\lambda_{n,m}\}_{n,m\in\mathbb{N}}$ is a computable double sequence of rationals such that as $m \to \infty$, $\lambda_{n,m} \to \lambda_n^*$, $\{\lambda_n^*\}_{n\in\mathbb{N}}$ is a sequence of computable real numbers. It further holds $\lambda_n^* \geq 0$ with equality if and only if the Turing machine $\mathfrak{T}_{\mathcal{A}}$ does not stop for input n.

We consider the computable double sequence of rational unifilar FSCs $\{P_{\lambda_{n,m}}\}_{n,m\in\mathbb{N}^2}$ defined by the computable double sequence of rational channels $\{W_{\lambda_{n,m}}\}_{n,m\in\mathbb{N}}$ and the function f defined in Table 4.1.

For arbitrary $n \in \mathbb{N}$ and for all $m \geq M$, $m, M \in \mathbb{N}$, we have

$$\begin{aligned} D(P_{\lambda_{n,m}}, P_{\lambda_{n,M}}) &= \max_{s'\in\mathcal{S}} \max_{x\in\mathcal{X}} \sum_{y\in\mathcal{Y}} \sum_{s\in\mathcal{S}} |P_{\lambda_{n,m}}(y, s|x, s') - P_{\lambda_{n,M}}(y, s|x, s')| \\ &= \max_{s'\in\mathcal{S}} \max_{x\in\mathcal{X}} \sum_{y\in\mathcal{Y}} \sum_{s\in\mathcal{S}} |W_{\lambda_{n,m}}(y|x, s') - W_{\lambda_{n,M}}(y|x, s')| \mathbb{1}(s = f(s', x, y)) \end{aligned} \tag{4.10}$$

$$\begin{aligned} &= \max_{s'\in\mathcal{S}} \max_{x\in\mathcal{X}} \sum_{y\in\mathcal{Y}} |W_{\lambda_{n,m}}(y|x, s') - W_{\lambda_{n,M}}(y|x, s')| \\ &= 2|\lambda_{n,m} - \lambda_{n,M}| < \frac{1}{2^{M-1}}, \end{aligned} \tag{4.11}$$

where (4.10) holds, since $\{P_{\lambda_{n,m}}\}_{n,m\in\mathbb{N}}$ are unifilar. (4.11) results from (4.7) in the following way:

$$\begin{aligned}
&\sum_{y\in\mathcal{Y}} |W_{\lambda_{n,m}}(y|0,0) - W_{\lambda_{n,M}}(y|0,0)| \\
&\qquad = \sum_{y\in\mathcal{Y}} |W_{\lambda_{n,m}}(y|1,0) - W_{\lambda_{n,M}}(y|1,0)| \\
&\qquad = \sum_{y\in\mathcal{Y}} |W_{\lambda_{n,m}}(y|1,1) - W_{\lambda_{n,M}}(y|1,1)| \\
&\qquad = |(1-\lambda_{n,m}) - (1-\lambda_{n,M})| + |\lambda_{n,m} - \lambda_{n,M}| \\
&\qquad = 2|\lambda_{n,m} - \lambda_{n,M}|
\end{aligned}$$

and

$$\sum_{y\in\mathcal{Y}} |W_{\lambda_{n,m}}(y|0,1) - W_{\lambda_{n,M}}(y|0,1)| = 0.$$

As a result of (4.9) and (4.11), we have that $W_{\lambda_{n,m}} \to W_{\lambda_n^*}$ as $m \to \infty$, for every $n \in \mathbb{N}$. Hence $\{W_{\lambda_n^*}\}_{n\in\mathbb{N}}$ is a sequence of computable channels.

We prove our result by contradiction. So we assume that C_{FB} is Borel-Turing computable, and we construct an example that proves the contrary. Since $C_{FB}(\{W, f, 0\})$ and $C_{FB}(\{W, f, 1\})$ are assumed to be Borel-Turing computable functions, the difference $\phi(\{W, f\}) = C_{FB}(\{W, f, 1\}) - C_{FB}(\{W, f, 0\})$ is a Borel-Turing computable function as well.

Further we use a sequence of computable unifilar FSCs $\{P_{\lambda_n^*}\}_{n\in\mathbb{N}} = \{W_{\lambda_n^*}, f\}_{n\in\mathbb{N}}$, where the state transition function is fixed and ϕ defines the sequence $\{\mu_n\}_{n\in\mathbb{N}}$ as follows:

$$\mu_n = \phi(\{W_{\lambda_n^*}, f\}), \quad n \in \mathbb{N}.$$

μ_n is a computable sequence of computable real numbers. From Definition 13, we have that for every computable sequence of computable real numbers there is a computable double sequence $\{\nu_{n,m}\}_{n,m\in\mathbb{N}}$ of rational numbers converging effectively to the computable sequence of computable real numbers, i.e.,

$$\left|\mu_n - \nu_{n,m}\right| < \frac{1}{2^m}.$$

For every n, we can consider the following Turing machine $\mathfrak{T}_*$: For input n, we set $m = 1$ and check if

$$\nu_{n,1} > \frac{1}{2}$$

is satisfied. If this is true, the Turing machine stops. Otherwise, we set $m = 2$ and check if

$$\nu_{n,2} > \frac{1}{4}$$

is satisfied. If this is true, the Turing machine stops. Otherwise, it continues as described. Next, we show that this Turing machine $\mathfrak{T}_*$ stops for input n if and only if $\mu_n > 0$.

"$\Leftarrow$" If $\mu_n > 0$, then there exists an $\tilde{M}$ with

$$\frac{1}{2^{\tilde{M}}} < \frac{\mu_n}{2}$$

so that

$$\begin{aligned}\mu_n &= \mu_n - \nu_{n,\tilde{M}} + \nu_{n,\tilde{M}} \leq \left|\mu_n - \nu_{n,\tilde{M}}\right| + \nu_{n,\tilde{M}} \\ &< \frac{1}{2^{\tilde{M}}} + \nu_{n,\tilde{M}} < \frac{\mu_n}{2} + \nu_{n,\tilde{M}},\end{aligned}$$

i.e., the Turing machine $\mathfrak{T}_*$ stops for input n within $\tilde{M}$ steps.

"$\Rightarrow$" It holds $\nu_{n,\hat{M}} > \frac{1}{2^{\hat{M}}}$ for a certain $\hat{M}$. Then,

$$\begin{aligned}\frac{1}{2^{\hat{M}}} &< \nu_{n,\hat{M}} = \nu_{n,\hat{M}} - \mu_n + \mu_n \\ &\leq \left|\nu_{n,\hat{M}} - \mu_n\right| + \mu_n < \frac{1}{2^{\hat{M}}} + \mu_n\end{aligned}$$

so that $\mu_n > 0$ is true.

This means that there exists a Turing machine $\mathfrak{T}_{\mathcal{S}}$ with

$$\mathfrak{T}_{\mathcal{S}}(n) = \begin{cases} n \in \mathcal{A} & \text{if } \mathfrak{T}_{\mathcal{A}} \text{ stops for input } n \\ n \in \mathcal{A}^c & \text{if } \mathfrak{T}_* \text{ stops for input } n. \end{cases} \tag{4.12}$$

This implies that $\mathcal{A}$ is a recursive set, which is a contradiction. This contradiction shows that $C_{FB}(\{W, f, 0\})$ and $C_{FB}(\{W, f, 1\})$ cannot be Banach-Mazur computable. This immediately implies that they cannot be Borel-Turing computable as well.

To extend the proof to $|\mathcal{X}| \geq 2$, $|\mathcal{Y}| \geq 2$, and $\mathcal{S} \geq 2$, we will divide the extension in two steps:

- The state set remains binary and the input and output alphabets may grow, i.e., $|\mathcal{S}| = 2$ and $|\mathcal{X}| \geq 2$, $|\mathcal{Y}| \geq 2$.
- We allow the state set to grow, i.e., $|\mathcal{S}| \geq 2$.

Step I: For $|\mathcal{X}| \geq 2$, $|\mathcal{Y}| \geq 2$, and $|\mathcal{S}| = 2$, we take the sequence of parameters $\{W_\lambda, f\}$ as above and extend them as follows: We set $W_\lambda(y_n|x_n, s_{n-1}) = 0$ for $y_n \in \mathcal{Y} \setminus \{0, 1\}$, $x_n \in \mathcal{X}$ and $s_{n-1} \in \mathcal{S}$ and also for $y_n \in \mathcal{Y}$, $x_n \in \mathcal{X} \setminus \{0, 1\}$ and $s_{n-1} \in \mathcal{S}$. For every pair $(x_n, y_n) \in (\mathcal{X} \setminus \{0, 1\} \times \mathcal{Y}) \cup (\mathcal{X} \times \mathcal{Y} \setminus \{0, 1\})$, we define the state transition function to be $f(x_n, y_n, s_{n-1}) = s_{n-1}$.

Step II: Let $|\mathcal{S}| \geq 2$. For every $s \in \mathcal{S} \setminus \{0, 1\}$ set

$$\begin{aligned}W_\lambda(0, |0, s) &= 1 - \left(\epsilon + \left(\frac{1}{2} - \epsilon\right)^{s-1}\right), \\ W_\lambda(1, |0, s) &= \epsilon + \left(\frac{1}{2} - \epsilon\right)^{s-1}, \\ W_\lambda(0, |1, s) &= 0, \quad W_\lambda(1, |1, s) = 1.\end{aligned}$$

Note that for $(x, y) \in \{0,1\}^2$, for every s we have the Z-channel with the probability of transmitting bit 0 incorrectly of $\delta_s = \epsilon + \left(\frac{1}{2} - \epsilon\right)^{s-1}$. It also holds that for every $s \in \mathcal{S} \setminus \{0,1\}$, $\epsilon < \delta_s \leq \frac{1}{2}$, and hence the channel at state $s = 1$ is less noisy than the channels at states $s \geq 2$.

We set $W_\lambda(y_n|x_n, s) = 0$ for $y_n \in \mathcal{Y} \setminus \{0,1\}$, $x_n \in \mathcal{X}$ and $s_{n-1} \in \mathcal{S}$ and also for $y_n \in \mathcal{Y}$, $x_n \in \mathcal{X} \setminus \{0,1\}$ and $s_{n-1} \in \mathcal{S}$.

Next we modify the state transition function f as follows: For $|\mathcal{S}| = 3$ we have a new state $s = 2$. For $(x_n, y_n, s_{n-1}) = (0,1,0)$, we modify the function f by setting the next state s_n to be $s_n = f(0,1,0) = 2$. For the state $s_{n-1} = 2$ we complete the state transition function as follows:

$$f(x_n, y_n, 2) = \begin{cases} 0 & \text{for every } (x_n, y_n) \text{ s.t. } W(x_n, y_n, 2) > 0, \\ 2 & \text{for every } (x_n, y_n) \text{ s.t. } W(x_n, y_n, 2) = 0. \end{cases}$$

The diagram of the state transition function f for $|\mathcal{X}| = |\mathcal{Y}| = 2$ and $|\mathcal{S}| = 3$ is illustrated in Fig. 4.4.

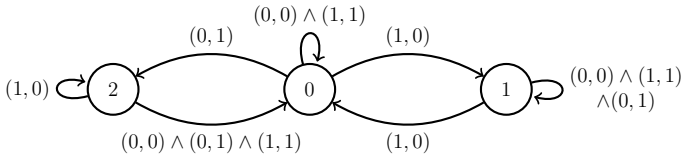

Figure 4.4: Diagram of the state transition function f for $|\mathcal{X}| = |\mathcal{Y}| = 2$ and $|\mathcal{S}| = 3$.

If $|\mathcal{S}| \geq 4$, we extend the state transition function iteratively as described above:

- Let $2 \leq s < |\mathcal{S}| - 1$. For $s_{n-1} = s$ we set $s_n = f(x_n, y_n, s_{n-1})$ to be

$$f(x_n, y_n, s) = \begin{cases} 0 & (x_n, y_n) \in (\mathcal{X} \times \mathcal{Y}) \setminus \{(0,1)\} \text{ s.t. } W(x_n, y_n, s) > 0, \\ s & (x_n, y_n) \in \mathcal{X} \times \mathcal{Y} \text{ s.t. } W(x_n, y_n, s) = 0, \\ s+1 & (x_n, y_n) = (0,1) \text{ s.t. } W(x_n, y_n, s) = 0. \end{cases}$$

- For $s = |\mathcal{S}| - 1$ and $s \geq 2$ we have

$$f(x_n, y_n, s) = \begin{cases} 0 & (x_n, y_n) \in \mathcal{X} \times \mathcal{Y} \text{ s.t. } W(x_n, y_n, s) > 0, \\ s & (x_n, y_n) \in \mathcal{X} \times \mathcal{Y} \text{ s.t. } W(x_n, y_n, s) = 0. \end{cases}$$

This way the FSCs with $|\mathcal{X}| \geq 2$, $|\mathcal{Y}| \geq 2$, and $|\mathcal{S}| \geq 2$ preserve the properties of the FSCs constructed above, i.e., they are unifilar and strongly connected. □

The fact that the FSC feedback capacity is not Banach-Mazur computable automatically implies that the feedback capacity of FSCs is not Borel-Turing computable. This leads us to the following corollary:

Corollary 1. *For all $|\mathcal{X}| \geq 2$, $|\mathcal{Y}| \geq 2$, and $|\mathcal{S}| \geq 2$, the feedback capacity function $C_{FB}(\{W, f, s_0\})\colon \mathcal{W}_c \times \mathcal{S}^{\mathcal{S}\times\mathcal{X}\times\mathcal{Y}} \times \mathcal{S} \to \mathbb{R}$ of unifilar FSCs feedback with parameters $\{W, f, s_0\}$ is not Borel-Turing computable.*

Corollary 1 states that the FSC feedback capacity is not Borel-Turing computable. This implies that there is no Turing machine, which for a fixed alphabet $|\mathcal{X}| \geq 2$, $|\mathcal{Y}| \geq 2$, and $|\mathcal{S}| \geq 2$, takes $\{W, f, s_0\}$ as input and computes the capacity $C_{FB}(\{W, f, s_0\})$. This gives us a negative answer to Question 1.

We further obtain the statement for the impossibility of obtaining the integrity condition for Turing machines, aimed to calculate the FSC feedback capacity C_{FB}.

Corollary 2. *The integrity requirement cannot be fulfilled for the computation of C_{FB} on the basis of Turing machines.*

Proof. If we find a Turing Machine TM_* that computes C_{FB} and satisfies the integrity requirement, then for every representation of $\{W, f, s_0\}$, TM_* must necessarily compute a representation of a certain number $z_* \in \mathbb{R}_c$. But since C_{FB} is not Borel-Turing computable, there must exist a triplet $\{W^*, f, s_0\}$ with a computable channel W^* such that TM_* generates two different numbers z_1^* and z_2^* for two different representations of $\{W^*, f, s_0\}$. So, we cannot have $C_{FB}(\{W^*, f, s_0\}) = z_1^*$ and $C_{FB}(\{W^*, f, s_0\}) = z_2^*$. Thus TM_* does not fulfill the integration requirement when computing C_{FB}.

□

Remark 19. *We have shown that the FSC feedback capacity is not Banach-Mazur computable for a special class of FSCs, the unifilar FSCs. This result holds for more general classes of FSCs as well.*

4.4 Non Approximability of the Capacity and Consequences for Achievability and Converse

In the previous section, we showed that the FSC feedback capacity is not a computable function. Here we are interested in finding a computable function that approximates the FSC feedback capacity. Moreover, we are interested in finding computable tight upper and lower bounds on the feedback capacity function and therewith a computable representation of tight achievability and converse.

Let us now revisit the concept of compound capacity for computable compound channels that was previously mentioned in Remark 18. Specifically, we will explore the information-theoretic implications arising from the non-computability of the number $C(\{W_n\}_{n\in\mathbb{N}})$. Let $M \in \mathbb{N}$ be arbitrary but fixed for a compound set $\{W_n\}_{n=1,\dots,M}$; we have $C(\{W_n\}_{n=1,\dots,M}) = \max_{p\in\mathcal{P}(\mathcal{X})} \min_{1\leq n\leq M} I(p; W_n)$. Hence, $C(\{W_n\}_{n=1,\dots,M})$ is a computable number. With $z_M = C(\{W_n\}_{n=1,\dots,M})$ we have the following relation: $z_M \geq z_{M+1}$, and it follows $\lim_{M\to\infty} z_M = C(\{W_n\}_{n\in\mathbb{N}})$. With this, we can find a computable sequence of ever-improving upper bounds for the compound capacity $C(\{W_n\}_{n\in\mathbb{N}})$ that converges to the compound capacity. In other words, we can always

find better converses for the coding theorem of compound channels $\{W_n\}_{n\in\mathbb{N}}$. However, there are no computable sequences $\{u_m\}_{m\in\mathbb{N}}$ of computable numbers, such that it holds that $u_m \leq u_{m+1}$ and $\lim_{m\to\infty} u_m = C(\{W_n\})$, since otherwise $C(\{W_n\}_{n\in\mathbb{N}})$ would be a computable number. In the sense of coding theory, therefore, the achievability cannot determine a sequence of ever-improving lower bounds for compound capacity. Specifically, there cannot exist a computable sequence $\{R_n\}_{n\in\mathbb{N}}$ of achievable computable rates R_n, with $n \in \mathbb{N}$, such that $\{R_n\}_{n\in\mathbb{N}}$ converges to the number $C(\{W_n\}_{n\in\mathbb{N}})$, i.e., there is no computable capacity achieving sequence $\{R_n\}_{n\in\mathbb{N}}$ of achievable rates for the computable compound channel $\{W_n\}_{n\in\mathbb{N}}$. In other words, it is impossible to find a computable sequence of achievable rates that approaches $C(\{W_n\}_{n\in\mathbb{N}})$ effectively.

In [65], a new technique for deriving an upper bound for the feedback capacity for unifilar FSC was developed. This bound was shown to be tight for some channels. In this section, we study the question of whether there is a computable function that takes any FSC and approximates its feedback capacity within a certain margin of error. We show that it is impossible to find a computable function that approximates the FSC feedback capacity within a small margin of error, providing us with a negative answer to Question 2.

Theorem 10. *For $|\mathcal{X}| \geq 2$, $|\mathcal{Y}| \geq 2$, and $|\mathcal{S}| \geq 2$ arbitrary but fixed. Let $C_{FB}(\{W, f, s_0\})\colon \mathcal{W}_c \times \mathcal{S}^{\mathcal{S}\times\mathcal{X}\times\mathcal{Y}} \times \mathcal{S} \to \mathbb{R}$ be the FSC feedback capacity. Let $G\colon \mathcal{W}_c \times \mathcal{S}^{\mathcal{S}\times\mathcal{X}\times\mathcal{Y}} \times \mathcal{S} \to \mathbb{R}$ be a function such that for a fixed $f \in \mathcal{S}^{\mathcal{S}\times\mathcal{X}\times\mathcal{Y}}$ and for every $s_0 \in \mathcal{S}$, we have*

$$\sup_{W\in\mathcal{W}_c} |G(\{W, f, s_0\}) - C_{FB}(\{W, f, s_0\})| \leq \delta,$$

with $\delta = 1 - \log_2\left(1 + 2^{g(\epsilon)}\right)$ with $\epsilon \in (0, \frac{1}{2})$ and $g(\epsilon) = \frac{H_2(\epsilon)}{1-\epsilon}$. Then G cannot be a computable function.

Proof. We prove the result by contradiction. We consider the FSC $\{W_\lambda, f, s_0\}_{n\in\mathbb{N}}$ for $\lambda \in [0, \frac{1}{2}] \cap \mathbb{R}_c$ as in (4.7). For $\lambda = 0$, we have

$$C_{FB}(\{W_0, f, 0\}) - C_{FB}(\{W_0, f, 1\}) = 1 - \log_2\left(1 + 2^{-g(\epsilon)}\right) > 0, \tag{4.13}$$

with $\epsilon \in (0, \frac{1}{2})$ and $g(\epsilon) = \frac{H_2(\epsilon)}{1-\epsilon}$. Let $\delta = 1 - \log_2\left(1 + 2^{g(\epsilon)}\right)$. Assume that for C_{FB} there is a computable function $G\colon \mathcal{W}_c \times \mathcal{S}^{\mathcal{S}\times\mathcal{X}\times\mathcal{Y}} \times \mathcal{S} \to \mathbb{R}$ such that for every $f\colon \mathcal{S}\times\mathcal{X}\times\mathcal{Y} \to \mathcal{S}$ and for every $s_0 \in \mathcal{S}$ we have that

$$\sup_{W\in\mathcal{W}_c} |G(\{W, f, s_0\}) - C_{FB}(\{W, f, s_0\})| \leq \delta. \tag{4.14}$$

We define the sequence of unifilar FSCs $\{W_{\lambda_n *}, f, s_0\}_{n\in\mathbb{N}}$ as in the proof of Theorem 9. For every $n \in \mathbb{N}$ and for $s_0, \tilde{s}_0 \in \mathcal{S}$ with $C_{FB}(\{W, f, s_0\}) = \bar{C}_{FB}(\{W_{\lambda_n *}, f\})$ and $C_{FB}(\{W, f, \tilde{s}_0\}) = \underline{C}_{FB}(\{W_{\lambda_n *}, f\})$, we have that

$$|G(\{W_{\lambda_n *}, f, s_0\}) - G(\{W_{\lambda_n *}, f, \tilde{s}_0\})| \geq 0.$$

For every $n \in \mathcal{A}$ the FSC $\{W_{\lambda_n *}, f, s_0\}$ is indecomposable. Hence, we have

$$\begin{aligned}
|G(\{W_{\lambda_n *}, f, s_0\}) - G(\{W_{\lambda_n *}, f, \tilde{s}_0\})| &= |G(\{W_{\lambda_n *}, f, s_0\}) - \bar{C}_{FB}(\{W_{\lambda_n *}, f\}) \\
&\quad + \bar{C}_{FB}(\{W_{\lambda_n *}, f\}) - \underline{C}_{FB}(\{W_{\lambda_n *}, f\}) + \underline{C}_{FB}(\{W_{\lambda_n *}, f\}) - G(\{W_{\lambda_n *}, f, \tilde{s}_0\})| \\
&\leq |G(\{W_{\lambda_n *}, f, s_0\}) - C_{FB}(\{W_{\lambda_n *}, f, s_0\})| \\
&\quad + |C_{FB}(\{W_{\lambda_n *}, f, \tilde{s}_0\}) - G(\{W_{\lambda_n *}, f, \tilde{s}_0\})| \\
&\leq 2\delta.
\end{aligned}$$

On the other hand, for every $n \in \mathbb{N}$ and $n \notin \mathcal{A}$ we have

$$\begin{aligned}
|G(\{W_{\lambda_n *}, f, s_0\}) - G(\{W_{\lambda_n *}, f, \tilde{s}_0\})| &= |G(\{W_{\lambda_n *}, f, s_0\}) - \bar{C}_{FB}(\{W_{\lambda_n *}, f\}) \\
&\quad + \bar{C}_{FB}(\{W_{\lambda_n *}, f\}) - \underline{C}_{FB}(\{W_{\lambda_n *}, f\}) \\
&\quad + \underline{C}_{FB}(\{W_{\lambda_n *}, f\}) - G(\{W_{\lambda_n *}, f, \tilde{s}_0\})| \\
&\leq |G(\{W_{\lambda_n *}, f, s_0\}) - \bar{C}_{FB}(\{W_{\lambda_n *}, f\})| \\
&\quad + |\bar{C}_{FB}(\{W_{\lambda_n *}, f\}) - \underline{C}_{FB}(\{W_{\lambda_n *}, f\}) \\
&\quad + |\underline{C}_{FB}(\{W_{\lambda_n *}, f\}) - G(\{W_{\lambda_n *}, f, \tilde{s}_0\})| \\
&\leq 2\delta + |\bar{C}_{FB}(\{W_{\lambda_n *}, f\}) - \underline{C}_{FB}(\{W_{\lambda_n *}, f\})| \\
&= 2\delta + \phi(\{W_{\lambda_n *}, f\}) \\
&= 2\delta + \mu_n.
\end{aligned}$$

We can use the same Turing machine $\mathfrak{T}_*$ as for the proof of Theorem 9 to compute μ_n. Hence, we can construct a Turing machine as in (4.12) that decides for every $n \in \mathbb{N}$ if $n \in \mathcal{A}$ or $n \notin \mathcal{A}$. This is a contradiction, since $\mathcal{A}$ is a recursively enumerable but non-recursive set. So the assumption that G is computable is wrong. □

Theorem 10 states that there is no computable function that approximates the FSC feedback capacity within a tolerance of at most $\delta = 1 - \log_2\left(1 + 2^{-g(\epsilon)}\right)$, $\epsilon \in (0, \frac{1}{2})$ and $g(\epsilon) = \frac{H_2(\epsilon)}{1-\epsilon}$. From this result, we can conclude that there are no computable sequences of computable functions that converge pointwise to the FSC feedback capacity.

There are two approaches to prove coding theorems. Lower bounds on the capacity are derived via achievability results. We are interested in studying whether such lower bounds can be computed on digital computers. On the other hand, upper bounds are derived via converses. We study whether it is possible to find algorithms that compute such upper bounds on digital computers.

Upper and lower bounds enclose the FSC feedback capacity. If there are, in fact, computable upper and lower bounds, it would be desirable for them to be very close to the FSC feedback capacity. We are interested in determining the smallest distance between the computable upper and lower bounds.

The following theorem states that there exist no computable upper and lower bounds that are both tight to the capacity of FSCs. Moreover, Theorem 11 derives the minimum distance between the computable upper and lower bounds and the FSC feedback capacity that can be achieved. This provides us with a negative answer to Question 3.

Theorem 11. *Let $|\mathcal{X}| \geq 2$, $|\mathcal{Y}| \geq 2$, and $|\mathcal{S}| \geq 2$ be arbitrary but fixed. For all monotonically increasing computable sequences $\{F_N\}_{N\in\mathbb{N}}$ of computable continuous functions satisfying*

$$F_N(\{W, f, s_0\}) \leq F_{N+1}(\{W, f, s_0\})$$

and

$$F_N(\{W, f, s_0\}) \leq C_{FB}(\{W, f, s_0\})$$

for every $N \in \mathbb{N}$, every $W \in \mathcal{W}_c, f \in \mathcal{S}^{\mathcal{S}\times\mathcal{X}\times\mathcal{Y}}$ and $s_0 \in \mathcal{S}$, and for all monotonically decreasing computable sequences $\{G_N\}_{N\in\mathbb{N}}$ of computable continuous functions satisfying

$$G_{N+1}(\{W, f, s_0\}) \leq G_N(\{W, f, s_0\})$$

and

$$C_{FB}(\{W, f, s_0\}) \leq G_N(\{W, f, s_0\})$$

for every $N \in \mathbb{N}$, every $W \in \mathcal{W}_c, f \in \mathcal{S}^{\mathcal{S}\times\mathcal{X}\times\mathcal{Y}}$ and $s_0 \in \mathcal{S}$, we have that the following holds: For a fixed $f \in \mathcal{S}^{\mathcal{S}\times\mathcal{X}\times\mathcal{Y}}$ there is an $\hat{s_0} \in \mathcal{S}$ and a $\hat{W} \in \mathcal{W}_c$ such that

$$\begin{aligned}\liminf_{N\to\infty} \max\{&C_{FB}(\{\hat{W}, f, \hat{s_0}\}) - F_N(\{\hat{W}, f, \hat{s_0}\}),\\ &G_N(\{\hat{W}, f, \hat{s_0}\}) - C_{FB}(\{\hat{W}, f, \hat{s_0}\})\} > \delta\end{aligned}$$

with $\delta = 1 - \log_2\left(1 + 2^{-g(\epsilon)}\right)$, $\epsilon \in (0, \frac{1}{2})$ and $g(\epsilon) = \frac{H_2(\epsilon)}{1-\epsilon}$.

Proof. We prove this theorem by contradiction.

Assume that $\{F_N\}_{N\in\mathbb{N}}$ is a monotonically increasing computable sequence of computable functions such that

$$\lim_{N\to\infty} F_N(\{W, f, s_0\}) = C_{FB}(\{W, f, s_0\})$$

for every $W \in \mathcal{W}_c$, $s_0 \in \mathcal{S}$ and every $f \in \mathcal{S}^{\mathcal{S}\times\mathcal{X}\times\mathcal{Y}}$.

Assume $\{G_N\}_{N\in\mathbb{N}}$ is a monotonically decreasing computable sequence of computable functions such that

$$\lim_{N\to\infty} G_N(\{W, f, s_0\}) = C_{FB}(\{W, f, s_0\})$$

for every $W \in \mathcal{W}_c$, $s_0 \in \mathcal{S}$ and every $f \in \mathcal{S}^{\mathcal{S}\times\mathcal{X}\times\mathcal{Y}}$. Then for every $\{W, f, s_0\} \in \mathcal{W}_c \times \mathcal{S}^{\mathcal{S}\times\mathcal{X}\times\mathcal{Y}} \times \mathcal{S}$, we define the sequence of functions $\{\epsilon_N\}_{N\in\mathbb{N}}$ as follows:

$$\epsilon_N(\{W, f, s_0\}) = G_N(\{W, f, s_0\}) - F_N(\{W, f, s_0\}) \geq 0.$$

Since $\{G_N\}_{N\in\mathbb{N}}$ is a sequence of monotonically decreasing functions and $\{F_N\}_{N\in\mathbb{N}}$ is a sequence of monotonically increasing functions, we have that

$$\begin{aligned}\epsilon_N(\{W, f, s_0\}) &= G_N(\{W, f, s_0\}) - F_N(\{W, f, s_0\})\\ &\geq G_{N+1}(\{W, f, s_0\}) - F_N(\{W, f, s_0\})\\ &\geq G_{N+1}(\{W, f, s_0\}) - F_{N+1}(\{W, f, s_0\})\\ &= \epsilon_{N+1}(\{W, f, s_0\}),\end{aligned}$$

i.e., ϵ_N is a monotonically decreasing sequence of functions with

$$\lim_{N\to\infty} \epsilon_N(\{W, f, s_0\}) = 0$$

for every $\{W, f, s_0\} \in \mathcal{W}_c \times \mathcal{S}^{\mathcal{S}\times\mathcal{X}\times\mathcal{Y}} \times \mathcal{S}$. Since both $\{G_N\}_{N\in\mathbb{N}}$ and $\{F_N\}_{N\in\mathbb{N}}$ are computable sequences of computable functions this implies that $\{\epsilon_N\}_{N\in\mathbb{N}}$ is also a computable sequence of computable functions. From [8, Theorem 10], we have that $\{\epsilon_N\}_{N\in\mathbb{N}}$ converges effectively to 0.

On the other hand, for every $\{W, f, s_0\} \in \mathcal{W}_c \times \mathcal{S}^{\mathcal{S}\times\mathcal{X}\times\mathcal{Y}} \times \mathcal{S}$ and for every $N \in \mathbb{N}$ we have

$$\begin{aligned} 0 &\leq G_N(\{W, f, s_0\}) - C_{FB}(\{W, f, s_0\}) \\ &\quad + C_{FB}(\{W, f, s_0\}) - F_N(\{W, f, s_0\}) \\ &= G_N(\{W, f, s_0\}) - F_N(\{W, f, s_0\}) \\ &= \epsilon_N(\{W, f, s_0\}). \end{aligned}$$

Thus, we have

$$\sup_{W\in\mathcal{W}_c} |G_N(\{W, f, s_0\}) - C_{FB}(\{W, f, s_0\})| \leq \epsilon_N(\{W, f, s_0\})$$

and

$$\sup_{W\in\mathcal{W}_c} |C_{FB}(\{W, f, s_0\}) - F_N(\{W, f, s_0\})| \leq \epsilon_N(\{W, f, s_0\}).$$

If there exist computable sequences of functions $\{F_N\}_{N\in\mathbb{N}}$ and $\{G_N\}_{N\in\mathbb{N}}$ that converge effectively to C_{FB} for every $\{W, f, s_0\} \in \mathcal{W}_c \times \mathcal{S}^{\mathcal{S}\times\mathcal{X}\times\mathcal{Y}} \times \mathcal{S}$, then for every $\delta > 0$ there must be a computable function $G_\delta := \mathcal{W}_c \times \mathcal{S}^{\mathcal{S}\times\mathcal{X}\times\mathcal{Y}}\mathcal{S} \to \mathbb{R}$ such that for every $f \in \mathcal{S}^{\mathcal{S}\times\mathcal{X}\times\mathcal{Y}}$ and every $s_0 \in \mathcal{S}$ we have

$$\sup_{W\in\mathcal{W}_c} |G_\delta(\{W, f, s_0\}) - C_{FB}(\{W, f, s_0\})| \leq \delta.$$

However, this is a contradiction, since from Theorem 10 we have that for $\delta^* = 1 - \log_2\left(1 + 2^{g(\epsilon)}\right)$ with $\epsilon \in (0, \frac{1}{2})$ and $g(\epsilon) = \frac{H_2(\epsilon)}{1-\epsilon}$ there is no computable function G_{δ^*} such that for every $f \in \mathcal{S}^{\mathcal{S}\times\mathcal{X}\times\mathcal{Y}}$ and every $s_0 \in \mathcal{S}$

$$\sup_{W\in\mathcal{W}_c} |G_{\delta^*}(\{W, f, s_0\}) - C_{FB}(\{W, f, s_0\})| \leq \delta^*$$

holds. Hence there must be a $\hat{W}$ such that either

$$\liminf_{N\to\infty} |G_N(\{\hat{W}, f, s_0\}) - C_{FB}(\{\hat{W}, f, s_0\})| \geq \delta^*$$

or

$$\liminf_{N\to\infty} |C_{FB}(\{\hat{W}, f, s_0\}) - F_N(\{\hat{W}, f, s_0\})| \geq \delta^*.$$

□

In Theorem 11, we consider two sequences of computable functions. The first one is a monotonically increasing sequence of computable functions and the second one is a monotonically decreasing sequence of computable functions. If the sequences converged uniformly, then it would imply that the sequences of upper and lower bounds also converge pointwise to the FSC feedback capacity. However, Theorem 11 defines the channel $\hat{W}$ at which the convergence fails. The difference from Theorem 10 is that with the latter, we do not require finding sequences of bounds that converge monotonically. However, we do demand the existence of a single function that can approximate the FSC feedback capacity. In Theorem 10, although both the initial state s_0 and the state transition function f are fixed, it is shown that the maximum distance between the FSC feedback capacity and a computable approximation function over the channels cannot be smaller than $\delta^* = 1 - \log_2\left(1 + 2^{g(\epsilon)}\right)$ with $\epsilon \in (0, \frac{1}{2})$ and $g(\epsilon) = \frac{H_2(\epsilon)}{1-\epsilon}$.

The monotonically increasing sequence of functions $\{F_N\}_{N\in\mathbb{N}}$ is a sequence of lower bounds on the capacity and can be interpreted as achievability bounds. On the other hand, the monotonically decreasing sequence of functions $\{G_N\}_{N\in\mathbb{N}}$ is a sequence of upper bounds on the capacity and can be interpreted as converse bounds. As a consequence of Theorem 11, it is impossible to find tight computable achiveability and computable converse bounds simultaneously. Hence, at least one of them is not computable.

Thus, one cannot find techniques, such as the ones for DMCs, that can be implemented on a digital computer and give us, up to any desired precision, the range in which the optimal performance lies. Consequently, if one is interested in studying the behavior of a coding procedure for FSCs with feedback, it is impossible to numerically evaluate it by comparing it to tight bounds of its optimal performance.

4.5 Feedback Capacity as a Finite Multi-Letter Optimization Problem

In this section, we study whether or not it is possible to formulate the FSC feedback capacity as a finite multi-letter optimization problem. To this aim, we first have to study the continuity behavior of the capacity function. We show that the capacity function is discontinuous for certain $s_0 \in \mathcal{S}$, $f \in \mathcal{S}^{\mathcal{S}\times\mathcal{X}\times\mathcal{Y}}$ and computable $W \in \mathcal{W}_c$. The discontinuity result makes it impossible to describe the FSC feedback capacity as a finite multi-letter optimization problem providing us with a negative answer to Question 4.

Theorem 12. *For all $|\mathcal{X}| \geq 2$, $|\mathcal{Y}| \geq 2$, and $|\mathcal{S}| \geq 2$, the capacity function $C_{FB} : \mathcal{P}(\mathcal{Y}|\mathcal{X}\times\mathcal{S}) \times \mathcal{S}^{\mathcal{S}\times\mathcal{X}\times\mathcal{Y}} \to \mathbb{R}$ is discontinuous.*

Proof. We consider the channels $W(y_n|x_n, 0)$ and $W(y_n|x_n, 1)$ as in (4.5) and the state transition function f as described in Table 4.1.

Next, we consider $\{W_k, f, s_0\}$ for $k \geq 1$ with

$$W_k(y_n|x_n, 0) = \begin{pmatrix} 1-\frac{1}{k} & \frac{1}{k} \\ \frac{1}{k} & 1-\frac{1}{k} \end{pmatrix} \tag{4.15}$$

$$W_k(y_n|x_n, 1) = \begin{pmatrix} 1-\epsilon & \epsilon \\ \frac{1}{k} & 1-\frac{1}{k} \end{pmatrix}. \tag{4.16}$$

We observe that the FSC $\{W_k, f, s_0\}$, $s_0 \in \mathcal{S}$, $k \geq 1$, as defined above is unifilar and strongly connected, and therefore indecomposable. Note that for every $x_n \in \mathcal{X}, y_n \in \mathcal{Y}$ and $s_0 \in \mathcal{S}$ we have $W_k(y_n|x_n, s_0) \in \mathcal{W}_c$, which implies that the channels are computable.

For FSCs as defined in (4.5)-(4.15), we have for any $s_0 \in \mathcal{S}$, $D(\{W, f, s_0\}, \{W_k, f, s_0\}) = \frac{2}{k}$. Next, let us assume that $C_{FB}(\{W, f, s_0\})$, $s_0 \in \{0, 1\}$ is a continuous function on $\mathcal{P}(\mathcal{Y}|\mathcal{X} \times \mathcal{S}) \times \mathcal{S}^{\mathcal{S}\times\mathcal{X}\times\mathcal{Y}}$. Then we must have $\lim_{k\to\infty} C_{FB}(\{W_k, f, 0\}) = C_{FB}(\{W, f, 0\})$ and $\lim_{k\to\infty} C_{FB}(\{W_k, f, 1\}) = C_{FB}(\{W, f, 1\})$. Since for all $k \in \mathbb{N}$ the FSC $\{W_k, f, s_0\}$, $s_0 \in \mathcal{S}$ is indecomposable, we then have $C_{FB}(\{W_k, f, 0\}) = C_{FB}(\{W_k, f, 1\})$ and consequently obtain

$$\begin{aligned} 1 &= C_{FB}(\{W, f, 0\}) = \lim_{k\to\infty} C(\{W_k, f, 0\}) \\ &= \lim_{k\to\infty} C(\{W_k, f, 1\}) = C_{FB}(\{W, f, 1\}) \\ &= \log_2\left(1 + 2^{-g(\epsilon)}\right) < 1 \end{aligned}$$

with $g(\epsilon) = \frac{H_2(\epsilon)}{1-\epsilon}$ and $\epsilon \in (0, \frac{1}{2}) \cap \mathbb{Q}$. This is a contradiction. Hence, at least one of the functions $C(\{W, f, 0\})$ or $C(\{W, f, 1\})$ must be discontinuous, proving the desired result. □

Theorem 13. *Let $|\mathcal{X}| \geq 2$, $|\mathcal{Y}| \geq 2$, and $|\mathcal{S}| \geq 2$ be arbitrary. Then there is no natural number $n_0 \in \mathbb{N}$ such that the capacity $C_{FB}(\{W, f, s_0\})$ can be expressed as*

$$C_{FB}(\{W, f, s_0\}) = \max_{u\in\mathcal{U}} F(u, W, f, s_0), \tag{4.17}$$

with $\mathcal{U} \subset \mathbb{R}^{n_0}$ being a compact set and $F : \mathcal{U} \times \mathcal{P}(\mathcal{Y}|\mathcal{X} \times \mathcal{S}) \times \mathcal{S}^{\mathcal{S}\times\mathcal{X}\times\mathcal{Y}} \times \mathcal{S} \to \mathbb{R}$ a continuous function.

Proof. We use the same line of argument as for [9, Theorem 1]. The crucial observation is the following: To be able to express the capacity $C_{FB}(\{W, f, s_0\})$ as in (4.17), the capacity necessarily needs to be a continuous function. This cannot be the case as shown by Theorem 12. □

Theorem 13 implies that the feedback capacity cannot be expressed by a finite multi-letter formula. This further implies that there is no closed-form solution possible for the FSC feedback capacity in general.

4.6 Conclusions

In this chapter, we have studied the FSC feedback capacity from an algorithmic point of view. We have shown that the feedback capacity function is not Banach-Mazur computable, which is the weakest form of computability. Hence, the FSC feedback capacity is also not Borel-Turing computable. This means that there is no algorithm on digital computers that takes an arbitrary FSC $\{W, f, s_0\}$ as input and computes its feedback capacity effectively, i.e., that stops when a certain desired precision has been achieved. To prove this result, we used a restricted class of FSCs: unifilar FSCs. This automatically implies that the feedback capacity of more general channel classes is also not computable. We have further shown that if the feedback capacity of FSCs had been computable, then this could yield a solution for the halting problem, which has been proven to be an unsolvable problem. Unfortunatly, this implies that it is not possible to fulfill the integrity requirement of the upcoming generation of mobile communication using digital computers. Therefore, the trustworthiness requirement cannot be met for the computation of the feedback capacity.

There have been techniques developed for computing the capacity of some FSCs with feedback. Our results show that although these techniques are computable for the specific channels, they cannot be effective on the channel in general.

Since the capcity of FSCs with feedback is not computable, one could aim to design computable functions that approximate the capacity. We have shown that if such computable functions exist, they can approximate the capacity up to a certain margin of error. Yet, it is impossible to algorithmically approximate the capacity of FSCs within a smaller margin of error.

Unfortunately, the non-approximability of the FSC feedback capacity has a direct implication in computing upper and lower bounds on the FSC feedback capacity. We have shown that it is impossible to find tight upper and lower bounds that are simultaneously computable. This implies that if we aim for tight achievability and converse, either achievability or converse or both are non-computable. Tight upper and lower bounds are crucial for the study of code implementation for communication models. The lack of bounds, in this case for the FSC feedback capacity, makes it hard to evaluate how good a particular code is.

The FSC feedback capacity is given in terms of a multi-letter formula, which means that it is specified by the limit of a sequence of optimization problems. This makes it especially difficult to compute. Finding a finite letter formulation could facilitate computing the capacity. However, we have shown that the FSC feedback capacity cannot be expressed as a finite multi-letter optimization problem. Hence, none of the approaches studied in this paper to approximate the capacity allow us to compute the FSC feedback capacity.

5 Computability of the Additive Colored Gaussian Noise Channel Capacity

In Chapter 4, we have shown that the capacity of FSCs with feedback is not a computable function, due to its complicated description. This prompts the main question of this chapter: *What is the simplest communication channel for which such a numerical computation of the capacity is not possible?* We provide an answer to this query by showing that there are band-limited ACGN channels, which are standard communication channels with a very simple structure that do not have a computable capacity.

The band-limited ACGN channel is a very important model for wireless communication, as it can be used to model commonly encountered channels such as the frequency selective fading channel. The band-limited Gaussian channel, introduced in [58, 66], is a continuous-time channel. In [66], two noise models are introduced: white and colored Gaussian noise. Colored Gaussian noise is Gaussian distributed and has a psd that varies with frequency while the spectral density of white Gaussian noise is a constant over all frequencies. The capacity and error performance of codes for the band-limited ACGN channel were carefully studied in [67, 68, 69]. A detailed description of the band-limited ACGN channel and its results can be found in [70, 71, 72, 73, 74]. In [7], Gallager showed that the capacity-achieving psd of the linear ACGN channel can be determined using the water pouring technique. In [75], the authors provide an overview of techniques for constructing capacity-achieving codes for the ACGN channel.

Computing the capacity of the band-limited ACGN channel is a very important task for practical systems. The capacity serves as a benchmark for designing and optimizing systems to achieve optimal message transmission. This enables the design of codes that fulfill prescribed reliability and efficiency system requirements while also optimizing the use of communication resources.

This chapter addresses the question of whether the capacity of the band-limited ACGN channel can be computationally determined.

5.1 Continuous Gaussian Channels

In this section we consider a communication scenario where both the input and output of the channel are amplitude- and time-continuous. Amplitude-continuous means that the signal alphabets are uncountable infinite, and by time-continuous we allow the transmission to be continuous over time. The time continuous additive Gaussian channel is

represented by the following formula:

$$y(t) = x(t) + z(t),$$

where $x(t), y(t)$, and $z(t)$ are the channel input, channel output and noise at time instant $t \in \mathcal{T} \subset \mathbb{R}$ and they take values in $\mathbb{R}$. The noise $z(t)$ is zero-mean Gaussian distributed. The Fourier transforms of the input signals and noise are represented by

$$X(f) = \int_{-\infty}^{\infty} x(t) e^{-i2\pi ft}\, dt$$
$$Z(f) = \int_{-\infty}^{\infty} z(t) e^{-i2\pi ft}\, dt.$$

Let $x_T(t)$ be the fraction of the signal $x(t)$ that is equal to $x(t)$ in the time interval $[-\frac{T}{2}, \frac{T}{2}]$ and 0 outside. The total signal power P_{tot} is given by

$$P_{\text{tot}} = \lim_{T\to\infty} \frac{1}{T} \int_{-\frac{T}{2}}^{\frac{T}{2}} |x_T(t)|^2\, dt = \lim_{T\to\infty} \frac{1}{T} \int_{\infty}^{\infty} |X_T(f)|^2\, df = \int_{\infty}^{\infty} P_x(f)\, df$$

where

$$|X_T(f)|^2 = \int_{-\infty}^{\infty} \Big[\int_{-\infty}^{\infty} x_T(t-\tau) x_T(t)\, dt \Big] e^{-i2\pi f\tau}\, d\tau$$

and $P_x(f) = \lim_{T\to\infty} \frac{|X_T(f)|^2}{T}$ is the psd of the signal $x(t)$. Similarly, let $z_T(t)$ be equal to the noise $z(t)$ in the time interval $[-\frac{T}{2}, \frac{T}{2}]$ and 0 outside. The noise psd is given by

$$N(f) = \int_{-\infty}^{\infty} R_z(\tau) e^{-i2\pi f\tau}\, d\tau$$

with $R_z(\tau) = \lim_{T\to\infty} \frac{1}{T} \int_{-\infty}^{\infty} z_T(t-\tau) z_T(t)\, dt$.

We consider only band-limited signals. Letting the bandwidth be $B > 0$, the psd of band-limited signals with bandwidth B has the following structure:

$$P_x(f) = \begin{cases} P_x(f) & \text{for } |f| \in [0, B] \\ 0 & \text{else.} \end{cases}$$

$z(t)$ is a band-limited colored Gaussian noise with spectral density $N(f)$ with

$$N(f) = \begin{cases} \geq 0 & \text{for } |f| \in [0, B] \\ 0 & \text{else.} \end{cases}$$

We consider a communication scenario subject to a power constraint P. This means that the total signal power should not exceed P, and it is described by

$$\int_{-B}^{B} P_x(f)\, df \leq P.$$

We aim to find codes for the band-limited channel described above. The code should consist of band-limited signals. For this we consider the set $\mathcal{X}(B, T, P)$ which is the set

of approximately band-limited signals with bandwidth B, approximately time-limited to T seconds and with a total power not exceeding P, i.e., for every signal $x \in \mathcal{X}(B,T,P)$ it holds that $\int_{-B}^{B} P_x(f)\,df \leq P$. We define $\mathcal{Y}(B,T)$ to be the set of received signals, which are approximately band-limited with bandwidth B and approximately time-limited to T.

A code for the band-limited ACGN channel with power constraint consists of a pair of functions (f,ϕ), where f is an encoder function $f\colon \mathcal{M} \to \mathcal{C} \subset \mathcal{X}(B,T,P)$, where $\mathcal{C}$ is the codebook, and a decoder function $\phi\colon \mathcal{Y}(B,T) \to \mathcal{M}$. The transmission rate R is defined by

$$R = \frac{1}{T} \ln |\mathcal{M}|.$$

The average error probability P_e is given by

$$P_e = \frac{1}{|\mathcal{M}|} \sum_{i=1}^{|\mathcal{M}|} \Pr(\phi(f(i)) \neq i).$$

A rate R is called *achievable* for the band-limited ACGN channel, if one can find a code (f,ϕ) that operates at a transmission rate of R and for which the average error probability vanishes $P_e \to 0$ as $T \to \infty$. The channel *capacity* is defined as the supremum of all achievable rates.

Theorem 14 ([66])**.** *The capacity of the band-limited ACGN channel with bandwidth B, and continuous noise power spectrum N on the interval $[0,B]$ subject to a power constraint $P > 0$ is given by*

$$C(N,P) = \int_0^B \ln\left(1 + \frac{P_x^*(P,f)}{N(f)}\right) df.$$

The capacity-achieving power spectrum density is given by

$$P_x^*(P,f) = \begin{cases} \Big[\nu - N(f)\Big]_+ & \text{for } |f| \in [0,B] \\ 0 & \text{for } |f| \notin [0,B], \end{cases} \tag{5.1}$$

where ν is chosen such that $\int_{-B}^{B} P_x^*(P,f)\,df = P$ is satisfied.

There are a large number of different derivations for the formula, see [71, 76, 73, 77].

The capacity-achieving psd is given by the water pouring solution. Water pouring is well known for its simple derivation [78]. In general, the problem is approached by dividing the noise spectrum into n sub-channels of width Δf_n and assuming that each sub-channel is independent of the others. $N(f)$ is then approximated by $N(f_i)$ for $f \in [f_i - \frac{\Delta f_n}{2}, f_i + \frac{\Delta f_n}{2}]$ and $i \in \{1,\ldots,n\}$. The capacity of each sub channel f_i is given by

$$C_n(N,P,f_i) = \Delta f_n \ln\left(1 + \frac{P_x^*(P,f_i)}{N(f_i)}\right)$$

where $P_x^*(P, f_i) = \nu - N(f_i)$ and ν is derived by using the method of Lagrange multipliers. The total capacity and the total transmit power are given by

$$C_n(N, P) = \sum_{i=1}^{n} \Delta f_n \ln\left(1 + \frac{P_x^*(P, f_i)}{N(f_i)}\right) \tag{5.2}$$

$$P = \sum_{i=1}^{n} \Delta f_n P_x^*(P, f_i). \tag{5.3}$$

As $n \to \infty$ then $\Delta f_n \to 0$ and Eqs. 5.2 and 5.3 become integrals:

$$C(N, P) = \lim_{n\to\infty} C_n(N, P) = \int_0^B \ln\left(1 + \frac{P_x^*(P, f)}{N(f)}\right) df$$

$$P = \lim_{n\to\infty} \sum_{i=1}^{n} \Delta f_n P_x^*(P, f_i) = \int_0^B P_x^*(P, f)\, df.$$

5.2 Problem Formulation

In general, to show a channel capacity result, it is necessary to show achievability and converse. The achievability refers to the possibility of asymptotically achieving error-free communication at rates less than the capacity, and the converse shows the impossibility of asymptotically achieving error-free communication at rates exceeding the capacity.

Achievability results give lower bounds on the capacity. To establish the achievability of band-limited ACGN channels, one must demonstrate the possibility of constructing almost band-limited and almost time-limited codebooks that operate at a rate lower than the channel capacity, i.e., $R < C$. For a given error probability $P_e > 0$, the achievability provides us with a monotonically increasing sequence of achievable rates $\{R_n\}_{n\in\mathbb{N}}$ that converges to the capacity as the signal duration $\{T_n\}_{n\in\mathbb{N}}$ increases, i.e., $\{T_n\}_{n\in\mathbb{N}}$ is a monotonically increasing sequence of time duration. For $n \in \mathbb{N}$, the rate R_n describes the codebook size of band-limited signals of T_n time duration for which it is possible to find a decoder strategy, such that the error probability does not exceed P_e.

The converse gives an upper bound on the coding theorem. More specifically, a converse provides us with a monotonically decreasing sequence of rates $\{U_n\}_{n\in\mathbb{N}}$ that converges to the capacity. For every $n \in \mathbb{N}$, U_n is an upper bound on the codebook size of band-limited signals of T_n time duration for which it is possible to find a decoder strategy, such that the error probability does not exceed P_e.

Finding algorithms that can calculate both lower and upper bounds would be useful. Moreover, it would be desirable to have an algorithm that takes a band-limited ACGN channel and computes its corresponding capacity-achieving code. This prompts the following question:

Question 1: *Is it possible find an algorithm that takes a noise power spectrum N, a power constraint P, and a precision M as input and computes a codebook and a decoding strategy with rate R for the band-limited ACGN channel, such that $R \geq C(N, P) - \frac{1}{2^M}$ is achieved?*

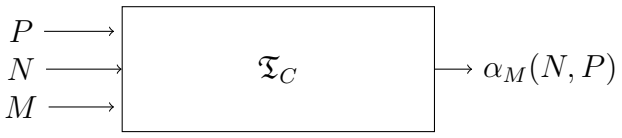

Figure 5.1: Turing machine $\mathfrak{T}_C$ for the computation of the capacity approximation of band-limited ACGN channels. It takes the power constraint P, noise power spectrum N and the approximation precision M and computes $\alpha_M(N,P)$ with $|C(N,P) - \alpha_M(N,P)| < \frac{1}{2^M}$.

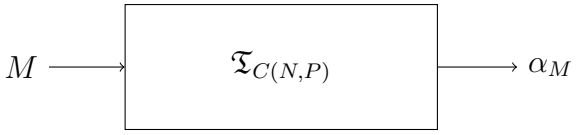

Figure 5.2: Turing machine $\mathfrak{T}_{C(N,P)}$ for the computation of the capacity approximation of band-limited ACGN channels for fixed N and P. For $P \in \mathbb{R}_c$ and N computable, $\mathfrak{T}_{(N,P)}$ takes the approximation precision M and computes α_M with $|C(N,P) - \alpha_M| < \frac{1}{2^M}$.

There has been a long-standing interest in the algorithmic computation of the capacity of communication scenarios in information theory. One example of such an algorithm is the Blahut-Arimoto algorithm, which can compute the capacity of any computable discrete memoryless channel given as input (see [4, 5]). However, there is no algorithm known to date that can compute the capacity of a band-limited ACGN channel with noise spectral density $N(f)$ similarly to the Blahut-Arimoto algorithm. Ideally, it is desirable to find such an algorithm.

Question 2: *Is it possible to find an algorithm that takes a noise spectral density N, a power constraint P, and a precision M as input and computes the number $\alpha_M(N,P)$ with*

$$|C(N,P) - \alpha_M(N,P)| < \frac{1}{2^M}?$$

The Turing machine that describes the algorithm of Question 2 is illustrated in Fig. 5.1.

We could simplify the requirements of the desired algorithm by fixing N and P. This prompts the following question:

Question 3: *For a fixed N and a fixed P, is it possible to find an algorithm that takes a precision M as input and computes the number α_M with*

$$|C(N,P) - \alpha_M| < \frac{1}{2^M}?$$

A Turing machine describing the algorithm of Question 3 is illustrated in Fig. 5.2.

If a constructive proof is found for Theorem 14, including an effective construction for capacity-achieving codes and an algorithmic description of the converse, then it would provide a positive answer to all three questions. However, our analysis demonstrates that for fixed N and P it is not possible to provide such a constructive proof.

5.3 Computability of the ACGN Channel Capacity

In this section, we aim to address Question 1, Question 2, and Question 3 from Section 5.1. Specifically, we construct an example of a noise power spectrum N for a band-limited ACGN channel that yields a negative answer to all three questions. To achieve this, we will introduce a band-limited ACGN channel that has computable parameters, including computable bandwidth, computable noise power spectrum, computable capacity-achieving power density spectrum, and a computable power constraint. Both the noise power spectrum and the capacity-achieving power spectrum will be computable continuous functions of the frequency domain $f \in \mathbb{R}$.

We consider the following band-limited channel with bandwidth $B \in \mathbb{R}_c$ and $B > 0$

$$y(t) = x(t) + z(t).$$

$N(f)$ is also band-limited with a B bandwidth. For $f \in [0, \frac{B}{2}]$, $N(f)$ is strictly monotonically decreasing and for $f \in [\frac{B}{2}, B]$ is strictly monotonically increasing. N is an even function with respect to $\frac{B}{2}$ and 0.

The communication is subject to a power constraint $P \in \mathbb{R}_c$, $P > 0$. The psd $P_x(f)$ with $f \in [-B, B]$ is a non-negative continuous function with

$$\int_{-B}^{B} P_x(f)\, df = P$$

We denote the capacity achieving psd by $P_x^*(f)$, which is uniquely determined by the water pouring technique.

We choose $f_1 \in \left(0, \frac{B}{2}\right]$, $f_1 \in \mathbb{R}_c$. We want to look for a capacity-achieving psd $P_{f_1}^*(f)$, that is different from zero only in the interval $\left[\frac{B}{2} - f_1, \frac{B}{2} + f_1\right]$. This optimal psd is uniquely coupled with the power P_{f_1} and is given by

$$P_{f_1} = \int_{\frac{B}{2}-f_1}^{\frac{B}{2}+f_1} \left(N\left(\frac{B}{2} + f_1\right) - N(f)\right) df.$$

Note that $N(\frac{B}{2} + f_1) = N\left(\frac{B}{2} - f_1\right)$. P_{f_1} is a computable number.

This means that when we are given a power P, such that for a certain $\hat{f}_1$ we have that $P = P_{\hat{f}_1}$, we have that the optimal power allocation is

$$P_{\hat{f}_1}^*(f) = \begin{cases} N\left(\frac{B}{2} + \hat{f}_1\right) - N(f), & \text{for } |f| \in \left[\frac{B}{2} + \hat{f}_1, \frac{B}{2} - \hat{f}_1\right] \\ 0, & \text{otherwise.} \end{cases}$$

The corresponding capacity can hence be expressed as a function of f_1 and is given by

$$C_1(N, f_1) = \int_0^B \ln(P_{f_1}^*(f) + N(f))\, df - \int_0^B \ln(N(f))\, df.$$

Now, if $C(N,P)$ is the capacity of the band-limited ACGN channel, then the following representation applies for $0 \leq f_1 \leq \frac{B}{2}$ and the corresponding power P_{f_1}:

$$C(N,P_{f_1}) = C_1(N,f_1),$$

i.e., the capacity is a function of f_1.

Theorem 15. *Let $B \in \mathbb{R}^c_{\geq 0}$. There are computable continuous functions $N \coloneqq [0,B] \to \mathbb{R}^c_{\geq 0}$, such that for each such function, and for all $f_1 \in [0,\frac{B}{2}]$, $f_1 \in \mathbb{R}_c$, we have that*

$$C_1(N,f_1) \notin \mathbb{R}_c.$$

Moreover, for each such function N, and for every $f_1 \in [0,\frac{B}{2}]$, $f_1 \in \mathbb{R}_c$, there is no computable sequence of computable numbers $\{u_n\}_{n\in\mathbb{N}}$ with $u_n \geq u_{n+1}$, $n \in \mathbb{N}$ and

$$\lim_{n\to\infty} u_n = C_1(N,f_1).$$

Proof. To prove the result of Theorem 15, we construct a non-negative computable continuous psd N. The construction of N is based on a recursively enumerable non-recursive set $\mathcal{A}$. There are countably infinitely many recursively enumerable non-recursive sets [38]. We denote $\{\mathcal{A}_i\}_{i\in\mathbb{N}}$ as the family of recursively enumerable non-recursive sets. For every such set $\mathcal{A}_i$, one can use the same approach to construct a different non-negative computable continuous psd N_i. The capacity of every N_i yields a non-computable number ξ_i.

Next, we start with the construction of the noise psd N. Let $B > 0$ be a fixed computable number. Let $n_0 \in \mathbb{N}$, such that $\frac{1}{n_0} < \frac{B}{2}$.

We consider the following function for $n \geq n_0$

$$G_n(f) = \begin{cases} -\frac{1}{|f-\frac{B}{2}|}, & \text{for } f \in \left[0,\frac{B}{2}-\frac{1}{n}\right] \cup \left[\frac{B}{2}+\frac{1}{n},\frac{B}{2}\right] \\ -n, & \text{for } f \in \left[\frac{B}{2}-\frac{1}{n},\frac{B}{2}+\frac{1}{n}\right] \end{cases} \tag{5.4}$$

G_n is a computable continuous function. Let

$$\begin{aligned} C_n &= \int_0^B G_n(f)\,df = 2\int_{\frac{B}{2}}^B G_n(f)\,df \\ &= -2\int_0^{\frac{1}{n}} n\,df - \int_{\frac{1}{n}}^{\frac{B}{2}} \frac{1}{f}\,df \\ &= -2 - 2\left(\log\frac{B}{2} - 2\ln\frac{1}{n}\right) \\ &= -2\left(1+\ln\frac{nB}{2}\right). \end{aligned} \tag{5.5}$$

Note that $\frac{nB}{2} > 1$, and hence $\ln\frac{nB}{2} > 0$.

We set $C_n^{(1)} = |C_n|$.

Let $\mathcal{A} \in \mathbb{N}$ be a recursively enumerable non-recursive set. Let $\varphi_{\mathcal{A}} \colon \mathbb{N} \to \mathcal{A}$ be a recursive function that lists all elements of the set $\mathcal{A}$.

We consider the following sequence of functions:

$$N_M(f) = \left(f - \frac{B}{2}\right)^2 \exp\left(\sum_{n=1}^{M} \frac{1}{2^{\varphi_{\mathcal{A}}(n)}} \frac{1}{C_n^{(1)}} G_n(f)\right) \tag{5.6}$$

N_M is a computable continuous function on $[0, B]$, since G_n are computable continuous functions for $1 \leq n \leq M$, the exponential function $\exp(\cdot)$ maps computable continuous functions to computable continuous functions, and the multiplication with $\left(f - \frac{B}{2}\right)^2$ generates, in any case, computable functions. Hence, $\{N_M\}_{M \in \mathbb{N}}$ is a computable sequence of computable continuous functions. N_M is itself a strictly monotonically increasing function in $[\frac{B}{2}, B]$ and an even function with respect to $\frac{B}{2}$.

Let $K \in \mathbb{N}$ be arbitrary. We have

$$\begin{aligned} N_{M+K}(f) - N_M(f) = \left(f - \frac{B}{2}\right)^2 &\exp\left(\sum_{n=1}^{M} \frac{1}{2^{\varphi_{\mathcal{A}}(n)}} \frac{1}{C_n^{(1)}} G_n(f)\right) \\ &\times \left[\exp\left(\sum_{n=M+1}^{M+K} \frac{1}{2^{\varphi_{\mathcal{A}}(n)}} \frac{1}{C_n^{(1)}} G_n(f)\right) - 1\right], \end{aligned}$$

$$\begin{aligned} |N_{M+K}(f) - N_M(f)| = \left(f - \frac{B}{2}\right)^2 &\exp\left(\sum_{n=1}^{M} \frac{1}{2^{\varphi_{\mathcal{A}}(n)}} \frac{1}{C_n^{(1)}} G_n(f)\right) \\ &\times \left|1 - \exp\left(\sum_{n=M+1}^{M+K} \frac{1}{2^{\varphi_{\mathcal{A}}(n)}} \frac{1}{C_n^{(1)}} G_n(f)\right)\right|. \end{aligned}$$

For $x \in [0, 1]$ we have

$$1 - e^{-x} \leq 2x.$$

Let $L \in \mathbb{N}$ with $L > n_0$ be arbitrary. On the interval $[0, \frac{B}{2} - \frac{1}{L}]$ and $[\frac{B}{2} + \frac{1}{L}, B]$ we have that for $M > L$

$$\frac{B}{2} - \frac{1}{M} > \frac{B}{2} - \frac{1}{L} \quad \text{and} \quad \frac{B}{2} + \frac{1}{M} < \frac{B}{2} + \frac{1}{L}.$$

Hence

$$G_M(f) = -\frac{1}{|f - \frac{B}{2}|} \quad \text{for} \quad f \in \left[0, \frac{B}{2} - \frac{1}{L}\right] \cup \left[\frac{B}{2} + \frac{1}{L}, B\right],$$

so that for $M > L$ the following holds:

$$\sum_{n=M+1}^{M+K} \frac{1}{2^{\varphi_{\mathcal{A}}(n)}} \frac{1}{C_n^{(1)}} G_n(f) = -\frac{1}{|f - \frac{B}{2}|} \sum_{n=M+1}^{M+K} \frac{1}{2^{\varphi_{\mathcal{A}}(n)}} \frac{1}{C_n^{(1)}}$$

and

$$\begin{aligned}\sum_{n=M+1}^{M+K} \frac{1}{2^{\varphi_{\mathcal{A}}(n)}} \frac{1}{C_n^{(1)}} &< \frac{1}{C_{M+1}^{(1)}} \sum_{n=M+1}^{M+K} \frac{1}{2^{\varphi_{\mathcal{A}}(n)}} \\ &< \frac{1}{C_{M+1}^{(1)}} \sum_{n=1}^{\infty} \frac{1}{2^{\varphi_{\mathcal{A}}(n)}} \\ &< \frac{1}{C_{M+1}^{(1)}} \sum_{n=1}^{\infty} \frac{1}{2^n} \\ &= \frac{1}{C_{M+1}^{(1)}}.\end{aligned}$$

Here we have used that $\{C_n^{(1)}\}_{n\in\mathbb{N}}$ is a monotonically increasing sequence. It also holds that

$$\begin{aligned}0 &\geq \sum_{n=M+1}^{M+K} \frac{1}{2^{\varphi_{\mathcal{A}}(n)}} \frac{1}{C_n^{(1)}} G_n(f) \\ &\geq -\frac{1}{|f-\frac{B}{2}|} \frac{1}{C_{M+1}^{(1)}}\end{aligned}$$

We also have that

$$\frac{1}{|f-\frac{B}{2}|} \leq \frac{1}{|\frac{B}{2}+\frac{1}{L}-\frac{B}{2}|} = L,$$

hence

$$0 \geq \sum_{n=M+1}^{M+K} \frac{1}{2^{\varphi_{\mathcal{A}}(n)}} \frac{1}{C_n^{(1)}} G_n(f) \geq -\frac{L}{C_{M+1}^{(1)}}. \tag{5.7}$$

For $M \in \mathbb{N}$ and $K \in \mathbb{N}$, such that $M > U3^{L^2}$ with $U \in \mathbb{N}$, $U \geq 1$, $\frac{B}{2}U > 1$ we have that

$$\begin{aligned}|N_{M+K}(f) - N_M(f)| &\leq \left(f-\frac{B}{2}\right)^2 \exp\left(\sum_{n=1}^{M} \frac{1}{2^{\varphi_{\mathcal{A}}(n)}} \frac{L}{C_n^{(1)}} G_n(f)\right) \frac{L}{C_{M+1}^{(1)}} \\ &< \left(\frac{B}{2}\right)^2 \times 1 \times \frac{L}{2(1+\ln\frac{BU3^{L^2}}{2})} \qquad (5.8) \\ &= \left(\frac{B}{2}\right)^2 \times 1 \times \frac{L}{(2+2\log\frac{BU}{2}+2\ln 3^{L^2})} \\ &< \left(\frac{B}{2}\right)^2 \frac{L}{(2\ln 3^{L^2})} \\ &< \left(\frac{B}{2}\right)^2 \frac{L}{2L^2}. \qquad (5.9)\end{aligned}$$

(5.8) follows from (5.5).

Let U_1 be the smallest natural number such that $U_1 > \frac{B}{2}$. Then for all $L > n_0$, $L \in \mathbb{N}$, for all $M \geq U3^{L^2}$, for all $K \in \mathbb{N}$, and for all $f \in [0, \frac{B}{2} - \frac{1}{L}] \cup [\frac{B}{2} + \frac{1}{L}, B]$, we have that

$$|N_{M+K}(f) - N_M(f)| < \frac{U_1^2}{2L}.$$

For $f \in [\frac{B}{2} - \frac{1}{L}, \frac{B}{2} + \frac{1}{L}]$ we have the following:

$$\begin{aligned}
|N_{M+K}(f) - N_M(f)| &\leq \left(f - \frac{B}{2}\right)^2 \\
&\quad \times \left[\exp\left(\sum_{n=1}^{M+K} \frac{1}{2^{\varphi_{\mathcal{A}}(n)}} \frac{1}{C_n^{(1)}} G_n(f)\right) + \exp\left(\sum_{n=1}^{M} \frac{1}{2^{\varphi_{\mathcal{A}}(n)}} \frac{1}{C_n^{(1)}} G_n(f)\right)\right] \\
&\leq 2\left(f - \frac{B}{2}\right)^2 \\
&\leq 2\left(\frac{B}{2} + \frac{1}{L} - \frac{B}{2}\right)^2 \\
&\leq \frac{1}{2L^2}.
\end{aligned}$$

Thus $\{N_M\}_{M \in \mathbb{N}}$ is an effective Cauchy sequence of computable continuous functions, and it converges effectively to the function

$$N(f) = \left(f - \frac{B}{2}\right)^2 \exp\left(\sum_{n=1}^{\infty} \frac{1}{2^{\varphi_{\mathcal{A}}(n)}} \frac{1}{C_n^{(1)}} G_n(f)\right). \tag{5.10}$$

Eq. (5.10) describes an algorithm that takes the recursive function $\varphi_{\mathcal{A}}$ as input and computes N. N is a computable continuous function with $N(f) \geq 0$ for $f \in [0, B]$. N is a strictly monotonically increasing function in the interval $[\frac{B}{2}, B]$ and it is an even function with respect to $\frac{B}{2}$.

We take an $f_1 \in (0, B]$, $f_1 \in \mathbb{R}_c$, and compute the number $C_1(N, f_1)$. For

$$P_{f_1} = \int_{\frac{B}{2} - f_1}^{\frac{B}{2} + f_1} \left(N\left(\frac{B}{2} + f_1\right) - N(f)\right) df$$

we have that

$$\begin{aligned}
C_1(N, f_1) &= \int_{\frac{B}{2} - f_1}^{\frac{B}{2} + f_1} \ln(P_{f_1}^*(f) + N(f))\, df + \int_0^{\frac{B}{2} - f_1} \ln(N(f))\, df \\
&\quad + \int_{\frac{B}{2} + f_1}^{B} \ln(N(f))\, df - \int_0^{B} \ln(N(f))\, df \\
&= \int_{\frac{B}{2} - f_1}^{\frac{B}{2} + f_1} \ln\left(N\left(\frac{B}{2} + f_1\right)\right) df_1 - \int_{\frac{B}{2} - f_1}^{\frac{B}{2} + f_1} \ln(N(f))\, df.
\end{aligned} \tag{5.11}$$

Since $f_1 \in \mathbb{R}_c$, we have that $\frac{B}{2} + f_1 \in \mathbb{R}_c$ and hence $N(\frac{B}{2} + f_1) \in \mathbb{R}_c$ and $N(\frac{B}{2} + f_1) > 0$. Consequently, we have that $\ln(N(\frac{B}{2} + f_1)) \in \mathbb{R}_c$ and therefore also $2f_1 \ln N(\frac{B}{2} + f_1) \in \mathbb{R}_c$.

Now we have to rewrite the number

$$\begin{aligned} Z(f_1) &= \int_{\frac{B}{2}-f_1}^{\frac{B}{2}+f_1} \ln N(f)\, df \\ &= \int_0^B \ln N(f)\, df - \int_0^{\frac{B}{2}-f_1} \ln N(f)\, df - \int_{\frac{B}{2}+f_1}^B \ln N(f)\, df \end{aligned} \tag{5.12}$$

We then have

$$\int_0^B \ln N(f)\, df = \int_0^B \ln\left(f - \frac{B}{2}\right)^2 df + \int_0^B \ln\left(\exp\left(\sum_{n=1}^{\infty} \frac{1}{2^{\varphi_{\mathcal{A}}(n)}} \frac{1}{C_n^{(1)}} G_n(f)\right)\right) df$$

We have that $\ln\left(\cdot - \frac{B}{2}\right)^2$ is a computable function in $\mathcal{L}^1[0,B]$, see [33]. This way, we have that

$$\int_0^B \ln\left(\cdot - \frac{B}{2}\right)^2 df \in \mathbb{R}_c. \tag{5.13}$$

Furthermore, we also have that

$$\exp\left(\sum_{n=1}^{\infty} \frac{1}{2^{\varphi_{\mathcal{A}}(n)}} \frac{1}{C_n^{(1)}} G_n(f)\right) = 2^{\log_2 \exp\left(\sum_{n=1}^{\infty} \frac{1}{2^{\varphi_{\mathcal{A}}(n)}} \frac{1}{C_n^{(1)}} G_n(f)\right)}$$

and

$$\int_0^B \log_2\left(\exp\left(\sum_{n=1}^{\infty} \frac{1}{2^{\varphi_{\mathcal{A}}(n)}} \frac{1}{C_n^{(1)}} G_n(f)\right)\right) df = \log_2(e) \int_0^B \sum_{n=1}^{\infty} \frac{1}{2^{\varphi_{\mathcal{A}}(n)}} \frac{1}{C_n^{(1)}} G_n(f)\, df.$$

We consider the following function for $f \in [0,B]$ and $M \in \mathbb{N}$:

$$\psi_M(f) = \sum_{n=1}^{M} \frac{1}{2^{\varphi_{\mathcal{A}}(n)}} \frac{1}{C_n^{(1)}} G_n(f).$$

Note that ψ is a continuous function. For $K \in \mathbb{N}$, it holds that

$$\begin{aligned} \int_0^B |\psi_{M+K}(f) - \psi_M(f)|\, df &= \int_0^M \sum_{n=M}^{M+K} \left| \frac{1}{2^{\varphi_{\mathcal{A}}(n)}} \frac{1}{C_n^{(1)}} G_n(f) \right| df \\ &\leq \sum_{n=M}^{M+K} \frac{1}{2^{\varphi_{\mathcal{A}}(n)}} \frac{1}{C_n^{(1)}} \int_0^M |G_n(f)|\, df \qquad (5.14) \\ &= \sum_{n=M}^{M+K} \frac{1}{2^{\varphi_{\mathcal{A}}(n)}} \qquad (5.15) \\ &< \sum_{n=M}^{\infty} \frac{1}{2^{\varphi_{\mathcal{A}}(n)}}. \end{aligned}$$

Eq. (5.14) holds due to the triangle inequality for the ℓ_1-norm. Eq. (5.15) holds from the definition of $C_n^{(1)}$. Consequently, the sequence $\{\psi_M(f)\}_{M\in\mathbb{N}}$ converges in the ℓ_1-norm to the function

$$\psi(f) = \sum_{n=M}^{\infty} \frac{1}{2^{\varphi_{\mathcal{A}}(n)}} \frac{1}{C_n^{(1)}} G_n(f)$$

for $f \in [0, B]$.

Thus, we have that

$$\begin{aligned}
\int_0^B \psi(f)\, df &= \lim_{M\to\infty} \int_0^B \sum_{n=1}^{M} \frac{1}{2^{\varphi_{\mathcal{A}}(n)}} \frac{1}{C_n^{(1)}} G_n(f)\, df \\
&= \lim_{M\to\infty} \sum_{n=0}^{M} \frac{1}{2^{\varphi_{\mathcal{A}}(n)}} \frac{1}{C_n^{(1)}} \int_0^B G_n(f)\, df \\
&= \lim_{M\to\infty} -\sum_{n=0}^{M} \frac{1}{2^{\varphi_{\mathcal{A}}(n)}} \\
&= -\sum_{n=0}^{\infty} \frac{1}{2^{\varphi_{\mathcal{A}}(n)}}.
\end{aligned}$$

Since $\mathcal{A} \subset \mathbb{N}$ is a recursively enumerable non-recursive set and from [33, Chapter 1], we have

$$-\sum_{n=0}^{\infty} \frac{1}{2^{\varphi_{\mathcal{A}}(n)}} = \xi \notin \mathbb{R}_c.$$

Next, we must analyze the integral $\int_0^{\frac{B}{2}-f_1} \log N(f)\, df$. We have already shown in the inequality (5.7) that for $L \in \mathbb{N}$ with $\frac{1}{L} < f_1$, it always holds the following relation for $M > U3^{L^2}$:

$$|N_{M+K}(f) - N_M(f)| < \frac{L}{C_{M+1}^{(1)}}.$$

Following similar calculations as in Eq. (5.9), for $f \in [0, \frac{B}{2} - f_1]$ and $M > U3^{L^2}$ we have that

$$|\psi_{M+K}(f) - \psi_M(f)| < \frac{1}{L}.$$

Since $\{\psi_M(f)\}_{M\in\mathbb{N}}$ is a computable continuous sequence of continuous functions on $[0, \frac{B}{2} - f_1]$. This sequence converges effectively on $[0, \frac{B}{2} - f_1]$ to the function ψ. ψ is itself a computable continuous function on $[0, \frac{B}{2} - f_1]$. With this and from [33], it follows that

$$\int_0^{\frac{B}{2}-f_1} \psi(f)\, df \in \mathbb{R}_c. \tag{5.16}$$

Following the same line of arguments, we get that ψ is also a computable continuous function on the interval $[\frac{B}{2} + f_1, B]$, and hence

$$\int_{\frac{B}{2}+f_1}^{B} \psi(f)\, df \in \mathbb{R}_c.$$

From Eqs. (5.12), (5.16) and (5.3) it must hold that

$$Z(f_1) \notin \mathbb{R}_c,$$

and hence

$$C_1(N, f_1) \notin \mathbb{R}_c.$$

Since $f_1 \in \mathbb{R}_c$ can take any value in the interval $(0, \frac{B}{2}]$, we have shown the first statement of Theorem 15.

Next, we show the second statement. We prove this by contradiction and assume that the second statement is wrong. Assume that there is a $f_1 \in (0, \frac{B}{2}]$, $f_1 \in \mathbb{R}_c$ so that we can find a computable sequence of computable numbers $\{u_n\}_{n\in\mathbb{N}}$, such that the following holds:

$$u_n \geq u_{n+1} \text{ for } n \in \mathbb{N} \quad \text{and } \lim_{n\to\infty} u_n = C_1(N, \hat{f}_1).$$

From the proof of the first statement, we have that

$$C_1(N, \hat{f}_1) = a(\hat{f}_1) + \xi$$

with $a(\hat{f}_1) \in \mathbb{R}_c$ and $\xi = -\sum_{n=1}^{\infty} \frac{1}{2^{\varphi_{\mathcal{A}}(n)}} \notin \mathbb{R}_c$.

We set $U(M) = -\sum_{n=1}^{M} \frac{1}{2^{\varphi_{\mathcal{A}}(n)}}$ for $M \in \mathbb{N}$. Then we have a computable sequence of computable numbers with

$$U(M) \geq U(M+1) \quad \text{and } \lim_{M\to\infty} U(M) = \xi.$$

If $C_1(N, f_1)$ were the limit value of a monotonically decreasing sequence of computable numbers, then this would also hold for $C_1(N, f_1) - a(\hat{f}_1)$. Since $a(\hat{f}_1)$ is a computable number, there is a monotonically increasing computable sequence $\{m_n\}_{n\in\mathbb{N}}$ of computable numbers with $\lim_{n\to\infty} m_n = a(\hat{f}_1)$. Furthermore, since $a(\hat{f}_1) \geq m_n$ it also holds that $-a(\hat{f}_1) \leq -m_n$. We then have

$$u_n - m_n \geq C_1(N, \hat{f}_1) - a(\hat{f}_1) \quad n \in \mathbb{N}$$

and

$$u_{n+1} - m_{n+1} \leq u_n - m_{n+1} \leq u_n - m_n,$$

i.e.,$\{u_n - m_n\}_{n\in\mathbb{N}}$ is a computable sequence of computable numbers and the sequence is monotonically decreasing. It then holds that

$$\lim_{n\to\infty}(u_n - m_n) = \lim_{n\to\infty} u_n - \lim_{n\to\infty} m_n = C_1(N, \hat{f}_1) - a(\hat{f}_1) = \xi. \tag{5.17}$$

This way is ξ the limit value of computable sequences. One of the computable sequences is a monotonically decreasing sequence and the other one is a monotonically increasing sequences. This automatically implies that $\xi \in \mathbb{R}_c$ which is a contradiction. This contradiction shows that our assumption is wrong and hence there is no monotonically decreasing computable sequence of computable numbers that converges to $C_1(N, \hat{f}_1)$. □

Remark 20. *Theorem 15 states that there are band-limited ACGN channels whose capacities are non-computable numbers. This result is the second known instance in information theory where capacity has been proven to be non-computable, following the compound channel case in [10]. In that study, the authors considered a computable compound channel $\{W_n\}_{n\in\mathbb{N}}$ with finite input and output alphabets. $\{W_n\}_{n\in\mathbb{N}}$ is a computable sequence, and $C(\{W_n\}_{n\in\mathbb{N}}) \notin R_c$. The capacity $C(\{W_n\}_{n\in\mathbb{N}})$ is the limit value of a monotonically decreasing computable sequence $\{u_n\}_{n\in\mathbb{N}}$ that serves as a computable upper bound for $C(\{W_n\}_{n\in\mathbb{N}})$, but there exists no computable sequence of lower bounds that converges to the capacity. In contrast, Theorem 15 shows that for $C(N,P)$, the capacity of the band-limited ACGN channel with colored noise, we have the opposite situation.*

Remark 21. *In previous works such as [66, 7, 71], the capacity of the ACGN channel was typically related to the capacity of the discrete Gaussian channel. This was achieved through a discrete approximation of the time-continuous Gaussian channel. When analyzing these solutions, it is observed that as the approximation of the discrete channels becomes finer, the sequence of capacities of the discrete channels approaches the capacity of the time-continuous ACGN channel. However, a stopping criterion for the approximation process has not yet been identified. In our case, such a stopping criterion refers to an algorithm that takes an approximation error of $\frac{1}{2^M}$ as input for the computation of the capacity of a fixed ACGN channel, and then the algorithm stops the approximation process when the result of the computation is within a margin of error of $\frac{1}{2^M}$ from the capacity of the time-continuous ACGN channel. Our result shows that there are band-limited ACGN channels with color noise for which such a stopping criterion cannot exist.*

Remark 22. *We not only demonstrate the existence of a non-negative computable continuous noise spectral density, but we also develop an algorithm that can effectively construct a noise power spectrum N for which the conclusion of Theorem 15 holds. The algorithm takes a recursive function $\varphi_{\mathcal{A}}$ as input and computes N. The recursive function $\varphi_{\mathcal{A}}$ generates a recursively enumerable non-recursive set $\mathcal{A}$. There are countably infinitely many recursive enumerable non-recursive sets $\{\mathcal{A}_1, \mathcal{A}_2, \dots\}$. By applying the same algorithm to the generative function $\varphi_{\mathcal{A}_i}$ of any other recursively enumerable non-recursive set $\mathcal{A}_i$, we obtain a different computable noise power spectrum N_i which has the same structure as N and that satisfies Theorem 15.*

Theorem 16. *Let $B \in \mathbb{R}^c_{\geq 0}$. There are computable continuous functions $N := [-B, B] \to \mathbb{R}^c_{\geq 0}$, such that for each such function there are infinitely many $\hat{P} \in [0, P_*]$, $\hat{P} \in \mathbb{R}_c$ where*

$$P_* = \int_{-B}^{B} (N(0) - N(f))\, df,$$

and for which the following holds:

$$C(N, \hat{P}) \notin \mathbb{R}_c.$$

Furthermore, for each such function N, there is no computable sequence of computable numbers $\{u_n\}_{n\in\mathbb{N}}$ with $u_n \geq u_{n+1}$, $n \in \mathbb{N}$, and

$$\lim_{n\to\infty} u_n = C(N, \hat{P}).$$

Proof. We consider the non-negative and computable continuous psd N from Eq. (5.10).

Let $f_1 \in (0, \frac{B}{2})$, $f_1 \in \mathbb{R}_c$ be arbitrary but fixed. We have that

$$P_{f_1} = \int_{\frac{B}{2}-f_1}^{\frac{B}{2}+f_1} (N(B+f_1) - N(f))\, df$$

is the corresponding power concentrated in the interval $[\frac{B}{2}-f_1, \frac{B}{2}+f_1]$. It holds that $\hat{P} = P_{f_1}$ and hence $C(N, \hat{P}) = C_1(N, f_1)$ however we have already shown that $C_1(N, f_1) \notin \mathbb{R}_c$. This way we have proven the first statement.

Consider the family of recursively enumerable non recursive sets $\{\mathcal{A}_i\}_{i\in\mathbb{N}}$. This result holds for every N computed from the algorithm for Eq. (5.10) that takes as input any recursive function $\varphi_{\mathcal{A}_i}$ generating a recursively enumerable non-recursive set $\mathcal{A}_i$. Note that $\hat{P}$ is also a function of $\varphi_{\mathcal{A}}$, since it depends on N, which in turn is determined by $\varphi_{\mathcal{A}}$.

The proof of the second statement of the theorem follows the same line of argument as in the proof for the second statement of Theorem 16. □

Theorem 17. *Let $B \in \mathbb{R}^c_{\geq 0}$. There are computable continuous functions $N := [-B, B] \to \mathbb{R}^c_{\geq 0}$, such that for each such function, and for every $P > P_*$ with $P \in \mathbb{R}_c$ and*

$$P_* = \int_{-B}^{B} (N(0) - N(f))\, df$$

we have that

$$C(N, P) \notin \mathbb{R}_c.$$

Furthermore, for each such function N, there is no computable sequence of computable numbers $\{u_n\}_{n\in\mathbb{N}}$ with $u_n \geq u_{n+1}$, $n \in \mathbb{N}$, and

$$\lim_{n\to\infty} u_n = C(N, P).$$

Proof. We consider the non-negative and computable continuous psd N from Eq. (5.10).

Let $P > P_*$, $P \in \mathbb{R}_c$ be arbitrary but fixed.

We have that

$$P = P_* + \Delta B$$

hence $\Delta = \frac{P-P_*}{B}$. Since $P, P_*, B \in \mathbb{R}_c$ then we have that $\Delta \in \mathbb{R}_c$.

We then have that the optimal psd for P is given by

$$P_x^*(P, f) = N(0) + \Delta - N(f)$$

for $f \in [0, B]$. This way we have

$$\begin{aligned} C(N,P) &= \int_0^B \ln(P_x^*(P,f) + N(f))\,df - \int_0^B \ln N(f)\,df \\ &= B\ln(N(0)+\Delta) - \int_0^B \ln N(f)\,df. \end{aligned}$$

We have that $B\ln(N(0)+\Delta) \in \mathbb{R}_c$ however we have already shown that $\int_0^B \ln N(f)\,df \notin \mathbb{R}_c$. This implies that $C(N,P) \notin \mathbb{R}_c$, which proves the first statement of the theorem.

Consider the family of recursively enumerable non-recursive sets $\{\mathcal{A}_i\}_{i\in\mathbb{N}}$. This result holds for every N computed from the algorithm for Eq.(5.10) that takes any recursive function $\varphi_{\mathcal{A}_i}$ generating a recursively enumerable non-recursive set $\mathcal{A}_i$. Note that P_* is also a function of $\varphi_{\mathcal{A}}$ since it depends on N, which in turn is determined by $\varphi_{\mathcal{A}}$.

To prove the second statement, we have to follow the same line of arguments as in the proof of the second statement of Theorem 15. □

Corollary 3. *There are infinitely many P with $P \in \mathbb{R}_c$ that fulfill the conditions of Theorem 16 or 17, and for which there is no computable sequence of computable upper-bounds $\{u_n\}_{n\in\mathbb{N}}$ with*

$$\lim_{n\to\infty} u_n = C_1(N, f_1).$$

Proof. Assume there is a computable sequence of computable upper bounds $\{u_n\}_{n\in\mathbb{N}}$ with $u_n \geq C(N,P)$ for all $n \in \mathbb{N}$. Consider N from Eq. (5.10). Let $\hat{u}_n$ be such that

$$\hat{u}_n = \min_{1\leq k\leq n} u_k.$$

$\{\hat{u}_n\}_{n\in\mathbb{N}}$ is a monotonically decreasing computable sequence of computable numbers. It then holds that

$$\lim_{n\to\infty} \hat{u}_n = C(N,P).$$

This implies that $C(N,P)$ must be a computable number. However, in the proof of Theorem 15, we have shown that $C(N,P) \notin \mathbb{R}_c$, leading to the conclusion that our initial assumption must be incorrect. □

Remark 23. *Corollary 3 states that we can find a computable sequence of achievable rates $\{R_n\}_{n\in\mathbb{N}}$ that converges effectively to the capacity, making the achievability part algorithmically computable. However, it is impossible to algorithmically compute how far the achievable rates R_n are from the capacity. Its implications are beyond the inability to compute a capacity-achieving codebook. Even if we relax the requirement to achieve capacity and allow for some decoding error, it is still impossible to compute an upper bound on the size of the codebook.*

Remark 24. *It is interesting to note that while there exist examples of band-limited ACGN channels with computable power spectral densities whose capacities are non-computable numbers, this does not necessarily imply that the converse results of non-computable capacities are also non-computable in general. By non-computable converses,*

we mean that there is no computable sequence of computable asymptotically sharp upper-bounds. To this end, consider the compound channel. Recent computability studies in [10] have shown a converse result: while the compound capacity's converse is computable, i.e., there exist computable sequences of computable upper-bounds that are asymptotically sharp, the achievability of this capacity is not algorithmically computable, i.e., there are no computable sequences of computable lower-bounds that are asymptotically sharp.

5.4 Conclusions

In this chapter, we have focused on studying the algorithmic properties of a simple communication channel: the band-limited ACGN channel. We have shown that there are ACGN channels whose capacities are non-computable numbers. Thus, for a given computable bandwidth, noise power spectrum, and power constraint, there is no algorithm that can effectively compute the capacity of such a channel within a certain desired precision level. Moreover, we have also shown that the converse result for those channels is also not algorithmically computable. Although one can algorithmically construct a sequence of achievable rates that converges to the capacity, it is impossible to compute how far they are from the capacity. So it is impossible to algorithmically compute an upper bound on the size of the codebook for the channel.

We have also studied the influence of the power constraint on the computability of the capacity of ACGN channels. Unfortunately, we have shown that for those computable channels whose capacity yields a non-computable number adjusting the power constraint does not influence the computability property of the capacity. Moreover, adjusting the power constraint would not enable one to algorithmically compute upper bounds on the capacity.

For more complex channels, such as the FSC, FSC with feedback, and identification of correlation-assisted DMC, it has been shown that the capacity is not Borel-Turing computable, meaning there is no universal algorithm capable of computing the capacity for any channel. However, it is still an open problem whether the capacity of those channels can be computed as a number. By showing that the capacity of this particular ACGN channel is a non-computable number, it immediately implies that the capacity cannot be expressed as a computable function of the channel and power constraint parameters. Therefore, there is no universal algorithm that can take a noise power spectrum, bandwidth, and power constraint as inputs and compute the capacity based on those parameters.

6 Complexity of Computing the Additive Colored Gaussian Noise Channel Capacity

In Chapter 5, it was shown that there exist band-limited computable noise psds whose capacities yield non-computable numbers. This implies that the capacity C of the band-limited ACGN channel is in general a non-computable function of its parameter, in this case, the noise psd N and power constraint P. The question now is whether it is possible to restrict the set of band-limited ACGN channels such that the capacity becomes computable for this set of channels. In other words, is there a Turing machine that computes an approximation α_M of the capacity $C(P, N)$? This means, that for every precision $M \in \mathbb{N}$, the Turing machine computes an approximation α_M such that

$$|C(P, N) - \alpha_M| \leq \frac{1}{2^M}.$$

In this chapter, the objective is to establish a sufficient condition for an ACGN channel to have a computable capacity. Once we have determined the subset of band-limited ACGN channels, whose capacities yield a computable number, we delve into the computational complexity associated with approximating the capacity of these channels. Our focus is on understanding the computational complexity of approximating the channel capacity when the input parameters, namely N and P, exhibit low complexity. Additionally, we examine the implications of the complexity of computing the capacity of band-limited ACGN channels for the implementation of finite blocklength performances.

6.1 Problem Formulation

We consider the band-limited ACGN channel with noise psd N on $[0, B]$. N is continuous and strictly positive. For the average noise power spectrum $\bar{N}$ with

$$\bar{N} = \frac{1}{B} \int_0^B N(f)\, df.$$

Let $L_* = \max_{f\in[0,B]} N(f)$ and $\hat{P} = LB - B\bar{N}$. We have that for a power constraint $P > \hat{P}$ and from Theorem 14, the optimal psd of the signal is given by $P_x^*(P, f) = (L - N(f))$, where $\int_0^B P_x^*(P, f)\, df = P$. Hence,

$$LB - \int_0^B N(f)\, df = P,$$

which implies that

$$L = \frac{P}{B} + \frac{1}{B}\int_0^B N(f)\,df.$$

From Theorem 14 we have that the capacity of the band-limited ACGN channel with power constraint P and noise psd N is

$$\begin{aligned} C(P,N) &= \int_0^B \ln(P_x^*(P,f) - N(f))\,df - \int_0^B \ln(N(f))\,df \\ &= B\ln L - \int_0^B \ln(N(f))\,df. \end{aligned} \tag{6.1}$$

We consider noise spectral densities N which are infinitely differentiable, strictly positive and can be computed in polynomial time. For these channels, we are interested in the capacity-achieving psds P_x^* whose values are computable numbers. This prompts the following question:

Question 1: What is the computational complexity of computing the values of the capacity-achieving psd $P_x^*(P,\cdot)$ as a function of the frequency?

In [21], it was shown, that there are ACGN channels with computable continuous non-negative psds, whose capacities yield a non-computable number. We consider noise spectral densities N which are infinitely differentiable, strictly positive and can be computed in polynomial time. Assume P is a rational number. We consider the computable subset of such N and P whose capacity yields a computable number. Namely, let M be the desired precision for the computation of the capacity, i.e., the capacity lies at most $\frac{1}{2^M}$ away from the computed approximation. Then there exists a Turing machine M_C, that takes P, N, M as input and computes an approximation $\tilde{C} = M_C(P,N,M)$ of the capacity $C(P,N)$ of the ACGN channel with psd N and power constraint P such that $|\tilde{C} - C(P,N)| \leq \frac{1}{2^M}$. We are interested in knowing how much time does such a Turing machine need to compute the capacity approximation within the desired precision. This prompts the following question:

Question 2: What is the computational complexity of computing the capacity $C(P,N)$?

We formalize and answer Question 1 and Question 2 in Section 6.2 using the framework of complexity theory introduced in Section 2.3.4.

Currently, the computation of problems is limited to digital machines. Therefore, to examine the fundamental limits of today's computers and address the former questions, the application of Alan Turing's computability theory, specifically Turing machines, is of central importance.

6.2 Complexity Blowup of the ACGN Capacity Computation

In this section we focus on studying the computational complexity property of the capacity-achieving psd and the capacity of the band-limited ACGN channel. The goal is to answer Question 1 and Question 2.

First we focus on the capacity-achieving psd. The capacity-achieving psd of the band-limited ACGN, can be approached using the water pouring technique. To study the complexity of computing the capacity-achieving psd we consider a band-limited channel with rational bandwidth and power constraint and a polynomial time computable continuous noise spectral density. The following theorem classifies the complexity class to which the computation of the capacity-achieving psd for $f = 0$ belongs.

Theorem 18. *Let $B \in \mathbb{R}_c$ be a polynomial time computable number, $N\colon [0, B] \to \mathbb{R}$ be a polynomial time computable continuous function and $P \in \mathbb{Q}$ with $P > \hat{P}$ be arbitrary. Then $P_x^*(P, 0)$ is in $\#\mathrm{P}_1$. Furthermore, there exists a strictly positive computable noise psd N_* that is infinitely differentiable on $[0, B]$, such that for all $P > \hat{P}_*$, where $\bar{N}_*$ is the average noise psd and $P \in \mathbb{Q}$, the function $P_x^*(P, 0)$ is complete in $\#\mathrm{P}_1$.*

The proof of Theorem 18 consists of two parts. In the first part, it is shown that for strictly positive computable continuous functions, the computation of the value $P_x(P, 0)$ for $P \in \mathbb{Q}$ is always in $\#\mathrm{P}_1$. This part of the proof thus provides an upper bound for the complexity class for the calculation of the value $P_x(P, 0)$.

The second part of the proof of Theorem 18 provides, for a special infinitely differentiable computable continuous function $\bar{N}$, a lower bound to the complexity class for computing the value $P_x(P, 0)$ for all $P \in \mathbb{Q}$, $P > \hat{P}$.

Proof. Let N be a strictly positive and polynomial time computable noise psd. We have that

$$P_x^*(P, f) = L - N(f), \quad f \in [0, B]$$

with

$$L = \frac{1}{B}(P + \int_0^B N(f)\, df).$$

Since B is a polynomial time computable number, we have that $\frac{1}{B}$ is also polynomial time computable. $P \in \mathbb{Q}$ and hence P is also a polynomial time computable number. From Theorem 4, we have that the computation of $\int_0^B N(f)\, df$ is in $\#\mathrm{P}_1$. This implies that L is also in $\#\mathrm{P}_1$. This proves the first statement of the theorem.

From Theorem 4, we have that there exists a computable function g defined on the interval $[0, 1]$ that is infinitely differentiable and polynomial time computable, such that the computation of $\int_0^1 g(f)\, df$ is $\#\mathrm{P}_1$-complete. We consider the function

$$g_1(f) \coloneqq \frac{1}{B} g\left(\frac{f}{B}\right) \quad 0 \le f \le B.$$

g_1 is an infinitely differentiable function, and it is polynomial time computable. There is a rational number α such that for every $\omega \in [0, 1]$

$$N_*(f) = \alpha + g_1(\omega)$$

fulfills the condition $\min_{f\in[0,B]} N_*(f) > 0$.

Then, for $P \in \mathbb{Q}$, $P > \frac{1}{B}\int_0^B N_*(f)\, df = \bar{N}_*$ we have

$$L = \frac{1}{B}(P + \int_0^B N_*(f)\, df).$$

Thus with

$$\int_0^B N_*(f)\,df = \int_0^1 g(f)\,df + \alpha B$$

we have

$$LB - P - \alpha B = \int_0^1 N_*(f)\,df.$$

This way, the computation of the number $LB - P - \alpha B$ is complete in $\#\mathrm{P}_1$. Since $P, \alpha \in \mathbb{Q}$ and B are polynomial time computable, we have that the computation of L is $\#\mathrm{P}_1$-complete.

Now we have

$$P_x^*(P, 0) = L - N_*(0).$$

Since $N_*(0)$ is polynomial time computable, we have that the computation of $P_x^*(P, 0)$ is $\#\mathrm{P}_1$-complete. □

Remark 25. *Theorem 18 considers parameters with simple characteristics, specifically, rational bandwidth and power constraints, along with continuous noise spectral densities that are computable in polynomial time. Since both the bandwidth and power constraints are rational, their computation can also be achieved in polynomial time. While all three parameters are polynomial time computable, it is only established that the capacity-achieving psd belongs to the class* $\#\mathrm{P}_1$*. Whether the capacity-achieving psd can be expressed as a polynomial time computable function of the frequency remains an unanswered question. Resolving this question would require understanding the relationship between* FP_1 *and* $\#\mathrm{P}_1$*, which is currently an open problem. It is widely assumed, though not proven, that* $\mathrm{FP}_1 \neq \#\mathrm{P}_1$*. If this assumption holds true, then we can derive the following result.*

From Theorem 18, we have that computing the capacity-achieving psd of the band-limited ACGN channel is complete in $\#\mathrm{P}_1$. This problem is more complex than NP-complete problems. #P-complete problems are generally more complex than NP-complete problems. While both classes represent computationally challenging problems, #P-complete problems involve counting or enumerating solutions, which typically requires more computational resources than verifying a solution (as in NP-complete problems).

Even computing the capacity-achieving distribution for discrete memoryless channels (DMCs) is challenging. While there exist algorithms that can compute the capacity of DMCs [4, 5], a general stopping criterion for computing the capacity-achieving distributions for DMCs cannot exist [6]. Even in cases where the capacity-achieving distribution of DMCs becomes computable, no definitive assertions have been made about the complexity of computing the optimal distribution. On the other hand, in the case of continuous channels, more precisely, the band-limited ACGN channel, Theorem 18 demonstrates that computing the capacity-achieving psd is $\#\mathrm{P}_1$-complete, indicating its hardness comparable to solving any problem in $\#\mathrm{P}_1$.

Theorem 19. *If* $\mathrm{FP}_1 \neq \#\mathrm{P}_1$*, then there exists an infinitely differentiable noise psd* N_* *defined on* $[0, B]$*, which is computable in polynomial time, such that* $P_x^*(P, 0)$ *for* $P \in \mathbb{Q}$ *and* $P > \hat{P}_*$ *cannot be computed in polynomial time.*

Proof. With Theorem 18, we have the complete characterization of the computational complexity of the capacity-achieving psd for $f = 0$ fully characterized. This result holds for all rational frequencies $f \in [0, B]$. □

Remark 26. *Based on Theorem 19, if the widely accepted assumption* $\mathrm{FP}_1 \neq \#\mathrm{P}_1$ *holds true, it implies the existence of a noise psd* N_* *that can be computed in polynomial time. Remarkably, in this case, the capacity-achieving psd exhibits a complexity-blowup phenomenon.*

Next, we focus on the capacity function of the band-limited ACGN channel with the goal of answering Question 2. In [21], it was shown that there are infinitely many noise psds whose capacity yields a non-computable number. In this work, we aim to characterize the classes of band-limited ACGN channels whose capacities yield a computable number. The following theorem provides a description of the structure of the noise psd that ensures the computability of the capacity.

Theorem 20. *If* N *is a strictly positive and computable continuous noise psd and* $P \in \mathbb{Q}$ *with* $P > \hat{P}$*, then the capacity* $C(P, N) \in \mathbb{R}_c$*.*

Proof. Let N be a computable continuous function, such that $\min_{f\in[0,B]} N(f) > 0$. From Eq.(6.1), we have that

$$C(P, N) = B \ln L - \int_0^B \ln(N(f))\, df. \tag{6.2}$$

The term $B \ln L$ is computable, since B is computable, L is computable and ln is a computable function. From the first statement of Lemma 7, we have that $\ln N(f)$ is also a computable continuous function. And hence, from Theorem 3, we have that $\int_0^B \ln(N(f))\, df \in \mathbb{R}_c$. □

While we have characterized the class of ACGN channels for which the capacity becomes computable, our focus now shifts to studying the computational complexity involved in computing the capacity. In other words, we aim to determine the level of complexity associated with computing the capacity of band-limited ACGN channels, given a noise spectral density with low complexity and a power constraint that can be computed in polynomial time.

Theorem 21. *Let* B *be a polynomial time computable number, and* N *be a strictly positive and polynomial time computable noise psd. Then the computation of the capacity* $C(P, N)$ *for the power constraint* $P \in \mathbb{Q}$*,* $P > \hat{P}$ *is in* $\#\mathrm{P}_1$*. Furthermore, there is an infinitely differentiable and strictly positive noise psd* N_* *and a power constraint* $P_* > \hat{P}_*$ *where* $\bar{N}_*$ *is the average noise psd, such that the computation of* $C(P_*, N_*)$ *cannot be polynomial time computable if* $\mathrm{FP}_1 \neq \#\mathrm{P}_1$*.*

Proof. Let N be such that it satisfies the conditions of Theorem 21. Then from Lemma 7 we have that $\ln N$ is polynomial time computable, since N is strictly a positive computable continuous function. Furthermore we have that $L > 0$. Since the computation of L is in $\#\mathrm{P}_1$ then from Lemma 7 we have that the computation of $\ln L$ is also in $\#\mathrm{P}_1$.

We start the proof by demonstrating the first statement. For $P > \hat{P}$, $P \in \mathbb{Q}$. From (6.1), we have that the capacity has the following form

$$C(P,N) = B \ln L - \int_0^B \ln N(f)\, df$$

We study the computation of each of the terms of the right hand side of (6.1). In the first term of (6.1), we have that $L > 0$ and since the computation of L is in $\#\mathrm{P}_1$, then the computation of $\ln L$ is also in $\#\mathrm{P}_1$. Now we consider the second term of (6.1). Let N be a noise psd satisfying the conditions of Theorem 21. Then we have that $\ln N$ is also polynomial time computable, since N is a strictly positive computable continuous function that is polynomial time computable. This implies that the computation of $C(P,N)$ is in $\#\mathrm{P}_1$. This way we have shown the first statement.

Now we prove the second statement. Let $\beta \in \mathbb{Q}$, $0 < \beta \leq 1$. We consider the following noise psd

$$N(f,\beta) = \beta N_*(f).$$

For $P > B\bar{N}_*$, $\beta_1, \beta_2 \in \mathbb{Q}$, $0 < \beta_l \leq 1$, $l = 1,2$ and $\beta_1 \neq \beta_2$, we have that the corresponding capacity is

$$C(P, N(\cdot,\beta_l)) = B \ln L_l - \int_0^B \ln N(f,\beta)\, df$$

with

$$L_l = \frac{P}{B} + \frac{1}{B}\int_0^B \ln N(f,\beta_l)\, df = \frac{P}{B} + \frac{\beta_l}{B}\int_0^B \ln N_*(f)\, df.$$

Assume that the computation of both numbers $C(P, N(\cdot,\beta_1))$ and $C(P, N(\cdot,\beta_2))$ are possible in polynomial time, then the computation of the number

$$\begin{aligned} C(P, N(\cdot,\mathrm{P}_1)) - C(P, N(\cdot,P_2)) & \\ = B \ln L_1 - B \ln L_2 - \beta_1 B + \beta_2 B & \\ = B \ln \frac{L_1}{L_2} - B(\beta_1 - \beta_2) & \end{aligned}$$

is polynomial time computable. Since $\beta_1, \beta_2 \in \mathbb{Q}$ we have that the number $B(\beta_1 - \beta_2)$ is polynomial time computable as well. This way we have that the number $z = B \ln \frac{L_1}{L_2}$ is polynomial time computable. Hence, $\frac{L_1}{L_2} = 2^{\frac{z}{B}} = c$ is polynomial time computable. We have that

$$c = \frac{L_1}{L_2} = \frac{P + \beta_1 \int_0^B \ln N_*(f)\, df}{P + \beta_2 \int_0^B \ln N_*(f)\, df},$$

and hence

$$z_* = \int_0^B \ln N_*(f)\, df = \frac{P(c-1)}{\beta_1 - c\beta_2},$$

where $\beta_1 \neq c\beta_2$ and $c \neq 1$. Since $P, \beta_1, \beta_2 \in \mathbb{Q}$ and c are polynomial time computable, then z_* must also be polynomial time computable. However, if $\mathrm{FP}_1 \neq \#\mathrm{P}_1$, then this

z_* cannot be polynomial time computable. This way we prove the second statement. That is, for $l = 1, 2$ at least one of the numbers $C(P, N(\cdot, \beta_l))$ is not polynomial time computable. □

Remark 27. *The proof of Theorem 21, like the proof of Theorem 18, consists of two parts and provides the exact characterization of the compelxity class of the computation of the number* $C(P_*, N_*)$.

Remark 28. *Considering Theorem 21, and assuming the widely accepted assumption* $\mathrm{FP}_1 \neq \#\mathrm{P}_1$ *is correct, we find that there exists a noise psd* N_* *and a corresponding power constraint* P_* *that can both be computed in polynomial time. Remarkably, under these conditions, the computation of the capacity of the band-limited ACGN channel demonstrates a complexity-blowup phenomenon.*

Remark 29. *Generally, the complexity problems that are studied aim to classify problems into the* P *or* NP *complexity classes. However, when it comes to computing the capacity-achieving psd, which is* $\#\mathrm{P}_1$*-complete, the task is typically more challenging compared to solving an* NP_1*-complete problem. Therefore computing the capacity-achieving psd and the capacity of the band-limited ACGN channels are harder problems than* NP_1.

Theorems 18 and 21 demonstrate, for the function N_*, that the corresponding capacity $C(P, N_*)$ and the value $P_x^*(P, 0)$ for $P > \hat{P}$, $P \in \mathbb{Q}$, cannot be computed in polynomial time. The computation of these numbers is even complete in $\#\mathrm{P}_1$. Regarding the dependence of this result on P, we observe a desired behavior, as this holds for all $P > \hat{P}$, $P \in \mathbb{Q}$.

It is now an interesting open question how often this behavior can occur in the set of admissible computable continuous noise psds. It should be mentioned at this point that, for many practically relevant computing scenarios, it is required that almost all problems in a problem class are not solvable in polynomial time. This is, for example, a central requirement for the applicability of today's cryptography, where parameter-dependent methods are used. Parameters are generally chosen at random when using cryptographic protocols, such as the two prime numbers in RSA. However, the cryptographic protocol should then not be computable with a very high probability, i.e., the corresponding computational problem must be difficult to solve for almost all parameters.

Whether such a behavior also occurs in the computation of the capacity of the ACGN channel is currently unclear. However, it is noteworthy that complexity classes $\#\mathrm{P}_1$ and #P play a crucial role in the capacity computation. For instance, computing the permanent of a matrix is known to be a #P-complete problem. When considering the permanent of random matrices from practically relevant number fields, it is established that for 'almost all' matrices, computing the permanent is a challenging task. Nevertheless, it remains uncertain whether such results also extend to the computation of the ACGN channel capacity.

In Theorem 21, the complexity of computing the number $C(P_*, N_*)$ for $P_* \in \mathbb{Q}$ with $P_* > \hat{P}_*$ is analyzed. Algorithms are considered that receive only the natural number

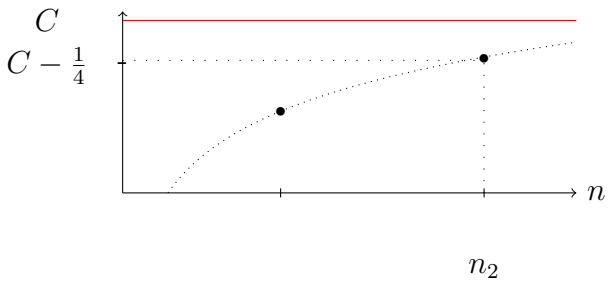

Figure 6.1: The red line represents the band-limited ACGN capacity $C = C(P, N_*)$ for the psd N_* and the power constraint P in the asymptotic regime. The black curve represents the finite blocklength achievable rate $R_n(\epsilon)$ for some fixed $\epsilon > 0$. For n_M we have $R_{n_M}(\epsilon) > C - \frac{1}{M}$.

M as input and compute the rational number α_M, such that $|C(P_*, N_*) - \alpha_M| < \frac{1}{2^M}$ holds. For practical implementations, it would be interesting to compute the capacity function $C(\cdot, N_*)$, i.e., to compute the function $C(P, N_*)$ with $P \in [0, \hat{P}]$ where $\hat{P} \in \mathbb{R}_c$ and $\hat{P} < \infty$. For this task, the algorithms should take $M \in \mathbb{N}$ as well as P as input. One can expect that the complexity of computing the capacity function $C(\cdot, N)$ is captured by the complexity class #P.

Theorem 21 has also interesting consequences for coding theory. Assume that a sequence, as a function of the blocklength, of capacity-achieving codes can be found such that the encoding and decoding processes corresponding to the blocklength have polynomial complexity, then it is possible to calculate the code's rate as a function of the blocklength in polynomial complexity. A more detailed analysis on this topic is given in the following subsection.

6.3 Implications for Finite Blocklength Performance

An important problem in coding theory and information theory is the analysis of the finite blocklength of capacity-achieving coding strategies.

In the finite blocklength regime, sequences of achievable rates and converses that are blocklength-dependent are derived while allowing a predefined decoding error. When considering a sequence of achievable rates $\{R_n\}_{n\in\mathbb{N}}$, the index n of each element R_n of the sequence indicates the blocklength of the code. Since we allow a decoding error $\epsilon > 0$ with $\epsilon \in \mathbb{Q}$, the achievable rates will be represented as a function of ϵ. Hence, the sequence of achievable rates blocklength-dependent is represented by $\{R_n(\epsilon)\}_{n\in\mathbb{N}}$ with $R_n(\epsilon) = \frac{1}{n} \log \mathcal{M}_n(\epsilon)$ for every $n \in \mathbb{N}$, where $\mathcal{M}_n(\epsilon)$ is the number of codewords, i.e., the size of the message set as a function of n and ϵ.

Now, from a practical point of view, it is interesting to determine, for $M \in \mathbb{N}$, where M describes the precision of the deviation of $C(P, N)$, for a certain blocklength n_M, when the following holds:

$$R_{n_M}(\epsilon) \geq C(P, N_*) - \frac{1}{2^M}. \tag{6.3}$$

This is visualized in Figure 6.1.

The complexity of computing the sequence $\{R_{n_M}(\epsilon)\}_{n\in\mathbb{N}}$ is addressed in the following corollary.

Corollary 4. *Let $\{R_{n_M}(\epsilon)\}_{M\in\mathbb{N}}$ be an arbitrary sequence of achievable rates for some $\epsilon > 0$ with $\epsilon \in \mathbb{Q}$, such that for all $M \in \mathbb{N}$ the following holds*

$$R_{n_M}(\epsilon) > C(P, N_*) - \frac{1}{2^M}. \tag{6.4}$$

Then, the sequence $\{R_{n_M}(\epsilon)\}_{M\in\mathbb{N}}$ cannot be computed in polynomial time.

Proof. Let us assume that there exists a sequences of blocklengths $\{n_M\}_{M\in\mathbb{N}}$, such that $\{R_{n_M}(\epsilon)\}_{M\in\mathbb{N}}$ can be computed in polynomial time. Let $P\colon \mathbb{N}\times\mathbb{N}\to\mathbb{N}$ be the corresponding polynomial from Definition 28. Then for $x \in \mathbb{N}$, the function $P_1(x) \coloneqq P(x+1, x+1)$ is also a polynomial. The numbers $\alpha_{n,M}$ with $n \in \mathbb{N}$ and $M \in \mathbb{N}$ are calculated in maximum $P(n, M)$ steps and the following holds

$$|R_{n_M}(\epsilon) - \alpha_{n,M}| < \frac{1}{2^M}.$$

We consider now the computable sequence $\{\tilde{\alpha}_M\}_{M\in\mathbb{N}}$ with $\tilde{\alpha}_M \coloneqq \alpha_{M+1,M+1}$ for $M \in \mathbb{N}$. The rational number $\tilde{\alpha}_M$ is computed in maximum $P_1(M)$ steps. Hence, $\{\tilde{\alpha}_M\}_{M\in\mathbb{N}}$ is a polynomial time computable sequence of rational numbers.

For $M \in \mathbb{N}$ arbitrary, we have

$$\begin{aligned}
|C(P, N_*) - \tilde{\alpha}_M| &= |C(P, N_*) - R_{n_{M+1}}(\epsilon) \\
&\quad + R_{n_{M+1}}(\epsilon) - \tilde{\alpha}_M| \\
&\leq |C(P, N_*) - R_{n_{M+1}}(\epsilon)| \\
&\quad + |R_{n_{M+1}}(\epsilon) - \tilde{\alpha}_M| \\
&\leq \frac{1}{2^{M+1}} + \frac{1}{2^{M+1}} = \frac{1}{2^M}.
\end{aligned}$$

In this manner, the computation of $C(P, N_*)$ would be polynomial-time feasible, leading to a contradiction to the Theorem 21. □

Remark 30. *We observe that $R_{n_M}(\epsilon) = \frac{1}{n}\log_2 \mathcal{M}_{n_M}(\epsilon)$ is, for a fixed M, a polynomial-time computable number. Consequently, $\{R_{n_M}(\epsilon)\}_{M\in\mathbb{N}}$ forms a computable sequence of polynomially time computable numbers. However, the sequence itself is not computable in polynomial time. In other words, for every M, we can find a polynomial P_M such that, for every $N \in \mathbb{N}$, there exists a rational number $\tilde{\alpha}^*_{M,N}$ computed in at most $P_M(N)$ steps, satisfying*

$$|R_{n_M}(\epsilon) - \tilde{\alpha}^*_{M,N}| \leq \frac{1}{2^N}.$$

However, Corollary 4 shows that the sequence of achievable rates $\{R_{n_M}(\epsilon)\}_{M\in\mathbb{N}}$ cannot be computed in polynomial time as a sequence. The parameters of the polynomial P_M, which depend on M, must grow faster than any polynomial.

Remark 31. *This result applies also for any capacity achieving coding scheme. Therefore, if the sequence $\{R_n(\epsilon)\}_{n\in\mathbb{N}}$ is a computable sequence of computable numbers, one of the following statements must always be valid:*

- *The sequence $\{R_n(\epsilon)\}_{n\in\mathbb{N}}$ is not a polynomial time computable sequence.*
- *The sequence of blocklegths $\{n_M\}_{M\in\mathbb{N}}$, for which the following holds*

$$R_{n_M}(\epsilon) > C(P, N_*) - \frac{1}{2^M},$$

is not a polynomial time computable sequence of natural numbers.

6.4 Conclusions

While the capacity of band-limited ACGN channels is generally not computable, we have successfully characterized the subset of these channels that do have computable capacities. Our findings demonstrate that as long as the continuous noise spectral densities are strictly positive and computable, the resulting capacity will always be a computable number.

Furthermore, we have studied the computational complexity involved in determining the capacity of such channels. Our analysis reveals that calculating the capacity of polynomial time computable continuous ACGN channels not only falls within the $\#\mathrm{P}_1$ class. Additionally, we have shown that if the widely accepted assumption $\mathrm{FP}_1 \neq \#\mathrm{P}_1$ holds true, then it is impossible to compute the capacity of band-limited ACGN channels in polynomial time.

Moreover, we have explored the computational complexity of determining the capacity-achieving psd of band-limited ACGN channels. Our analysis examines the relationship between the complexity of computing the capacity-achieving psd and the computational complexity of the power constraint and the ACGN channel itself. Our results demonstrate that when considering polynomial time computable parameters, such as the power constraint, bandwidth, and noise psd, the computation of the capacity-achieving psd becomes $\#\mathrm{P}_1$-complete.

This finding implies that if the assumption $\mathrm{FP}_1 \neq \#\mathrm{P}_1$ holds true, there can exist a noise psd that is computable in polynomial time, while the capacity-achieving psd associated with it cannot be calculated within polynomial time. This demonstrates a complexity blowup behavior, illustrating the increased difficulty of determining the capacity-achieving psd in such scenarios.

This has also interesting consequences for coding theory, especially in the finite blocklength regime. Assume that a sequence of capacity-achieving codes has polynomial complexity; this does not immediately imply that the achievable rate sequence can be computed in polynomial time. An achievable rate itself can be computed in polynomial time as a number. However, we have shown, that either the sequence of achievable rates as a function of the blocklength is not a polynomial time computable sequence, or the

sequence of blocklength corresponding to the achievable rates with guaranteed distance to capacity is not a polynomial time computable sequence of natural numbers.

The focus of studying the complexity of problems in information and communication theory lies in determining whether a computational problem belongs to the classes P or NP (or P_1 and NP_1, respectively). In this paper, we show that both the computation of the capacity of band-limited ACGN channels and the computation of the capacity-achieving psd fall within the class $\#\mathrm{P}_1$. In NP, one can efficiently verify a single certificate for a problem. However, NP_1 does not provide information about the number of certificates. $\#\mathrm{P}_1$ counts the certificates that can be efficiently verified, making it a more general and complex class than NP_1. And hence problems classified as $\#\mathrm{P}_1$-complete are harder than problems classified as NP_1-complete, under the common complexity assumptions.

7 Computability of Convex Optimization Problems

In Chapter 4 and Chapter 5, we have shown that both the capacity of the FSC with feedback and the capacity of the band-limited ACGN channel are non-computable functions. On the one hand, the capacity of the FSC with feedback has a multi-letter expression, indicating that it is the limit of a sequence of optimization problems. On the other hand, the capacity of the band-limited ACGN channel is represented by a Riemann integral. A Riemann integral represents the area under the curve of a function, which is determined by the limit of the sum of function values multiplied by infinitesimal widths of intervals. Hence, both capacity representations show a characteristic that can become challenging for digital computers to compute. In this chapter, our focus shifts to a more simpler setting: convex optimization problems.

Convex optimization is a crucial field within mathematical optimization. It focuses on minimizing convex functions (or, equivalently, maximizing concave functions) over convex sets arising from equality and inequality constraints imposed on the problem domain. One fundamental property of convex functions is that any local minimum is also a global minimum. Due to these characteristics, many convex optimization problems have efficient algorithms to find their global optima. Moreover, many optimization problems that may not initially appear to be convex can be reformulated as convex problems. In [29], there is a detailed exploration of techniques to recognize and harness the convex properties of a wide array of optimization problems.

Several techniques have been developed to solve convex optimization problems, particularly for those with a feasible set defined by inequality constraints. Examples of these techniques include interior-point methods [79, 80], cutting-plane methods [81], and sub-gradient methods [82], among others. Interior-point methods, for instance, can efficiently solve a class of convex optimization problems, such as linear programming. It was shown in [83] that linear programming can be solved in polynomial time by an interior-point method algorithm. Whether all constrained convex optimization problems can be efficiently solved is still an open question.

These methods are designed to be used on digital computers. Digital hardware inherently deals with discrete quantities, and this limitation poses a significant challenge when solving problems in a continuous scenario. To understand whether there is a discrepancy between discrete and continuous models, one should study how the digital solution approximates the solution of the continuous problem. In other words, it involves investigating whether algorithms can be developed to compute the continuous model's solution using digital hardware. Here, we address the question of whether, in general, all constrained convex optimization problems can be solved using digital computers.

7.1 Problem Formulation

In the forthcoming section, our focus lies on addressing the algorithmic computability of convex optimization problems.

In general, optimization problems are often complex and involve searching through a vast solution space to find the best or optimal solution. Frequently, in practical systems, the feasible sets of these optimization problems are implicitly defined by inequality constraints, which typically take the following forms:

- $\varphi(x) \geq \lambda$: This arises for instance, in scenarios where certain values, not too small, are permissible, as seen in power allocation for wireless systems. In this case, the transmission power cannot be set too low, as the amplifier operates effectively only within specific ranges.

- $\varphi(x) \leq \lambda$: Conversely, this arises, again for example in power allocation in wireless systems, where excessively high transmission powers are disallowed due to potential non-linear effects that could introduce power amplification outside the designated power band.

These constraints play a pivotal role in shaping the optimization process, ensuring that solutions are not only optimal but also viable within the given practical considerations.

Several numerical algorithms have been developed with the goal of solving these optimization problems. In particular, a large number of algorithms have been specifically designed to tackle convex optimization problems, which have the advantage of possessing a unique global minimum. Gradient descent, interior point methods, linear programming, dual decomposition, among others, are some of the algorithms commonly used for convex optimization problems; see, e.g., [29].

Thanks to the well-established numerical methods for solving convex optimization problems, there exists a belief that any convex optimization problem can be algorithmically solved. Convex optimization problems with computable, continuous, and convex objective functions, as well as computable intervals defining their feasible sets, are computable (see [33]). We aim to investigate whether a convex problem is computable when its feasible set is defined by a computable convex constraint function. Consequently, we are also interested in exploring whether the optimal point of such a convex problem is computable.

Next, we characterize a convex optimization problem by examining its primary components and pose the question of whether algorithms exist that can solve any convex optimization problem by utilizing its fundamental components as input.

Suppose $m \in \mathbb{N}$ and $\mathcal{B} = [0, b]^m$ with $b \in \mathbb{N}$. Let $f \colon \mathcal{B} \to \mathbb{R}$ be a strictly convex function and $\varphi \colon \mathcal{B} \to \mathbb{R}$ be a convex function. Let $\lambda \in \mathbb{R}_c$. We consider a convex optimization problem of the following form:

$$\begin{aligned} \text{minimize} \quad & f(x) \\ \text{subject to} \quad & \varphi(x) \leq \lambda \end{aligned} \tag{7.1}$$

We consider only those λ for which $\mathcal{M}_\lambda(\varphi) \neq \emptyset$ where

$$\mathcal{M}_\lambda(\varphi) = \{x \in \mathcal{B} \colon \varphi(x) \leq \lambda\}.$$

It then holds that $\mathcal{M}_\lambda(\varphi)$ is a closed convex set in $\mathcal{B}$. Specifically, we consider the convex functions f for which the following condition holds:

Condition 1. *Let $x^{(1)}, x^{(2)} \in \mathcal{B} \subset \mathbb{R}^n$ be arbitrary. If $x_k^{(1)} \leq x_k^{(2)}$ for every $1 \leq k \leq n$, element-wise, then it also holds that*

$$f(x^{(1)}) \leq f(x^{(2)}).$$

Convex functions for which Condition 1 is satisfied, is convex and pointwise monotonically increasing and has a unique optimal value. Furthermore, if the objective function is strictly convex and satisfies Condition 1, then the optimal point is unique.

An optimal value of a convex problem can be characterized as a function of the objective function f, constraint function φ, and its constraint value λ, i.e.,

$$\mathrm{OptV}(f, \varphi, \lambda) = \min_{x \in \mathcal{M}_\lambda(\varphi)} f(x). \tag{7.2}$$

Note that computing the optimal value OptV is a standard convex optimization task. In studying computabilty of this problem, we will consider only objective functions f that are strictly convex and computable continuous, constraint functions φ that are convex and computable continuous, and computable constraint values $\lambda \in \mathbb{R}_c$.

The optimal values of minimization or maximization problems are computable when the feasible sets consist of simple intervals, as demonstrated in [33]. We are now interested in feasible sets modeled by constraint inequalities ($\varphi(x) \geq \lambda$ or $\varphi(x) \leq \lambda$), which are highly relevant in practical scenarios. This prompts the following question:

Question 1: *Let f be an objective function, φ be a constraint function and λ be a constraint value of an optimization problem such that $\mathcal{M}_\lambda(\varphi) \neq \emptyset$. Is it possible to find an algorithm, specifically a Turing machine, that takes the parameters f, φ, and, λ as input and computes a description of the real number $OptV(f, \varphi, \lambda)$?*

In the convex optimization literature, there is often a tacit assumption that such a Turing machine exists, i.e., a universal algorithm capable of taking parameters from any convex optimization problem and computing its optimal value.

A less ambitious task would be to find a problem-specific algorithm for each individual convex optimization problem to compute its optimal value. This prompts the following question:

Question 2: *Let f be an objective function, φ be a constraint function and λ be a constraint value of an optimization problem such that $\mathcal{M}_\lambda(\varphi) \neq \emptyset$. Is it possible to find an algorithm that takes a precision $M \in \mathbb{N}$ as input and computes the rational number y_M, such that*

$$|OptV(f, \varphi, \lambda) - y_M| \leq \frac{1}{2^M}? \tag{7.3}$$

The optimal value in our context may be either rational or irrational. However, due to the limitations of Turing machines, which can only compute rational numbers, we

aim to find a Turing machine that can generate a rational approximation of the optimal value for a given problem. To assess the quality of such an approximation, it is crucial to set a precision threshold. In simpler terms, we consider an approximation of the optimal value as 'good enough' when it falls within the chosen level of precision.

In the context of Question 6, the objective is to develop an algorithm tailored for solving a particular convex optimization problem. This algorithm takes a desired precision parameter, denoted as M, as input and computes an approximation, denoted as y_M, for the optimal value. The requirement is that y_M should be accurate within a maximum deviation of 2^{-M} from the true optimal value.

Remark 32. *Note that there is no difference in using a different scale than 2^{-M} for the approximation error in (7.3).*

Remark 33. *If φ and λ are chosen, such that $\mathcal{M}_\lambda(\varphi) = \mathcal{B}$, then we have that $OptV(f, \varphi, \lambda)$ is a computable number, as defined in the following section.*

7.2 Convex Optimization Problems on Turing Machines

In this section, we delve into the algorithmic properties of convex optimization problems, with a particular emphasis on their solvability on Turing machines. This study is crucial for understanding the computational boundaries and capabilities associated with these optimization challenges.

A key aspect of our investigation is the computability of solutions to convex optimization problems. To address this, we introduce a foundational lemma that clarifies the relationship between the computability of a function's output and the computability of the corresponding input for a given domain.

Lemma 12. *Let $b \in \mathbb{N}$ and f be a strict monotonically increasing computable continuous function on $[0, b]$. If $f(x) \in \mathbb{R}_c$ for some $x \in [0, b]$, then $x \in \mathbb{R}_c$.*

Proof. We prove this result by contradiction by assuming that there is a $x_* \in [0, b]$ with $x_* \notin \mathbb{R}_c$, such that $f(x_*) \in \mathbb{R}_c$. We then generate a contradiction.

For $n \in \mathbb{N}$, we consider the lattice points $x_{k,n} = \frac{k}{2^n}$ for $0 \leq k \leq b2^n$. We use two Turing machines $TM_{>\lambda_*}$ and $TM_{<\lambda_*}$ for $\lambda_* = f(x_*)$, $\lambda_* \in \mathbb{R}_c$.

$TM_{<\lambda_*}$ receives as input a representation for $z \in \mathbb{R}_c$ and either stops the computation or calculates forever. The machine stops if and only if $z < \lambda_*$.

$TM_{<\lambda_*}$ has the same property. It receives as input a representation for $z \in \mathbb{R}_c$ and either stops the computation or calculates forever. The machine stops if and only if $z > \lambda_*$. The existence of such a machine is shown in [33].

We use both machines to compute $f(x_{*,n})$. $TM_{<\lambda_*}(f(x_{*,n}))$ stops if and only if $f(x_{*,n}) < \lambda_*$. This holds if and only if $x_{*,n} < x_*$. The machine can never stop for $x_{*,n} = x_*$, since $x_* \notin \mathbb{R}_c$. Similarly, $TM_{>\lambda_*}(f(x_{*,n}))$ stops if and only if $f(x_{*,n}) > \lambda_*$. This holds if and only if $x_{*,n} > x_*$.

Since f is a strictly monotonically increasing function, one of the Turing machines stops for $x_{*,n}$. We can generate two computable sequences $\{\underline{u}_n\}_{n\in\mathbb{N}}$ and $\{\bar{u}_n\}_{n\in\mathbb{N}}$, such that $\underline{u}_n \leq \underline{u}_{n+1}$, $\bar{u}_n \geq \bar{u}_{n+1}$ for $n \in \mathbb{N}$ and

$$\lim_{n\to\infty} \underline{u}_n = x_* = \lim_{n\to\infty} \bar{u}_n.$$

Then x_* must be a computable number, with which we have created a contradiction.

We proceed as follows: For $n = 1$, we start both Turing machines $TM_{<\lambda_*}$ and $TM_{>\lambda_*}$ in parallel for $f(x_{k,n})$, $0 \leq k \leq 2b$, for one calculation step each. If none of them stops , then we set $\underline{u}_1 = 0$ and $\bar{u}_1 = b$. If one of them stops, then we set $\underline{u}_1 = \max x_{k,1}$ if $TM_{<\lambda_*}$ stopped, and $\bar{u}_1 = \in x_{k,1}$ if $TM_{>\lambda_*}$ stopped.

For $n = 2$, we compute the second step for all $x_{k,1}$, for which the computation of $TM_{<\lambda_*}$ or of $TM_{>\lambda_*}$ did not stop. Further, we compute for $x_{k,2}$, $0 \leq k \leq 4b$ for $TM_{<\lambda_*}$ and $TM_{>\lambda_*}$ each one step. We then compute $\underline{u}_2 = \max \underline{x}_2$. $\underline{x}_2$ is in the set of $x_{k,2}$ rsp. $x_{k,1}$, $0 \leq k \leq 4b$, for which $TM_{<\lambda_*} f(\underline{x}_2)$ has stopped. Similarly, $\bar{u}_2$ is computed.

This way we get two computable sequences of rational number $\{\underline{u}_n\}_{n\in\mathbb{N}}$ and $\{\bar{u}_n\}$, such that $\underline{u}_n \leq \underline{u}_{n+1}$ and $\bar{u}_n \geq \bar{u}_{n+1}$ where $n \in \mathbb{N}$. Furthermore, for $x_{k,n} < x_*$ the $TM_{<\lambda_*} f(x_{k,n})$ must stop at some point. With this, there is an index $m \in \mathbb{N}$, such that $\underline{u}_m > x_{k,m}$. This $\lim_{n\to\infty} \underline{u}_n = x_*$. Following the same line of arguements, we have that $\lim_{n\to\infty} \bar{u}_n = x_*$ which then implies that $x_* \in \mathbb{R}_c$. Thus, we have generated a contradiction. □

We now present the central theorem of this chapter, which addresses the computability of optimal values within convex optimization problems. It demonstrates that for certain computable convex constraint sets, the optimal values of all computable strictly convex objective functions are non-computable.

Theorem 22. *Let $m \in \mathbb{N}$, $m \geq 1$ be arbitrary. Then there exists a computable continuous convex function φ_m and a number λ_m with $\mathcal{M}_{\lambda_m}(\varphi_m) \neq \emptyset$, such that for all $f\colon \mathcal{B} \to \mathbb{R}$ that are strictly convex, computable continuous, and satisfy condition A, the following holds:*

$$OptV(f, \varphi, \lambda) \notin \mathbb{R}_c. \tag{7.4}$$

Proof. We start with $m = 1$. Let x_* be arbitrary with $x_* \notin \mathbb{R}_c$. Let $\{\mu_n\}_{n\in\mathbb{N}}$ be a computable seqeunce of rational numbers with $\mu_n < \mu_{n+1}$ for $n \in \mathbb{N}$ and

$$\lim_{n\to\infty} \mu_n = x_*, \quad \mu_0 > 0.$$

Let $b > \mu_1$ be an arbitrary computable number. We set

$$g_n(x) = \begin{cases} \frac{x}{\mu_n} & \text{for } 0 \leq x \leq \mu_n \\ 1 & \text{for } \mu_n < x \leq b. \end{cases}$$

It holds that $g_n(x) = \min \frac{x}{\mu_n}$ for $x \in [0, b]$. This way we have that g_n is a computable continuous function. g_n is furthermore a concave function. We now consider the following function:

$$g_*(x) = \sum_{n=1}^{\infty} \frac{1}{2^{n+1}} g_n(x) \quad x \in [0, b].$$

For $M \in \mathbb{N}$ arbitrary, we have

$$|g_*(x) - \sum_{n=1}^{M} \frac{1}{2^{n+1}} g_n(x)| \leq \sum_{n=M+1}^{\infty} \frac{1}{2^{n+1}} = \frac{1}{2^{M+2}} \sum_{\ell=0}^{\infty} \frac{1}{2^\ell}$$
$$= \frac{1}{2^{M+2}} \frac{1}{1-\frac{1}{2}} = \frac{1}{2^{M+1}} < \frac{1}{2^M}.$$

Thus, we have that the computable sequence $\{\sum_{n=1}^{M} \frac{1}{2^{n+1}} g_n(\cdot)\}_{M\in\mathbb{N}}$ converges effectively to the function g_*. And this again implies that g_* is a computable continuous function. Let x_1, x_2 be two arbitrary real numbers with $0 \leq x_1 \leq x_2 \leq x_*$. Since $g_n(x) > 0$ for $x \in [0, b]$ and $n \in \mathbb{N}$ and since for all n with $\mu_n > x_*$ it holds that $g_n(x_1) < g_n(x_2)$, we have that $g_*(x_1) < g_*(x_2)$.

For all $x \in [0, x_*]$ we have that $g_n(x) = 1$ for $n \in \mathbb{N}$ and hence

$$g_*(x) = \sum_{n=1}^{\infty} \frac{1}{2^{n+1}} = \frac{1}{2}.$$

Therefore, $\max_{x\in[0,b]} g_*(x) = \frac{1}{2}$ and for all $x \in (x_*, b]$ we have $g_*(x) < \frac{1}{2}$.

Next, we consider the following function:

$$\varphi_*(x) = 2(1 - g_*(x)). \tag{7.5}$$

φ_* is a computable continuous convex function. It holds that $\varphi_*(x) > 1$ for all $x \in [0, x_*)$ $\varphi_*(x) = 1$ for $x \in [x_*, b]$.

For $\lambda_1 = 1$ we have

$$\mathcal{M}_{\lambda_1}(\varphi_*) = \{x \in [0, b] \colon \varphi_*(x) \leq 1\}$$

and hence $\mathcal{M}_{\lambda_1}(\varphi_*) = [x_*, b]$.

Let f be a strictly convex monotonically increasing computable continuous function. Since $x_* \in \Sigma_1$, $x_* \notin \mathbb{R}_c$, we have that $f(x_*) \notin \mathbb{R}_c$ from Lemma 12. Thus, we have proven Theorem 22 for $m = 1$.

Let $m \geq 2$ arbitrary. Let $f \colon [0, b]^m \to \mathbb{R}$ an arbitrary computable continuous strictly convex function for which condition A holds. Then it holds that

$$\min_{x\in\mathcal{M}_{\lambda_1}(\varphi_*)} = f(x_*)$$

and hence, from Lemma 12 we have that $\mathrm{OptV}(f, \varphi_*, \lambda_*) \notin \mathbb{R}_c$. This way we prove Theorem 22. □

Corollary 5. *For each $m \in \mathbb{N}$, there exists a computable continuous convex function φ_m and a number λ_m with $\mathcal{M}_{\lambda_m}(\varphi_m) \neq \emptyset$, such that for all computable continuous and strictly convex functions f Question 2 has a negative answer.*

Proof. For every function f, for which Theorem 22 holds, we have that $\mathrm{OptV}(f, \varphi, \lambda) \notin \mathbb{R}_c$. And hence, y_M cannot be generated such that Eq. (7.3) holds. □

Remark 34. *According to Theorem 22, we observe that, regardless of the simplicity of the objective function, i.e., whether it is computable and strictly convex, the use of convex inequalities to describe the feasible set of a constrained optimization problem results in a non-computable optimal value for every possible strictly convex objective function. In other words, when dealing with a specific computable convex problem, the existence of an algorithm that approximates the optimal value up to any preselected precision becomes impossible.*

Consequently, this implies a negative response to Question 2. Furthermore, it suggests the impossibility of designing a universal algorithm capable of taking core input parameters of a convex optimization problem—namely, the objective function, constraint inequality function, and constraint values—and computing the optimal value to any desired precision. This negative result provides an answer to Question 1.

Remark 35. *This shows that when solving algorithmically convex optimization problems on digital computers, both the computation of the optimal value of a function and the computation of the optimal point can never be algorithmically transparent and, therefore, can never fulfill the integrity condition.*

Remark 36. *Note that the results of Theorem 22 and Corollary 5 also hold for a convex maximization problem, where the objective function is a strictly concave function, and the feasible set is defined by the same constraint functions.*

7.3 Lagrangian Dual Problem on Turing Machines

We consider the following optimization problem:

$$\begin{aligned} &\text{minimize} && f(x) \\ &\text{subject to} && \varphi(x) \leq \lambda \end{aligned}$$

where $f\colon \mathcal{D} \to \mathbb{R}$ is a computable continuous function and $\varphi\colon \mathcal{D} \to \mathbb{R}$ is a computable continuous convex function with domain $\mathcal{D} = [a,b]^m$, where $m \in \mathbb{N}$ and $a, b \in \mathbb{R}_c$. The constraint function φ determines the feasible set $\mathcal{M}_\lambda = \{x \in \mathcal{D}\colon \varphi(x) \leq \lambda\}$. Hence, the goal is to solve the following problem:

$$\min_{x \in \mathcal{M}} f(x). \tag{7.6}$$

We consider the Lagrangian function $\mathcal{L}\colon \mathcal{D} \times \mathbb{R} \to \mathbb{R}$ associated with the problem (7.6):

$$\mathcal{L}(x,u) = f(x) + u\varphi(x).$$

We also consider the Lagrangian dual function associated with (7.6):

$$g(u) = \min_{x \in \mathcal{D}} \mathcal{L}(x,u). \tag{7.7}$$

Note that since $\mathcal{L}$ is a computable continuous function in both variables x and u, then, in particular, $\mathcal{L}$ is continuous in x for a fixed u. Since $\mathcal{D}$ is a compact set, then (7.7) is actually a minimum and for every $u \geq 0$ there is a $x(u) \in \mathcal{D}$ with $g(u) = \mathcal{L}(x(u), u)$.

Since $u \geq 0$ is not a compact set, we consider the problem

$$C_{\mathcal{D}} = \sup_{u \geq 0} g(u) = C_{\mathcal{D}}(f, \varphi, \lambda)$$

We would like to know whether for f computable continuous and φ computable continuous and convex, the number $C_{\mathcal{D}}$ can be computed by a Turing machine or not.

Theorem 23. *There exists a computable continuous convex function $\varphi\colon \mathcal{D} \to \mathbb{R}$, and a number $\lambda \in \mathbb{R}_c$ such that for all $f\colon \mathcal{D} \to \mathbb{R}$ that are strictly convex, computable continuous and satisfy condition A, the following holds:*

$$C_{\mathcal{D}}(f, \varphi, \lambda) = \sup_{u \geq 0} g(u) \notin \mathbb{R}_c$$

and for all $K \in \mathbb{N}$

$$\max_{u \in [0,K]} g(u) < C_{\mathcal{D}}(f, \varphi, \lambda).$$

And, thus, we have

$$\lim_{K \to \infty} \left(\max_{u \in [0,K]} g(u) \right) = C_{\mathcal{D}}(f, \varphi, \lambda) \tag{7.8}$$

Proof. We prove the result for $m = 1$. For $m > 1$ the proof follows the same steps.

Let f, φ, λ be such that Theorem 22 holds.

For $u \geq 0$, we consider

$$\mathcal{L}(x, u) = f(x) + u(\varphi(x) - 1).$$

We observe that $\mathcal{L}(x, u) \geq 0$ for $x \in [0, b]$ and $u \geq 0$. Since f is strictly convex and $\varphi(x) - 1$ is convex, it follows that $\mathcal{L}(x, u)$ is strictly convex with respect to x. Additionally, because $\varphi(x) - 1 \geq 0$ for $x \in [0, b]$, $\mathcal{L}(x, u)$ is monotonically increasing with respect to u. With this we have that

$$g(u) = \min_{x \in [0,b]} \mathcal{L}(x, u)$$

is monotonically increasing and concave for $u \geq 0$.

Furthermore, it holds that

$$\min_{x \in [0,b]} \mathcal{L}(x, u) \leq \mathcal{L}(x_*, u) = \mathrm{OptV}(f, \varphi, \lambda)$$

and also $g(u) \leq \mathrm{OptV}(f, \varphi, \lambda)$ for $u \geq 0$.

Since $\mathcal{L}(x, u)$ is strictly convex for $u > 0$, there is exactly one $\hat{x} = \hat{x}(u)$ with $\hat{x} \in [0, b]$ and

$$g(u) = \min_{x \in [0,b]} \mathcal{L}(x, u) = \mathcal{L}(\hat{x}(u), u). \tag{7.9}$$

It also holds that $\hat{x} \in (0, b)$. Furthermore, from Lemma 2 we have that for $u_1 \neq u_2$ where $u_1, u_2 \in (0, u)$ it holds that

$$\hat{x}(u_1) \neq \hat{x}(u_2).$$

Hence, for $0 < u_1 < u_2 < +\infty$,$u(\mu) = (1-\mu)u_1 + \mu u_2$ and $\mu \in [0,1]$ arbitrary we have

$$\begin{aligned} g(u(\mu)) &= \min_{x\in[0,b]} \mathcal{L}(x, u(\mu)) \\ &= \min_{x\in[0,b]} ((1-\mu)\mathcal{L}(x,u_1) + \mu\mathcal{L}(x,u_2)) \\ &> (1-\mu)\mathcal{L}(\hat{x}(\mu),u_1) + \mu\mathcal{L}(\hat{x}(\mu),u_2) \\ &> (1-\mu)\mathcal{L}(\hat{x}_1,u_1) + \mu\mathcal{L}(\hat{x}_2,u_2) \\ &= (1-\mu)g(u_1) + \mu g(u_2). \end{aligned}$$

This shows that g is strictly concave and monotonically increasing. Furthermore we have that $g(u) \leq \mathrm{OptV}(f,\varphi,\lambda)$ and thus there exists a limit value

$$C_{\mathcal{D}} = \sup_{u\geq 0} g(u) \leq \mathrm{OptV}(f,\varphi,\lambda).$$

However, from Lemma 3, we have that $C_{\mathcal{D}} = \mathrm{OptV}(f,\varphi,\lambda)$. Using Theorem 22, we get that $C_{\mathcal{D}} = \mathrm{OptV}(f,\varphi,\lambda) \notin \mathbb{R}_c$. □

The following corollary follows from Theorem 23.

Corollary 6. *Let f,φ,λ be fixed and for which Theorem 23 holds. There exists no algorithm that receives as input $N \in \mathbb{N}$ and computes a $K(N) \in \mathbb{N}$ such that*

$$|C_{\mathcal{D}}(f,\varphi,\lambda) - \max_{u\in[0,K(N)]} g(u)| < 2^{-N}. \tag{7.10}$$

Proof. The proof follows directly from Theorem 23, since $C_{\mathcal{D}}(f,\varphi,\lambda) \notin \mathbb{R}_c$ and hence $C_{\mathcal{D}}(f,\varphi,\lambda)$ cannot be effectively approximated. □

Remark 37. *We observe that the sequence $\{\max_{u\in[0,K(N)]} g(u)\}_{N\in\mathbb{N}}$ converges to $C_{\mathcal{D}}(f,\varphi,\lambda)$. However, this convergence is not effective. Consequently, it is not possible to algorithmically control the approximation error $|C_{\mathcal{D}}(f,\varphi,\lambda) - \max_{u\in[0,K(N)]} g(u)|$. In other words, one cannot exploit the convergence of (7.8) to determine $K(N)$ for a given error of 2^{-N} in such a way that the approximation error is respected.*

Theorem 24. *Let $y_* \in \Sigma_1$, $y_* > 0$. Let $b > y_*$. Then thre exists a computable continously differentiable function φ_* and a computable number λ_*, such that φ_* is strictly monotonically decreasing and convex on $[0,y_*]$ and for $x \in [y_*,b]$ we have $\varphi_*(x) = \lambda_*$. Furthermore, for*

$$\mathcal{L}_1(x,u) = x - u(\varphi_*(x) - \lambda_*)$$

it holds that

$$\sup_{u\geq 0} g_1(u) = y_*.$$

Proof. An analysis of the proof of Theorem 23 shows that if φ_* is strictly convex on $[0,y_*]$, then the function f does not necessarily need to be strictly convex. Hence, the convexity property of f is sufficient to perform all the proof steps also for the Lagrangian function $x + u(\varphi_*(x) - \lambda_*)$. □

7.4 Conclusions

In this chapter, we have studied the algorithmic properties of convex optimization problems. We have considered minimization problems with strictly convex, continuous, and computable objective functions and with inequality constraints that are convex and also computable. We have demonstrated the existence of a computable continuous, and convex constraint function such that the optimal value is a non-computable number for any computable continuous objective function that satisfies the strict convexity property. Consequently, for this specific convex optimization problem, it is impossible to design an algorithm that takes a pre-specified precision as input and computes an approximation of the problem's optimal value within the desired precision. Thus, constrained convex optimization problems cannot always be algorithmically solved. This result immediately implies that constructing a universal algorithm that takes an arbitrary optimization problem with objective and constraint functions as parameters and computes the problem's optimal value is impossible.

Furthermore, we have demonstrated that, although a converging sequence of computable numbers exists for the solution of the Lagrangian dual problem in optimization problems subject to such constraint functions, it is impossible to compute the distance of each element in the sequence from the solution. Consequently, there is no algorithm that, given such a constrained convex optimization problem, can compute its optimal solution up to any desired precision. Hence, there are constraint functions for which the Lagrangian dual problem cannot be algorithmically solved.

8 Conclusions

In this work, we have explored the computational and continuity properties of the capacity function of various communication models and convex optimization problems. Here, we summarize the key results of this work.

In Chapter 3, we have demonstrated the robustness of the compound BCC model, showing that minor changes in the uncertainty set have minimal impact on the capacity region. However, when expanding to the next level of uncertainty model, namely arbitrarily varying BCC model, which incorporates symmetrizability challenges, we already observed notable discontinuities in the capacity region.

In Chapter 4, we have shown that the feedback capacity of FSCs is not Banach-Mazur computable, which also implies that it is not Borel-Turing computable. This inherently means that devising a universal algorithm to approximate the capacity based on channel parameters at any desired precision is unfeasible. This complicates establishing sharp bounds for code performance evaluation, suggesting achievability, converse, or both may be non-computable, thus challenging code efficacy assessment.

In Chapter 5, we have shown that, even for a rather simple channel model, computing it capacity can become very challenging. In particular, we examine the algorithmic properties of band-limited ACGN channel capacities, revealing that there are computable noise psd whose capacities yield a non-computable number; this again implies the impossibility of finding a universal algorithm that takes the describing parameters of an ACGN channel and computes the capacity at any desired precision. Fortunately, one can algorithmically construct a sequence of achievable rates that converge to capacity. However, it is impossible to compute how far they are from capacity. Consequently, it is impossible to algorithmically compute an upper bound on the size of the codebook for the channel.

In Chapter 6, we have successfully identified the conditions for computable capacities in ACGN channels, showing that as long as the continuous noise spectral densities are strictly positive and computable, the resulting capacity will always be a computable number. Additionally, we have shown that computing the ACGN channel capacity lies within the $\#\mathrm{P}_1$ complexity class. Moreover, we have shown that the computation of the capacity-achieving psd becomes $\#\mathrm{P}_1$-complete. Under the widely accepted assumption that $\mathrm{FP}_1 \neq \#\mathrm{P}_1$, this implies that computing both the channel capacity and the optimal psd within polynomial time is unfeasible.

In Chapter 7, we have shown that the constraint functions in convex optimization problems can negatively affect solutions, potentially resulting in non-computable optimal points, even with strictly convex functions. We have found that certain computable convex constraints make it impossible to compute the optimal point to a desired precision, regardless of the strict convexity of the objective function. This indicates the

impossibility of constructing a universal algorithm that takes an arbitrary optimization problem with objective and constraint functions as parameters and computes its optimal point.

Our findings suggest significant challenges in meeting integrity requirements for reliable communication operating at rates close to the capacity in future mobile communications networks. This is attributed to the existence of commonly encountered channels whose capacity cannot be precisely determined within acceptable margins of error by digital computers. Consequently, this makes it very challenging to algorithmically generate codes at high rates and evaluate their efficacy for those particular channel models. Exploring alternative technologies, such as analog computing, might offer more effective solutions for computing these benchmarks.

Bibliography

[1] G. P. Fettweis and H. Boche, "6G: The personal Tactile Internet—and open questions for information theory," *IEEE BITS Inf. Theory Mag.*, vol. 1, no. 1, pp. 71–82, Sep. 2021.

[2] ——, "On 6G and trustworthiness," *Commun. ACM*, vol. 65, no. 4, pp. 48–49, Apr. 2022.

[3] C. E. Shannon, R. G. Gallager, and E. R. Berlekamp, "Lower bounds to error probability for coding on discrete memoryless channels. I," *Inf. Contr.*, vol. 10, no. 1, pp. 65–103, Jan. 1967.

[4] S. Arimoto, "An algorithm for computing the capacity of arbitrary discrete memoryless channels," *IEEE Trans. Inf. Theory*, vol. 18, no. 1, pp. 14–20, Jan. 1972.

[5] R. Blahut, "Computation of channel capacity and rate-distortion functions," *IEEE Trans. Inf. Theory*, vol. 18, no. 4, pp. 460–473, Jul. 1972.

[6] H. Boche, R. F. Schaefer, and H. V. Poor, "Algorithmic computability and approximability of capacity-achieving input distributions," *IEEE Trans. Inf. Theory*, vol. 69, no. 9, pp. 5449–5462, Sep. 2023.

[7] R. G. Gallager, *Information Theory and Reliable Communication*. John Wiley & Sons, Inc., 1968.

[8] H. Boche, R. F. Schaefer, and H. V. Poor, "Shannon meets Turing: Non-computability and non-approximability of the finite state channel capacity," *Commun. Inf. Syst.*, vol. 20, no. 2, pp. 81–116, Nov. 2020.

[9] ——, "Identification capacity of correlation-assisted discrete memoryless channels: Analytical properties and representations," in *Proc. IEEE Int. Symp. Inf. Theory*, Paris, France, Jul. 2019, pp. 470–474.

[10] ——, "Communication under channel uncertainty: An algorithmic perspective and effective construction," *IEEE Trans. Signal Process.*, vol. 68, pp. 6224–6239, Oct. 2020.

[11] A. M. Turing, "On computable numbers, with an application to the Entscheidungsproblem. A correction," *Proc. London Math. Soc.*, vol. 2, no. 43, pp. 544–546, Jan. 1938.

[12] K. Weihrauch, *Computable Analysis: An Introduction*. Springer Science & Business Media, 2000.

[13] K. Gödel, "Die Vollständigkeit der Axiome des logischen Funktionenkalküls," *Monatshefte für Mathematik und Physik*, vol. 37, no. 1, pp. 349–360, Dec. 1930.

[14] ——, "On undecidable propositions of formal mathematical systems," May 1934, Lectures at the Institute for Advanced Study, Princeton, NJ. [Online]. Available: https://albert.ias.edu/entities/publication/9587ae8c-4cc2-424f-ac21-8681a9c19d8a/details

[15] S. C. Kleene, *Introduction to Metamathematics*. Amsterdam : North-Holland Publishing ; Groningen : P. Noordhoff N.V., 1952.

[16] M. L. Minsky, "Recursive unsolvability of Post's problem of "tag" and other topics in theory of Turing machines," *Annals Math.*, pp. 437–455, Nov. 1961.

[17] A. Grigorescu, H. Boche, R. F. Schaefer, and H. V. Poor, "Capacity region continuity of the compound broadcast channel with confidential messages," in *Proc. IEEE Inf. Theory Workshop*, Jerusalem, Israel, Apr. 2015, pp. 1–5.

[18] R. F. Schaefer, A. Grigorescu, H. Boche, and H. V. Poor, "Broadcast channels with confidential messages: Channel uncertainty, robustness, and continuity," in *Physical and Data-Link Security Techniques for Future Commun. Syst.*, M. Baldi and S. Tomasin, Eds., vol. 358. Springer, Cham, 2016, pp. 69–91.

[19] A. Grigorescu, H. Boche, R. F. Schaefer, and H. V. Poor, "Capacity of finite state channels with feedback: Algorithmic and optimization theoretic properties," in *Proc. IEEE Int. Symp. Inf. Theory*, Espoo, Finland, Jul. 2022, pp. 498–503.

[20] ——, "Capacity of finite state channels with feedback: Algorithmic and optimization theoretic properties," *IEEE Trans. Inf. Theory*, vol. 70, no. 8, pp. 5413–5426, Aug. 2024.

[21] H. Boche, A. Grigorescu, R. F. Schaefer, and H. V. Poor, "Algorithmic computability of the capacity of Gaussian channels with colored noise," in *Proc. IEEE Global Telecommun. Conf.*, Kuala Lumpur, Malaysia, Dec. 2023, pp. 4375–4380.

[22] ——, "Characterization of the complexity of computing the capacity of colored noise Gaussian channels," in *Proc. IEEE Int. Conf. Commun.*, Denver, CO, USA, Jun. 2024, pp. 4114–4119.

[23] ——, "Characterization of the complexity of computing the capacity of colored Gaussian noise channels," *IEEE Trans. Commun.*, vol. 72, no. 8, pp. 4844–4856, Aug. 2024.

[24] ——, "On the non-computability of convex optimization problems," in *Proc. IEEE Int. Symp. Inf. Theory*, Athens, Greece, Jul. 2024, pp. 3083–3088.

[25] G. Kramer, "Information theory," Lecture Notes, Institute for Communications Engineering, Technische Universität München, Munich, Germany, WS 2012-2013.

[26] J. Massey *et al.*, "Causality, feedback and directed information," in *Proc. Int. Symp. Inf. Theory and its Applicat.*, Waikiki, Hawaii, U.S.A., Nov. 1990, pp. 303–305.

[27] G. Kramer, "Directed information for channels with feedback," Ph.D. dissertation, Eidgenössische Technische Hochschule Zürich, 1998.

[28] ——, "Capacity results for the discrete memoryless network," *IEEE Trans. Inf. Theory*, vol. 49, no. 1, pp. 4–21, Jan. 2003.

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

[30] H. W. Kuhn, "Nonlinear programming: A historical view," in *SIAM-AMS Proc.*, vol. 9. American Mathematical Society, 1976, pp. 1–26.

[31] H. W. Kuhn and A. Tucker, "Nonlinear programming," in *Proc. of the Second Berkeley Symp. Math. Statist. Prob.* University of California Press, 1951, pp. 481–492.

[32] A. M. Turing, "On computable numbers, with an application to the Entscheidungsproblem," *Proc. London Math. Soc.*, vol. 2, no. 42, pp. 230–265, Nov. 1936.

[33] M. B. Pour-El and J. I. Richards, *Computability in Analysis and Physics*. Cambridge University Press, 2017.

[34] K. Ko, *Complexity Theory of Real Functions*. Springer Science & Business Media, 2012.

[35] S. Arora and B. Barak, *Computational Complexity: A Modern Approach*. Cambridge University Press, 2009.

[36] X. Zheng and K. Weihrauch, "The arithmetical hierarchy of real numbers," *Math. Log. Quarterly*, vol. 47, no. 1, pp. 51–65, 2001.

[37] J. Avigad, V. Brattka, and R. Downey, *Computability and Analysis: The Legacy of Alan Turing*. Cambridge, UK: Cambridge University Press, 2014.

[38] R. I. Soare, "Recursively enumerable sets and degrees," *Bull. Am. Math. Soc.*, vol. 84, no. 6, pp. 1149–1181, 1978.

[39] H. Friedman, "The computational complexity of maximization and integration," *Adv. in Math.*, vol. 53, no. 1, pp. 80–98, 1984.

[40] R. F. Schaefer and H. Boche, "Robust broadcasting of common and confidential messages over compound channels: Strong secrecy and decoding performance," *IEEE Trans. Inf. Theory*, vol. 9, no. 10, pp. 1720–1732, Aug. 2014.

[41] I. Csiszár, "Almost independence and secrecy capacity," *Problemy Peredachi Informatsii*, vol. 32, no. 1, pp. 48–57, 1996.

[42] U. Maurer and S. Wolf, "Information-theoretic key agreement: From weak to strong secrecy for free," in *International Conference on the Theory and Applications of Cryptographic Techniques*. Springer, Jan. 2000, pp. 351–368.

[43] H. Boche, R. F. Schaefer, and H. V. Poor, "On the continuity of the secrecy capacity of compound and arbitrarily varying wiretap channels," *IEEE Trans. Inf. Forensics Security*, vol. 10, no. 12, pp. 2531–2546, Aug. 2015.

[44] K. Kobayashi and H. Morita, "An input/output recursion for the trapdoor channel," in *Proc. IEEE Int. Symp. Inf. Theory*, Lausanne, Switzerland, Jun. 2002, p. 423.

[45] B. Huleihel, O. Sabag, H. H. Permuter, N. Kashyap, and S. Shamai Shitz, "Computable upper bounds on the capacity of finite-state channels," *IEEE Trans. Inf. Theory*, vol. 67, no. 9, pp. 5674–5692, Sep. 2021.

[46] S. Verdú and T. Han, "A general formula for channel capacity," *IEEE Trans. Inf. Theory*, vol. 40, no. 4, pp. 1147–1157, Jul. 1994.

[47] C. Shannon, "The zero error capacity of a noisy channel," *IRE Trans. Inf. Theory*, vol. 2, no. 3, pp. 8–19, Sep. 1956.

[48] S. Tatikonda, "Control under communication constraints." Ph.D. dissertation, Massachusetts Institute of Technology, 2000.

[49] S. Tatikonda and S. Mitter, "The capacity of channels with feedback," *IEEE Trans. Inf. Theory*, vol. 55, no. 1, pp. 323–349, Dec. 2008.

[50] H. H. Permuter, T. Weissman, and A. J. Goldsmith, "Finite state channels with time-invariant deterministic feedback," *IEEE Trans. Inf. Theory*, vol. 55, no. 2, pp. 644–662, Feb. 2009.

[51] R. Dabora and A. J. Goldsmith, "On the capacity of indecomposable finite-state channels with feedback," *IEEE Trans. Inf. Theory*, vol. 59, no. 1, pp. 193–203, Jan. 2013.

[52] J. Chen and T. Berger, "The capacity of finite-state Markov channels with feedback," *IEEE Trans. Inf. Theory*, vol. 51, no. 3, pp. 780–798, Feb. 2005.

[53] H. H. Permuter, P. Cuff, B. Van Roy, and T. Weissman, "Capacity of the trapdoor channel with feedback," *IEEE Trans. Inf. Theory*, vol. 54, no. 7, pp. 3150–3165, Jun. 2008.

[54] O. Elishco and H. H. Permuter, "Capacity and coding for the Ising channel with feedback," *IEEE Trans. Inf. Theory*, vol. 60, no. 9, pp. 5138–5149, Sep. 2014.

[55] O. Sabag, H. H. Permuter, and N. Kashyap, "Feedback capacity and coding for the BIBO channel with a no-repeated-ones input constraint," *IEEE Trans. Inf. Theory*, vol. 64, no. 7, pp. 4940–4961, Jul. 2018.

[56] ——, "The feedback capacity of the binary erasure channel with a no-consecutive-ones input constraint," *IEEE Trans. Inf. Theory*, vol. 62, no. 1, pp. 8–22, Jan. 2016.

[57] Z. Aharoni, O. Sabag, and H. H. Permuter, "Computing the feedback capacity of finite state channels using reinforcement learning," in *Proc. IEEE Int. Symp. Inf. Theory*, Paris, France, Jul. 2019, pp. 837–841.

[58] C. E. Shannon, "Communication theory of secrecy systems," *Bell Syst. Tech. J.*, vol. 28, no. 4, pp. 656–715, Oct. 1949.

[59] E. Specker, "Der Satz vom Maximum in der rekursiven Analysis," *Ernst Specker Selecta*, pp. 148–159, Feb. 1990.

[60] A. Ephremides and B. Hajek, "Information theory and communication networks: An unconsummated union," *IEEE Trans. Inf. Theory*, vol. 44, no. 6, pp. 2416–2434, Oct. 1998.

[61] A. D. Wyner, "The wire-tap channel," *Bell Syst. Tech. J.*, vol. 54, no. 8, pp. 1355–1387, Oct. 1975.

[62] M. B. Pour-El, "A comparison of five "computable" operators," *Math. Log. Quart.*, vol. 6, no. 15-22, pp. 325–340, 1960.

[63] H. Boche, R. F. Schaefer, and H. V. Poor, "Denial-of-service attacks on communication systems: Detectability and jammer knowledge," *IEEE Trans. Signal Process.*, vol. 68, pp. 3754–3768, May 2020.

[64] H. Boche, R. F. Schaefer, S. Baur, and H. V. Poor, "On the algorithmic computability of the secret key and authentication capacity under channel, storage, and privacy leakage constraints," *IEEE Trans. Signal Process.*, vol. 67, no. 17, pp. 4636–4648, Jul. 2019.

[65] O. Sabag, H. H. Permuter, and H. D. Pfister, "A single-letter upper bound on the feedback capacity of unifilar finite-state channels," *IEEE Trans. Inf. Theory*, vol. 63, no. 3, pp. 1392–1409, Mar. 2017.

[66] C. E. Shannon, "Communication in the presence of noise," *Proc. IRE*, vol. 37, no. 1, pp. 10–21, 1949.

[67] A. D. Wyner, "The capacity of the band-limited Gaussian channel," *Bell Syst. Tech. J.*, vol. 45, no. 3, pp. 359–395, Mar. 1966.

[68] R. B. Ash, "Capacity and error bounds for a time-continuous Gaussian channel," *Inf. Contr.*, vol. 6, no. 1, pp. 14–27, 1963.

[69] C. E. Shannon, "Probability of error for optimal codes in a Gaussian channel," *Bell Syst. Tech. J.*, vol. 38, no. 3, pp. 611–656, May 1959.

[70] R. B. Ash, *Information Theory*. Courier Corporation, 2012.

[71] T. M. Cover, *Elements of Information Theory*. John Wiley & Sons, 1999.

[72] S. Ihara, *Information Theory for Continuous Systems*. World Scientific, 1993, vol. 2.

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

[74] M. S. Pinsker, *Information and Information Stability of Random Variables and Processes*. Holden-Day, 1964.

[75] G. Forney and G. Ungerboeck, "Modulation and coding for linear Gaussian channels," *IEEE Trans. Inf. Theory*, vol. 44, no. 6, pp. 2384–2415, Oct. 1998.

[76] J. G. Proakis and M. Salehi, *Digital Communications*. McGraw-hill New York, 2001, vol. 4.

[77] S. Haykin, *Communication Systems /*, 4th ed. John Wiley & Sons.„ 2001.

[78] M. F. Flanagan, "On proving the water pouring theorem for information rate optimization," in *Int. Conf. on Signals and Electron. Syst.* Citeseer, 2006.

[79] A. Ben-Tal and A. Nemirovski, *Lectures on Modern Convex Optimization: Analysis, Algorithms, and Engineering Applications*. SIAM, 2001.

[80] Y. Nesterov and A. Nemirovskii, *Interior-Point Polynomial Algorithms in Convex Programming*. SIAM, 1994.

[81] J. E. Kelley, Jr, "The cutting-plane method for solving convex programs," *J. of Soc. Ind. and Appl. Math.*, vol. 8, no. 4, pp. 703–712, Dec. 1960.

[82] N. Z. Shor, *Minimization Methods for Non-Differentiable Functions*. Springer Science & Business Media, 2012, vol. 3.

[83] N. Karmarkar, "A new polynomial-time algorithm for linear programming," in *Proc. Sixteenth Ann. ACM Symp. on Theory of Comput.*, ser. STOC '84. Association for Computing Machinery, Dec. 1984, p. 302–311.

List of Figures

List of Tables