

Bucknell University

Bucknell Digital Commons

Honors Theses

Student Theses

Spring 2022

A Contraction Based Approach to Tensor Isomorphism

Anh Kieu

aqk001@bucknell.edu

Follow this and additional works at: https://digitalcommons.bucknell.edu/honors_theses



Part of the [Algebra Commons](#)

Recommended Citation

Kieu, Anh, "A Contraction Based Approach to Tensor Isomorphism" (2022). *Honors Theses*. 621.
https://digitalcommons.bucknell.edu/honors_theses/621

This Honors Thesis is brought to you for free and open access by the Student Theses at Bucknell Digital Commons. It has been accepted for inclusion in Honors Theses by an authorized administrator of Bucknell Digital Commons. For more information, please contact dcadmin@bucknell.edu.

A CONTRACTION BASED APPROACH TO TENSOR ISOMORPHISM

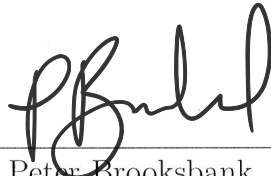
by

Anh Kieu

A Proposal Submitted to the Honors Council
For Honors in the Department of Mathematics

May 11, 2022

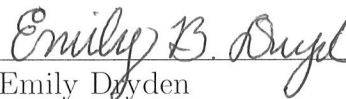
Approved:



Peter Brooksbank
Thesis Advisor



Ben Vollmayr-Lee
Second Reader



Emily Dryden
Chair, Department of Mathematics

Acknowledgements

I would like to first thank my advisor, Prof. Brooksbank, for his support and guidance throughout the year. Working with him has been a great pleasure; Prof. Brooksbank was always there to answer all of my questions and listen to every worry or story that I had. Doing this honor thesis was a great experience, and I could not have completed the thesis without him. I also want to thank Prof. Vollmayr-Lee and Prof. Stayton for being in my honor thesis committee and giving me all the valuable comments to improve my thesis.

I would like to give thanks to my family; to Chinh Kieu, Quynh Hoang, and Son Kieu, you guys made my education at Bucknell possible and have been a great support during all my 5 years of college. Then, to my friend groups, the boys and the VSA, I am glad that I have met such wonderful people and receive so much love, care and support from you. 5 years seem like a long time, but you guys have made my college experience so much more enjoyable and memorable.

Finally, to all my professors and the department assistants, Polly Doyle and April Ritter, thank you so much for being a part of my Bucknell journey. I will miss you all dearly!

Contents

Abstract	7
1 Introduction	8
2 Background	10
2.1 Linear transformations	11
2.2 Matrix equivalence	12
3 Tensors	14
3.1 Tensor space intuition	16
3.2 Actions on n -tensors	19
3.3 Tensor isomorphism	20
3.4 Tensor contraction	21
4 Classification of $K^2 \otimes K^2 \otimes K^2$	26
4.1 Degenerate 3-tensors	27
4.2 Nondegenerate 3-tensors	30

5	Contraction labelling	33
6	Applications to Quantum Information Theory	37
6.1	Tensor representation in QIT	37
6.1.1	Projective Geometry for \mathbb{C}^2	38
6.2	3-qubit classification	38
6.3	4-qubit state inequivalence	42
6.3.1	Procedure	43
6.3.2	Application to nilpotent classes	46
7	Conclusion	50
	Appendix A Procedure for 4-tensor labelling	52

List of Figures

3.1	Timetable of bus routes (source: Transfort); retrieved from [6]	14
3.2	Nutrition table for cultivated fruits (source USDA); retrieved from [6]	15
3.3	Visualization of 3-tensor with dimension $a \times b \times c$	18
3.4	Visualization of t_{fnc} contraction process along f axis; retrieved from [6]	22
4.1	Axis direction for 3-tensor visualization.	27
4.2	General 3-tensor representation	27
4.3	Visualization of 3-tensor with radicals in 3 axes.	28
4.4	Visualization of 3-tensor with a radical along axis 1.	29
4.5	Different types of 3-tensor with radical(s).	29
4.6	Complete classification of 3-tensors in $K^2 \otimes K^2 \otimes K^2$	32
5.1	Visualization of a collineation g	35
6.1	$ \text{GHZ}\rangle = 000\rangle + 111\rangle$	39
6.2	$ \text{W}\rangle = 001\rangle + 010\rangle + 100\rangle$	39

6.3	Visualization of $\mathbf{C}(x)$ as contraction	44
-----	---	----

List of Tables

6.1	The complete list of 3-qubit equivalence classes	40
6.2	Nilpotent classes for 4-qubit states from [1]	47
6.3	Axis labels of orbit 17 and orbit 18	49

Abstract

Tensor isomorphism is a hard problem in computational complexity theory. Tensor isomorphism arises not just in mathematics, but also in other applied fields like Machine Learning, Cryptography, and Quantum Information Theory (QIT). In this thesis, we develop a new approach to testing (non)-isomorphism of tensors that uses local information from “contractions” of a tensor to detect differences in global structures. Specifically, we use projective geometry and tensor contractions to create a labelling data structure for a given tensor, which can be used to compare and distinguish tensors. This contraction labelling isomorphism test is quite general, and its practical potential remains largely unexplored. As a proof of concept, however, we apply the technique to a very recent classification of 4-qubit states in QIT.

Keywords: Tensor isomorphism, tensor contraction, projective geometry

Chapter 1

Introduction

Tensors are multi-dimensional mathematical objects that can be represented as multi-way arrays. Since data in real life have different variables or parameters, tensors are often used in a lot of applied fields to record data. One fundamental problem that arises is tensor isomorphism: how can we tell if two tensors are equivalent?

Tensor isomorphism is, in general, a hard problem to solve. However, it is also a problem of considerable interest in many fields such as Machine Learning, Cryptography, and especially Quantum Information Theory (QIT). In QIT, researchers want to know when two states of quantum bits (qubits) are equivalent because, in principle, equivalent states can perform the same computational function. By telling when the qubit states are equivalent, researchers can eliminate redundancy. In 2008, W. Dür, G. Vidal and J. I. Cirac provided a classification for 3-qubit states [2]. Some efforts were made to classify 4-qubit states in the following years [4], but not until recently was there a complete classification of 4-qubit system states [1].

Since tensor isomorphism is an important problem in a lot of fields, the aim of this thesis is to develop a new approach to the tensor isomorphism problem. Specifically, we want to construct a method to tell when two tensors are not equivalent. The thesis will introduce the readers to our approach to the tensor (non)-isomorphism problem, and, as a proof of concept, we will apply the method to 4-qubit states in QIT.

The outline of the thesis is as follows: In Chapter 2, we will go over the necessary background on linear transformations and matrix equivalence. In Chapter 3, we will introduce tensors and tensor concepts such as tensor isomorphism and tensor

contraction. We will apply those tensor concepts to derive a known classification of tensors in $K^2 \otimes K^2 \otimes K^2$ space in Chapter 4. In Chapter 5, we introduce our contraction based approach to the tensor isomorphism. Then, in Chapter 6, we apply our methods to both the 3-qubit classification problem and the 4-qubit inequivalence problem, and present our results. Lastly, we summarize our work and discuss future directions and improvements.

Chapter 2

Background

In this chapter, we will expose the reader to the concept of linear transformation, which leads naturally to a notion of matrix equivalence. As matrices are special cases of tensors, being able to understand matrix equivalence will help with the reader's intuition of tensor equivalence.

For convenience, we will index coordinates of vectors, matrices, and tensors starting from 0 to match with existing notations in our main application, quantum information theory. In particular, we denote the entry in row i and column j of an $m \times n$ matrix A by a_{ij} , where $i \in [m]$ and $j \in [n]$. Here, $[m] = \{0, 1, \dots, m - 1\}$.

A *vector space* is simply a set of vectors, equipped with addition and scalar multiplication operations that satisfy certain familiar properties. Vectors in a vector space can be written uniquely as linear combination of some core vectors called a *basis*. The size of the basis will determine the *dimension* of the vector space. A vector space can have many bases and each basis will allow the vectors in the vector space to be represented differently. In this thesis, we will work exclusively with K^n , which is the vector space over a field K of column vectors of dimension n . For a vector space K^n , $\{e_0, e_1, \dots, e_n\}$ is called the standard basis for K^n with e_i being a vector with 1 at position i and 0 everywhere else.

2.1 Linear transformations

Let K^m and K^n be two vector spaces. Let $\mathcal{B}_m = \{u_0, \dots, u_{m-1}\}$ be the standard basis for K^m and $\mathcal{B}_n = \{v_0, \dots, v_{n-1}\}$ be the standard basis for K^n . Let $T : K^m \rightarrow K^n$ be a linear transformation, meaning that $\forall x, v \in K^n$ and $\forall x \in K$, $T(u + xv) = T(u) + xT(v)$. We can represent T as an $m \times n$ matrix A , such that, $T(u) = A \cdot u$.

Since $T(u_i) \in K^n$, $T(u_i) = \sum_{j=0}^{n-1} a_{ij}v_j$ for some a_{ij} . Thus, let A be the matrix with entries a_{ij} . In other words, the i -th column of A is $T(u_i)$. We claim that this matrix A will give us our representation of T with respect to \mathcal{B}_m and \mathcal{B}_n .

Lemma 2.1.1. $\forall u \in K^m$, there exists a matrix A such that $T(u) = A \cdot u$.

Proof. Let $u \in K^m$. Thus, $u = \sum_{i=0}^{m-1} x_i u_i$ for some $x_i \in K$. Applying linear transformation T to u , we obtain

$$\begin{aligned} T(u) &= T\left(\sum_{i=0}^{m-1} x_i u_i\right) \\ &= \sum_{i=0}^{m-1} x_i T(u_i) \text{ (linearity)}. \end{aligned}$$

On the other hand, let A be an $m \times n$ matrix such that its i -th column is $T(u_i)$. Thus,

$$\begin{aligned} A \cdot u &= (T(u_1) \quad T(u_2) \quad \dots \quad T(u_{m-1})) \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_{m-1} \end{pmatrix} \\ &= x_1 T(u_1) + x_2 T(u_2) + \dots + x_{m-1} T(u_{m-1}) \\ &= \sum_{i=0}^{m-1} x_i T(u_i) = T(u) \end{aligned}$$

Thus, A is the representation matrix of T relative to bases \mathcal{B}_m and \mathcal{B}_n . □

2.2 Matrix equivalence

Let $T : K^m \rightarrow K^n$ be a linear transformation. By Lemma 2.1.1, there exists a $m \times n$ matrix A that represents T relative to some bases \mathcal{B}_m and \mathcal{B}_n . Suppose we changed our “reference frame” by selecting a different pair of bases \mathcal{B}'_m and \mathcal{B}'_n for K^m and K^n respectively. Then, T would be represented by a different matrix A' . Since A and A' are representations with respect to different bases, A and A' have different matrix entries. However, A and A' are both matrix representations of the linear transformation T ; therefore, A and A' are considered *equivalent*. This difference in the reference frame is due to changes of basis between \mathcal{B}_m and \mathcal{B}'_m , and between \mathcal{B}_n and \mathcal{B}'_n , and so we define matrix equivalence as follows:

Definition 2.2.1. Let A, A' be two matrices in $K^{m \times n}$. A and A' are equivalent if there exist invertible matrices $P \in K^{m \times m}$ and $Q \in K^{n \times n}$ such that

$$A' = PAQ.$$

Here, P and Q are change of basis matrices from $\mathcal{B}_m \rightarrow \mathcal{B}'_m$ and from $\mathcal{B}_n \rightarrow \mathcal{B}'_n$ respectively. In the language of abstract algebra, P, Q are in the groups of invertible matrices, $\text{GL}(m, K)$ and $\text{GL}(n, K)$, which **act** on the vector spaces K^m and K^n , respectively. In fact, $\text{GL}(m, K)$ and $\text{GL}(n, K)$ act on the sets of bases of K^m and K^n , and therefore bring about changes in the reference frame relative to which linear transformations are recorded. In this thesis, we will frequently use the term “actions” to describe the process of exchanging one reference frame for another.

It is straightforward to evaluate whether A and A' are equivalent. We learn the techniques in MATH 245, or any introductory Linear Algebra course. If A and A' have the same rank (dimension of a matrix’s column space), then A and A' are equivalent. To find the changes of basis P and Q , we need to reduce both A and A' to a row-reduced, column-reduced form R such that

$$R = \begin{pmatrix} I_r & 0_{r \times (n-r)} \\ 0_{(m-r)} & 0_{(m-r)(n-r)} \end{pmatrix}$$

with I_r being the $r \times r$ identity matrix $\begin{pmatrix} 1 & \dots \\ \vdots & \ddots \\ \dots & \vdots \\ & & 1 \end{pmatrix}$ (1 along diagonal, 0 elsewhere). This form R is often called the *Rank Normal Form*. To obtain R , we first apply Gaussian elimination to bring A and A' to reduced-row echelon form. Then, we apply column operations to eliminate all non-leading 1’s entries and swap columns to create an

identity matrix block. Since all row and column operations can be represented as some elementary matrices E_i and F_j , we have:

$$E_1 E_2 \dots E_i A F_1 F_2 \dots F_j = E'_1 E'_2 \dots E'_k A' F_1 F_2 \dots F_l = R$$

Thus, if we let $P = (E'_1 E'_2 \dots E'_k)^{-1} E_1 E_2 \dots E_i$ and $Q = F_1 F_2 \dots F_l (F'_1 E'_2 \dots E'_k)^{-1}$, then P and Q are both invertible and $PAQ = A'$. The matrices E_1, E_2, \dots perform Elementary Row Operations (EROs) on A , and F_1, F_2, \dots perform Elementary Column Operations (ECOs).

Bringing a matrix to a Rank Normal Form is not only easy in theory, but also easy in practice as well. There are many algorithms available on standard computing platforms created to reduce a matrix to its Rank Normal Form. In general, the equivalence problem for matrices is completely solved, and the result can be easily computed.

Chapter 3

Tensors

In this chapter, we will explore the world of tensors. While not everyone is familiar with the definition, in its simplest form a tensor is just a multi-way array—a high-dimensional data structure that we encounter on a daily basis. For example, a spreadsheet containing different tables with multiple rows and columns is considered to be a multi-way array. However, it is not the shape of the spreadsheet that makes it a tensor, but rather its interpretation.

Let’s consider the following spreadsheet:

	Riverside	Stover	Mulberry	Downtown
BUS 1	6:33	6:36	6:40	6:48
BUS 2	8:33	8:36	8:40	8:48
BUS 3	10:33	10:36	10:40	10:48

Weekdays Weekends

Figure 3.1: Timetable of bus routes (source: Transfort); retrieved from [6]

This bus schedule spreadsheet is an example of a multi-way array. There are two main pages in this spreadsheet, “Weekdays” and “Weekends”. The current displayed page is “Weekdays”, containing rows of different buses and columns of their destination. The entries record the time of arrival at the destinations of each bus. The

“Weekends” page also has the same exact structure of rows and columns.

In this spreadsheet, the entries are just discrete points, and their linear combinations would not really make sense. For example, if a person wants to be at downtown around 10 am, their only possible choice is to take BUS 2. It will be more convenient for them to have an alternative bus that arrives between BUS 2 and BUS 3, i.e arrival time = $\frac{1}{2}$ BUS 2 + $\frac{1}{2}$ BUS 3, but the option does not exist in reality. In short, this spreadsheet does not offer any more information other than the recorded entries.

On the other hand, the spreadsheet in Table 3.2 can provide more than just the displayed information. Table 3.2 also has a similar layout to the bus schedule, where it has two main pages, “Conventional” and “Organic”. Each page contains rows of fruit and columns of nutrition type. The entries show how many grams or percent daily values of each nutrition type are contained in each fruit. Even though the two tables have the same page-row-column layout, the linear combinations of entries in Table 3.2 actually make sense.

	Protein	Fat	Carbs	Vit. C
Strawberry	0.7g	0.3g	7.7g	70%
Peach	0.5g	0g	26g	2%
Apple	0.3g	0.2g	13.8g	2%
Conventional	Organic			

Figure 3.2: Nutrition table for cultivated fruits (source USDA); retrieved from [6]

Consider the scenario where a consumer wants to have at least 100% intake of Vitamin C but not exceed 50g of carbs. Since they are not forced to take the whole serving size of apples or strawberries, the consumer can make a portion out of a variety of fruit. With these constraints, the consumer can have a serving size of s strawberries, p peaches and a apples such that:

$$(70\%)s + (2\%)p + (2\%)a \geq 100\% \tag{3.1}$$

and

$$(7.7g)s + (26g)p + (13.8g)a \leq 50g$$

Thus, taking linear combinations of the column entries gives us new information that has meaning.

These two examples of a bus schedule spreadsheet and a nutrition spreadsheet briefly show us that not all multi-way arrays should be interpreted in the same way. While the bus schedule entries should be read as discrete points, the nutrition table entries can be interpreted as base knowledge, and the reader can use those entries to derive more information. To be considered as a tensor, the multi-way array needs to provide more information than the surface level.

Definition 3.0.1 (Multi-way Array). A *multi-way array* is a function $\Gamma : I_1 \times \cdots \times I_n \rightarrow K$, where

- K is a field in which the entries of the array live;
- I_1, \dots, I_n are finite sets, called the *axes* of Γ ;
- n is the *valence* of Γ ; and
- $|I_j|$ is the *dimension* of axis j .

We denote the entries of the array by $\Gamma_{i_1, i_2, \dots, i_n}$ or Γ_{i_*} for short.

If $l = 2$, we obtain a multi-way array $\Gamma : I_1 \times I_2 \rightarrow K$. This is another way to express a $|I_1| \times |I_2|$ matrix with entries from K . When this interpretation of Γ as a multi-linear map carries meaningful information, then Γ is said to be a *tensor*.

3.1 Tensor space intuition

For the purpose of our application, we are only interested in studying the tensor space $\mathcal{T}_n = K^{d_1} \otimes K^{d_2} \otimes \cdots \otimes K^{d_n}$, which is just an n -tensor space that we will define later in the section. We will use \mathcal{T}_n to indicate this n -tensor space throughout the whole thesis. In this section, we will build up our intuition for what a tensor space is and what an element in a tensor space looks like. We begin with examining the tensor product of two vector spaces.

Let $v = (v_0, v_1, \dots, v_{a-1})^T \in K^a$ and $w = (w_0, w_1, \dots, w_{b-1})^T \in K^b$. The tensor product is a function $\otimes : K^a \times K^b \rightarrow K^{a \times b}$, where $v \otimes w = v \cdot w^T$. Thus, $v \otimes w$ will

have entry $(v \otimes w)_{ij} = v_i w_j$.

$$\begin{pmatrix} v_0 \\ v_1 \\ \vdots \\ v_{a-1} \end{pmatrix} \otimes \begin{pmatrix} w_0 \\ w_1 \\ \vdots \\ w_{b-1} \end{pmatrix} = \begin{pmatrix} v_0 \\ v_1 \\ \vdots \\ v_{a-1} \end{pmatrix} \cdot (w_0 \ w_1 \ \cdots \ w_{b-1}) = \begin{pmatrix} v_0 w_0 & v_0 w_1 & \cdots & v_0 w_{b-1} \\ v_1 w_0 & v_1 w_1 & \cdots & v_1 w_{b-1} \\ \vdots & \vdots & \vdots & \vdots \\ v_{a-1} w_0 & v_{a-1} w_1 & \cdots & v_{a-1} w_{b-1} \end{pmatrix}$$

Example 3.1.1. Let $v = \begin{pmatrix} 0 \\ 1 \end{pmatrix}$ and $w = \begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix}$. Thus, $v \otimes w = v \cdot w^T = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \end{pmatrix}$

Example 3.1.2. Let $v = \begin{pmatrix} 2 \\ 0 \\ 1 \end{pmatrix}$ and $w = \begin{pmatrix} 4 \\ 3 \end{pmatrix}$. Thus, $v \otimes w = \begin{pmatrix} 8 & 6 \\ 0 & 0 \\ 4 & 3 \end{pmatrix}$

Notice that, when we take a tensor product of an a -vector and a b -vector, we obtain an $a \times b$ matrix. Thus, the dimension of the resulting matrix space is the product of the input vector space dimensions.

Let $\{e_0, \dots, e_{m-1}\}$ be a basis for K^m and $\{f_0, \dots, f_{n-1}\}$ be a basis for K^n . Then, applying tensor product to the pair (e_i, f_j) for each pair of i, j , we get a matrix $e_i \otimes f_j$. We define $K^m \otimes K^n$ to be the K -linear span of these matrices $e_i \otimes f_j$. In other words, if we take every linear combination of the vectors in the set $\{e_i \otimes f_j\}$ with scalars from K , we will form the tensor space $K^a \otimes K^b$. Notice that since $K^a \otimes K^b$ is the linear span of $\{e_i \otimes f_j\}$, and $e_i \otimes e_j$ is just the (i, j) -elementary matrix, $K^a \otimes K^b$ is isomorphic (in fact, equal) to the space $K^{m \times n}$ of all $a \times b$ matrices.

Let us go one step further and evaluate the tensor product of a matrix with a vector. Let $M \in M_{a \times b}$ and $u \in K^c$. By the same idea of “multiplying” dimension, we obtain an $a \times b \times c$ dimensional object, as follows:

$$\begin{aligned} M \otimes v &= \begin{pmatrix} m_{00} & m_{01} & \cdots & m_{0(b-1)} \\ m_{10} & m_{11} & \cdots & m_{1(b-1)} \\ \vdots & \vdots & \vdots & \vdots \\ m_{(a-1)0} & m_{(a-1)1} & \cdots & m_{(a-1)(b-1)} \end{pmatrix} \otimes \begin{pmatrix} u_0 \\ u_1 \\ \vdots \\ u_{c-1} \end{pmatrix} \\ &= (u_0 \cdot M \quad u_1 \cdot M \quad \cdots \quad u_j \cdot M \quad \cdots \quad u_{c-1} \cdot M) \end{aligned}$$

Visually, we can imagine $M \otimes v$ as a “stack of c multiples of M ” as in Figure 3.3.

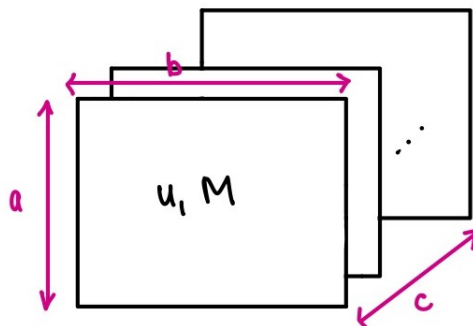
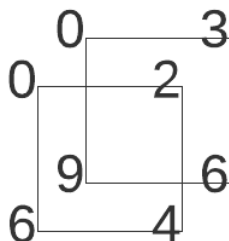


Figure 3.3: Visualization of 3-tensor with dimension $a \times b \times c$

Example 3.1.3. Let $M = \begin{pmatrix} 0 & 2 \\ 3 & 4 \end{pmatrix}$ and $u = \begin{pmatrix} 0 \\ 3 \end{pmatrix}$.

$$M \otimes u = \left(\begin{pmatrix} 0 & 2 \\ 6 & 4 \end{pmatrix} \begin{pmatrix} 0 & 3 \\ 9 & 6 \end{pmatrix} \right)$$

or more visually:



The result of a tensor product between a matrix and a vector is what we call a **3-tensor**. As illustrated in the visualization, a 3-tensor has 3 main axes, and the 3-tensor can be associated with a 3-way array. A 3-tensor lives in the tensor space $\mathcal{T}_3 = K^{d_1} \otimes K^{d_2} \otimes K^{d_3}$ with $d_1, d_2, d_3 \in \mathbb{N}$. Using the same technique of “multiplying dimension”, we can build an n -tensor space $T_n : K^{d_1} \otimes K^{d_2} \otimes \dots \otimes K^{d_n}$.

3.2 Actions on n -tensors

Tensor isomorphism is concerned with invertible linear transformations that act independently on each axis of the tensor, effecting changes of basis like we saw in the previous chapter for matrix equivalence. We will examine how tuples of matrices act on tensor axes by defining their effect on the basis elements of the tensor space and then extending linearly.

Let \mathcal{T}_n be an n -tensor space with basis elements of the form $|i_1 i_2 \dots i_n\rangle$, denoted $|i_*\rangle$ for short. The tensor $|i_1 i_2 \dots i_n\rangle$ can be interpreted as a multi-way array $e_{i_1} \otimes e_{i_2} \otimes \dots \otimes e_{i_n}$ with e_{i_j} being the j -th canonical basis vector of K^{d_j} . This multi-way array has entry 1 at position i_1, i_2, \dots, i_n and 0 everywhere else.

Thus, each n -tensor can be written as a linear combination of the basis elements:

$$|\phi\rangle = \sum_{i_*} \alpha_* |i_*\rangle$$

When $n = 1$, \mathcal{T}_n is just K^{d_1} , with basis $|i_1\rangle$. Since K^{d_1} just a column vector space, a tensor $|\phi\rangle$ in \mathcal{T}_1 is a column vector. The action of a matrix A on $|\phi\rangle$ is just the matrix-vector product $|\phi\rangle \rightarrow A|\phi\rangle$.

Notice that A maps $|i_1\rangle$ to a linear combination of $|i_*\rangle$. To demonstrate how this mapping works, let's examine A 's action on the basis elements of the 1-tensor space K^2 . Let $A = \begin{pmatrix} a & b \\ c & d \end{pmatrix} \in \text{GL}(2, K)$. With $|0\rangle$ and $|1\rangle$ being the basis vectors of K^2 ,

$$\begin{aligned} A|0\rangle &= A \cdot e_0 \\ &= A \cdot \begin{pmatrix} 1 \\ 0 \end{pmatrix} \\ &= \begin{pmatrix} a & b \\ c & d \end{pmatrix} \cdot \begin{pmatrix} 1 \\ 0 \end{pmatrix} \\ &= \begin{pmatrix} a \\ c \end{pmatrix} \end{aligned}$$

Thus, $|0\rangle \xrightarrow{A} (a|0\rangle + c|1\rangle)$. Similarly, $|1\rangle \xrightarrow{A} (b|0\rangle + d|1\rangle)$.

If $n = 2$, our tensor space becomes $\mathcal{T}_2 = K^{d_1} \otimes K^{d_2}$ with basis element of the form $|i_1 i_2\rangle$. This tensor space is again familiar: we saw earlier that it is just the matrix space $K^{d_1} \times K^{d_2}$, with the basis element $|i_1 i_2\rangle$ interpreted as a matrix $e_{i_1} \otimes e_{i_2} = e_{i_1} \cdot e_{i_2}^T$. To act on two axes, we will need two matrices. We can represent this action as a pair of matrices such that $(A_1, A_2)|i_1 i_2\rangle = (A_1 e_{i_1}) \otimes (A_2 e_{i_2}) = (A_1 e_{i_1})(A_2 e_{i_2})^T = A_1 e_{i_1} e_{i_2}^T A_2^T$.

Extending the effect of (A_1, A_2) on the basis elements $|i_1 i_2\rangle$ linearly, we obtain an action on a 2-tensors. Let $|\phi\rangle$ be associated with the matrix M_ϕ , then:

$$(A_1, A_2)|\phi\rangle = A_1 M_\phi A_2^T$$

This equation is similar to the matrix equation equivalence in Definition 2.2.1. As we mentioned, A_1 can be thought of as performing a sequence of row operations on M_ϕ , while A_2^T can be thought of as performing a sequence of column operations. Notice also that, by associativity of matrix multiplication, the actions on the first and the second axes of the tensor space commute. In the language of abstract algebra, this means there is an action of the direct product $\text{GL}(d_1, K) \times \text{GL}(d_2, K)$ on the tensor space. This commutativity property extends to tensor spaces of higher valence.

In general, $(A_1, A_2, \dots, A_n)|i_1 i_2 \dots i_n\rangle := (A_1 e_{i_1}) \otimes (A_2 e_{i_2}) \otimes \dots \otimes (A_n e_{i_n})$. This again extends linearly to give an action on the entire tensor space.

For 2-tensors, actions on axis 1 and 2 correspond to different EROs and ECOs respectively. Similarly, the actions on 3-tensors represent 3 different kinds of “face” operations. Similar to how we can swap rows, multiply a row by a scalar, or add a row into another, we can swap tensor face, multiply a face by a scalar, or add a face into another for 3-tensor. Later on, we will use this concept of face actions to prove some important lemmas in Chapter 4.

3.3 Tensor isomorphism

Definition 3.3.1 (Tensor isomorphism). Let $|\phi\rangle, |\psi\rangle \in K^{d_1} \otimes \dots \otimes K^{d_n}$. The tensors $|\phi\rangle$ and $|\psi\rangle$ are **isomorphic** if there exists $(A_1, \dots, A_n) \in \text{GL}(d_1, K) \times \dots \times \text{GL}(d_n, K)$ such that

$$(A_1, A_2, \dots, A_n)|\phi\rangle = |\psi\rangle$$

In other words, if $|\phi\rangle$ and $|\psi\rangle$ are isomorphic, we can obtain $|\psi\rangle$ by applying various actions to $|\phi\rangle$. If $|\phi\rangle$ and $|\psi\rangle$ are nonzero 1-tensors (vectors), they are isomorphic as we can always find a transformation that takes one non-zero vector to another. If $|\phi\rangle$ and $|\psi\rangle$ are 2-tensors, $|\phi\rangle$ and $|\psi\rangle$ are isomorphic when they have the same rank as matrices (see Chapter 2).

However, moving from 2-tensors to 3-tensors, the isomorphism problem becomes extremely difficult in general. In fact, a recent result in computational complexity

theory shows that the isomorphism problem for general tensors reduces to the 3-tensor case [3]. Nevertheless, this thesis still contributes to the tensor isomorphism problem by providing a practical, computational method to identify non-isomorphic tensors.

3.4 Tensor contraction

Let's revisit the nutrition spreadsheet in the chapter introduction. To obtain the inequality in Equation 3.1, we need to multiply our column of vitamin C amount in the "Conventional" page with a vector $v_f = (s, p, a)$. We do the same thing with the carbs amount column in the same page. The condensed information from the nutrition spreadsheet thus obtained is known as a *contraction* of the spreadsheet.

As n -tensors are the tensor product of $(n - 1)$ -tensors and column vectors, we observe that for every $|\phi\rangle$, there always exist $|\phi_i\rangle$'s such that

$$|\phi\rangle = \sum_{i=0}^{d_1-1} |i\rangle|\phi_i\rangle$$

For example, consider the 2-tensor (matrix) $|m\rangle = \begin{pmatrix} a & b \\ c & d \end{pmatrix} \in K^2 \otimes K^2$. By tensor construction, $|m\rangle = a|00\rangle + b|01\rangle + c|10\rangle + d|01\rangle$. Let $|i\rangle|j\rangle = e_i \otimes e_j = |ij\rangle$. At the same time, $|m\rangle$ can also be written as:

$$\begin{aligned} |m\rangle &= a|0\rangle|0\rangle + b|0\rangle|1\rangle + c|1\rangle|0\rangle + d|1\rangle|1\rangle \\ &= |0\rangle(a|0\rangle + b|1\rangle) + |1\rangle(c|0\rangle + d|1\rangle) \\ &= \sum_{i=0}^1 |i\rangle|m_i\rangle \end{aligned}$$

for $|m_0\rangle = a|0\rangle + b|1\rangle$ and $|m_1\rangle = c|0\rangle + d|1\rangle$.

Definition 3.4.1. Let $|\phi\rangle \in \mathcal{T}_n$ be a tensor such that $|\phi\rangle = \sum_{i=0}^{d_1-1} |i\rangle|\phi_i\rangle$. Let $v = (\alpha_0, \dots, \alpha_{d_1-1})^T \in K^{d_1}$. The v -contraction along axis 1 of $|\phi\rangle$ is defined to be:

$$\langle v|\phi\rangle := \sum_{i=0}^{d_1-1} \alpha_i |\phi_i\rangle \in \mathcal{T}_{n-1}$$

Note that since $|\phi_i\rangle$'s are $(n - 1)$ -tensors, $\langle v|\phi\rangle$ is also an $(n - 1)$ -tensor. This definition can be extended to contractions along any axis, not just along axis 1. We

denote the general v -contraction of $|\phi\rangle$ along axis j as $\langle v^{(j)}|\phi\rangle$. If we don't denote the axis, we assume it is a contraction along the first axis, or the axis does not matter in the context.

To visualize the process of forming contractions, we go back to our nutrition spreadsheet example. This spreadsheet is a $4 \times 3 \times 2$ multi-way array with 3 main axes: fruit type, nutrition type, and cultivation. We denote our spreadsheet as $|t_{fnc}\rangle$. Our v_f -contraction along the the fruit axis will yield the following 4×2 multi-way array (colored part):

$$\langle v_f^{(2)}|t_{fnc}\rangle = \begin{array}{|c|c|c|} \hline & \text{Conventional} & \text{Organic} \\ \hline \text{Protein} & 0.7s + 0.5p + 0.3a & \dots \\ \text{Fat} & 0.3s + 0p + 0.2a & \dots \\ \text{Carbs} & 7.7s + 26p + 13.8a & \dots \\ \text{Carbs} & 70s + 2p + 2a & \dots \\ \hline \end{array}$$

Let's visualize how we get this table. Since we are contracting along the “fruit type” axis, we slice our spreadsheet to 3 different slices corresponding to 3 dimensions in the axis. Each slice now will have dimension 4×2 . Then, we form $\langle v_f^{(2)}|t_{fnc}\rangle$ by taking a linear combination of the slices with coefficients from v_f . The visualization of this contraction is described in Figure 3.4.

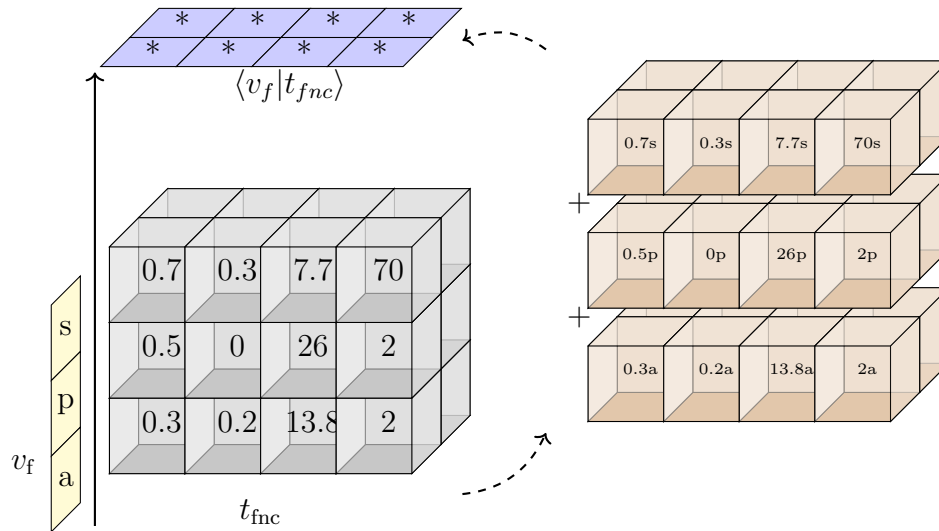


Figure 3.4: Visualization of t_{fnc} contraction process along f axis; retrieved from [6]

To obtain the right-hand side of the inequality in 3.1, we need to contract once

more along the cultivation axis. Specifically, we need to obtain just the “conventional” section from our current contraction. As the “conventional” table is the first tab of the spreadsheet, the “conventional” table is considered our first basis vector for the spreadsheet. Let $v_c = (1, 0)$. Thus, the v_c -contraction of $\langle v_f^{(2)} | t_{fnc} \rangle$ is the following 4-vector:

$$\langle v_c^{(3)} | \langle v_f^{(2)} | \phi \rangle = \begin{array}{|c|c|} \hline & \text{Conventional} \\ \hline \text{Protein} & 0.7s + 0.5p + 0.3a \\ \text{Fat} & 0.3s + 0p + 0.2a \\ \text{Carbs} & 7.7s + 26p + 13.8a \\ \text{Carbs} & 70s + 2p + 2a \\ \hline \end{array} \quad (3.2)$$

From here, we can easily obtain our Vitamin C linear combination by taking the 3rd entry of this contraction (the act of taking an entry from an array itself is also a contraction).

Notice that the order in which we perform the contractions does not matter; we can contract along the “fruit type” axis first or the “cultivation” axis first, and still yield the same result. If we contract the v_c -contraction of $|t_{fnc}\rangle$ along the “cultivation” axis first, we will obtain a 4×3 multi-way array $\langle v_c^{(3)} | t_{fnc} \rangle$ with “nutrition type” and “fruit type” axes. Then, if we further contract $\langle v_c^{(3)} | t_{fnc} \rangle$ with v_f along the “fruit type” axis, we would receive a 4-array that is identical to array in Table 3.2. This example illustrates a general property of tensors, namely that contractions along different axes commute:

$$\langle v^{(i)} | \langle u^{(j)} | \phi \rangle = \langle u^{(j)} | \langle v^{(i)} | \phi \rangle$$

for $v \in K^{d_i}$, $u \in K^{d_j}$, and $|\phi\rangle \in \mathcal{T}_n$.

Additionally, for all axes, contractions along the same axis preserve addition and scalar multiplication. That is,

$$\begin{aligned} \langle v_1 | \phi \rangle + \langle v_2 | \phi \rangle &= \langle v_1 + v_2 | \phi \rangle \\ \alpha \langle v | \phi \rangle &= \langle \alpha v | \phi \rangle \end{aligned}$$

Next, we study the behavior of contractions under basis changing actions. Let’s revisit the case of a 2-tensor (matrix) in $K^2 \otimes K^2$.

Lemma 3.4.1. *Let $v \in K^2$, $M \in \text{GL}(2, K)$ and $|\phi\rangle \in \mathcal{T}_2$. Then, $\langle v | (M, I) | \phi \rangle = \langle M^T v | \phi \rangle$ with I being the 2×2 identity matrix.*

Proof. Let $v = \begin{pmatrix} v_0 \\ v_1 \end{pmatrix}$ and $M = \begin{pmatrix} a & b \\ c & d \end{pmatrix}$. By definition,

$$\begin{aligned} |\phi\rangle &= \sum_{i_*} \alpha_{i_*} |i_*\rangle \\ &= \alpha_{00}|00\rangle + \alpha_{01}|01\rangle + \alpha_{10}|10\rangle + \alpha_{11}|11\rangle \\ &= |0\rangle(\alpha_{00}|0\rangle + \alpha_{01}|1\rangle) + |1\rangle(\alpha_{10}|0\rangle + \alpha_{11}|1\rangle) \end{aligned}$$

Thus,

$$\begin{aligned} (M, I)|\phi\rangle &= (a|0\rangle + c|1\rangle)(\alpha_{00}|0\rangle + \alpha_{01}|1\rangle) + (b|0\rangle + d|1\rangle)(\alpha_{10}|0\rangle + \alpha_{11}|1\rangle) \\ &= |0\rangle(a\alpha_{00}|0\rangle + a\alpha_{01}|1\rangle + b\alpha_{10}|0\rangle + b\alpha_{11}|1\rangle) \\ &\quad + |1\rangle(c\alpha_{00}|0\rangle + c\alpha_{01}|1\rangle + d\alpha_{10}|0\rangle + d\alpha_{11}|1\rangle), \text{ and} \\ \langle v|(M, I)|\phi\rangle &= v_0(a\alpha_{00}|0\rangle + a\alpha_{01}|1\rangle + b\alpha_{10}|0\rangle + b\alpha_{11}|1\rangle) \\ &\quad + v_1(c\alpha_{00}|0\rangle + c\alpha_{01}|1\rangle + d\alpha_{10}|0\rangle + d\alpha_{11}|1\rangle). \end{aligned}$$

On the other hand, $M^T v = \begin{pmatrix} v_0 a + v_1 c \\ v_0 b + v_1 d \end{pmatrix}$, so

$$\begin{aligned} \langle M^T v|\phi\rangle &= (v_0 a + v_1 c)(\alpha_{00}|0\rangle + \alpha_{01}|1\rangle) + (v_0 b + v_1 d)(\alpha_{10}|0\rangle + \alpha_{11}|1\rangle) \\ &= v_0(a\alpha_{00}|0\rangle + a\alpha_{01}|1\rangle + b\alpha_{10}|0\rangle + b\alpha_{11}|1\rangle) \\ &\quad + v_1(c\alpha_{00}|0\rangle + c\alpha_{01}|1\rangle + d\alpha_{10}|0\rangle + d\alpha_{11}|1\rangle) \\ &= \langle v|(M, I)|\phi\rangle \end{aligned}$$

□

The following proposition expresses the general situation:

Proposition 3.4.1. *Let $v \in K^{d_j}$, $M_j \in \text{GL}(d_j, K)$ for $j \in \{1, 2, \dots, n\}$ and $|\phi\rangle \in \mathcal{T}_n$. Then, $\langle v^{(j)}|(M_1, M_2, \dots, M_n)|\phi\rangle = (M_1, \dots, M_{j-1}, M_{j+1}, \dots, M_n)\langle (M_j^T v)^{(j)}|\phi\rangle$*

One other tool that will help us decide isomorphism is *tensor radicals*.

Definition 3.4.2 (Radical). Let $|\phi\rangle \in \mathcal{T}_n$. We define the j -radical of $|\phi\rangle$ to be $\text{rad}_j(|\phi\rangle) = \{v \in K^{d_j} | \langle v^{(j)}|\phi\rangle = 0\}$.

Definition 3.4.3 (Non-degeneracy). $|\phi\rangle$ is said to be nondegenerate when $\text{rad}_j(|\phi\rangle) = 0$.

Lemma 3.4.2. *Let $|\phi\rangle$ and $|\psi\rangle$ be isomorphic tensors. If (A_1, \dots, A_n) is an isomorphism from $|\phi\rangle$ to $|\psi\rangle$ then $v \rightarrow A_j^{-T}v$ is a linear isomorphism from $\text{rad}_j(|\phi\rangle)$ to $\text{rad}_j(|\psi\rangle)$.*

Proof. Without loss of generality, let $j = 1$. Let $u \in K^{d_1}$ such that $u = (A_1^{-T})v$. Thus,

$$\begin{aligned}
 \langle u|\psi\rangle &= \langle u|(A_1, \dots, A_n)|\phi\rangle \\
 &= (A_2, \dots, A_n)\langle A_1^T u|\phi\rangle \\
 &= (A_2, \dots, A_n)\langle (A_1)^T (M_1^{-T})v|\phi\rangle \\
 &= (A_2, \dots, A_n)\langle Iv|\phi\rangle \\
 &= (A_2, \dots, A_n)\langle v|\phi\rangle = 0
 \end{aligned}$$

Hence, there exists a radical point u along axis 1 of $|\psi\rangle$. □

Thus, (non)-degeneracy is a tensor isomorphism invariant. In particular, if two tensors $|\phi\rangle$ and $|\psi\rangle$ are isomorphic and $|\phi\rangle$ has a non-trivial radical along axis j , then $|\phi\rangle$ also has a non-trivial radical along axis j . Similarly, $|\phi\rangle$ is nondegenerate if and only if $|\psi\rangle$ is nondegenerate.

Chapter 4

Classification of $K^2 \otimes K^2 \otimes K^2$

We have already noted that 3-tensor isomorphism is as hard as the general tensor isomorphism problem. However, if we restrict to the tensor space $K^2 \otimes K^2 \otimes K^2$, not only can we solve the isomorphism problem, but we can also give an effective classification of 3-tensors up to isomorphism. The result is folklore in mathematics and would likely have been known by Kronecker in the late nineteenth century. The result for this tensor space over complex field \mathbb{C} has also been reproved in QIT in the context of 3-qubit states. In this chapter, we revisit this result for completeness, and to pave the way for our application of tensor contraction to 4-qubit states.

For consistency, we will establish some notations for tensors and tensor visualization. Let $|\phi\rangle \in \mathcal{T}_n = K^{d_1} \otimes \dots \otimes K^{d_n}$. We denote the multi-way array representations for the contractions $\langle e_0 | \phi \rangle, \dots, \langle e_{d_1-1} | \phi \rangle$ along axis 1 by $\Phi_0, \dots, \Phi_{d_1-1}$. We think of, and often refer to the Φ_i as “basis contractions” along axis 1 of $|\phi\rangle$ as well. When needed, the basis contractions along another axis will have a superscript indicating the axis, e.g. $\Phi_0^{(j)}$. Furthermore, for 3-tensors, we denote the directions of the axes in our visualization as in Figure 4.1.

Assume we have a 3-tensor $|\phi\rangle \in K^2 \otimes K^2 \otimes K^2$ represented by a cube in Figure 4.2. Hence, Φ_0 and Φ_1 will be the following matrix representations:

$$\Phi_0 = \begin{pmatrix} 0 & 1 \\ 2 & 3 \end{pmatrix} \qquad \Phi_1 = \begin{pmatrix} 4 & 5 \\ 6 & 7 \end{pmatrix}.$$

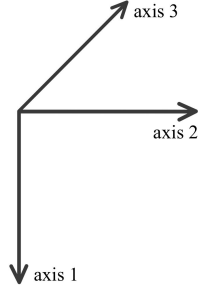


Figure 4.1: Axis direction for 3-tensor visualization.
This is similar to how our computational tool denotes axis.

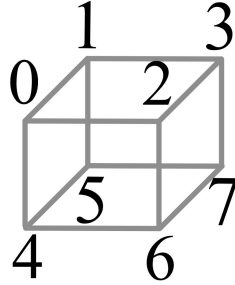


Figure 4.2: General 3-tensor representation

4.1 Degenerate 3-tensors

Let $|\phi\rangle$ be a non-zero degenerate 3-tensor; this means that

$$\exists v \neq 0, v \in K^{d_j} \quad \langle v|\phi\rangle = 0. \quad (4.1)$$

Recall that, without superscript, $\langle v|\phi\rangle$ denotes a contraction along axis 1. In this section, we will work with axis 1 only; other axes are handled identically.

Lemma 4.1.1. *If $|\phi\rangle$ has a radical along axis 1, there exists an isomorphic tensor $|\psi\rangle$ to $|\phi\rangle$ that has $\langle e_0|\psi\rangle = 0$.*

Proof. Since $|\phi\rangle$ has a radical along axis 1, there exists non-zero $v = \begin{pmatrix} a \\ b \end{pmatrix}$ such that $\langle v|\phi\rangle = 0$. Thus, $a\langle e_0|\phi\rangle + b\langle e_1|\phi\rangle = 0$. Let $|\psi\rangle = (A, I_2, I_2)\phi$. For each case of v , we want to find A such that $\langle e_0|\psi\rangle = 0$.

If $a = 0$, then $|\phi\rangle$ has $b\langle e_1|\phi\rangle = 0$. Let $A = \begin{pmatrix} 0 & b \\ 1 & 0 \end{pmatrix}$. Then, $A^T e_0 = \begin{pmatrix} 0 & 1 \\ b & 0 \end{pmatrix} \cdot \begin{pmatrix} 1 \\ 0 \end{pmatrix} = \begin{pmatrix} 0 \\ b \end{pmatrix} = b \cdot e_1$. Thus, $\langle e_0|\psi\rangle = \langle e_0|(A, I_2, I_2)|\phi\rangle = \langle A^T e_0|\phi\rangle = \langle b \cdot e_1|\phi\rangle = b\langle e_1|\phi\rangle = 0$.

If $b = 0$, then $|\phi\rangle$ has $a\langle e_0|\phi\rangle = 0$. Let $A = \begin{pmatrix} a & 0 \\ 0 & 1 \end{pmatrix}$. Then, $A^T e_0 = \begin{pmatrix} a & 0 \\ 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 \\ 0 \end{pmatrix} = \begin{pmatrix} a \\ 0 \end{pmatrix} = a \cdot e_0$. Thus, $\langle e_0|\psi\rangle = \langle e_0|(A, I_2, I_2)|\phi\rangle = \langle A^T e_0|\phi\rangle = \langle a \cdot e_0|\phi\rangle = a\langle e_0|\phi\rangle = 0$.

Lastly, assume $a, b \neq 0$. Then, $a\langle e_0|\phi\rangle + b\langle e_1|\phi\rangle = 0$, or similarly, $\langle e_0|\phi\rangle + \frac{b}{a}\langle e_1|\phi\rangle = 0$. Let $A = \begin{pmatrix} 1 & \frac{b}{a} \\ 0 & 1 \end{pmatrix}$. Then, $A^T e_0 = \begin{pmatrix} 1 & 0 \\ \frac{b}{a} & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 \\ 0 \end{pmatrix} = \begin{pmatrix} 1 \\ \frac{b}{a} \end{pmatrix} = e_0 + \frac{b}{a}e_1$. Thus, $\langle e_0|\psi\rangle = \langle e_0|(A, I_2, I_2)|\phi\rangle = \langle A^T e_0|\phi\rangle = \langle e_0 + \frac{b}{a}e_1|\phi\rangle = \langle e_0|\phi\rangle + \frac{b}{a}\langle e_1|\phi\rangle = 0$.

Since A in every case is an invertible matrix, $|\psi\rangle \cong |\phi\rangle$. Therefore, there is always an isomorphic tensor to $|\phi\rangle$ such that its e_0 -contraction is 0. \square

We can assume by Lemma 4.1.1 that the contraction $\langle e_0|\phi\rangle = 0$, so that $\Phi_0 = 0$. Notice that however we act on axis 2 and axis 3, we will still have $\Phi_0 = 0$. From Chapter 2, we know that there exist matrices A_2 and A_3 that take Φ_0 to its corresponding Rank Normal Form. Equivalently, we can act on axis 2 and axis 3 of $|\phi\rangle$ to obtain $\Phi_1 = \begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix}$, $\begin{pmatrix} 0 & 0 \\ 0 & 1 \end{pmatrix}$, or $\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$. The three potential Rank Normal Form matrices correspond to three different classes for $|\phi\rangle$.

1. Suppose $\Phi_1 = \begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix}$. Then, $|\phi\rangle = 0$ as $\Phi_0 = 0$ as well.
2. Suppose $\Phi_1 = \begin{pmatrix} 0 & 0 \\ 0 & 1 \end{pmatrix}$. Then, $|\phi\rangle$ can be represented as in Figure 4.3.

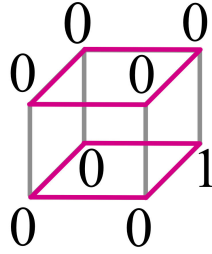


Figure 4.3: Visualization of 3-tensor with radicals in 3 axes.

Notice that $|\phi\rangle$ now also has radicals along axis 2 and 3. We refer to this case of $|\phi\rangle$ as the 3 radical class.

3. Suppose $\Phi_1 = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$. We obtain the following visualization for $|\phi\rangle$ in Figure 4.4.

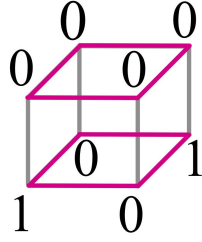


Figure 4.4: Visualization of 3-tensor with a radical along axis 1.

Here, $|\phi\rangle$ has no nontrivial radical along the other axes.

A similar analysis of radicals on the other two axes leads to the same trichotomy. However, notice that the first two cases (where Φ_1 has rank 0 or 1) are common to all three axes; only the rank 2 case leads to a new isomorphism class for each axis.

Since $|\phi\rangle$ can have radicals along one of 3 axes or along all three axes, there are 4 classes of $|\phi\rangle$ that are degenerate. Overall, we obtain the following classes of non-zero degeneracy in Figure 4.5:

- a) $|\phi\rangle$ has radicals along all axes, denoted $|\mathbb{R}_*\rangle$
- b) $|\phi\rangle$ has a radical along axis 1, denoted $|\mathbb{R}_1\rangle$
- c) $|\phi\rangle$ has a radical along axis 2, denoted $|\mathbb{R}_2\rangle$
- d) $|\phi\rangle$ has a radical along axis 3, denoted $|\mathbb{R}_3\rangle$

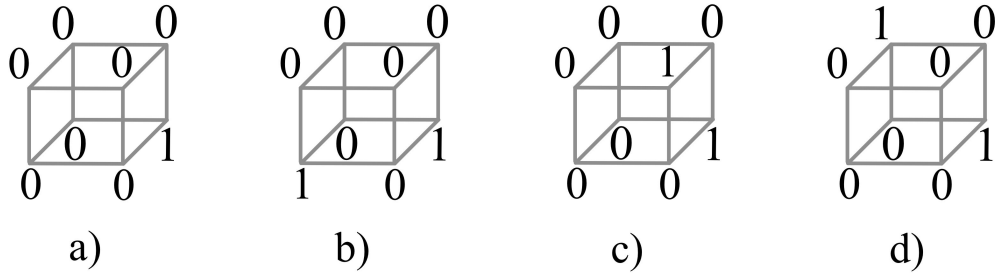


Figure 4.5: Different types of 3-tensor with radical(s).

4.2 Nondegenerate 3-tensors

We now assume that $|\phi\rangle$ is nondegenerate: there are no nontrivial radicals along any axes. In particular,

$$\forall j \in \{1, 2, 3\}, \quad \forall v \in K^2, \quad \langle e_0^{(j)} | \phi \rangle \neq 0. \quad (4.2)$$

Similar to Lemma 4.1.1, we will state the following lemma with respect to axis 1, and later expand our result to other axes as well.

Lemma 4.2.1. *If $|\phi\rangle$ is nondegenerate, there exists at least one $v \in K^2$ such that $\langle v | \phi \rangle$ is invertible.*

Proof. Let $|\phi\rangle$ be a nondegenerate 3-tensor, so $\Phi_0 = \langle e_0 | \phi \rangle$ is non-zero. If Φ_0 is invertible, we are done, so assume Φ_0 is not invertible and therefore, $\text{Rank}(\Phi_0) = 1$. Since we can row-reduce any rank 1 matrix to $\begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix}$ using a tuple of invertible matrices (I_2, A_2, A_3) , we can replace $|\phi\rangle$ with an isomorphic tensor that has $\Phi_0 = \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix}$.

Let $\Phi_1 = \begin{pmatrix} a & b \\ c & d \end{pmatrix}$. As every ERO and ECO action affects Φ_0 and Φ_1 simultaneously, to keep $\Phi_0 = \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix}$, we can only add multiples of the second row to the first row. Let's consider two cases, $d = 0$ and $d \neq 0$.

Suppose $d = 0$. If either $b = 0$ or $c = 0$, Φ_1 will have a column or a row of 0's, which makes $|\phi\rangle$ have a radical in another axis. Since $\text{Det}(\Phi_1) = ad - bc = -bc \neq 0$, Φ_1 is invertible.

Suppose $d \neq 0$. Thus, with suitable EROs and ECOs, we can add multiple of d to eliminate b and c . If $a = 0$, we can add multiple of Φ_0 to Φ_1 to make $a \neq 0$. Hence, $\text{Det}(\Phi_1) = ad \neq 0$, and so, Φ_1 is invertible. \square

Thus, we can assume Φ_0 is invertible. Moreover, using row and column-reduce actions on axis 2 and 3, we can further assume that $\Phi_0 = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$.

To complete the classification, we can again perform EROs and ECOs simultaneously on Φ_0 and Φ_1 . To keep the classification process simple, we wish to keep Φ_0 as the identity matrix and manipulate Φ_1 without changing Φ_0 . That means we can only "conjugate" by a single invertible matrix, i.e. $\Phi_i \mapsto C\Phi_i C^{-1}$. Restricting attention now to Φ_1 , this means we must consider its possible *Rational Canonical Forms*.

Rational Canonical Form (RCF) is a special form of matrix that can be obtained by conjugation by invertible matrices. A $n \times n$ matrix can only be reduced to one form of RCF, meaning that matrix M and matrix M' are similar/equivalent if they have the same RCF. RCF is also defined in terms of minimal polynomials. In the 2×2 matrices context, for each type of minimal polynomial, we get a different RCF.

For 2×2 matrices over a general field K , there are just 4 different cases of minimal polynomials $m(t)$: $m(t) = (t - a)$, $m(t) = (t - a)(t - b)$, $m(t) = t^2 - at - b$, and $m(t) = (t - a)^2$ with $a \neq b$.

Since Φ_1 cannot be a scalar multiple of Φ_0 , or the identity, there are only 3 cases for the minimal polynomial $m(t)$ and the corresponding matrix for Φ_1 .

1. $m(t) = (t - a)(t - b)$, $a, b \neq 0 \in K$

$$\Phi_0 = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \quad \Phi_1 = \begin{pmatrix} a & 0 \\ 0 & b \end{pmatrix}. \quad (4.3)$$

2. $m(t) = t^2 - at - b$, $a, b \neq 0 \in K$; $m(t)$ is irreducible over K

$$\Phi_0 = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \quad \Phi_1 = \begin{pmatrix} 0 & 1 \\ a & b \end{pmatrix}. \quad (4.4)$$

3. $m(t) = (t - a)^2$, $a \neq 0 \in K$

$$\Phi_0 = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \quad \Phi_1 = \begin{pmatrix} a & 1 \\ 0 & a \end{pmatrix}. \quad (4.5)$$

Therefore, over a general field K , there are 8 different classes of 3-tensors: The zero tensor, 4 classes of non-zero degenerate tensors, and three classes of nondegenerate tensors.

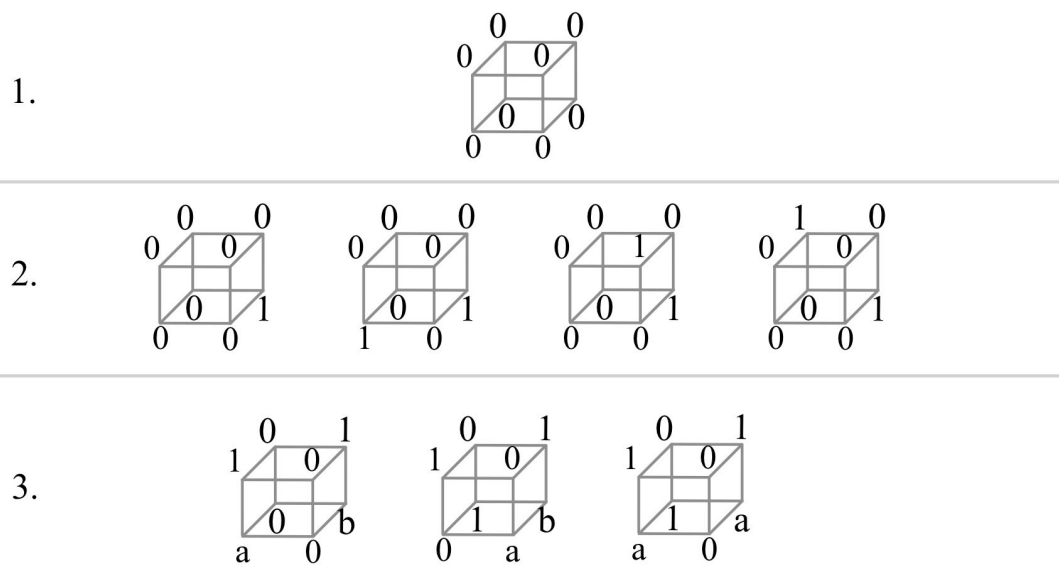


Figure 4.6: Complete classification of 3-tensors in $K^2 \otimes K^2 \otimes K^2$
 1. Zero class 2. Degenerate classes 3. Non-degenerate classes

Chapter 5

Contraction labelling

In this chapter, we will develop a general contraction-based approach to the n -tensor isomorphism problem. The method assumes access to an effective “labelling” of $(n - 1)$ -tensors that is invariant under isomorphism. Labelling all of the contractions of a tensor along each of its axes produces an isomorphism invariant that we call a “contraction label”. In particular, if two tensors are isomorphic, one should easily be able to verify that their contraction labels along all axes are compatible. Let $\lambda : \mathcal{T}_{n-1} \rightarrow \Lambda$ be a labelling function for $(n - 1)$ -tensors such that if $|t_1\rangle \cong |t_2\rangle$ then $\lambda(|t_1\rangle) = \lambda(|t_2\rangle)$.

Let’s observe two contractions $\langle v|\phi\rangle$ and $\langle \alpha v|\phi\rangle$ of $|\phi\rangle \in \mathcal{T}_n$. Since $\langle \alpha v|\phi\rangle = \alpha\langle v|\phi\rangle$, it is clear that $\langle v|\phi\rangle \cong \langle \alpha v|\phi\rangle$. As mentioned, our method relies on prior knowledge of $(n - 1)$ -tensor labels. Let Λ be the set of labels for $(n - 1)$ -tensors. In particular, for the contractions $\langle v|\phi\rangle$ and $\langle \alpha v|\phi\rangle$ of $|\phi\rangle$, we have $\lambda(\langle v|\phi\rangle) = \lambda(\langle \alpha v|\phi\rangle)$. Since v -contractions of $|\phi\rangle$ are isomorphic up to the scalar multiple of v , we don’t need to consider contractions against every single vector v in K^{d_j} . Instead, we only need to compute contractions against a representative for each 1-space $\langle v\rangle$. That is, instead of labelling contractions for all $0 \neq v \in V$, we label the points in the projective space.

Projective geometry, or projective space, is the set of all subspaces of a vector space V , denoted as $PG(V)$. The subspaces of dimension 1 (1-spaces) in $PG(V)$ are called “projective points” of V . We denote the set of “projective points” of V as $PG_0(V)$. Since subspaces of dimension 1 can be generated by a single vector $v \in V$, we associate $PG_0(V)$ with the set of all the generator vectors for the 1-spaces of v . Since projective points of V represent different 1-spaces of V , we only need to

contract the tensors against these projective points to determine the local structure of the tensor.

We can easily compute contractions for any tensor. The question is how can we use contractions of two given tensors to try to distinguish them? We continue to work just with axis 1, but again, all observations may be adapted to any axis. First, we observe that if $|\phi\rangle$ and $|\psi\rangle$ are isomorphic n -tensors, for each contraction of $|\phi\rangle$ there is an isomorphic contraction of $|\psi\rangle$.

Lemma 5.0.1. *Let $|\phi\rangle$ and $|\psi\rangle$ be isomorphic tensors in \mathcal{T}_n . If we let $\langle v|\phi\rangle$ be a contraction of $|\phi\rangle$, then there exists u such that $\langle u|\psi\rangle \cong \langle v|\phi\rangle$.*

Proof. Let $|\phi\rangle, |\psi\rangle \in \mathcal{T}_n$ such that $|\phi\rangle \cong |\psi\rangle$. Thus, there exists (M_1, M_2, \dots, M_n) with $M_j \in GL(d_j, K)$ such that $(M_1, M_2, \dots, M_n)|\phi\rangle = |\psi\rangle$. Let $\langle v|\phi\rangle$ be a contraction of ϕ along the first axis, and let $u = (M_1^{-1})^T v$. Then,

$$\begin{aligned} \langle u|\psi\rangle &= \langle u|(M_1, \dots, M_n)|\phi\rangle \\ &= (M_2, \dots, M_n)\langle M_1^T u|\phi\rangle \\ &= (M_2, \dots, M_n)\langle v|\phi\rangle \end{aligned}$$

Since M_2, \dots, M_n are invertible, $\langle u|\psi\rangle \cong \langle v|\phi\rangle$ □

Since the two contractions $\langle v|\phi\rangle$ and $\langle u|\psi\rangle$ from Lemma 5.0.1 are isomorphic, they must also have the same label. Thus, we know that for any two tensors to be isomorphic, the multi-sets of their contraction labels must be identical. Put another way, if the contraction labels do not line up, the two tensors are not isomorphic. This is the essence of our test for non-isomorphism.

We will now construct a new data structure to store the labelling of the individual projection points called *Label*. We define $Label_{|\phi\rangle}^{(j)}$ to be the set of tuples (l, S) such that $l \in \Lambda$ and $S = \{p \in PG_0(K^{d_j}) : \lambda(\langle p|\phi\rangle) = l\}$. We also write $Label_{|\phi\rangle}^{(j)}[l] := S$ when $(l, S) \in Label_{|\phi\rangle}^{(j)}$. Again, if there is no clear subscript for the axis, we assume $Label_{|\phi\rangle}$ to be the contraction labelling along axis 1.

We can easily decide tensor non-isomorphism by comparing how many contractions yield a specific label. Specifically, if we are trying to determine whether $|\phi\rangle$ and $|\psi\rangle$ are isomorphic, we first compute $Label_{|\phi\rangle}^{(j)}$ and $Label_{|\psi\rangle}^{(j)}$ for every axis j . Then, for each axis j and for each l in Λ , we will compare the sizes of $Label_{|\phi\rangle}^{(j)}[l]$ and $Label_{|\psi\rangle}^{(j)}[l]$.

If there is any mismatch among the axes labels, we will know for sure that $|\phi\rangle$ and $|\psi\rangle$ are not isomorphic.

However, comparing sizes of axis labels is not enough to guarantee isomorphism between $|\phi\rangle$ and $|\psi\rangle$. By just comparing the cardinals of the axis labels, we only take permutations of projective points into consideration. For $|\phi\rangle$ and $|\psi\rangle$ to be isomorphic, we need a label-preserving permutation that is also a *collineation* of the projective geometries. A collineation of $PG(V)$ is a bijection from $PG_0(V_\phi) \rightarrow PG_0(V_\psi)$ that preserves geometry incidence. Here, geometry incidence means collinear points in one projective geometry get mapped to collinear points in the other. In particular, a collineation will always map sets of collinear points (points that are on the same line) to sets of collinear points. Since invertible linear transformations always induce collineations, if $|\phi\rangle \cong |\psi\rangle$, there is a collineation that maps one label set to another. Therefore, even if there is a label-preserving permutation of points, if no such permutation is a collineation then $|\phi\rangle$ and $|\psi\rangle$ will not be isomorphic.

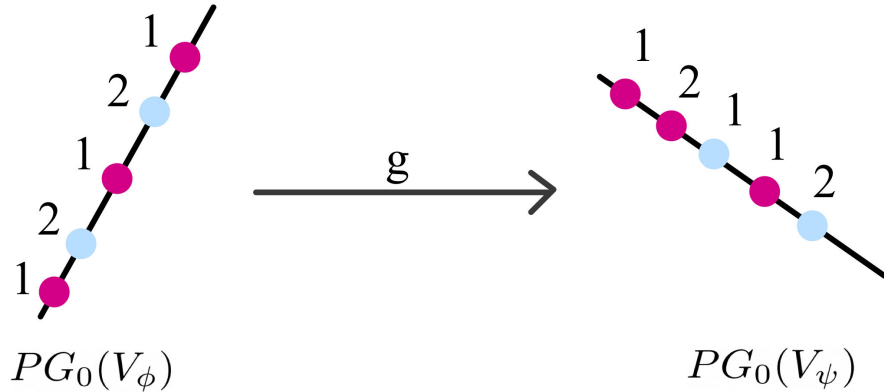


Figure 5.1: Visualization of a collineation g .

Here, we mark different labels for contractions with colors and numbers. A collineation preserves the permutation of labels, while respecting the linearity of the points. All the points that are collinear are mapped to new points that are also collinear.

Despite being quite straightforward, our method's efficacy depends on a lot of other factors. Firstly, this method is built on the assumption that we have a good labelling function for $(n-1)$ -tensors. For this method to be useful, the $(n-1)$ -tensor labelling function not only needs to be accurate, but also needs to be refined. For instance, if Λ size is small or λ maps almost every contraction to the same label, using local information will not be enough to detect any differences among the global tensor structures. Furthermore, there are potentially lots of projective points to be

covered, especially in higher dimension tensors. It is impractical or even impossible to list all the projective points, not to mention compute the contractions.

Nevertheless, there are still ways to implement this method without relying on the perfect labelling functions nor listing all the projective points. One workaround is to generate a statistical report for each tensor, with a large sample size from the projective points. With a large enough sample size, it is possible that we will be able to discern tensor structures with some statistical certainty. Another case where this method can be implemented is when we have “nicer” tensor spaces. In such “nice” tensor spaces, we are able to have a perfect labelling for $(n - 1)$ -tensors, and we don’t have to compute every projective point contraction to detect tensor non-isomorphism. One of those situations is the application to 4-qubit states, discussed in the next chapter. There already exists a perfect labelling for 3-tensors, and there are ways to determine “special contraction” without exhausting the entire set of points.

The discussion so far can be summarized by the following theorem, which establishes the correctness of our contraction labelling algorithm to detect non-isomorphism of tensors.

Theorem 5.0.1. *Any isomorphism (A_1, A_2, \dots, A_n) from n -tensor $|\phi\rangle$ to n -tensor $|\psi\rangle$ induces on each axis a label-preserving collineation of projective geometries. In particular, if it is determined that no such label-preserving collineation exists, then $|\phi\rangle$ and $|\psi\rangle$ belong to distinct isomorphism classes.*

Chapter 6

Applications to Quantum Information Theory

6.1 Tensor representation in QIT

Quantum computing studies systems of interacting particles and their states. Because of the particles' entanglement property, the states are inherently correlated, and they are typically represented as tensors.

The simplest component of a quantum system is the *qubit*. Different from a classical state, where the bit can only be either 0 or 1, a qubit can have probability of being 0 or 1. More technically, a qubit is a superposition of the state of 0 and 1. Thus, a qubit is a state vector in two-dimensional state space with two basis vectors $|0\rangle$ and $|1\rangle$. An arbitrary state vector in the state space can be written as

$$|\psi\rangle = a|0\rangle + b|1\rangle$$

where $a, b \in \mathbb{C}$ and $|\psi\rangle$ is a unit vector, such that $|a|^2 + |b|^2 = 1$.

A quantum system can have more than just 1 qubit. For example, a 2-qubit system state is a linear combination

$$|\psi\rangle = a_0|00\rangle + a_1|01\rangle + a_2|10\rangle + a_3|11\rangle$$

Similarly, an n -qubit state is a linear combination of 2^n basis state vectors of

$|i_1 i_2 \dots i_n\rangle$ where each $i_j \in \{0, 1\}$. Since we can represent the system data in a multi-way array, and linear combinations of these basis states carry physical meaning, it is natural to represent the n -qubit systems as n -tensors.

In quantum computing, it is important to know whether two states are equivalent since equivalent quantum states are, in principle, able to perform the same quantum computations. In QIT, these equivalent quantum states are considered “stochastic local operation and classical communication (SLOCC) equivalent”. The SLOCC equivalence problem has been a topic of interest in the QIT community [2, 4, 5, 7]. In tensor context, SLOCC equivalence means the same as tensor isomorphism. Thus, this question of quantum state equivalence can be answered using tools from tensor algebra.

6.1.1 Projective Geometry for \mathbb{C}^2

Our method requires us to use the projective geometry $PG(\mathbb{C}^2)$. Since \mathbb{C}^2 has dimension 2, every projective point lies on a single line, so preserving incidence is a non-issue.

Let $v = \begin{pmatrix} a \\ b \end{pmatrix}$ be a non-zero vector in \mathbb{C}^2 . If $a = 0$, $\begin{pmatrix} a \\ b \end{pmatrix} = \begin{pmatrix} 0 \\ b \end{pmatrix}$. Thus, v is in the 1-space $\langle \begin{pmatrix} 0 \\ 1 \end{pmatrix} \rangle$. If $a \neq 0$, then v is in the space $\langle \begin{pmatrix} 1 \\ c \end{pmatrix} \rangle$ with $c = \frac{b}{a}$. Since $PG_0(\mathbb{C}^2)$ contains all the 1-spaces for \mathbb{C}^2 , $\langle v \rangle \in PG_0(\mathbb{C}^2)$ has the form $\langle \begin{pmatrix} 0 \\ 1 \end{pmatrix} \rangle$ or $\langle \begin{pmatrix} 1 \\ c \end{pmatrix} \rangle$ with $c \in \mathbb{C}$. Therefore, we only consider contractions of tensors against $v = \begin{pmatrix} 0 \\ 1 \end{pmatrix}$ and $\begin{pmatrix} 1 \\ c \end{pmatrix}$ with $c \in \mathbb{C}$ in our contraction-based method.

6.2 3-qubit classification

In the case of $K = \mathbb{C}$, $m(t) = t^2 - at - b$ is always reducible. Thus, over \mathbb{C} , $m(t)$ will always have roots, and be of the form $m(t) = (t - a)(t - b)$ or $m(t) = (t - a)^2$. In the 3-qubit system context, these two non-degenerate cases for nondegenerate tensors correspond to the states $|\text{GHZ}\rangle$ and $|\text{W}\rangle$, respectively.

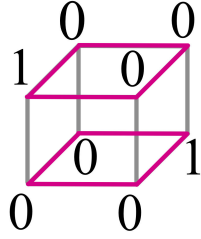


Figure 6.1: $|\text{GHZ}\rangle = |000\rangle + |111\rangle$

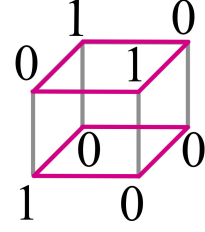


Figure 6.2: $|\text{W}\rangle = |001\rangle + |010\rangle + |100\rangle$

Lemma 6.2.1. *In our classification of $K^2 \otimes K^2 \otimes K^2$, the first nondegenerate class—where $m(t)$ has distinct roots—corresponds to the 3-qubit state $|\text{GHZ}\rangle$.*

Proof. Let $|\phi\rangle$ be a 3-tensor of the first class of nondegenerate tensor such that

$$\Phi_0^{(1)} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \quad \Phi_1^{(1)} = \begin{pmatrix} a & 0 \\ 0 & b \end{pmatrix}. \quad (6.1)$$

By equivalence, we can let $\Phi_1^{(1)} = -a\Phi_0^{(1)} + \Phi_1^{(1)} = \begin{pmatrix} 0 & 0 \\ 0 & b-a \end{pmatrix}$. Let $c = b - a$. Let $M_1 = \begin{pmatrix} 1 & -\frac{1}{c} \\ 0 & \frac{1}{c} \end{pmatrix}$. Thus,

$$\begin{aligned} (M_1, I, I)\langle e_0^{(1)}|\phi\rangle &= \langle M_1^T e_0|\phi\rangle = \langle v_0|\phi\rangle, & v_0 &= \begin{pmatrix} 1 \\ -\frac{1}{c} \end{pmatrix} \\ (M_1, I, I)\langle e_1^{(1)}|\phi\rangle &= \langle M_1^T e_1|\phi\rangle = \langle v_1|\phi\rangle, & v_1 &= \begin{pmatrix} 0 \\ \frac{1}{c} \end{pmatrix} \end{aligned}$$

Notice that $\langle v_0|\phi\rangle = \Phi_0^{(1)} - \frac{1}{c}\Phi_1^{(1)} = \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix} = |\text{GHZ}\rangle_0^{(1)}$ and $\langle v_1|\phi\rangle = \frac{1}{c}\Phi_1^{(1)} = \begin{pmatrix} 0 & 0 \\ 0 & 1 \end{pmatrix} = |\text{GHZ}\rangle_1^{(1)}$.

Thus, $(M_1, I, I)|\phi\rangle = |\text{GHZ}\rangle$. Since $M_1, I \in \text{GL}_2(K)$, $|\phi\rangle$ is equivalent to $|\text{GHZ}\rangle$. \square

Lemma 6.2.2. *In our classification of $K^2 \otimes K^2 \otimes K^2$, the third nondegenerate class—where $m(t)$ has a repeated root—corresponds to the 3-qubit state $|\text{W}\rangle$.*

Proof. Let $|\phi\rangle$ be a 3-tensor of the third class of nondegenerate tensor such that

$$\Phi_0^{(1)} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \quad \Phi_1^{(1)} = \begin{pmatrix} a & 1 \\ 0 & a \end{pmatrix}. \quad (6.2)$$

Replace $|\phi\rangle$ with an equivalent tensor such that $\Phi_0^{(1)} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$ and $\Phi_1^{(1)} = -a\mathbf{A}_0^{(1)} + \mathbf{A}_1^{(1)} = \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix}$. With $M_1 = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$, we obtain $|\psi\rangle = (M_1, I, I)\phi$ that is equivalent to ϕ .

$$\langle e_0^{(1)}|\psi\rangle = \langle M_1^T e_0|\phi\rangle = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} = |\mathbf{W}\rangle_0^{(1)} \quad \langle e_1^{(1)}|\psi\rangle = \langle M_1^T e_1|\phi\rangle = \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix} = |\mathbf{W}\rangle_1^{(1)} \quad (6.3)$$

Since the system of forms of $|\psi\rangle$ along axis 1 is the same as $|\mathbf{W}\rangle$, $|\psi\rangle$ is equivalent to $|\mathbf{W}\rangle$. Thus, $|\phi\rangle$ is also equivalent to $|\mathbf{W}\rangle$. \square

Overall, there are 7 distinct classes for 3-qubit states:

label	3-qubit state	description
0	Zero	zero tensor
1	$ \mathbf{R}_*\rangle$	degenerate: radicals in all three directions
2	$ \mathbf{R}_1\rangle$	degenerate: radical along 1-axis
3	$ \mathbf{R}_2\rangle$	degenerate: radical along 2-axis
4	$ \mathbf{R}_3\rangle$	degenerate: radical along 3-axis
5	$ \mathbf{W}\rangle$	nondegenerate: repeated eigenvalue
6	$ \mathbf{GHZ}\rangle$	nondegenerate: distinct eigenvalues

Table 6.1: The complete list of 3-qubit equivalence classes

Now, we are going to construct a procedure for identifying the class of a 3-qubit state $|\phi\rangle$. If $|\phi\rangle = \mathbf{0}$, the problem is trivial. Thus, let $|\phi\rangle$ be a non-zero tensor. Firstly, we want to compute whether $|\phi\rangle$ has a radical in any of the axes.

We start with checking whether $|\phi\rangle$ has a radical in axis 1. We will have to find if there exists a point $\mathbf{x} \in PG_0(\mathbb{C}^2)$ that give us a zero contraction. Equivalently, we need to know if there is an $\mathbf{x} = \langle(x_0, x_1)^T\rangle \in PG_0(\mathbb{C}^2)$ such that $x_0|\phi_0\rangle + x_1|\phi_1\rangle = 0$.

As mentioned in the previous chapter, we can associate $|\phi_0\rangle$ with the 2×2 “face” matrix Φ_0 , and similarly for $|\phi_1\rangle$ with Φ_1 . Moreover, $\mathbf{x} \in PG_0(\mathbb{C}^2)$ only has the form $\begin{pmatrix} 0 \\ 1 \end{pmatrix}$ or $\begin{pmatrix} 1 \\ x \end{pmatrix}$ for $x \in \mathbb{C}$. Hence, we only need to check when $\Phi_1 = 0$ or $\Phi_0 + x\Phi_1 = 0$. It is easy to check whether $\Phi_1 = 0$. To check if there exists an x for $\Phi_0 + x\Phi_1 = 0$, we simply check whether Φ_0 and Φ_1 are linearly dependent. If either $\Phi_1 = 0$ or Φ_0 and Φ_1 are linearly dependent, then $|\phi\rangle$ has a radical in axis 1.

Similarly, we can compute whether $|\phi\rangle$ has a radical in axis 2 and axis 3. If $|\phi\rangle$ has radical in just one of the axes, then $|\phi\rangle$ belongs to either class $|\mathbf{R}_1\rangle$, $|\mathbf{R}_2\rangle$ or $|\mathbf{R}_3\rangle$ respectively. Otherwise, if $|\phi\rangle$ has radicals in all axes, $|\phi\rangle$ belongs to $|\mathbf{R}_*\rangle$.

On the other hand, $|\phi\rangle$ might not have a radical in any axis. In that case $|\phi\rangle$ is nondegenerate, and hence isomorphic to $|\mathbf{W}\rangle$ or $|\mathbf{GHZ}\rangle$. Recall that basis vectors for $|\mathbf{GHZ}\rangle$ along axis 1 has the following matrix representation:

$$|\mathbf{GHZ}\rangle_0 = \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix} \quad |\mathbf{GHZ}\rangle_1 = \begin{pmatrix} 0 & 0 \\ 0 & 1 \end{pmatrix}$$

Hence, with $PG_0(\mathbb{C}^2)$, our contractions will have the form:

$$0 \cdot |\mathbf{GHZ}\rangle_0 + 1 \cdot |\mathbf{GHZ}\rangle_1 = |\mathbf{GHZ}\rangle_1 = \begin{pmatrix} 0 & 0 \\ 0 & 1 \end{pmatrix} \quad \text{for } \mathbf{x} = \begin{pmatrix} 0 \\ 1 \end{pmatrix}$$

or

$$1 \cdot |\mathbf{GHZ}\rangle_0 + x \cdot |\mathbf{GHZ}\rangle_1 = \begin{pmatrix} 1 & 0 \\ 0 & x \end{pmatrix} \quad \text{for } \mathbf{x} = \begin{pmatrix} 1 \\ x \end{pmatrix}.$$

Notice that when $\mathbf{x} = \begin{pmatrix} 0 \\ 1 \end{pmatrix}$, the \mathbf{x} -contraction has rank 1. Similarly, when $x = 0$, or $\mathbf{x} = \begin{pmatrix} 1 \\ 0 \end{pmatrix}$, the \mathbf{x} -contraction also has rank 1. For other \mathbf{x} 's that have $x \neq 0$, \mathbf{x} -contraction yields a matrix representation of rank 2. Thus, $|\mathbf{GHZ}\rangle$ has two contraction points that give us rank 1 matrix representation.

However, it is a different case for $|\mathbf{W}\rangle$. The basis vectors of $|\mathbf{W}\rangle$ along axis 1 are represented by the following matrices:

$$|\mathbf{W}\rangle_0 = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \quad |\mathbf{W}\rangle_1 = \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix}$$

Here, there is only one case of \mathbf{x} that gives us a contraction with a rank 1 matrix representation. In particular, when $\mathbf{x} = \begin{pmatrix} 0 \\ 1 \end{pmatrix}$, \mathbf{x} -contraction becomes $|\mathbf{W}\rangle_1 = \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix}$, which is a singular matrix. For other cases of \mathbf{x} , we obtain an x -contraction of $\begin{pmatrix} x & 1 \\ 1 & 0 \end{pmatrix}$. With every $x \in \mathbb{C}$, that \mathbf{x} -contraction always yields a rank 2 matrix representation.

Consequently, if we want to decide whether $|\phi\rangle$ is in $|\mathbf{W}\rangle$ or $|\mathbf{GHZ}\rangle$, we compute how many contraction points result in rank 1 matrices. From linear algebra, we know that a nonzero 2×2 matrix has rank 1 if it has determinant of 0. Hence, we can compute the determinant of the contractions to compute the rank of the contractions. Firstly, we want to check the determinant of the contraction against $\mathbf{x} = \begin{pmatrix} 0 \\ 1 \end{pmatrix}$. This is straightforward as we already have a formula for computing the determinant. For

general case of $\mathbf{x} = \begin{pmatrix} 1 \\ x \end{pmatrix}$, it is more complicated due to the constant x . Since we do not want to evaluate every contraction, we consider x to be a variable. As a result, we obtain the general contraction

$$\Phi_0 + x\Phi_1 = \begin{pmatrix} p_{00} & p_{01} \\ p_{10} & p_{11} \end{pmatrix} + \begin{pmatrix} xq_{00} & xq_{01} \\ xq_{10} & xq_{11} \end{pmatrix} = \begin{pmatrix} p_{00} + q_{00}x & p_{01} + q_{01}x \\ p_{10} + q_{10}x & p_{11} + q_{11}x \end{pmatrix} = \begin{pmatrix} f_{00}(x) & f_{01}(x) \\ f_{10}(x) & f_{11}(x) \end{pmatrix}$$

Each entry $f(x)$ in this contraction can be regarded as a polynomial with variable x . Since each entry can only be degree 1 or degree 0 polynomial, the determinant $\delta = f_{00}(x)f_{11}(x) - f_{01}(x)f_{10}(x)$ is also a polynomial of x with degree up to 2. Then, we compute the roots of δ . The number of roots of δ , together with the evaluation of $\begin{pmatrix} 0 \\ 1 \end{pmatrix}$ -contraction, will determine how many points of rank 1 matrix contraction. From that, we can tell which non-degenerate class $|\phi\rangle$ is in.

Overall, to classify 3-qubit state $|\phi\rangle$, we need to know whether the state is degenerate. If it is degenerate, it can be in one of the four classes $|\mathbf{R}_1\rangle$, $|\mathbf{R}_2\rangle$, $|\mathbf{R}_3\rangle$ or $|\mathbf{R}_*\rangle$, determined by the existence of radicals along different axes. Otherwise, $|\phi\rangle$ is non-degenerate. Then, $|\phi\rangle$ can be classified as $|\mathbf{W}\rangle$ or $|\mathbf{GHZ}\rangle$, depending on how many rank 1 contractions there are in $|\phi\rangle$.

6.3 4-qubit state inequivalence

Our method can be applied to 4-qubit states and, at least for certain families of states, can distinguish inequivalent states. Since 4-qubit states' contractions are just 3-qubit states, we can apply the results and ideas from 3-qubit classification to 4-qubit context. Let $|\phi\rangle$ be a 4-qubit state in $\mathbb{C}^2 \otimes \mathbb{C}^2 \otimes \mathbb{C}^2 \otimes \mathbb{C}^2$. We will now describe the procedure of constructing the contraction labelling for $|\phi\rangle$. The general idea is that, for each axis, our procedure will try to find "special" projective points for each class, and put them in the contraction label structure discussed in Chapter 6. For example, we will find the projective points that give us $|\mathbf{GHZ}\rangle$ 3-tensor contractions. Then, we will put those projective points in the $|\mathbf{GHZ}\rangle$ label set within our contraction label structure. More interestingly, we will present some sub-procedures to obtain the special points for each type without having to compute every contraction.

6.3.1 Procedure

Similar to the procedure of 3-tensor classification, we first see if $|\phi\rangle$ is a zero tensor. If $|\phi\rangle = \mathbf{0}$, $|\phi\rangle$ is in the zero tensor class, and is clearly inequivalent to other non-zero tensors.

With $|\phi\rangle$ being non-zero, we will construct contraction labelling $Label_{|\phi\rangle}^{(j)}$ along each axis. We will demonstrate our procedure by building the contraction labelling $Label_{|\phi\rangle}$ for axis 1. Because we are only talking about $|\phi\rangle$, we denote $Label_{|\phi\rangle}$ as $Label$ for short. Since there are 7 classes for 3-tensor states, there will be 7 labels that we assign to the projective points.

Since we are contracting 4-tensors against the points \mathbf{x} in $PG_0(\mathbb{C}^2)$, our contraction has the form Φ_1 if $\mathbf{x} = \begin{pmatrix} 0 \\ 1 \end{pmatrix}$ or $\Phi_0 + x\Phi_1$ if $\mathbf{x} = \begin{pmatrix} 1 \\ x \end{pmatrix}$. As we have to treat some arbitrary 4-tensor contractions later in our procedure, it is more convenient to evaluate $v = \begin{pmatrix} 0 \\ 1 \end{pmatrix}$ separately, then generalize arbitrary contractions as $\Phi_0 + x\Phi_1$. We use the procedure from 3-tensor classification to label the point $\begin{pmatrix} 0 \\ 1 \end{pmatrix}$. After labelling $\begin{pmatrix} 0 \\ 1 \end{pmatrix}$, we can assume that the contractions always have the form $\Phi_0 + x\Phi_1$.

The first class we will examine is the **Zero** class. We start with evaluating $|\phi\rangle$'s radicals along all axes to acquire contractions that are in the **Zero** class. This can be easily computed as we can always check if any of the two basis vectors, Φ_0 and Φ_1 , is 0, or if they are linearly dependent. If $\Phi_0 = 0$, then $\begin{pmatrix} 1 \\ 0 \end{pmatrix} \in Label[\mathbf{Zero}]$. If $\Phi_1 = 1$, then $\begin{pmatrix} 0 \\ 1 \end{pmatrix} \in Label[\mathbf{Zero}]$. Finally, if $c_0\Phi_0 + c_1\Phi_1 = 0$ for $c_0, c_1 \neq 0$, then $\begin{pmatrix} 1 \\ \frac{c_1}{c_0} \end{pmatrix} \in Label[\mathbf{Zero}]$.

We will now move on to finding projective points that yield contractions of the non-zero, degenerate classes. Since the methods of finding points for $|R_1\rangle$, $|R_2\rangle$, or $|R_3\rangle$ are the same, we will focus on the process for the $|R_1\rangle$ class. To figure out which contractions would yield $|R_1\rangle$, we would need to contract again along axis 1 of the contractions. Let \mathbf{C} be the contraction $\langle \mathbf{x} | \phi \rangle$ with $\mathbf{x} = \begin{pmatrix} 1 \\ x \end{pmatrix}$. Thus, $\mathbf{C} = \Phi_0 + x\Phi_1$. Here, \mathbf{C} is represented as a 3-way array with entries containing x . Since we are finding x such that \mathbf{x} is in $Label[|R_1\rangle]$, we consider x as a variable. Then, \mathbf{C} can be denoted as $\mathbf{C}(x)$ to indicate that we are treating x as a variable. Notice that, with x as the variable, $\mathbf{C}(x)$ has entries of polynomial of degree at most 1.

From here, we need to take another contraction of $\mathbf{C}(x)$ along axis 1. Thus, $\mathbf{C}_0(x)$ and $\mathbf{C}_1(x)$ are matrices with entries of polynomials with variable x . Now we need to know for which x that $\mathbf{C}_0(x)$ and $\mathbf{C}_1(x)$ are linearly dependent. Compared to the

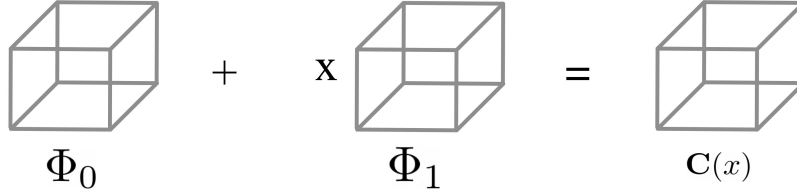


Figure 6.3: Visualization of $\mathbf{C}(x)$ as contraction

process of evaluating $|R_1\rangle$ in 3-tensors classification, this process is a bit trickier as our matrices have the variable x . First of all, we have to evaluate if there exists any x such that $\mathbf{C}_0(x)$ or $\mathbf{C}_1(x)$ is 0. If there is, we add the point $\mathbf{x} = \begin{pmatrix} 1 \\ x \end{pmatrix}$ to $Label[|R_1\rangle]$.

Now, we can assume that $\mathbf{C}_0(x)$ and $\mathbf{C}_1(x) \neq 0$. In particular, there are two cases to consider. The first case is that, as polynomial vectors, $\mathbf{C}_0(x)$ and $\mathbf{C}_1(x)$ are linearly dependent. For example, $\begin{pmatrix} x & 2 \\ 1 & 3x \end{pmatrix}$ and $\begin{pmatrix} 2x & 4 \\ 2 & 6x \end{pmatrix}$ are linear dependent as polynomial vectors. Then, $\mathbf{C}(x)$ will have a radical along axis 1 regardless of the value of x , and every points \mathbf{x} will be labeled $|R_1\rangle$. The other case is when $\mathbf{C}_0(x)$ and $\mathbf{C}_1(x)$ are not linear independent as vectors. Yet, we can still find an x such that $\mathbf{C}_0(x)$ and $\mathbf{C}_1(x)$ are scalar multiple of each other. For example, $\begin{pmatrix} 0 & 3x \\ x+2 & 5 \end{pmatrix}$ and $\begin{pmatrix} x-2 & 6 \\ 2x & 2x+1 \end{pmatrix}$ are not scalar multiples of each other. However, if we evaluate those matrices at $x = 2$, the two matrices become the same, and clearly, linear dependent. Again, if there exists such an x , we will add $\mathbf{x} = \begin{pmatrix} 1 \\ x \end{pmatrix}$ into our $Label[|R_1\rangle]$.

Following the same procedure for other axes, we obtain some values for $Label[|R_2\rangle]$ and $Label[|R_3\rangle]$. If there is any point in all three radical class labels, it belongs to $Label[|R_*\rangle]$ instead. At this stage, we have completed finding the projective points that result in degenerate 3-tensors.

We move on to identify the points for non-degenerate classes, $|GHZ\rangle$ and $|W\rangle$. As we described in the previous section, $|GHZ\rangle$ and $|W\rangle$ differ in their number of rank 1 contractions. Just like how we rely on determinants to decide whether a matrix has rank 1, we will do the same here. Again, there needs to be some extra steps as our contractions now contain “variable”. We use the same notation $\mathbf{C}(x)$ for our general contractions, and $\mathbf{C}_0(x)$ and $\mathbf{C}_1(x)$ for our basis vector contraction along axis 1.

We first evaluate the rank of $\mathbf{C}_1(x)$, as $\mathbf{C}_1(x)$ is the $\begin{pmatrix} 0 \\ 1 \end{pmatrix}$ contraction of $\mathbf{C}(x)$. $\mathbf{C}_1(x)$ has the following form

$$\mathbf{C}_1(x) = \begin{pmatrix} p_{00}(x) & p_{01}(x) \\ p_{10}(x) & p_{11}(x) \end{pmatrix}$$

with $p_{ij}(x)$ being polynomials of degree at most 1 with variable x . Thus, $\delta = \text{Det}(\mathbf{C}_1(x)) = p_{00}(x)p_{11}(x) - p_{01}(x)p_{10}(x)$. As $p_{ij}(x)$ has degree at most 1, δ has degree at most 2. Hence, to calculate which x will yield a determinant of 0, we can compute the root of the determinant. Based on the degree of δ , we have the following cases:

1. *Degree*(δ) = -1: This means $\delta = 0$. If $\delta = 0$ for every x , then every point \mathbf{x} has a contraction of rank 1 at $\begin{pmatrix} 0 \\ 1 \end{pmatrix}$.
2. *Degree*(δ) = 0: This means $\delta \neq 0$ and δ is a constant polynomial. Then, there is no \mathbf{x} that has a contraction of rank 1 at $\begin{pmatrix} 0 \\ 1 \end{pmatrix}$.
3. *Degree*(δ) ≥ 1 : This means δ is either a linear polynomial or a quadratic polynomial. We can easily compute the roots for δ , and for every root, we have a corresponding point \mathbf{x} that has a contraction of rank 1 at $\begin{pmatrix} 0 \\ 1 \end{pmatrix}$.

Thus, we have found all the points \mathbf{x} having $\begin{pmatrix} 0 \\ 1 \end{pmatrix}$ -contraction of rank 1 (*).

Then, we continue to evaluate other general contractions of $\mathbf{C}(x)$. We denote the contraction of $\mathbf{C}(x)$ as \mathcal{C} , and $\mathcal{C} = \mathbf{C}_0(x) + y \cdot \mathbf{C}_1(x)$ with $y \in \mathbb{C}$. If we consider y as a variable as well, then \mathcal{C} can be considered as the matrix with two variables x, y . We denote \mathcal{C} as $\mathcal{C}(x, y)$, and

$$\mathcal{C}(x, y) = \mathbf{C}_0(x) + y\mathbf{C}_1(x) = \begin{pmatrix} f_{00}(x) + yp_{00}(x) & f_{01}(x) + yp_{01}(x) \\ f_{10}(x) + yp_{10}(x) & f_{11}(x) + yp_{11}(x) \end{pmatrix}$$

with f_{ij}, p_{ij} being polynomials with variable x of degree at most 1. Then,

$$\delta = \text{Det}(\mathcal{C}) = [f_{00}(x) + yp_{00}(x)][f_{11}(x) + yp_{11}(x)] - [f_{01}(x) + yp_{01}(x)][f_{10}(x) + yp_{10}(x)].$$

Thus, we can consider δ as a polynomial in y of degree at most 2. Our goal is to see how many more rank 1 contractions a point \mathbf{x} can have, then together with (*), compute the total number of rank 1 contraction at each point. Depending on different possible degrees of δ , we use a different method to determine the number of rank 1 contractions at each point. Thus, we obtain the following cases:

1. *Degree*(δ) = -1: This case is not possible as this means every contraction $\mathcal{C}(x)$ of $\mathbf{C}(x, y)$ is singular. Since $\mathbf{C}(x)$ is a non-degenerate 3-tensor, by Lemma 4.2.1, $\mathbf{C}(x)$ has at least one invertible contraction. This is a contradiction.

2. $Degree(\delta) = 0$: This means that there is no x that has other rank 1 contractions. Hence, every point in (*) belongs to $Label[|W\rangle]$, and other non-radical point belongs to $Label[|GHZ\rangle]$.
3. $Degree(\delta) = 1$: We can consider $\delta = b(x) + a(x) \cdot y$. If $a(x) \neq 0$, then $\delta = 0$ will always have a solution $y = \frac{-b(x)}{a(x)}$. Thus, we find x such that $a(x) = 0$ to find out which point does not have rank 1 contraction when contracted against $\begin{pmatrix} 1 \\ y \end{pmatrix}$. Combined with (*), we will be able to compute which \mathbf{x} gives us only 1 rank 1 contraction.
4. $Degree(\delta) = 2$: Here, $\delta = c(x) + b(x) \cdot y + c(x) \cdot y^2$. We want to find x such that $\delta = 0$ only at one point y . This means δ has only one repeated root. Thus, we can evaluate when the discriminant $\Delta(\delta) = 0$. If $\Delta(\delta) = 0$ for every x , every x has a contraction of rank 1 in the general contraction case. If not, we compute the roots for $\Delta(\delta)$. These roots will correspond to points \mathbf{x} that have a contraction of rank 1. Together with (*), we will be able to tell which points have only 1 rank 1 contraction.

After these sub-procedures compute degenerate and non-degenerate contraction points, we obtain a complete axis label for $|\phi\rangle$ along axis 1. If we do the same for other axes, we will get labels of every contraction point along every axis. Consequently, when we compare two tensors $|\phi\rangle$ and $|\psi\rangle$, we only need to compare $Label_{|\phi\rangle}^{(j)}$ and $Label_{|\psi\rangle}^{(j)}$ for every axis j .

6.3.2 Application to nilpotent classes

We applied our procedure to test non-isomorphism among the nilpotent classes in [1]. The nilpotent classes are listed in Table 6.3.2. Our procedure managed to

Orbit	Representative	\mathcal{S} -conjugate to
1	$ 1100\rangle$	N_2
2	$ 1100\rangle + 0000\rangle$	N_3
3	$ 1100\rangle + 1001\rangle$	N_3
4	$ 1100\rangle + 1010\rangle$	N_3
5	$ 1101\rangle + 0100\rangle$	N_3
6	$ 1110\rangle + 0100\rangle$	N_3
7	$ 1110\rangle + 1101\rangle$	N_3
8	$ 1101\rangle + 0100\rangle + 1000\rangle$	N_6
9	$ 1110\rangle + 0100\rangle + 1000\rangle$	N_6
10	$ 1110\rangle + 1101\rangle + 1000\rangle$	N_6
11	$ 1110\rangle + 1101\rangle + 0100\rangle$	N_6
12	$ 0101\rangle + 1100\rangle + 1001\rangle + 0000\rangle$	N_9
13	$ 0110\rangle + 1100\rangle + 1010\rangle + 0000\rangle$	N_9
14	$ 1111\rangle + 1100\rangle + 1001\rangle + 1010\rangle$	N_9
15	$ 0111\rangle + 1110\rangle + 1101\rangle + 0100\rangle$	N_9
16	$ 1110\rangle + 1101\rangle + 0100\rangle + 1000\rangle$	N_4
17	$ 1110\rangle + 1101\rangle + 0000\rangle$	N_5
18	$ 1110\rangle + 0100\rangle + 1001\rangle$	N_5
19	$ 1101\rangle + 0100\rangle + 1010\rangle$	N_5
20	$ 0101\rangle + 1110\rangle + 1000\rangle$	N_5
21	$ 0110\rangle + 1101\rangle + 1000\rangle$	N_5
22	$ 1111\rangle + 0100\rangle + 1000\rangle$	N_5
23	$ 1110\rangle + 0100\rangle + 0000\rangle + 1001\rangle$	N_8
24	$ 0110\rangle + 1101\rangle + 1000\rangle + 0000\rangle$	N_8
25	$ 1111\rangle + 0100\rangle + 1000\rangle + 1001\rangle$	N_8
26	$ 1111\rangle + 0100\rangle + 1000\rangle + 1010\rangle$	N_8
27	$ 0101\rangle + 1110\rangle + 0000\rangle + 1001\rangle$	N_7
28	$ 0110\rangle + 1101\rangle + 0000\rangle + 1010\rangle$	N_7
29	$ 1111\rangle + 0100\rangle + 1001\rangle + 1010\rangle$	N_7
30	$ 1111\rangle + 0110\rangle + 0101\rangle + 1000\rangle$	N_7
31	0	

Table 6.2: Nilpotent classes for 4-qubit states from [1]

distinguish all the nilpotent classes using their contraction label by axis. We can look at orbit 17 and orbit 18 for example. Table 6.3.2 contains our procedure outputs for orbit 17 and orbit 18. It is obvious that orbit 17 and orbit 18 are in different classes. Specifically, apart from axis 3, all other axes have mismatch in their label sets. For example, while orbit 17 has only a $|R_*$ type contraction and all the other points are $|GHZ\rangle$, orbit 18 has one $|R_*$, one $|R_2\rangle$, and other points are W . However, even though they are supposed to be in different equivalence class, orbit 17 and orbit 18 are in the same equivalence class if we allow axis permutation. If we ignore the mismatch between each axis label, we notice that the structure of orbit 17 and orbit 18 line up: any axis that has an 1-radical and a 3-radical contraction also has other points labeled as $|W\rangle$. On the other hand, any axis that has only a 3-radical has other points labeled as $|GHZ\rangle$. This similarity in structure might be due to the axis permutation that maps orbit 17 to orbit 18. Upon further manual investigation, we notice that our labels have potential to discern (non)-isomorphism up to permutation for other equivalence up to permutation classes as well.

	Label	Orbit 17	Orbit 18
	Zero		
	$ R_1\rangle$	(0, 1)	
Axis 1:	$ R_2\rangle$		
	$ R_3\rangle$		
	$ R_*$	(1, 0)	(1, 0)
	$ W\rangle$	other points	
	$ GHZ\rangle$		other points
	Label	Orbit 17	Orbit 18
	Zero		
	$ R_1\rangle$	(0, 1)	
Axis 2:	$ R_2\rangle$		
	$ R_3\rangle$		(0, 1)
	$ R_*$	(1, 0)	(1, 0)
	$ W\rangle$	other points	other points
	$ GHZ\rangle$		
	Label	Orbit 17	Orbit 18
	Zero		
	$ R_1\rangle$		
Axis 3:	$ R_2\rangle$		
	$ R_3\rangle$		
	$ R_*$	(0, 1)	(0, 1)
	$ W\rangle$		
	$ GHZ\rangle$	other points	other points
	Label	Orbit 17	Orbit 18
	Zero		
	$ R_1\rangle$		
Axis 4:	$ R_2\rangle$		(1, 0)
	$ R_3\rangle$		
	$ R_*$	(0, 1)	(0, 1)
	$ W\rangle$		other points
	$ GHZ\rangle$	other points	

Table 6.3: Axis labels of orbit 17 and orbit 18

* In this table, we denote the points with row vector for better presentation. The blank space entries indicate empty sets; there are no points with those labels in that axis.

Chapter 7

Conclusion

In this thesis, we introduced the reader to tensors, drawing upon similar concepts from linear algebra. We defined tensor concepts like tensor isomorphism, actions on tensors, and tensor contractions. Using tensor contractions, we derived the known classifications of 3-tensors, showing that there are 8 different equivalence classes for a 3-tensor over an arbitrary field.

More importantly, we developed a new contraction-based approach for the tensor isomorphism problem. From 3-tensor isomorphism classification, we generalized to n -tensor isomorphism. We construct a new method to compare n -tensors using tensor contraction, projective geometry, and contraction labels. Projective geometry helps decide what contraction points we should include in our contraction labelling process. Then, we described how to build contraction label for each axis of a tensor.

To confirm the validity of our method, we proved that if two tensors $|\phi\rangle$ and $|\psi\rangle$ are isomorphic, and if a v -contraction of $|\phi\rangle$ is in some classification χ , then there exists u -contraction of $|\psi\rangle$ whose classification is also χ . Thus, to distinguish two tensors, our method suggested that we can build contraction labels along every axis of those tensors, and compare the axis contraction labels. If there is a mismatch along any axis, the two tensors are not isomorphic.

Lastly, as a proof of concept, we applied our method to QIT. In particular, we re-confirmed 3-tensor classifications in [2]. Then, we devised a general procedure to tell 4-qubit state inequivalence. We tested our procedure on the nilpotent classes in [1], and we were able to discern all the nilpotent classes. Moreover, we also spotted

some contraction structure similarity up to permutation class.

In the future, we wish to apply this method to more scenarios. Firstly, we can expand our procedure to accommodate isomorphism up to axis permutation. Secondly, we can apply this to semi-simple and mixed states in [1] to have a more concrete result. Last but not least, if our method is good enough for the 4-qubit case, it can be used to quickly compute 5-qubit inequivalence by random sampling the contraction labels, and compare the statistical results of any two tensors.

Appendix A

Procedure for 4-tensor labelling

Here are the main functions from the Magma implementation. Available on GitHub.

```
1: function LABEL(t)
2:   if t eq 0 then
3:     return "Zero"
4:   else
5:     l := AssociativeArray();           ▷ container for Label(j)(t), j ∈ [1..4]
6:     for axis in [1..4] do
7:       rads := FindAxisRadical(t, axis)
8:       if rads is not empty then
9:         l[axis]["Zero"] := rads
10:      end if
11:      assign point (0,1) to the right label
12:      c := Compute Contraction(t)
13:      l[axis]["R1"] := FindAxisRadical(c, 1)
14:      l[axis]["R2"] := FindAxisRadical(c, 2)
15:      l[axis]["R3"] := FindAxisRadical(c, 3)
16:      l[axis]["R*"] := R1 ∩ R2 ∩ R3
17:      remove elements in R* from R1, R2 and R3
18:      DEG := set of all degenerate points
19:      l[axis]["W"], l[axis][GHZ] := FindNonDegeneratePoints(c, DEG)
20:    end for
21:  end if
22:  return l
```

```

23: end function
24:
25: function FindAxisRadical(t, axis)
26:   v1, v2 := SliceTensor(t, axis)
27:   if v1 and v2 are linearly dependent then
28:     return "all"
29:   else
30:     if exists an x such that v1 and v2 are linearly dependent then
31:       return {(0, x)}
32:     else
33:       return {}
34:     end if
35:   return {}
36:   end if
37: end function
38:
39: function FindNonDegeneratePoints(t, DEG)
40:   v1, v2 := SliceTensor(t, 1)      ▷ the slice axis does not matter here
41:   m1, m2 := Matrix(v1), Matrix(v2)
42:   d := Det(m2)                      ▷ d is a polynomial with variable x
43:   all_point := false                  ▷ keeping track of rank 1 point
44:   if Degree(d) eq -1 then             ▷ d is 0
45:     all_inf := true
46:   else if Degree(d) eq 0 then
47:     C := {}
48:   else
49:     C := {Roots(d) : d ∉ DEG}
50:   end if
51:   p := m1 + y · m2                  ▷ y ∈ PolynomialRing
52:   d := Det(p)
53:   if Degree(d) eq 0 then              ▷ d = a(x) + b(x)y
54:     roots := {Roots(a(x)) : x ∉ DEG}
55:     if all_inf then
56:       return roots, "all"
57:     else
58:       return "all", roots △ C
59:     end if
60:   else                                  ▷ d is quadratic
61:     dis := Discriminant(d)

```

```

62:     if dis eq 0 then
63:         if all_inf then
64:             return {}, "all"
65:         else
66:             return "all", C
67:         end if
68:     else
69:         roots := {Roots(dis) : r ∉ DEG}
70:         if all_inf then
71:             return "all", roots
72:         else
73:             return roots, "all"
74:         end if
75:     end if
76: end if
77: end function

```


References

- [1] Heiko Dietrich, Willem A. de Graaf, Alessio Marrani, and Marcos Origlia. Classification of four qubit states and their stabilisers under SLOCC operations. Journal of Physics A: Mathematical and Theoretical, 55(9):095302, feb 2022.
- [2] W. Dür, G. Vidal, and J. I. Cirac. Three qubits can be entangled in two inequivalent ways. Physical Review A, 62(6), Nov 2000.
- [3] Joshua A. Grochow and Youming Qiao. Isomorphism problems for tensors, groups, and cubic forms: completeness and reductions. 2019.
- [4] D. Li, X. Li, H. Huang, and X. Li. Slocc classification for nine families of four-qubits, 2009.
- [5] Dafa Li, Xiangrong Li, Hongtao Huang, and Xinxin Li. Simple criteria for the slocc classification. Physics Letters A, 359(5):428–437, Dec 2006.
- [6] James Wilson Peter Brooksbank, Joshua Maglione. Tools to Tame the Tensor. In progress.
- [7] Tinggui Zhang, Ming-Jing Zhao, and Xiaofen Huang. Criterion for slocc equivalence of multipartite quantum states. Journal of Physics A: Mathematical and Theoretical, 49(40):405301, Sep 2016.