universität wien

MASTERARBEIT | MASTER'S THESIS

Titel | Title

On optimizing three party Bell Inequalities or how Alice, Bob and Charlie play the Perfect Game

verfasst von | submitted by

Philipp Neusser BSc

angestrebter akademischer Grad | in partial fulfilment of the requirements for the degree of

Master of Science (MSc)

Wien | Vienna, 2024

Studienkennzahl lt. Studienblatt | Degree programme code as it appears on the student record sheet:	UA 066 876
Studienrichtung lt. Studienblatt | Degree programme as it appears on the student record sheet:	Masterstudium Physics
Betreut von | Supervisor:	Dipl.-Phys. Dr. Markus Müller

Acknowledgements

There are two people, I want to give a special mention. There is Miguel Navascués, who got me interested in the field of non local physics. But the biggest thanks without a doubt goes to Mateus Araújo, who supervised me through out this project and on which work this thesis is based on!

Abstract

One of the most surprising discoveries in 20th century physics is the violation of locality. This violation can be shown in so called Bell scenarios, where measurements on entangled particles in space-like separated regions lead to the violation of certain statistical inequalities, which have to hold for any local theory. These inequalities are in general just a sum of conditional probabilities $P(a, b|x, y)$ for the measurement outputs a, b given inputs x, y weighted by some coefficients $M_{xy}^{ab} \in \mathbb{R}$, which is smaller than some local bound L for all local theories, i.e. $\sum_{abxy} M_{xy}^{ab} P(a, b|x, y) \leq L$. One of the problems is that there are infinitely many different inequalities referring to the same physics, but with different local and quantum bounds, which can be difficult to handle especially by the experimenter.

A solution to that is to bring these inequalities into a form, by which they can be understood as some sort of game [1], in which a referee sends questions x, y from a probability distribution $\mu(x, y)$ to the players, most often called Alice and Bob. They then have to provide answers a, b, which are verified by the referee with probability $V(a, b|x, y)$. The inequalities have then the form $\sum_{abxy} \mu(x, y) V(a, b|x, y) P(a, b|x, y) \leq L$ with several advantages:

- It is clear and straightforward how it can be implemented as an experiment
- The local- and the quantum bound are not arbitrary numbers, but lay between 0 and 1 representing the probability by which the game is won in single round, thus it is much easier to draw statistical conclusions from it
- Given that the bounds represent the winning chances of the games, several different games can be compared e.g. by how much advantage a quantum strategy can give over local strategy.

In a paper from 2020 [2], it was shown how a general two party inequality can be brought into its optimal game form, i.e. maximising the gap between the local and quantum bound, by linear programming. This thesis can be understood as a generalisation of this paper to general three party scenarios.

Abstract

The thesis is divided into four parts:

In the first part I will give a basic introduction to the modern non-local physics giving the state of the art mathematical formalism, which is used in the field, because there is a lot of confusion in the historical literature about several definitions.

In the second part I show how to optimise a general three party inequality and code to do so. This code is provided in the Julia language and uses the JuMP package in conjunction with the Hypatia solver for the linear optimisation.

In the third part I compare all [3] the 46 three player inequalities in the two input to output scenario, finding several inequalities stronger than CHSH. I do special analysis of equation #7 in the list [2]. As it has the largest gap besides the the famous GHZ [4] inequality.

In the fourth part I will analyse several inequalities with more then 2 outputs found in the literature [1]and improve or complete their results by providing the best numbers.

[1]Basile Grandjean, Yeong-Cherng Liang, Jean-Daniel Bancal, Nicolas Brunner, Nicolas Gisin,Bell inequalities for three systems and arbitrarily many measurement outcomes arXiv:1204.3829v2

Kurzfassung

Eine der überraschendsten Entdeckungen in der Physik des 20. Jahrhunderts ist die Verletzung der Lokalität. Diese Verletzung kann in sogenannten Bell-Szenarien gezeigt werden, bei denen Messungen an verschränkten Teilchen in raumartig getrennten Regionen zu Verletzungen bestimmter statistischer Ungleichungen führen, die für jede lokale Theorie gelten müssen.

Diese Ungleichungen sind im Allgemeinen nur eine Summe von bedingten Wahrscheinlichkeiten $P(a,b|x,y)$ der Ausgaben a,b der Messungen bei gegebenen Eingaben x,y gewichtet mit Koeffizienten $M_{xy}^{ab} \in \mathbb{R}$, die kleiner sind als eine lokale Grenze L für lokale Theorien, d.h. $\sum_{abxy} \mu(x,y)V(a,b|x,y)P(a,b|x,y) \leq L$. Ein Problem besteht darin, dass es unendlich viele verschiedene Ungleichungen gibt, die sich auf dieselbe Physik beziehen, aber unterschiedliche lokale und quantenmechanische Grenzen haben, was insbesondere für den Experimentator schwierig sein kann.

Eine Lösung dafür besteht darin, diese Ungleichungen in eine Form zu bringen, in der sie als eine Art Spiel verstanden werden können [1], bei dem ein Schiedsrichter Fragen aus einer Wahrscheinlichkeitsverteilung $\mu(x,y)$ an die Spieler sendet, die meistens Alice und Bob genannt werden. Sie müssen dann Antworten a,b geben, die vom Schiedsrichter mit einer Wahrscheinlichkeit $V(a,b|x,y)$ als korrekt annimmt. Die Ungleichungen haben in dieser Form, $\sum_{abxy} \mu(x,y)V(a,b|x,y)P(a,b|x,y) \leq L$, mehrere Vorteile:

1. Es ist klar und einfach, wie es als Experiment umgesetzt werden kann.

2. Die lokale und die quantenmechanische Grenze sind keine willkürlichen Zahlen, sondern liegen zwischen 0 und 1 und repräsentieren die Wahrscheinlichkeit, mit der das Spiel in einer einzelnen Runde gewonnen wird. Daher ist es viel einfacher, statistische Schlussfolgerungen daraus zu ziehen.

3. Angesichts der Tatsache, dass die Grenzen die Gewinnchancen der Spiele repräsentieren, können mehrere Spiele verglichen werden, z.B. inwieweit eine Quantenstrategie gegenüber einer lokalen Strategie einen Vorteil bieten kann.

In einer Arbeit aus dem Jahr 2020 [2] wurde gezeigt, wie eine allgemeine Zweiparteien-Ungleichung in ihre optimale Spielform gebracht werden kann, d.h. durch lineare Programmierung den Abstand zwischen der lokalen und quantenmechanischen Grenze zu maximieren. Diese Arbeit kann als Verallgemeinerung dieser Arbeit auf allgemeine Dreiparteien-Szenarien verstanden werden.

Die Arbeit ist in vier Teile unterteilt:

Im ersten Teil werde ich eine grundlegende Einführung in die moderne nichtlokale Physik geben und den aktuellen mathematischen Formalismus vorstellen, der in diesem

Bereich verwendet wird, da es in der historischen Literatur viele Missverständnisse über verschiedene Definitionen gibt.

Im zweiten Teil zeige ich, wie man eine allgemeine Dreiparteien-Ungleichung optimiert und den dazu gehörigen Code erstellt. Dieser Code ist in der Julia-Sprache verfasst und verwendet das JuMP-Paket in Verbindung mit dem Hypatia-Löser für die lineare Optimierung.

Im dritten Teil vergleiche ich alle 46 Dreiparteien-Ungleichungen in einem Szenario mit zwei Eingaben und zwei Ausgaben und finde dabei mehrere Ungleichungen, die stärker sind als die CHSH-Ungleichung. Ich führe eine spezielle Analyse der Gleichung Nr. 7 in der Liste durch, da sie die größte Kluft neben der berühmten GHZ-Ungleichung [4] aufweist.

Im vierten Teil analysieren wir Ungleichungen in der Literatur mit mehr als 2 Ausgaben[1] und verbessern bzw. vervollständigen deren die Resultate.

[1] Basile Grandjean, Yeong-Cherng Liang, Jean-Daniel Bancal, Nicolas Brunner, Nicolas Gisin,Bell inequalities for three systems and arbitrarily many measurement outcomes arXiv:1204.3829v2

Contents

List of Tables

List of Figures

1. Introduction

At about 500 B.C. Zeno from Elea has formulated the famous paradox, where Achilles and a turtle are having a race. The argument goes, that Achilles, who is starting behind the much slower turtle, can never catch up to it, because when he reaches the point where the turtle was, the turtle will have moved to a further point and when he reaches this further point, again the turtle will have moved and so on to infinity. So, although he is much faster and the distance he has to cover from one point to another becomes shorter and shorter, Achilles has to go over infinitely many points in a finite amount of time to catch the turtle, which seems impossible. Of course, using modern analysis, we can set up the problem, such that when we take the limit, we see that there is no paradox at all. Achilles can indeed cover all the distances in a finite amount of time.

Note that, the paradox between Achilles and the turtle only bother us as such, because Achilles has to cover the distance between him and the turtle to catch it. If he could slaughter the turtle by just using some non-local super power, we wouldn't be worried by these kinds of gedankenexperiments. Things can only effect other things within their

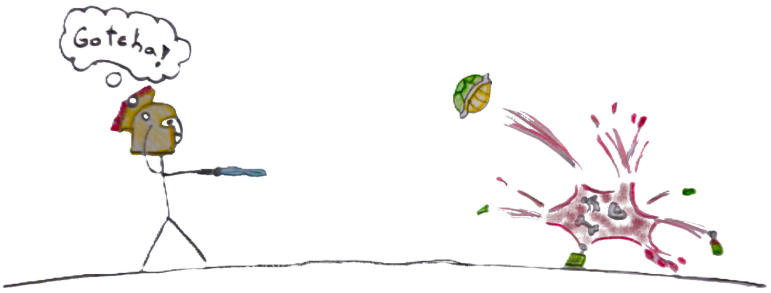

Figure 1.1.: Achilles slaughters the turtle with non-local super powers!

direct neighbourhood. This is true for Achilles and turtles and intuitively it seems to be true for all physical objects. This notion of locality is deeply engraved in our minds.

This is also the reason why Newton was struggling with his own law of gravity. Despite the great success of the law, its non-local nature seems unacceptable. In Newton's law

$$F = G\frac{m_1 m_2}{r^2} \tag{1.1}$$

one particle of mass effects another particle of mass somewhat directly, that is instantaneously no matter the distance. There is no explanation, how gravity is mediated from one

particle to another, other than non-locally via "action at a distance". The same problem arises with Coulomb's Law

$$F = k_e \frac{q_1 q_2}{r^2}, \tag{1.2}$$

which describes the force between two charged particles. Here, one electrical charge effects another electrical charge also via "action at a distance". In this case, it took about one hundred years for locality to be restored. In 1861 guided by experiments from physicists like Faraday and Lorentz, which showed the intimate connection between moving charges and magnetism, Maxwell was able to describe both the electric force and the magnetic force with a field approach. In his theory of electromagnetism the forces between charges are mediated by the electromagnetic field. This field carries the energy and the momentum from one charge to another via electromagnetic waves. These waves propagate with a finite amount of speed and only by distorting its direct neighbourhood and the world is local again.

Restoring locality was also one of the main motivations for Einstein, when he was working on a new law of gravity. He also succeeded with a field approach. In general relativity the matter determines the geometry of space-time and vice versa the geometry of space-time determines the dynamics of the matter. In this way, space-time itself acts as a field, by which gravity is mediated. A particle of mass distorts the space-time around it. This distortion propagates through space-time, e.g. by gravitational waves, with a finite speed eventually effecting another particle and the world is local again.

With Maxwell's and Einstein's equations in place, locality seemed to be well assured. This was until experiments advanced further and further into the microscopic world and phenomena arise, like the spectral lines of the hydrogen atom, which couldn't be explained by any known theory. So, to describe these phenomena, physicist such as Schrödinger and Heisenberg, came up with a new theory called quantum mechanics. This theory, though hugely successful in describing these phenomena, turned out to be non-local again as Einstein, Podolsky and Rosen pointed out.

Their argument is best understood in a version David Bohm came up with. In his version a pair of spin-$\frac{1}{2}$-particles is produced in the singlet state, which we will consider in a basis along the z-axis

$$|\Psi\rangle = \frac{1}{\sqrt{2}}\Big(|z_+\rangle\,|z_-\rangle - |z_-\rangle\,|z_+\rangle\Big) \tag{1.3}$$

and in a basis along the x-axis

$$|\Psi\rangle = -\frac{1}{\sqrt{2}}\Big(|x_+\rangle\,|x_-\rangle - |x_-\rangle\,|x_+\rangle\Big). \tag{1.4}$$

The particles are then separated and send off into two different labs. In each of the labs it is possible to take spin measurements. Lets cal them A and B. Now it easy to see that,

if the labs take spin measurements both along the z-axis or both along the x-axis, the measurements will turn out to be perfectly anti-correlated. That is if one lab measures spin up, the other lab will measure spin down with certainty. On the other hand, if one lab measures along the z-axis and the other lab measures along the x-axis, it turns out there is no correlation between the two measurements at all. This becomes clear, if we consider for example, the state for the first lab in the z-basis and the state for the second lab in x-basis

$$|\Psi\rangle = \frac{1}{\sqrt{2}}\left(|z_+\rangle \frac{|x_+\rangle - |x_-\rangle}{\sqrt{2}} - |z_-\rangle \frac{|x_+\rangle + |x_-\rangle}{\sqrt{2}}\right). \tag{1.5}$$

Now it is straightforward to see, that if lab one measures along z no matter the outcome, there is no definite spin along x in the second lab and the measurements will come out uncorrelated. Note that this prediction is independent of when and where these measurements take place. Therefore if we consider one lab to take its measurement before the other, it influences instantaneously whether the other particle will have definitive spin along x or z, even if the particles are light years apart. This is clearly at odds with locality and Einstein, Podolsky and Rosen took it as evidence for the a lack of 'completeness' in the theory quantum mechanics, implying that it eventually has to be replaced by a local theory.

It took about thirty years until John Bell saw the whole significance of this gedankenexperiment. In several papers he was able to extend the argument, showing that not only is quantum mechanics non-local, but also that any theory able to replace quantum mechanics will necessarily be non-local too. Bell noticed, that the original EPR-argument although showing the non-locality of quantum mechanics, doesn't show the necessity for it. In fact, Bell was able to write down a completely local model, that explains all the correlations of the EPR-experiment, such that quantum mechanics seems to be just a bad non-local explanation for a simple local fact. The crucial point is, that in such an EPR-scenario [1], we can also test for other statistical correlations between the measurements of the two labs. Correlations, that are much harder to describe by a local model. In fact we can choose the correlations, such that it is impossible for any local model to describe them. This statistical correlations are usually given in form of an inequality, which has to hold for any local model. The most famous of these inequalities is the CHSH-inequality

$$|\langle A_1 B_1\rangle + \langle A_1 B_2\rangle + \langle A_2 B_1\rangle - \langle A_2 B_2\rangle| \leq 2, \tag{1.6}$$

where A_1 and A_2 are two directions of spin measurement in lab A and B_1 and B_2 are two directions of spin measurement in lab B. Then $\langle A_1 B_2\rangle$ is the expectation value on a spin measurement on a shared singlet state in direction A_1 in lab 1 and B_2 in lab 2 and so on. This for itself is not astonishing, surely there are correlation which are impossible to describe by a local model. It is only interesting, if after a big number of these spin

[1] A scenario in, which two or more experimenters are taking measurements in space-like separated region, eventually comparing statistical correlations of their outcomes, is now generally known as Bell scenario and we will discuss the formal details in the first chapter of this thesis

measurements we really would find an outcome > 2. Then all local theories are proven wrong. Note, that it is not necessary to use quantum mechanics and spin measurements in conjunction with this CHSH-inequality to find statistical evidence for non-locality, but quantum mechanics predicts an outcome of $2\sqrt{2} \sim 2.828$, if the angles between the measurement directions are chosen in the right way[2], and therefore provides a recipe to 'break' locality. It might well be, that future experiments can show correlations, that even quantum mechanics is not able to replicate, but without a concrete theory, it is not clear how to prepare such experiments. By now quantum mechanics has been verified in its prediction several times, most notably in three experiments 2015 [5][6][7], where all [3], so called loopholes, that would provide a backdoor for local models where closed. The world is non-local and no local theory will come to a rescue.

So, what? Does that really mean Achilles can slay the turtle with his non-local super-powers. (Un)fortunately, most certainly not ?! Because what we can also learn from these experiments is, that non-local doesn't mean non-local, there can be important differences in the non-locality different theories provide. For example, the non-locality in Newton's law is in some much 'stronger' than the non-locality in quantum mechanics. [4]. there is a question of how non-locality is part of this world and, which non-local features are therefore implied and which are not. This is of uttermost importance, because the findings will eventually underlie all of our fundamental laws in physics. By now a whole field of physics has centred around non-locality and its features. In this field of non-local physics Bell scenarios and Bell inequalities are used to understand, categorise and test non-locality and its features. For that it is always important to find new inequalities to study and test for. Although it is formally possible give all the Bell inequalities of a given scenario, the number of them explodes quite quickly, when the scenarios get more complicated and to find useful inequalities can become tricky [5]. The problem is, that depending on how you are looking for these inequalities, you might get different versions of them, which look different, but are actually the same [6] and a lot of them are 'weak' in their statistical significance, so it is very hard to test them, this does also depend on the version of the inequality. So ideally we would like to have way to find the 'strongest' version of an inequality and a way to compare to the strength of other inequalities in the scenario. In the Bell scenarios with only two parties, people were already quite successful in finding and categorise a lot of useful inequalities and also provide a meaningful way to compare different inequalities in their strength and find their best versions [2].In scenarios with more then two parties things are not quite so clear. People were able to categorise some of them [3], but a lot of work is left. In this thesis we will provided several algorithms

[2] Ideally, one wants the directions within a lab to be orthogonal and the directions between the labs tilted by 45°

[3] At least all known loopholes and those, that are in the hands of the experiments. Loopholes such as super-determinism or retro causation can never been closed.

[4] We will see what that exactly means in the first chapter of this thesis

[5] Remember the needle in a haystack ?!

[6] e.g. in the early days of non-local physics the CHSH-inequality and the CH-inequality, were first thought to be different inequalities

to find the best versions of an arbitrary inequality in a three party scenario and thereby give a meaningful measure of comparing different inequalities with one and another.

Therefore we will first give an introduction into the necessary mathematical framework. Then we will spend some time on the two main notations for Bell inequalities, that is the correlator notation and the Collins-Gisin notation. This is necessary, because we need to understand how to translate them into a full probability notation, which is required for the optimisation algorithm. Finally in the last two parts of the thesis, we will see how we can use a linear optimisation to find the best version of a given Bell inequality and apply this algorithm to various inequalities in the literature. Notably, we will analyse all three party inequalities in the 222222-scenario.

2. Mathematical Framework

In the following chapter we will setup the mathematical framework needed for this thesis. In each of the following subsection we will introduce a key concept or mathematical object necessary to express the physics of Bell non-locality. To make sure that the framework follows the modern state of the art, we will mostly hold on to the book of Valerio Scarani [8] "Bell Non-locality" (2019) and the lecture notes of Miguel Navascues on the topic of space-like quantum correlations (2022), which widely agree with each other. We will only deviate from this standard to make things more practical for dealing with three players as their lies the main focus of this work. Note that we will give most of the definitions for the two player scenario, if it generalises trivially to more than two, just because it is cleaner to read and easier to understand. Also we will most often give definitions and equations only for one player or one possible pair of players, when for the other players or pairs these equations only differ by a simple substitution

2.1. Scenario

As mentioned in the introduction a Bell scenario can be seen as some sort of a game. In each round of the game a number of players are each provided some input, on which they have to produce some output. The inputs and the outputs are finally compared and depending on the rules of the game it is decided if the round was won or lost. The inputs and outputs are drawn from fixed list of possible inputs and outputs, although these lists don't have to be the same for each player.

Therefore a Bell scenario is defined by the number of players also called parties and the number of possible inputs and outputs each party is provided. For practical reasons, it became standard to refer to name the players alphabetically, if their number allows for it. The first player is called **A**lice with inputs x and outputs a, the second **B**ob with inputs y and outputs b and third **C**harlie with inputs z and outputs c.

Further we will refer to total number of possible inputs and outputs for Alice by M and m respectively. The same goes for Bob, where we use N for the inputs and n for the outputs. Finally and for Charlie we use L, l in the same way. This notation is a slight deviation from the standard, but there less fiddling with indices and as long we stick to two and three player scenarios, it is convenient . A scenario is then given by defining (M, N, m, n) in the two player case and (M, N, L, m, n, l) in the three player case, and the order of course matters. The CHSH-Scenario for example is $(2, 2, 2, 2)$-scenario, if we would allow Alice third possible output it would be a $(2, 2, 3, 2)$-scenario

an so forth and a three player scenario might look like $(3, 3, 2, 3, 3, 2)$, where Alice and Bob have three inputs and three outputs and Charlie only has two inputs and two outsputs.

The inputs and outputs themselves are usually named by numbers, which is useful, but it is important to note that it is by no means necessary as they are just labels. One could also use Paris, Vienna, New York,... to name the outputs or the inputs, it would be just impractical. In this thesis we will most often start the labelling with 1, such that depending on the scenario, we have $x \in \{1, 2, \ldots, N\}$,$y \in \{1, 2, \ldots, M\}$,$z \in \{1, 2, \ldots, L\}$, $a \in \{1, 2, \ldots, n\}$, $x \in \{1, 2, \ldots, m\}$,$z \in \{1, 2, \ldots, l\}$. Although starting with zero might be standard, we want to be in line with `Julia` language, which starts counting by 1. An exception are scenario with exactly two outputs for all players, there it will often be useful to take the outputs $a, b, c \in \{-1, 1\}$ instead of $\{1, 2\}$ or $\{0, 1\}$, as we will see.

2.2. Process

A process $\mathcal{P}_\lambda$ is the strategy the players agree on to produce their output. This strategy might be deterministic or stochastic and is specified by giving the conditional probabilities $P_\lambda(ab|xy)$, i.e. the probabilities that the output x, y is produced by the players given the input a, b.
We only require for these probabilities to be non-negative, i.e.

$$P_\lambda(ab|xy) \geq 0 \qquad \forall a, b, x, y \tag{2.1}$$

and to be properly normalised, i.e.

$$\sum_{ab} P_\lambda(ab|xy) = 1 \forall x, y. \tag{2.2}$$

Note that we don't ask how the outputs are produced, i.e. by which physical process or method, therefore a process $\mathcal{P}_\lambda$ is entirely specified by only giving all these probabilities, i.e.

$$\mathcal{P}_\lambda := \{P_\lambda(ab|xy) | a \in \mathcal{A}, b \in \mathcal{B}, x \in \mathcal{X}, y \in \mathcal{Y}\} \tag{2.3}$$

and can be conveniently written in form of a matrix. In the simple 2222-scenario such a matrix would look the following

$$\left(\begin{array}{cc|cc} P(00|00) & P(01|00) & P(00|01) & P(01|01) \\ P(10|00) & P(11|00) & P(10|01) & P(11|01) \\ \hline P(00|10) & P(01|10) & P(00|11) & P(01|11) \\ P(10|10) & P(11|10) & P(10|11) & P(11|11) \end{array}\right) \tag{2.4}$$

The inputs vary by each block, indicated by the lines and outputs vary within each of these block. Although for more complicated scenarios, which are typically handled by computers, it is even more convenient to store a process in a multi-dimensional

$M \times N \times L \times m \times n$-array, where each entry gives the appropriate probability $P(ab|xy)$. In fact using the normalisation condition 2.2 (NC), we don't have to give all the probabilites to specify a process completely. For each pair x, y we can recover one probability by the NC, thus in the end we will need

$$D_{gen} = M \cdot N \cdot (n-1) \cdot (m-1) \tag{2.5}$$

parameters to uniquely describe a process. At this point the output a can still depend on the input y of the other player, indicating that a processes in its most general form, might involve the players to communicate their inputs and outputs with each other. Although we typically impose some conditions on the processes, which restricts the processes to some type we want to study. The most important conditions are the no-signaling (NS) condition and the locality-condition and we will discuss both of them in detail as they are crucial to our argument.

2.3. Behaviour

In the end we don't know which strategy Alice and Bob will use each round, they might stick to one processes or switch randomly between a variety of different processes. Therefore what we observe might be a mixture of different processes. This mixture of different processes $P(ab|xy)$ is called behaviour $P(ab|xy)$. For example the players might agree on the following strategy involving two processes $\mathcal{P}_1, \mathcal{P}_2$,

$$\mathcal{P} = \begin{cases} \mathcal{P}_1 & \text{on saturday and sunday} \\ \mathcal{P}_2 & \text{from Monday to Friday.} \end{cases} \tag{2.6}$$

The observed behavior would then be a mixture

$$P(ab|xy) = \frac{2}{7} P_1(ab|xy) + \frac{5}{7} P_2(ab|xy) \tag{2.7}$$

Of course in the most general case the players can switch between an uncountable numbers of processes following some probability distribution between them. by using the λ as a parameter, which runs over all the possible processes, we can describe this by

$$P(ab|xy) = \int d\lambda Q(\lambda) P_\lambda(ab|xy), \tag{2.8}$$

where $Q(\lambda)$ is the probability distribution choosing from all processes $\mathcal{P}_\lambda$. Therefore we need

$$Q(\lambda) \geq 0 \quad \text{and} \quad \int d\lambda Q(\lambda) = 1. \tag{2.9}$$

2.4. No-Signaling

In this section we will look at the no-signaling condition. The no-signaling condition is a very general restriction and it is assumed to hold through this thesis for all processes. The no-signaling condition excludes all processes, that enables the players to communicate their input to other players during a round of the game, even by chance. Therefore it has to ensure, that the output statistics of a player is independent from another players input. Otherwise, a player could use this dependence to communicate over space-like separated regions, which would be in conflict with Einstein's theory of relativity. This is easy to see, just assume Bob's output statistic $P(b|x,y)$ would depend on x, then Alice could manipulate her input and thereby encoding information in Bob's statistic. So formally speaking we need,

$$P(a|x,y) = P(a|x,y') \quad \forall a \in A, x \in X, y, y' \in Y \tag{2.10}$$

$$P(b|x,y) = P(b|x',y) \quad \forall b \in B, y \in Y, x, x' \in X \tag{2.11}$$

Note that this is not equivalent with saying that a players output does not depend on another players input, e.g. $a(x) \neq a(x,y)$. In fact this is the subtle difference becomes important, when we define locality in the next chapter.
First we want to take a closer look at the constraints a no-signaling condition imposes on processes. These constraints are simply called no-signaling constraints (NSC). We derive these constraints simply by plugging in concrete values into the NS-conditions 2.10 and 2.11 and sum out the other players output (as implied). For example in the 2222-scenario if we fix Alice's output $a = 0$ and her input $x = 0$, we get

$$P(0|0y) = P(0|0y'). \tag{2.12}$$

We then write out the implied sum over b

$$P(00|0y) + P(01|0y) = P(00|0y') + P(01|0y') \tag{2.13}$$

and finally choose $y \neq y'$ to find a non trivial NSC, that is

$$P(00|00) + P(01|00) - P(00|01) - P(01|01) = 0. \tag{2.14}$$

Of course this is just one of the NSCs in the 2222-scenario, to constrain the set of all possible no-signaling processes, we have to find all the NSCs in the scenario. In the 2222-scenario this is easy, because there is only one possible way to choose $y \neq y'$. One has to be 0 and the other has to be 1 and switching the two will just change the signs of the NSC, but doesn't lead to a new NSC. But we can change x from 0 to 1, which gives an additional NSC. Now we have found two independent NSC

$$P(00|00) + P(01|00) - P(00|01) - P(01|01) = 0 \tag{2.15}$$

and

$$P(00|10) + P(01|10) - P(00|11) - P(01|11) = 0 \tag{2.16}$$

Now one is tempted to change also a from 0 to 1, leading to a total of 4 NSCs. But these NSCs are redundant, because of the normalisation condition

$$\sum_a P(a|x) = 1. \tag{2.17}$$

So knowing the NSCs for all outputs a but one is enough. So we are done. There are exactly two independent NSC for Alice in the 2222-scenario. Further because of the symmetry of the scenario, by similar arguments we get two NSCs for Bob,

$$P(00|00) + P(10|00) - P(00|10) - P(10|10) = 0 \tag{2.18}$$

and

$$P(00|01) + P(10|01) - P(00|11) - P(10|11) = 0. \tag{2.19}$$

Therefore in the $(2, 2, 2, 2)$-scenario we have a total of 4 NSCs. As we go up with the number of inputs, let's say to a $(2, 2, 3, 3)$-scenario, we would expect two more NSCs for Alice for a given x and a, because there are now 3 ways to choose $y \neq y'$ namely the pairs $(0, 1)$,$(1, 2)$,$(0, 2)$, e.g. let again be $x = 0$ and $a = 1$ then we have

$$NSC(0, 1) = P(00|00) + P(01|00) - P(00|01) - P(01|01) = 0 \tag{2.20}$$

$$NSC(1, 2) = P(00|01) + P(01|01) - P(00|02) - P(01|02) = 0 \tag{2.21}$$

$$NSC(0, 2) = P(00|00) + P(01|00) - P(00|02) - P(01|02) = 0, \tag{2.22}$$

All three are valid NSCs, but given a closer look, they are not independent. We can get each one of them by adding the other two together. For example if we add NSC$(0, 1)$ to the NSC$(1, 2)$, we can derive the NSC$(0, 2)$, assuming all other entries remain the same of course. It is trivially to see, that "chaining" of NSC will generalise to arbitrarily high numbers of inputs, i.e. for an n input scenario we only need to give the NSCs with $(0, 1), (1, 2), \ldots, (n - 2, n - 1)$. All other NSCs$(y \neq y')$ are redundant and can be derived "chaining" the aforementioned NSCs together, e.g. $NSC(2, 5) = NSC(2, 3) + NSC(3, 4) + NSC(4, 5)$. Therefore for each a, but one because of the normalisation, and each x, we get a family of $n - 1$ independent NSC$(y \neq y')$s and surely a similar argument is true for Bob. Therefore in the general two player (M, N, m, n)-scenario, the number of independent NSCs is

$$\#NSC = M \cdot (N - 1) \cdot (m-1) + N \cdot (M - 1) \cdot (n-1), \tag{2.23}$$

where the -1 accounts for the normalisation condition.

All this gets a little bit more complicated, when we want to find all independent NSCs in

a three player scenario. On the one hand, we will have NSCs, ensuring that the statistics of two of the players are independent of the remaining one. On the other hand we will have NSCs, ensuring that the statistics of one player is independent of the other two players input. Lets do the two player case first. For example the NSCs for Alice and Bob with respect to Charlie, are formally given by

$$P(a,b|x,y,z) = P(a,b|x,y,z') \quad \forall a \in A, x \in X, b \in B, y \in Y, z, z' \in Z. \tag{2.24}$$

This case is quite similar to the two party case, if we think of Alice and Bob as one party with more inputs and outputs, namely all the pairs (a,b) and (x,y). We just have to remember, that we have now two normalisation conditions, one for Alice $\sum_a P(a|x) = 1$ and one for $\sum_b P(b|x) = 1$. Therefore to find all the constraints, we have to go through all the a, but one, all the b, but one, all the x, all the y and finally through the input pairs $(z \neq z')$ of Charlie in the same way as in the two player case, such that by chaining them together, we will find that the remaining NSCs are redundant. So the number of NSCs, denoted $\#NSC_{AB;C}$, for the statistics Alice and Bob being independent of Charlie in a general $(MNLmnl)$-scenario are

$$\#NSC_{AB;C} = M \cdot N \cdot (m-1) \cdot (n-1) \cdot (l-1), \tag{2.25}$$

where again the -1 account for the normalisation conditions for Alice and Bob respectively. The case for all other players is analogously and is derived by just substituting one players for another.

Now we want to do the single player case. We will use Alice as an example, the NSC condition will then formally look the following

$$P(a|x,y,z) = P(a|x,y',z') \quad \forall a \in A, x \in X, y, y' \in Y, z, z' \in Z. \tag{2.26}$$

If we want to find all the independent NSCs in this case, we have to go through all the x and again all the a, but one, because of the normalisation condition. Whats a bit more tricky now, is to find all the quadruples (y,z,y',z') with $(y,z) \neq (y',z')$, while still avoiding any redundancies. One way of doing this, is by thinking of Bob and Charlie as one party with $N \cdot L$ inputs, namely the pairs (y,z). If we order these pairs somehow from 0 to $N \cdot L$, we can run the same argument, as we have done it in the two player case. We just go through all the input pairs $(0,1),(1,2),\ldots,(N \cdot L - 2, N \cdot L - 1)$, which are now pairs of pairs and again by the argument of chaining we see that all other NSCs are then redundant. Lets do a small example, take a scenario, where Bob and Charlie have 3 inputs. In the table 2.4 below we see how we can order all the input pairs and straight forwardly get the necessary NSCs from it.

(y , z)	$NSC(y, z, y', z')$
(0 , 0)	$NSC(0, 0, 0, 1)$
(0 , 1)	$NSC(0, 1, 0, 2)$
(0 , 2)	$NSC(0, 2, 1, 0)$
(1 , 0)	$NSC(1, 0, 1, 1)$
(1 , 1)	$NSC(1, 1, 1, 2)$
(1 , 2)	$NSC(1, 2, 2, 0)$
(2 , 0)	$NSC(2, 0, 2, 1)$
(2 , 1)	$NSC(2, 1, 2, 2)$
(2 , 2)	

Table 2.1.: Ordered input pairs of Bob and Charlie (left column), NSC chaining one pair with the next pair (right column). Note: that The last $NSC(2, 2, 0, 0)$ is left out, because it already follows from chaining all the others together

Keeping this in mind, we can also compute the number of this NSCs easily,

$$\#NSC_{A;BC} = M \cdot (m-1) \cdot (l \cdot n - 1). \tag{2.27}$$

Finally we can compute the total number of NSCs in a three player scenario by adding all up, that is

$$\begin{aligned} \#NSC_{Total} =& \#NSC_{AB;C} + \#NSC_{BC;A} + \#NSC_{AC;B} \\ &+ \#NSC_{A;BC} + \#NSC_{B;AC} + \#NSC_{C;AB} \end{aligned} \tag{2.28}$$

or as a function of M, N, L, m, n, l using 2.27 and 2.25, we can write

$$\begin{aligned} \#NSC_{Total} =& (M \cdot N + M \cdot L + N \cdot L)(m-1)(n-1)(l-1) \\ &+ M(m-1)(l \cdot n - 1) + N(n-1)(m \cdot l - 1) + L(l-1)(m \cdot n - 1) \end{aligned} \tag{2.29}$$

So for example in the 222222-scenario, we have

$$\#NSC_{AB;C} = \#NSC_{BC;A} = \#NSC_{AC;B} = 2 \cdot 2 \cdot (2-1) \cdot (2-1) \cdot (2-1) = 4 \tag{2.30}$$

and

$$\#NSC_{A;BC} = \#NSC_{B;AC} = \#NSC_{C;AB} = 2 \cdot (2-1) \cdot (2 \cdot 2 - 1) = 6, \tag{2.31}$$

leading to a total of

$$\#NSC_{Total} = 4 + 4 + 4 + 6 + 6 + 6 = 30 \tag{2.32}$$

NSCs. Because these numbers escalate quite quickly and impossible to handle by a human being (the 222333-scenario has already 126 independent NSCs), we provide the program `TriNS.jl` A.6, which derives all independent NCSs for an arbitrary three party scenario using the method explained above. The NSCs stored in form of an $M \times N \times L \times m \times n \times l$-array, which is convenient with all the other programs provided in this thesis.

2.5. Locality

The next restriction we need is locality and it is where the main focus of this thesis lies. As discussed in the introduction, locality should capture the notion, that space-like separated events do not effect each other. For a Bell scenario this means, that nothing in one players lab should depend on anything of another players lab. The best depiction of this comes from John Bell himself [9]

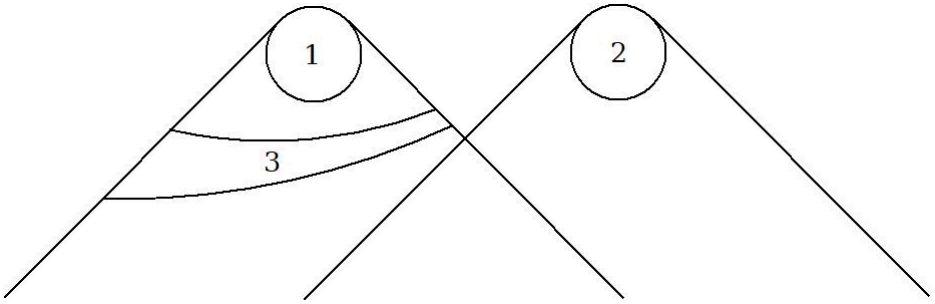

Figure 2.1.: Two space-time space-like seperated regions 1 and 2 with its past light cones. Region 3 cuts off region 1 and has a complete description of its past

We can think of the labs of the two players as region 1 and 2. Now from a local theory, we expect, that any events in region 1 can only depend on events from it's own causal past, i.e. events laying in its own past light cone. But then a region like 3, should contain all necessary information to predict any event in region 1 [1]. Note, that any information in region two lies either in the common past of region 1 and 2 or doesn't lie in the causal past of region 1 at all. In the first case the information will be also contained in region 3 and is therefore redundant for making any prediction about region 1 and in the second case it can only contain information about space-like separated events and is therefore irrelevant, if the world can be described locally. Either way no information from region 2 would change any predictions a player makes about events in region 1. In our Bell scenario the only relevant information in Bob's Lab (region 2) are the input y and the output b. Therefore in any local theory the measurement outcome in Alice's Lab cant depend on them, i.e.

$$P_\lambda(a|bxy) = P_\lambda(a|x) \tag{2.33}$$

[1] At least probabilisticly, if the theory, we use is not deterministic

and the same is vice versa true for Bob. This means that the joint probability of our Bell scenario will separate, that is

$$\begin{aligned} P_\lambda(ab|xy) &= P_\lambda(a|bxy)P_\lambda(b|xy) \\ &= P_\lambda(a|x)P_\lambda(b|y), \end{aligned} \tag{2.34}$$

which gives us the definition of locality. Therefore the only correlation between the outputs solely depends on the process λ, to which the players agreed on before in their common past. More generally, if we allow the players to switch between processes, we have that a local process can be written as

$$P(ab|xy) = \int d\lambda Q(\lambda) P_\lambda(a|x) P_\lambda(b|y), \tag{2.35}$$

where $Q(\lambda)$ is a probability distribution, as seen before.

2.6. The local polytope

One of the great wisdoms in physics and mathematics, if one has a set, it is almost always useful to look at the geometry of that set. Non-local-physic is no exception to that. In fact a lot of the great discoveries in this field, where made from a geometrical point of view. Although for this work we will only need the most basic geometrical insight, namely that the set of local behaviours, or the local set L for short, is a convex polytope. A polytope can be defined as compact convex set with finitely many extreme points. An extreme point of a convex set is a point, that can be written as a convex mixture of other points of the set. Let S be a convex set, then $p \in S$ is an extreme point, if $\forall p_1, p_2 \in S$ and $\forall \lambda \in (0, 1)$

$$p = \lambda p_1 + (1 - \lambda) p_2 \;\Rightarrow\; p_1 = p_2 = p. \tag{2.36}$$

Now, L is convex, because given to local processes $\mathcal{P}_1, \mathcal{P}_2 \in L$, one can always realise the convex mixture of these processes

$$\mathcal{P} = p\mathcal{P}_1 + (1 - p)\mathcal{P}_2 \tag{2.37}$$

by just sampling in a fraction of p rounds from $\mathcal{P}_1$ and in the other rounds from $\mathcal{P}_2$ and because by linearity $\mathcal{P}$ will again lead to a local behaviour. It immediately follows, that L is compact, because probabilities are bounded. Finally, that the number of extreme points is finite, follows from Fine's theorem, which we will see, when we talk about local deterministic processes in the next section. By a similar arguments, it can be shown, that the NS-set is also a polytope. Interestingly, it was proven, that the quantum set is not a polytope. This is not really important for this thesis, but it shows how geometry can give us insight into non-local physics.

2.7. Deterministic Process

A special kind of local process is a deterministic (D) processes, where the probabilities $P(ab|xy)$ are each either 0 or 1. We can define such a deterministic process

$$P_\lambda^D(ab|xy) = \delta_{f(x,y,\lambda)=a}\delta_{g(x,y,\lambda)=b} \tag{2.38}$$

by using a generalisation of the Kronecker delta

$$\delta_{f(x)=a} := \begin{cases} 1 & \text{if} \quad f(x) = a \\ 0 & \text{if} \quad \text{otherwise.} \end{cases} \tag{2.39}$$

Although these processes in general can be signaling, as long as we allow $f(x, y, \lambda)$ to depend on some other players input, only the local deterministic processes, will be important for us. Local deterministic (LD) processes are exactly those processes, in which the functions f and g only depend on the same players input and the strategy λ, that is

$$P_\lambda^{LD} := \delta_{f(x,\lambda)=a}\delta_{g(y,\lambda)=b}. \tag{2.40}$$

Note, that the functions f and g are maps from the inputs to the outputs, which are both finite sets. So there are finitely many of these functions. The functions map each input to one of the possible outputs. The number of these # functions is then the # outputs to the power of the #inputs. So in the two player case, we have

$$\#P^{LD} = m^M \cdot n^N \tag{2.41}$$

local deterministic processes. The important thing about LD-processes is now, that for all we want to now about the local processes in given scenario, it is enough to analyse only LD-processes and as there are finitely many we can run through all of them. To see this, first notice that every deterministic process is either signaling or local. We can proof this by assuming a deterministic process $f = f(x, y, \lambda)$, which explicitly depends on the other players input, say y and assume further that the no-signaling condition to hold, then we need

$$\delta_{f(x,y,\lambda)=a} = \delta_{f(x,y',\lambda)=a} \qquad \forall y, y', \tag{2.42}$$

and therefore we also need

$$f(x, y, \lambda) = f(x, y', \lambda) \qquad \forall y, y', \tag{2.43}$$

but this can only hold, if f is independent of y, which contradicts the assumption. So within the no-signaling set, all deterministic processes are local. The second observation we need is known as Fine's Theorem. A proof of it can be found Scarani's book [8] or in this paper [10] Fine's Theorem states, that a behaviour is local if and only if it can be modelled using a convex mixture of LD-processes, i.e.

$$P^L(ab|xy) = \sum_{i=1}^{m^M}\sum_{j=1}^{n^N} q_{ij}\delta_{f_i(x)=a}\delta_{g_j(y)=b}, \tag{2.44}$$

where the sums run over all possible LD-processes $\lambda = (i, j)$ and we require that $\sum_{ij} q_{ij} = 1$, so it is indeed a valid convex mixture. So for the record we have, that all the deterministic processes with in the no-signaling set are local, we have that there are finitely many of these LD-processes and we have that all local behaviours can be modelled by a convex mixture of LD-processes. From the last point it immediately follows, that only LD-processes can be extreme points of the local polytope. On the other hand to see, that all LD-processes are indeed extreme points, assume a LD-processes $\mathcal{P}_{LD}$, that is not extreme. Then $\mathcal{P}_{LD}$ can be written as convex mixture

$$\mathcal{P}_{LD} = p\mathcal{P}_{LD1} + (1-p)\mathcal{P}_{LD2} \tag{2.45}$$

of two other LD-processes with $\mathcal{P}_{LD1} \neq \mathcal{P}_{LD2}$, which means there is at least one input pair (x, y), such that $P_{LD1}(x, y) \neq P_{LD2}(x, y)$. But then for this input $\mathcal{P}_{LD}$ will give $P_{LD1}(x, y)$ in p fractions of rounds and $P_{LD2}(x, y)$ in $(1-p)$ fractions of rounds. Therefore $\mathcal{P}_{LD}$ is not deterministic, which is against the assumption. So we have that a local process is extreme, if and only if it is a LD-process. Therefore to find the local bound for a linear Bell inequality, it is enough to inspect all LD-processes, which can easily be checked by a computer. Note that in principle a Bell inequality does not have to be linear, but obviously linear inequalities are much easier to analyse. In this work we will focus only on linear Bell inequalities and refer to them just as Bell inequalities. For an example see the program `TriLB.jl`A.2 finds the local bound for an a Bell inequality in a $nml222$-scenario by a straightforward brute force method.

3. Bell Inequalities

As we mentioned in the introduction, to test non-locality in a Bell scenario, we use certain inequalities, called Bell inequalities, which separates some non-local behaviours from all local behaviours. They are given as a sum of probabilities $P(ab|xy)$ weighed by some coefficients $M_{xy}^{ab} \in \mathbb{R}$, which has to be smaller or equal to some bound $\mathcal{K} \in \mathbb{R}$, i.e.

$$\sum_{abxy} M_{xy}^{ab} P(ab|xy) \leq \mathcal{K}, \tag{3.1}$$

or for short we will just write

$$\langle M, P \rangle \leq \mathcal{K}, \tag{3.2}$$

using the definition for the inner product

$$\langle M, P \rangle = \sum_{abxy} M_{xy}^{ab} P(ab|xy). \tag{3.3}$$

The probabilities refer to the measurement outcome x, y given the input a, b. The weights are just some cleverly tweaked numbers, so indeed the inequality indeed separates some non-local behaviours from all the local ones. Ideally the numbers are choosen such that, the inequalitiy represents a cut along the facet of the local polytope. In this case a Bell inequality is called tight and the bound $\mathcal{K}$ is called the local bound $\mathcal{L}(\mathcal{M})$. Note that some times people will refer to these inequalities, also if they are given with their quantum bounds, or so called Tsirleson bound, $\mathcal{S}(\mathcal{M})$. The inequality separates quantum behaviours from even stronger non-local behaviours. Technically, these inequalities are no Bell inequalities and should be referred as quantum Bell inequalities, if there is a possibility of confusion. The most prominent example of a Bell inequality is the CHSH inequality. It is a tight inequality of the (2222)-scenario, with the following weights,

$$(M_{xy}^{ab}) = \left(\begin{array}{cc|cc} M_{0000} & M_{0100} & M_{0001} & M_{0101} \\ M_{1000} & M_{1100} & M_{1001} & M_{1101} \\ \hline M_{0010} & M_{0110} & M_{0011} & M_{0111} \\ M_{1010} & M_{1110} & M_{1011} & M_{1111} \end{array}\right) = \left(\begin{array}{cc|cc} 1 & -1 & 1 & -1 \\ -1 & 1 & -1 & 1 \\ \hline 1 & -1 & -1 & 1 \\ -1 & 1 & 1 & -1 \end{array}\right) \tag{3.4}$$

leading to the inequality

$$\begin{aligned} &P(00|00) - P(01|00) - P(10|00) + P(11|00) \\ &+P(00|01) - P(01|01) - P(10|01) + P(11|01) \\ &+P(00|10) - P(01|10) + P(10|10) - P(11|10) \\ &-P(00|11) + P(01|11) + P(10|11) - P(11|11) \leq 2, \end{aligned} \tag{3.5}$$

with the local bound of 2. Of course this notation is cumbersome and that's one of the reasons why people have come up with more concise ways to write the them, which we will see in the Chapter on Correlators 6 and the Collins-Gisin 7 representation. For now we want understand these inequalities a bit more in general and focus on the symmetries they have.

3.1. Symmetries of Bell Inequalities

There are two kinds of transformations we are interested in. The first kind of transformations is about relabelling the inputs, outputs or player and can be seen as a symmetry transformation of the local polytope. For example if we relabel the outputs of Alice from $(0, 1)$ to $(1, 0)$, you get a new valid inequality, which describes technically a different facet of the local polytope, given the inequality was tight, but basically its the same inequality. You haven't really changed much, other than the name tags on your measurement apparatus[1]. Therefore these symmetries reduce the number of facets of the polytope, we have to check, which is nice, because even mildly complicated scenarios can already have hundreds or even thousands of facets[2]. The other kind of transformations change the weights of the inequality, such that the inequality remains unchanged on the set of no-signaling condition, but it gives you a better or worse version of the inequality, thus the advantage of a quantum strategy compared to a local one changes for the better or worse[3]. To find the best weights for a given inequality is the content of this thesis.

3.1.1. Symmetries of Bell Inequalities

By construction a Bell inequality has certain kinds of symmetries, i.e. by relabelling the inputs, outputs or player, we get a new version of the inequality of the same inequality, which only differs in the labels, we have assigned to the inputs, outputs and players.

The first of these transformation concerns inputs x, y. Remember that, there is nothing special about how we name our inputs, therefore by relabelling the inputs one has to again get a valid Bell inequality.
For example in the CHSH-scenario if we relabel Alice's inputs, i.e. $0 \mapsto 1$ and $1 \mapsto 0$, the inequality is still valid and represents still the CHSH-scenario and the same is true for Bob.
So in a general scenario with M and N inputs any relabelling, that is just a permutation of the inputs σ_N and σ_M can be used to get different version of the same inequality, i.e.

$$M_{xy}^{ab} \longmapsto M_{\sigma_N(x)\sigma_M(y)}^{ab}. \tag{3.6}$$

After relabelling the inputs, of course one can relabel the outputs. So for a scenario with n and m outputs, yet again we get a different version of the same inequality by relabelling

[1] Think of smart phones. Technically you get a new phone if you write 15 Pro on it instead of 14 plus an round the edges, but come on ...
[2] e.g. the 3322-scenario has already 684 facets
[3] We will see what that exactly means in section 3.1.2

the outputs by some permutations σ_n and σ_m, i.e.

$$M_{xy}^{ab} \longmapsto M_{xy}^{\sigma_n(a)\sigma_m(b)}. \tag{3.7}$$

Finally if the players have the same number of inputs and outputs, therefore $N = M$ and $n = m$, we can also permute the players, to get yet another version of the inequality, i.e.

$$M_{xy}^{ab} \longmapsto M_{yx}^{ba}. \tag{3.8}$$

All this generalises trivially to any number of players, specifically to the three player case. Important to remember is that, although all these different versions will have the same properties, e.g. local- and quantum bounds, technically they are not the same inequalities in the sense that they represent different cuts in the space of all local behaviours or even more so different facets of the local polytope, if the inequality is tight.

3.1.2. Scaling, Translation and adding NS-conditions

The other kind of transformations, changes the weights of an inequality. Although this will change some the properties of the inequality, e.g. it's local and quantum bound, it can be done in such a way that this new version will still represents the same inequality. The first two transformations of this kind are scaling and translation. Because of the linearity of Bell inequalities, multiplying with a constant, will just scale everything up by that constant and because of the affine properties we can also add a constant. Finally we can also add a no-signaling condition. This is possible, because the set of no-signaling behaviours only represents a subset of zero measure within the set of all behaviours and a Bell inequality is a cut through these sets, defining a half space on both of them. Now, because of the zero measure of the NS-set, we can vary the cut, such that it will depict the same half space on it. To see this (Fig 3.1.2), assume for simplicity, that the set of all behaviours is a square and the NS-set will be a line (grey) through this square. Then two Bell inequalities (red and blue) can cut through the set, such that they depict the same half space on the NS-set, although the half space on the whole set will change (shaded regions). Therefore on the set of NS-behaviours we can consider these Bell inequalities as two versions of the same inequality, but as long as we leave the half space of the NS-set untouched, we are fine. There are three kinds of transformation, that does this and they are at the core of this thesis, because we will use them to optimise our Bell inequalities.

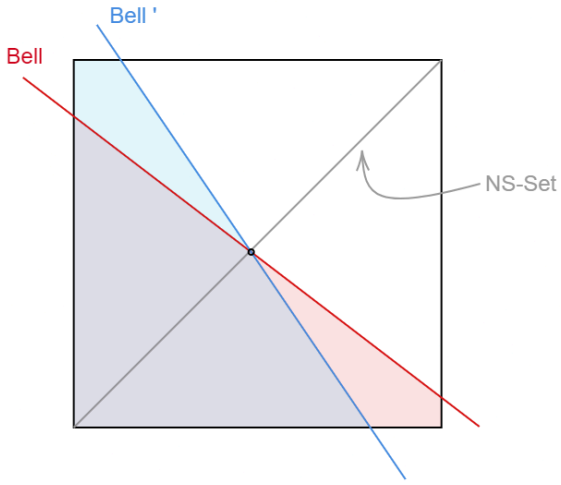

Figure 3.1.: Bell Cut

The first transformation is just a multiplication by a scalar c, i.e. $M \mapsto c * M$. This will leave the Bell inequality intact, when we multiply the bound by the very same scalar. We can see this, if we multiply our inequality with c on both sides and use the linearity of the scalar product, i.e.

$$\langle M, P\rangle \leq \mathcal{K} \tag{3.9}$$

$$c * \langle M, P\rangle \leq c * \mathcal{K} \tag{3.10}$$

$$\langle cM, P\rangle \leq c\mathcal{K}. \tag{3.11}$$

The second transformation is adding a constant. This will also add the constant to the bound. To see that the Bell inequality is left intact by this transformation, we can add the constant on both sides

$$\langle M, P\rangle + c \leq \mathcal{K} + c. \tag{3.12}$$

Then we use the normalisation conditions $\sum_{ab} P(ab|xy) = 1 \quad \forall x, y$, to rewrite c in terms of these normalisation conditions, i.e.

$$c = \sum_{xy} c_{xy} = \sum_{xy}\sum_{ab}(c_{xy} * P(ab|xy)). \tag{3.13}$$

Note, that we are free to choose how to distribute c over the c_{xy}. Finally, we can absorb the c_{xy}s into the brackets using again the linearity of the scalar product, thus

$$\langle M, P\rangle + \sum_{xy} c_{xy} \leq \mathcal{K} + c \tag{3.14}$$

$$\langle M + c_{xy}, P\rangle \leq \mathcal{K} + c \tag{3.15}$$

Here a small example might be helpful. Lets add 4 to our beloved CHSH-inequality

$$\langle M_{CHSH}, P\rangle + 4 \leq 2 + 4 \tag{3.16}$$

We can distribute now 4, on the normalisation condition of two different input sets, e.g. $c_{00} = 1$ and $c_{11} = 3$. Absorbing these into the scalar product and adding it to M_{CHSH}. How the new version $\tilde{M}_{CHSH}$ will look like, becomes clear if we use matrix notation for

$$\left(\begin{array}{cc|cc} 1 & -1 & 1 & -1 \\ -1 & 1 & -1 & 1 \\ \hline 1 & -1 & -1 & 1 \\ -1 & 1 & 1 & -1 \end{array}\right) + \left(\begin{array}{cc|cc} 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ \hline 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{array}\right) + \left(\begin{array}{cc|cc} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ \hline 0 & 0 & 3 & 3 \\ 0 & 0 & 3 & 3 \end{array}\right) = \left(\begin{array}{cc|cc} 2 & 0 & 1 & -1 \\ 0 & 2 & -1 & 1 \\ \hline 1 & -1 & 2 & 4 \\ -1 & 1 & 4 & 2 \end{array}\right). \tag{3.17}$$

We see, that our new version of M_{CHSH} looks pretty messed and not as clean and symmetric than the original version. This is a good illustration of the problem, when we find these inequalities out in the wild with various methods, it is not at all clear, if two inequalities are actually the same. It is hard to see that $\tilde{M}_{CHSH}$, we have created has anything to do with the actual CHSH-inequality and we can mess it up even further by adding a NS-condition. This is the third transformation we can use to tinker with the weights. This leaves the bound unchanged, because it amounts to adding a zero on

both side of the inequality. Remember the definition $P(a|x,y) - P(a|x,y') = 0$ for the NS-condition, if we add that to our inequality, we get

$$\langle M, P\rangle + P(a|x,y) - P(a|x,y') \leq K + 0 \tag{3.18}$$

Again we can use the linearity of the scalar product to absorb everything into the brackets. To get a feeling for the result, we will as always use CHSH as a simple example. For the NS-condition we choose $a = 1$, $x = 0$, $y = 0$ and $y' = 1$. To calculate the new $\tilde{M}_{CHSH}$, we can again use matrix notation

$$\left(\begin{array}{cc|cc} 1 & -1 & 1 & -1 \\ -1 & 1 & -1 & 1 \\ \hline 1 & -1 & -1 & 1 \\ -1 & 1 & 1 & -1 \end{array}\right) + \left(\begin{array}{cc|cc} 0 & 0 & 0 & 0 \\ 1 & 1 & -1 & -1 \\ \hline 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{array}\right) = \left(\begin{array}{cc|cc} 1 & 0 & 1 & -1 \\ 1 & 2 & -2 & 0 \\ \hline 1 & -1 & -1 & 1 \\ -1 & 1 & 1 & -1 \end{array}\right). \tag{3.19}$$

Like above, we get a messed up version of the original M_{CHSH}. In the next chapters we will use this transformations in two ways. First we will transform our Bell inequalities into a special form, known as a Bell Game, which brings several advantages. Secondly, we will use these transformation to find the best versions of these games, after all we want a version, that is easy and not the one that just looks pretty. Going back to our example, it could well be, that one of the messed up version might be the better one.

4. Non-Local Games

There are several issues, when dealing with Bell inequalities. As mentioned above it is not clear, how to choose the best version of a given inequality or even more so to choose the best of several different inequalities. They simply lack a good measure of comparison and weights and bounds are some what arbitrary numbers. Using the transformation from above, i can easily make quantum mechanics go over 9000 in the CHSH scenario, but is this really better than $2\sqrt{2}$? Last but not least Bell inequality as such, can make the impression of very abstract statistical tools. Its hard to explain, why adding $2 * P(00|01)$ to $P(11|00)$ and subtracting a bit of $P(10|10)$ .... and so forth, has anything to do with proving non-local physics.

These issues with can be neatly resolved by treating a Bell scenario as sort of a game. This idea was introduced in [1] and uses the strong similarity between a Bell scenario and a multi-prover interactive-proof model [11], which is used to examine problems cryptography and computer complexity . The picture is the following, there are two cooperating players called Alice and Bob trying to win in a game, which is played over several rounds. In each round a referee draws questions from a set, which we will assume to be finite [1], according to some probability distribution and sends them to the players. Without communicating the players have to send back an answer, also chosen from a finite set, to the referee. The referee then evaluates the answers given the questions following some predicate. This sounds an awful lot like a Bell scenario and it is. But as we will see, it solves various issues , when dealing with Bell inequalities.

The formal definition of a non-local game is as followed, a non-local game G is a Bell functional

$$\langle G, P\rangle \leq \omega_l(G), \tag{4.1}$$

where $\omega_l(G)$ is the local bound of the game and G has the form

$$G_{xy}^{ab} := \mu(x,y) * V(ab|xy). \tag{4.2}$$

$\mu(x,y)$ is the probability distribution from which the questions are drawn, and therefore has to be semi-positive and normalised, i.e. $\mu(x,y) \in [0,1] \quad \forall x,y$ and $\sum_{xy} \mu(x,y) = 1$. $V(ab|xy)$ is the predicate and are the 'rules of the game'. It gives the probability, that the referee accepts the answers a, b given the question x, y, therefore we need $V(ab|xy) \in [0,1]$ and in the special case of a deterministic game we need $V(ab|xy) \in \{0,1\}$. If these conditions are met, then the local bound $\omega_l(G)$ will be between 0 and 1 and gives the

[1] Although in general one can also assume the set to be infinite

probability, that the players win in a single round of the game, following the best local strategy. Likewise, if they follow the best quantum strategy, the probability to win will be given by the quantum bound $\omega_q(G)$.

This immediately solves some of the problems, when we are dealing with Bell inequalities. Not only is the notion of a simple question-and-answer game much more intuitive for a human to understand, than just summing up some statistics, but also the local- and quantum- bounds have a well defined meaning by giving the winning chances in a single round of the game, instead of being a completely arbitrary number. Therefore it is easy to see how much of an advantage a quantum strategy has over a local strategy by just comparing $\omega_l(G)$ and $\omega_q(G)$. The larger the gap

$$\chi(G) := \omega_q(G) - \omega_l(G) \tag{4.3}$$

the bigger the advantage. This also gives a good measure, when comparing different inequalities. The larger the gap the stronger the inequality.
It can be shown [2] that every Bell functional M is a non-local game if and only if M is positive,

$$M_{xy}^{ab} \geq 0 \quad \forall a, b, x, y \tag{4.4}$$

and normalised

$$\sum_{xy} \max_{ab} M_{xy}^{ab} \leq 1. \tag{4.5}$$

To see that this is sufficient, one can define

$$\mu(x, y) := \frac{\max_{ab} M_{xy}^{ab}}{\sum_{x'y'} \max_{a'b'} M_{x'y'}^{a'b'}} \tag{4.6}$$

and

$$V(ab|xy) := \frac{1}{\mu(x, y)} M_{xy}^{ab}. \tag{4.7}$$

The other direction follows trivial from the definition of a non-local game (4.14) by using the properties of μ and V.
For the case that a Bell functional doesn't have those properties (4.4) and (4.5), we can use the equivalent transformations from Chapter (3.1.2) to get the right form.

As an example we will use the CHSH inequality 3.5 again

$$\begin{aligned} &P(00|00) - P(01|00) - P(10|00) + P(11|00) \\ &+ P(00|01) - P(01|01) - P(10|01) + P(11|01) \\ &+ P(00|10) - P(01|10) + P(10|10) - P(11|10) \\ &- P(00|11) + P(01|11) + P(10|11) - P(11|11) \leq 2. \end{aligned} \tag{4.8}$$

which in this form is neither positive (-) nor normalised (2) and transform it into a non-local game. First by adding a constants, to make it positive and then by multiplying

with another constant to normalise it. If we add 4 distributed evenly, we get

$$\left(\begin{array}{cc|cc} 1 & -1 & 1 & -1 \\ -1 & 1 & -1 & 1 \\ \hline 1 & -1 & -1 & 1 \\ -1 & 1 & 1 & -1 \end{array}\right) + \left(\begin{array}{cc|cc} 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ \hline 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \end{array}\right) = \left(\begin{array}{cc|cc} 2 & 0 & 2 & 0 \\ 0 & 2 & 0 & 2 \\ \hline 2 & 0 & 0 & 2 \\ 0 & 2 & 2 & 0 \end{array}\right) \tag{4.9}$$

and hence the new inequality

$$\begin{aligned} &2P(00|00) + 2P(11|00) + 2P(00|01) + 2P(11|01) \\ &+2P(00|10) + 2P(10|10) + 2P(01|11) + 2P(10|11) \leq 6, \end{aligned} \tag{4.10}$$

which is positive and has a local bound of 6. Now to normalise the Bell inequality, we can multiply by $\frac{1}{8}$ to get

$$\begin{aligned} &\frac{1}{4}\,[\; P(00|00) + P(11|00) + P(00|01) + P(11|01) \\ &+P(00|10) + P(10|10) + P(01|11) + P(10|11)\;] \leq \frac{3}{4}. \end{aligned} \tag{4.11}$$

and we are done, CHSH the Game! Note, that we could have also added and multiplied different constants. Similar to the general version of the Bell inequality, the game is not unique. There are good and bad versions. Here we have chosen the constants, such that we get a particular nice version of the CHSH-game[2]. We can easily identify the input distribution $\mu(x, y) = \frac{1}{4}$. It is just a constant and the predicate

$$V(ab|xy) = \left(\begin{array}{cc|cc} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ \hline 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 \end{array}\right) \tag{4.12}$$

is deterministic. The players will win the game in 3out of 4 rounds or 75% of the time, following the best local strategy. The quantum bound can be calculated and amounts to $\omega_q(G) = \frac{1}{4}(2 + \sqrt{2}) \approx 0.85355$. Therefore following the best quantum strategy the players will win about 85% of the time, which amounts to a rough gap of

$$\chi(G) := \omega_q(G) - \omega_l(G) \approx 0.85 - 0.75 = 0.10. \tag{4.13}$$

We immediately see, that these numbers make literally sense. We can even neatly summarise the rules of the game. In the CHSH-Game, the two players, Alice and Bob, are separated and each round the referee sends one of two query $x, y \in \{0, 1\}$ to each of the players. The question for each player is chosen randomly and independently with equal probability $\mu = \frac{1}{4}$. Now, to win a round of the game Alice and Bob have to give the same answer $a = b$, if one or both of the queries x and y were 0, but give different answers $a \neq b$, if x and y were both 1.

[2]One can also show , that this is indeed the best version of the game, but we will come to that later, when talking about the optimisation algorithm

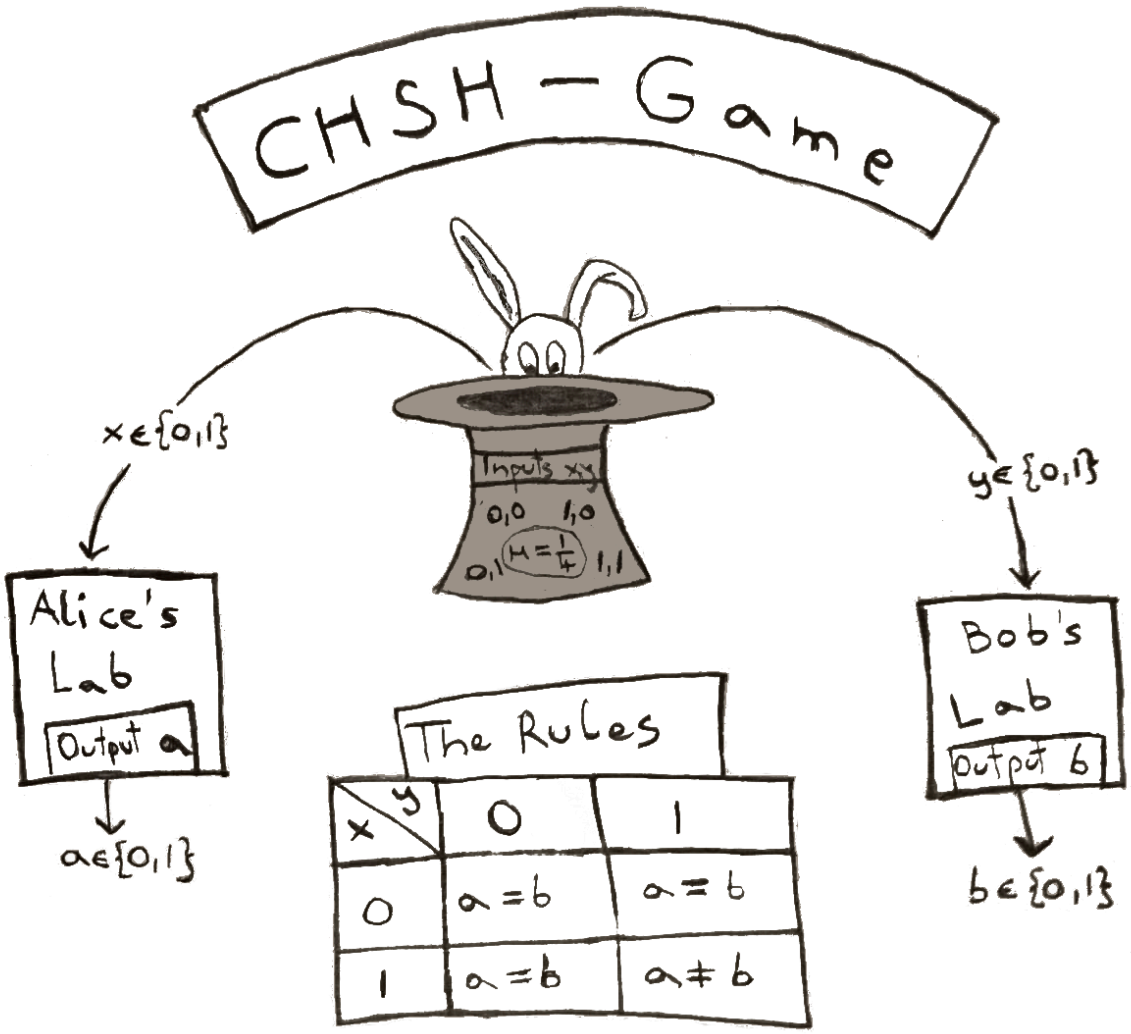

Figure 4.1.: The CHSH-Game

4.1. Three Player Games

Of course, for our purposes, we need to generalise this formalism to the three player case. Fortunately, because of the linearity of the formalism, the generalisation is straightforward and we don't have to worry. We can just define a three player game using a 6-Tensor instead of a 4-tensor, i.e.

$$G_{xyz}^{abc} := \mu(x,y,z) * V(abc|xyz), \tag{4.14}$$

again $\mu(x,y,z)$ is the probability distribution, from which the question are drawn, which now of course also depends on the third players input z and as before we need it to be semi-positive and normalised. Of course these 6-Tensor become quite huge even for a small number of in- and outputs. But there is a particular nice one, which is not to cumbersome to write it out as an example. This game is called GHZ [4] and it is the most prominent three player game. The game is a 222222-scenario. The game use only 4 $(000, 110, 101, 011)$ of the 8 possible inputs and those with equal probability, therefore it reads

$$\mu(x,y,z) = \begin{cases} \frac{1}{4} & (xyz) = \{000, 110, 101, 011\} \\ 0 & \text{else} \end{cases} \tag{4.15}$$

For the predicate, we only need the slices of the tensor for the used inputs. We can set all entries in other slices to 0, they are not used anyway. Now, to win the game the outputs of Alice,Bob and Charlie has to add to an even number, if the input was (000) and to an odd number in all the other cases. Therefore we have the following predicate:

$$V(.,.,0,0,0,0) = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \qquad V(.,.,1,0,0,0) = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \tag{4.16}$$

for the input (000) and for all the other inputs, e.g. 110, we have

$$V(.,.,0,1,1,0) = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \qquad V(.,.,1,1,1,0) = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \tag{4.17}$$

That being said, we are ready to see how we can optimise these games in the next chapter.

5. Linear Optimisation

Now to find the best version of a non-local game, we have to define a value we want to optimise, e.g. maximise and we have to give a procedure using the transformation from Section 3.1.2, which eventually converges on the this maximum. Fortunately, we will see, that we can formulate our problem in form of a linear program, which is a very well known optimisation method. We can therefore fall back on a broad number of algorithms specifically developed for these kinds of problems.

5.1. Linear Program (LP)

Here we will give a super short introduction to linear programming, such that the steps in our optimisation procedure become as clear as possible. A linear program is an optimisation method for linear functions subject to linear constraints. In the standard form, a linear program consist of three parts. A linear or affine function to maximise

$$f : \mathbb{R}^n \to \mathbb{R}, \qquad x \mapsto \langle x, c\rangle + a \tag{5.1}$$

with $c, a \in \mathbb{R}^n$. Some linear constraints of the form

$$A \cdot x \leq b, \tag{5.2}$$

where A is a $m \times n$-matrix, $b \in \mathbb{R}^m$ and m is the number of constraints. Finally, we also have positivist constraint for x, that is

$$x \geq 0. \tag{5.3}$$

Typical examples come from economics. Lets imagine a company, that produces two products x_1 and x_2 and the company will earn 3\$ and 2\$ from them respectively. To optimise the profit P, we need to maximise

$$P(x_1, x_2) = 3x_1 + 2x_2, \tag{5.4}$$

but the company will be constraint by their labour hours and raw materials available. Assume, the company has 20 labour hours to work with and 10 units of raw material. Assume further, that product x_1 needs 2 hours of labour and 4 units of material to fabricate and product x_2 needs 1 hour and 5 units of material. Then the company will have the following constraints

$$\begin{aligned} 2x_1 + \ \ x_2 &\leq 20 \\ 4x_1 + 5x_2 &\leq 60. \end{aligned} \tag{5.5}$$

Finally, there are two positivity constraints, because negative numbers of products can not be produced, that is

$$x_1, x_2 \geq 0. \tag{5.6}$$

The problem is now, how to choose x_1 and x_2, such that the company maximises its profit $P(x_1, x_2)$ under the given constraints. Because of its simplicity, the problem can be pictured in the following way 5.3: We see the positivity constraints in green and the

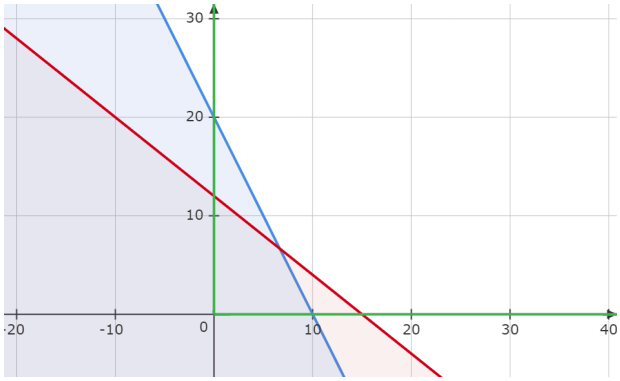

Figure 5.1.: Illustration of the LP problem

labour hour and raw material constraint in blue and red respectively. The enclosed region is a convex polytope and is called the feasible region. It is the set of all points, that satisfies all the constraints of the LP. If the set is empty, we call the problem unfeasible. If the set is unbounded, we call the problem unbounded. Of course, in higher dimension we will not be able to illustrate the problem and just see if there is a feasible region or not, but the feasible region will still be a convex polytope described by the hyper surfaces, that are the constraints of the problem. Because of the linearity of the function, the maxima and minima will always be realised in some corner of the polytope[1]. This should be intuitively clear, by thinking about the monotonicity of linear functions. If we start at any interior point of the feasible region, we can just go along a line. Then because of the monotonicity the function will increase in one direction and decrease in the other and we can follow along in the direction of increase until hitting an edge. At the edge, we can do the same and follow the edge in the direction of increase until we end up in a corner. At the corner there is either another edge along the function increases, which we can follow to another corner or we are stuck and found a maximum. It is also not hard to prove it, by the means of multi dimensional calculus, but we wont really need it. The optimisation will be done by one of the many LP solvers out there anyway. I just think its nice, to give an intuitive understanding of LP.

[1]This doesn't mean, that the extremes are exclusively realised in a corner, e.g. if $f = const$ all points will be extreme, but a corner will always do.

5.2. Optimising non-local games

Now, to formulate our problem as an LP, we need to ask ourselves, what is the best function to optimise for and what are the correct constraints. Note, that we want to optimise the weights of the non-local game not the strategy of the players. For the game to be optimal, we want the quantum advantage as big as possible. For that we will maximise the gap $\chi(G)$ and we will see, why this is better than to maximise the ratio $\frac{\omega_q(G)}{\omega_l(G)}$, as it is usually done in the literature. The constraints of our model will be given by the equivalent transformations from section 3.1.2, because this is the only way we can change the weights without altering the inequality within the NS-set. Finally, we will have also positivity constraint, because we are optimising the inequality as a non-local game.

5.3. Optimising for the gap χ

On his blog Mateus Araújo [12] makes the argument, that the gap is good to optimise, because it is a good measure of the strength of a non-local game. In this section we will explain this argument as it is crucial to this work. Additionally we will give a detailed proof, that the argument really holds true.

First we want to remind ourselves, what we actually want to show by a non-local game. In the end we want to use the data D from an experiment, corresponding to a non-local game, to show that a non-local model H_N is preferable over a local model H_L. Let $P(D|T_l)$ be the probability for the data given a local model and $P(D|T_n)$ be the probability for the data given a non-local model, then the ratio

$$K = \frac{P(D|T_l)}{P(D|T_n)}, \tag{5.7}$$

known as Bayes factor, gives a measure of how strongly we should believe in one or the other model. If the number is close to 0, we prefer the model in the denominator. If the number is way above 1, we prefer the model in the numerator. A number around 1, should be considered as inconclusive. The data in our case will only consist of the number of rounds n and the number of victories v within these rounds. The probability for such an outcome is simply described by a binomial distribution and the probability of v victories in n rounds is just

$$P(n, v) = \binom{n}{v} p^v (1-p)^{n-v}, \tag{5.8}$$

where p is the probability of winning a round. Now, for our local- and quantum model the probability of winning a round assuming the best strategy is just ω_l and ω_q. Therefore we derive at a Bayes factor

$$K(n, v) = \frac{\omega_l^v (1-\omega_l)^{n-v}}{\omega_q^v (1-\omega_q)^{n-v}}, \tag{5.9}$$

where we have already cancelled out the binomial coefficients. Now, lets assume that our experiment runs against the number of victories quantum mechanics predicts[2]. Then $v = n\omega_q$ and our Bayes factor becomes

$$K(n,v) = \frac{\omega_l^{n\omega_q}(1-\omega_l)^{n-n\omega_q}}{\omega_q^{n\omega_q}(1-\omega_q)^{n-n\omega_q}} \tag{5.10}$$

Now by taking the logarithm, we can factor out n

$$K(n,v) = \exp\left(n\log\left(\frac{\omega_l^{\omega_q}(1-\omega_l)^{1-\omega_q}}{\omega_q^{\omega_q}(1-\omega_q)^{1-\omega_q}}\right)\right) \tag{5.11}$$

and by using the basic logarithm identities even further

$$K(n,v) = \exp\left(n\omega_q\log\left(\frac{\omega_l}{\omega_q}\right) + n(1-\omega_q)\log\left(\frac{(1-\omega_l)}{(1-\omega_q)}\right)\right), \tag{5.12}$$

we derive a at a well known expression, called the Kullback–Leibler divergence. The Kullback–Leibler divergence, also known as relative entropy, which is a measure how different one probability distribution is from another and it is defined as

$$D_{\mathrm{KL}}(P \parallel Q) = \sum_{x\in\mathcal{X}} P(x)\ \log\left(\frac{P(x)}{Q(x)}\right) \tag{5.13}$$

where $P(x)$ and $Q(x)$ are the probability distribution on the same sample space $\mathcal{X}$. Therefore Bayes factor for n rounds with $v = n\omega_q$ can simply be given in terms of the relative entropy, i.e.

$$K_n = \exp(-nD(\omega_q||\omega_l)). \tag{5.14}$$

So, maximising Bayes factor is equivalent with maximising relative entropy, which in principle could be done directly. But the relative entropy is not linear and therefore hard to optimise. To make things easier, we will linearise our problem by maximising the gap χ instead. To see that this is indeed sufficient, note that χ gives a lower bound for the entropy.We can see this by showing, that

$$D(\omega_q||\omega_l) \geq -\chi\log(1-\chi). \tag{5.15}$$

Now, to see this, we can use the definition of $D(.||.)$ 5.13 on the left hand side and write out the sum, i.e.

$$D(\omega_q||\omega_l) = -\omega_q\log\left(\frac{\omega_l}{\omega_q}\right) - (1-\omega_q)\log\left(\frac{(1-\omega_l)}{(1-\omega_q)}\right). \tag{5.16}$$

Then note, that in general

$$\min_X[f(x)+g(x)] \geq \min_x[f(x)] + \min_x[g(x)], \tag{5.17}$$

so we can look at the terms one by one and to compare the left and the right hand side, we can do a coordinate change. The obvious coordinate change to incorporate χ on the

[2]Otherwise we would have to discard quantum mechanics and ω_q

left hand side is

$$\chi = \omega_q - \omega_l \quad \text{and} \quad \phi = \omega_q + \omega_l \tag{5.18}$$

with $\chi \in [0,1]$ and $\phi \in [0,2]$. Aside from that, we obviously have $\phi \geq \chi$ and $\phi \leq 2 - \chi$ This leads to

$$\frac{1}{2}(\phi+\chi)\log\left(\frac{\phi+\chi}{\phi-\chi}\right) + (1-\frac{1}{2}(\phi+\chi))\log\left(\frac{(1-\frac{1}{2}(\phi+\chi))}{(1-\frac{1}{2}(\phi-\chi))}\right) \geq -\chi\log(1-\chi) \tag{5.19}$$

Notice, that we want to derive a lower in χ independent of ϕ. Therefore,we have to minimise $D(\frac{1}{2}(\phi+\chi)||\frac{1}{2}(\phi-\chi))$ with respect to ϕ. Now, the derivatives of the terms, we want to minimise don't vanish in the region of interest. Hence their minimum is reached at the boundary. The region has three boundaries defined by $\chi \geq 0$ and $\phi \geq \chi$ and $\phi \leq 2 - \chi$, we have to look at.

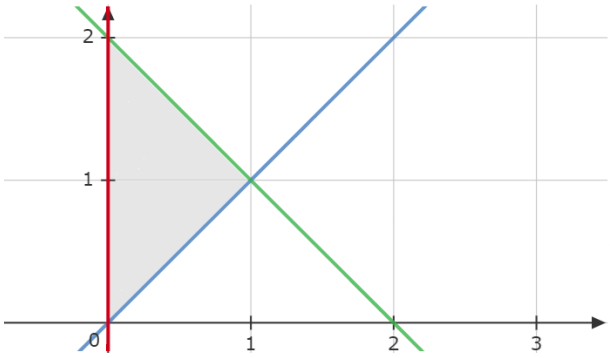

Figure 5.2.: The Region defined by the three boundary conditions, $\chi \geq 0$ (red), $\phi \geq \chi$ (blue) and $\phi \leq 2 - \chi$ (green)

At the boundary $\chi \geq 0$ the inequality both terms of the right hand side go to 0. At the boundary $\phi \geq \chi$ the first term goes to

$$\lim_{\phi\to\chi} \frac{1}{2}(\phi+\chi)\log\left(\frac{\phi+\chi}{\phi-\chi}\right) = \infty \tag{5.20}$$

and the second term goes to

$$\lim_{\phi\to\chi} (1-\frac{1}{2}(\phi+\chi))\log\left(\frac{(1-\frac{1}{2}(\phi+\chi))}{(1-\frac{1}{2}(\phi-\chi))}\right) = (1-\chi)\log(1-\chi), \tag{5.21}$$

Finally at the boundary $\phi \leq 2 - \chi$, the first term becomes

$$\lim_{\phi\to 2-\chi} \frac{1}{2}(\phi+\chi)\log\left(\frac{\phi+\chi}{\phi-\chi}\right) = -\log(1-\chi) \tag{5.22}$$

and the second term goes to 0. Now we can add up the minima of both terms, that it adding 5.22 and 5.21 and indeed we see that,

$$D(\omega_q||\omega_l) \geq -\chi \log(1-\chi) \tag{5.23}$$

. We see that the gap χ really gives a lower bound to the entropy and it is therefore sufficient to optimise it to get a strong Bell inequality. In the same vain, we can show that the ratio, which is also used to optimise Bell inequalities gives a lower bound to the entropy,

$$D(\omega_q||\omega_l) \leq \log\Big(\frac{\omega_q}{\omega_l}\Big) \tag{5.24}$$

and not sufficient, but only necessary for a strong Bell inequality. The reason, why people often optimise is, because a big ratio gives high noise tolerance. But as we have seen it doesn't necessary lead to a good Bayesian factor. Therefore we will optimise the gap.

5.4. Finding the right constraints

As mentioned before and explained in section 3.1.2, the only transformations, we are allowed to do without changing the Bell inequality, are scaling, translation and adding NS-conditions. To find the right way to tackle this problem, let's look at scaling and translation first. Because adding and multiplying a constant is somewhat easy and we can find the largest gap by hand. For an intuitive approach, lets start with a messed up version of CHSH

$$(M_{xy}^{ab}) = \left(\begin{array}{cc|cc} 4 & 0 & 2 & -2 \\ 0 & 4 & -2 & 2 \\ \hline 3 & -1 & 4 & 8 \\ -1 & 3 & 8 & 4 \end{array}\right) \tag{5.25}$$

and assume, that we know that version of the CHSH-game 4.12, we have introduced is the strongest version. How we can get there?
Well, to get a non-local game, we need the entries to be positive and intuitively it seems nice to have a lot of 0. To accomplish we can subtract the minimum for each input pair (x, y), i.e.

$$(M_{xy}^{ab} - \min_{a,b} M_{xy}^{ab}) = \left(\begin{array}{cc|cc} 4 & 0 & 2 & -2 \\ 0 & 4 & -2 & 2 \\ \hline 3 & -1 & 4 & 8 \\ -1 & 3 & 8 & 4 \end{array}\right) - \left(\begin{array}{cc|cc} 0 & 0 & -2 & -2 \\ 0 & 0 & -2 & -2 \\ \hline -1 & -1 & 4 & 4 \\ -1 & -1 & 4 & 4 \end{array}\right) = \left(\begin{array}{cc|cc} 4+0 & 0 & 2+2 & 0 \\ 0 & 4+0 & 0 & 2+2 \\ \hline 3+1 & 0 & 0 & 8-4 \\ 0 & 3+1 & 8-4 & 0 \end{array}\right) \tag{5.26}$$

Now, that all entries are positive, we need to normalise it. For that, imagine two 'perfect' player using a signaling strategy. Those players will make always the maximum amount of points given each input. So in our example they will score $(4+0)+(2+2)+(3+1)+(8-4) = 16$ points. Therefore to normalise, we need to divide by 16, which is nothing else than the difference between sum of the maximums $(4+2+3+8=17)$ and the sum of the minimums $(0-2-1+4=1)$ of our original messed up CHSH . To put it formally, given

a Bell inequality in its tensor form M^{ab}_{xy}, we can always transform it into a non-local game G using the following formula

$$G^{ab}_{xy} := \frac{1}{\beta - \alpha}(M^{ab}_{xy} - \alpha_{xy}), \tag{5.27}$$

where α and α_{xy} is defined over the minimum and β over the maximums of M, that is

$$\alpha_{xy} = \min_{a,b} M^{ab}_{xy}, \qquad \alpha = \sum_{x,y} \alpha_{xy} \tag{5.28}$$

$$\beta_{xy} = \max_{a,b} M^{ab}_{xy}, \qquad \beta = \sum_{x,y} \beta_{xy}. \tag{5.29}$$

The neat thing about this formula is, that it already gives us the non-local game with the largest gap, which by linearity is

$$\chi(G) = \frac{\mathcal{Q}(M) - \mathcal{L}(M)}{\beta - \alpha}. \tag{5.30}$$

To see this we can introduce additional dummy variables $\tilde{\alpha}_{xy}$, $\tilde{\alpha} := \sum_{xy} \tilde{\alpha}_{xy}$ and $\tilde{\beta}$ and write

$$\tilde{G}^{ab}_{xy} := \frac{1}{\beta + \tilde{\beta} - \alpha + \tilde{\alpha}}(M^{ab}_{xy} - \alpha_{xy} + \tilde{\alpha}_{xy}). \tag{5.31}$$

Because there are no constraints on $\tilde{\alpha}_{xy}$, $\tilde{\alpha} := \sum_{xy} \tilde{\alpha}_{xy}$ and $\tilde{\beta}$, this is the most general form of a non-local game, we can get from a Bell tensor M. But of course, for $\tilde{G}$ to be a valid non-local game, we need it to be positive and properly normalised. But to require positivity means to require $\tilde{\alpha}_{xy} \geq 0$, because

$$\forall x, y \qquad \min_{a,b}(M^{ab}_{xy} - \alpha_{xy} + \tilde{\alpha}_{xy}) = \tilde{\alpha}_{xy}, \tag{5.32}$$

which also implies $\tilde{\alpha} \geq 0$. Further, the 'perfect' player will score

$$\sum_{x,y} \max_{a,b} \tilde{G}^{ab}_{xy} = \frac{1}{\beta + \tilde{\beta} - \alpha + \tilde{\alpha}}(\beta - \alpha + \tilde{\alpha}). \tag{5.33}$$

So for G to be properly normalised, we need $\tilde{\beta} \geq 0$. But if we need $\tilde{\beta} \geq 0$ and $\tilde{\alpha} \geq 0$, then the gap

$$\tilde{\chi}(G) = \frac{\mathcal{Q}(M) - \mathcal{L}(M)}{\beta + \tilde{\beta} - \alpha + \tilde{\alpha}} \tag{5.34}$$

is maximised, if $\tilde{\beta} = \tilde{\alpha} = 0$. So indeed our formula 5.27 finds the maximum gap, if we only use scaling and translation. This brings us to the NS-signaling conditions. As we have seen in section 3.1.2, adding a NS-condition to a Bell inequality will change neither change the local nor the quantum bound. But what it will change is α and β. So, to find

the 'true' largest gap, that is NS-conditions included, we need to find $\tilde{M}$ by adding the right amount $\gamma_i \in \mathbb{R}$ of all the NSC_i in a scenario to M, i.e.

$$\tilde{M} = M + \sum_i \gamma_i NSC_i, \tag{5.35}$$

such that $\beta - \alpha$ becomes a minimum and therefore the gap

$$\chi(G) = \frac{\mathcal{Q}(M) - \mathcal{L}(M)}{\beta - \alpha}. \tag{5.36}$$

becomes a maximum. So, using the definitions for α and β from above 5.28 5.29, the objective of our linear program will be

$$\min_{\gamma_i}(\beta - \alpha) = \min_{\gamma_i} \sum_{x,y} \left(\max_{ab}(\tilde{M}^{ab}_{xy}) - \min_{ab}(\tilde{M}^{ab}_{xy}) \right). \tag{5.37}$$

The constraints are implicitly given by the max(.) and $-\min(.)$ functions we want to minimise. The max(.) in itself is not a linear function, but the maximum of several linear functions $f(x) = \max_i\{\langle a_i, x\rangle + b\}$ is always convex and therefore defines a convex region 5.4 on which we can define a linear function t. This function can then be minimised using an LP, where the set of linear functions $f(x)$ become the constraints.

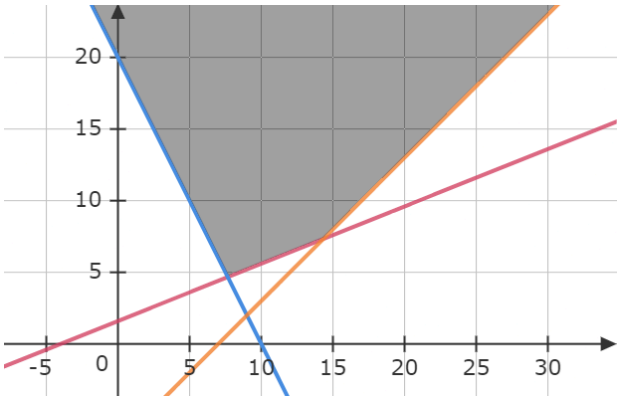

Figure 5.3.: Convex region (grey) defined by the maximum of several linear functions

Therefore, if we have a function $f(x) = \max_i\{\langle a_i, x\rangle + b\}$ and want to find the $\min_x \max_i\{\langle a_i, x\rangle + b\}$, which is not an LP by itself. We rather find

$$\min_{x,t} t \qquad \text{subject to} \qquad t \geq a_i \cdot x + b \qquad \forall i, \tag{5.38}$$

where we have introduced an additional variable t. One can imagine x and t running against each other eventually settle on $\min_x \max_i\{\langle a_i, x\rangle + b\}$[3]. Note, that minimising $-\min(.)$ comes down to the same as minimising a max(.). So also the second term is not an issue. Only min(min(.)) and max(max(.)) runs into trouble, we fortunately not have to deal with.

[3]In game theory this famous Nash equilibrium

The translation of our problem into this form is straightforward. Our final LP reads

$$\min_{\alpha_{xyz},\beta_{xyz},\gamma_i} \qquad \sum_{x,y,z} \beta_{xyz} - \alpha_{xyz} = \beta - \alpha \tag{5.39}$$

subject to

$$\tilde{M}^{abc}_{xyz} \leq \beta_{xyz} \qquad \forall a, b, c \tag{5.40}$$

$$\tilde{M}^{abc}_{xyz} \geq \alpha_{xyz} \qquad \forall a, b, c \tag{5.41}$$

Note, that we have add a third player by inserting the in- and outputs for Charlie c and z. The program `TriLP.jl`, which finds the optimum for this LP is part of this thesis and is given in `Julia` in the Appendix A.3.

6. Correlators

Correlators are a convenient and concise way of writing Bell inequalities in a two output scenario. Originally a correlator in a (NM22)-scenario was defined

$$\langle A_x B_y \rangle := \sum_{ab} abP(ab|xy) \tag{6.1}$$

using the labeling $a, b \in \{-1, 1\}$. But in general it is more convenient to use the following definition

$$\langle A_x B_y \rangle := P(a = b|xy) - P(a \neq b|xy), \tag{6.2}$$

which is independently of the labeling. In this notation for example the CHSH-inequality has the short and compact form

$$\langle A_0 B_0 \rangle + \langle A_0 B_1 \rangle + \langle A_1 B_0 \rangle - \langle A_1 B_1 \rangle \leq 2. \tag{6.3}$$

Further by using the definition introduced in [13] and [14], this notation can be generalised to an arbitrary number of n parties. Because the general definition for n parties is a bit cumbersome, we will only give the definition for three parties, because don't need any correlators for more than three parties. That being said, the defintion of the three partiy correlator in a (NML222)-scenario is defined by

$$\langle A_x B_y C_z \rangle := \sum_{a,b,c} (-1)^{a+b+c} P(abc|xyz). \tag{6.4}$$

using the labelling $a, b, c \in 0, 1$. In the same way also a single party correlator is defined,

$$\langle A_x \rangle = \sum_{a} (-1)^{a} P(a|x), \tag{6.5}$$

where the the marginal is of course defined by a sum over all other players outputs, e.g. $P(a|x) := \sum_b P(ab|xy)$. Any n party Bell inequalities in a two output scenario can be written, using correlators for 1 to n parties [3]. Also in the two party scenario, we need single party correlators to address the most general cases. Note that in the literature people often write just E_{xy} for the correlator $\langle A_x B_y \rangle$, but to generalises this notation to three or more parties, one would have to fiddle extra indices in to specify to whom to whom a given two party correlator refers to, e.g. Alice and Bob $\langle A_x B_y \rangle$ or Bob and Charlie $\langle B_y C_z \rangle$. so its more clearly to just write big letters for Alice, Bob and Charlie. As an example here is inequality #4 from [3]:

$$\begin{aligned} 0 \leq 2 - 2\langle A_0 \rangle - \langle B_0 C_0 \rangle + \langle A_0 B_0 C_0 \rangle - \langle B_1 C_0 \rangle + \langle A_0 B_1 C_0 \rangle \\ - \langle B_0 C_1 \rangle + \langle A_0 B_0 C_1 \rangle + \langle B_1 C_1 \rangle - \langle A_0 B_1 C_1 \rangle, \end{aligned} \tag{6.6}$$

which given a closer look is just CHSH for Bob and Charlie. Although for our purposes it will be necessary to translate a Bell inequality from its correlator form back to its full probability form to address all the weights individually, when we optimise the inequality. So it is important to notice, that this translation comes with a small ambiguity, which has to be addressed. Lets see this in an example first. In a (3322)-scenario a single party correlator for Alice with input $x = 1$ might appear in a Bell inequality with a weighting parameter $m_a = 2$, i.e.

$$m_a\langle A_1\rangle = 2\sum_a (-1)^a P(a|1), \tag{6.7}$$

Now if we write out the sum over a and b

$$m_a\langle A_1\rangle = 2\Big(P(00|1y) + P(01|1y) - P(10|1y) - P(11|1y)\Big), \tag{6.8}$$

we see that Bob's input y is not fixed. This is fine in so far, that the no-signaling constraint is implied and $P(a|x) = \sum_b P(ab|xy)$, but when we go from the correlator to the full probability form, we have to write the parameter m_a somewhere. As $P(a|x)$ doesn't depend on y, one solution would be to just choose one of three inputs 0,1,2 for y. Another way is to choose a convex combination of all the inputs an distribute the weight over all of them. We will go for the later an make the obvious choice by distributing the weight equally over all inputs. This gives the formula and the code a bit more symmetry, which makes it easier to read. Thus in our example we get

$$\begin{aligned} m_a\langle A_1\rangle = \frac{2}{3}\Big(&P(00|10) + P(01|10) - P(10|10) - P(11|10) \\ &+P(00|11) + P(01|11) - P(10|11) - P(11|11) \\ &+P(00|12) + P(01|12) - P(10|12) - P(11|12)\Big) \end{aligned} \tag{6.9}$$

and in a general scenario (NM22)-scenario, we define

$$\langle A_x\rangle = \frac{1}{M}\sum_y\sum_{ab}(-1)^a P(ab|xy). \tag{6.10}$$

The same goes for the three party scenario (MNL222). For the two party correlator, e.g. for Alice and Bob, we will distribute evenly over the outputs of Charlie, i.e.

$$\langle A_x B_y\rangle = \frac{1}{L}\sum_z\sum_{ab}(-1)^{a+b} P(ab|xyz) \tag{6.11}$$

and mutatis mutandi for all other combinations. For the one party correlator we distribute evenly over all combinations of inputs of the two other players, e.g. for Alice

$$\langle A_x\rangle = \frac{1}{M*L}\sum_{yz}\sum_a (-1)^a P(a|xyz) \tag{6.12}$$

and again similarly for the others. Of course this translation process becomes quite tedious even for a simple scenario, thus the program `TriCRP.jl` is provided A.4, which translates an arbitrary two output three party inequality from correlator into full probability form. One has just to give the weights of the correlators in a $(M+1)\times(N+1)\times(L+1)$-array E and the program returns the full probability form in a $2\times 2\times 2\times n\times m\times l$-array, where the $+1$ gives space for the marginals, which we have choose to be in the first entries of the array (See Figure 6.1). The entry $E[1,1,1]$ is not used by any correlator so we can use it to store the local bound of the inequality. For example in a (3,3,3,2,2,2)-scenario, the input for the program has to be a $4\times 4\times 4$-array and the entries will be ordered as seen in the picture below, where we have left out the angle brackets for a cleaner look

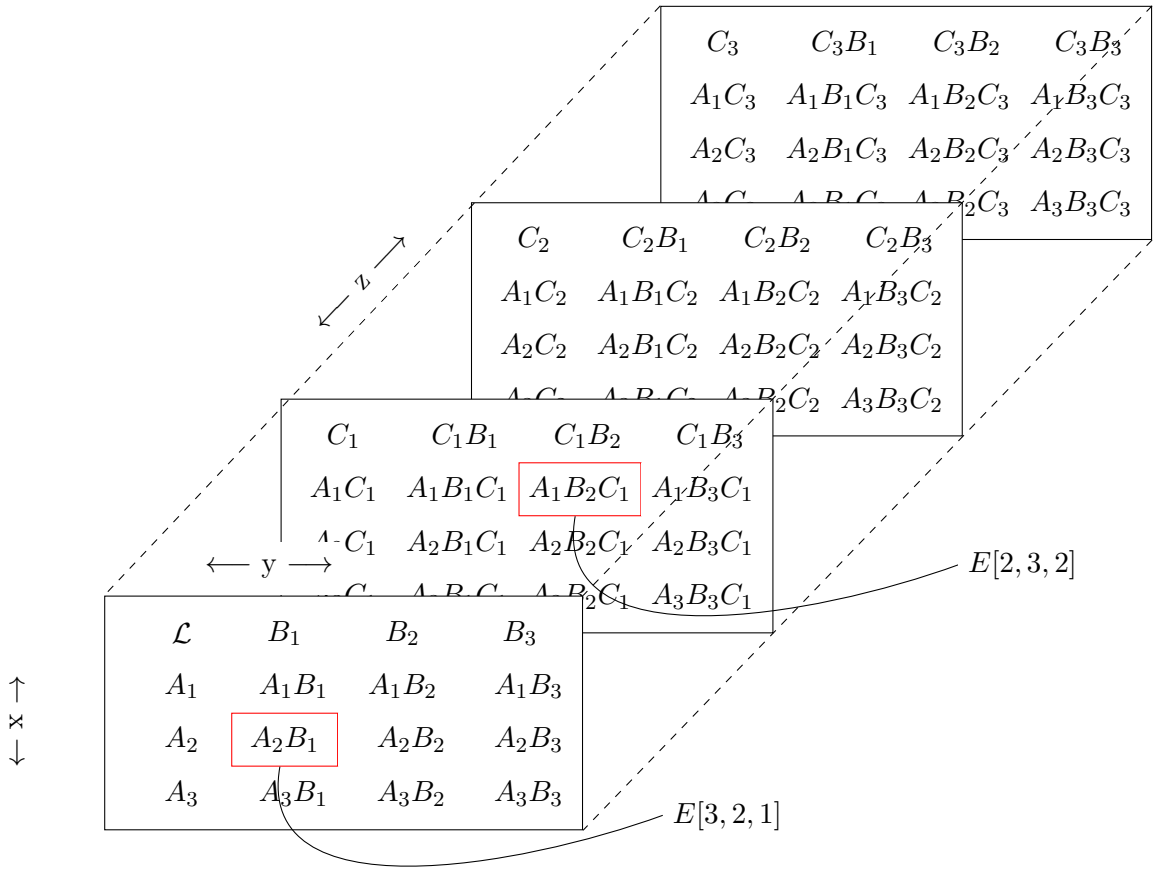

Figure 6.1.: Correlator array for `TriCRP.jl`

Finally to do a concrete example here is the GHZ-inequality in correlator form, which is inequality #2 on Śliwa's List [3],

$$0 \leq 2 - \langle A_1 B_1 C_1 \rangle - \langle A_2 B_2 C_1 \rangle - \langle A_2 B_1 C_2 \rangle + \langle A_1 B_2 C_2 \rangle. \tag{6.13}$$

It is a (2,2,2,2,2,2)-scenario, so the input will be a $3 \times 3 \times 3$-array and the local bound for GHZ in this form is a lower bound equal to -2 and is stored at $E[1,1,1]$. In `Julia` the array will look like this

```
3x3x3 Array{Int64, 3}:
[:, :, 1] =
 -2  0  0
  0  0  0
  0  0  0

[:, :, 2] =
 0  0  0
 0  1  0
 0  0  1

[:, :, 3] =
 0  0   0
 0  0  -1
 0  1   0
```

and the output of the program will be the full probability form and is given in a huge $2 \times 2 \times 2 \times 2 \times 2 \times 2$-array with $2^6 = 64$ entries addressing all the conditional probabilities $P(a, b, c|x, y, z)$ directly.

7. Collins-Gisin Representation

For handling more general scenarios a popular notation was introduced by D.Collins and N.Gisin. It describes an arbitrarily complex no-signaling behaviour with the minimum amount of parameters. Of course if one is given all the parameters to uniquely describe a general no-signaling behaviour, then also a general Bell inequality can be expressed in quite the same way, where instead of the probabilities describing the behaviour, we give the weights describing a Bell inequality.

To see how the Collins-Gisin notation works, it is easiest to do an example. Lets say we are given all the probabilities $P(ab|xy)$ and their marginals, i.e all $P(a|x) := \sum_b P(ab|x)$ and $P(b|y) := \sum_a P(ab|y)$, of a behavior of a 2222-scenario. Note that because of the no-signaling assumption in the Collins-Gisin notation, we don't have to bother with the other players input in the marginals. A conclusive table of this behavior might look like this:

	P(b=1|y=1)	P(b=2|y=1)	P(b=1|y=2)	P(b=2|y=2)
P(a=1|x=1)	P(11|11)	P(12|11)	P(11|12)	P(12|12)
P(a=2|x=1)	P(21|11)	P(22|11)	P(21|12)	P(22|12)
P(a=1|x=2)	P(11|21)	P(12|21)	P(11|22)	P(12|22)
P(a=2|x=2)	P(21|21)	P(22|21)	P(21|22)	P(22|22)

Now we can use the NC, e.g. using $P(1|1) + P(2|1) = 1$, to spare one of the marginals for each x and y .

	P(b=1|y=1)	P(b=2|y=1)	P(b=1|y=2)	P(b=2|y=2)
P(a=1|x=1)	P(11|11)	P(12|11)	P(11|12)	P(12|12)
P(a=2|x=1)	P(21|11)	P(22|11)	P(21|12)	P(22|12)
P(a=1|x=2)	P(11|21)	P(12|21)	P(11|22)	P(12|22)
P(a=2|x=2)	P(21|21)	P(22|21)	P(21|22)	P(22|22)

Table 7.1.: The marginals, that can be spared highlighted in red

Next we can use the marginals and NSC, e.g.

$$P(11|11) + P(12|11) = P(a = 1|x = 1) = P(11|12) + P(12|12),$$

to eliminate even more entries.

	P(b=1|y=1)	P(b=2|y=1)	P(b=1|y=2)	P(b=2|y=2)
P(a=1|x=1)	P(11|11)	P(12|11)	P(11|12)	P(12|12)
P(a=2|x=1)	P(21|11)	P(22|11)	P(21|12)	P(22|12)
P(a=1|x=2)	P(11|21)	P(12|21)	P(11|22)	P(12|22)
P(a=2|x=2)	P(21|21)	P(22|21)	P(21|22)	P(22|22)

Table 7.2.: The conditional probabilities, that can be spared highlighted in blue

Note that we could of course spare different marginals instead, but it is a convenient convention to eliminate the ones with the highest output a,b. So cleaning all up all the redundancies, we are left with

	P(b=1|y=1)	P(b=1|y=2)
P(a=1|x=1)	P(11|11)	P(11|12)
P(a=1|x=2)	P(11|21)	P(11|22)

Table 7.3.: Collins-Gisin Notation for NS-Behaviour for a 2222-scenario

which is the Collins-Gisin notation for a no-signaling behaviour in a 2222-scenario. So we can leave out all entries with $a = 2$ or $b = 2$, as they can be recovered by using the marginals and NCs and NSCs and are therefore redundant. If we raise the number of inputs and go from a 2222- to a 2233-scenario, we have to add rows and columns for $a = 2$ and $b = 2$ respectively.

	P(b=1|y=1)	P(b=2|y=1)	P(b=1|y=2)	P(b=2|y=2)
P(a=1|x=1)	P(11|11)	P(12|11)	P(11|12)	P(12|12)
P(a=2|x=1)	P(21|11)	P(22|11)	P(21|12)	P(22|12)
P(a=1|x=2)	P(11|21)	P(12|21)	P(11|22)	P(12|22)
P(a=2|x=2)	P(21|21)	P(22|21)	P(21|22)	P(22|22)

Table 7.4.: Collins-Gisin Notation for NS-Behaviour for a 2233-scenario

Now the entries for $a = 3$ and $b = 3$ are redundant and can be spared. If we raise the number of outputs and go from a 2233- to a 3333-scenario, we have to add $a - 1$ additional rows and $b - 1$ columns

	P(b=1|y=1)	P(b=2|y=1)	P(b=1|y=2)	P(b=2|y=2)	P(b=1|y=3)	P(b=2|y=3)
P(a=1|x=1)	P(11|11)	P(12|11)	P(11|12)	P(12|12)	P(11|13)	P(12|13)
P(a=2|x=1)	P(21|11)	P(22|11)	P(21|12)	P(22|12)	P(21|13)	P(22|13)
P(a=1|x=2)	P(11|21)	P(12|21)	P(11|22)	P(12|22)	P(11|23)	P(12|23)
P(a=2|x=2)	P(21|21)	P(22|21)	P(21|22)	P(22|22)	P(21|23)	P(22|23)
P(a=1|x=3)	P(11|31)	P(12|31)	P(11|32)	P(12|32)	P(11|33)	P(12|33)
P(a=2|x=3)	P(21|31)	P(22|31)	P(21|32)	P(22|32)	P(21|33)	P(22|33)

Table 7.5.: Collins-Gisin Notation for NS-Behaviour for a 3333-scenario

All this generalizes trivially to an arbitrary number of in- and outputs, such that this example should be sufficient to understand how this works and it is easy to to see that the number of parameters D_{NS} for a general NS-behavior or Bell inequality will be

$$D_{NS} = \underbrace{N * (n-1) + M * (m-1)}_{marginals} + \underbrace{N * (n-1) * M * (m-1)}_{Joint\ probabilities}. \tag{7.1}$$

Now to write a Bell inequality like that, one puts just the appropriate weights in the table instead of the probabilities of the behaviour. Of course the inequality will not have the same form as it might be in full probability notation or in correlator notation. For example the CHSH-equation is often given in the so called CH-form introduced by Clauser and Horne [15]

	-1	0
-1	1	1
0	1	-1

Table 7.6.: CH-form of the CHSH inequality in CG-notation

with a local bound $\mathcal{L}_{CH} = 0$, leading to the following inequality

$$\sum_{x,y=0}^{1} (-1)^{xy} P(0,0|xy) - P(a=0|x=0) - P(b=0|y=0) \leq 0. \tag{7.2}$$

Unfortunately as mentioned before, for our purposes we need the inequality in full probability notation, therefore we have to translate the Collins-Gisin notation into a full probability version. Concerning the marginals again we have to make a choice how to distribute the weight to the individual terms of the sum. Here again we decide to distribute evenly over all terms. So for a marginal $P(\tilde{a}|\tilde{x})$ and weight $m^{\tilde{a}}_{\tilde{x}}$, we get

$$m^{\tilde{a}}_{\tilde{x}} * P(\tilde{a}|\tilde{x}) = \frac{m^{\tilde{a}}_{\tilde{x}}}{N} \sum_{b,y} P(\tilde{a}b|\tilde{x}y) \tag{7.3}$$

and mutatis mutandi for all other players.

For example going back to our example of the CH-inequality, the weight for the marginal $P(a = 1|x = 1)$ is -1 to add this to the full probability tensor, we would distribute this the following way

$$(-1) * P(a = 1|x = 1) = -\frac{1}{2}\left[P(11|11) + P(12|11) + P(11|12) + P(12|12)\right]. \tag{7.4}$$

So adding the weights of the marginals is easy. Whats a bit involved is to find the entries in the full probability tensor corresponding to the entries in the Collins-Gisin table respectively.
To do that systematically its easiest to cut the table into separate arrays for the marginals $P(a|x)$, $P(b|y)$ and the conditional probabilities separately and add them to to an empty $m \times n \times M \times N$-array representing the full probability tensor.

		P(b=1|y=1)	P(b=2|y=1)	P(b=1|y=2)	P(b=2|y=2)
		$\longleftarrow j \longrightarrow$			
P(a=1|x=1)	$\uparrow$	P(11|11)	P(12|11)	P(11|12)	P(21|12)
P(a=2|x=1)	i	P(21|11)	P(22|11)	P(12|12)	P(22|12)
P(a=1|x=2)	$\downarrow$	P(11|21)	P(12|21)	P(11|22)	P(21|22)
P(a=2|x=2)		P(21|21)	P(22|21)	P(12|22)	P(22|22)

Table 7.7.: Collins-Gisin Notation for NS-Behaviour for a 2233-scenario

So the task, while running over i, j or both, is to deduce a, x and from i and b, y from j. For a and b, we can use the modulo function (a mod b), which gives the remainder of $\frac{a}{b}$ for $a, b \in \mathbb{N}$, then

$$a = [(i-1) \bmod (m-1)] + 1 \tag{7.5}$$

$$b = [(j-1) \bmod (n-1)] + 1 \tag{7.6}$$

where the -1 at i, j makes sure that the lowest entry has a remainder of 0 and the $+1$ at the end is just to follow the standard convention counting the entries of an array starting at 1, which is also the convention in `Julia`.
For the inputs it is easiest to use the ceiling function ($\lceil \frac{a}{b} \rceil$), which rounds the fraction up to the next whole number, then

$$x = \left\lceil \frac{i}{m-1} \right\rceil \tag{7.7}$$

$$y = \left\lceil \frac{j}{n-1} \right\rceil \tag{7.8}$$

and note that we have to divide by the number of outputs not inputs.

Sticking to our example in the form table 7, its easy to derive the full-probability form of the CH-equation

$$\underbrace{\left(\begin{array}{cc|cc} -\frac{1}{2} & -\frac{1}{2} & 0 & 0 \\ 0 & 0 & 0 & 0 \\ \hline -\frac{1}{2} & -\frac{1}{2} & 0 & 0 \\ 0 & 0 & 0 & 0 \end{array}\right)}_{P(a=1|x=1)} + \underbrace{\left(\begin{array}{cc|cc} -\frac{1}{2} & 0 & -\frac{1}{2} & 0 \\ -\frac{1}{2} & 0 & -\frac{1}{2} & 0 \\ \hline 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{array}\right)}_{P(b=1|y=1)} + \underbrace{\left(\begin{array}{cc|cc} 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \\ \hline 1 & 0 & -1 & 0 \\ 0 & 0 & 0 & 0 \end{array}\right)}_{P(ab|xy)} = \left(\begin{array}{cc|cc} 0 & -\frac{1}{2} & \frac{1}{2} & 0 \\ -\frac{1}{2} & 0 & -\frac{1}{2} & 0 \\ \hline \frac{1}{2} & -\frac{1}{2} & -1 & 0 \\ 0 & 0 & 0 & 0 \end{array}\right) \tag{7.9}$$

The generalisation to the three player scenario is straightforward. We can think of the Collins-Gisin table as three dimensional similarly as we did for with the three party correlator form. with the two player marginals on the sides and the one player marginals at the edges. For the two player marginals we distribute over all inputs of the remaining player, e.g. for Alice and Bob

$$m^{\tilde{a}\tilde{b}}_{\tilde{x}\tilde{y}} * P(\tilde{a}\tilde{b}|\tilde{x}\tilde{y}) = \frac{m^{\tilde{a}}_{\tilde{x}}}{L} \sum_{c,z} P(\tilde{a}\tilde{b}c|\tilde{x}\tilde{y}z) \tag{7.10}$$

and for the one player marginals, we distribute the weights over all input combinations of the remaining two players, e.g.

$$m^{\tilde{a}}_{\tilde{x}} * P(\tilde{a}|\tilde{x}) = \frac{m^{\tilde{a}}_{\tilde{x}}}{N * L} \sum_{b,c,y,z} P(\tilde{a}bc|\tilde{x}yz) \tag{7.11}$$

but even for moderately complex scenarios this becomes very tedious. Therefore this Thesis provides the program `TriCGP.jl` A.5 to translate an arbitrary three party scenario from its Collin-Gisin form into its full-probability form. The program simply takes in the Collin-Gisin form as a array and a vector `s=[m,n,l,M,N,L]` describing the scenario and spits out the full probability form arranged in a multi-dimensional $m \times n \times l \times M \times N \times L$-array.

8. Śliwa's List

Finally we can apply our results to some of the known three party equations out there. One of the best sources is the paper [3], which provides a complete list of all Bell inequalities for the $(2, 2, 2, 2, 2, 2)$-scenario up to symmetry transformations. We provide this list in a `Julia` usable form under the name `SliwList.jl` A.7, where we have included the quantum bounds, which we have taken from [16]. To translate this list from PDF to `Julia` was quite a tedious work and I hope to start the useful practice by providing such lists in computer readable form.

It is now straight forward to run `TriCRP.jl` over the whole list to translate it into its full-probability form and then run `TriLP.jl` to find the optimal non-local games for each of these inequalities and the parameters α and β. Now from the parameters α, β and the local and quantum bounds from `SliwList.jl`, we are able to compute the optimal gaps using,

$$\chi_G = \frac{\mathcal{Q}(M) - \mathcal{L}(M)}{\beta + \alpha}, \tag{8.1}$$

and by rearranging the equation, we can find the winning chances of the best local $\omega_l(G)$ and quantum strategy $\omega_q(G)$. This provides us with the following list 8.1. We see, that two inequalities (#1, #10) have a gap of 0, thus there is no quantum advantage. This is no surprise, because inequality #1 is one of the trivial facets, corresponding to the condition that probabilities are between 0 and 1. Any reasonable process has to obey it. Inequality #10 is the well-known GYNI inequality [17]. GYNI stands for "Guess Your Neighbors Input", which already states the rules of the game. The players win, if each can predict the input of the next player, closing in a circle so the last player has to predict the input of the first. But any prediction about another player's input, that is better than random guessing amounts to signaling, it is impossible for any non-signaling behaviour to get any advantage. The next inequality we should give a closer look is #4. This inequality turns out to be just CHSH, where one player just acts as a dummy. So, we see that CHSH has a gap $\chi_{CHSH} \sim 0.1036$, which we can use as a reasonable benchmark for all the other inequalities. Therefore, there are 8 inequalities (#2, #7, #8, #24, #26, #28, #33, #42) stronger than CHSH, marked with a +. Clearly the strongest of them all is inequality #2 with a gap of $\chi_{GHZ} = 0.2500$, which is the famous and well-studied GHZ inequality [4] [18] [19]. So, our strongest 'newcomer' is inequality #7 with a gap of $\chi_{G7} \sim 0.1333$, which we want to study a bit in detail in the next chapter.

Table 8.1.: Gaps χ_G, local bounds $\omega_q(G)$ and quantum bounds $\omega_q(G)$. The inequalities stronger, than CHSH are marked with a "+"

#	χ_G	$\omega_l(G)$	$\omega_q(G)$	Note
1	0	1	1	Trivial Facet
2	0.2500	0.7500	1	+GHZ
3	0.1036	0.7500	0.8536	CHSH
4	0.1036	0.7500	0.8536	
5	0.0885	0.7500	0.8385	
6	0.0828	0.8000	0.8828	
7	0.1333	0.7000	0.8333	+ Guess 2!
8	0.1111	0.6674	0.7785	+
9	0.0690	0.6661	0.7352	
10	0	0.8280	0.8280	GYNI
11	0.0690	0.6778	0.7468	
12	0.0690	0.7473	0.8163	
13	0.0690	0.6362	0.7053	
14	0.0690	0.6667	0.7357	
15	0.0833	0.8333	0.9167	
16	0.0939	0.6765	0.7704	
17	0.1034	0.7495	0.8529	
18	0.0731	0.8333	0.9064	
19	0.0743	0.8333	0.9076	
20	0.0888	0.6429	0.7316	
21	0.0764	0.6875	0.7639	
22	0.0848	0.7353	0.8201	
23	0.0238	0.7500	0.7738	
24	0.1050	0.7515	0.8565	+
25	0.0570	0.7397	0.7967	
26	0.1098	0.8000	0.9098	+
27	0.0611	0.7365	0.7976	
28	0.1137	0.7093	0.8230	+
29	0.0962	0.7187	0.8148	
30	0.0872	0.6842	0.7714	
31	0.0501	0.7410	0.7912	
32	0.0588	0.7455	0.8043	
33	0.1093	0.7898	0.8991	+
34	0.0612	0.6522	0.7134	
35	0.0464	0.7214	0.7677	
36	0.0945	0.6667	0.7612	
37	0.0872	0.6887	0.7759	
38	0.0857	0.6854	0.7711	
39	0.0860	0.7720	0.8580	
40	0.0521	0.7935	0.8456	
41	0.0936	0.7222	0.8158	
42	0.1113	0.7382	0.8496	+
43	0.0690	0.7432	0.8123	
44	0.0971	0.7031	0.8002	
45	0.0909	0.6829	0.7739	
46	0.0466	0.7668	0.8135	

9. Game 7

As we have pointed out in the last chapter discussing Śliwas-List, inequality # 7 is one of the more interesting ones, as it brings the largest gap besides GHZ. So lets give it a closer look and see, if we can "put some rules" to the game to get a human intuition for the game. The full array of game # 7 is given in A.1, the array is ordered such that the slices with the same inputs are in the same row and they vary in the output of Charlie. Within each slice the outputs of Alice vary down the columns and the outputs of Bob vary along the rows, e.g. the 0.3 in the listing below on the top right corner marked by the arrow refers to the outputs $(1, 2, 2)$ given the inputs $(1, 1, 1)$.

```
[:, :, 1, 1, 1, 1] =          [:, :, 2, 1, 1, 1] =
 0.3  0.0                      0.0  0.3  <-
 0.0  0.3                      0.3  0.0

```

The first thing that sticks out are the 0.3s in the input slices $(1, 1, 1)$. Note that all other entries are 0.1. Thinking in terms of a game, this means that the input set $(1, 1, 1)$ is chosen three times more often, than all the other input sets. Therefore we can deduce the distribution, from which the inputs are drawn. In matrix form it will look like this

$$\mu(x, y, 1) = \begin{bmatrix} 0.3 & 0.1 \\ 0.1 & 0.1 \end{bmatrix} \qquad \mu(x, y, 2) = \begin{bmatrix} 0.1 & 0.1 \\ 0.1 & 0.1 \end{bmatrix}$$

where each entry simply gives the probability, that a specific set of inputs is chosen. Further if we let alone the factor of 3, the slices for the input sets

$$(1, 1, 1), (1, 1, 2), (1, 2, 1), (2, 1, 1), (2, 2, 2) \tag{9.1}$$

look all the same, e.g.

```
[:, :, 1, 1, 2, 1] =          [:, :, 2, 1, 2, 1] =
 0.1  0.0                      0.0  0.1
 0.0  0.1                      0.1  0.0

```

Alice, Bob and Charlie win if they give the outputs

$$(1, 1, 1), (2, 2, 1), (1, 2, 2), (2, 1, 2). \tag{9.2}$$

Whereas for the remaining input sets

$$(2,2,1),(1,2,2),(2,1,2) \tag{9.3}$$

the slices are flipped, e.g.

```
[:, :, 1, 2, 2, 1] =          [:, :, 2, 2, 2, 1] =
 0.0  0.1                      0.1  0.0
 0.1  0.0                      0.0  0.1
```

Alice,Bob and Charlie win if they give the outputs

$$(1,2,1),(2,1,1),(1,1,2),(2,2,2). \tag{9.4}$$

By comparing the input sets (9.3) and (9.1), one can see, that the second set contains precisely those inputs with exactly two 2s in them. Comparing the output sets (9.4) and (9.2), we see that the first inputs sum all to an odd number, whereas in the second case they sum to an even number. So the rules of the game are can be explained as follow:

Each round Alice,Bob and Charlie are given an input drawn from the distribution $\mu(x,y,z)$ and they have to provide an answer, guessing the number of 2s.

1. If they get two 2s, they win if there answers sum to an even number.
2. If they get an odd number of 2s or non at all, they win if there answers sum to an odd number.

The game has a gap of $\chi_{G7} = 0.1333$ in its best version and is therefore much stronger than CHSH . It has a winning chance of $\omega_l(G7) = 0.7000$ given a local strategy and a winning chance of $\omega_q(G7) = 0.8333$ with a quantum strategy.

10. Going for more Outcomes

Of course our tools are not restricted to the 222222-scenario. But as mentioned in the introduction, the number of inequalities is vast and throwing our code against arbitrary inequalities, will most likely not lead to any particular interesting results. It's much more promising to search the literature for inequalities, which we might be able to improve or shine light on their strength. Although, the number of inequalities, which have been analysed, in the three party scenario with more than two in- or outputs is a bit sparse, there is a paper [20], which provides particularly interesting food for our code. In the paper they suggest a method to find three party equation, that are symmetric with respect to permutations of parties, for an arbitrary number of outputs K. Their method takes in such a party symmetric equation of the 222222-scenario, e.g. Mermin's equation, and generalises it into inequalities in a $KKK222$-scenario. Although their method fails in finding tight inequalities for arbitrary high Ks, they were able find all party symmetric classes of inequalities for the 333222-scenario. These 9 inequalities indexed $A1 - A9$ in the paper [20]. Translating them into full probability notation and running our code over them leads to the following table. We, see that with exception of inequality $A2$, all of

#	χ_G	$\omega_l(G)$	$\omega_q(G)$
A1	0.1540	0.6666	0.8207
A2	0.0881	0.8462	0.9342
A3	0.1259	0.6923	0.8182
A4	0.1167	0.7368	0.8535
A5	0.1238	0.6875	0.8113
A6	0.1101	0.7647	0.8748
A7	0.1056	0.7500	0.8556
A8	0.1131	0.7143	0.8274
A9	0.1275	0.8182	0.9457

Table 10.1.: The optimal gaps for the nine party symmetric tight inequalities in the 333222-scenario

them are stronger than CHSH. Although inequality $A7$ is almost no improvement over CHSH, the difference between $A3, A5, A9$ and especially $A1$ is significant.

Another nice result of the paper is a generalisation of Śliwa's 7th inequality, which we have analysed in detail in the last chapter. In their paper, they suggest that this inequality keeps to be tight for at least $K = 8$ and they compute its local- and quantum

bound and visibility v_{qm} up to $K = 6$ [1]. An important note is, that the bounds from the paper are actually lower bounds, therefore they correspond to the worst strategy. That's why we denoted these worst strategies in our games with an asterisk ($\omega_l^*(G)$, $\omega_q^*(G)$). Of course this has no effect on the gap, because minimising my game score has to be as hard as maximising it. Running our code over it, we find the following gaps.

K	χ_G	$\omega_l^*(G)$	$\omega_q^*(G)$	v_{qm}
2	0.1333	0.3000	0.1667	0.6000
3	0.1154	0.2632	0.1478	0.6460
4	0.1146	0.2500	0.1354	0.6516
5	0.1167	0.2432	0.1266	0.6495
6	0.1193	0.2391	0.1198	0.6456
7	0.1694	0.2364	0.0670	-
8	0.1841	0.2344	0.0503	-

Table 10.2.: Gaps, Local-, Quantum bounds and visibility v_{qm} for game 7 up to K=6

We see, that their generalisation method of inequality #7 $K > 2$ leads to a sudden loss on strength, but on the other hand comes with a sudden increase of visibility. For higher K the gap grows larger again and their seems to be no correlation between the gap and the visibility. Note, that the quantum bounds for $K = 7$ and $K = 8$ are only upper bounds computed using the second level of the NPA [21] hierarchy and might not be good enough to confirm the trend of a growing gap.

[1] The bounds for K=7 and K=8 were added by us

11. Conclusion

After a brief introduction into modern non-local physics, we have seen how we can transform a Bell inequality into a non-local game, which provides us not only with a pedagogical way of thinking about Bell inequalities, but also gives meaning to these otherwise arbitrary numbers. This also provides us eventually with a good measure of comparing the strength of different inequalities and versions thereof. Given that, we have analysed the three party case in the 2222222-scenario. We have found that there are 8 inequalities stronger than the CHSH-inequality, which is arguably a good benchmark.

These inequalities might give a good starting point for further experimental tests of non-locality and the predictions of quantum mechanics in multi-party scenarios. Further we hope that by providing a tool kit of programs for this kind of analysis, people will eventually find even stronger inequalities in more complex scenarios. The programs are written, such that they can be easily adapted to scenarios with even more than three players. There are a lot of unanswered question [22][23] about non-locality in multi-partite scenarios and the rise of quantum cryptography and quantum computing, will make these scenarios even more important.

As mentioned in the introduction, the non-local nature of our world is one of the greatest discoveries in 20th century physics. Quantum mechanics has showed us, that non-locality must not mean action at a distance. After all quantum non-locality is much weaker, than signaling non-locality. Nobody thinks, that in a future world Achilles can kill a turtle with non-local super powers and most, me included, would advocate, that non-local correlations, which are that strong don't even make sense in principle. But whats clear is that quantum mechanics has only opened up the window to non-local physics. After all quantum mechanics might be replaced by another theory, but non-locality will remain. That's why it is crucial to get better understanding of non-locality. How strong these non-local correlations can get in a theory, which is beyond quantum mechanics is one of the central questions in non-local physics and the answer to that, will be a good compass on how a theory beyond quantum mechanics, e.g. a theory of quantum gravity, might look like. Maybe this thesis can help a to make tiny step forward in this direction and physics can come closer to a final theory of everything, which might or might not exist :)

Bibliography

[1] Richard Cleve, Peter Hoyer, Ben Toner, and John Watrous. Consequences and limits of nonlocal strategies. 04 2004. doi:10.1109/CCC.2004.1313847.

[2] Mateus Araújo, Flavien Hirsch, and Marco Quintino. Bell nonlocality with a single shot. *Quantum*, 4:353, 10 2020. doi:10.22331/q-2020-10-28-353.

[3] Cezary Śliwa. Symmetries of the bell correlation inequalities. *Physics Letters A*, 317:165–168, 05 2003. doi:10.1016/S0375-9601(03)01115-0.

[4] Daniel Greenberger, Michael Horne, and Anton Zeilinger. Going beyond bell's theorem. 01 2008. doi:10.1007/978-94-017-0849-4_10.

[5] B. Hensen, H. Bernien, A. E. Dréau, A. Reiserer, N. Kalb, M. S. Blok, J. Ruitenberg, R. F. L. Vermeulen, R. N. Schouten, C. Abellán, W. Amaya, V. Pruneri, M. W. Mitchell, M. Markham, D. J. Twitchen, D. Elkouss, S. Wehner, T. H. Taminiau, and R. Hanson. Loophole-free bell inequality violation using electron spins separated by 1.3 kilometres. *Nature*, 526(7575):682–686, Oct 2015. doi:10.1038/nature15759.

[6] Marissa Giustina, Marijn A. M. Versteegh, Sören Wengerowsky, Johannes Handsteiner, Armin Hochrainer, Kevin Phelan, Fabian Steinlechner, Johannes Kofler, Jan-Åke Larsson, Carlos Abellán, Waldimar Amaya, Valerio Pruneri, Morgan W. Mitchell, Jörn Beyer, Thomas Gerrits, Adriana E. Lita, Lynden K. Shalm, Sae Woo Nam, Thomas Scheidl, Rupert Ursin, Bernhard Wittmann, and Anton Zeilinger. Significant-loophole-free test of bell's theorem with entangled photons. *Phys. Rev. Lett.*, 115:250401, Dec 2015. doi:10.1103/PhysRevLett.115.250401.

[7] Lynden K. Shalm, Evan Meyer-Scott, Bradley G. Christensen, Peter Bierhorst, Michael A. Wayne, Martin J. Stevens, Thomas Gerrits, Scott Glancy, Deny R. Hamel, Michael S. Allman, Kevin J. Coakley, Shellee D. Dyer, Carson Hodge, Adriana E. Lita, Varun B. Verma, Camilla Lambrocco, Edward Tortorici, Alan L. Migdall, Yanbao Zhang, Daniel R. Kumor, William H. Farr, Francesco Marsili, Matthew D. Shaw, Jeffrey A. Stern, Carlos Abellán, Waldimar Amaya, Valerio Pruneri, Thomas Jennewein, Morgan W. Mitchell, Paul G. Kwiat, Joshua C. Bienfang, Richard P. Mirin, Emanuel Knill, and Sae Woo Nam. Strong loophole-free test of local realism. *Phys. Rev. Lett.*, 115:250402, Dec 2015. doi:10.1103/PhysRevLett.115.250402.

[8] Valerio Scarani. *Bell Nonlocallity*. Oxford University Press, second edition, 2019. User's Guide and Reference Manual.

[9] John Stewart Bell. *Speakable and Unspeakable in Quantum Mechanics: Collected Papers on Quantum Philosophy*. Cambridge University Press, New York, 1987.

[10] J.J. Halliwell. Two proofs of fine's theorem. *Physics Letters A*, 378(40):2945–2950, 2014. doi:https://doi.org/10.1016/j.physleta.2014.08.012.

[11] Michael Ben-Or, Shafi Goldwasser, Joe Kilian, and Avi Wigderson. Multi-prover interactive proofs: How to remove intractability assumptions. In *Proceedings of the Twentieth Annual ACM Symposium on Theory of Computing*, STOC '88, page 113–131, New York, NY, USA, 1988. Association for Computing Machinery. doi:10.1145/62212.62223.

[12] Mateus Araújo. P-values and bayes factors. 02 2021. URL: https://mateusaraujo.info/2021/02/04/p-values-and-bayes-factors/.

[13] Marek Żukowski and Časlav Brukner. Bell's theorem for general n-qubit states. *Phys. Rev. Lett.*, 88:210401, May 2002. doi:10.1103/PhysRevLett.88.210401.

[14] Reinhard Werner and Michael Wolf. Bell inequalities and entanglement. *Quantum information & computation*, 1, 08 2001. doi:10.26421/QIC1.3-1.

[15] John F. Clauser and Michael A. Horne. Experimental consequences of objective local theories. *Phys. Rev. D*, 10:526–535, Jul 1974. doi:10.1103/PhysRevD.10.526.

[16] Sheila López-Rosa, Zhen-Peng Xu, and Adán Cabello. Maximum nonlocality in the (3,2,2) scenario. *Physical Review A*, 94(6), December 2016. doi:10.1103/physreva.94.062121.

[17] Mafalda Almeida, Jean-Daniel Bancal, Nicolas Brunner, Antonio Acín, Nicolas Gisin, and Stefano Pironio. Guess your neighbor's input: A multipartite nonlocal game with no quantum advantage. *Physical review letters*, 104:230404, 06 2010. doi:10.1103/PhysRevLett.104.230404.

[18] Daniel M. Greenberger, Michael A. Horne, Abner Shimony, and Anton Zeilinger. Bell's theorem without inequalities. *American Journal of Physics*, 58(12):1131–1143, 12 1990. doi:10.1119/1.16243.

[19] Gilles Brassard, Anne Broadbent, and Alain Tapp. Quantum pseudo-telepathy. *Foundations of Physics*, 35, 07 2004. doi:10.1007/s10701-005-7353-4.

[20] Basile Grandjean, Yeong-Cherng Liang, Jean-Daniel Bancal, Nicolas Brunner, and Nicolas Gisin. Bell inequalities for three systems and arbitrarily many measurement outcomes. *Physical Review A*, 85:052113, 05 2012. doi:10.1103/PhysRevA.85.052113.

[21] Miguel Navascues, Stefano Pironio, and Antonio Acín. Convergent hierarchy of semidefinite programs characterizing the set of quantum correlations. *New Journal of Physics*, 10, 07 2008. doi:10.1088/1367-2630/10/7/073013.

[22] Stefano Pironio, Jean-Daniel Bancal, and Valerio Scarani. Extremal correlations of the tripartite no-signaling polytope. *Journal of Physics A: Mathematical and Theoretical*, 44(6):065303, January 2011. doi:10.1088/1751-8113/44/6/065303.

[23] Rafael Chaves, Daniel Cavalcanti, and Leandro Aolita. Causal hierarchy of multipartite bell nonlocality. *Quantum*, 1:23, August 2017. doi:10.22331/q-2017-08-04-23.

A. Appendix

A.1. Game 7

This is the whole 2×2×2×2×2×2 Array of Game 7

```
[:, :, 1, 1, 1, 1] =          [:, :, 2, 1, 1, 1] =
 0.3  0.0                      0.0  0.3
 0.0  0.3                      0.3  0.0

[:, :, 1, 2, 1, 1] =          [:, :, 2, 2, 1, 1] =
 0.1  0.0                      0.0  0.1
 0.0  0.1                      0.1  0.0

[:, :, 1, 1, 2, 1] =          [:, :, 2, 1, 2, 1] =
 0.1  0.0                      0.0  0.1
 0.0  0.1                      0.1  0.0

[:, :, 1, 2, 2, 1] =          [:, :, 2, 2, 2, 1] =
 0.0  0.1                      0.1  0.0
 0.1  0.0                      0.0  0.1

[:, :, 1, 1, 1, 2] =          [:, :, 2, 1, 1, 2] =
 0.1  0.0                      0.0  0.1
 0.0  0.1                      0.1  0.0

[:, :, 1, 2, 1, 2] =          [:, :, 2, 2, 1, 2] =
 0.0  0.1                      0.1  0.0
 0.1  0.0                      0.0  0.1

[:, :, 1, 1, 2, 2] =          [:, :, 2, 1, 2, 2] =
 0.0  0.1                      0.1  0.0
 0.1  0.0                      0.0  0.1

[:, :, 1, 2, 2, 2] =          [:, :, 2, 2, 2, 2] =
 0.1  0.0                      0.0  0.1
 0.0  0.1                      0.1  0.0
```

A.2. Three Party Local Bounds nml2222-scenario

```
function TriLB(G::Array,s::Vector)

  #Number of Posssible Outputs

    a_s=s[1]
    b_s=s[2]
    c_s=s[3]

  #Number of Possible Inputs

    x_s=s[4]
    y_s=s[5]
    z_s=s[6]

#Setup In\Out

a=ones(Int64,2,1)
b=ones(Int64,2,1)
c=ones(Int64,2,1)

#Empty String for Storing
ScoreList=[]
#Run and Evaluate all Strategies

  for i1=1:a_s, i2=1:a_s
    for j1=1:b_s, j2=1:b_s
      for k1=1:c_s, k2=1:z_s

        a[1]=i1
        a[2]=i2
        b[1]=j1
        b[2]=j2
        c[1]=k1
        c[2]=k2
          P=zeros(a_s,b_s,c_s,2,2,2)
                for r=1:2, s=1:2, t=1:2
                  P[a[r],b[s],c[t],r,s,t]=1 #Evaluate Probability Matrix
    P(a,b,c|x,y,z)
                end
          Score=sum(G.*P)                    #Evaluate GameScore
         push!(ScoreList,Score)                      #Save Game Score

      #G==3 && display([A B C])             #Show Strategy if G has some
    property (e.g. ==x, >y...)
      end
    end
 end
  HighScore= maximum(ScoreList)
  LowScore = minimum(ScoreList)
    return(LowScore,HighScore)  #Print (LowScore,HighScore)
end
```

A.3. Linear Program

```
using JuMP
import Hypatia

function TriLP(M::Array,Gammas::Array,s::Array)

  #Setup for JuMP
  g = length(Gammas)   #Counts the number of NS-Conditions

  #Number of Posssible Outputs and Number of Possible Inputs
    a_s=s[1]
    b_s=s[2]
    c_s=s[3]
    x_s=s[4]
    y_s=s[5]
    z_s=s[6]

  #Setup JuMP-Model
    model= Model(Hypatia.Optimizer) #Choose Solver
      @variable(model, γ[1:g])
            @variable(model, β[1:x_s,1:y_s,1:z_s])
            @variable(model, α[1:x_s,1:y_s,1:z_s])

      G = sum(γ.*Gammas)
      M_new= M+G

        for x=1:x_s, y=1:y_s, z=1:z_s
            @constraint(model, M_new[:,:,:,x,y,z] .<= β[x,y,z])
        end

    for x=1:x_s, y=1:y_s, z=1:z_s
            @constraint(model, M_new[:,:,:,x,y,z] .>= α[x,y,z])
        end

    @objective(model, Min , sum( β - α ))

  #Run Optimization
    optimize!(model)

        α = value.(α)
        β = value.(β)

    game = value.(M_new)        #Writes the Game
        for x=1:x_s, y=1:y_s, z=1:z_s
            game[:,:,:,x,y,z] .-= α[x,y,z]
        end
      game /= sum(β - α)

        return(sum(β),sum(α),game)
end
```

A.4. Correlator Notation to Full Probability Notation

```
function TriCRP(E::Array)

#Setup Full Probability Tensor
  x_s = size(E,1)-1 # #Alices Inputs
  y_s = size(E,2)-1   # #Bobs Inputs
  z_s = size(E,3)-1 # #Charlies Inputs

  M=zeros( 2 , 2 , 2 , x_s , y_s , z_s )

#Triple Correlators
  for x=1:x_s, y=1:y_s, z=1:z_s
      for a=1:2, b=1:2, c=1:2
        M[ a , b , c , x , y , z] = (-1)^(1+a+b+c)*E[ x+1, y+1 , z+1 ]
      end
  end
#Dual Correlators Alice/Bob E_xy
  for x=1:x_s, y=1:y_s, z=1:z_s
      for a=1:2, b=1:2, c=1:2
        M[ a , b , c , x , y , z] += (-1)^(a+b)*(1/z_s)*E[ x+1 , y+1 , 1
    ]
      end
  end
#Dual Correlators Alice/Charlie E_yz
  for x=1:x_s, y=1:y_s, z=1:z_s
      for a=1:2, b=1:2, c=1:2
        M[ a , b , c , x , y , z] += (-1)^(a+c)*(1/y_s)*E[ x+1 , 1 , z+1
    ]
      end
  end
#Dual Correlators Bob/Charlie E_yz
  for x=1:x_s, y=1:y_s, z=1:z_s
      for a=1:2, b=1:2, c=1:2
        M[ a , b , c , x , y , z] += ((-1)^(b+c))*(1/x_s)*E[ 1 , y+1 , z
    +1 ]
      end
  end
#Single Correlators Alice E_x
  for x=1:x_s, y=1:y_s, z=1:z_s
      for a=1:2, b=1:2, c=1:2
        M[ a , b , c , x , y , z] += ((-1)^(1+a))*(1/(y_s*z_s))*E[ x+1 ,
    1 , 1 ]
      end
  end
#Single Correlators Bob E_y
  for x=1:x_s, y=1:y_s, z=1:z_s
      for a=1:2, b=1:2, c=1:2
        M[ a , b , c , x , y , z] += ((-1)^(1+b))*(1/(x_s*z_s))*E[ 1 , y
    +1, 1 ]
      end
  end
#Single Correlators Charlie E_z
```

```
  for x=1:x_s, y=1:y_s, z=1:z_s
      for a=1:2, b=1:2, c=1:2
        M[ a , b , c , x , y , z] += ((-1)^(1+c))*(1/(x_s*y_s))*E[ 1 , 1
    , z+1]
      end
  end
  return(M)
end
```

A.5. Collins-Gisin Notation to Full Probability Notation

```
function TriCGP(E::Array, s::Vector)


#E...Bell Scenario in Collins-Gisin
#S...NUmber of Inputs and Outputs for Scenario


#Defining concise variables

  #Number of Posssible Outputs

    a_s=s[1]
    b_s=s[2]
    c_s=s[3]

  #Number of Possible Inputs

    x_s=s[4]
    y_s=s[5]
    z_s=s[6]

#Slice up E for concise handling

  #Conditional Probablities P(abc|xyz) and  size E

    E_xyz=E[2:end,2:end,2:end]

    end_x=size(E_xyz,1)
    end_y=size(E_xyz,2)
    end_z=size(E_xyz,3)

    display(end_x)
    display(end_y)
    display(end_z)
  #Conditional Probalities P(ab|xy), P(ac|xz), P(bc|yz)

    E_xy=E[2:end,2:end, 1 ]
    E_xz=E[2:end, 1 , 2:end]
    E_yz=E[ 1 , 2:end , 2:end]

```

```
41   #Conditional Probabilites P(a|x), P(b|y), P(c|z)
42
43     E_x=E[2:end, 1 , 1 ]
44     E_y=E[ 1 ,2:end, 1 ]
45     E_z=E[ 1 , 1 ,2:end]
46
47 #Setup Empty Full Probability Array M
48
49     M_xyz=zeros(a_s,b_s,c_s,x_s,y_s,z_s)
50     M_xy=zeros(a_s,b_s,c_s,x_s,y_s,z_s)
51     M_xz=zeros(a_s,b_s,c_s,x_s,y_s,z_s)
52     M_yz=zeros(a_s,b_s,c_s,x_s,y_s,z_s)
53     M_x=zeros(a_s,b_s,c_s,x_s,y_s,z_s)
54     M_y=zeros(a_s,b_s,c_s,x_s,y_s,z_s)
55     M_z=zeros(a_s,b_s,c_s,x_s,y_s,z_s)
56
57 #Fill up M Note: i,j,k run over E i.e. from 1:end_(.) ; x,y,z,a,b,c run
       from 1:(.)_s
58
59   #Fill P(abc|xyz)
60     for i=1:end_x, j=1:end_y, k=1:end_z
61         M_xyz[mod(i-1,a_s-1)+1,mod(j-1,b_s-1)+1,mod(k-1,c_s-1)+1,ceil(Int
       ,i/(a_s-1)),ceil(Int,j/(b_s-1)),ceil(Int,k/(c_s-1))]=E_xyz[i,j,k]
62     end
63
64   #Add P(ab|xy)
65     for i=1:end_x, j=1:end_y
66       for z=1:z_s, c=1:c_s
67         M_xy[ mod(i-1,a_s-1)+1 , mod(j-1,b_s-1)+1 , c , ceil(Int,i/(a_s
       -1)) , ceil(Int,j/(b_s-1)) , z ]+= (1/z_s)*E_xy[i,j]
68       end
69     end
70
71   #Add P(ac|xz)
72     for i=1:end_x, k=1:end_z
73       for y=1:y_s, b=1:b_s
74         M_xz[ mod(i-1,a_s-1)+1 , b , mod(k-1,c_s-1)+1 , ceil(Int,i/(a_s
       -1)) , y , ceil(Int,k/(c_s-1)) ]+= (1/y_s)*E_xz[i,k]
75       end
76     end
77
78   #Add P(ac|xz)
79     for j=1:end_y, k=1:end_z
80       for x=1:x_s, a=1:a_s
81         M_yz[ a , mod(j-1,b_s-1)+1, mod(k-1,c_s-1)+1 , x , ceil(Int,j/(
       b_s-1)) , ceil(Int,k/(c_s-1))] += (1/x_s)*E_yz[j,k]
82       end
83     end
84
85   #Add P(a|x)
86     for i=1:end_x
87       for y=1:y_s, b=1:b_s, z=1:z_s, c=1:c_s
```

```
88          M_x[ mod(i-1,a_s-1)+1 , b , c , ceil(Int,i/(a_s-1)) , y , z ] +=
    (1/(y_s*z_s))*E_x[i]
89        end
90      end
91
92    #Add P(b|y)
93      for j=1:end_y
94        for x=1:x_s, a=1:a_s, z=1:z_s, c=1:c_s
95          M_y[ a , mod(j-1,b_s-1)+1 , c , x , ceil(Int,j/(b_s-1)) , z ] +=
    (1/(x_s*z_s))*E_y[j]
96        end
97      end
98
99    #Add P(b|y)
100     for k=1:end_z
101       for x=1:x_s, a=1:a_s, y=1:y_s, b=1:b_s
102         M_y[ a , b , mod(k-1,c_s-1)+1 , x , y , ceil(Int,k/(c_s-1)) ] +=
    (1/(x_s*y_s))*E_z[k]
103       end
104     end
105
106   #Full M
107     M = M_xyz+ M_xy + M_yz + M_xz + M_x + M_y + M_z
108   return(M)
109 end
```

A.6. No-Signaling Conditions for three party scenario

```
1  function TriNS(s::Vector)
2    #Number of Posssible Outputs
3      a_s=s[1]
4      b_s=s[2]
5      c_s=s[3]
6
7    #Number of Possible Inputs
8
9      x_s=s[4]
10     y_s=s[5]
11     z_s=s[6]
12
13
14 #Setup Empty NoSignal Array and  storage array Gamma
15     g_o=zeros( a_s , b_s , c_s , x_s , y_s , z_s )
16       Gammas=[]
17 #No Signaling Conditions for Alice and Bob
18     for a=1:a_s-1, b=1:b_s-1, x=1:x_s, y=1:y_s
19         for z=1:z_s-1
20           g=copy(g_o)
21             for c=1:c_s
22               g[ a , b , c , x , y ,  z   ] = 1
23               g[ a , b , c , x , y , z+1  ] = -1
```

```
                        end
          push!(Gammas,g)
        end
      end
      display(length(Gammas))

#No Signaling Conditions for Alice and Charlie
    for a=1:a_s-1, c=1:c_s-1, x=1:x_s, z=1:z_s
        for y=1:y_s-1
          g=copy(g_o)
            for b=1:b_s
              g[ a , b , c , x ,  y  , z ] = 1    g[ a , b , c , x , y+1
    , z ] = -1
                        end
          push!(Gammas,g)
        end
      end
      display(length(Gammas))

#No Signaling Conditions for Bob und Charlie
    for b=1:b_s-1, c=1:c_s-1, y=1:y_s, z=1:z_s
        for x=1:x_s-1
          g=copy(g_o)
            for a=1:a_s
              g[ a , b , c ,  x  , y , z ] = 1
              g[ a , b , c , x+1 , y , z ] = -1
                        end
          push!(Gammas,g)
        end
      end
      display(length(Gammas))

#No Signaling Conditions for Alice
    n_yz=y_s*z_s
    yz=zeros(Int64,n_yz,2)

    for y=1:y_s, z=1:z_s
        yz[(y-1)*z_s + z , 1] = y
        yz[(y-1)*z_s + z , 2] = z
    end

    for a=1:a_s-1, x=1:x_s
      for n=1:(n_yz)-1
        g=copy(g_o)
        for b=1:b_s, c=1:c_s
          g[ a , b , c , x ,  yz[n,1]  ,  yz[n,2] ]=1
                    g[ a , b , c , x , yz[n+1,1] , yz[n+1,2]  ]=-1
                end
        push!(Gammas,g)
      end
    end

    display(length(Gammas))
```

```
76
77 #No Signaling Condition for Bob
78     n_xz=x_s*z_s
79     xz=zeros(Int64,n_xz,2)
80
81     for x=1:x_s, z=1:z_s
82         xz[(x-1)*z_s + z , 1] = x
83         xz[(x-1)*z_s + z , 2] = z
84     end
85
86     for b=1:b_s-1, y=1:y_s
87       for n=1:(n_xz)-1
88         g=copy(g_o)
89           for a=1:a_s, c=1:c_s              g[ a , b , c , xz[n,1] , y ,
    xz[n,2] ] = 1
90             g[ a , b , c , xz[n+1,1] , y , xz[n+1,2] ]= -1
91           end
92         push!(Gammas,g)
93       end
94     end
95     display(length(Gammas))
96
97 #No Signaling Condition for Charlie
98     n_xy=x_s*y_s
99     xy=zeros(Int64,n_xy,2)
100
101    for x=1:x_s, y=1:y_s
102        xy[(x-1)*y_s + y , 1] = x
103        xy[(x-1)*y_s + y , 2] = y
104    end
105    for c=1:c_s-1, z=1:z_s
106      for n=1:(n_xz)-1
107        g=copy(g_o)
108          for a=1:a_s, b=1:b_s
109          g[ a , b , c , xy[n,1] , xy[n,2] , z ] = 1    g[ a , b , c , xy
    [n+1,1] , xy[n+1,2] , z ]= -1  end
110        push!(Gammas,g)
111      end
112    end
113  display(length(Gammas))
114  return(Gammas)
115 end
```

A.7. Sliwa's List

```
function SliwList()

 List=[]
#1
E=zeros(3,3,3)
E[1,1,1]=1; E[2,1,1]=-1; E[1,2,1]=-1;  E[2,2,1]=1; E[1,1,2]=-1; E
    [2,1,2]=1;  E[1,2,2]=1; E[2,2,2]=-1;
push!(List,E)
#2
E=zeros(3,3,3)
E[1,1,1]=2; E[2,2,2]=-1; E[3,3,2]=-1; E[3,2,3]=-1; E[2,3,3]=1;
push!(List,E)
#3
E=zeros(3,3,3)
E[1,1,1]=2; E[2,2,2]=-1; E[3,2,2]=-1; E[2,3,3]=-1; E[3,3,3]=1;
push!(List,E)
#4
E=zeros(3,3,3)
E[1,1,1]=2; E[2,1,1]=-2; E[1,2,2]=-1; E[2,2,2]=1; E[1,3,2]=-1; E
    [2,3,2]=1; E[1,2,3]=-1; E[2,2,3]=1; E[1,3,3]=1; E[2,3,3]=-1;
push!(List,E)
#5
E=zeros(3,3,3)
E[1,1,1]=3; E[2,1,1]=-1; E[1,2,1]=-1; E[3,2,1]=-1; E[2,3,1]=-1; E
    [3,3,1]=1; E[1,1,2]=-1; E[3,1,2]=-1; E[2,2,2]=1; E[3,2,2]=1; E
    [1,3,2]=-1; E[2,3,2]=1; E[2,1,3]=-1; E[3,1,3]=1; E[1,2,3]=-1; E
    [2,2,3]=1; E[1,3,3]=1; E[3,3,3]=-1;
push!(List,E)
#6
E=zeros(3,3,3)
E[1,1,1]=3; E[2,1,1]=-1; E[1,2,1]=-1; E[2,2,1]=-1; E[1,1,2]=-1; E
    [3,1,2]=-1; E[2,2,2]=1; E[3,2,2]=1; E[1,3,2]=-1; E[2,3,2]=1; E
    [2,1,3]=-1; E[3,1,3]=1; E[1,2,3]=1; E[3,2,3]=-1; E[1,3,3]=-1; E
    [2,3,3]=1;
push!(List,E)
#7
E=zeros(3,3,3)
E[1,1,1]=4; E[2,2,2]=-3; E[3,2,2]=-1; E[2,3,2]=-1; E[3,3,2]=1; E
    [2,2,3]=-1; E[3,2,3]=1; E[2,3,3]=1; E[3,3,3]=-1;
push!(List,E)
#8
E=zeros(3,3,3)
E[1,1,1]=4; E[2,2,1]=-1; E[3,2,1]=-1; E[2,3,1]=-1; E[3,3,1]=-1; E
    [2,2,2]=-2; E[3,3,2]=2; E[2,2,3]=-1; E[3,2,3]=1; E[2,3,3]=1; E
    [3,3,3]=-1;
push!(List,E)
#9
E=zeros(3,3,3)
E[1,1,1]=4; E[2,2,1]=-1; E[3,2,1]=-1; E[2,3,1]=-1; E[3,3,1]=-1; E
    [2,2,2]=-2; E[2,3,2]=2; E[2,2,3]=-1; E[3,2,3]=1; E[2,3,3]=-1; E
    [3,3,3]=1;
```

```
39 push!(List,E)
40 #10
41 E=zeros(3,3,3)
42 E[1,1,1]=4; E[2,2,1]=-1; E[3,2,1]=-1; E[2,3,1]=-1; E[3,3,1]=-1; E
       [2,1,2]=-1; E[3,1,2]=1; E[1,2,2]=-1; E[2,2,2]=-1; E[1,3,2]=1; E
       [3,3,2]=1; E[2,1,3]=-1; E[3,1,3]=1; E[1,2,3]=1; E[3,2,3]=-1; E
       [1,3,3]=-1; E[2,3,3]=1;
43 push!(List,E)
44 #11
45 E=zeros(3,3,3)
46 E[1,1,1]=4; E[2,2,1]=-2; E[3,3,1]=-2; E[2,2,2]=-1; E[3,2,2]=-1; E
       [2,3,2]=1; E[3,3,2]=1; E[2,2,3]=-1; E[3,2,3]=1; E[2,3,3]=-1; E
       [3,3,3]=1;
47 push!(List,E)
48 #12
49 E=zeros(3,3,3)
50 E[1,1,1]=4; E[2,2,1]=-2; E[3,3,1]=-2; E[2,1,2]=-1; E[3,1,2]=-1; E
       [1,2,2]=1; E[3,2,2]=-1; E[1,3,2]=1; E[2,3,2]=1; E[2,1,3]=-1; E
       [3,1,3]=-1; E[1,2,3]=1; E[3,2,3]=1; E[1,3,3]=1; E[2,3,3]=-1;
51 push!(List,E)
52 #13
53 E=zeros(3,3,3)
54 E[1,1,1]=4; E[2,2,1]=-2; E[3,2,1]=-2; E[2,2,2]=-1; E[3,2,2]=1; E
       [2,3,2]=-1; E[3,3,2]=1; E[2,2,3]=-1; E[3,2,3]=1; E[2,3,3]=1; E
       [3,3,3]=-1;
55 push!(List,E)
56 #14
57 E=zeros(3,3,3)
58 E[1,1,1]=4; E[2,2,1]=-2; E[3,2,1]=-2; E[2,1,2]=-1; E[3,1,2]=1; E
       [2,3,2]=-1; E[3,3,2]=1; E[2,1,3]=-1; E[3,1,3]=1; E[2,3,3]=1; E
       [3,3,3]=-1;
59 push!(List,E)
60 #15
61 E=zeros(3,3,3)
62 E[1,1,1]=4; E[2,2,1]=-2; E[3,2,1]=-2; E[2,1,2]=-1; E[3,1,2]=-1; E
       [1,2,2]=2; E[2,3,2]=-1; E[3,3,2]=1; E[2,1,3]=-1; E[3,1,3]=-1; E
       [1,2,3]=2; E[2,3,3]=1; E[3,3,3]=-1;
63 push!(List,E)
64 #16
65 E=zeros(3,3,3)
66 E[1,1,1]=4; E[2,1,1]=-1; E[3,1,1]=-1; E[2,2,1]=-1; E[3,2,1]=-1; E
       [2,1,2]=-1; E[3,1,2]=-1; E[3,2,2]=2; E[2,3,2]=-1; E[3,3,2]=1; E
       [2,2,3]=-1; E[3,2,3]=1; E[2,3,3]=1; E[3,3,3]=-1;
67 push!(List,E)
68 #17
69 E=zeros(3,3,3)
70 E[1,1,1]=4; E[2,1,1]=-1; E[3,1,1]=-1; E[2,2,1]=-1; E[3,2,1]=-1; E
       [2,1,2]=-1; E[3,1,2]=-1; E[2,2,2]=1; E[3,2,2]=1; E[2,3,3]=-2; E
       [3,3,3]=2;
71 push!(List,E)
72 #18
73 E=zeros(3,3,3)
74 E[1,1,1]=4; E[2,1,1]=-1; E[3,1,1]=-1; E[2,2,1]=-1; E[3,2,1]=-1; E
```

```
    [2,1,2]=-1; E[3,1,2]=-1; E[1,2,2]=2; E[2,3,2]=-1; E[3,3,2]=1; E
    [2,2,3]=-1; E[3,2,3]=1; E[1,3,3]=-2; E[2,3,3]=1; E[3,3,3]=1;
push!(List,E)
#19
E=zeros(3,3,3)
E[1,1,1]=4; E[2,1,1]=-1; E[3,1,1]=-1; E[2,2,1]=-1; E[3,2,1]=-1; E
    [2,1,2]=-1; E[3,1,2]=-1; E[1,2,2]=2; E[1,3,2]=-2; E[2,3,2]=1; E
    [3,3,2]=1; E[2,2,3]=-1; E[3,2,3]=1; E[2,3,3]=-1; E[3,3,3]=1;
push!(List,E)
#20
E=zeros(3,3,3)
E[1,1,1]=4; E[2,1,1]=-1; E[3,1,1]=-1; E[2,2,1]=-1; E[3,2,1]=1; E
    [2,3,1]=-1; E[3,3,1]=1; E[2,1,2]=-1; E[3,1,2]=1; E[1,2,2]=1; E
    [2,2,2]=-1; E[3,2,2]=-1; E[1,3,2]=1; E[2,3,2]=-1; E[3,3,2]=-1; E
    [1,2,3]=-1; E[2,2,3]=1; E[3,2,3]=1; E[1,3,3]=1; E[2,3,3]=-1; E
    [3,3,3]=-1;
push!(List,E)
#21
E=zeros(3,3,3)
E[1,1,1]=4; E[2,1,1]=-1; E[3,1,1]=-1; E[1,2,1]=-1; E[2,2,1]=-1; E
    [1,3,1]=-1; E[3,3,1]=1; E[2,1,2]=-1; E[3,1,2]=-1; E[1,2,2]=-1; E
    [2,2,2]=2; E[3,2,2]=1; E[1,3,2]=-1; E[2,3,2]=1; E[2,2,3]=-1; E
    [3,2,3]=1; E[2,3,3]=1; E[3,3,3]=-1;
push!(List,E)
#22
E=zeros(3,3,3)
E[1,1,1]=4; E[2,1,1]=-1; E[3,1,1]=-1; E[1,2,1]=-1; E[2,2,1]=-1; E
    [1,3,1]=-1; E[3,3,1]=1; E[1,1,2]=-1; E[2,1,2]=-1; E[1,2,2]=-1; E
    [2,2,2]=2; E[3,2,2]=1; E[2,3,2]=1; E[3,3,2]=-1; E[1,1,3]=-1; E
    [3,1,3]=1; E[2,2,3]=1; E[3,2,3]=-1; E[1,3,3]=1; E[2,3,3]=-1;
push!(List,E)
#23
E=zeros(3,3,3)
E[1,1,1]=4; E[2,1,1]=-1; E[3,1,1]=-1; E[1,2,1]=-1; E[2,2,1]=1; E
    [3,2,1]=1; E[1,3,1]=-1; E[2,3,1]=1; E[3,3,1]=1; E[2,1,2]=-1; E
    [3,1,2]=1; E[2,2,2]=1; E[3,2,2]=-1; E[2,3,2]=1; E[3,3,2]=-1; E
    [1,2,3]=-1; E[2,2,3]=1; E[3,2,3]=1; E[1,3,3]=1; E[2,3,3]=-1; E
    [3,3,3]=-1;
push!(List,E)
#24
E=zeros(3,3,3)
E[1,1,1]=5; E[2,1,1]=-1; E[1,2,1]=-1; E[3,2,1]=-1; E[2,3,1]=-1; E
    [3,3,1]=-1; E[1,1,2]=-1; E[3,1,2]=-1; E[1,2,2]=1; E[2,2,2]=-2; E
    [3,2,2]=1; E[3,3,2]=2; E[2,1,3]=-1; E[3,1,3]=-1; E[3,2,3]=2; E
    [2,3,3]=1; E[3,3,3]=-1;
push!(List,E)
#25
E=zeros(3,3,3)
E[1,1,1]=5; E[2,1,1]=-1; E[1,2,1]=-1; E[3,2,1]=-1; E[2,3,1]=-1; E
    [3,3,1]=-1; E[1,1,2]=-1; E[3,1,2]=-1; E[1,2,2]=1; E[2,2,2]=-2; E
    [3,2,2]=1; E[3,3,2]=2; E[2,1,3]=-1; E[3,1,3]=-1; E[2,2,3]=2; E
    [2,3,3]=-1; E[3,3,3]=1;
push!(List,E)
```

```
#26
E=zeros(3,3,3)
E[1,1,1]=5; E[2,1,1]=-1; E[1,2,1]=-1; E[2,2,1]=-1; E[3,3,1]=-2; E
    [1,1,2]=-1; E[2,1,2]=-1; E[1,2,2]=-1; E[2,2,2]=1; E[3,3,2]=2; E
    [3,1,3]=-2; E[3,2,3]=2; E[1,3,3]=2; E[2,3,3]=-2;
push!(List,E)
#27
E=zeros(3,3,3)
E[1,1,1]=5; E[2,1,1]=-2; E[3,1,1]=-1; E[1,2,1]=-1; E[2,2,1]=1; E
    [2,3,1]=-1; E[3,3,1]=-1; E[1,1,2]=-1; E[2,1,2]=1; E[2,2,2]=-2; E
    [3,2,2]=2; E[1,3,2]=-1; E[2,3,2]=1; E[2,1,3]=-1; E[3,1,3]=-1; E
    [1,2,3]=-1; E[2,2,3]=1; E[1,3,3]=-1; E[2,3,3]=2; E[3,3,3]=1;
push!(List,E)
#28
E=zeros(3,3,3)
E[1,1,1]=6; E[2,1,1]=-1; E[3,1,1]=-1; E[2,2,1]=-1; E[3,2,1]=1; E
    [2,1,2]=-1; E[3,1,2]=1; E[1,2,2]=1; E[2,2,2]=-2; E[3,2,2]=-1; E
    [1,3,2]=-1; E[2,3,2]=1; E[3,3,2]=2; E[1,2,3]=-1; E[2,2,3]=1; E
    [3,2,3]=2; E[1,3,3]=-1; E[2,3,3]=3;
push!(List,E)
#29
E=zeros(3,3,3)
E[1,1,1]=6; E[2,1,1]=-1; E[3,1,1]=-1; E[2,2,1]=-1; E[3,2,1]=1; E
    [2,1,2]=-1; E[3,1,2]=1; E[1,2,2]=1; E[2,2,2]=-2; E[3,2,2]=-1; E
    [1,3,2]=-1; E[2,3,2]=1; E[3,3,2]=2; E[1,2,3]=-1; E[2,2,3]=3; E
    [1,3,3]=-1; E[2,3,3]=1; E[3,3,3]=2;
push!(List,E)
#30
E=zeros(3,3,3)
E[1,1,1]=6; E[2,1,1]=-1; E[3,1,1]=-1; E[2,2,1]=-2; E[3,2,1]=2; E
    [2,3,1]=-1; E[3,3,1]=1; E[2,1,2]=-1; E[3,1,2]=1; E[1,2,2]=1; E
    [2,2,2]=-2; E[3,2,2]=-1; E[1,3,2]=1; E[2,3,2]=-1; E[3,3,2]=-2; E
    [1,2,3]=-1; E[2,2,3]=2; E[3,2,3]=1; E[1,3,3]=1; E[2,3,3]=-2; E
    [3,3,3]=-1;
push!(List,E)
#31
E=zeros(3,3,3)
E[1,1,1]=6; E[2,1,1]=-1; E[3,1,1]=-1; E[1,2,1]=-1; E[3,2,1]=1; E
    [1,3,1]=-1; E[2,3,1]=1; E[2,1,2]=-1; E[3,1,2]=1; E[3,2,2]=-2; E
    [2,3,2]=1; E[3,3,2]=-3; E[1,2,3]=-1; E[2,2,3]=2; E[3,2,3]=1; E
    [1,3,3]=1; E[2,3,3]=-2; E[3,3,3]=-1;
push!(List,E)
#32
E=zeros(3,3,3)
E[1,1,1]=6; E[2,1,1]=-1; E[3,1,1]=-1; E[1,2,1]=-1; E[3,2,1]=1; E
    [1,3,1]=-1; E[2,3,1]=1; E[2,1,2]=-2; E[3,1,2]=2; E[3,2,2]=-2; E
    [3,3,2]=-2; E[2,1,3]=-1; E[3,1,3]=1; E[1,2,3]=1; E[2,2,3]=-2; E
    [3,2,3]=-1; E[1,3,3]=-1; E[2,3,3]=1; E[3,3,3]=2;
push!(List,E)
#33
E=zeros(3,3,3)
E[1,1,1]=6; E[2,1,1]=-1; E[3,1,1]=-1; E[1,2,1]=-1; E[3,2,1]=1; E
    [1,3,1]=-1; E[2,3,1]=1; E[1,1,2]=-1; E[3,1,2]=1; E[3,2,2]=-2; E
```

```
    [1,3,2]=1; E[2,3,2]=-2; E[3,3,2]=-1; E[1,1,3]=-1; E[2,1,3]=1; E
    [1,2,3]=1; E[2,2,3]=-2; E[3,2,3]=-1; E[2,3,3]=-1; E[3,3,3]=3;
push!(List,E)
#34
E=zeros(3,3,3)
E[1,1,1]=6; E[2,1,1]=-1; E[3,1,1]=-1; E[1,2,1]=-1; E[3,2,1]=1; E
    [1,3,1]=-1; E[2,3,1]=1; E[1,1,2]=-1; E[3,1,2]=1; E[1,2,2]=1; E
    [2,2,2]=2; E[3,2,2]=-1; E[1,3,2]=2; E[2,3,2]=-2; E[3,3,2]=-2; E
    [1,1,3]=-1; E[2,1,3]=1; E[1,2,3]=2; E[1,3,3]=1; E[2,3,3]=-1; E
    [3,3,3]=2;
push!(List,E)
#35
E=zeros(3,3,3)
E[1,1,1]=6; E[2,1,1]=-1; E[3,1,1]=-1; E[1,2,1]=-1; E[2,2,1]=1; E
    [3,2,1]=2; E[1,3,1]=-1; E[2,3,1]=2; E[3,3,1]=1; E[2,1,2]=-1; E
    [3,1,2]=1; E[2,2,2]=1; E[3,2,2]=-1; E[2,3,2]=2; E[3,3,2]=-2; E
    [1,2,3]=-1; E[2,2,3]=2; E[3,2,3]=1; E[1,3,3]=1; E[2,3,3]=-2; E
    [3,3,3]=-1;
push!(List,E)
#36
E=zeros(3,3,3)
E[1,1,1]=6; E[2,1,1]=-2; E[2,2,1]=-1; E[3,2,1]=-1; E[2,3,1]=-1; E
    [3,3,1]=-1; E[2,1,2]=-1; E[3,1,2]=-1; E[1,2,2]=-1; E[2,2,2]=2; E
    [3,2,2]=-1; E[1,3,2]=1; E[2,3,2]=-1; E[3,3,2]=2; E[2,1,3]=-1; E
    [3,1,3]=-1; E[1,2,3]=1; E[2,2,3]=-1; E[3,2,3]=2; E[1,3,3]=1; E
    [2,3,3]=-2; E[3,3,3]=1;
push!(List,E)
#37
E=zeros(3,3,3)
E[1,1,1]=6; E[2,1,1]=-2; E[2,2,1]=-1; E[3,2,1]=-1; E[2,3,1]=-1; E
    [3,3,1]=-1; E[2,1,2]=-1; E[3,1,2]=-1; E[1,2,2]=-1; E[2,2,2]=3; E
    [1,3,2]=1; E[2,3,2]=-2; E[3,3,2]=1; E[2,1,3]=-1; E[3,1,3]=-1; E
    [1,2,3]=1; E[2,2,3]=-2; E[3,2,3]=1; E[1,3,3]=1; E[2,3,3]=-1; E
    [3,3,3]=2;
push!(List,E)
#38
E=zeros(3,3,3)
E[1,1,1]=6; E[2,1,1]=-2; E[2,2,1]=-2; E[3,2,1]=-2; E[2,1,2]=-1; E
    [3,1,2]=-1; E[1,2,2]=1; E[2,2,2]=-1; E[3,2,2]=2; E[1,3,2]=-1; E
    [2,3,2]=2; E[3,3,2]=-1; E[2,1,3]=-1; E[3,1,3]=-1; E[1,2,3]=1; E
    [2,2,3]=-1; E[3,2,3]=2; E[1,3,3]=1; E[2,3,3]=-2; E[3,3,3]=1;
push!(List,E)
#39
E=zeros(3,3,3)
E[1,1,1]=6; E[2,1,1]=-2; E[1,2,1]=-2; E[2,2,1]=1; E[3,2,1]=-1; E
    [2,3,1]=-1; E[3,3,1]=-1; E[1,1,2]=-2; E[2,1,2]=1; E[3,1,2]=-1; E
    [1,2,2]=1; E[2,2,2]=-2; E[3,2,2]=1; E[1,3,2]=-1; E[2,3,2]=1; E
    [3,3,2]=2; E[2,1,3]=-1; E[3,1,3]=-1; E[1,2,3]=-1; E[2,2,3]=1; E
    [3,2,3]=2; E[1,3,3]=-1; E[2,3,3]=2; E[3,3,3]=-1;
push!(List,E)
#40
E=zeros(3,3,3)
E[1,1,1]=6; E[2,1,1]=-2; E[3,1,1]=-2; E[1,2,1]=-2; E[2,2,1]=1; E
```

```
    [3,2,1]=1; E[2,3,1]=-1; E[3,3,1]=-1; E[2,1,2]=-1; E[3,1,2]=-1; E
    [1,2,2]=-2; E[2,2,2]=1; E[3,2,2]=1; E[1,3,2]=-2; E[2,3,2]=2; E
    [3,3,2]=2; E[2,1,3]=-1; E[3,1,3]=1; E[2,2,3]=2; E[3,2,3]=-2; E
    [2,3,3]=-1; E[3,3,3]=1;
push!(List,E)
#41
E=zeros(3,3,3)
E[1,1,1]=7; E[2,1,1]=-1; E[1,2,1]=-1; E[2,2,1]=-1; E[1,1,2]=-1; E
    [3,1,2]=-1; E[2,2,2]=3; E[3,2,2]=1; E[1,3,2]=-1; E[2,3,2]=1; E
    [3,3,2]=2; E[2,1,3]=-1; E[3,1,3]=1; E[1,2,3]=-1; E[2,2,3]=4; E
    [3,2,3]=-1; E[1,3,3]=1; E[2,3,3]=-1; E[3,3,3]=-2;
push!(List,E)
#42
E=zeros(3,3,3)
E[1,1,1]=8; E[2,1,1]=-1; E[3,1,1]=-1; E[1,2,1]=-1; E[2,2,1]=-1; E
    [1,3,1]=-1; E[3,3,1]=1; E[2,1,2]=-1; E[3,1,2]=1; E[1,2,2]=-1; E
    [2,2,2]=2; E[3,2,2]=1; E[1,3,2]=1; E[2,3,2]=1; E[3,3,2]=-4; E
    [3,1,3]=-2; E[2,2,3]=1; E[3,2,3]=3; E[1,3,3]=-2; E[2,3,3]=3; E
    [3,3,3]=1;
push!(List,E)
#43
E=zeros(3,3,3)
E[1,1,1]=8; E[2,1,1]=-2; E[1,2,1]=-2; E[2,2,1]=1; E[3,2,1]=-1; E
    [2,3,1]=-1; E[3,3,1]=1; E[2,1,2]=-1; E[3,1,2]=-1; E[1,2,2]=-1; E
    [2,2,2]=2; E[3,2,2]=3; E[1,3,2]=1; E[2,3,2]=-1; E[3,3,2]=-2; E
    [2,1,3]=-1; E[3,1,3]=1; E[1,2,3]=-1; E[2,2,3]=3; E[1,3,3]=-1; E
    [2,3,3]=4; E[3,3,3]=-1;
push!(List,E)
#44
E=zeros(3,3,3)
E[1,1,1]=8; E[2,1,1]=-2; E[3,1,1]=-2; E[2,2,1]=-2; E[3,2,1]=2; E
    [2,1,2]=-1; E[3,1,2]=1; E[1,2,2]=2; E[2,2,2]=-2; E[3,2,2]=-2; E
    [1,3,2]=-2; E[2,3,2]=1; E[3,3,2]=3; E[2,1,3]=-1; E[3,1,3]=1; E
    [1,2,3]=2; E[2,2,3]=-2; E[3,2,3]=-2; E[1,3,3]=2; E[2,3,3]=-3; E
    [3,3,3]=-1;
push!(List,E)
#45
E=zeros(3,3,3)
E[1,1,1]=8; E[2,1,1]=-3; E[3,1,1]=-1; E[2,2,1]=-2; E[3,2,1]=2; E
    [2,3,1]=-1; E[3,3,1]=1; E[2,1,2]=-2; E[3,1,2]=2; E[1,2,2]=2; E
    [2,2,2]=-2; E[3,2,2]=-2; E[1,3,2]=2; E[2,3,2]=-2; E[3,3,2]=-2; E
    [2,1,3]=-1; E[3,1,3]=1; E[1,2,3]=2; E[2,2,3]=-2; E[3,2,3]=-2; E
    [1,3,3]=-2; E[2,3,3]=3; E[3,3,3]=1;
push!(List,E)
#46
E=zeros(3,3,3)
E[1,1,1]=10; E[2,1,1]=-3; E[3,1,1]=-1; E[1,2,1]=-3; E[2,2,1]=2; E
    [3,2,1]=1; E[1,3,1]=-1; E[2,3,1]=1; E[3,3,1]=2; E[2,1,2]=-2; E
    [3,1,2]=2; E[1,2,2]=-1; E[2,2,2]=3; E[3,2,2]=-4; E[1,3,2]=-1; E
    [2,3,2]=1; E[3,3,2]=-2; E[2,1,3]=-1; E[3,1,3]=-1; E[1,2,3]=-2; E
    [2,2,3]=3; E[3,2,3]=1; E[1,3,3]=2; E[2,3,3]=-4; E[3,3,3]=-2;
push!(List,E)

```

```
189 L = zeros(46,1)
190
191 for i=1:46
192     L[i] = List[i][1,1,1]
193 end
194
195 List = -List;
196
197 Q = [1, 4, 2sqrt(2), 4sqrt(2)-2, 8sqrt(5)-13 , 4sqrt(2)-1, 20/3 , 20/3 ,
        4sqrt(2), 4 , 4sqrt(2), 4sqrt(2), 4sqrt(2), 4sqrt(2), 6 , 6.12883 , 4
        sqrt(2) , 2*(7-sqrt(17)) , 5.7829 , 6sqrt(2)-2, 5.95539 , 6.19794 ,
        (3/2)*(sqrt(17)-1), 7.94016 , 6.82421 , 1+4sqrt(3), 6.95465 , 9.90976
        , 8sqrt(2)-2, 8sqrt(2)-2, 7.80425 , 8.15161 , 9.78988 , 8.25142 ,
        7.85524 , 9.46139 , 8sqrt(2)-2, 8sqrt(2)-2, 9.32530 , 8.12983 ,
        10.3680 , 13.0471 , 8sqrt(2), 12sqrt(2)-4, 12sqrt(2)-4, 12.9852]
198 reshape(Q,46,1)
199
200 return(List,L,Q);
201
202 end
```