UNIVERSITY OF AUCKLAND

DOCTORAL THESIS

# Equiangular lines, projective symmetries and nice error frames

*Author:*
Tuan-Yow CHIEN

*A thesis submitted in fulfilment of the requirements*
*for the degree of Doctor of Philosophy*

*in the*

Department of Mathematics

*"This thesis is for examination purposes only and is confidential to the examination process."*

January 2015

*“Rabbit’s clever,” said Pooh thoughtfully.*
*“Yes,” said Piglet, “Rabbit’s clever.”*
*“And he has Brain.”*
*“Yes,” said Piglet, “Rabbit has Brain.”*
*There was a long silence.*
*“I suppose,” said Pooh, “that that’s why he never understands anything.”*

“Winnie-the-Pooh” by A. A. Milne.

# *Abstract*

The existence of maximal sets of equiangular lines (SIC-POVMs) is of interest to the mathematics and physics communities due to their connection to quantum information theory, quantum cryptography, spherical 2-designs and cubature rules. This thesis looks at a way to recover analytic SIC-POVMs from numerical SIC-POVMs, including the discovery of a new analytic SIC-POVM in 17 dimensions. It also develops a way to comprehensively find nice error bases (used for finding SIC-POVMs). Finally, a general theory to classify sequences of vectors up to projective unitary equivalence is developed and applied to study the projective symmetry groups of sequences of vectors.

# *Acknowledgements*

I would like to thank Shayne Waldron, my primary supervisor, for his guidance, advice, collaborations, support and his many invested hours bouncing and developing ideas together; Tom ter Elst for his mathematical insight, kind advice and support; Marcus Appleby for sharing his wealth of knowledge and experience regarding the SIC problem with me, as well as helping to proof read the thesis; Huangjun Zhu for helpful discussions regarding symmetry groups; Ingemar Bengtsson for discussions on MUBs, complex multiplication and hosting my stay in Stockholm; Steve Flammia for hosting my stay in Sydney; Hwan Goh for proof reading the whole thesis; Kieran Roberts for helping to proof read the thesis; my family and friends for supporting me through the ups and downs; and the administrative staff for making the red tape more bearable.

I am grateful for the financial support received. Part of this work is supported by the University of Auckland Doctoral Scholarship and the Marsden Fund Council from Government funding administered by the Royal Society of New Zealand.

# Contents

# List of Figures

# List of Tables

# Abbreviations

| | |
|---|---|
| **SIC-POVM** | **S**ymmetric **I**nformationally **C**omplete **P**ositive **O**perator **V**alued **M**easure. See page 12. |
| **MUB** | **Mutually Unbiased Bases** See page 55. |
| **NEB** | **Nice Error Basis**. See page 87. |

# Symbols

| | |
|---|---|
| $\langle p, q \rangle_s$ | Symplectic form $p_2 q_1 - p_1 q_2$ as defined in (2.3.2). See page 14. |
| $D_p$ | A Weyl-Heisenberg displacement operator. See page 14. |
| H($d$) | Weyl-Heisenberg group in $d$ dimensions. See page 14. |
| C($d$) | Clifford group in $d$ dimensions. See page 16. |
| ECL($d$) | Extended Clifford group. See page 16. |
| $\mathbb{T}$ | One dimensional torus, i.e., all complex numbers of modulus 1. See page 20. |
| $\tau$ | $-e^{i\pi/d}$. See page 15. |
| $\Pi$ | A SIC-POVM projector. See page 27. |
| $\chi_p$ | A SIC-POVM overlap. See page 27. |
| $\mathrm{Tr}(A)$ | The trace of the matrix $A$. See page 27. |
| $a$ | $\sqrt{\frac{(d-3)(d+1)}{k}}$. See page 28. |
| $\mathbb{E}$ | Smallest normal extension of $\mathbb{Q}$ containing the components of $\Pi$ and $\tau$. See page 28. |
| $\bar{\mathbb{E}}$ | $\mathbb{E}(\sqrt{d})$. See page 28. |
| $\bar{\mathcal{G}}$ | Galois group of $\bar{\mathbb{E}}$ over $\mathbb{Q}$. See page 29. |
| $S_\Pi$ | Stabiliser of $\Pi$. See page 28. |
| $\tilde{S}_\Pi$ | $\{F \in \mathrm{ESL}(2, \mathbb{Z}_{\bar{d}}) : V_F \in S_\Pi\}$. See page 30. |
| $\bar{S}_\Pi$ | $\{(\mathrm{Det}\, F)F : F \in \tilde{S}_\Pi\}$. See page 30. |
| $\bar{C}_\Pi$ | Centraliser of $\bar{S}_\Pi$. See page 31. |
| $\rho$ | In Chapter 3, they are polynomials with SIC-POVM overlaps as roots (see page 32). |
| $\rho$ | In Chapters 5 and 6 they are unitary representations (see pages 76 and 87). |
| $\zeta$ | Polynomials with the coefficients of $\rho$ as roots. See page 39. |
| $T_{ijk}$ | Triple product (3-product). See page 42. |
| $\Delta$ | $m$-product. See page 42. |
| Sym($\Phi$) | Symmetry group of $\Phi$. See page 64. |
| Sym$_P$($\Phi$) | Projective symmetry group of $\Phi$. See page 65. |

# Chapter 1

# Introduction

The overarching motivation behind this thesis is to study the problem of the existence of maximal sets of equiangular lines in $\mathbb{C}^d$. Consider a set of orthogonal basis vectors. The angle of intersection between any two vectors in this set is 90 degrees. This set of vectors is equiangular as the angle of intersection between any two vectors is constant. A natural question to ask is, how many vectors can one have in a given dimension before it is no longer possible to have this equiangular property. For example, the 3 spokes of the Mercedes-Benz logo is an example of a maximal set of 3 equiangular lines in $\mathbb{R}^2$. This question of existence (for maximal sets of complex equiangular lines) has been open for more than 10 years and has been of great academic interest to the physics community.

A set of equiangular lines gives rise to a corresponding basis on the vector space of $d \times d$ complex matrices ($M_d$) with the property that the inner product between any two basis elements is constant (and non zero). Moreover, it implies that in $\mathbb{C}^d$ there are at most $d^2$ equiangular lines.

This is a desirable symmetry in quantum information theory (see introduction of [RBKSC04]) and has applications in the construction of optimal quantum error correcting codes (see Section 5.4.2 of [Ren04]). They also exhibit rich structure and connections with several areas of mathematics, including frame theory, number theory and algebra. It makes the problem fun and interesting to study in its own right.

Proof on the existence of maximal sets of equiangular lines (in $\mathbb{C}^d$) is only known in dimensions 1-16, 19, 24, 35 and 48. Historically, the approach to finding maximal sets of equiangular lines has involved solving systems of polynomial equations by hand ([RBKSC04]), using Grobner bases in a brute force computational search ([SG10]), or with specific geometric constructions

([Hog98], [Hug07]). Amongst other things, this thesis contributes dimension 17 to this list. This is done using a novel Ansatz, which turns numerical approximations of maximal sets of equiangular lines into analytic expressions by studying certain algebraic extensions of $\mathbb{Q}$ in which the numerical values are conjectured to live.

This thesis looks at four aspects of the existence problem (Chapters 3 to 6). Firstly, using numerical approximations for sets of equiangular lines, how can one recover analytic expressions for these equiangular lines? Secondly, how does one classify sequences of vectors up to projective equivalence? Thirdly, how does one generate the group of symmetries of projective objects? Fourthly, almost all known constructions of maximal sets of equiangular lines arise as orbits of the Weyl-Heisenberg group. How could one generate equiangular lines with groups other than the Weyl-Heisenberg group? These questions are elaborated under their relevant chapter headings below.

## Chapter 2, Background.

This chapter lays down common foundational definitions and theorems (without proof) for the rest of the chapters. Section 2.1 is about the theory of frames. Frames are given in Definition 2.1.1 and equiangular lines are formally defined in Definition 2.1.12 with their link to frame theory. There is then a survey of the key tools used to numerically construct equiangular lines as orbits of group actions.

The next two sections give an overview of the problem of the existence of maximal equiangular lines. Maximal equiangular lines (SIC-POVMs or Symmetric Informationally Complete Positive Operator Valued Measures) are formally defined in Definition 2.2.2, with a formulation of the existence problem in Conjecture 2.2.3 (Zauner's conjecture), which conjectures that for any $d \in \mathbb{N}$, there exists a set of $d^2$ equiangular lines in $\mathbb{C}^d$. Basic facts about the important Weyl-Heisenberg group are presented as well as a look into the the normaliser of the Weyl-Heisenberg group (Clifford group), along with the extended Clifford group.

Lastly, the curious black sheep of the existence problem (dimension 3 SIC-POVMs) is presented in Section 2.5 with its continuous properties and its connection to elliptic curves.

## Chapter 3, Reverse engineering numerical SIC-POVMs.

This chapter presents new research conducted jointly with Marcus Appleby and Shayne Waldron which obtains new analytic SIC-POVMs from numerical SIC-POVMs by studying the algebraic extension fields of $\mathbb{Q}$ in which known SIC-POVMs reside. SIC-POVM existence is proved in dimension 17 (previously unknown) by reverse engineering the numerical solutions and making use of conjectures about the SIC-POVM fields.

The approach here is novel and takes a numerical SIC-POVM vector, attempts to construct the exact values of the inner products of the SIC-POVM in order to reconstruct the original vector analytically. This Ansatz relies on conjectures relating to the field structure of the SIC-POVM and ad hoc calculations, in order to speculate a structure to the analytic solution. It essentially reduces the problem down to a factorisation problem (of polynomials). The speculated solution is then verified afterwards to be a SIC-POVM. While there is no formal proof that any of this should work, "the proof is in the pudding (SIC-POVM)", so to speak, since a SIC-POVM can in fact be found using this process.

Section 3.1 establishes some fundamental Galois theory regarding the solubility of polynomials by radicals. The key hypothesis on the structure of the SIC-POVM fields and the Galois groups of those fields over $\mathbb{Q}$ is stated in Conjecture 3.2.17. Theorem 3.2.18 and Conjectures 3.2.20 and 3.2.21 then provide practical assistance in the reverse engineering efforts. Section 3.3 outlines the procedure developed for reverse engineering the new analytic SIC-POVMs, including an algorithm for precision bumping numerical SIC-POVMs (algorithm 3.1). Potential improvements to the method are discussed in Section 3.4 and the new SIC-POVM fiducial vectors for dimension 17 is given in Appendix C.

## Chapter 4, Projective unitary equivalences of frames.

When presented with two sequences of vectors, one could ask whether the two sequences are fundamentally the same. This is often captured by the notion of unitary equivalence, since unitary transformations preserve key geometric properties like the angle between vectors and their lengths. For projective objects like lines, an analogous concept of projective unitary equivalence arises. The additional complication arises where each vector is allowed to be scaled by any arbitrary complex number of unit length. This complication makes the problem of classifying

lines difficult, a feat some researchers thought to be impossible in a general setting (e.g., see comments in Section 5, paragraph 2 of [VW10] regarding type III equivalences). Studying the projective equivalence between sequences of vectors is important as it could be used to classify frames and count distinct solutions in the SIC-POVM existence problem. It also makes it possible to compute the projective symmetry group (see Chapter 5), which is an important object of study (e.g., can be used to determine whether a sequence of vectors is a group frame).

The seminal paper of [AFF11] (Theorem 3 and the remark on p. 13) first utilised (3–vertex) Bargmann invariants (so called "triple products") to classify SIC-POVMs up to projective unitary equivalence. This chapter takes this idea of a Bargmann invariant and recasts it in language of frame graphs (Definition 4.0.5)), paving the way for a general classification method.

The main theorems are presented in Sections 4.2 and 4.3. Whilst [AFF11] only gives a classification for SIC-POVMs, Theorem 4.2.2 greatly generalises their result by characterising *any* two sequences of vectors up to projective unitary equivalence via $m$–products. Theorem 4.3.6 gives an indication of which $m$–products are sufficient. An algorithm to solve the inverse problem, i.e., given a sufficient set of $m$–products, construct all sequences which give rise to these products, is provided by the proof of Theorem 4.2.2. The implications of this to the classification of MUBs (Mutually Unbiased Bases) is later considered.

Section 4.4 extends the classification of sequences of vectors up to (projective) similarity and Section 4.5 applies the theory to the classification of harmonic frames up to projective unitary equivalence. In particular, projective unitary equivalence of harmonic frames is characterised up to affine equivalence in Theorem 4.5.6. An explicit description of dimension 2 harmonic frames up to projective unitary equivalence is then given by Proposition 4.5.7. Finally, tables of the harmonic frame calculations are presented.

The research in this chapter was conducted jointly with Shayne Waldron and adapted from [CW14a].

## Chapter 5, Projective symmetry groups.

This chapter studies the projective symmetry group of a sequence of vectors. This group can really only be computed once it is possible to compute projective unitary equivalence (e.g., by the characterisation theorems of Chapter 4). These projective symmetry groups have only been

computed previously for the SIC-POVM case in Chapter 10 of [Zhu12] and has hence received little study. The results and algorithm in this chapter enable projective symmetry groups to be computed for *any* finite sequence of vectors. These groups are important as they describe all the symmetries present in a sequence of vectors. This can lead to the simplification of systems, discovering new symmetries and even determine whether the sequence arises as the orbit of a group.

Theorem 5.2.5 establishes that a frame and is complement have the same projective symmetry group. Furthermore, Theorem 5.3.1 gives a set of projective invariants which determine a sequence of vectors up to projective similarity when the underlying field is closed under complex conjugation.

Section 5.4 gives an algorithm for calculating the projective symmetry group from $m$–products (projective invariants). The only other known algorithm (Section 10.2.3 of [Zhu12]) for computing projective symmetry groups is for the special case of sequences characterised by 3–products. That algorithm was applied specifically to the situation of $d^2$ equiangular vectors in $\mathbb{C}^d$ which are given as a group orbit.

Section 5.5 briefly extends the analysis to "anti-linear symmetries" and the extended projective symmetry group. Section 5.6 considers simplifications to the algorithm in Section 5.4 for group frames and computes the extended projective symmetry group of certain SIC-POVMs and MUBs. Section 5.7 presents some results from extensive calculations of the projective symmetry group and extended projective symmetry group of harmonic frames. The full table of harmonic frame data is given in Appendix B.

The research in this chapter was conducted jointly with Shayne Waldron and adapted from [CW14b].

## Chapter 6, Nice error frames.

Most SIC-POVMs arise as the orbit of the Weyl-Heisenberg group. Other groups are also known to give rise to SIC-POVMs. All of these groups are in a class of groups called nice error groups and their associated nice error bases are sufficient to generate SIC-POVMs. A complete catalogue of nice error groups was previously difficult to find and the catalogue of nice error bases in the web link in `http://www.cs.tamu.edu/faculty/klappi/ueb/ueb.html` had

issues of over and under counting. This chapter rectifies these problems by introducing the notion of a canonical nice error group, allowing for a systematic way to exhaustively find all inequivalent nice error groups. This is applied to hunt for non Weyl-Heisenberg SIC-POVMs in dimensions up to 16. Of the SIC-POVMs found, the analytic SIC-POVMs for index group `<36,11>` and `<64,78>` were independently discovered by Grassl in Section 4.2 of [Gra05]. The numerical SIC-POVMs in dimensions 6 and 8 were independently discovered by Zhu in Table 10.2 of [Zhu12].

While the excitement level of discovering these analytic SIC-POVMs (see Appendix A) diminished after being alerted of these independent discoveries, these analytic solutions still have value. The comprehensive search conducted here was over all nice error groups in dimensions up to (but not including dimension 16). This is an improvement on the catalogue computed in `http://faculty.cs.tamu.edu/klappi/ueb/ueb.html`, which is an incomplete catalogue of nice error bases in dimension 8. These analytic results also act as verification of the numerical work done in Table 10.2 of [Zhu12].

The basic theory is presented in Sections 6.1. The nice error frame is introduced by Definition 6.2.1 and its canonical representation by Definition 6.2.3. Nice error frames are equivalent to unitary faithful projective representations (Proposition 6.2.8) and equivalent error frames are then shown have the same canonical error group (Proposition 6.2.6). A tensor construction of nice error groups is also given. Theorem 6.2.14 characterises nice error frames with abelian index groups. Section 6.3 then focuses on calculating nice error groups and a way of generating equivalent nice error frames. Subection 6.3.1 discusses the connection to SIC-POVMs briefly and a detailed calculation of a SIC-POVM in 6 dimensions is given in Section 6.4.

Section 6.5 concludes the chapter with some short remarks on the equivalence of known SIC-POVMs with Weyl-Heisenberg SIC-POVMs. The exceptional case of the Hoggar SIC-POVM is discussed. By using the theory of projective symmetry groups developed in this thesis, the Hoggar lines were then found as a subgroup of the Clifford group, leading to a conjecture that all group covariant SIC-POVMs are covariant to a subgroup of the normaliser of the Weyl-Heisenberg group (Conjecture 6.5.1).

Tables of nice error groups, bases, and an extensive catalogue of SIC-POVMs for 8 dimensional nice error bases are given in Appendix A.

The research in this chapter was conducted jointly with Shayne Waldron and adapted from an unpublished work in progress.

# Chapter 2

# Background

## 2.1 Frame theory

Unless otherwise stated, all vector spaces $X$ in this thesis are finite dimensional and over a subfield of $\mathbb{C}$. The material in this section is found in standard textbooks on frame theory, cf. Chapters 1-5 of [Chr03], Chapters 1-3, 5 of [CKE13], Chapters 1-3, 8, 10, 12 of [Wal15]. The Hilbert spaces $\mathcal{H}$ here have Euclidean inner product.

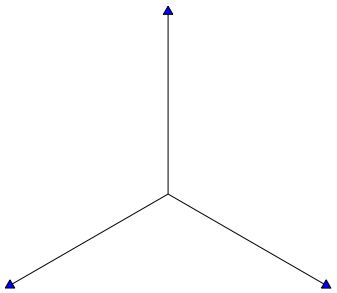
**Definition 2.1.1.**
Let $\Phi := (v_j)_{j\in J}$ be a sequence of vectors in a real or complex Hilbert space $\mathcal{H}$. $\Phi$ is a **frame** for $\mathcal{H}$ with **frame bounds** $A, B > 0$ if

$$A\|f\|^2 \leq \sum_{j\in J} |\langle f, v_j\rangle|^2 \leq B\|f\|^2, \qquad \forall f \in \mathcal{H}. \tag{2.1.1}$$

If $A = B$, then $\Phi$ is a **tight frame**. If $A = B = 1$, then it is a **normalised tight frame**. If $J$ is finite, then $\Phi$ is a **finite frame**.

This definition of a frame is well defined for infinite dimensional Hilbert spaces. In finite dimensional spaces, any finite spanning sequence of vectors satisfies (2.1.1) and is a frame. Tight frames are generalisations of orthonormal bases. Examples of tight frames containing more vectors than a basis include SIC-POVMs, MUBs, and harmonic frames.

**Example 2.1.2** (Mercedes Benz frame)**.**
A classic example of a tight frame is the Mercedes Benz frame given by the three equally spaced vectors in $\mathbb{R}^2$:



From another point of view, they are the three vectors $\{i, e^{7\pi i/6}, e^{-\pi i/6}\}$ in $\mathbb{C}$.

**Theorem 2.1.3.**

Let $\Phi := (v_j)_{j\in J}$ be a tight frame. Then (2.1.1) is equivalent (by the polarisation identity) to

$$f = \frac{1}{A}\sum_{j\in J}\langle f, v_j\rangle v_j, \qquad \forall f \in \mathcal{H}. \tag{2.1.2}$$

The ability to decompose vectors using this inner product expansion gives tight frames similar utility to orthogonal bases. It allows for efficient encoding and decoding of information. When redundancy is needed, tight frames with redundant expansions can be used.

**Corollary 2.1.4.**

The image of a tight frame under a unitary transformation is still a tight frame.

**Definition 2.1.5.**

Let $\Phi := (v_j)_{j\in J}$ be a finite sequence in $\mathcal{H}$. The **synthesis operator** of $\Phi$ is the linear map

$$V := [v_j]_{j\in J} : \ell_2(J) \to \mathcal{H} : a \mapsto \sum_{j\in J} a_j v_j.$$

The **frame operator** of $\Phi$ is the linear map $S = S_V = VV^* : \mathcal{H} \to \mathcal{H}$,

$$Sf := \sum_{j\in J}\langle f, v_j\rangle v_j, \qquad \forall f \in \mathcal{H}.$$

**Definition 2.1.6.**

Let $\Psi := (f_j)_{j\in J}$ and $\Phi := (g_j)_{j\in J}$ be finite sequences of vectors in a Hilbert space $\mathcal{H}$. Then $\Psi$ and $\Phi$ are **unitarily equivalent** when there exists a unitary transformation $U \in \mathcal{U}(\mathcal{H})$ such that

$$U(f_j) = g_j, \qquad \text{for all } j \in J.$$

**Definition 2.1.7.**

Let $\Phi = (v_j)_{j\in J}$ be a sequence of $n$ vectors. The **Gramian** of $\Phi$ is the $n \times n$ matrix

$$\mathrm{Gram}(\Phi) := V^*V = [\langle v_k, v_j\rangle]_{j,k\in J},$$

i.e., the matrix containing inner products between the vectors of $\Phi$.

**Theorem 2.1.8.**

A $n \times n$ matrix $P = [p_{j,k}]_{j,k\in J}$ is the Gramian of a normalised tight frame $(v_j)_{j\in J}$ if and only if $P$ is an orthogonal projection matrix.

**Corollary 2.1.9.**

Normalised tight frames are unitarily equivalent if and only if their Gramians are equal.

While the Gramian can quickly resolve whether two frames are unitarily equivalent (without reindexing), it may still be an expensive exercise to compute equivalence when reindexing is allowed. The Gramian also fails to identify when two sequences of vectors are essentially the same, but one sequence has had its vectors arbitrarily scaled by a constant of modulus 1 (see Chapter 4).

**Theorem 2.1.10.**

Every finite normalised tight frame is the orthogonal projection of an orthonormal basis.

**Example 2.1.11.**

The Gramian of a finite normalised tight frame $\Phi$ acting on the standard basis will produce a frame which is unitarily equivalent to $\Phi$.

**Definition 2.1.12.**

A tight frame is **equiangular** if its vectors have equal norms and there exists a $C > 0$ such that

$$|\langle v_j, v_k\rangle| = C, \qquad \forall j \neq k.$$

If the vectors are of unit length, they are sometimes called **equiangular lines**.

**Definition 2.1.13.**

Let $\mathbb{S} = \{f \in \mathcal{H} : \|f\| = 1\}$ be the unit sphere in $\mathcal{H}$, where $\mathcal{H}$ has dimension $n$. The **frame potential** is the function

$$FP : \mathbb{S}^n \to [n, \infty) : (v_j)_{j=1}^n \mapsto \sum_{j=1}^{n}\sum_{k=1}^{n} |\langle v_j, v_k\rangle|^2.$$

**Theorem 2.1.14** (Variational characterisation)**.**
Let $\Phi := (v_j)_{j\in J}$ be a frame for $\mathcal{H}$. Then

$$FP(\Phi) \geq \frac{1}{d}\Big(\sum_{j\in J}\|v_j\|^2\Big)^2, \qquad d = \dim(\mathcal{H}),$$

with equality, if and only if, $\Phi$ is a tight frame.

The frame potential was introduced by Benedetto and Fickus in Section 6 of [BF03]. It allows tight frames to be found numerically by minimising the frame potential and applying the variational characterisation. For example, by constructing a random sequence of vectors and attempting to converge towards a tight frame through random perturbations of the vectors which produce smaller frame potentials.

**Definition 2.1.15.**
Let $\Phi := (v_j)_{j\in J}$ be a frame for $\mathcal{H}$. The **second frame potential** is the function

$$SFP : \mathbb{S}^n \to [0,\infty) : (v_j)_{j=1}^n \mapsto \sum_{j=1}^n\sum_{k=1}^n |\langle v_j, v_k\rangle|^4.$$

**Theorem 2.1.16** (Welch bound)**.**
Let $\Phi := (v_j)_{j\in J}$ be a sequence of $n = d^2$ unit vectors in $\mathbb{C}^d$. Then

$$\sum_{j=1}^n\sum_{k=1}^n |\langle v_j, v_k\rangle|^4 \geq \frac{2d^3}{d+1},$$

with equality, if and only if, $\Phi$ is a sequence of equiangular lines, i.e., there exists a $C > 0$ such that for all $j \neq k$, $|\langle v_j, v_k\rangle|^2 = C$.

By a similar approach to finding tight frames, numerical equiangular tight frames can be found by minimising the second frame potential and checking whether they meet the Welch bound.

**Definition 2.1.17.**
Let $G$ be a group. A **group frame** (or $G$**–frame**) for $\mathcal{H}$ is a frame $\Phi = (v_g)_{g\in G}$ where there exists a unitary representation of $G$ with

$$gv_h := \rho(g)v_h = v_{gh}, \qquad \forall g, h \in G.$$

**Theorem 2.1.18.**

Let $\rho$ be an irreducible unitary representation of a group $G$ on $\mathcal{H}$ and $v \neq 0 \in \mathcal{H}$. Then $(\rho(g)v)_{g\in G}$ is a tight frame for $\mathcal{H}$.

This theorem allows for the efficient construction of tight frames as group orbits.

## 2.2 SIC-POVM existence problem (Zauner's conjecture)

Equiangular lines always exist in all finite dimensions. A simple example is a sequence of orthonormal basis vectors which have an inner product of 0 between any two distinct vectors. Naturally one asks, what is the maximum allowable number of equiangular lines in any given dimension? This will be considered for complex vector spaces.

**Theorem 2.2.1.**

Let $d > 1$ and $(f_j)_{j\in J}$ be a sequence of $n$ equiangular lines in $\mathbb{C}^d$, where the $f_j$ are not collinear. The orthogonal projections

$$P_j : f \mapsto \langle f, f_j\rangle f_j, \qquad j = 1, \ldots, n$$

are linearly independent and $n \leq d^2$ with equality, if and only if, $\{P_j\}_{j=1}^n$ is a basis for the $d \times d$ Hermitian matrices.

Over complex vector spaces, an upper bound for the maximum number of equiangular lines in any given dimension is $d^2$ by Theorem 2.2.1. Over real vector spaces, an upper bound is $\frac{d(d+1)}{2}$ (Theorem 2.2 of [Tre08]).

The real dimensional case exhibits several dimensions which do not meet this upper bound (they have smaller upper bounds) and several which do. After more than 60 years of research, it is still not clear for all dimensions what the maximum is.

The complex situation is somehow less dire. The recent computational study [SG10] produced numerical approximations (to about 200 digit precision) of sequences of $d^2$ equiangular lines in every $d \leq 67$. This strongly supports the conjecture that the maximum is $d^2$.

**Definition 2.2.2.**

An equiangular tight frame $\Phi := (v_j)_{j\in J}$ for $\mathbb{C}^d$ having the maximal number of vectors ($d^2$)

is a **maximal equiangular tight frame**. When the vectors are of unit length they are **maximal equiangular lines**. In quantum information theory, they (or their corresponding orthogonal projections $P_j = v_j v_j^*$) are known as a **symmetric informationally complete positive operator valued measure**, **SIC-POVM**.

In light of [SG10] and Theorem 2.2.1, the remaining challenge is to show that $d^2$ equiangular lines can always be constructed for any finite dimension $d$.

**Conjecture 2.2.3** (Zauner's conjecture)**.**
For each $d \in \mathbb{N}$, there exists a sequence of $d^2$ equiangular lines (SIC-POVM) in $\mathbb{C}^d$.

**Theorem 2.2.4.**
Let $\Phi := (v_j)_{j \in J}$ be a tight frame with $|J| = d^2$. Then $\Phi$ is a SIC-POVM if and only if

$$|\langle v_j, v_k \rangle|^2 = \frac{1}{d+1}, \qquad j \neq k.$$

Since SIC-POVMs are tight frames they might be constructed as the orbit of an irreducible group action. Recent efforts have primarily focused on group constructions of SIC-POVMs.[1]

**Definition 2.2.5.**
A SIC-POVM is **covariant** to a group $G$ if it is the orbit of $G$ acting on some vector $v \in \mathcal{H}$.

**Definition 2.2.6.**
Let $\mathcal{H} = \mathbb{C}^d$, $G$ a group with irreducible representation $\rho$ in $\mathcal{H}$. A SIC-POVM **fiducial** is a vector $v \in \mathcal{H}$ such that

$$\Phi := \{\rho(g)v : g \in G\}$$

is a SIC-POVM for $\mathcal{H}$, i.e., it is a vector whose orbit under the action of $G$ is a SIC-POVM.

**Definition 2.2.7.**
Let $\mathbb{F}$ be a field and $\mathcal{H}$ a Hilbert space over $\mathbb{F}$ of dimension $d$. The space of homogeneous multivariate polynomials ($\mathcal{H} \to \mathbb{F}$) of degree $t$ in $z$ and $\bar{z}$ is

$$\mathrm{Hom}(t,t) := \mathrm{span}\{z \mapsto z^\alpha \bar{z}^\beta : |\alpha| = |\beta| = t\},$$

where $\alpha$ and $\beta$ are multi-indices, e.g., $z^\alpha = z_1^{\alpha_1} z_2^{\alpha_2} \cdots z_t^{\alpha_t}$.

[1] Group independent searches of SIC-POVMs were done numerically by Zhu in Chapter 10 of [Zhu12] in a small number of dimensions. All the SIC-POVMs found were covariant to the Weyl-Heisenberg group. One might speculate that only group covariant SIC-POVMs exist.

**Theorem 2.2.8.**

Let $\Phi := (v_j)_{j\in J}$ be a frame satisfying Theorem 2.1.16. Then

$$\int_{\mathbb{S}} p(x)d\sigma(x) = \frac{1}{\sum_{\ell=1}^{n}\|v_\ell\|^4}\sum_{j=1}^{n} p(v_j), \qquad \forall p \in \mathrm{Hom}(t,t), \tag{2.2.1}$$

where $\sigma(x)$ is the normalised surface area measure on $\mathbb{S}$ (see [Sei01]).

In design theory, sequences of vectors satisfying (2.2.1) are **spherical 2-designs**. SIC-POVMs are examples of 2–designs (they are in fact minimal 2–designs). Spherical designs allow integrals of homogeneous polynomials over the unit sphere to be readily computed by evaluating a finite sum. See the classic Seidel paper [DGS77] for more information about spherical designs.

## 2.3 Weyl-Heisenberg group

Let $\omega := e^{2\pi i/d}$ and $(e_i)_{i=1}^{d}$ be an orthonormal basis for $\mathcal{H}$. Define the operators $S$ and $\Omega$ so that

$$S(e_i) = e_{i+1 \bmod d}, \qquad \Omega(e_i) = \omega^i e_i.$$

The $S$ and $\Omega$ are cyclic shift and modulation operators with matrix representations:

$$S := \begin{pmatrix} 0 & 0 & \cdots & 0 & 1 \\ 1 & 0 & \cdots & 0 & 0 \\ 0 & 1 & \cdots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & 1 & 0 \end{pmatrix}, \qquad \Omega := \begin{pmatrix} 1 & & & & \\ & \omega & & & \\ & & \omega^2 & & \\ & & & \ddots & \\ & & & & \omega^{d-1} \end{pmatrix}. \tag{2.3.1}$$

**Definition 2.3.1.**

Let $p = (p_1, p_2)$, $q = (q_1, q_2) \in \mathbb{Z}^2$. Define the *symplectic form*

$$\langle p, q\rangle_s := p_2q_1 - p_1q_2. \tag{2.3.2}$$

The **Weyl-Heisenberg displacement operators** are

$$D_p := \tau^{p_1p_2} S^{p_1}\Omega^{p_2}.$$

The displacement operators generate a discrete **Weyl-Heisenberg group** $\mathrm{H}(d)$ of order $d^3$.

Let $\tau = -e^{i\pi/d}$. The Weyl-Heisenberg operators satisfy the relations

$$D_p D_q = \tau^{\langle p,q\rangle_s} D_{p+q}, \tag{2.3.3}$$

$$D_p^* = D_{-p}, \tag{2.3.4}$$

$$D_{p+dq} = \begin{cases} D_p, & \text{if } d \text{ is odd;} \\ (-1)^{\langle p,q\rangle_s} D_p, & \text{if } d \text{ is even.} \end{cases} \tag{2.3.5}$$

Modulo the centre $Z := \text{Centre}(\text{H}(d))$, $\text{H}(d)/Z$ is isomorphic to $\mathbb{Z}_d \times \mathbb{Z}_d$ of order $d^2$.

The Weyl-Heisenberg group is of special interest to the SIC-POVM existence problem. Almost all constructed SIC-POVMs are covariant to the Weyl-Heisenberg group or is *projectively unitarily equivalent* (see Chapter 4) to a Weyl-Heisenberg SIC-POVM. Furthermore, the only known exception (Hoggar lines) is covariant to a subgroup of the normaliser of the Weyl-Heisenberg group (Clifford group) (see Section A.3 of Appendix A). This makes the Weyl-Heisenberg group or the Clifford group central to the study of SIC-POVMs.

**Theorem 2.3.2** (Theorem 8.1, [Zhu12])**.**
Any group covariant SIC-POVM in a prime dimension is covariant with respect to the Weyl-Heisenberg group.

**Definition 2.3.3** (Fourier matrix)**.**
Let $\omega := e^{2\pi i/d}$. The **Fourier matrix** $F = (f_{(jk)})_{0\leq j,k\leq d-1}$ is the $d \times d$ matrix with entries given by

$$f_{jk} := \frac{\omega^{jk}}{\sqrt{d}}, \qquad 0 \leq j, k \leq d-1.$$

**Definition 2.3.4** (Zauner matrix)**.**
Let $F$ be the Fourier matrix and $G = (g_{rs})_{0\leq r,s\leq d-1}$ be the diagonal matrix with entries given by

$$g_{ss} = e^{\pi i(d+1)s^2/d}, \qquad 0 \leq s \leq d-1.$$

The **Zauner matrix** is the matrix:

$$\mathcal{Z} := e^{i\pi(d-1)/12} FG.$$

**Conjecture 2.3.5** (Zauner's conjecture (stronger version))**.**
For every dimension, there exists a Weyl-Heisenberg covariant SIC-POVM fiducial which is an eigenvector of the Zauner matrix $\mathcal{Z}$.

This stronger version of the conjecture is also supported by the numerical study [SG10]. It is often used as an additional constraint when attempting to solve a polynomial system for SIC-POVMs.

## 2.4 Clifford group

**Definition 2.4.1.**

Let $K : \mathbb{C}^d \to \mathbb{C}^d$ such that

$$Kz := \overline{z} = (\overline{z_j}).$$

Then $K$ is the **complex conjugate operator**. A product of a linear (unitary) map with complex conjugation is an **anti-linear** (**anti-unitary**) map. Anti-unitary maps have the property that

$$\langle Ux, Uy\rangle = \overline{\langle x, y\rangle}, \qquad \forall x, y \in \mathcal{H}.$$

Since the product of two anti-linear (anti-unitary) maps is linear (unitary), the groups of linear maps and unitary maps can be extended to

$$\mathcal{EGL}(\mathbb{C}^d) := \{LK^s : L \in \mathrm{GL}(\mathbb{C}^d), s = 0, 1\}, \qquad \mathcal{EU}(\mathbb{C}^d) := \{UK^s : L \in \mathcal{U}(\mathbb{C}^d), s = 0, 1\}.$$

**Definition 2.4.2.**

Let $\mathrm{H}(d)$ be the Weyl-Heisenberg group as defined by Definition 2.3.1. Then the **Clifford group** is the normaliser of $\mathrm{H}(d)$ in the group of unitary matrices, i.e.,

$$\mathrm{C}(d) := \{U \in \mathrm{U}(d) : U^{-1}\mathrm{H}(d)\,U = \mathrm{H}(d)\}.$$

The **extended Clifford group** is all unitary and anti-unitary operators which normalise $\mathrm{H}(d)$, i.e.,

$$\mathrm{ECL}(d) := \{U \in \mathcal{EU}(\mathbb{C}^d) : U^{-1}\mathrm{H}(d)\,U = \mathrm{H}(d)\}.$$

The Clifford group is important to the study of SIC-POVMs for a variety of reasons. For example, let $U$ be a unitary in the Clifford group, $v \in \mathcal{H}$ a Weyl-Heisenberg SIC-POVM fiducial, $\rho$

an irreducible representation of H($d$) on $\mathcal{H}$, then for all $g \in \mathrm{H}(d)$, $g \neq 1$,

$$|\langle Uv, U\rho(g)Uv\rangle| = |\langle v, U^*\rho(g)Uv\rangle| = |\langle v, U^{-1}\rho(g)Uv\rangle| = \frac{1}{\sqrt{d+1}},$$

i.e., the Clifford group maps a SIC-POVM to a SIC-POVM. Its action on a fiducial partitions the Weyl-Heisenberg fiducials into projectively unitarily equivalent orbits. This is useful for counting the number of projectively inequivalent solutions found (cf. Tables 1,2 of [SG10]). The Clifford group also shows up in the study of projective symmetry groups of SIC-POVMs (see Chapter 5), leads to additional symmetry equations to further constrain the polynomial system to solve (for a SIC-POVM) and has played a role in the discovery of new analytic SIC-POVMs (see Section 7 of [ABB+12]).

Define

$$\bar{d} = \begin{cases} d & \text{if } d \text{ is odd,} \\ 2d & \text{if } d \text{ is even.} \end{cases}$$

Let $\mathrm{SL}(2, \mathbb{Z}_{\bar{d}})$ be the special linear group of $2 \times 2$ matrices over $\mathbb{Z}_{\bar{d}}$ with determinant 1. Let $\mathrm{ESL}(2, \mathbb{Z}_{\bar{d}})$ be the extended special linear group containing $2 \times 2$ matrices over $\mathbb{Z}_{\bar{d}}$ with determinant $\pm 1$.

**Remark 2.4.3.**

$\mathrm{ESL}(2, \mathbb{Z}_{\bar{d}}) = \mathrm{SL}(2, \mathbb{Z}_{\bar{d}}) \cup J\,\mathrm{SL}(2, \mathbb{Z}_{\bar{d}})$, where

$$J = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}. \tag{2.4.1}$$

Define the semidirect product $\mathrm{SL}(2, \mathbb{Z}_{\bar{d}}) \ltimes (\mathbb{Z}_d \times \mathbb{Z}_d)$ with operation

$$(F_1, \chi_1) \circ (F_2, \chi_2) := (F_1F_2, \chi_1 + F_1\chi_2),$$

where $F_1, F_2 \in \mathrm{SL}(2, \mathbb{Z}_{\bar{d}})$, $\chi_1, \chi_2 \in \mathbb{Z}_d \times \mathbb{Z}_d$ and the entries of $F_1$ are taken mod $d$ before computing $F_1\chi_2$. Appleby in Lemma 1 of [App05] implicitly gives the following surjective homomorphism to characterise the structure of the extended Clifford group:

$$f_E : \mathrm{ESL}(2, \mathbb{Z}_{\bar{d}}) \ltimes (\mathbb{Z}_{\bar{d}} \times \mathbb{Z}_{\bar{d}}) \to \mathrm{ECL}(d), \qquad f_E(F, \chi) = U, \tag{2.4.2}$$

such that $UD_kU^* = \omega^{\langle \chi, Fk \rangle} D_{Fk}$, where $F \in \mathrm{SL}(2, \mathbb{Z}_{\bar{d}})$, $k \in \mathbb{Z}^2$ and $D$ is a displacement operator.

If $d$ is odd, $f_E$ is an isomorphism. If $d$ is even,

$$\ker(f_E) := \left\{ \begin{pmatrix} 1+rd & sd \\ td & 1+rd \end{pmatrix}, \begin{pmatrix} sd/2 \\ td/2 \end{pmatrix} : r, s, t \in \{0, 1\} \right\}.$$

Consider

$$F = \begin{pmatrix} \alpha & \beta \\ \gamma & \delta \end{pmatrix} \in \mathrm{ESL}(2, \mathbb{Z}_{\bar{d}}). \tag{2.4.3}$$

Let $e_i$ be the standard basis vectors. Suppose $F$ in (2.4.3) is in $\mathrm{SL}(2, \mathbb{Z}_{\bar{d}})$. If $\beta$ is invertible in $\mathbb{Z}_{\bar{d}}$, the image of $(F, \chi)$ under $f_E$ is $D_\chi V_F$ where:[2]

$$V_F = \frac{1}{\sqrt{d}} \sum_{r,s=0}^{d-1} \tau^{\beta^{-1}(\alpha s^2 - 2rs + \delta r^2)} e_r e_s^*. \tag{2.4.4}$$

If $\beta$ is not invertible in $\mathbb{Z}_{\bar{d}}$ then $V_F = V_{F_1} V_{F_2}$ where:[3]

$$F_1 = \begin{pmatrix} 0 & -1 \\ 1 & x \end{pmatrix}, \qquad F_2 = \begin{pmatrix} \gamma + x\alpha & \delta + x\beta \\ -\alpha & -\beta \end{pmatrix}. \tag{2.4.5}$$

Note that $x$ can be chosen so that $\delta + x\beta$ is invertible, so $V_{F_2}$ can be computed with (2.4.4).

Let $J$ be from (2.4.1) and $\mathbf{0} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$, then $f_E(J, \mathbf{0})$ produces the complex-conjugation operator $\hat{J}$ (equation 106 of [App05]):

$$\hat{J} : \sum_{r=0}^{d-1} a_r e_r \mapsto \sum_{r=0}^{d-1} a_r^* e_r. \tag{2.4.6}$$

If $\det(F) = -1$ then $\det(FJ) = 1$ so (2.4.4) together with $f_E(J, 0)$, determines the elements of $\mathrm{ESL}(2, \mathbb{Z}_{\bar{d}}) \ltimes (\mathbb{Z}_d \times \mathbb{Z}_d)$.

Under the Appleby homomorphism, the Zauner matrix (in dimension $d$) becomes

$$F_z = \begin{pmatrix} 0 & d-1 \\ d+1 & d-1 \end{pmatrix}.$$

[2] See Lemma 2 of [App05].
[3] See Lemma 4 of [App05].

**Definition 2.4.4** (Clifford trace)**.**
Let $\mathrm{I}(d)$ be the set of all matrices $e^{\theta i} I_d$. Let $[F, \chi] \in \mathrm{ECL}(d)/\mathrm{I}(d)$ be the image of $(F, \chi)$ under $f_E$. Then the **Clifford trace** of any $U \in [F, \chi]$ is $\mathrm{Tr}(F)$ mod $d$.

The Clifford trace is interesting because with the exception of 3 dimensions, it is both necessary and sufficient for Clifford unitaries of order 3 to have Clifford trace $-1$ (Lemma 7 of [App05]). The sole exception in dimension 3 is due to the identity operator having Clifford trace $-1$. Non-identity Clifford unitaries with Clifford trace $-1$ are sometimes referred to as **canonical order 3 unitaries**.

All known Weyl-Heisenberg SIC-POVM fiducials are eigenvectors of canonical order 3 elements of the Clifford group. In particular, the Zauner matrix $\mathcal{Z}$ is a canonical order 3 element. Not all Weyl-Heisenberg SIC-POVM fiducials are eigenvectors of $\mathcal{Z}$ however. There exist other canonical order 3 unitaries for which SIC-POVM fiducials are eigenvectors. For example (with the Appleby indexing),

$$F_a := \begin{pmatrix} 1 & d+3 \\ d+3k & d-2 \end{pmatrix}, \qquad F_b := \begin{pmatrix} -k & d \\ d & d-k \end{pmatrix}, \qquad F_c := \begin{pmatrix} s & d-2s \\ d+2s & d-s \end{pmatrix},$$

where $k \in \mathbb{Z}_{\tilde{d}}$, $s = 3k^2 \pm k + 1$, and $d = 9k+3$ for $F_a$, $d = k^2 - 1$ for $F_b$ and $d = (3k \pm 1)^2 + 3$ for $F_c$ (See (4.7), (4.8), (4.9) of [SG10]).

**Definition 2.4.5.**
A **monomial** (or **generalised permutation**) **matrix** is a $d \times d$ matrix with exactly one nonzero entry (of modulus 1) in each row and column.

**Theorem 2.4.6** (Theorem 3, [ABB+12])**.**
There exists a monomial representation of the Clifford group which contains the Weyl-Heisenberg group as an irreducible subgroup if and only if the dimension is a square, i.e., $d = n^2$.

Comparing Theorem 2.4.6 with Theorem A.2.2, note that it is possible to find monomial representations for groups (e.g., nice error groups) which give rise to SIC-POVM orbits even when not of square dimension. The significance of the Clifford group being monomial is that it allows for canonical order 3 unitaries like the Zauner matrix to take a simplified representation. This can lead to a reduced set of equations to solve.

An example is $d = 16$ where a new analytic solution was obtained by considering the monomial representation of the Zauner matrix (Section 7 of [ABB+12]). The symmetry equations under

the monomial representation were then used to form a system of multivariate polynomials and the method of Gröbner bases is applied to solve for the fiducial. This feat was not achievable with the standard representation when attempted by the [SG10] study.

## 2.5 Three dimensional SIC-POVMs

The only known continuous families of SIC-POVMs exist in 3 dimensions. For every other dimension, computational searches strongly suggest that the number of Weyl-Heisenberg SIC-POVMs is finite ([SG10]).

**Example 2.5.1.**

The family of fiducial vectors

$$v(t) := \frac{1}{\sqrt{2}}(0, 1, -e^{it})^T, \qquad t \in \mathbb{T},$$

gives a continuous family of Weyl-Heisenberg SIC-POVMs.

Another curiosity of 3 dimensional SIC-POVMs arises when observed through the looking glass of elliptic curves.

**Definition 2.5.2.**

An **elliptic curve** is a cubic of the form

$$P = x^3 + y^3 + z^3 + txyz, \qquad t \in \mathbb{C}.$$

The inflection points on the elliptic curve are given by the points where the Hessian $H$ vanishes, i.e.,

$$P = H = \det \partial_i \partial_j P = (6^3 + 2t^3)xyz - 6t^2(x^3 + y^3 + z^3) = 0.$$

By Bézout's theorem, two cubics in the complex projective plane intersect at nine points, the nine inflection points. These are given by

$$\begin{pmatrix} x \\ y \\ z \end{pmatrix} = \left\{ \begin{pmatrix} 0 \\ 1 \\ -1 \end{pmatrix} \begin{pmatrix} 0 \\ 1 \\ -q \end{pmatrix}, \begin{pmatrix} 0 \\ 1 \\ -q^2 \end{pmatrix}, \begin{pmatrix} -1 \\ 0 \\ 1 \end{pmatrix}, \begin{pmatrix} -q \\ 0 \\ 1 \end{pmatrix}, \begin{pmatrix} -q^2 \\ 0 \\ 1 \end{pmatrix}, \begin{pmatrix} 1 \\ -1 \\ 0 \end{pmatrix}, \begin{pmatrix} 1 \\ -q \\ 0 \end{pmatrix}, \begin{pmatrix} 1 \\ -q^2 \\ 0 \end{pmatrix} \right\},$$

where $q = e^{2\pi i/3}$ is a primitive third root of unity.

These nine inflection points form a Weyl-Heisenberg SIC-POVM. They also represent the geometric configuration of 9 points and 12 lines over the complex projective plane in which every line has 3 points and every point is on lines. This is known as the *Hesse configuration*. In design theory, the Hesse configuration is known as an affine plane of order 3. See also [Hug07], [Ben10].

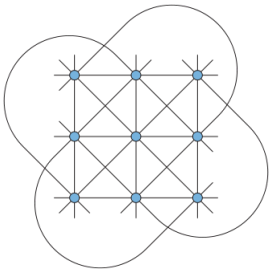


FIGURE 2.5.1: Hesse configuration.[4]

At this stage it is unclear whether this coincidence between the inflection points of elliptic curves and SIC-POVMs is specialised to 3 dimensions or whether a deeper theory is at work. Attempts to generalise this construction have so far yielded little success (cf. [Ben10]).

[4]Diagram by David Eppstein. Source: https://upload.wikimedia.org/wikipedia/commons/thumb/e/eb/Hesse_configuration.svg/360px-Hesse_configuration.svg.png

# Chapter 3

# Reverse engineering numerical SIC-POVMs

In [RBKSC04], SIC-POVMs were numerically approximated (up to machine precision) for all dimensions $d \leq 45$ by minimising the second frame potential. The recent computational study [SG10] extends this to $d \leq 67$.

This chapter explores ways to utilise this information and attempt to recover analytic SIC-POVMs from the numerical approximations to prove the existence of SIC-POVMs in new dimensions.

Numerical SIC-POVM data has proved to be very useful in the calculation of analytic SIC-POVMs. In some cases, the components of a fiducial could be easily guessed (e.g., if a component is $0$). In other cases, various relationships emerged from the numerical data, allowing for extra symmetries and reduced equations (e.g., Theorems 6.2 and 6.6 of [BW07] or the comments on p. 8 of [SG10], which made SIC-POVMs in dimensions $24, 35, 48$ viable to calculate).

Recent efforts to study the field structure of SIC-POVM fiducials ([SG10], [AYAZ13]) have led to some remarkable conjectures (see Section 7 of [AYAZ13]). By leveraging these conjectures, a systematic framework for attempting to reverse engineer SIC-POVM fiducials from numerical SIC-POVMs was developed with Appleby et al. These ideas have fascinating analogies with the theory of complex multiplication (see Chapter 6 of [ST92]).

Using this framework, the aim will be to recover an analytic expression for a SIC-POVM using its numerical approximation. This technique is applied to obtain a previously unknown SIC-POVM in 17 dimensions.

## 3.1 Galois theory fundamentals

The material in this section can be found in standard algebra text books (e.g., Chapter 4 of [Jac85], Chapters 5,6 and 7 of [Rom06], Chapter 32 of [Gal09], Chapter 9 of [Fra03] ). They are reproduced without proof for completeness.

**Definition 3.1.1.**

Let $\mathbb{E}$ and $\mathbb{F}$ be fields such that $\mathbb{F} \subset \mathbb{E}$ and $\mathbb{E}$ inherits the field operations of $\mathbb{F}$. Then $\mathbb{E}$ is a **field extension** of $\mathbb{F}$.

**Definition 3.1.2.**

Let $\mathbb{F}$ be a field and $\alpha \notin \mathbb{F}$. Then $\mathbb{E} := \mathbb{F}(\alpha)$ is the smallest field extension of $\mathbb{F}$ containing $\alpha$ ($\mathbb{F}$ is **extended** by $\alpha$). If $\alpha$ is the root of a non zero polynomial with coefficients from $\mathbb{F}$ then $\mathbb{E}$ is an **algebraic extension**.

**Definition 3.1.3.**

Let $\mathbb{E}$ be an algebraic extension of $\mathbb{F}$ and $\alpha \in \mathbb{E}$. The **minimal polynomial** of $\alpha$ is the monic polynomial $f$ of smallest degree in $F[x]$, such that $f(\alpha) = 0$.

Extension fields can be viewed as vector spaces over the base field and the **degree** of the extension is the dimension of the corresponding vector space. The degree of an extension $\mathbb{E}$ over $\mathbb{F}$ is often written $[\mathbb{E} : \mathbb{F}]$.

**Proposition 3.1.4.**

Let $\mathbb{E} := \mathbb{F}(\alpha)$. The degree of the minimal polynomial $f$ of $\alpha$ over $\mathbb{F}$ is the degree of the extension, i.e., $[\mathbb{E} : \mathbb{F}]$ =deg($f$).

**Theorem 3.1.5.**

Let $u$ be an element of an extension field $\mathbb{E}$ of a field $\mathbb{F}$. Then $u$ is algebraic over $\mathbb{F}$ if and only if $\mathbb{F}(u)$ is finite dimensional over $\mathbb{F}$.

**Definition 3.1.6.**

Let $\mathbb{F}$ be a field, $f(x)$ a monic polynomial in $\mathbb{F}[x]$. Then an extension field $\mathbb{E}$ over $\mathbb{F}$ (or $\mathbb{E}/\mathbb{F}$) is

called a **splitting field** over $\mathbb{F}$ of $f(x) \in \mathbb{E}[x]$ if

$$f(x) = (x - r_1)(x - r_2) \cdots (x - r_n),$$

and

$$\mathbb{E} = \mathbb{F}(r_1, r_2, \ldots, r_n).$$

**Definition 3.1.7.**

A polynomial is **separable** if its roots are all distinct.

**Definition 3.1.8.**

Let $\mathbb{E}$ be an extension of $\mathbb{F}$. Then $\mathbb{E}/\mathbb{F}$ is **separable** if the minimal polynomial of every element of $\mathbb{E}$ is separable. $\mathbb{E}/\mathbb{F}$ is **normal** if every irreducible polynomial in $\mathbb{F}[x]$ which has a root in $\mathbb{E}$ is a product of linear factors in $\mathbb{E}[x]$ (i.e., the polynomial "**splits**" in $\mathbb{E}[x]$).

Note that all algebraic extensions of fields of characteristic 0 are separable. In particular, all algebraic extensions of $\mathbb{Q}$ are separable, i.e., all extensions considered in this thesis are automatically separable.

**Definition 3.1.9.**

An $n$–$th$ root of unity is a root of the polynomial $x^n - 1$.

**Definition 3.1.10.**

A cyclotomic extension is an extension of the form $\mathbb{Q}(\omega)$, where $\omega$ is an $n$–$th$ root of unity.

**Definition 3.1.11.**

Let $G$ be a subgroup of the automorphism group of a field $\mathbb{E}$. Then

$$\mathrm{Fix}(G) := \{x \in \mathbb{E} \,|\, \varphi(x) = x,\ \varphi \in G\}.$$

**Theorem 3.1.12.**

Let $\mathbb{E}$ be an extension field of a field $\mathbb{F}$. Then the following conditions on $\mathbb{E}/\mathbb{F}$ are equivalent:

1. $\mathbb{E}$ is a splitting field over $\mathbb{F}$ of a separable polynomial $f(x)$.
2. $\mathbb{F} = \mathrm{Fix}(G)$ for some finite group $G$ of automorphisms of $\mathbb{E}$.
3. $\mathbb{E}$ is finite dimensional, normal and separable over $\mathbb{F}$.

An extension satisfying one of the conditions of Theorem 3.1.12 is a **Galois extension**.

**Definition 3.1.13** (Galois group)**.**
Let $\mathbb{E}$ be a Galois extension of $\mathbb{F}$ and $\mathcal{G}$ the set of automorphisms of $\mathbb{E}/\mathbb{F}$, i.e., the automorphisms of $\mathbb{E}$ that fix all elements of $\mathbb{F}$. Then $\mathcal{G}$ is the **Galois group** of $\mathbb{E}$ over $\mathbb{F}$ and is sometimes written $\mathrm{Gal}(\mathbb{E}/\mathbb{F})$. Elements of $\mathcal{G}$ are called **Galois automorphisms**.

**Theorem 3.1.14** (Fundamental theorem of Galois theory)**.**
Let $\mathbb{E}$ be a Galois extension of $\mathbb{F}$, $\mathcal{G}$ be the Galois group of $\mathbb{E}$ over $\mathbb{F}$, $\Gamma = \{H\}$ be the set of subgroups of $\mathcal{G}$, $\Sigma$ be the set of subfields of $\mathbb{E}/\mathbb{F}$ (intermediate fields between $\mathbb{F}$ and $\mathbb{E}$). Then there is a one-to-one correspondence between $\Sigma$ and $\Gamma$. Furthermore,

1. $H_1, H_2 \in \Gamma$ and $H_1 \subset H_2 \iff \mathrm{Fix}(H_2) \subset \mathrm{Fix}(H_1)$.
2. $H \in \Gamma$, $|H| = [\mathbb{E} : \mathrm{Fix}(H)]$, $[\mathcal{G} : H] = [\mathrm{Fix}(H) : F]$.
3. $H$ is normal in $\mathcal{G} \iff \mathrm{Fix}(H)$ is normal over $\mathbb{F} \iff \mathrm{Gal}(\mathrm{Fix}(H)/F) \simeq \mathcal{G}/H$.

Theorem 3.1.14 says that each subgroup of the Galois group of $\mathbb{E}$ over $\mathbb{F}$ corresponds to each of the subfields in the tower of extensions from $\mathbb{F}$ to $\mathbb{E}$. Furthermore, a normal subgroup of the Galois group corresponds to a normal field extension over $\mathbb{F}$.

**Definition 3.1.15.**
Let $[\mathbb{E} : \mathbb{F}]$ be a finite Galois extension. Then $\mathbb{E}$ over $\mathbb{F}$ is an **abelian extension** if $\mathrm{Gal}(\mathbb{E}/\mathbb{F})$ is abelian.

**Theorem 3.1.16** (Kronecker-Weber)**.**
Every abelian extension of $\mathbb{Q}$ is contained in a cyclotomic extension.

**Remark 3.1.17.**
The Kronecker-Weber and the fundamental theorem of Galois theory are deep powerful theorems in mathematics which took great time and effort from mathematicians to get a good handle on and have interesting applications within other areas of mathematics.

**Definition 3.1.18.**
Let $G$ be a group. A **normal series** is a sequence of subgroups such that

$$G = G_1 \trianglerighteq G_2 \trianglerighteq \cdots \trianglerighteq G_s \trianglerighteq G_{s+1} = 1.$$

**Definition 3.1.19.**

A group $G$ is **soluble** (**solvable**) if it has a normal series and the factors in the sequence

$$G_1/G_2,\, G_2/G_3\, \ldots, G_s/G_{s+1} \simeq G_s.$$

are all abelian.

**Definition 3.1.20.**

Let $f(x) \in \mathbb{F}[x]$ be monic of positive degree. The equation $f(x) = 0$ is said to be **soluble by radicals** over $\mathbb{F}$ if there exists an extension field $\mathbb{E}/\mathbb{F}$ possessing a tower of subfields

$$\mathbb{F} = \mathbb{F}_1 \subset \mathbb{F}_2 \subset \cdots \subset \mathbb{F}_{r+1} = \mathbb{E},$$

where each $\mathbb{F}_{i+1} = \mathbb{F}_i(d_i)$, $d_i^{n_i} = a_i \in \mathbb{F}_i$, and $\mathbb{E}$ contains a splitting field over $\mathbb{F}$ of $f(x)$.

As each subfield in the tower from Definition 3.1.20 is generated by the previous field by adding a radical ( $\sqrt[n_i]{a_i}$), the extension field $\mathbb{E}$'s elements are always expressible as radical combinations (potentially nested) of elements from $\mathbb{F}$. Since $f(x)$ splits over some intermediate extension of $\mathbb{F}$ and $\mathbb{E}$ containing only radical combinations (potentially nested), its roots must be expressible as radical combinations (potentially nested).

**Theorem 3.1.21.**

An equation $f(x) = 0$ is soluble by radicals over a field $\mathbb{F}$ of characteristic 0 if and only if its Galois group is soluble.

**Remark 3.1.22.**

The theory of solutions to polynomial equations has been of great interest to mathematicians over time and was of great interest to Galois. While Abel was the first to show that no general algebraic solutions existed for quintic equations, Galois theory laid a pathway for a slick characterisation of when polynomial equations would yield algebraic solutions.

### 3.1.1 Representing roots as radicals

Although Theorem 3.1.21 indicates when polynomial equations have radical roots, the statement of the theorem alone gives no indication as to how such roots can be obtained from a polynomial equation that is soluble by radicals.

Given a polynomial equation $f(x) = 0$ that is soluble by radicals, one general approach to recovering the roots as radicals is to formulate the splitting field $\mathbb{E}$ of $f(x)$, calculate its Galois group and the appropriate subgroup tower (normal series). Next, calculate the corresponding subfield tower of $\mathbb{E}$ over $\mathbb{F}$ and at each level of the subfield tower, find the radical representation of each cyclic extension as hinted by Definition 3.1.20. Once this is complete, each of the extension field generators can be represented as radicals and the polynomial can be factorised over $\mathbb{E}$ with the field generators converted to their radical representation.

In general this is not a trivial exercise, as it assumes one can readily compute things like the Galois group, tower of field extensions or factorise polynomials easily over field extensions. While some researchers have studied more sophisticated methods for approaching this problem (c.f., [AY02], Problem 2.7.5 of [Stu08]), the naive approach outlined was sufficient for this project.

## 3.2 SIC-POVM field structure

Let $v$ be a SIC-POVM fiducial and $\Pi := vv^*$ be the corresponding fiducial projector. Consider the minimal field extension (over $\mathbb{Q}$) containing the elements of the projector (with standard basis representation)[1], $\tau = -e^{i\pi/d}$ and let the Galois group of this field extension over $\mathbb{Q}$ be $\mathcal{G}$. Let $q = (i, j) \in \mathbb{Z}_d{}^2$, $\Pi_q$ be the SIC-POVM projector $D_q \Pi D_q^*$, where $D_q$ is a Weyl-Heisenberg displacement operator.

**Definition 3.2.1.**

A SIC-POVM **overlap** is

$$\chi_q := \mathrm{Tr}(\Pi D_q).$$

**Definition 3.2.2.**

Let $A, B$ be two square matrices in $M_d(\mathbb{C})$. The Hilbert-Schmidt (Frobenius) inner product between $A, B$ is

$$\langle A, B\rangle := \mathrm{trace}(AB^*), \qquad A, B \in M_d(\mathbb{C}).$$

**Example 3.2.3.**

The Weyl-Heisenberg displacement operators form an orthogonal basis for the space $M_d(\mathbb{C})$ of $d \times d$ matrices relative to the Hilbert-Schmidt inner product.

[1] the choice of basis is important as a complicated basis could even result in transcendental extensions

The overlaps are the coefficients of the frame expansion (with Hilbert-Schmidt inner product)

$$\Pi = \frac{1}{d} \sum_{q \in \mathbb{Z}_d{}^2} \langle \Pi, D_q^* \rangle D_q = \frac{1}{d} \sum_{q \in \mathbb{Z}_d{}^2} \chi_q D_q. \tag{3.2.1}$$

Because SIC-POVM overlaps are the coefficients for the frame expansion of a fiducial, they uniquely determine a fiducial projector. The frame expansion formula also suggests a very close relationship between the minimal field the overlaps reside in (as an extension of $\mathbb{Q}$) and the minimal field of the fiducial projectors. These fields were first studied in [SG10] and later by [AYAZ13]. Although knowing the minimal fields does not prove SIC-POVM existence, it yields interesting insight into the complexity of the fiducials and provides important information when attempting to reverse engineer a numerical solution.

Unless otherwise indicated, the results in this section arise from [AYAZ13] and discussions with Appleby.

**Definition 3.2.4.**

Let

$$a := \sqrt{\frac{(d-3)(d+1)}{k}},$$

where $k$ is the biggest perfect square that divides $(d-3)(d+1)$. Let $\mathbb{E}$ be the smallest normal extension of $\mathbb{Q}$ containing the components of $\Pi$ (with respect to the standard basis), $\tau = -e^{i\pi/d}$, and $\bar{\mathbb{E}} = \mathbb{E}(\sqrt{d})$.

**Conjecture 3.2.5.**

$\mathbb{E}$ and $\bar{\mathbb{E}}$ are abelian extension fields of the real quadratic field $\mathbb{Q}(a)$.

This conjecture is true in all cases studied so far (see [AYAZ13]).

**Definition 3.2.6.**

Let $\Pi$ be a fiducial projector. The **stabiliser** of $\Pi$ is the group

$$S_\Pi := \{U \in \mathrm{ECL}(d) : U\Pi U^* = \Pi\}.$$

**Definition 3.2.7.**

Define $U_J$ to be the anti-unitary which acts by complex conjugation on the standard basis, i.e.,

$$U_J(v) = \sum_{r \in \mathbb{Z}_d} \overline{\langle v, e_r \rangle} e_r, \quad \text{for all vectors } v.$$

**Definition 3.2.8.**

Let $\bar{\mathcal{G}} = \bar{\mathcal{G}}_{\mathbb{E}}$ be the Galois group of $\bar{\mathbb{E}}$ over $\mathbb{Q}$. For each $g \in \bar{\mathcal{G}}$ and linear operator $\Gamma$ defined in terms of the standard basis and with its elements in $\bar{\mathbb{E}}$, define

$$g(\Gamma) = \sum_{i,j \in \mathbb{Z}_d} g\Big(\langle \Gamma(e_j), e_i \rangle\Big) e_i e_j^*.$$

If $\Gamma$ is an anti-linear operator in $\bar{\mathbb{E}}$, define

$$g(\Gamma) = g(\Gamma U_J) U_J, \qquad U_J \text{ from Definition 3.2.7.}$$

**Definition 3.2.9.**

Let $g_c \in \bar{\mathcal{G}}$ be the complex conjugation operator. Then $\bar{\mathcal{G}}_c$ is the centraliser of $g_c$.

**Theorem 3.2.10.**

If $\Pi'$ is a fiducial projector in $\bar{\mathbb{E}}$ and $g \in \bar{\mathcal{G}}_c$ then $g(\Pi')$ is also a fiducial projector.

In other words, as long as the Galois automorphisms considered commute with complex conjugation, fiducial vectors will be mapped to other fiducial vectors under the action of the Galois automorphism.

**Theorem 3.2.11.**

Let $\bar{\mathcal{G}}_0$ be the set of $g \in \bar{\mathcal{G}}_c$ such that $g(\Pi)$ is in the same extended Clifford orbit as $\Pi$. Then $\bar{\mathcal{G}}_0$ is a subgroup of $\bar{\mathcal{G}}_c$.

**Theorem 3.2.12.**

For all $g \in \bar{\mathcal{G}}_0$ and $p \in \mathbb{Z}_{\bar{d}} \times \mathbb{Z}_{\bar{d}}$, there exists $G \in \mathrm{GL}(2, \mathbb{Z}_{\bar{d}})$ (depends on $g$) and a vector $r_g \in \mathbb{Z}_{\bar{d}} \times \mathbb{Z}_{\bar{d}}$ (depends on $g$) such that

$$g(\chi_p) = \begin{cases} \chi_{Gp}, & d \neq 0 \bmod 3 \\ \sigma^{\langle r_g, p \rangle} \chi_{Gp}, & d = 0 \bmod 3 \end{cases}$$

where $\sigma = e^{2\pi i/3}$.

**Definition 3.2.13.**

Let $V_F$ be of the form (2.4.4). A fiducial $\Pi$ for which $S_\Pi$ consists only of unitaries and anti-unitaries of the form $e^{i\xi} V_F$ is **displacement-free**.

For displacement-free fiducials $\Pi$, define

$$\tilde{S}_\Pi = \left\{F \in \mathrm{ESL}(2, \mathbb{Z}_{\bar{d}}) : V_F \in S_\Pi\right\},$$

and

$$\bar{S}_\Pi = \left\{(\mathrm{Det}\, F)F : F \in \tilde{S}_\Pi\right\}.$$

**Lemma 3.2.14.**

Let $G \in \mathrm{ESL}(2, \mathbb{Z}_{\bar{d}})$. Then the following statements are equivalent

1. $G \in \bar{S}_\Pi$.
2. $\chi_{G_p} = \chi_p$ for all $p \in \mathbb{Z}_{d^2}$.

These lemmas and theorems provide a good understanding of the action of Galois groups on the overlaps, aiding the reverse engineering efforts.

**Definition 3.2.15.**

A fiducial $\Pi$ is **simple** if

1. $S_\Pi$ contains a canonical order 3 unitary, and
2. $S_\Pi$ is displacement-free.

Interestingly, all known Weyl-Heisenberg fiducial orbits in the extended Clifford group (both numerical and analytic) contain simple fiducials. Many of the expressions involved simplify when dealing with simple fiducials as the name suggests. The fiducials listed in the [SG10] study are all simple fiducials.

**Definition 3.2.16.**

A **singlet** arises when a SIC-POVM orbit (under the Clifford group action) is stabilised by the Galois group. A **doublet** is when two different SIC-POVM Clifford orbits are related via a Galois automorphism, and a $n$**–tuplet** is when $n$ orbits are related by Galois action.

While the possibility of there being $n$-tuplets ($n > 2$) is left open, it should be noted that all analytic solutions found in the Scott-Grassl study have been singlets and doublets. This chapter focuses primarily on the analysis of singlets.

**Conjecture 3.2.17.**

Let $\bar{C}_\Pi$ be the **centraliser** of $\bar{S}_\Pi$ (as a subgroup of $\mathrm{GL}(2,\mathbb{Z}_{\bar{d}})$). Then $\mathbb{E}$ has the following field and Galois group structure (refer to equations 157, 158, 165 on pages 19-20 of [AYAZ13]).

Singlet situation:

$$\mathbb{Q} \trianglelefteq \mathbb{Q}(a) \trianglelefteq \mathbb{E}_1 \trianglelefteq \mathbb{E},$$

where $\mathrm{Gal}(\mathbb{Q}(a)/\mathbb{Q}) = \mathbb{Z}_2$, $\mathrm{Gal}(\mathbb{E}_1/\mathbb{Q}(a)) = \bar{C}_\Pi/\bar{S}_\Pi$ and $\mathrm{Gal}(\mathbb{E}/\mathbb{E}_1) = \mathbb{Z}_2$.

Doublet situation:

$$\mathbb{Q} \trianglelefteq \mathbb{Q}(a) \trianglelefteq \mathbb{E}_0 \trianglelefteq \mathbb{E}_1 \trianglelefteq \mathbb{E},$$

where $\mathrm{Gal}(\mathbb{Q}(a)/\mathbb{Q}) = \mathbb{Z}_2$, $\mathrm{Gal}(\mathbb{E}_0/\mathbb{Q}(a)) = \mathbb{Z}_2$, $\mathrm{Gal}(\mathbb{E}_1/\mathbb{Q}(a)) = \bar{C}_\Pi/\bar{S}_\Pi$ and $\mathrm{Gal}(\mathbb{E}/\mathbb{E}_1) = \mathbb{Z}_2$.

When $\bar{\mathbb{E}} \neq \mathbb{E}$, $\mathrm{Gal}(\bar{\mathbb{E}}/\mathbb{E}) = \mathbb{Z}_2$.

Note that these conjectures imply that the Galois group for $\mathbb{E}$ or $\bar{\mathbb{E}}$ over $\mathbb{Q}$ is soluble.

**Theorem 3.2.18.**

Let $\Pi$ be a simple fiducial such that $\bar{S}_\Pi$ is abelian and contains a matrix $F$ conjugate to $F_z$. Then $\bar{C}_\Pi$ consists of all matrices in $\mathrm{GL}(2,\mathbb{Z}_{\bar{d}})$ which can be written in the form

$$nI + mF$$

for some $n, m \in \mathbb{Z}_{\bar{d}}$. In particular, $\bar{C}_\Pi$ and consequently $\bar{C}_\Pi/\bar{S}_\Pi$, are abelian.

Note that this theorem applies to SIC-POVM projectors with Zauner symmetry $F_z$. Interestingly, in all known cases, $\bar{S}_\Pi$ is abelian.

**Theorem 3.2.19.**

Let $d = 9k + 3$ for some positive integer $k$ and let $F \in \mathrm{SL}(2,\mathbb{Z}_{\bar{d}})$ be conjugate to $F_a$. Let $C_F$ be the centraliser of $\langle F \rangle$ considered as a subgroup of $\mathrm{GL}(2,\mathbb{Z}_{\bar{d}})$. Let $G$ be any solution to the equation $3G = F - I$. Then $C_F$ consists of all matrices in $\mathrm{GL}(2,\mathbb{Z}_{\bar{d}})$ which can be written in the form

$$nI + mG + \frac{\bar{d}}{3}H,$$

for some $n, m \in \mathbb{Z}_{\bar{d}}$ and arbitrary matrix $H$.

Unlike the previous theorem, this theorem only applies to non Zauner SIC-POVM projectors. Note that $C_F$ is not always abelian and figuring out the structure of $C_F$ for non $F_z$ symmetries (e.g., $F_a$) is another reason why reverse engineering maybe a useful endeavour. However in the two known analytic solutions where the symmetry is $F_a$ instead of $F_z$ ($12b$ and $48g$ in the [SG10] indexing), the group $C_F$ is abelian. For orbits $21e$, $30d$, $39i, j$, $48e$, where only numerical solutions are known, the group appears to be non-abelian. Being able to obtain analytic solutions in those 5 cases would also be interesting in order to learn about their $C_F$ structure.

The solubility of the Galois group is another key conjecture in pushing this forward. All known analytic equiangular lines so far have had soluble Galois groups. If the Galois group was not thought to be soluble, then this whole approach would not work.

Let $p \in \mathbb{Z}_d \times \mathbb{Z}_d$. Then applying the action of the Galois group of $\mathbb{E}_1$ over $\mathbb{Q}(a)$ (for a singlet) or $\mathbb{E}_0$ (for a doublet), i.e., letting $\bar{C}_\Pi / \bar{S}_\Pi$ act on the overlaps, will split the overlaps into various orbits. Within each of these orbits, the elements will be closed under the action of the Galois group. This implies that the polynomials formed from $\prod_{p \in \text{orbit}} (x - \chi_p)$ are invariant under the Galois actions, and hence the coefficients of this polynomial must be in $\mathbb{E}_0$ (or $\mathbb{Q}(a)$).

**Conjecture 3.2.20** (Appleby)**.**
Let $\mathcal{G}$ (the Galois group of $\mathbb{E}_1$ over $\mathbb{Q}(a)$) act on the overlaps of a Weyl-Heisenberg SIC-POVM fiducial $\Pi$ (singlet or doublet) and $\mathcal{O}$ be one of the orbits under this action. Then $\mathbb{Q}(a)$ is the base field of the polynomial

$$\rho := \prod_{\chi \in \mathcal{O}} (x - \chi).$$

Observe that Conjecture 3.2.20 is a natural consequence of $\mathcal{G}$ fixing $\mathbb{Q}(a)$. Since $\mathcal{G}$ also fixes the polynomial $\rho$, its coefficients must come from $\mathbb{Q}(a)$.

**Conjecture 3.2.21** (Appleby)**.**
For fiducials with Zauner symmetry, the Galois group of $\bar{\mathbb{E}}$ over $\mathbb{Q}(a)$ when $d = 5 \bmod 6$ is cyclic.

These results and theories about the Galois group and the overlaps conspire to direct most of the focus to constructing overlaps polynomials and solving these polynomials for analytic roots. The next section deals with this procedure in more detail.

## 3.3 Methodology

The basic procedure for reverse engineering will be outlined with the details of each step in the following subsections. It assumes that Conjectures 3.2.17, 3.2.20 and conj:5mod6cyclic hold . A modified version will be presented in Section 3.4 which attempts to leverage Conjecture 3.2.21.

The procedure is outlined as follows:

(i) Precision bump the numerical fiducials to a high degree.

(ii) Generate overlaps and apply the action of the Galois group to the overlaps.

(iii) For each orbit, form a polynomial using all the overlaps as the roots.

(iv) Find and construct the necessary tower of field extensions to split the overlaps polynomials.

(v) Factorise the overlaps polynomials over the tower of extensions.

(vi) Match the analytic overlaps with the numerical ones.

(vii) Reconstruct the fiducial vector.

### 3.3.1 Precision bumping

The numerical SIC-POVMs found in [SG10] are of relatively low accuracy. One of the key ideas in this procedure is to use the conjectured field structure to exactly identify what the overlaps look like. Since field extensions are vector spaces, elements of extension fields over $\mathbb{Q}$ can be written as $\mathbb{Q}$-linear (and hence $\mathbb{Z}$-linear) combinations of basis elements. This involves using integer relation algorithms like LLL or PSLQ which have ready implementations in packages like Mathematica or Maple. In order for these algorithms to function correctly, a threshold precision must be present. This could be a few hundred digits of accuracy or sometimes more than a thousand, so precision bumping is necessary. Figuring this out is an ad hoc process. For the 17c SIC-POVM, 2000 digit precision was used.

To precision bump a SIC-POVM, a couple of known options are available. One option is to let the numerical SIC-POVM be the initial starting vector for a method that attempts to find a numerical SIC-POVM and demand a higher precision in the procedure. For example, by

demanding a lower error tolerance when attempting to minimise the second frame potential

$$\sum_{p,q\in\mathbb{Z}_d} |\langle D_p v, D_q v\rangle|^4.$$

Alternatively, by an algorithm of Appleby:[2]

**Algorithm 3.1.** *This is an iterative procedure. Let* $\Pi$ *be a fiducial projector which is an eigenprojector of a canonical order 3 unitary (e.g., the Zauner matrix). Let* $P$ *be the appropriate eigenspace projector of* $\Pi$ *so that* $P\Pi = \Pi P = \Pi$. *Set* $\Pi_0 = \Pi$. *Obtain* $\Pi_{n+1}$ *from* $\Pi_n$ *as follows.*

*Step 1: Calculate the frame expansion coefficients for* $\Pi_n$*:*

$$c_p = \frac{1}{d}\operatorname{Tr}(\Pi_n^2 P D_{-p}).$$

*Step 2: Calculate the frame expansion coefficients for* $\Pi_{n+1}$*:*

$$c'_p = \begin{cases} \frac{1}{d\sqrt{d+1}|c_p|}c_p & p \neq 0 \\ \frac{1}{d|c_p|}c_p & p = 0. \end{cases}$$

*Step 3: Construct* $\Pi_{n+1}$ *as*

$$\Pi_{n+1} = \sum_{p\in\mathbb{Z}_d^2} c'_p D_p.$$

A check can now be performed on the new numerical SIC-POVM fiducial by verifying against the equiangularity conditions, or its accuracy as an eigenprojector.

**Remark 3.3.1.**

Precision bumping using the Appleby algorithm can often be a time consuming process, for example to raise the precision of a SIC-POVM to more than 1000 digit, one may need to setup an overnight calculation on an average 2014 desktop computer.

[2]Private correspondence with Appleby. The algorithm has no proof of correctness and is based on the speculation that it will converge to a fixed point.

### 3.3.2 Applying the Galois action

Assuming Conjecture 3.2.17, the Galois group $\mathcal{G}$ of $\mathbb{E}_1$ over $\mathbb{Q}(a)$ is $\bar{C}_\Pi$. Note that $\mathcal{G}$ acts on $\mathbb{Z}_d \times \mathbb{Z}_d$ as the overlaps of a SIC-POVM are indexed by $p \in \mathbb{Z}_d \times \mathbb{Z}_d$. Now, by Theorem 3.2.18, construct the centraliser (the Galois group). Apply the action of $\mathcal{G}$ to $\mathbb{Z}_d \times \mathbb{Z}_d$ to partition $\mathbb{Z}_d \times \mathbb{Z}_d$ into various orbits.

**Remark 3.3.2.**

Note that in the doublet situation, one considers the Galois group of $\mathbb{E}_1$ over $\mathbb{E}_0$, so that Theorem 3.2.18 can be used to partition $\mathbb{Z}_d \times \mathbb{Z}_d$ into orbits. However, the base field of the overlaps polynomials $\rho$ are no longer conjectured to have base field $\mathbb{Q}(a)$ (cf. Conjecture 3.2.20).

For every orbit $B$, and $d \neq 0 \bmod 3$, construct the overlaps polynomials as

$$\rho_B := \prod_{p \in B} (x - \chi_p).$$

When $d \bmod 3 = 0$,

$$\rho_B := \prod_{p \in B} (x - \chi_p^3).$$

While not essential, the cubing of the overlaps eliminates the need to deal with the technicality of extra third roots of unity (Theorem 3.2.12). The original overlaps can be recovered once the $\rho$ polynomials are factorised by taking cube roots of the roots of $\rho$ and then scaling by the appropriate third root of unity.

#### 3.3.2.1 Parity and order 3 symmetries

With the goal of factorising the various $\rho$ in mind, it is beneficial to keep the degrees of $\rho$ low so the job of factorisation is more manageable. This can be accomplished by deleting the roots of the $\rho_B$ which are repeats, or are complex conjugates of other roots. The parity matrix

$$P := \begin{pmatrix} -1 & 0 \\ 0 & -1 \end{pmatrix}$$

corresponding to the complex conjugation and the Zauner matrix $F_z$ are elements of $\bar{C}_\Pi$ and generate a subgroup

$$\langle P, F_z \rangle \subset \bar{C}_\Pi$$

of order 6. For fiducials stabilised by the Zauner symmetry, this allows for the degrees of the $\rho$ polynomials to decrease by up to a factor of 6.

If dealing with a different symmetry SIC-POVM (e.g., $F_a$), then use that symmetry in lieu of $F_z$. The rest of the procedure will follow through.

### 3.3.3 Constructing the splitting fields for the overlaps polynomials

This step is the crux of the method, involving the most computational resources to accomplish. Firstly, the exact (as opposed to numerical) overlaps polynomials need to be known. Then the intermediate field extensions need to be recovered to build a tower of field extensions.

#### 3.3.3.1 Finding exact overlap polynomials

To recover the exact coefficients of each $\rho_B$ it is important to know whether the Clifford orbit of the particular SIC-POVM is stabilised by the Galois action or not. When the orbit is stabilised and is a singlet, Conjecture 3.2.20 suggests the coefficients of $\rho_B$ lie in $\mathbb{Q}(a)$. Since extension fields are vector spaces over their base field, each element of the field extension is a linear combination of a set of basis elements (generators) of the field extension. As $\mathbb{Q}(a)$ is a quadratic extension, the coefficients must be a linear combination of 1 and $a$. By applying an appropriate integer relation finding algorithm (e.g., PSLQ or LLL), the coefficients of $\rho_B$ can be expressed as a linear combination of 1 and $a$.

In practice this requires experimentally increasing the precision of the digits of the overlaps until software implementations of PSLQ or LLL (e.g., in Maple or Mathematica) are reliably identifying the coefficients.

When there are SIC-POVMs that are not singlets, conduct a survey of the overlap orbits for all identified SIC-POVMs (up to unitary equivalence) for the given dimension and attempt to merge the overlaps that are related by a Galois action. This is determined by seeing if $\mathbb{Q}(a)$ is the base field of

$$\prod_{\chi \in B_1 \cup B_2 \cup \ldots \cup B_k} (x - \chi),$$

where $B_1$, $B_2$, …, $B_k$ are orbits related by Galois action.

Since [SG10] found all Weyl-Heisenberg unitarily equivalent SIC-POVMs (with probability $1 - e^{-30}$) for all dimensions less than $48$ (with the exception of dimension 3), such a survey is possible by utilising their findings.

A significant drawback to this approach is it potentially increases the degree of the overlaps polynomial to solve, .e.g., if there are three related orbits of size 10 each, then the polynomial to factorise is of degree 30 rather than 10. A possible improvement to this technique is proposed in Section 3.4.

#### 3.3.3.2 Constructing the tower of field extensions

The intermediate field information found in tables from p. 32 onwards of [AYAZ13] and discussions with Appleby greatly improve chances of successfully computing a tower of extensions.

In MAGMA, $\mathbb{Q}$ would first be extended by adding $a$. Then the various square roots of the prime divisors of $a^2$ are added and finally the square roots of the prime divisors of $d$. Since each of these elements are square roots, the first few intermediate fields are just layers of quadratic extensions. Let $\mathbb{K}_1$ be the extension field containing all the mentioned quadratic extensions. The overlaps polynomials will factorise (not fully) over $\mathbb{K}_1$. These extensions are chosen because in terms of known solutions, this has been the case (see Table 5 of [AYAZ13]).

It is required for $e^{2\pi i/d}$ to be in the field as it is used in the Weyl-Heisenberg displacement operators. By making sure $\cos(\pi/d)$ and $\sin(\pi/d)$ are in the field, $e^{2\pi i/d}$ is forced to be in the field. To do this, find the minimal polynomial of $\cos(\pi/d)$ or $\sin(\pi/d)$ over $\mathbb{Q}$ (see [WZ93]). It is often the case that the degree of the minimal polynomials of the trigonometric functions differ, so the one with highest degree is used in order to increase the chance of factorising the overlaps polynomials. The minimal polynomials of the trigonometric functions will factorise over $\mathbb{K}_1$ so $\mathbb{K}_1$ is extended to the field $\mathbb{K}_2$ by the polynomial factor of the highest degree and containing either $\cos(\pi/d)$ or $\sin(\pi/d)$ as a root. Over $\mathbb{K}_2$, the overlaps polynomial should factorise even further.

**Remark 3.3.3.**

The quantities $\cos(\pi/d)$ and $\sin(\pi/d)$ are added to the field instead of $e^{\pi i/d}$ because in the known cases, $\mathbb{E}_1$ does not contain $i$ and so $i$ is added as an extension further up the tower.

To recover the last field extension, the overlaps polynomial is factorised over $\mathbb{K}_2$ and in practice this often leaves polynomial factors of degree $4$ or less, which can be used to construct the

remaining field extension. Once this is done the overlaps polynomial will fully split over the full extension field $\bar{\mathbb{E}}$. The final tower of extensions looks like:

$$\mathbb{Q} \subset \mathbb{Q}(a) \subset \ldots \subset \mathbb{K}_1 \subset \mathbb{K}_2 \subset \bar{\mathbb{E}}.$$

Sometimes, in particular when not dealing with singlets, the last extension field will be harder to find as factorising the overlaps polynomial over $\mathbb{K}_2$ may give factors of degree $5$ or more. In this case it is sometimes possible to find the missing polynomial of degree $4$ or less by using the same procedure to construct a tower of field extensions for a different overlap polynomial (when more than one overlaps orbit exists). This occured in dimension 15 when solving for the $a, b$ fiducials (using Scott-Grassl indexing).

### 3.3.4 Recovering roots

Once $\bar{\mathbb{E}}$ is constructed, it is possible to factorise the polynomial over $\bar{\mathbb{E}}$. The linear factors will all be represented in terms of the generators of the intermediate fields. Since all of the generators are known square roots, trigonometric functions or roots of polynomials of degree 4 or less, they will be explicitly known and so the overlaps will be in radical form (plus the trigonometric part).

When these overlaps are found (as roots of the overlaps polynomial), they can be matched with their numerical counterparts in order to find the right indexing for the overlaps. The overlaps can then be fed into the frame expansion formula 3.2.1.

## 3.4 Improvements

The practical bottleneck for this technique is the computer's ability to factorise polynomials in a reasonable amount of time. The computation time for computing solutions varies greatly depending on things like the degree of the polynomials and the complexity of the coefficients and factors (e.g., the singlet solution for dimension 17 took much less computation time than the doublets in dimension 15).

### 3.4.1 Lowering the degree of overlaps polynomials

One strategy for decreasing the degree of the overlaps polynomials $\rho$ is using subgroups $H$ of the Galois group $\mathcal{G}$ of $\bar{\mathbb{E}}$ over $\mathbb{Q}(a)$. For dimensions $d = 5 \bmod 6$, the Galois group of $\bar{\mathbb{E}}$ over $\mathbb{Q}(a)$ is conjectured to be cyclic (Conjecture 3.2.21), making the job of finding candidate subgroups much simpler. In general, this leads to the number of orbits increasing and the size of the orbits decreasing. This results in a lowering of the degrees of the $\rho$ involved, but comes at the cost of having more polynomials to solve, with potentially complicated coefficients. To further confound matters, Conjecture 3.2.20 only speculates that the overlaps polynomials have base field $\mathbb{Q}(a)$ under the action of $\mathcal{G}$ and not $H$. The new $\rho$ likely have base fields that are extensions of $\mathbb{Q}(a)$. It presents a real problem when trying to find the analytic form of the coefficients of the $\rho$'s.

This new problem can be worked around by constructing a further set of polynomials $\zeta$, which contain the coefficients of $\rho$ as roots. Let $B_1, B_2, \ldots, B_m$ be an ordered list of the overlap orbits under the action of $H$ and $\rho_{B_i}$ be the respective overlaps polynomial. Let $c_{ij}$ be the coefficient of the degree $j$ term of $\rho_{B_i}$. Define

$$\zeta_j := \prod_{1 \leq i \leq m} (x - c_{ij}).$$

Then each of the $\zeta_j$ contain, as roots, the coefficients of degree $j$ of each $\rho$ polynomial. The $\zeta_j$ all have base field $\mathbb{Q}(a)$ as they are stabilised by the action of the full Galois group $\mathcal{G}$. Applying the previous technique, the $\zeta_j$ can be factorised to find the coefficients of the $\rho$'s. The $\rho$ can be factorised by the same method.

This modified approach can also be used to tackle the non singlet cases (doublets, etc) where the base field is not obvious (i.e., not $\mathbb{Q}(a)$). By constructing the $\zeta$ to find the coefficients, increasing the degree of the $\rho$ polynomials is avoided (as well as the chaos of trying to find the right polynomials to multiply together).

Another approach to finding the coefficients involves using software packages to attempt to find minimal polynomials of the coefficients and using an integer relation algorithm like PSLQ or LLL to identify the analytic forms of the coefficients. This approach has yielded mixed results.

### 3.4.2 Simplifying the representation of solutions

Once various polynomials have been fully factorised, a secondary concern is whether it may be possible to simplify the presentation of the solution. One way to achieve this is to look for better field generators. Construct the Galois automorphisms of $\mathcal{G}$ and find the subgroups of $\mathcal{G}$ (which will correspond to the field extensions). Using MAGMA, apply the automorphism to a random element of an intermediate subfield to get a new element $x$. MAGMA is able to construct the minimal polynomial of $x$ and attempt to find a nicer minimal polynomial with smaller coefficients by calling the `OptimisedRepresentation` function (attempts to find a minimal polynomial that minimises the polynomial discriminant) [mag]. By first looking at low index subgroups of the Galois group, i.e., the groups corresponding to low degree subfields, simpler generators can be found first for those degree extensions. After rebuilding the tower of extensions, the remaining higher degree extensions often simplify to some extent. The process can be repeated on those higher degree extensions if necessary. In practice this method may not always be fruitful as computations may take too long.

## 3.5 Results

The recovered analytic SIC-POVM for dimension 17 (17c) is given in Appendix C.

Factorising overlaps polynomials continues to be the major bottleneck in this project. Unless a new approach is pioneered, or computer resources are significantly improved, it will remain a barrier for reverse engineering higher dimensional numerical SIC-POVMs.

# Chapter 4

# Projective unitary equivalences of frames

One natural question to consider when studying frames is when two frames are *alike*. A classic notion to describe likeness is the idea of unitary equivalence. Unitary transformations are desirable as they preserve the inner product relations and hence the geometry of the transformed vectors. The unitary operators form a group as well.

When studying projective objects like lines, one hopes to be able to leverage the machinery of vector spaces. This can be accomplished by considering the lines as vectors of a fixed length (customarily unit length) and consider vectors scaled by arbitrary constants (of unit modulus) as equivalent. The analogue is captured in the following definition.

**Definition 4.0.1.**
Let $\Psi := (f_j)_{j\in J}$ and $\Phi := (g_j)_{j\in J}$ be finite sequences of vectors in a Hilbert space $\mathcal{H}$. Then $\Psi$ and $\Phi$ are **projectively unitarily equivalent** if there exists a unitary operator $U \in \mathcal{U}(\mathcal{H})$ and a sequence $(c_j)_{j\in J}$ with $|c_j| = 1$ such that

$$c_j U(f_j) = g_j, \qquad \text{for all } j \in J.$$

$\Phi$ and $\Psi$ are unitarily equivalent if and only if (cf. Corollary 2.1.9)

$$\mathrm{Gram}(\Phi) = \mathrm{Gram}(\Psi). \tag{4.0.1}$$

$\Phi$ and $\Psi$ are projectively unitarily equivalent if and only if

$$\mathrm{Gram}(\Psi) = C\,\mathrm{Gram}(\Phi)C^*, \tag{4.0.2}$$

where $C$ is the diagonal matrix with complex diagonal entries $c_j$ of unit modulus.

Prima facie, solving (4.0.2) looks promising, but for complex inner product spaces, the only thing to work with is

$$\langle w_j, w_k\rangle = c_j\overline{c_k}\langle v_j, v_k\rangle, \qquad \forall j,k.$$

From a survey of the literature, it appears that projective unitary equivalence has only been previously calculated in a few situations using situation specific approaches.

These situations are: $\mathcal{H} = \mathbb{R}^d$ by considering all possible $c_j = \pm 1$ in (4.0.2) or by using two–graphs in the case of equiangular lines (Chapter 11 of [GR01]), for a specific construction of MUBs (using Weyl-Heisenberg or Clifford groups) via symplectic spreads, Hadamard matrices, and for the equal–norm tight frames of $n$ vectors in $\mathbb{C}^2$ robust to $n-2$ erasures (Section 5 of [CCKS97], Section 3 of [GR09], Section 3 of [Ben07], for SIC-POVMs Theorem 3 of [AFF11], and Section 2 of [HP04]).

Following in the footsteps of [MACR01], a general method is developed for determining whether **any** two finite sequence of vectors are projectively unitarily equivalent.

**Definition 4.0.2.**
The $m$**–vertex Bargmann invariants** or $m$**–products** of a sequence of $n$ vectors $\Phi = (v_j)_{j\in J}$ is

$$\Delta(v_{j_1}, v_{j_2}, \ldots, v_{j_m}) := \langle v_{j_1}, v_{j_2}\rangle\langle v_{j_2}, v_{j_3}\rangle\cdots\langle v_{j_m}, v_{j_1}\rangle, \quad 1 \le j_1, \ldots, j_m \le n. \tag{4.0.3}$$

In particular, (adopting the terminology of [AFF11]), the **triple products** (3**–products**) are

$$T_{jk\ell} := \Delta(v_j, v_k, v_\ell) = \langle v_j, v_k\rangle\langle v_k, v_\ell\rangle\langle v_\ell, v_j\rangle. \tag{4.0.4}$$

**Remark 4.0.3.**

The $m$–products are *projective unitary invariants*, e.g., for $m = 3$,

$$\begin{aligned}\Delta(c_jUv_j, c_kUv_k, c_\ell Uv_\ell) &= \langle c_jUv_j, c_kUv_k\rangle\langle c_kUv_k, c_\ell Uv_\ell\rangle\langle c_\ell Uv_\ell, c_jUv_j\rangle \\ &= c_j\overline{c_k}\langle Uv_j, Uv_k\rangle c_k\overline{c_\ell}\langle Uv_k, Uv_\ell\rangle c_\ell\overline{c_j}\langle Uv_\ell, Uv_j\rangle \\ &= \langle v_j, v_k\rangle\langle v_k, v_\ell\rangle\langle v_\ell, v_j\rangle \\ &= \Delta(v_j, v_k, v_\ell). \end{aligned} \tag{4.0.5}$$

**Remark 4.0.4.**

The $m$–products are closed under complex conjugation, i.e.,

$$\overline{\Delta(v_{j_1}, v_{j_2}, \ldots, v_{j_m})} = \Delta(v_{j_m}, \ldots, v_{j_2}, v_{j_1}). \tag{4.0.6}$$

The graph theory definitions and basic results in this Chapter can be found in a standard textbook on graph theory, e.g., Chapter 1 of [Die10]. The key result in this chapter is Theorem 4.2.2 which shows that a sequence of vectors $(v_j)$ is determined up to projective unitary equivalence by its $m$–products for all $m$. While these results apply to arbitrary finite sequences of vectors, it is sometimes convenient to refer to them as frames on the space that they span.

The proof of Theorem 4.2.2 identifies certain subsets of the $m$–products which classify the sequences. These depend on which of the $m$–products are nonzero, a fact conveniently encapsulated by the notion of a *frame graph*.

**Definition 4.0.5.**

The **frame graph** (cf. Section 3 of [AN13]) of a sequence of vectors $(v_j)_{j\in J}$ is the graph with vertices $\{v_j\}_{j\in J}$ (or the indices $j$ themselves) and

$$\text{an edge between } v_j \text{ and } v_k,\ j \neq k \iff \langle v_j, v_k\rangle \neq 0.$$

**Remark 4.0.6.**

Projectively unitarily equivalent frames have the same frame graph.

## 4.1 Complete frame graphs

A generic sequence of vectors is unlikely to have any pairs of orthogonal vectors most of the time. Hence, its corresponding frame graphs is often a complete graph. Appleby et al showed that $d^2$ equiangular vectors in $\mathbb{C}^d$ are characterised up to projective unitary equivalence by their triple products (3–products) in [AFF11]. This section extends their proof to cover frames with complete graphs, however these results do not extend to a general sequence of vectors.

**Definition 4.1.1.**

The **angles** of a sequence of vectors $\Phi = (v_j)_{j\in J}$ are the $\theta_{jk} \in \mathbb{R}/(2\pi\mathbb{Z})$ defined by

$$\langle v_j, v_k\rangle = |\langle v_j, v_k\rangle| e^{i\theta_{jk}}, \qquad \langle v_j, v_k\rangle \neq 0.$$

Since $\langle v_j, v_k\rangle = \overline{\langle v_k, v_j\rangle}$, these angles satisfy the relation

$$\theta_{jk} = -\theta_{kj}.$$

**Remark 4.1.2.**

A sequence of vectors might only have few angles, e.g., an orthogonal basis has no angles.

**Lemma 4.1.3.**

Let $\Phi = (v_j)_{j\in J}$ and $\Psi = (w_j)_{j\in J}$ be finite sequences of vectors in Hilbert spaces, with angles $\theta_{jk}$ and $\theta'_{jk}$. Then $\Phi$ and $\Psi$ are projectively unitarily equivalent if and only if

1. Their Gramians have entries with equal moduli, i.e.,

$$|\langle w_j, w_k\rangle| = |\langle v_j, v_k\rangle|, \qquad \forall j, k.$$

2. Their angles are "gauge equivalent", i.e., there exist $\phi_j \in \mathbb{R}/(2\pi\mathbb{Z})$ with

$$\theta'_{jk} = \theta_{jk} + \phi_j - \phi_k, \qquad \forall j, k.$$

*Proof.* Suppose $\Phi$ and $\Psi$ are projectively unitarily equivalent, i.e., $w_j = c_j U v_j$, where $U$ is unitary and $c_j = e^{i\phi_j}$. Then

$$\begin{aligned} e^{i\theta'_{jk}}|\langle w_j, w_k\rangle| = \langle w_j, w_k\rangle = \langle c_j U v_j, c_k U v_k\rangle = c_j\overline{c_k}\langle Uv_j, Uv_k\rangle = e^{i(\phi_j-\phi_k)}\langle v_j, v_k\rangle \\ = e^{i(\phi_j-\phi_k)}e^{i\theta_{jk}}|\langle v_j, v_k\rangle| \end{aligned}$$

By equating the moduli and then the arguments, 1 and 2 are obtained.

Conversely, suppose that 1 and 2 hold. Let $\tilde{v}_j := e^{i\phi_j}v_j$. Then

$$\begin{aligned} \langle \tilde{v}_j, \tilde{v}_k\rangle = \langle e^{i\phi_j}v_j, e^{i\phi_k}v_k\rangle = e^{i(\phi_j-\phi_k)}\langle v_j, v_k\rangle = e^{i(\phi_j-\phi_k)}e^{i\theta_{jk}}|\langle v_j, v_k\rangle| \\ = e^{i\theta'_{jk}}|\langle w_j, w_k\rangle| = \langle w_j, w_k\rangle. \end{aligned}$$

Therefore $(w_j)_{j\in J}$ is unitarily equivalent to $(\tilde{v}_j)_{j\in J}$, which is projectively unitarily equivalent to $(v_j)_{j\in J}$, so $\Psi$ and $\Phi$ are projectively unitarily equivalent. □

$|\langle v_j, v_k\rangle|$ is calculated from the triple products of (4.0.4), as

$$T_{jjj} = \langle v_j, v_j\rangle^3 = \|v_j\|^6, \qquad T_{jjk} = \langle v_j, v_j\rangle|\langle v_j, v_k\rangle|^2 = T_{jjj}^{\frac{1}{3}}|\langle v_j, v_k\rangle|^2. \tag{4.1.1}$$

**Theorem 4.1.4** (Characterisation)**.**
Let $\Phi = (v_j)_{j\in J}$ and $\Psi = (w_j)_{j\in J}$ be finite sequences of vectors in Hilbert spaces. Then

1. $\Phi$ and $\Psi$ are unitarily equivalent if and only if their Gramians are equal, i.e.,

$$\langle v_j, v_k\rangle = \langle w_j, w_k\rangle, \qquad \forall j, k.$$

2. If the frame graphs of $\Phi$ and $\Psi$ are complete, then they are projectively unitarily equivalent if and only if their triple products are equal, i.e.,

$$\langle v_j, v_k\rangle\langle v_k, v_\ell\rangle\langle v_\ell, v_j\rangle = \langle w_j, w_k\rangle\langle w_k, w_\ell\rangle\langle w_\ell, w_j\rangle, \qquad \forall j, k, \ell.$$

*Proof.* The condition for unitary equivalence is well known. It is included for comparison. For 2, suppose that $\Phi$ and $\Psi$ are projectively unitarily equivalent, i.e., $w_j = c_j U v_j$. By (4.0.5), their triple products are equal.

Conversely, suppose that $\Phi$ and $\Psi$ have the same triple products and their common frame graph is complete, i.e., all the triple products are nonzero.

It follows from (4.1.1) that their Gramians have entries with equal moduli, i.e.,

$$|\langle v_j, v_k\rangle| = |\langle w_j, w_k\rangle|, \qquad \forall j, k.$$

Let $\theta_{jk}$ and $\theta'_{jk}$ be the angles of $\Phi$ and $\Psi$. Then

$$T_{jk\ell} = \langle v_j, v_k\rangle\langle v_k, v_\ell\rangle\langle v_\ell, v_j\rangle = e^{i(\theta_{jk}+\theta_{k\ell}+\theta_{\ell j})}|\langle v_j, v_k\rangle\langle v_k, v_\ell\rangle\langle v_\ell, v_j\rangle|,$$

implies

$$\theta_{jk} + \theta_{k\ell} + \theta_{\ell j} = \theta'_{jk} + \theta'_{k\ell} + \theta'_{\ell j}.$$

Fix $\ell$ and let $\phi_j := \theta_{\ell j} - \theta'_{\ell j}$. Since $\theta_{k\ell} = -\theta_{\ell k}$ and $\theta'_{k\ell} = -\theta'_{\ell k}$,

$$\theta'_{jk} = \theta_{jk} + (\theta_{\ell j} - \theta'_{\ell j}) + (\theta_{k\ell} - \theta'_{k\ell}) = \theta_{jk} + (\theta_{\ell j} - \theta'_{\ell j}) - (\theta_{\ell k} - \theta'_{\ell k}) = \theta_{jk} + \phi_j - \phi_k,$$

i.e., the angles of $\Phi$ and $\Psi$ are gauge equivalent. The conditions of Lemma 4.1.3 hold, so it follows that $\Phi$ and $\Psi$ are projectively unitarily equivalent. □

The real case is closely connected with the theory of *two–graphs* (cf. Chapter 11 of [GR01]).

**Example 4.1.5.**

Recall from Example 2.5.1 that

$$v(t) := \frac{1}{\sqrt{2}}(0, 1, -e^{it})^T, \qquad t \in \mathbb{T},$$

gives a continuous family of Weyl-Heisenberg SIC-POVMs in 3 dimensions. By studying the triple products, it is apparent that the $v(t)$ give projectively unitarily *inequivalent* SIC-POVMs. The Weyl-Heisenberg SIC-POVMs in 3 dimensions were studied extensively in Section 8.5 of [Zhu12] and used to categorise the fiducials based on the angles (arguments of the complex numbers) of the triple products.

**Example 4.1.6** (Equiangular lines in $\mathbb{R}^d$)**.**

Let $\Phi = (v_j)_{j\in J}$ be a sequence of $n > d$ equiangular unit vectors (lines) in $\mathbb{R}^d$, i.e., there is an $\alpha > 0$ with

$$\langle v_j, v_k\rangle = \pm\alpha, \qquad j \neq k.$$

Then

$$G_\Phi = \mathrm{Gram}(\Phi) = I + \alpha S_\Phi,$$

where $S = S_\Phi$ is a **Seidel matrix**, i.e., $S$ is symmetric, with zero diagonal and off diagonal entries $\pm 1$. Each Seidel matrix is associated with a sequence of equiangular lines. $S$ is associated with the graph $\mathrm{gr}(S)$ which has an edge between $j \neq k$ if and only if $S_{jk} = -1$. Let $\mathcal{C}$ be the set of all diagonal matrices with diagonal entries $\pm 1$. Then the projective unitary equivalence class of $\Phi$ is determined by all the possible Gram matrices conjugated by members of $\mathcal{C}$, i.e.,

$$\mathcal{G} := \{CG_\Phi C^* : C \in \mathcal{C}\},$$

and hence all the possible Seidel matrices

$$\mathcal{S} := \{CS_\Phi C^* : C \in \mathcal{C}\},$$

and all the corresponding graphs $\mathrm{gr}(\mathcal{S})$.

The set of graphs $\mathrm{gr}(\mathcal{S})$ is called the **switching class** of $\mathrm{gr}(S_\Phi)$, or a **two–graph**.

Since the frame graph of $\Phi$ is complete, the projective unitary equivalence class of $\Phi$ (equivalently $\mathcal{G}$, $\mathcal{S}$ or $\mathrm{gr}(\mathcal{S})$) is characterised by the triple products of $\Phi$ (Theorem 4.1.4). It is only necessary to consider triple products with distinct indices. If an index is repeated twice or thrice, by (4.1.1), the triple products depends only on $\alpha$. These triple products are also independent of the ordering of the indices.

**Example 4.1.7** (Equiangular lines in $\mathbb{C}^d$)**.**
Let $\Phi$ be a sequence of $n$ equiangular unit vectors (lines) in $\mathbb{C}^d$, with $C > 0$. Then $\Phi$ is determined by its triple products (up to projective unitary equivalence). When $n = d^2$ (a SIC-POVM), this is the result in Theorem 3 and its remarks in [AFF11].

**Example 4.1.8** ($n$–cycle)**.**
Let $(e_j)_{j\in J}$ be the standard basis vectors in $\mathbb{C}^n$. Fix $|z| = 1$ and let

$$v_j := \begin{cases} e_j + e_{j+1}, & 1 \leq j < n, \\ e_n + ze_1, & j = n. \end{cases}$$

The frame graph of $(v_j)_{j\in J}$ is the $n$–cycle $(v_1, \ldots, v_n)$, so the only nonzero $m$–products for distinct vectors are

$$\Delta(v_j) = \|v_j\|^2 = 2, \qquad 1 \leq j \leq n, \tag{4.1.2}$$

$$\Delta(v_j, v_{j+1}) = |\langle v_j, v_{j+1}\rangle|^2 = 1, \quad 1 \leq j < n, \tag{4.1.3}$$

$$\Delta(v_1, v_2, \ldots, v_n) = z, \tag{4.1.4}$$

and their complex conjugates. Different choices of $z$ give projectively inequivalent frames. Thus, for $n > 3$, the vectors $(v_j)_{j\in J}$ are not defined up to projective unitary equivalence by their triple products.

## 4.2 Characterisation of projective unitary equivalence

A sequence of $n$ vectors $(v_j)_{j\in J}$ is determined up to projective unitary equivalence by its $m$–products for all $1 \leq m \leq n$. Verifying that two projectively unitarily equivalent frames induce the same $m$–products is a standard fare calculation. The crux is in proving the converse. The strategy will be to construct a sequence of vectors $(w_j)_{j\in J}$ with given $m$–products $\Delta(v_{j_1}, \ldots, v_{j_m})$. It amounts to reconstructing all Gramians for a given set of $m$–products.

**Example 4.2.1.**

A sequence of $n$ vectors $(v_j)_{j\in J}$ can be constructed in different ways for a given a Gram matrix $G$. If $V$ is a frame of $n$ vectors spanning a dimension $d$ space, then $G$ is positive semidefinite of rank $d$ and unitarily diagonalisable.

$$G = U^*\Lambda U, \qquad \Lambda = \begin{pmatrix} \gamma_1 & & \\ & \ddots & \\ & & \gamma_n \end{pmatrix}, \quad \gamma_1, \ldots, \gamma_d > 0, \quad \gamma_{d+1}, \ldots, \gamma_n = 0,$$

Taking $V = [v_1, \ldots, v_n] = \Lambda^{\frac{1}{2}}U$ gives vectors $(v_j)$ in $\mathbb{C}^n$ which are zero in the last $n - d$ components and can be identified with vectors in $\mathbb{C}^d$. Alternatively, take the Cholesky decomposition $G = V^*V$.

As the diagonal entries of the Gram matrices are given by the 1–products and the moduli of its off diagonal entries by the 2–products with distinct arguments, i.e.,

$$\Delta(v_j) = \|v_j\|^2, \qquad \Delta(v_j, v_k) = |\langle v_j, v_k\rangle|^2, \quad j \neq k, \tag{4.2.1}$$

it remains to choose arguments for the nonzero off diagonal entries of $G$, which are consistent with the $m$–products for $m \geq 3$. By choosing $C$ in (4.0.2), some of these can be taken to be arbitrary. When this is done fully, the remaining arguments will be determined by the $m$–products.

**Theorem 4.2.2** (Characterisation)**.**
Two sequences $(v_j)_{j\in J}$ and $(w_j)_{j\in J}$ of $n$ vectors are projectively unitarily equivalent if and only if their $m$–products are equal, i.e.,

$$\Delta(v_{j_1}, v_{j_2}, \ldots, v_{j_m}) = \Delta(w_{j_1}, w_{j_2}, \ldots, w_{j_m}), \qquad 1 \leq j_1, \ldots, j_m \leq n, \quad 1 \leq m \leq n.$$

*Proof.* It suffices to find a Gram matrix

$$G = [\langle w_k, w_j\rangle] = C\,\mathrm{Gram}(\Phi)C^*$$

by using only the $m$–products of $\Phi = (v_j)_{j\in J}$. The frame graph of $\Phi$ and the modulus of each entry of $G$ are known by (4.2.1). It remains to determine the arguments of the (nonzero) inner products. These correspond to edges of the frame graph. Without loss of generality, assume the frame graph of $\Phi$ is connected. Otherwise apply the method to each connected component of $\Gamma$.

Let $\mathcal{T}$ be a spanning tree of $\Gamma$ with root vertex $r$. While fanning out from the root $r$ (breadth-first search), multiply the vertices $v \in \Gamma \setminus \{r\}$ by unit scalars so that the arguments of the inner products corresponding to the edges of $\Gamma$ take arbitrarily assigned values. Continue in this fashion until you have reached every vertex.

The remaining undefined entries of the Gram matrix $G$ are given by the edges of $\Gamma$ which are not in $\mathcal{T}$. Since $\mathcal{T}$ is a spanning tree, adding each such edge to $\mathcal{T}$ gives an $m$–cycle. The corresponding nonzero $m$–product has all inner products already determined, except the one corresponding to the edge added. It is therefore uniquely determined by the $m$–product. □

**Corollary 4.2.3.**

A frame is projectively unitarily equivalent to a real frame if and only if its $m$–products are all real.

**Example 4.2.4.**

Let $\Phi = (e_j)_{j\in J}$ be an orthonormal basis for $\mathbb{C}^3$, which has Gram matrix

$$\mathrm{Gram}(\Phi) = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}.$$

The frame graph is totally disconnected (see Fig. 4.2.1), so each possible $G$ (as in the proof of Theorem 4.2.2) is determined by only 1–products and 2–products.

**Example 4.2.5.**

Let $\Phi = (v_j)_{j\in J}$ be three equiangular lines in $\mathbb{C}^2$. Then

$$\mathrm{Gram}(\Phi) = \begin{pmatrix} 1 & \frac{1}{2} & -\frac{1}{2} \\ \frac{1}{2} & 1 & \frac{1}{2} \\ -\frac{1}{2} & \frac{1}{2} & 1 \end{pmatrix}.$$

A spanning tree with root $v_1$ is given by the path $v_1, v_2, v_3$. Working out from the root $v_1 = w_1$, scale $v_2$ and $v_3$ to $w_j = c_j v_j$, so that the arguments of $\langle w_1, w_2\rangle$ and $\langle w_2, w_3\rangle$ are arbitrary, say $a$ and $b$ respectively with $|a| = |b| = 1$. The only inner product which is not yet determined is $\langle w_1, w_3\rangle$. This is obtained by completing the 3–cycle $v_1, v_2, v_3, v_1$, i.e.,

$$\begin{aligned} \Delta(w_1, w_2, w_3) = \Delta(v_1, v_2, v_3) \quad &\Longrightarrow \quad \left(\frac{1}{2}a\right)\left(\frac{1}{2}b\right)\left(\frac{1}{2}\langle w_3, w_1\rangle\right) = \left(\frac{1}{2}\right)^3 \\ &\Longrightarrow \quad \langle w_1, w_3\rangle = ab. \end{aligned}$$

The Gram matrices $G$ of vectors $(w_j)_{j\in J}$ which are projectively unitarily equivalent to $\Phi = (v_j)_{j\in J}$ are given by

$$G = \begin{pmatrix} 1 & \frac{1}{2}\overline{a} & \frac{1}{2}\overline{a}\overline{b} \\ \frac{1}{2}a & 1 & \frac{1}{2}\overline{b} \\ \frac{1}{2}ab & \frac{1}{2}b & 1 \end{pmatrix}, \qquad |a| = |b| = 1.$$

FIGURE 4.2.1: The frame graph of an orthonormal basis for $\mathbb{C}^3$ (Example 4.2.4) and the frame graph for three equiangular vectors in $\mathbb{C}^2$ (Example 4.2.5).

**Example 4.2.6.**

Let $\Phi = (v_j)_{j\in J}$ be the "two mutually unbiased bases" for $\mathbb{C}^2$ given by

$$\Phi = \left\{ \begin{pmatrix} 1 \\ 0 \end{pmatrix}, \begin{pmatrix} 0 \\ 1 \end{pmatrix}, \begin{pmatrix} \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} \end{pmatrix}, \begin{pmatrix} \frac{1}{\sqrt{2}} \\ -\frac{1}{\sqrt{2}} \end{pmatrix} \right\}, \qquad \mathrm{Gram}(\Phi) = \begin{pmatrix} 1 & 0 & \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ 0 & 1 & \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} & 1 & 0 \\ \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} & 0 & 1 \end{pmatrix}.$$

$\Phi$ has the 4–cycle $(v_1, v_3, v_2, v_4, v_1)$ frame graph. No nonzero triple products with distinct indices exist. A spanning tree with root $v_1$ is given by the path $v_1, v_3, v_2, v_4$. Fix $v_1 = w_1$ and scale in the order $v_3$, $v_2$, $v_4$ to $w_j = c_j v_j$, so that

$$\langle w_1, w_3\rangle = \frac{a}{\sqrt{2}}, \qquad \langle w_3, w_2\rangle = \frac{\overline{b}}{\sqrt{2}}, \qquad \langle w_2, w_4\rangle = \frac{c}{\sqrt{2}}.$$

Then $\langle w_4, w_1\rangle = \frac{1}{\sqrt{2}}\overline{z}$ is determined by completing the 4–cycle, i.e.,

$$\langle w_1, w_3\rangle\langle w_3, w_2\rangle\langle w_2, w_4\rangle\langle w_4, w_1\rangle = \langle v_1, v_3\rangle\langle v_3, v_2\rangle\langle v_2, v_4\rangle\langle v_4, v_1\rangle \quad \Longrightarrow \quad a\overline{b}c\overline{z} = -1$$

All the Gram matrices which match all the $m$–products of $\Phi$ have the form

$$G = \begin{pmatrix} 1 & 0 & \frac{\overline{a}}{\sqrt{2}} & \frac{\overline{z}}{\sqrt{2}} \\ 0 & 1 & \frac{\overline{b}}{\sqrt{2}} & \frac{\overline{c}}{\sqrt{2}} \\ \frac{a}{\sqrt{2}} & \frac{b}{\sqrt{2}} & 1 & 0 \\ \frac{z}{\sqrt{2}} & \frac{c}{\sqrt{2}} & 0 & 1 \end{pmatrix}, \qquad |a| = |b| = |c| = 1, \quad z := -\frac{ac}{b}.$$

$\Phi$ is determined up to projective unitary equivalence by its 1–products and 2–products, since Sylvester's criterion for $G$ to be positive semidefinite gives

$$\det(G) = -\frac{1}{4}\frac{(bz+ac)^2}{abcz} = -\frac{1}{4}\left|\frac{bz}{ac}+1\right|^2 \geq 0 \quad \Longrightarrow \quad \frac{bz}{ac}+1=0 \quad \Longrightarrow \quad z=-\frac{ac}{b}.$$

Contrast this to Example 4.1.8, which is not determined up to projective unitary equivalence by its 1–products and 2–products.

## 4.3 Reconstruction from the $m$–products

Theorem 4.2.2 shows that only a small subset of the $m$–products is required to determine a sequence of vectors up to projective unitary equivalence.

**Definition 4.3.1.**

A **determining set** is a subset of the $m$–products (or the corresponding indices) that can determine all $m$–products.

**Corollary 4.3.2.**

Let $\Gamma$ be the frame graph of a sequence of $n$ vectors $\Phi$. For each connected component $\Gamma_j$ of $\Gamma$, let $\mathcal{T}_j$ be a spanning tree. Then $\Phi$ is determined up to projective unitary equivalence by the following $m$–products

(i) The 2–products

(ii) The $m$–products, $3 \leq m \leq n$, used to obtain $\Gamma_j$ from $\mathcal{T}_j$ by completing $m$–cycles (these have indices in $\Gamma_j$), as detailed in the proof of Theorem 4.2.2.

**Remark 4.3.3.**

If $M$ is the number of edges of $\Gamma$ which are not in any $\mathcal{T}_j$, then it is sufficient to know all of the 2–products and $M$ of the $m$–products, $3 \leq m \leq n$.

**Remark 4.3.4.**

The $m$–products from (i) and (ii) are a determining set.

Although Example 4.2.6 shows the $m$–products of (ii) may not be a minimal set of $m$-products needed to characterise a sequence of vectors, it is a minimal subset from which the $m$–products can be determined using only the proof of Theorem 4.2.2.

**Example 4.3.5.**

Let $\Phi = (v_j)_{j\in J}$ be four equiangular vectors with $C > 0$. The frame graph of $\Phi$ is complete and $M = 6 - 3 = 3$. Spanning trees (see Fig. 4.3.1) include

$$\mathcal{T}_p := \text{the path } v_1, v_2, v_3, v_4,$$
$$\mathcal{T}_s := \text{the star graph internal vertex } v_1 \text{ and leaves } v_2, v_3, v_4.$$

For $\mathcal{T}_p$, complete either the 4–cycle $v_1, v_2, v_3, v_4, v_1$ followed by any one of the four 3–cycles or two of the 3–cycles, in order to determine the rest of the $m$–products. For example, a determining set is given by the 2–products and

$$\Delta(v_1, v_2, v_3, v_4), \quad \Delta(v_1, v_2, v_3), \quad \Delta(v_1, v_2, v_4). \tag{4.3.1}$$

For $\mathcal{T}_s$, adding any edge completes a 3–cycle containing $v_1$. Then there are two edges which can be added to complete a 3–cycle. Adding another edge afterwards completes the remaining two 3–cycles. For example, a determining set is given by the 2–products and

$$\Delta(v_1, v_2, v_3), \quad \Delta(v_1, v_2, v_4), \quad \Delta(v_1, v_3, v_4). \tag{4.3.2}$$

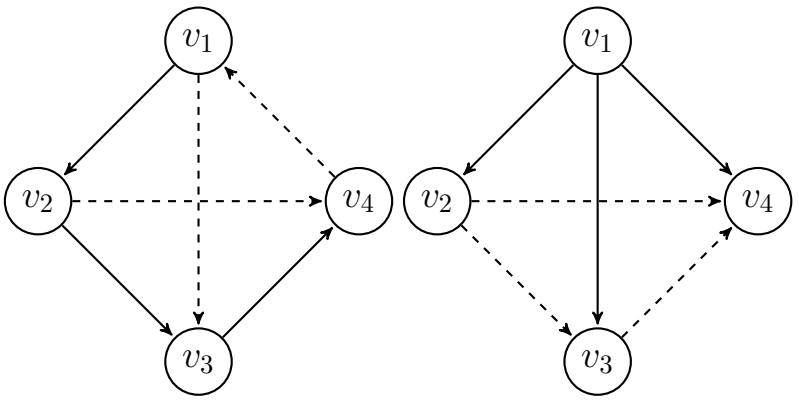



FIGURE 4.3.1: The spanning trees $\mathcal{T}_p$ and $\mathcal{T}_s$ (and cycle completions) of Example 4.3.5.

Note that the 4–product of (4.3.1) is decomposable into smaller $m$–products, e.g.,

$$\Delta(v_1, v_2, v_3, v_4) = \frac{\Delta(v_1, v_2, v_3)\Delta(v_1, v_3, v_4)}{\Delta(v_1, v_3)}, \tag{4.3.3}$$

so $\Delta(v_1, v_2, v_3, v_4)$ can be replaced by the 3–cycle $\Delta(v_1, v_3, v_4)$, which gives the determining set of (4.3.2).

The $m$–products of (ii) can be those corresponding to the **fundamental cycles**, i.e., the unique cycle completed by adding an edge to $\mathcal{T}_j$ from $\Gamma_j \setminus \mathcal{T}_j$. These fundamental cycles form a basis for the **cycle space**, i.e., the $\mathbb{Z}_2$–formal linear combinations of cycles.

Leveraging this linear algebra framework, Theorem 4.2.2 is generalised as follows.

**Theorem 4.3.6** (Characterisation II)**.**
Two finite sequences of vectors $(v_j)_{j\in J}$ and $(w_j)_{j\in J}$ are projectively unitarily equivalent if and only if their 2–products are equal and the $m$–products corresponding to the $m$–cycles that form a basis for the cycle space of the frame graph are equal.

*Proof.* All cycles in the frame graph can be calculated from those in a basis and hence all $m$–products can be calculated from those corresponding to a basis, as indicated in (4.3.3). □

**Example 4.3.7.**
The fundamental cycles corresponding to the trees $\mathcal{T}_p$ and $\mathcal{T}_s$ of Example 4.3.5 give the following $m$–products

$$\Delta(v_1, v_2, v_3, v_4), \quad \Delta(v_1, v_2, v_3), \quad \Delta(v_2, v_3, v_4), \tag{4.3.4}$$

$$\Delta(v_1, v_2, v_3), \quad \Delta(v_1, v_2, v_4), \quad \Delta(v_1, v_3, v_4), \tag{4.3.5}$$

respectively.

Theorem 4.1.4 is restated as follows.

**Theorem 4.3.8.**
Let $\Phi = (v_j)_{j\in J}$ be a finite sequence of vectors. Then $\Phi$ is determined up to projective unitary equivalence by its 3–products if the cycle space of its frame graph is spanned by 3–cycles (there exists a basis of 3–cycles).

**Example 4.3.9.**
(Chordal graphs) A graph is **chordal** (or **triangulated**) if each of its cycles of four or more vertices has a chord. The cycle space is spanned by the 3–cycles. If the frame graph of $\Phi$ is chordal, (e.g., equiangular lines), then $\Phi$ is determined by its triple products.

The extreme cases are a totally disconnected graph (orthogonal bases) where there are no cycles and the complete graph where all subsets of three vectors lie on a 3-cycle.

(Theorem 4.3.11) provides an example of a non chordal frame graph where the cycle space admits a basis of 3–cycles.

**Definition 4.3.10.**

A family of orthonormal bases $\mathcal{B}_1, \mathcal{B}_2, \ldots, \mathcal{B}_k$ for $\mathbb{C}^d$ is **mutually unbiased** if

$$\text{for all } 1 \leq r, j \leq k, r \neq j, \qquad |\langle v, w\rangle|^2 = \frac{1}{d}, \qquad \forall v \in B_r, \quad \forall w \in B_j,$$

and $\mathcal{B}_1, \ldots, \mathcal{B}_k$ is a sequence of $k$ **MUBs** (**mutually unbiased bases**).

The frame graph of two or more MUBs ($d > 1$) is not chordal, because there is a 4–cycle $(v_1, w_1, v_2, w_2)$, $v_1, v_2 \in \mathcal{B}_r$, $w_1, w_2 \in \mathcal{B}_s$ not containing a chord. For three or more MUBs the cycle space of the frame graph is spanned by the 3–cycles. This is not case for two MUBs (cf. Example 4.2.6).

**Theorem 4.3.11** (MUBs)**.**

Let $\Phi$ consist of three or more MUBs in $\mathbb{C}^d$. Then $\Phi$ is determined up to projective unitary equivalence by its 3–products.

*Proof.* It suffices to show the cycle space of the frame graph $\Gamma$ of $\Phi$ has a basis of 3–cycles.

Let $\mathcal{B}_j$, $j = 1, \ldots, k$, be the MUBs for $\mathbb{C}^d$, so that $\Gamma$ is a complete $k$–partite graph (with partite sets $\mathcal{B}_j$). Fix $v_1 \in \mathcal{B}_1$ and $v_2 \in \mathcal{B}_2$. A spanning tree $\mathcal{T}$ for $\Gamma$ is given by taking an edge from $v_1$ to each vertex of $\mathcal{B}_j$, $j \neq 1$ and an edge from $v_2$ to each vertex of $\mathcal{B}_1 \setminus v_1$. Each of the remaining edges of $\Gamma \setminus \mathcal{T}$ gives a fundamental cycle. These have two types (see Fig. 4.3.2):

1. $\frac{1}{2}d^2(k-1)(k-2)$ edges between vertices in $\mathcal{B}_r$ and $\mathcal{B}_s$, $r, s \neq 1$, which give fundamental 3–cycles involving $v_1$.

2. $(d-1)((k-1)d-1)$ edges between vertices $u \in \mathcal{B}_1 \setminus v_1$ and $w \in \cup_{j\neq 1}\mathcal{B}_j \setminus v_2$, which give fundamental 4–cycles $(u, w, v_1, v_2)$. These can be written as a union of the 3–cycles $(u, w, v_2)$ and $(v_1, v_2, w)$.

Hence the cycle space is spanned by 3–cycles. □

**Remark 4.3.12.**

The maximal number of MUBs is of interest in quantum information theory. For $d$ a prime, or

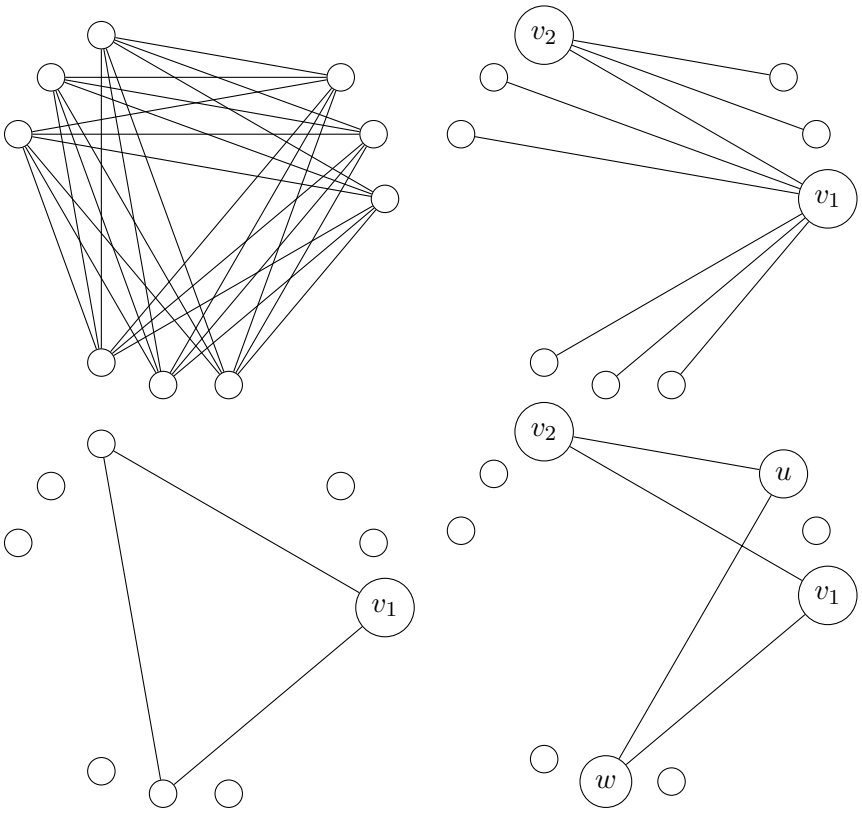



FIGURE 4.3.2: The proof of Theorem 4.3.11 for MUBs $\mathcal{B}_1, \mathcal{B}_2, \mathcal{B}_3$ in $\mathbb{C}^3$. The frame graph $\Gamma$, the spanning tree $\mathcal{T}$ and fundamental cycles of type 1 and 2.

a power of a prime, the maximal number of MUBs in $\mathbb{C}^d$ is $d+1$, see Section 7 of [Ben07], Section 3 [Iva81], p. 369 onwards of [WF89] for constructions. These constructions all have a special (Weyl-Heisenberg) structure, which has been used to classify them up to projective unitary equivalence, see Theorem 2.3 of [Kan12], Section 3 of [GR09], Section 3 of [Ben07].

In light of Theorem 4.3.11, 3–products present an alternative way to classify MUBs up to projective unitary equivalence by using only properties inherent to the vectors themselves (and hence also independent of their construction method).

**Remark 4.3.13.**

There exists graphs which are not chordal, with every edge on a 3–cycle (as is the case for the frame graph of three or more MUBs), but for which the cycle space is not spanned by 3–cycles (see Fig. 4.3.3).

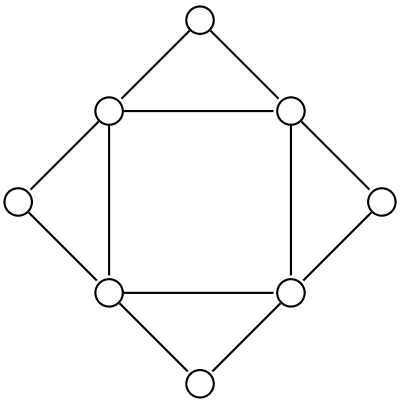


FIGURE 4.3.3: A nonchordal graph for which each edge is on a 3–cycle.

## 4.4 Similarity and $m$–products for vector spaces

Considering the more general setting of frames in vector spaces, the role of unitary equivalence can be played by "similarity" [Wal11] and the role of $m$–products by "canonical $m$–products". So the "projective symmetry group" can be recast in a general setting.

**Definition 4.4.1.**

Let $\Phi = (v_j)_{j\in J}$ and $\Psi = (w_j)_{j\in J}$ be finite sequences of vectors spanning vector spaces $X$ and $Y$ over a subfield $\mathbb{F}$ of $\mathbb{C}$. Then $\Phi$ and $\Psi$ are **similar** if there is an invertible linear map $Q : X \to Y$ with

$$w_j = Qv_j, \qquad \forall j,$$

and **projectively similar** if there is an invertible linear map $Q : X \to Y$ and unit scalars $c_j$ with

$$w_j = c_j Qv_j, \qquad \forall j.$$

**Definition 4.4.2.**

Let $V$ be the synthesis map of $\Phi$ (see Definition 2.1.5). The subspace of all linear dependencies between the vectors of $\Phi$ is

$$\operatorname{dep}(\Phi) := \ker(V) = \{a \in \mathbb{F}^J : \textstyle\sum_j a_j v_j = 0\},$$

and denote the orthogonal projection onto $\operatorname{dep}(\Phi)^\perp$ (orthogonal complement) by $P_\Phi$.

The following characterisation of similarity is cast in terms of linear dependencies.

**Lemma 4.4.3** (Lemma 3.3, [Wal11])**.**
Let $\Phi = (v_j)_{j\in J}$ and $\Psi = (w_j)_{j\in J}$ be spanning sequences for the $\mathbb{F}$–vector spaces $X$ and $Y$. The following are equivalent,

(a) $\Phi$ and $\Psi$ are similar, i.e., there is a invertible linear map $Q : v_j \mapsto w_j$.

(b) $\mathrm{dep}(\Phi) = \mathrm{dep}(\Psi)$ (the dependencies are equal).

(c) $P_\Phi = P_\Psi$ (the associated projections are equal).

$\Phi = (v_j)$ is similar to the columns of $P = P_\Phi$ (proof of Lemma 4.4.3). These columns $(Pe_j)$ span a subspace of $\mathbb{F}^J$ inheriting the Euclidean inner product. Indeed

$$\langle Pe_j, Pe_k\rangle = P_{jk},$$

i.e., the Gramian of $(Pe_j)$ is $P = P_\Phi$.

**Definition 4.4.4.**
The $m$–products of $(Pe_j)_{j\in J}$ are the **canonical $m$–products** of $(v_j)_{j\in J}$, given by

$$\Delta_C(v_{j_1}, \ldots, v_{j_m}) := \Delta(Pe_{j_1}, \ldots, Pe_{j_m}) = P_{j_1j_2}P_{j_2j_3}\cdots P_{j_mj_1}. \tag{4.4.1}$$

**Theorem 4.4.5** (Characterisation)**.**
Let $\Phi = (v_j)_{j\in J}$ and $\Psi = (w_j)_{j\in J}$ be finite sequences of vectors in vector spaces over a subfield $\mathbb{F}$ (of $\mathbb{C}$) closed under complex conjugation. Then

1. $\Phi$ and $\Psi$ are similar if and only if $P_\Phi = P_\Psi$.
2. $\Phi$ and $\Psi$ are projectively similar if and only if their canonical $m$–products (for a determining set) are equal.

*Proof.* The first follows from Lemma 4.4.3 and implies that $\Phi$ and $\Psi$ are projectively similar, i.e.,

$$w_j = c_jQv_j = Q(c_jv_j), \qquad \forall j,$$

if and only if $\Psi = (w_j)_{j\in J}$ and $\Phi' = (c_jv_j)_{j\in J}$ are similar, for some choice of unit scalars $(c_j)$, i.e.,

$$P_\Psi = P_{(c_jv_j)} = C^*P_\Phi C. \tag{4.4.2}$$

Since $\Phi$ and $\Psi$ are similar to $(P_\Phi e_j)_{j\in J}$ and $(P_\Psi e_j)_{j\in J}$, (with Gram matrices $P_\Phi$ and $P_\Psi$ respectively), it follows from (4.0.2) that (4.4.2) is equivalent to $(P_\Phi e_j)_{j\in J}$ and $(P_\Psi e_j)_{j\in J}$ being projectively unitary equivalent. By Theorem 4.1.4, this is equivalent to their $m$–products, (i.e., the canonical $m$–products of $(v_j)_{j\in J}$ and $(w_j)_{j\in J}$) being equal. □

For projective similarity, $c_j$ and $Q$ in $w_j = c_j Q v_j$ can be calculated explicitly.

**Example 4.4.6.**

Let $\Phi = (v_j)_{j\in J}$ span a 2–dimensional space, i.e.,

$$\alpha v_1 + \beta v_2 + \gamma v_3 = 0, \qquad |\alpha|^2 + |\beta|^2 + |\gamma|^2 = 1.$$

Then $\mathrm{dep}(\Phi) = \mathrm{span}\{u\}$, $u = (\alpha, \beta, \gamma)$, so that

$$P_\Phi = I - uu^* = \begin{pmatrix} 1-|\alpha|^2 & -\alpha\overline{\beta} & -\alpha\overline{\gamma} \\ -\overline{\alpha}\beta & 1-|\beta|^2 & -\beta\overline{\gamma} \\ -\overline{\alpha}\gamma & -\overline{\beta}\gamma & 1-|\gamma|^2 \end{pmatrix}.$$

The canonical 2–products are uniquely determined by $|a|, |b|, |c|$, e.g.,

$$\Delta_C(v_1, v_1) = (1-|\alpha|^2)^2, \qquad \Delta_C(v_1, v_2) = |-\overline{\alpha}\beta|^2 = |\alpha|^2|\beta|^2,$$

Likewise the canonical 3–product corresponding to the unique 3–cycle

$$\Delta_C(v_1, v_2, v_3) = (-\overline{\alpha}\beta)(-\overline{\beta}\gamma)(-\alpha\overline{\gamma}) = -|\alpha|^2|\beta|^2|\gamma|^2.$$

If $\Psi = (w_j)_{j\in J}$ is given by

$$\tilde{\alpha} w_1 + \tilde{\beta} w_2 + \tilde{\gamma} w_3 = 0, \qquad |\tilde{\alpha}|^2 + |\tilde{\beta}|^2 + |\tilde{\gamma}|^2 = 1,$$

then

1. $\Psi$ is similar to $\Phi$ if and only if $P_\Psi = P_\Phi$, i.e.,

$$\tilde{\alpha}\overline{\tilde{\beta}} = \alpha\overline{\beta}, \qquad \tilde{\alpha}\overline{\tilde{\gamma}} = \alpha\overline{\gamma}, \qquad \tilde{\beta}\overline{\tilde{\gamma}} = \beta\overline{\gamma}.$$

2. $\Psi$ is projectively similar to $\Phi$ if and only if their canonical $m$–products are equal, i.e.,

$$|\tilde{\alpha}| = |\alpha|, \qquad |\tilde{\beta}| = |\beta|, \qquad |\tilde{\gamma}| = |\gamma|.$$

When $\Psi$ and $\Phi$ are projectively similar, i.e., $w_j = c_j Q v_j$ (the canonical $m$–products are equal), one has $P_\Psi = C^* P_\Phi C$.

Suppose $\alpha, \beta, \gamma \neq 0$, then the allowable scalings take the form:

$$\overline{c_1} c_2 \alpha \overline{\beta} = \tilde{\alpha}\overline{\tilde{\beta}}, \quad \overline{c_1} c_3 \alpha \overline{\gamma} = \tilde{\alpha}\overline{\tilde{\gamma}}, \quad \overline{c_2} c_3 \beta \overline{\gamma} = \tilde{\beta}\overline{\tilde{\gamma}} \quad \Longrightarrow \quad c_2 = \frac{\tilde{\alpha}}{\alpha}\frac{\beta}{\tilde{\beta}} c_1, \quad c_3 = \frac{\tilde{\alpha}}{\alpha}\frac{\gamma}{\tilde{\gamma}} c_1.$$

where $Q$ is defined by

$$Q v_1 := \overline{c_1} w_1, \qquad Q v_2 := \overline{c_2} w_2 = \frac{\alpha}{\tilde{\alpha}}\frac{\tilde{\beta}}{\beta}\overline{c_1} w_2.$$

## 4.5 Projectively equivalent harmonic frames

The following definitions relating to harmonic frames can be found in [CW11].

**Definition 4.5.1.**

Let $G$ be a finite abelian group of order $n$, with irreducible characters $\xi \in \hat{G}$. $\hat{G}$ is the character group and it is isomorphic to $G$. Let $J \subset G$, with $|J| = d$. The equal–norm tight frame for $\mathbb{C}^J \approx \mathbb{C}^d$ given by

$$\Phi_J = (\xi|_J)_{\xi \in \hat{G}}$$

is a **harmonic frame** and **cyclic** if $G$ is a cyclic group. Any frame that is unitarily equivalent to a harmonic frame is also called a harmonic frame.

Harmonic frames are tight frames that are the orbit of a group of unitary transformations on $\mathbb{C}^d$, which is isomorphic to $G$ (see Section 5 of [VW05], [CW11]). The harmonic frames were studied up to unitary equivalence in [WH06], [CW11].

**Definition 4.5.2.**

Let $G$ be a fixed finite abelian group. Subsets $J$ and $K$ of $G$ are **multiplicatively equivalent** if there is an automorphism $\sigma : G \to G$ for which $K = \sigma J$. In this case,

$$\hat{\sigma} : \hat{G} \to \hat{G} : \chi \mapsto \chi \circ \sigma^{-1}$$

is an automorphism of $\hat{G}$ and

$$\langle \xi|_J, \eta|_J \rangle = \langle \hat{\sigma}\xi|_K, \hat{\sigma}\eta|_K \rangle,$$

i.e., $\Phi_J$ and $\Phi_K$ are unitarily equivalent after reindexing by the automorphism $\hat{\sigma}$.

**Definition 4.5.3.**

The **translations** of $G$ are the bijections

$$\tau_b : G \to G : j \mapsto j - b, \qquad b \in G,$$

and $K$ is a **translate** of $J$ if $K = J - b$, i.e., $K = \tau_b J$.

**Definition 4.5.4.**

The **affine group** of $G$ is the group of bijections $\pi : G \to G$ generated by the translations and automorphisms, i.e., the $|G|\,|\mathrm{Aut}(G)|$ maps of the form

$$\pi(g) = \sigma(g) - b, \qquad \sigma \in \mathrm{Aut}(G), \quad b \in G.$$

If $K = \pi J$, for some $\pi$ in the affine group, then $J$ and $K$ are **affinely equivalent**.

**Lemma 4.5.5.**

If $J$ and $K$ are subsets of a finite abelian group $G$, which are translates of each other, then the harmonic frames $\Phi_J$ and $\Phi_K$ are projectively unitarily equivalent.

*Proof.* Suppose $K = J - b$. Since $\Phi_J = (\xi|_J)_{\xi \in \hat{G}}$, it remains to prove

$$\xi|_K = c_\xi U(\xi|_J), \qquad \xi \in \hat{G},$$

where $U : \mathbb{C}^J \to \mathbb{C}^K$ is unitary. Let $U_b : \mathbb{C}^J \to \mathbb{C}^K$ be the unitary map

$$(U_b v)(k) := v(k + b), \qquad k \in K.$$

Since $\xi$ is a character,

$$(U_b \xi|_J)(k) = \xi|_J(k + b) = \xi(k + b) = \xi(k)\xi(b) = \xi|_K(k)\xi(b),$$

the map $U = U_b$ and $c_\xi = 1/\xi(b)$ will suffice. □

Numerical evidence suggests the converse is true, i.e., that projective unitary equivalence implies $J$ and $K$ are translates of each other.

**Theorem 4.5.6.**

Let $J$ and $K$ be subsets of a finite abelian group $G$.

1. If $J$ and $K$ are translates, then $\Phi_J$ and $\Phi_K$ are projectively unitarily equivalent.
2. If $J$ and $K$ are multiplicatively equivalent, then $\Phi_J$ and $\Phi_K$ are unitarily equivalent after reindexing by an automorphism of $G$.
3. If $J$ and $K$ are affinely equivalent, then $\Phi_J$ and $\Phi_K$ are projectively unitarily equivalent after reindexing by an automorphism.

*Proof.* The first part is Lemma 4.5.5, the second is given in Theorem 3.5 of [CW11] and third follows by combining the first two. □

**Proposition 4.5.7.**

Let $p > 2$ be a prime. All harmonic frames of $p$ vectors in $\mathbb{C}^2$ are projectively unitarily equivalent up to reindexing (to $p$ equally spaced vectors in $\mathbb{R}^d$). There exists a unique affine map that takes a sequence of two distinct elements of $\mathbb{Z}_p$ to any other.

*Proof.* Let $(v_j)_{j\in J}$ be a cyclic harmonic frame in $\mathbb{C}^2$ of $p$ vectors, where $J = \{0, 1, 2, \ldots, p-1\}$. Then $v_k = \frac{1}{\sqrt{p}}(\omega^{sk}, \omega^{tk})^T$ for some $s \neq t$. Consider the projectively unitarily equivalent frame given by

$$v'_k := \frac{1}{\omega^{sk}} v_k = (1, \omega^{(t-s)k})^T.$$

Since $p$ is prime, $\omega^{(t-s)k}$, $k \in \{0, 1, \ldots, p-1\}$ are all the roots of unity. So $(v'_k)_{j\in J}$ is a reindexing of $(1, \omega^k)^T$, $k \in \{0, 1, \ldots, p-1\}$. Therefore all cyclic harmonic frames in $\mathbb{C}^2$ of $p$ vectors are projectively equivalent to each other. □

**Example 4.5.8.**

After allowing for reindexing, the two harmonic frames of three vectors in $\mathbb{C}^2$ which are unitarily inequivalent (one is real, one is complex) are projectively unitarily equivalent.

Using Theorem 4.5.6, the various equivalences between cyclic harmonic frames were calculated using the computer algebra package MAGMA [BCP97]. The results are summarised in Fig. 4.1.

The number of projective unitary equivalence classes appears to be much smaller than the number of unitary equivalence classes (up to any reindexing). Few cases arise where the number of unitarily equivalent classes is smaller than the group theoretic predictions. These occur when there exists an equivalence inducing reindexing that is *not* an automorphism of the group, as previously observed in Theorem 3.5 of [CW11]. In the projective setting, the number of classes closely follows the number of equivalence classes generated by applying the affine group. The $n = 8, d = 4$ case was the only computed exception.

$d = 2$

| n | uni | proj |
|---|---|---|
| 2 | 1 | 1 |
| 3 | 2 | 1 |
| 4 | 3 | 2 |
| 5 | 3 | 1 |
| 6 | 6 | 3 |
| 7 | 4 | 1 |
| 8 | 7 | 3 |
| 9 | 6 | 2 |
| 10 | 9 | 3 |
| 11 | 6 | 1 |
| 12 | 13 | 5 |
| 13 | 7 | 1 |
| 14 | 12 | 3 |
| 15 | 13 | 3 |

$d = 3$

| n | uni | proj |
|---|---|---|
| 3 | 1 | 1 |
| 4 | 3 | 1 |
| 5 | 3 | 1 |
| 6 | 11 | 3 |
| 7 | 7 | 2 |
| 8 | 16<br>17 | 4 |
| 9 | 15 | 3 |
| 10 | 29 | 4 |
| 11 | 17 | 2 |
| 12 | 56<br>57 | 9 |
| 13 | 25 | 3 |

$d = 4$

| n | uni | proj |
|---|---|---|
| 5 | 2 | 1 |
| 6 | 9 | 3 |
| 7 | 7 | 2 |
| 8 | 21<br>23 | 6<br>5 |
| 9 | 23<br>24 | 4 |
| 10 | 53<br>54 | 9 |
| 11 | 34 | 4 |
| 12 | 138<br>141 | 21 |

$d = 5$

| n | uni | proj |
|---|---|---|
| 5 | 1 | 1 |
| 6 | 4 | 1 |
| 7 | 4 | 1 |
| 8 | 19<br>20 | 4 |
| 9 | 23<br>24 | 4 |
| 10 | 67 | 9 |
| 11 | 48 | 6 |

$d = 6$

| n | uni | proj |
|---|---|---|
| 6 | 1 | 1 |
| 7 | 2 | 1 |
| 8 | 11 | 3 |
| 9 | 16 | 3 |
| 10 | 55<br>56 | 9 |
| 11 | 48 | 6 |

$d = 7$

| n | uni | proj |
|---|---|---|
| 7 | 1 | 1 |
| 8 | 4 | 1 |
| 9 | 8 | 2 |
| 10 | 32 | 4 |
| 11 | 34 | 4 |
| 12 | 228<br>234 | 25 |

TABLE 4.1: The number of unitary and projective unitary equivalence classes (up to reindexing) of cyclic harmonic frames of $n$ vectors in $\mathbb{C}^d$, $d = 2, \ldots, 7$. When the group theoretic estimate of Theorem 4.5.6 is larger (because there are reindexings which are not automorphisms) it is given in the row below.

# Chapter 5

# Projective symmetry groups

## 5.1 Introduction

**Definition 5.1.1.**

Let $\Phi = (v_j)_{j\in J}$ be a finite sequence of vectors. A **projective symmetry** of $\Phi$ is an invertible linear map $L$ such that, for all $j$,

$$Lv_j = c_j v_{\sigma j}, \qquad |c_j| = 1, \quad \exists\, \sigma \in S_J. \tag{5.1.1}$$

The example $\Phi = (e_1, -e_1, e_2, -e_2)$ shows that the choice for $L$ and $(c_j)_{j\in J}$ is far from unique and $\Phi = (e_1, e_1, e_2, e_2)$ shows choosing $L$ and $(c_j)_{j\in J}$ does not provide sufficient information to determine a unique permutation $\sigma$. However, fixing $\sigma$ will determine $L$ and $(c_j)_{j\in J}$ (if they exist). Thus it makes sense to define the projective symmetry group as a group of permutations on the index set $J$ (or the vectors themselves).

**Definition 5.1.2.**

Let $\Phi = (v_j)_{j\in J}$ be a finite sequence of vectors with span $X$. Then

1. The **projective symmetry group** of $\Phi$ is

$$\mathrm{Sym}_P(\Phi) := \{\sigma \in S_J : \exists L \in \mathrm{GL}(X), |c_j| = 1 \text{ with } Lv_j = c_j v_{\sigma j}, \forall j \in J\}.$$

2. The **symmetry group** of $\Phi$ is

$$\mathrm{Sym}(\Phi) := \{\sigma \in S_J : \exists L \in \mathrm{GL}(X) \text{ with } Lv_j = v_{\sigma j}, \forall j \in J\}.$$

These groups are subgroups of $S_J$ (the symmetric group on $J$) and the symmetry group is a subgroup of the projective symmetry group, i.e.,

$$\mathrm{Sym}(\Phi) \subset \mathrm{Sym}_P(\Phi) \subset S_J.$$

An analogue for anti-linear maps is also possible (see Section 5.5).

Projectively similar sequences of vectors have the same projective symmetry group and similar sequences of vectors have the same symmetry group.

The symmetry group and its calculation from the Gram matrix of the canonical tight frame was studied in [VW05], [VW10]. At that time (cf. §5 of [VW10]), the authors believed that the projective symmetry group was generally incalculable (except over $\mathbb{R}$) because of the nonuniqueness of the $L$ and $(c_j)$ in (5.1.1). It is now possible to perform such a calculation using Theorem 4.2.2's characterisation of projective similarity.

## 5.2 Tight frames and the complement of a frame

**Definition 5.2.1.**

Let $\Phi = (v_j)_{j\in J}$ be a frame. The frame operator $S$ of $\Phi$ is invertible and the **dual frame** $\tilde{\Phi} = (\tilde{v}_j)_{j\in J}$ is defined by

$$\tilde{v}_j := S^{-1}v_j, \qquad \forall j \in J. \tag{5.2.1}$$

The **canonical tight frame** $\Phi^{\mathrm{can}} = (v_j^{\mathrm{can}})_{j\in J}$ is defined by

$$v_j^{\mathrm{can}} = S^{-\frac{1}{2}}v_j, \qquad \forall j \in J. \tag{5.2.2}$$

A frame and its dual satisfy the expansion

$$f = \sum_{j\in J}\langle f, v_j\rangle \tilde{v}_j = \sum_{j\in J}\langle f, \tilde{v}_j\rangle v_j, \qquad \forall f \in \mathcal{H},$$

and the canonical tight frame is a normalised tight frame, i.e.,

$$f = \sum_{j \in J} \langle f, v_j^{\text{can}} \rangle v_j^{\text{can}}, \qquad \forall f \in \mathcal{H}.$$

A frame, its dual and canonical tight frame are all similar (Definitions (5.2.1) and (5.2.2)). Normalised tight frames are similar if and only if they are unitarily equivalent, i.e., without a loss of generality, the $Q$ in Definition 4.4.1 is unitary.

Since $S_V$ invertible and $V$ is onto if and only if $V^*$ is one-to-one, $R_\Phi$ is an invertible linear map. A direct calculation shows that

$$R_\Phi = V^* S_V^{-1} = V^* (V V^*)^{-1}$$

takes the vectors of a frame $\Phi$ to the columns of

$$\text{Gram}(\Phi^{\text{can}}) = (S_V^{-\frac{1}{2}} V)^* S_V^{-\frac{1}{2}} V = V^* (V V^*)^{-1} V,$$

i.e.,

$$R_\Phi v_j = V^* (V V^*)^{-1} v_j = \text{Gram}(\Phi^{\text{can}}) e_j,$$

where $e_j$ is the $j$–th standard basis vector.

**Theorem 5.2.2** (Theorem 1.7, p. 26, [CKE13])**.**
A sequence of vectors is a normalised tight frame (for its span) if and only if its Gramian matrix $P$ is an orthogonal projection matrix, i.e., $P^2 = P$ and $P = P^*$.

To determine similarity, $\Phi$ can be replaced by the normalised tight frame given by the columns of the orthogonal projection matrix $\text{Gram}(\Phi^{\text{can}})$. This is extendible to vector spaces (without an inner product), by replacing $\text{Gram}(\Phi^{\text{can}})$ with the orthogonal projection matrix $P_\Phi$ (cf. Theorem 4.4.5, [Wal11]). It is necessary for the underlying field $\mathbb{F}$ of the vector space $X$ to be a subfield of $\mathbb{C}$ which is closed under conjugation, e.g., $\mathbb{F} = \mathbb{Q}, \mathbb{R}, \mathbb{C}$. **For the rest of this chapter, this is assumed to be the case.**

Recall the orthogonal projection maps $P_\Phi$ from Definition 4.4.2.

If $X = \text{span}(\Phi)$ and $\Lambda : X \to \mathbb{F}^m$ is an injective linear map, then

$$P_\Phi = (\Lambda V)^\dagger \Lambda V,$$

where $A^\dagger$ is the pseudoinverse of $A$. For $\Phi$ a frame, taking $\Lambda = V^*$, gives

$$P_\Phi = \mathrm{Gram}(\Phi^{\mathrm{can}}).$$

The sequence $\Phi$ is similar to the normalised tight frame (for a subspace of $\mathbb{F}^J$) given by $(Pe_j)_{j\in J}$, the columns of $P = P_\Phi$ via the well defined injective linear map $R_\Phi : X \to \mathbb{F}^J$ with

$$R_\Phi v_j = P_\Phi e_j, \qquad \forall j. \tag{5.2.3}$$

The Euclidean inner product between these vectors is

$$\langle Pe_j, Pe_k\rangle = P_{kj},$$

i.e., $P$ is the Gram matrix of $(Pe_j)_{j\in J}$.

**Definition 5.2.3.**

Two frames $\Phi$ and $\Psi$ are **complementary** (or **complements** of each other) if $P_\Phi$ and $P_\Psi$ are complementary projection matrices, i.e.,

$$P_\Phi + P_\Psi = I. \tag{5.2.4}$$

The complement of a frame is well defined up to similarity and the complement of a tight frame in the class of normalised tight frames is well defined up to unitary equivalence (Theorem 4.4.5). The complement of a frame in the class of projectively similar frames is defined up to projective similarity.

There is a bijection between permutations $\sigma \in S_J$ and the $J \times J$ permutation matrices, given by $\sigma \mapsto P_\sigma$, where

$$P_\sigma e_j := e_{\sigma j}.$$

**Lemma 5.2.4.**

Let $\Phi = (v_j)_{j\in J}$ be a finite frame for $X$. Then

1. $\sigma \in \mathrm{Sym}(\Phi) \iff P_\sigma^* P_\Phi P_\sigma = P_\Phi$.
2. $\sigma \in \mathrm{Sym}_P(\Phi) \iff P_\sigma^* P_\Phi P_\sigma = C^* P_\Phi C$, for some unitary diagonal matrix $C$.

*Proof.* Observe that $\sigma \in \mathrm{Sym}(\Phi)$ if and only if $\Phi = (v_j)_{j\in J}$ is similar to $\Psi = (v_{\sigma j})_{j\in J}$, and $\sigma \in \mathrm{Sym}_P(\Phi)$ if and only if $\Phi$ is projectively similar to $\Psi = (v_{\sigma j})_{j\in J}$. Let $V = [v_j]$, and $\Lambda : X \to \mathbb{F}^m$ be an injective linear map. Then

$$[v_{\sigma j}] = [Ve_{\sigma j}] = V[e_{\sigma j}] = VP_\sigma,$$

so that

$$P_\Psi = (\Lambda V P_\sigma)^\dagger \Lambda V P_\sigma = P_\sigma^*(\Lambda V)^\dagger \Lambda V P_\sigma = P_\sigma^* P_\Phi P_\sigma.$$

Applying (4.4.2) and Theorem 4.4.5 obtains the desired result. □

Theorem 3.7 of [VW10] shows that if $\Psi$ is a complement of $\Phi$ up to similarity, i.e., $P_\Phi + P_\Psi = I$, then

$$\mathrm{Sym}(\Psi) = \mathrm{Sym}(\Phi).$$

The next theorem is an analogous result for projective symmetry groups.

**Theorem 5.2.5.**

Suppose that $\Phi = (v_j)_{j\in J}$ is a finite frame for $X$. If $\Psi$ is a complement of $\Phi$ up to projective similarity, i.e., $P_\Phi + CP_\Psi C^* = I$, where $C$ is a unitary diagonal matrix, then

$$\mathrm{Sym}_P(\Psi) = \mathrm{Sym}_P(\Phi).$$

*Proof.* Since the definition of frames being complementary is symmetric, it suffices to show that $\mathrm{Sym}_P(\Phi) \subset \mathrm{Sym}_P(\Psi)$. Suppose $\sigma \in \mathrm{Sym}_P(\Phi)$. By Lemma 5.2.4,

$$P_\sigma^* P_\Phi P_\sigma = C_\sigma^* P_\Phi C_\sigma,$$

for some unitary diagonal matrix $C_\sigma$. Now

$$P_\sigma^* P_\Psi P_\sigma = P_\sigma^*(I - C^* P_\Phi C)P_\sigma = I - P_\sigma^* C^* P_\Phi C P_\sigma.$$

Let $c_j$ be the diagonal entries of the matrix $C$. Since $(c_j v_j)_{j\in J}$ is projectively similar to $\Phi = (v_j)_{j\in J}$, $P_{(c_j v_j)} = C^* P_\Phi C$, and $\sigma \in \mathrm{Sym}_P((c_j v_j)) = \mathrm{Sym}_P(\Phi)$, then

$$P_\sigma^* C^* P_\Phi C P_\sigma = \tilde{C}_\sigma^* C^* P_\Phi C \tilde{C}_\sigma,$$

for some unitary diagonal matrix $\tilde{C}_\sigma$. Thus

$$P_\sigma^* P_\Psi P_\sigma = I - \tilde{C}_\sigma^* C^* P_\Phi C \tilde{C}_\sigma = I - \tilde{C}_\sigma^* (I - P_\Psi) \tilde{C}_\sigma = \tilde{C}_\sigma^* P_\Psi \tilde{C}_\sigma.$$

By Lemma 5.2.4, $\sigma \in \mathrm{Sym}_P(\Psi)$. □

**Example 5.2.6.**

Let $\Phi = (v_j)_{j\in J}$ be an equal–norm tight frame of $d+1$ vectors for $\mathbb{C}^d$, e.g., the vertices of the regular simplex. As $P_\Phi$ has a constant diagonal, the complement $\Psi$ of $\Phi$ consists of $d+1$ equal–norm vectors $(w_j)_{j\in J}$ in $\mathbb{C}^1$. Let $w_j = [a_j]$. For any $\sigma \in S_J$,

$$w_j = c_j w_{\sigma j}, \qquad c_j := a_j a_{\sigma j}^{-1},$$

so that $\sigma \in \mathrm{Sym}_P(\Phi)$ and hence

$$\mathrm{Sym}_P(\Phi) = \mathrm{Sym}_P(\Psi) = S_J.$$

Therefore all equal–norm tight frames $\Phi = (v_j)_{j\in J}$ of $d+1$ vectors in $\mathbb{C}^d$ are projectively similar, with projective symmetry group $\mathrm{Sym}_P(\Phi) = S_J$.

**Proposition 5.2.7.**

Let $(v_j)_{j\in J}$ be a normalised tight frame with triple products $T_{jk\ell}$. The complement of $(v_j)$ has triple products $T^c_{jk\ell}$:

$$\begin{aligned} T^c_{jk\ell} &= -\langle v_j, v_k\rangle\langle v_k, v_\ell\rangle\langle v_\ell, v_j\rangle = -T_{jk\ell}, \qquad j \neq k \neq \ell \neq j, \\ T^c_{jjk} &= (1 - \|v_j\|^2)(\delta_{jk}(1 - 2\|v_j\|^2) + |\langle v_j, v_k\rangle|^2), \end{aligned}$$

where $\|v_j\|$ and $|\langle v_j, v_k\rangle|$ are calculated using (4.1.1).

*Proof.*

$$\begin{aligned} T^c_{jk\ell} :&= (\delta_{jk} - \langle v_j, v_k\rangle)(\delta_{k\ell} - \langle v_k, v_\ell\rangle)(\delta_{\ell j} - \langle v_\ell, v_j\rangle) \\ &= \delta_{jk}\delta_{k\ell}\delta_{\ell j} - \delta_{jk}\delta_{k\ell}\langle v_\ell, v_j\rangle - \delta_{jk}\delta_{\ell j}\langle v_k, v_\ell\rangle - \delta_{k\ell}\delta_{\ell j}\langle v_j, v_k\rangle \\ &\quad + \delta_{jk}\langle v_k, v_\ell\rangle\langle v_\ell, v_j\rangle + \delta_{k\ell}\langle v_\ell, v_j\rangle\langle v_j, v_k\rangle + \delta_{\ell j}\langle v_j, v_k\rangle\langle v_k, v_\ell\rangle \\ &\quad - \langle v_j, v_k\rangle\langle v_k, v_\ell\rangle\langle v_\ell, v_j\rangle. \end{aligned}$$

This gives the first equation. Letting $\ell = j$ leads to the second equation. □

## 5.3 Projective invariants

To determine projective similarity between $\Phi = (v_j)_{j\in J}$ and $\Psi = (w_j)_{j\in J}$ (and calculate $\mathrm{Sym}_P(\Phi)$), $\Phi$ and $\Psi$ are assumed to be *normalised tight frames* given by the columns of $P_\Phi$ and $P_\Psi$. If $\Phi$ and $\Psi$ are projectively similar, i.e., $w_j = c_j Q v_j$ (equivalently $W = QVC$), then $Q$ is unitary, since

$$I = WW^* = QVCC^*V^*Q^* = QVV^*Q^* = QQ^*.$$

**Theorem 5.3.1.**
Let $\Phi = (v_j)_{j\in J}$ be a finite frame for $X$. Then $\sigma \in \mathrm{Sym}_P(\Phi)$ if and only if

$$\Delta_C(v_{j_1}, \ldots, v_{j_m}) = \Delta_C(v_{\sigma j_1}, \ldots, v_{\sigma j_m}),$$

for all cycles $(j_1, \ldots, j_m)$ from a determining set for $\Phi$.

*Proof.* Since $\sigma \in \mathrm{Sym}_P(\Phi)$ if and only if $\Phi$ is projectively similar to $\Psi = (v_{\sigma j})_{j\in J}$, this follows from Theorem 4.4.5. □

## 5.4 The algorithm

For frames $\Phi = (v_j)_{j\in J}$ and $\Psi = (w_j)_{j\in J}$ of $n$ vectors, the algorithm presented will determine the set of $\sigma$ for which $\Phi$ and $(w_{\sigma j})$ are projectively similar, i.e.,

$$\Delta_C(v_{j_1}, \ldots, v_{j_m}) = \Delta_C(w_{\sigma j_1}, \ldots, w_{\sigma j_m}), \tag{5.4.1}$$

for all cycles $(j_1, \ldots, j_m)$ from a determining set for $\Phi$. If $\Psi = \Phi$, it calculates $\mathrm{Sym}_P(\Phi)$.

A priori, the calculation of (5.4.1) seems like it requires considering all $n!$ permutations $\sigma \in S_J$. To help reduce the search space, the algorithm uses the $m$–product condition to prune down the possibilities. The more distinct $m$–products exist, the more efficient the pruning.

**Example 5.4.1.**
An extreme example of few $m$–products is when considering the vertices of a regular $d$–simplex

(cf. Example 5.2.6), where

$$(P_\Phi)_{jk} = \begin{cases} \frac{d}{d+1}, & j = k; \\ -\frac{1}{d+1}, & j \neq k, \end{cases} \qquad \mathrm{Sym}_P(\Phi) = S_J.$$

Since the $m$–products are all equal (for fixed $m$), it is relatively simple to check whether $\sigma \in S_J$ is a projective symmetry.

When $\mathrm{Sym}_P(\Phi)$ is large, one could try to construct it by adding generators. Start with a subgroup $X$ containing only the identity. Generate a random permutation not in $X$ and check whether it is in $\mathrm{Sym}_P(\Phi)$. If it is in $\mathrm{Sym}_P(\Phi)$ but not in $X$, then add it as a group generator for $X$ to obtain a larger subgroup. If the index of $\mathrm{Sym}_P(\Phi)$ in $S_J$ is small then this method has a high probability of discovering $\mathrm{Sym}_P(\Phi)$ rather quickly.

The generic case is likely to have many distinct $m$–products, and this is where the algorithm shines.

**Definition 5.4.2.**
For an index set $J$ of size $n$, define a $k$**–flag**

$$f = (j_1, j_2, \ldots, j_k),$$

to be an ordering of $k$ distinct elements of $J$.

For a fixed $n$–flag

$$f_b = (j_1, \ldots, j_n),$$

the permutation $\sigma : j_\ell \mapsto \sigma j_\ell$ (giving a projective similarity or symmetry) is represented by the $n$–flag

$$f_\sigma = (\sigma j_1, \ldots, \sigma j_n).$$

The problem of determining projective equivalence or similarity of $\Phi = (v_j)_{j\in J}$ and $\Psi = (w_{\sigma j})_{j\in J}$ can be refactored into a question about which of the $n!$ permutations $\sigma$ ($n$–flags $f_\sigma$), satisfy (5.4.1).

Each $n$–flag $f_\sigma = (\sigma j_1, \ldots, \sigma j_n)$ can be built from the 0-flag $()$ by concatenating successive entries, e.g.,

$$f_\sigma^0 = (), \quad f_\sigma^1 = (\sigma j_1), \quad f_\sigma^2 = (\sigma j_1, \sigma j_2), \quad \ldots \quad f_\sigma^n = (\sigma j_1, \sigma j_2, \ldots, \sigma j_n).$$

The operation of transitioning from a set $\mathcal{F}_{k-1}$ of $(k-1)$–flags to a set $\mathcal{F}_k$ of $k$–flags (via concatenation) is called **growing**. At the $k$–th stage there are $n-k+1$ choices for the next entry, so

$$|\mathcal{F}_k| = (n-k+1)|\mathcal{F}_{k-1}|.$$

If $|\operatorname{Sym}(\Phi)| < n!$, then not all $f_\sigma^k \in \mathcal{F}_k$ can be grown to an $n$–flag satisfying (5.4.1) for all cycles in a determining set. Removing elements from $\mathcal{F}_k$ for failing this condition is called **pruning**.

### 5.4.1 Algorithm

To determine the $n$–flags $\mathcal{F}_n$ giving a projective similarity:

```
Let F_0 := {()} consist of the empty flag
for k from 1 to n do
  Grow F_{k-1} to F_k
  Prune F_k
end for
```

There is an art to how often to prune and which parts to prune along the way. For calculations regarding Weyl-Heisenberg SIC-POVMs for example, pruning at each growth stage still returns results in a reasonable amount of time.

The algorithm can be parallelised by partitioning $\mathcal{F}_k$ into smaller blocks, at any stage $k$, and applying the procedure to each block.

**Example 5.4.3.**

The simplest example of a SIC-POVM (cf. Section 5.6) is the equiangular tight frame $\Phi :=$

$(v, Sv, \Omega v, S\Omega v)$ of four vectors for $\mathbb{C}^2$, where

$$v := \frac{1}{\sqrt{6}} \begin{pmatrix} \sqrt{3+\sqrt{3}} \\ e^{\frac{\pi}{4}i}\sqrt{3-\sqrt{3}} \end{pmatrix}, \qquad S := \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}, \qquad \Omega := \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}.$$

Then

$$P_\Phi = \frac{1}{2} \begin{pmatrix} 1 & \frac{1}{\sqrt{3}} & \frac{1}{\sqrt{3}} & -\frac{i}{\sqrt{3}} \\ \frac{1}{\sqrt{3}} & 1 & \frac{-i}{\sqrt{3}} & \frac{1}{\sqrt{3}} \\ \frac{1}{\sqrt{3}} & \frac{i}{\sqrt{3}} & 1 & -\frac{1}{\sqrt{3}} \\ \frac{i}{\sqrt{3}} & \frac{1}{\sqrt{3}} & -\frac{1}{\sqrt{3}} & 1 \end{pmatrix}.$$

The empty flag (0–flag) $\mathcal{F}_0 = \{()\}$ grows to the set of 1-flags

$$\mathcal{F}_1 = \{(1), (2), (3), (4)\}.$$

The pruning rule preserves the norm, i.e., $\langle v_1, v_1 \rangle = \langle w_{\sigma 1}, w_{\sigma_1} \rangle$. Since SIC-POVM vectors are unit length, there is no pruning. Growing gives

$$\mathcal{F}_2 = \{(1,2), (1,3), (1,4), (2,1), (2,3), (2,4), (3,1), (3,2), (3,4), (4,1), (4,2), (4,3)\},$$

and pruning gives no reduction as $\Phi$ is equiangular. Now consider the 2–flag $(3,2)$ (the others being similar). This grows to the 3–flags $(3,2,1)$, $(3,2,4)$. Since

$$\Delta_C(v_1, v_2, v_3) = \frac{i}{24\sqrt{3}}, \qquad \Delta_C(w_3, w_2, w_1) = -\frac{i}{24\sqrt{3}}, \quad \Delta_C(w_3, w_2, w_4) = \frac{i}{24\sqrt{3}},$$

the 3–flag $(3,2,1)$ is pruned. Repeating this procedure obtains

$$\begin{aligned} \mathcal{F}_3 = \{&(1,2,3), (1,3,4), (1,4,2), (2,1,4), (2,3,1), (2,4,3), \\ &(3,1,2), (3,2,4), (3,4,1), (4,1,3), (4,2,1), (4,3,2)\}. \end{aligned}$$

The final stage $k = n$ does not increase the size of $\mathcal{F}_{n-1}$ and in this case nothing gets pruned. So the symmetry group of $\Phi$ is

$$\begin{aligned} \mathrm{Sym}_P(\Phi) = \mathcal{F}_4 = \{&(1,2,3,4), (1,3,4,2), (1,4,2,3), (2,1,4,3), (2,3,1,4), (2,4,3,1), \\ &(3,1,2,4), (3,2,4,1), (3,4,1,2), (4,1,3,2), (4,2,1,3), (4,3,2,1)\}. \end{aligned}$$

This is the alternating group $A_4$.

**Example 5.4.4.**

Let $\Phi = (v_j)_{j\in J}$ be the maximal MUB (see Definition 4.3.10, Remark 4.3.12) in $\mathbb{C}^2$,

$$v_1 = e_1, \quad v_2 = e_2, \quad v_3 = \frac{1}{\sqrt{2}}(e_1 + e_2), \quad v_4 = \frac{1}{\sqrt{2}}(e_1 - e_2).$$

Then

$$P_\Phi = \frac{1}{2}\begin{pmatrix} 1 & 0 & \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} \\ 0 & 1 & \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} & 1 & 0 \\ -\frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} & 0 & 1 \end{pmatrix}.$$

Like Example 5.4.3, the procedure up to $\mathcal{F}_2$ results in no pruning. The pruning rule now requires the modulus of the inner product between $v_1$ and $v_2$ to be preserved, so the index pairs in $\mathcal{F}_2$ must correspond to pairs of orthogonal vectors, giving

$$\mathcal{F}_2 = \{(1,2),(2,1),(3,4),(4,3)\}.$$

Growing again gives

$$\mathcal{F}_3 = \{(1,2,3),(1,2,4),(2,1,3),(2,1,4),(3,4,1),(3,4,2),(4,3,1),(4,3,2)\}.$$

As all 3–products for distinct vectors are zero, there is no further pruning. Similarly, all 4–products for distinct vectors are zero, so

$$\begin{aligned}\mathrm{Sym}_P(\Phi) = \mathcal{F}_4 = \{&(1,2,3,4),(1,2,4,3),(2,1,3,4),(2,1,4,3),\\ &(3,4,1,2),(3,4,2,1),(4,3,1,2),(4,3,2,1)\}.\end{aligned}$$

This group is isomorphic to the dihedral group of order 8 (the unique isomorphism class of order 8), generated by the permutations

$$(1324) \quad \text{(rotation through 90 degrees)}, \qquad (34) \quad \text{(reflection in the } x\text{–axis)}.$$

## 5.5 The extended projective symmetry group

**Definition 5.5.1.**

Let $\Phi = (v_j)_{j\in J}$ be a finite sequence of vectors with span $\mathbb{C}^d$. Then the **extended projective**

**symmetry group** of $\Phi$ is

$$\mathrm{Sym}_{EP}(\Phi) := \{\sigma \in S_J : \exists A \in \mathcal{EGL}(\mathbb{C}^d), |c_j| = 1 \text{ with } Av_j = c_j v_{\sigma j}, \forall j \in J\}.$$

If $A$ is anti-linear then $\sigma$ is an **anti-projective symmetry**.

A permutation $\sigma$ is an anti-projective symmetry of $\Phi = (v_j)_{j\in J}$ if and only if for some linear $L$

$$L\overline{v_j} = c_j v_{\sigma j}, \qquad \forall j \in J,$$

i.e., $\overline{\Phi} = (\overline{v_j})_{j\in J}$ and $(v_{\sigma j})_{j\in J}$ are projectively similar. Since $P_{\overline{\Phi}} = \overline{P_\Phi}$,

$$\Delta_C(\overline{v_{j_1}}, \ldots, \overline{v_{j_m}}) = \overline{\Delta_C(v_{j_1}, \ldots, v_{j_m})} = \Delta_C(v_{j_m}, \ldots, v_{j_1}).$$

By Theorem 4.4.5, $\sigma$ is an anti-projective symmetry of $\Phi = (v_j)_{j\in J}$ if and only if

$$\Delta_C(v_{j_m}, \ldots, v_{j_1}) = \Delta_C(v_{\sigma j_1}, \ldots, v_{\sigma j_m}), \tag{5.5.1}$$

for all cycles $(j_1, \ldots, j_m)$ from a determining set for $\Phi$. Condition (5.5.1) is (5.4.1) with the ordering of $(v_{j_1}, \ldots, v_{j_m})$ reversed, so the algorithm can calculate the anti-projective symmetries by replacing $P_\Phi$ by its transpose $P_\Phi^T$.

**Example 5.5.2.**

Applying the algorithm to Example 5.4.3, with the $m$–products $\Delta_C(v_{j_1}, \ldots, v_{j_m})$ replaced by their conjugates gives the following anti-projective symmetries

$$\begin{aligned}\mathcal{F}_4 = \{&(1,2,4,3),(1,3,2,4),(1,4,3,2),(2,1,3,4),(2,3,4,1),(2,4,1,3),\\ &(3,1,4,2),(3,2,1,4),(3,4,2,1),(4,1,2,3),(4,2,3,1),(4,3,1,2)\}.\end{aligned}$$

Therefore,

$$\mathrm{Sym}_P(\Phi) = A_4 \subset \mathrm{Sym}_{EP}(\Phi) = S_4.$$

The product of two anti-projective symmetries is a projective symmetry. If there exists anti-projective symmetries of $\Phi$, then

$$|\mathrm{Sym}_{EP}(\Phi)| = 2|\mathrm{Sym}_P(\Phi)|.$$

$\mathrm{Sym}_{EP}(\Phi)$ will be generated by either the projective symmetries alone or together with any anti-projective symmetry, so that

$$\mathrm{Sym}_P(\Phi) \triangleleft \mathrm{Sym}_{EP}(\Phi), \qquad [\mathrm{Sym}_{EP}(\Phi) : \mathrm{Sym}_P(\Phi)] = 1 \text{ or } 2.$$

## 5.6 Group frames, nice error bases, SIC-POVMs and MUBs

When a tight frame comes as the orbit of a unitary group action on $\mathcal{H} = \mathbb{R}^d, \mathbb{C}^d$ the calculation of $\mathrm{Sym}_P(\Phi)$ can be simplified (See Chapter 5, [CKE13]).

Let $G$ be a finite group and $\rho : G \to \mathcal{U}(\mathcal{H})$ be a unitary representation of $G$. When $\rho(G)$ contains non-identity scalar matrices, there are repeated vectors (up to scalar multiples) in $\Phi = (gv)_{g\in G}$. In this situation, it might be convenient to consider $(gv)_{g\in\rho(G)/Z}$, where $Z$ is the group of scalar matrices in $\rho(G)$ (cf. *projective representations*, *nice error bases* cf. [Kni96a], [KR02], [CW14b]). The group frame $\Phi = (gv)_{gZ\in\rho(G)/Z}$ is well defined if the vectors $gv$ are identified together when they are unit scalar multiples of each other. As the vectors are unit length, it is formally in a projective space. In practise, one just selects a vector to use from the representative class of vectors which are unit scalar multiples of each other. Note that the $m$–products are well defined in this case as they are invariant under unit scalings of the vectors. Abusing notation slightly, $g \in G$ will also stand for $gZ \in \rho(G)/Z$ when the context is clear.

Let $\Phi = (gv)_{g\in G}$ be a group frame. The projective symmetries are permutations of $G$ (as $G$ indexes $\Phi$). Each $h \in G$ induces a projective symmetry via the map

$$\tau_h : G \to G : g \mapsto hg,$$

with the subgroup $\tau_G = \{\tau_h\}_{h\in G}$ of $\mathrm{Sym}_P(\Phi)$ acting transitively on $G$. Let $\sigma \in \mathrm{Sym}_P(\Phi)$ and $h \in G$, then $\sigma$ is a product of an element of $\tau_G$ and a $\sigma_h \in \mathrm{Sym}_P(\Phi)$ which fixes the vector $hv$, i.e.,

$$\sigma = \tau_{\sigma(h)h^{-1}}\sigma_h, \qquad \sigma_h := \tau_{h\sigma(h)^{-1}}\sigma.$$

**Proposition 5.6.1.**

Let $\Phi = (gv)_{g\in G}$ be a group frame, then $\mathrm{Sym}_P(\Phi)$ is generated by the "translations" $\tau_G$, and

the stabiliser of any vector $hv$, i.e.,

$$\mathrm{Stab}(h) := \{\sigma \in \mathrm{Sym}_P(\Phi) : \sigma(h) = h\}.$$

In light of Proposition 5.6.1, computing $\mathrm{Sym}_P(\Phi)$ boils down to finding the stabiliser of a vector in your frame.

**Example 5.6.2.**

If $|G| = n$, the calculation of $\mathrm{Stab}(h)$ requires the examination of the $(n-1)!$ permutations which fix $h$. Checking (5.4.1) for cycles in a determining set for $\Phi = (gv)_{g\in G}$ which do not involve $h$ further improves the pruning efficiency.

**Example 5.6.3.**

Consider Example 5.4.3. This is a group frame of order 8 with $\rho(G) = \langle S, \Omega\rangle$ and centre $Z = \{I, -I\}$. $\rho(G)/Z$ is isomorphic to $\mathbb{Z}_2 \times \mathbb{Z}_2$ (as $\Omega S = -S\Omega$). Giving the elements the order $(Z, SZ, \Omega Z, S\Omega Z)$, the translations are

$$\tau_Z = I, \quad \tau_{SZ} = (12)(34), \quad \tau_{\Omega Z} = (13)(24), \quad \tau_{S\Omega Z} = (14)(23).$$

Considering only projective symmetries with $\sigma 1 = 1$ (that fix the first vector), the algorithm gives

$$\mathcal{F}_2 = \{(1,2), (1,3), (1,4)\}.$$

Growing and pruning gives

$$\mathcal{F}_3 = \{(1,2,3), (1,3,4), (1,4,2)\}, \qquad \mathcal{F}_4 = \{(1,2,3,4), (1,3,4,2), (1,4,2,3)\}.$$

The stabliser of the first vector is generated by the cycle $(234)$ and $\tau_{\rho(G)/Z}$ is generated by $(12)(34)$ and $(13)(24)$, so

$$\mathrm{Sym}_P(\Phi) = \langle (234), (12)(34), (13)(24)\rangle.$$

The rest of the chapter presents symmetry group calculations for SIC-POVMs and MUBs. These were done with Maple and MAGMA.

It is natural to consider SIC-POVMs up to projective equivalence, i.e., as sequences of equiangular lines. Let $G$ be the index group of the Weyl Heisenberg group (Weyl Heisenberg group

factored by its centre). Then Weyl Heisenberg SIC-POVMs are determined (up to projective unitary equivalence) by their 3–products (cf. [AFF11], Theorem 4.3.8)

$$\Lambda^j_{g,h} := d^3\|v\|^6\Delta_C(jv, gv, hv) = \langle jv, gv\rangle\langle gv, hv\rangle\langle hv, jv\rangle, \qquad j, g, h \in G.$$

So a permutation $\sigma : G \to G$ is a projective symmetry if and only if

$$\Lambda^{\sigma j}_{\sigma g,\sigma h} = \Lambda^j_{g,h}, \qquad \forall j, g, h.$$

To compute the projective symmetry, it is sufficient to find the stabiliser of some fixed $j \in G$ (Proposition 5.6.1), i.e., the subgroup of $\sigma$ satisfying

$$\Lambda^j_{\sigma g,\sigma h} = \Lambda^j_{g,h}, \qquad \forall g, h.$$

Zhu observed this in Chapter 10 of [Zhu12] and replaced the Hermitian $G \times G$ matrix $[\Lambda^j_{g,h}]$ by the real antisymmetric matrix $\Lambda^{(j)}$, with entries

$$\Lambda^{(j)}_{g,h} := \arg\Big(\frac{\Lambda^j_{g,h}}{|\Lambda^j_{g,h}|}\Big) \in (-\pi, \pi].$$

Revisiting Example 5.6.3 (with the same ordering),

$$\Lambda^I = \frac{1}{8}\begin{pmatrix} 1 & \frac{1}{3} & \frac{1}{3} & \frac{1}{3} \\ \frac{1}{3} & \frac{1}{3} & \frac{i}{3\sqrt{3}} & -\frac{i}{3\sqrt{3}} \\ \frac{1}{3} & -\frac{i}{3\sqrt{3}} & \frac{1}{3} & \frac{i}{3\sqrt{3}} \\ \frac{1}{3} & \frac{i}{3\sqrt{3}} & -\frac{i}{3\sqrt{3}} & \frac{1}{3} \end{pmatrix}, \qquad \Lambda^{(I)} = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & \frac{\pi}{2} & -\frac{\pi}{2} \\ 0 & -\frac{\pi}{2} & 0 & \frac{\pi}{2} \\ 0 & \frac{\pi}{2} & -\frac{\pi}{2} & 0 \end{pmatrix}.$$

The angle matrix $\Lambda$ can be viewed as a weighted (directed) graph with vertices $G$. In light of this, the stabiliser of $\Lambda^{(j)}_{g,h}$, i.e., the set of permutations satisfying

$$\Lambda^{(j)}_{\sigma g,\sigma h} = \Lambda^{(j)}_{g,h}, \qquad \forall g, h,$$

is the automorphism group of $\Lambda^{(j)}$, so the stabiliser problem is equivalent to the well studied weighted graph isomorphism problem. Zhu's algorithm in section 10.2.3 of [Zhu12] utilises the machinery of the weighted graph isomorphism problem to calculate the stabiliser of $\Lambda^{(j)}$. Zhu applied it to numerically known SIC-POVMs in dimensions $d \neq 3$ (see Table 10.2 of [Zhu12]).

The algorithm (5.4.1) presented here is able to compute the same extended projective symmetry. Furthermore, it was possible to identify the subgroup of it that lies in the extended Clifford group (the normaliser of the Weyl–Heisenberg group in the unitary and antiunitary matrices), as given in [SG10].

**Corollary 5.6.4.**

Let $\Phi$ be a finite frame. Then a necessary condition for $\Phi$ to be a group frame for $G/Z$ is that $\mathrm{Sym}_P(\Phi)$ has a transitive subgroup which is isomorphic to $G/Z$.

**Lemma 5.6.5.**

Let $\Phi = (v_j)_{j\in J}$ be a finite frame. If the frame graph of $\Phi$ is connected, then the map $\mathrm{Sym}_P(\Phi) \to \mathrm{PGL}(n,\mathbb{C})$ given by $\sigma \mapsto L_\sigma$, where $L_\sigma$ is

$$Lv_j = c_j v_{\sigma j}, \quad |c_j| = 1, \qquad \forall j,$$

is well defined and a projective representation of $\mathrm{Sym}_P(\Phi)$.

*Proof.* Let $(v_j)_{j\in J}$ be a normalised tight frame. Then $L = U$ is unitary and $\langle v_k, v_j\rangle = \langle P_\Phi e_k, P_\Phi e_j\rangle = (P_\Phi)_{kj}$. Suppose

$$Uv_j = c_j v_{\sigma j}, \quad \tilde{U}v_j = \tilde{c}_j v_{\sigma j}, \qquad \forall j, \tag{5.6.1}$$

with $U$, $\tilde{U}$ unitary, and $c_j$, $\tilde{c}_j$ unit scalars. Then

$$\langle \overline{c_j}Uv_j, \overline{c_k}Uv_k\rangle = \langle v_{\sigma j}, v_{\sigma k}\rangle = \langle \overline{\tilde{c}_j}\tilde{U}v_j, \overline{\tilde{c}_k}\tilde{U}v_k\rangle, \qquad \forall j,k,$$

giving

$$\frac{c_k}{c_j}\langle v_j, v_k\rangle = \frac{\tilde{c}_k}{\tilde{c}_j}\langle v_j, v_k\rangle, \qquad \forall j,k.$$

Hence,

$$\langle P_\Phi e_j, P_\Phi e_k\rangle = \langle v_j, v_k\rangle \neq 0 \quad \Longrightarrow \quad \frac{c_j}{\tilde{c}_j} = \frac{c_k}{\tilde{c}_k}. \tag{5.6.2}$$

Since the frame graph of $\Phi$ is connected, for any $j$ and $k$, there exists a sequence of vectors $v_j, v_{\ell_1}, v_{\ell_2}, \ldots, v_{\ell_m}, v_k$ with

$$\langle v_j, v_{\ell_1}\rangle \neq 0, \quad \langle v_{\ell_1}, v_{\ell_2}\rangle \neq 0, \quad \ldots \quad \langle v_{\ell_{m-1}}, v_{\ell_m}\rangle \neq 0, \quad \langle v_{\ell_m}, v_k\rangle \neq 0,$$

so by (5.6.2),

$$\frac{c_j}{\tilde{c}_j} = \frac{c_{\ell_1}}{\tilde{c}_{\ell_1}} = \frac{c_{\ell_2}}{\tilde{c}_{\ell_2}} = \cdots = \frac{c_{\ell_m}}{\tilde{c}_{\ell_m}} = \frac{c_k}{\tilde{c}_k}.$$

Thus for some unit scalar $\alpha$, $\tilde{c}_j = \alpha c_j$ for all $j$ and by (5.6.1),

$$\tilde{U}v_j = \tilde{c}_j v_{\sigma j} = \alpha c_j v_{\sigma j} = \alpha U v_j, \qquad \forall j,$$

i.e., $\tilde{U} = \alpha U$.

For the general case, rewrite $Lv_j = c_j v_{\sigma j}$ (with (5.2.3)) as

$$\left(R_\Phi L R_\Phi^{-1}\right) P_\Phi e_j = c_j (P_\Phi e_j), \qquad \forall j,$$

where the inverse of the injective linear map $R_\Phi$ is defined on its image (the range of $P_\Phi$). Since $(P_\Phi e_j)$ is a normalised tight frame and $U = R_\Phi L R_\Phi^{-1}$ is unitary,

$$R_\Phi \tilde{L} R_\Phi^{-1} = \alpha R_\Phi L R_\Phi^{-1} \quad \Longrightarrow \quad \tilde{L} = \alpha L.$$

The homomorphism property follows since

$$L_\sigma L_\tau(v_j) = L_\sigma(L_\tau(v_j)) = c_j^\tau L_\sigma(v_{\tau j}) = c_j^\tau c_j^\sigma v_{\sigma\tau j} = L_{\sigma\tau}(v_j).$$

□

**Theorem 5.6.6.**

Let $\Phi = (v_j)_{j\in J}$ be a frame of $n$ vectors for $\mathcal{H}$. If the frame graph of $(P_\Phi e_j)_{j\in J}$ is connected and no vector in $\Phi$ is repeated (up to a unit scalar multiple), then $\Phi$ is a group frame for $G/Z$ if and only if $\mathrm{Sym}_P(\Phi)$ has a transitive subgroup which is isomorphic to $G/Z$. In the $G/Z$–frame, the vectors of $\Phi$ are repeated $\frac{|G/Z|}{n}$ times.

*Proof.* By Corollary 5.6.4, it is sufficient to show, for a given transitive subgroup $H$ of $\mathrm{Sym}_P(\Phi)$, there exists a group $G$ of linear maps such that, for all $v \in \Phi$, the orbit of $G$ acting on $v$, up to repeats, is $\Phi$. Let $d$ be the dimension of $\mathcal{H}$, and $\omega := e^{\frac{2\pi i}{d}}$. For each $\sigma \in H$, choose a $L_\sigma$ as in Lemma 5.6.5, with

$$\det(L_\sigma) = 1.$$

There are $d$ scalar multiples of $L_\sigma$ with determinant 1, namely $L_\sigma, \omega L_\sigma, \ldots, \omega^{d-1} L_\sigma$. Let

$$G := \{\omega^j L_\sigma : j = 0, \ldots, d-1, \sigma \in H\}, \qquad Z := \{\omega^j I\}_{j=0}^{d-1}.$$

Since

$$L_{\sigma\tau} v_j = c_j^{\sigma\tau} v_{\sigma\tau j}, \qquad L_\sigma L_\tau v_j = L_\sigma c_j^\tau v_{\tau j} = c_j^\tau c_{\tau j}^\sigma v_{\sigma\tau j},$$

$G$ is a group and $G/Z \simeq H$. As the vectors in $\Phi$ are not repeated, $G$ has $d|H|$ elements. By the uniqueness of $L_{\sigma\tau}$,

$$\begin{aligned} L_\sigma L_\tau = \alpha L_{\sigma\tau}, \quad |\alpha| = 1 \quad &\implies \quad 1 = \det(L_\sigma L_\tau) = \det(\alpha L_{\sigma\tau}) = \alpha^d \\ &\implies \quad \alpha \in \{1, \omega, \ldots, \omega^{d-1}\}, \quad L_\sigma L_\tau \in G. \end{aligned}$$

As $H$ is transitive, $\Phi$ is the $G$–orbit of any one of its vectors (up to repeats). □

Consider the generalisation of Example 5.4.4.

**Example 5.6.7.**

It is possible to add a third orthonormal basis to the MUB of Example 5.4.4, to obtain a maximal set of MUBs for $\mathbb{C}^2$:

$$\Phi = (v_j)_{j\in J} = \Big( \begin{pmatrix} 1 \\ 0 \end{pmatrix}, \begin{pmatrix} 0 \\ 1 \end{pmatrix}, \frac{1}{\sqrt{2}} \begin{pmatrix} 1 \\ 1 \end{pmatrix}, \frac{1}{\sqrt{2}} \begin{pmatrix} -1 \\ 1 \end{pmatrix}, \frac{1}{\sqrt{2}} \begin{pmatrix} 1 \\ i \end{pmatrix}, \frac{1}{\sqrt{2}} \begin{pmatrix} 1 \\ -i \end{pmatrix} \Big).$$

The projective symmetry group $\mathrm{Sym}_P(\Phi)$ has order 24 and is generated by the permutations

$$(13)(24)(56), \quad (3645).$$

It has three transitive subgroups

$$\begin{aligned} H_1 &= \langle (12)(36)(45), (164)(253) \rangle \approx S_3, \qquad |H_1| = 6, \\ H_2 &= \langle (164)(253), (34)(56), (12)(56) \rangle, \qquad |H_2| = 12, \\ H_3 &= \mathrm{Sym}_P(\Phi), \qquad |H_3| = 24. \end{aligned}$$

The three MUBs are a group frame for $G/Z \simeq H_j$. For $G/Z \simeq S_3$, it is possible to choose $G$ so that $Z = \{1\}$, e.g., take

$$G = \langle A, B \rangle, \qquad A := \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}, \quad B := \frac{1}{\sqrt{2}} \begin{pmatrix} 0 & -1-i \\ -1+i & 0 \end{pmatrix}.$$

For $d$ a prime power, there is a standard construction for $\Phi$ a maximal set of $d+1$ MUBs in $\mathbb{C}^d$, as the eigenvectors of elements of a Weyl–Heisenberg group [All80], [WF89]. This $\Phi$ is a group frame where the group is a subgroup of the Clifford group [Bla14].

| $d$ | $m$ | $n$ | $\mathcal{C}(\mathbb{Z}_d)/Z$ | $\mathrm{Sym}_P(\Phi)$ | $\mathrm{Sym}_{EP}(\Phi)$ | TS of $\mathrm{Sym}_P(\Phi)$ |
|---|---|---|---|---|---|---|
| 2 | 3 | 6 | <24,12> | <24,12> | <48,48> | <6,1>, <12,3> |
| 3 | 4 | 12 | <216,153> | <216,153> | <432,734> | <72,41> |
| 4 | 5 | 20 | <768,1088659> | 1920 | 3840 | <20,3>,<60,5>, <80,49>, <120,34>, <160,234>, <320,1635>, <960,11357> |
| 5 | 6 | 30 | 3000 | 3000 | 6000 | <600,150> |
| 7 | 8 | 56 | 16464 | | | |

TABLE 5.1: The symmetry groups $\mathrm{Sym}_P(\Phi)$ and $\mathrm{Sym}_{EP}(\Phi)$ for $\Phi$ the tight frame of $n$ vectors given by $m$ MUBs in $\mathbb{C}^d$, including the proper transitive subgroups of $\mathrm{Sym}_P(\Phi)$. The last column lists the transitive subgroups of $\mathrm{Sym}_P(\Phi)$. Let <n,k> be the $k$–th group of order $n$ in the Small Groups Library (as used in MAGMA). When the MAGMA routine IdentifyGroup(G) is unable to identify the group, the order of the group is given instead.

For $d$ a prime, the appropriate Weyl–Heisenberg group is generated by $S$ and $\Omega$ of (2.3.1) and the Clifford group is generated by the Fourier matrix $F$ and the diagonal matrix $R$, where

$$F_{jk} := \frac{1}{\sqrt{d}} \omega^{-jk}, \qquad R_{jk} := \mu^{j(j+d)} \delta_{jk}, \qquad \mu := e^{\frac{2\pi i}{2d}}.$$

Let $\mathcal{C}(\mathbb{Z}_d)/Z$ be this Clifford group factored by its centre. Then

$$\Phi = (ge_j)_{g \in \mathcal{C}(\mathbb{Z}_d)/Z},$$

where $e_j$ is a standard basis vector. The calculations in Table 5.1 indicate that for $d \leq 5$, the Clifford group accounts for all the projective symmetries and the extended Clifford group (see Section 2.4 and Section 8 of [App05]) for all extended projective symmetries.

**Conjecture 5.6.8.**

Let $d$ be a prime. The projective symmetry group of maximal MUBs in $\mathbb{C}^d$ with the standard

construction is the Clifford group $\mathcal{C}$ just defined.

For $d$ a prime power $p^m$, $m > 1$, the appropriate Clifford group is the *Galoisian Clifford group* of [App09] (Section 2) and [KT12] (Section 6).

**Example 5.6.9.**

Let $\Phi$ be the five MUBs in $\mathbb{C}^4$ (unique up to projective similarity). The projective symmetry group $\mathrm{Sym}_P(\Phi)$ is transitive, with order $1920 = 2^7 \cdot 3 \cdot 5$. The Galoisian Clifford group has order $11520 = 2^8 \cdot 3^2 \cdot 5$ (see Section 6 of [KT12]). Since $\Phi$ is a group frame for a subgroup of the Galoisian Clifford group (see [WS07]), $\Phi$ is not a group frame for the full Galoisian Clifford group. It is a group frame for groups with orders 20, 60, 80, 120, 160, 320, 960, 1920 (see Table 5.1). Whether all of these groups appear as subgroups of the Galoisian Clifford group is unknown.

## 5.7 Harmonic frames

**Definition 5.7.1.**

A frame $\Phi = (v_j)_{j\in J}$ is **projectively real** if all its $m$–products are real (i.e., projectively similar to a frame in $\mathbb{R}^d$). Otherwise it is **projectively complex**. $\Phi$ has **projectively distinct vectors** if for any $i, j \in J$, $i \neq j$,

$$v_i \neq c_j v_j, \qquad \text{for all } c_j \in \mathbb{T},$$

i.e., none of the lines corresponding to vectors of equal length are equal.

**Example 5.7.2.**

The harmonic frame (for $\mathbb{C}^2$)

$$\Phi = \Phi_{\{0,1\}} = \{v_0, v_1, v_2\} = \left\{ \begin{pmatrix} 1 \\ 1 \end{pmatrix}, \begin{pmatrix} 1 \\ \omega \end{pmatrix}, \begin{pmatrix} 1 \\ \omega^2 \end{pmatrix} \right\}, \qquad \omega = e^{\frac{2\pi}{3}},$$

is not similar to real frame, but is projectively similar to a real frame, since

$$\mathrm{Gram}(\Phi) = \begin{pmatrix} 2 & 1+\omega & 1+\omega^2 \\ 1+\omega^2 & 2 & 1+\omega \\ 1+\omega & 1+\omega^2 & 2 \end{pmatrix}, \qquad \langle v_0, v_1\rangle\langle v_1, v_2\rangle\langle v_2, v_0\rangle = (1+\omega^2)^3 = -1.$$

The classes of projectively equivalent cyclic harmonic frames (up to reordering) were calculated in Figure 4.1. The projective symmetry group and the extended projective symmetry group were calculated with the algorithm (5.4.1) for the cases with projectively distinct vectors. The results for $d = 2, \ldots, 7$ are summarised in the tables of the Appendix B.

**Example 5.7.3.**

For projectively complex cyclic harmonic frames the conjugation map gives an anti-projective symmetry $\sigma : k \mapsto -k$, since

$$\overline{v_k} = \overline{(\omega^{kj_1}, \ldots, \omega^{kj_d})} = (\omega^{-kj_1}, \ldots, \omega^{-kj_d}) = v_{-k}, \qquad k \in \mathbb{Z}_n.$$

$\mathrm{Sym}_P(\Phi)$ is an index 2 subgroup of $\mathrm{Sym}_{EP}(\Phi)$ (and hence normal). In all of the cases considered so far (including Example 5.5.2), $\mathrm{Sym}_{EP}(\Phi)$ is not isomorphic to the direct product $\mathrm{Sym}_P(\Phi) \times \mathbb{Z}_2$.

The following examples are about cyclic harmonic frames. The notation from Chapter 4, section 4.5 is adopted.

**Example 5.7.4** (Four vectors in $\mathbb{C}^2$)**.**

There are two projective equivalence classes, given by $\{0, 1\}$ and $\{1, 3\}$. The first has projectively distinct vectors, is projectively real and requires the 4–products to calculate its projective symmetry group <8,3>. The second does not have vectors which are projectively distinct.

**Example 5.7.5** (Six vectors in $\mathbb{C}^3$)**.**

There are three projective equivalence classes. The first, $\{1, 3, 5\}$, does not have projectively distinct vectors,. The second $\{1, 2, 3\}$, has projectively distinct vectors, is projectively real and requires the 4–products to calculate its projective symmetry group <12,4>. The third, $\{0, 1, 4\}$, is projectively complex, with projectively distinct vectors, none of which are orthogonal. Its projective and extended projective symmetry groups are the symmetry groups <18,3> and <36,10>. This is the first example of a projectively complex cyclic harmonic frame. For $d = 3$, they also exist for $n = 7, 8, 9, 10, 11, 12$.

**Example 5.7.6** (Eight vectors in $\mathbb{C}^3$)**.**

The harmonic frame $\Phi = \Phi_J$ of eight vectors for $\mathbb{C}^3$ given by $J = \{0, 1, 4\}$ is projectively complex, with

$$\mathrm{Sym}_P(\Phi) = \texttt{<32,11>}, \qquad \mathrm{Sym}_{EP}(\Phi) = \texttt{<64,134>}.$$

The projective symmetry group has transitive subgroups of order eight isomorphic to $\mathbb{Z}_8$, $\mathbb{Z}_4 \times \mathbb{Z}_2$, $D_4$ (dihedral group) and $Q_8$ (quaternian group). $\Phi$ is a group frame for each of these groups. The only other group of order eight is $\mathbb{Z}_2 \times \mathbb{Z}_2 \times \mathbb{Z}_2$, which occurs as a transitive subgroup of $\mathrm{Sym}_{EP}(\Phi)$.

**Example 5.7.7** (Projectively real frames)**.**

In 2 dimensions all cyclic harmonic frames of $n$ vectors are projectively real. For all $n > 4$, they are determined by their 3–products and all the 3–products are real. For $d \geq 3$, projectively complex cyclic harmonic frames of $n$ projectively distinct vectors appear to always exist, with the smallest $n$ being $n = d + 3$.

**Example 5.7.8** (Erasures)**.**

The **erasures** of a frame $\Phi$ of $n$ vectors for $\mathbb{C}^d$ is the maximum number of vectors that can be removed from $\Phi$ before it stops spanning $\mathbb{C}^d$. Frames with a large number of erasures are useful for robust data transmission [GVT98], [GKK01]. The cyclic harmonic frame given by $J = \{0, 1, 2, \ldots, d-1\}$ has $n - d$ erasures. In general, this is not true, e.g., for six vectors in $\mathbb{C}^2$ the erasures of a cyclic harmonic frame can be $2, 3, 4$. However for $n$ prime, the tabulated data in Appendix B suggests the number of erasures *is* always $n - d$

# Chapter 6

# Nice error frames

SIC-POVMs can arise as orbits of groups which are not the Weyl-Heisenberg groups. One category of groups where this phenomenon has occurred is the nice error groups. The Weyl-Heisenberg groups belong to this category. All known constructions of SIC-POVMs have arisen as the orbit of a nice error group or is equivalent to a SIC-POVM that is (e.g., Hoggar lines in 8 dimensions, elliptic curve construction in 3 dimensions). The aim of this chapter is develop a way to search for SIC-POVMs amongst the nice error groups. It has led to the natural generalisation of "nice error frames".

## 6.1 Background

**Definition 6.1.1.**

The **Pauli** matrices are

$$\sigma_1 = \sigma_x := \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}, \qquad \sigma_2 = \sigma_y := \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix}, \qquad \sigma_3 = \sigma_z := \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}. \quad (6.1.1)$$

The Pauli matrices are used to study spin in quantum mechanics. Together with the identity, they form an orthonormal basis for the $2 \times 2$ matrices.

**Definition 6.1.2.**

An **error operator basis** is an orthonormal basis for the $d \times d$ matrices $M_d(\mathbb{C}) = \mathbb{C}^{d \times d}$, with respect to the (Hilbert-Schmidt/Frobenius) inner product (see Definition 3.2.2).

A special class of the error operator basis called a "nice error basis" was defined by Knill in p. 3 of [Kni96b] and Section 2 of [Kni96a].

**Definition 6.1.3** (p. 3 [Kni96b])**.**
Let $G$ be a group of order $d^2$. Then unitary matrices $(E_g)_{g\in G}$ in $M_d(\mathbb{C})$ are a **nice** (**unitary**) **error basis** and $G$ is referred to as the **index group** if

1. $E_1$ is a scalar multiple of the identity $I$.
2. $E_g E_h = w_{g,h} E_{gh}, \forall g, h \in G$, where $w_{g,h} \in \mathbb{C}$.
3. $\mathrm{Tr}(E_g) = 0, g \neq 1, g \in G$,

   (i.e., they are an error operator basis.)

A group which gives rise to a nice error basis or nice error frame is called a **nice error group**.

In group theory terms (cf. Section 2 of [KR02], Section 2 of [Hig08]), this is equivalent to the map

$$\rho : g \mapsto E_g$$

being a *unitary irreducible special faithful projective representation of* $G$ *of degree* $d$.

A motivating example is the Pauli matrices indexed by $\mathbb{Z}_2 \times \mathbb{Z}_2$:

$$G = \mathbb{Z}_2 \times \mathbb{Z}_2 \mapsto M_2(\mathbb{C}) : (j,k) \mapsto E_{(j,k)} = S^j \Omega^k, \qquad S := \sigma_1, \Omega := \sigma_3.$$

Condition 3 of Definition 6.1.3 guarantees that a nice error basis yields the following orthogonal expansion

$$A = \frac{1}{d} \sum_{g\in G} \langle A, E_g \rangle E_g, \qquad \forall A \in M_d(\mathbb{C}). \tag{6.1.2}$$

The notion of a *nice error frame* is chosen so that the index group $G$ can have order greater than $d^2$, so that (6.1.2) becomes the tight frame expansion

$$A = \frac{d}{|G|} \sum_{g\in G} \langle A, E_g \rangle E_g, \qquad \forall A \in M_d(\mathbb{C}).$$

## 6.2 Nice error frames and canonical abstract error groups

Suppose $\mathcal{H} = M_d(\mathbb{C})$, and $(E_g)_{g\in G}$ are unitary matrices satisfying 1 and 2 of Definition 6.1.3, where $G$ is *any* finite group. These matrices have the equal norms

$$\|E_g\|^2 = \langle E_g, E_g\rangle = \mathrm{Tr}(E_g E_g^*) = \mathrm{Tr}(I) = d.$$

Since $w_{gh^{-1},h}E_gE_h^{-1} = E_{gh^{-1}}$, this means

$$\langle E_g, E_h\rangle = \mathrm{Tr}(E_g E_h^*) = \mathrm{Tr}(E_g E_h^{-1}) = \frac{1}{w_{gh^{-1},h}}\,\mathrm{Tr}(E_{gh^{-1}}).$$

By the variational characterisation (Theorem 2.1.14), the condition for $(E_g)_{g\in G}$ to be a tight frame for $M_d(\mathbb{C})$ reduces to

$$\sum_{g\in G}\sum_{h\in G}|\langle E_g, E_h\rangle|^2 = \sum_{g\in G}\sum_{h\in G}|\,\mathrm{Tr}(E_{gh^{-1}})|^2 = |G|\sum_{g\in G}|\,\mathrm{Tr}(E_g)|^2 = \frac{1}{d^2}\Big(|G|d\Big)^2. \quad (6.2.1)$$

**Definition 6.2.1.**
Let $G$ be a group (of order $\geq d^2$). Then unitary matrices $(E_g)_{g\in G}$ in $M_d(\mathbb{C})$ are a **nice** (**unitary**) **error frame** and $G$ is an **index group** if

1. $E_1$ is a scalar multiple of the identity $I$, and no other $E_g$ is.
2. $E_gE_h = w_{g,h}E_{gh}$, $\forall g, h \in G$, where $w_{g,h} \in \mathbb{C}$.
3. $\sum_{g\in G}|\,\mathrm{Tr}(E_g)|^2 = |G|$.

As observed in (6.2.1), a nice error frame is an equal–norm tight frame for $M_d(\mathbb{C})$, i.e.,

$$A = \frac{d}{|G|}\sum_{g\in G}\langle A, E_g\rangle E_g, \qquad \forall A \in M_d(\mathbb{C}). \quad (6.2.2)$$

Nice error frames generalise nice error bases, since condition 3 can be written as

$$\sum_{\substack{g\neq 1\\ g\in G}}|\,\mathrm{Tr}(E_g)|^2 = |G| - d^2.$$

Conditions 1 and 2 say that $g \mapsto E_g$ is a unitary faithful projective representation of $G$ of degree $d$. It is also *irreducible*, i.e., $\mathrm{span}\{E_g w\}_{g\in G} = \mathbb{C}^d$, $\forall w \neq 0$. To see this, expand the matrix $A = vw^*$, $v \in \mathbb{C}^d$, using (6.2.2) to obtain

$$vw^* = \frac{d}{|G|}\sum_{g\in G}\langle vw^*, E_g\rangle E_g \qquad \Longrightarrow \qquad v\|w\|^2 = \frac{d}{|G|}\sum_{g\in G}\langle vw^*, E_g\rangle E_g w.$$

These properties characterise nice error frames (Proposition 6.2.8).

In general, the matrices $(E_g)_{g\in G}$ of a nice error frame will not have finite order, and hence not generate a finite group. By finding the right scalings, these matrices can have finite order.

Let $\omega$ be the $d$–th root of unity

$$\omega := e^{\frac{2\pi i}{d}}.$$

**Remark 6.2.2.**
Since $\det(cA) = c^d \det(A)$, $c \in \mathbb{C}$, $A \in M_d(\mathbb{C})$, there are *exactly* $d$ scalings of a given $E_g$ which have determinant 1, i.e.,

$$\hat{E}_g = \frac{\omega^j}{\det(E_g)^{1/d}} E_g, \qquad j = 0, 1, \ldots, d-1,$$

where $\det(E_g)^{1/d}$ is any fixed $d$–th root of $\det(E_g)$. Knill calls a nice error basis *very* nice if each matrix has unit determinant (p. 2, [Kni96a]).

From now on, let $\hat{E}_g$ refer to any of these scalings, so that

$$\det(\hat{E}_g) = 1, \qquad \forall g \in G.$$

Then the $d|G|$ matrices

$$\{\omega^j \hat{E}_g : j = 0, \ldots, d-1, g \in G\}$$

are distinct. They form a group, since after scaling, condition 2 becomes

$$\hat{E}_g \hat{E}_h = \hat{w}_{g,h} \hat{E}_{gh}. \qquad \forall g, h \in G.$$

Taking the determinants of this gives

$$1 = \hat{w}_{g,h}^d \quad \Longrightarrow \quad \hat{w}_{g,h} \in \{1, \omega, \omega^2, \ldots \omega^{d-1}\}, \quad \forall g, h \in G.$$

**Definition 6.2.3.**

Let $(E_g)_{g\in G}$ be a nice error frame for $M_d(\mathbb{C})$. The associated **canonical error group** is

$$H := \{\omega^j \hat{E}_g : j = 0, \ldots, d-1,\ g \in G\},$$

and the abstract version of this group is called the **canonical abstract error group**.

The centre of a canonical error group $H$ is

$$Z(H) = \langle \omega I \rangle = \mathbb{Z}_d,$$

because if a matrix commutes with the spanning set $(E_g)_{g\in G}$ for $M_d(\mathbb{C})$, it commutes with all of $M_d(\mathbb{C})$, and is therefore a scalar matrix. Consequently, a group can be a canonical (abstract) error group for at most one dimension $d$. The index group $G$ of a canonical (abstract) error group $H$ is given by

$$G = H/Z(H).$$

Groups (abstract) will be labelled according to the "Small Groups Library". This labelling is used by the computer algebra systems MAGMA and GAP, e.g., the dihedral group of order $8$ is

$$D_4 = \texttt{SmallGroup(8,3)} = \texttt{<8,3>}.$$

**Example 6.2.4.**

The Pauli matrices $\{\sigma_1, \sigma_2, \sigma_3\}$ have determinant $-1$ and generate the group `<16,13>` of order 16. The canonical error group for the nice error basis $\{I, \sigma_1, \sigma_2, \sigma_3\}$ is

$$H = \langle i\sigma_1, i\sigma_2, i\sigma_3 \rangle,$$

the *quaternion group* `<8,4>`.

The earlier works of [KR02],[Kni96a] would have considered all the groups <16,13>, <8,3>, <8,4> to be *abstract error groups* or $w$*–coverings* of the nice error basis $\{I, \sigma_1, \sigma_2, \sigma_3\}$. Criteria will now be established for when nice error frames are considered "equivalent".

**Definition 6.2.5.**

Nice error frames $(E_g)_{g\in G}$ and $(F_h)_{h\in H}$ for $M_d(\mathbb{C})$ are **equivalent** if there is bijection

$$\sigma : G \to H$$

between their index groups, scalars $(c_g)_{g\in G}$ and an invertible $T \in M_d(\mathbb{C})$, such that

$$F_{\sigma g} = c_g T^{-1} E_g T, \qquad \forall g \in G. \tag{6.2.3}$$

This definition is more general than *projective equivalence*, where $G = H$ and reindexing the elements of $(E_g)_{g\in G}$ is not allowed.

**Proposition 6.2.6.**

Equivalent nice error frames have the *same* canonical abstract error group and the same index group.

*Proof.* Suppose the nice error frames $(E_g)_{g\in G}$ and $(F_h)_{h\in H}$ for $M_d(\mathbb{C})$ are equivalent. Then (6.2.3) scales to

$$\hat{F}_{\sigma g} = T^{-1}(\hat{c}_g E_g)T, \qquad \forall g \in G,$$

where $\hat{c}_g \in \{1, \omega, \omega^2, \ldots, \omega^{d-1}\}$ (by considering its determinants). Thus the canonical error groups are conjugate via $T$, and hence isomorphic. As the index group is the abstract error group factored by its centre, the nice error frames also have the same index groups.

□

**Example 6.2.7** (Heisenberg nice error basis)**.**

A nice error basis (projective representation) is given by

$$G = \mathbb{Z}_d \times \mathbb{Z}_d \mapsto M_d(\mathbb{C}) : (j,k) \mapsto E_{(j,k)} = S^j \Omega^k,$$

where $S$ is the cyclic shift matrix, and $\Omega$ is the modulation matrix, given by

$$(S)_{jk} := \delta_{j,k+1}, \qquad (\Omega)_{jk} = \omega^j \delta_{j,k}, \quad \omega := e^{\frac{2\pi i}{d}}.$$

This is the only nice error basis (up to equivalence) for $M_d(\mathbb{C})$ with index group $G = \mathbb{Z}_d \times \mathbb{Z}_d$ (cf. Theorem 3.5 of [BK73]).

It is now possible to show condition 3 of Definition 6.2.1 is equivalent to the projective representation $g \mapsto E_g$ being *irreducible*. Proposition 6.2.8 generalises Theorem 1 of [KR02], which is the case of a nice error basis.

**Proposition 6.2.8** (Characterisation)**.**
Let $G$ be a group and $(E_g)_{g\in G}$ be unitary matrices in $M_d(\mathbb{C})$. Then the following are equivalent

1. $(E_g)_{g\in G}$ is a nice error frame for $M_d(\mathbb{C})$.
2. $g \mapsto E_g$ is an irreducible unitary faithful projective representation of $G$ of degree $d$.

The action of the canonical error group $H$ on $\mathbb{C}^d$ is an irreducible unitary faithful ordinary representation of the canonical abstract error group of dimension $d$.

*Proof.* It suffices to show condition 2 implies condition 1. If $g \mapsto E_g$ is an irreducible unitary projective representation of $G$ on $\mathbb{C}^d$, then the canonical error group $H$ is defined (Definition 6.2.3). Its action on $\mathbb{C}^d$ (via multiplication) gives an *irreducible* unitary ordinary representation of $H$. The corresponding character $\chi$ is irreducible, hence its Euclidean inner product with itself is $|H| = d|G|$, giving

$$\langle \chi, \chi \rangle = \sum_{h\in H} |\operatorname{Tr}(h)|^2 = \sum_{j=0}^{d-1} \sum_{g\in G} |\operatorname{Tr}(\omega^j E_g)|^2 = d \sum_{g\in G} |\operatorname{Tr}(E_g)|^2 = d|G| = |H|.$$

which is condition 3 of Definition 6.2.1. Therefore conditions 1 and 2 are equivalent.

□

**Example 6.2.9.**
For $d = 1$, the only canonical abstract error group is $H = 1$.

**Example 6.2.10.**

For $d = 2$, a canonical error group is given by the **generalised quaternion group** or **dicyclic group** of order $4n$ ($n > 1$), generated by the matrices

$$\begin{pmatrix} \omega_{2n} & 0 \\ 0 & \omega_{2n}^{-1} \end{pmatrix}, \quad \begin{pmatrix} 0 & -1 \\ 1 & 0 \end{pmatrix}, \qquad \omega_{2n} := e^{\frac{2\pi i}{2n}}.$$

A MAGMA calculation shows that the only other canonical abstract error groups of order $\leq 200$ (with $d = 2$) are

$$H = \texttt{<24,3>},\ G = \texttt{<12,3>}, \qquad H = \texttt{<48,28>},\ G = \texttt{<24,12>}.$$

These are the canonical abstract error groups obtained from the Shephard–Todd reflection groups numbers $4$ and $8$, respectively.

**Example 6.2.11.**

Any irreducible group of $d \times d$ matrices which is finite after quotienting with its centre gives rise to a canonical abstract error group. By the N/C theorem, the normaliser of such a group in $GL_d(\mathbb{C})$ does too.

Take the Heisenberg group, generated by the matrices $S$ and $\Omega$ of Example 6.2.7, and its normaliser (the **Clifford group**). The canonical abstract error groups for the Heisenberg group and its normaliser are

$$\texttt{<8,4>}, \texttt{<48,28>} \quad (d = 2) \qquad \texttt{<27,3>}, \texttt{<648,532>} \quad (d = 3).$$

For a general $d$, the Clifford group gives a canonical abstract error group $H$, with order

$$|H| = d^6 \prod_{p|d} \Big(1 - \frac{1}{p^2}\Big), \qquad \text{(where } p \text{ is prime)}.$$

The subgroups of the Clifford group which contain the Heisenberg group give a family of nice error frames. Tensor products provide another avenue for constructing nice error frames.

**Proposition 6.2.12.**

Let $(E_{g_1})_{g_1 \in G_1}$,$(F_{g_2})_{g_2 \in G_2}$ be nice error frames for $M_{d_1}(\mathbb{C})$, $M_{d_2}(\mathbb{C})$ respectively. Their tensor

product

$$(E_{g_1} \otimes F_{g_2})_{(g_1,g_2)\in G_1\times G_2}$$

is a nice error frame for $M_{d_1 d_2}(\mathbb{C})$. In particular, a product of index groups is an index group. The canonical error group is

$$H = \{\omega^j(h_1 \otimes h_2) : 0 \le j < d-1, h_1 \in H_1, h_2 \in H_2\}, \qquad \omega := e^{\frac{2\pi i}{d}}, \quad d := d_1 d_2,$$

where $H_1$,$H_2$ are the canonical error groups of the nice error frames.

*Proof.* By Proposition 6.2.8, the first part follows from the theory of (projective) representations, or it can be verified directly. The tensor product satisfies condition 3 of Definition 6.2.1 since

$$\begin{aligned}\sum_{(g_1,g_2)\in G_1\times G_2} |\operatorname{Tr}(E_{g_1}\otimes F_{g_2})|^2 &= \sum_{g_1}\sum_{g_2} |\operatorname{Tr}(E_{g_1})\operatorname{Tr}(F_{g_2})|^2 \\ &= \Big(\sum_{g_1} |\operatorname{Tr}(E_{g_1})|^2\Big)\Big(\sum_{g_2} |\operatorname{Tr}(F_{g_2})|^2\Big) \\ &= |G|\,|H| = |G\times H|.\end{aligned}$$

The tensor product group $H_1 \otimes H_2$ consists of scalar multiples of each $E_{g_1} \otimes E_{g_2}$, with determinant 1. It might not contain all $d$–roots of unity (if $d_1$ and $d_2$ are not coprime), so these are added. □

**Example 6.2.13.**

The tensor product of the two nonabelian index groups for $d = 4$, with the (abelian) index group for $d = 2$, gives two nonabelian index groups for $d = 8$, i.e.,

$$<16,3> \times <4,2> = <64,193>, \qquad <16,11> \times <4,2> = <64,261>.$$

**Theorem 6.2.14** (Abelian index groups)**.**

A nice error frame has an abelian index group only if it is a nice error basis.

*Proof.* Let $H$ be the canonical abstract error group of a nice error frame, and $\chi : H \to \mathbb{C}$ be the character of a faithful irreducible representation of degree $\chi(1) = d$. Recall the **centre** of a character $\chi : H \to \mathbb{C}$ is the subgroup

$$Z(\chi) := \{h \in H : |\chi(h)| = \chi(1)\}.$$

If $\chi$ is irreducible,

$$Z(\chi)/\ker(\chi) = Z\Big(H/\ker(\chi)\Big), \qquad \ker(\chi) := \{h \in H : \chi(h) = \chi(1)\}.$$

As $\chi$ is irreducible, $\ker(\chi) = 1$,

$$Z(\chi) = Z(H),$$

so the index group is $G = H/Z(\chi)$ (Theorem 2.13 of [Isa94]). If $G$ is abelian, then

$$|G| = [H : Z(\chi)] = \chi(1)^2 = d^2.$$

□

**Remark 6.2.15.**

A canonical error group is an example of a *central group frame* (see Section 5 of [VW08]). This means

$$\Phi = \big(\rho(g)\big)_{h\in H}, \qquad \text{where } \rho : H \to SL_d(\mathbb{C}) : h \to \rho(h) \text{ is a represenation,}$$

is a tight frame for $M_d(\mathbb{C})$ satisfying the "symmetry condition"

$$\langle h\phi, g\phi\rangle = \langle h\psi, g\psi\rangle, \qquad \forall g, h \in H, \quad \forall \phi, \psi \in \Phi.$$

## 6.3 Calculations

Finding the centre of a group and its irreducible representations is fast. A representation can be made unitary by an appropriate conjugation. Proposition 6.3.1 utilises these facts to give a practical way for calculating abstract error groups.

**Proposition 6.3.1.**

A group $H$ is a canonical abstract error group if and only if

1. Its centre $Z(H)$ is cyclic of order $d$; and
2. it has a faithful irreducible ordinary representation $\rho$ of degree $d$, which is **special**, i.e., $\det(h) = 1, \forall h \in H$.

Consequently, all canonical abstract error groups for $d > 1$ are nonabelian.

**Remark 6.3.2.**

To find all canonical abstract error groups in the small groups catalog of a given order, just iterate through the list and the ones with a cyclic centre of order $d$ and have faithful irreducible ordinary representations of degree $d$ will correspond to canonical abstract error groups (Proposition 6.3.1).

Nice error frames given by the representations in Proposition 6.3.1 are of the form $(E_g)_{g\in G}$, where

$$G := H/Z(H), \qquad E_g \in \rho(g).$$

The next question considered is, which nice error frames are equivalent?

**Proposition 6.3.3** (Equivalence condition)**.**

If $\rho : H \to M_d(\mathbb{C})$ is a faithful irreducible special unitary ordinary representation of $H$, then so is

$$\rho_\sigma : h \mapsto \rho(\sigma h), \qquad \sigma \in \mathrm{Aut}(H),$$

where $\mathrm{Aut}(H)$ is the automorphism group of $H$. Furthermore, $\rho$ and $\rho_\sigma$ are equivalent nice error frames, even if the representations are not equivalent (e.g., if $\sigma$ is an outer automorphism).

*Proof.* An automorphism $\sigma$ of $H$ fixes the centre $Z(H)$ and induces an automorphism $\sigma_G \in \mathrm{Aut}(G)$ on the index group $G = H/Z(H)$. So a nice error frame $(F_g)_{g\in G}$ for $\rho_\sigma$ is just reindexing a nice error frame for $\rho$, since

$$F_g \in \rho_\sigma(g) = \rho(\sigma g) = \rho(\sigma_G(g)), \qquad \forall g \in G.$$

If $\sigma$ is an inner automorphism, i.e., $\sigma h = k^{-1}hk$, then $\rho$ and $\rho_\sigma$ are equivalent ordinary representations of $H$, since

$$\rho_\sigma(h) = \rho(k^{-1}hk)) = \rho(k)^{-1}\rho(h)\rho(k).$$

□

The ordinary representations of $H$ can be calculated in MAGMA with

```
AbsolutelyIrreducibleModules(H,Rationals());
```

### 6.3.1 Nice error bases and SIC-POVMs

All SIC-POVMs which have been constructed to date (see [SG10]) are $G$**–covariant** to a nice error basis $(E_g)_{g\in G}$ for $M_d(\mathbb{C})$, i.e., of the form

$$\Phi = (E_g v)_{g\in G}, \quad \text{for some fiducial vector } v.$$

Klappenecker and Rötteler found all the nonabelian index groups and some (incomplete list) of the nice error bases (up to equivalence as projective representations) for $d \leq 10$ See the *Catalogue of Nice Error Bases* at

`http://faculty.cs.tamu.edu/klappi/ueb/ueb.html`

Renes, et al in [RBKSC04] used this catalogue to construct $G$–covariant SIC-POVMs (to within $10^{-15}$) for $G = \mathbb{Z}_d \times \mathbb{Z}_d$ and for each $d = 6, 8, 9$ using a non abelian group. The non abelian groups were not tested exhaustively.

In the literature, analytic SIC-POVMs have been proved for

$$d = 2, 3, 4, \ldots, 16, 19, 24, 35, 48$$

by analytic constructions using the Weyl-Heisenberg nice error basis (index group $G = \mathbb{Z}_d \times \mathbb{Z}_d$).

**Theorem 6.3.4** (Variational characterisation)**.**
Let $(E_g)_{g\in G}$ be a nice error basis for $M_d(\mathbb{C})$, where $G$ is an abstract error group factored by its centre. Then the orbit $(E_g v)_{g\in G}$ of $v \in \mathbb{C}^d$ is a SIC-POVM in $\mathbb{C}^d$ if and only if

$$\sum_{g\in G} |\langle E_g v, v\rangle|^4 = 1 + \frac{d^2-1}{(d+1)^2} = \frac{2d}{d+1}, \qquad \|v\| = 1. \tag{6.3.1}$$

In Tables A.1,A.2 of Appendix A, all the canonical abstract error groups for nice error bases in dimensions $d < 14$ are listed. They were calculated using MAGMA by making use of the small groups library. The small groups library does not contain groups of order $14^3 = 2744$ calculations were not possible for dimensions $d \geq 14$. By pruning out groups without non trivial cyclic centres, it was relatively efficient to find all canonical error groups (irreducible faithful special representations) using a list of all groups of order $d^3$.

The tabulated canonical abstract error groups allowed for a comprehensive search for numerical SIC-POVMs using the variational approach of [RBKSC04] over all nice error bases in dimensions $d < 14$.

The following observations were made about the tabulated data.

**Example 6.3.5** (Inequivalent nice error bases)**.**
There exist nice error bases with same index group $G$ which are not equivalent. For $d = 8$, there are $47$ canonical abstract error groups, and only $42$ index groups. In particular, there are three canonical abstract error groups for $G = \langle 64, 67 \rangle$, and hence at least three inequivalent nice error bases with this index group.

Example 6.2.10 for $d = 2$ shows the general position, that for each $d \geq 2$ there is an infinite family of canonical abstract error groups. These can be constructed as *monomial* representations. A set of monomial matrices with nonzero entries given by $m$–th roots of unity ($m$ fixed) generates a finite group. The group can be enlarged to ensure its action on $\mathbb{C}^d$ is irreducible. In this way infinitely many canonical abstract error groups can be constructed for a given $d$.

Table A.3 of Appendix A, the first few canonical abstract error groups are listed as well as the index groups for nice error frames (which are not bases) when $2 \leq d \leq 7$.

**Example 6.3.6** (Repeated index groups)**.**
A group $G$ may be the index group for nice error frames in more than one dimension, e.g., $G = \langle 12, 3 \rangle$ is the index group for a nice error frame for $M_2(\mathbb{C})$ ($H = \langle 24, 3 \rangle$), and also one for $M_3(\mathbb{C})$ ($H = \langle 36, 11 \rangle$).

## 6.4 SIC-POVMs from nonabelian group in 6 dimensions

This section presents the calculations for a SIC-POVM obtained as the orbit of a nonabelian nice error basis (with index group `SmallGroup(36,11)` and canonical abstract error group `SmallGroup(216,42)`).

The calculations are presented to illustrate the recovery of a SIC-POVM from nice error groups and as a historical comparison for Chapter 3.

The index group `SmallGroup(36,11)` is isomorphic to $G = \mathbb{Z}_3 \times A_4$. One corresponding error group of $G$ is $H$, consisting of $6 \times 6$ matrices with blocks given by the Pauli matrices $\sigma_1$,

$\sigma_2, \sigma_3$ (see Definition 6.1.1). It has a fiducial vector of the form

$$v = \begin{pmatrix} \alpha r_0 \\ r_0 \tau^{63} \\ r_1 \xi_1 \\ \alpha r_1 \xi_1 \tau^{63} \\ r_2 \xi_2 \\ \alpha r_2 \xi_2 \tau^{45} \end{pmatrix}, \qquad \begin{gathered} \alpha \approx 0.5176, \\ r_0 \approx 0.6774, \\ r_1 \approx 0.3690, \\ r_2 \approx 0.4400, \\ \xi_1 \approx -0.9170 - 0.3988i, \ |\xi_1| = 1, \\ \xi_2 \approx 0.2044 + 0.9789i, \ |\xi_2| = 1, \\ \tau^9 = \tfrac{1+i}{\sqrt{2}}. \end{gathered} \tag{6.4.1}$$

This SIC-POVM was numerically observed first by Joe Renes (cf. final comments in Section 5 of [RBKSC04]).

The rest of the section is outlined as follows. The group $H$ is described with its associated nice error basis (with nonabelian index group) with discussion on the key features of the numerical SIC-POVMs obtained from it. Analytic solutions for $\alpha, r_0, r_1, r_2, \xi_1, \xi_2$ are derived (with some help from MAPLE) and $v$ (from (6.4.1)) is verified to be a SIC-POVM fiducial.

### 6.4.1 Nice error bases and numerical SIC-POVMs

Aside from `SmallGroup(36,11)` there is one other nonabelian index group for 6 dimensions. Namely, $G =$ `SmallGroup`$(36, 13)$. Extensive calculations suggest SIC-POVMs do not exist with this index group.

The nice error group used for $G$ in these calculations is a variation of one in the catalogue of nice error basis. Let

$$S^2 = \begin{pmatrix} 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \end{pmatrix}, \qquad \begin{pmatrix} 0 & \tau^{16} & 0 & 0 & 0 & 0 \\ \tau^{52} & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & \tau^{22} & 0 & 0 \\ 0 & 0 & \tau^{22} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \tau^{10} & 0 \\ 0 & 0 & 0 & 0 & 0 & \tau^{46} \end{pmatrix}, \qquad \tau := e^{2\pi i/72}.$$

The group generated by these matrices is `SmallGroup(216,39)`. $S^2$ does not have determinant 1.

Define matrices (with $2 \times 2$ blocks) by

$$B := \begin{pmatrix} i\sigma_1 & & \\ & i\sigma_2 & \\ & & i\sigma_3 \end{pmatrix}, \quad S^2 := \begin{pmatrix} 0 & 0 & I \\ I & 0 & 0 \\ 0 & I & 0 \end{pmatrix}, \quad A := \begin{pmatrix} I & & \\ & \omega I & \\ & & \omega^2 I \end{pmatrix}, \quad \omega := e^{2\pi i/3}. \tag{6.4.2}$$

where the Pauli matrices $\sigma_j$ of Definition 6.1.1 are normalised to have determinant 1.

**Proposition 6.4.1** (Nice error basis)**.**
Let $B, S^2, A$ be the matrices of (6.4.2), then

$$H := \langle B, S^2, A \rangle \subset SL_6(\mathbb{C}), \qquad |H| = 216 = 6^3,$$

gives a unitary faithful irreducible representation of `SmallGroup(216,42)`, with centre

$$Z(H) = \langle -\omega I \rangle, \qquad |Z(H)| = 6.$$

Taking a matrix $E_g$ from each coset of

$$G := H/Z(H) = \texttt{SmallGroup}(36, 11)$$

gives a nice error basis $(E_g)_{g \in G}$ for $M_6(\mathbb{C})$ with index group $G$.

Since

$$A^3 = (S^2)^3 = I, \quad B^2 = -I, \qquad AB = BA, \qquad AS^2 = \omega S^2 A,$$

$H$ has normal subgroups $H_1 = \langle A, -\omega I \rangle$, $H_2 := \langle B, S^2, \omega I \rangle$. So

$$H_1 \cap H_2 = Z(H), \qquad H_1/Z(H) = \mathbb{Z}_3, \qquad H_2/Z(H) = A_4,$$

and by the third isomorphism theorem,

$$G = \mathbb{Z}_3 \oplus A_4.$$

By using the variational characterisation, SIC-POVMs are found numerically by minimising the second frame potential and looking for the ones which meet the Welch bound exactly. This yielded 864 numerical solutions (up to scaling).

### 6.4.2 The analytic form of the SIC-POVM

The ratio of successive pairs of entries of solutions $v$ appeared (numerically) to take the form

$$\frac{v_1}{v_2}, \frac{v_3}{v_4}, \frac{v_5}{v_6} \in \left\{ \alpha^j \left( \frac{1+i}{\sqrt{2}} \right)^{1+2k} : j = \pm 1, k = 0, 1, 2, 3, \right\}. \tag{6.4.3}$$

The moduli of the entries are

$$\Big\{ \{|v_1|, |v_2|\}, \{|v_3|, |v_4|\}, \{|v_5|, |v_6|\} \Big\} = \Big\{ \{r_0, \alpha r_0\}, \{r_1, \alpha r_1\}, \{r_2, \alpha r_2\} \Big\}$$

and the signs of the ratios of entries from different pairs are

$$\left\{ \xi_1, \xi_2, \frac{1}{\xi_1}, \frac{1}{\xi_2}, \frac{\xi_1}{\xi_2}, \frac{\xi_2}{\xi_1} \right\} \{ \tau^j : 0 \leq j < 72 \}, \qquad \tau := e^{\frac{2\pi i}{72}},$$

where $\text{sign}(z) := z/|z|$.

This means all the found solutions can be constructed from $\alpha, r_0, r_1, r_2, \xi_1, \xi_2$ and $\tau$.

**Theorem 6.4.2.**

Let $(E_g)_{g \in G}$ be the nice error basis of Proposition 6.4.1 with the nonabelian index group $G := \texttt{SmallGroup}(36, 11)$. Then the unit vector

$$v := \begin{pmatrix} \alpha r_0 \\ r_0 \frac{1-i}{\sqrt{2}} \\ r_1 \xi_1 \\ \alpha r_1 \xi_1 \frac{1-i}{\sqrt{2}} \\ r_2 \xi_2 \\ \alpha r_2 \xi_2 \frac{-1-i}{\sqrt{2}} \end{pmatrix}, \tag{6.4.4}$$

where

$$\alpha := \frac{\sqrt{2}}{1+\sqrt{3}} = \frac{\sqrt{3-\sqrt{3}}}{\sqrt{3+\sqrt{3}}}, \qquad r_1 := \frac{1}{\sqrt{14}}\frac{\sqrt{7-\sqrt{21}}}{\sqrt{3-\sqrt{3}}},$$

$$r_0 := r_+, \quad r_2 := r_-, \qquad r_\pm := \frac{\sqrt{7+\sqrt{21} \pm \sqrt{14}\sqrt{\sqrt{21}-3}}}{2\sqrt{7}\sqrt{3-\sqrt{3}}},$$

$$\xi_1 = \tau^{50}\sqrt[3]{\beta - i\sqrt{1-\beta^2}}, \qquad \xi_2 = \frac{\tau^{31}}{4}\left(\sqrt{7}-\sqrt{3}-i\sqrt{6+2\sqrt{21}}\right), \tag{6.4.5}$$

$$\beta := -\frac{1}{8}\sqrt{46-6\sqrt{21}+6\sqrt{6\sqrt{21}-18}}, \qquad \tau := e^{\frac{2\pi i}{72}},$$

where $\sqrt[3]{\beta - i\sqrt{1-\beta^2}} \approx 0.680 - 0.728i$, gives a $G$–covariant SIC-POVM $(E_g v)_{g\in G}$ for $\mathbb{C}^6$.

*Proof.* The strategy will be to use the system of equations that define a SIC-POVM, impose the hypothesized structure of $v$ in (6.4.4) and show this more restricted system has a solution with the parameters given in the theorem. Motivated by the numerical fiducial vector (6.4.1), let $v$ have the form (6.4.4), where

$$\alpha, r_0, r_1, r_2 > 0, \qquad \xi_1, \xi_2 \in \mathbb{C}, \quad |\xi_1| = |\xi_2| = 1.$$

The condition that $v$ have unit norm is

$$(r_0^2 + r_1^2 + r_2^2)(1+\alpha^2) = 1. \tag{6.4.6}$$

Since $(E_g)_{g\in G}$ is a nice error basis,

$$|\langle E_g v, E_h v\rangle| = |\langle v, E_g^* E_h v\rangle| = |\langle v, E_g^{-1} E_h v\rangle| = |\langle v, E_{g^{-1}h} v\rangle|,$$

so the angle condition (Theorem 2.2.4) for $v$ to be a fiducial vector becomes

$$|\langle v, E_g v\rangle|^2 = \frac{1}{7}, \quad \forall g \in G,\ g \neq 1. \tag{6.4.7}$$

Of the $|G| - 1 = 35$ angle moduli equations, 17 are independent of $\xi_1$ and $\xi_2$, which further reduces to a system of eight equations after accounting for repeated and proportional equations. This system is symmetric in $r_0$ and $r_2$, and four of the equations factor, giving

$$r_0 r_2(\alpha^2 + 1) = \frac{1}{\sqrt{7}}, \tag{6.4.8}$$

$$(1 - \alpha^2)r_0^2 - \sqrt{2}\alpha r_1^2 + \sqrt{2}\alpha r_2^2 = \frac{1}{\sqrt{7}}, \tag{6.4.9}$$

$$\sqrt{2}\alpha r_0^2 - \sqrt{2}\alpha r_1^2 + (1 - \alpha^2)r_2^2 = \frac{1}{\sqrt{7}}, \tag{6.4.10}$$

$$\sqrt{2}\alpha r_0^2 - (1 - \alpha^2)r_1^2 + \sqrt{2}\alpha r_2^2 = \frac{1}{\sqrt{7}}. \tag{6.4.11}$$

From (6.4.6) and equations (6.4.8), (6.4.9), (6.4.10), (6.4.11), the $\alpha, r_0, r_1, r_2$ are recovered. These are used to verify that all eight equations and (6.4.6) hold.

It follows from $\alpha$ that

$$1 + \alpha^2 = 3 - \sqrt{3}, \qquad 1 - \alpha^2 = \sqrt{2}\alpha = \frac{1}{\sqrt{3}}(3 - \sqrt{3}).$$

Let $R_j := \sqrt{3 - \sqrt{3}}\, r_j$. Then the equations (6.4.6), (6.4.8), (6.4.9), (6.4.10) and (6.4.11) simplify to

$$(R_0^2 + R_2^2) + R_1^2 = 1, \qquad R_0 R_2 = \frac{1}{\sqrt{7}}, \qquad (R_0^2 + R_2^2) - R_1^2 = \frac{\sqrt{3}}{\sqrt{7}}.$$

The remaining 18 angle moduli equations reduce to three equations (repeated 6 times).

Using $1 + \alpha^2 i = 2\alpha\tau^3$ and $\alpha(1 - i) = \sqrt{2}\alpha\tau^{-9}$, these are reformulated as

$$p_j(\xi_1, \xi_2) := \left|2\alpha r_1 r_2 \tau^3 \omega^j \frac{\xi_1}{xi_2} + 2\alpha r_0 r_1 \omega^{-j} \frac{1}{\xi_1} + \sqrt{2}\alpha r_0 r_2 \tau^{-9} \xi_2\right|^2 = \frac{1}{7}, \qquad j = 0, 1, 2. \tag{6.4.12}$$

The equations (6.4.12) cannot be symmetrised (to find a simpler equation), since

$$\frac{1}{3}(p_0 + p_1 + p_2) = \frac{1}{7},$$

but the variational characterisation of tight frames (Proposition 6.3.4) does allow them to be replaced by single equation

$$\frac{1}{3}(p_0^2+p_1^2+p_2^2)=\Big(\frac{1}{7}\Big)^2. \tag{6.4.13}$$

While this equation doubles the degree of the three original equations, it has a simple form (with many zero terms), i.e.,

$$\frac{64r_0^2r_1^2r_2^2}{(\sqrt{3}+1)^4}\Big\{\sqrt{2}r_0r_1\Big(\frac{\xi_1^3}{\tau^6}+\frac{\tau^6}{\xi_1^3}\Big)+\sqrt{2}r_1r_2\Big(\frac{\tau^{15}\xi_1^3}{\xi_2^3}+\frac{\xi_2^3}{\tau^{15}xi_1^3}\Big)+r_0r_2\Big(\frac{\xi_2^3}{\tau^{21}}+\frac{\tau^{21}}{\xi_2^3}\Big)\Big\} \tag{6.4.14}$$
$$+\frac{33-4\sqrt{21}}{441}=\frac{1}{7^2}.$$

The constants $\tau^6,\tau^{15},\tau^{21}$ appear because

$$\tau^6=\frac{1}{2}\big(\sqrt{3}+i\big),\quad \tau^{15}=\frac{1}{2\sqrt{2}}\big(\sqrt{3}-1+(1+\sqrt{3})i\big),\quad \tau^{21}=\frac{1}{2\sqrt{2}}\big(1-\sqrt{3}+(1+\sqrt{3})i\big).$$

Splitting (6.4.14) into two parts:

$$\sqrt{2}r_0r_1\Big(\frac{\xi_1^3}{\tau^6}+\frac{\tau^6}{\xi_1^3}\Big)+\sqrt{2}r_1r_2\Big(\frac{\tau^{15}\xi_1^3}{\xi_2^3}+\frac{\xi_2^3}{\tau^{15}\xi_1^3}\Big)=\frac{\frac{2}{7}\sqrt{21}-2}{3-\sqrt{3}}, \tag{6.4.15}$$

$$r_0r_2\Big(\frac{\xi_2^3}{\tau^{21}}+\frac{\tau^{21}}{\xi_2^3}\Big)=\frac{\frac{1}{2}-\frac{3}{14}\sqrt{21}}{3-\sqrt{3}}. \tag{6.4.16}$$

Therefore it follows from

$$\frac{64r_0^2r_1^2r_2^2}{(\sqrt{3}+1)^4(3-\sqrt{3})}=\frac{14-2\sqrt{21}}{441},\qquad \frac{14-2\sqrt{21}}{441}\Big(\frac{2}{7}\sqrt{21}-2+\frac{1}{2}-\frac{3}{14}\sqrt{21}\Big)=\frac{4\sqrt{21}-24}{441},$$

that a solution of (6.4.15) and (6.4.16) is a solution of (6.4.14).

To complete the proof, it suffices to show $\xi_1$ and $\xi_2$ defined by (6.4.5) satisfy the equations (6.4.15) and (6.4.16). □

#### 6.4.2.1 The determination of $\xi_2$

The equation (6.4.16) is rewritten as

$$2\Re\left(\frac{\xi_2^3}{\tau^{21}}\right) = \frac{\xi_2^3}{\tau^{21}} + \frac{\tau^{21}}{\xi_2^3} = \frac{\sqrt{7}-3\sqrt{3}}{2}, \tag{6.4.17}$$

implying

$$\frac{\xi_2^3}{\tau^{21}} = \frac{\sqrt{7}-3\sqrt{3}}{4} + \frac{i}{4}\sqrt{6\sqrt{21}-18}.$$

To avoid taking a cube root, observe that the minimal polynomial of $z := \frac{\xi_2}{\tau^{31}}$ over $\mathbb{Q}$,

$$1 - x^2 - 3x^4 - x^6 + x^8 \qquad \text{(found with } \texttt{MinimalPolynomial()} \text{ in } \texttt{maple}\text{)},$$

has a factor

$$x^2 - \frac{\sqrt{7}-\sqrt{3}}{2}x + 1$$

with $z$ as a root, giving

$$\frac{\xi_2}{\tau^{31}} = \frac{\sqrt{7}-\sqrt{3}}{4} - \frac{i}{4}\sqrt{6+2\sqrt{21}}. \tag{6.4.18}$$

Define $\xi_2$ using (6.4.18). This choice of $\xi_2$ satisfies (6.4.16), since

$$\frac{\xi_2^3}{\tau^{21}} = \left(\frac{\xi_2}{\tau^{31}}\right)^3 = \left(\frac{\sqrt{7}-\sqrt{3}}{4} - \frac{i}{4}\sqrt{6+2\sqrt{21}}\right)^3 = \frac{\sqrt{7}-3\sqrt{3}}{4} + \frac{i}{4}\sqrt{6\sqrt{21}-18}.$$

### 6.4.3 The determination of $\xi_1$

The minimal polynomial of both $\frac{\xi_1^3}{\tau^6} + \frac{\tau^6}{\xi_1^3}$ and $\frac{\xi_1^3\tau^{15}}{\xi_2^3} + \frac{\xi_2^3}{\xi_1^3\tau^{15}}$ is

$$16x^8 - 184x^6 + 780x^4 - 1018x^2 + 1.$$

They are roots of the factor

$$4x^4 + (3\sqrt{21}-23)x^2 - 12\sqrt{21} + 55,$$

giving

$$\frac{\xi_1^3}{\tau^6}+\frac{\tau^6}{\xi_1^3}=-\frac{1}{4}\sqrt{46-6\sqrt{21}+6\sqrt{6\sqrt{21}-18}}, \tag{6.4.19}$$

$$\frac{\tau^{15}\xi_1^3}{\xi_2^3}+\frac{\xi_2^3}{\tau^{15}\xi_1^3}=-\frac{1}{4}\sqrt{46-6\sqrt{21}-6\sqrt{6\sqrt{21}-18}}. \tag{6.4.20}$$

Following from (6.4.19),

$$\frac{\xi_1^3}{\tau^6}=\beta-i\sqrt{1-\beta^2},\quad \xi_1=\tau^{50}\sqrt[3]{\beta-i\sqrt{1-\beta^2}},\qquad \beta:=-\frac{1}{8}\sqrt{46-6\sqrt{21}+6\sqrt{6\sqrt{21}-18}}.$$

Since $\xi_1$ and $\xi_2$ are defined as solutions of (6.4.19) and (6.4.17), these definitions need to be checked for consistency with (6.4.20).

This follows by the calculations

$$\begin{aligned}\frac{\tau^{15}\xi_1^3}{\xi_2^3}+\frac{\xi_2^3}{\tau^{15}\xi_1^3}=2\Re\Big(\frac{\xi_1^3}{\tau^6}\frac{\tau^{21}}{\xi_2^3}\Big)&=2\Re\Big\{(\beta-i\sqrt{1-\beta^2})\Big(\frac{\sqrt{7}-3\sqrt{3}}{4}-\frac{i}{4}\sqrt{6\sqrt{21}-18}\Big)\Big\}\\&=\beta\frac{\sqrt{7}-3\sqrt{3}}{2}-\frac{\sqrt{1-\beta^2}}{2}\sqrt{6\sqrt{21}-18}<0,\end{aligned}$$

$$\Big(\beta\frac{\sqrt{7}-3\sqrt{3}}{2}-\frac{\sqrt{1-\beta^2}}{2}\sqrt{6\sqrt{21}-18}\Big)^2=\frac{1}{4^2}(46-6\sqrt{21}-6\sqrt{6\sqrt{21}-18}).$$

Finally, to verify $\xi_1$ and $\xi_2$ satisfy (6.4.15), substitute

$$\sqrt{2}r_\pm r_1=\frac{\sqrt{28\pm\sqrt{14}(7-\sqrt{21})\sqrt{\sqrt{21}-3}}}{14(3-\sqrt{3})}$$

and (6.4.19), (6.4.20) into the LHS of (6.4.15) to get

$$\frac{-1}{56(3-\sqrt{3})}\Big\{\Big(56+4\sqrt{14}\sqrt{\sqrt{21}-3}-8\sqrt{21}\Big)+\Big(56-4\sqrt{14}\sqrt{\sqrt{21}-3}-8\sqrt{21}\Big)\Big\}=\frac{\frac{2}{7}\sqrt{21}-2}{3-\sqrt{3}},$$

as required.

**Remark 6.4.3.**

There are other relations between $\xi_1$ and $\xi_2$, e.g., the minimal polynomial of $\xi_1/\xi_2^{1/2}$ is

$$16x^{48} - 31x^{24} + 16,$$

leading to

$$\frac{\xi_1}{\xi_2^{1/2}} = \tau^{33} \sqrt[24]{\frac{31 - 3\sqrt{7}i}{32}},$$

but it is not clear how $\xi_1, \xi_2$ developed from this satisfy the angle equations.

## 6.5 Equivalence to Heisenberg SIC-POVMs

Many SIC-POVMs arise from nice error groups with a different index group to the Weyl-Heisenberg group. An interesting question is whether they are projectively equivalent to a known Weyl-Heisenberg solution. By using triple products, Zhu showed numerically in Section 10.4 of [Zhu12] that with the exception of the Hoggar lines, which occur with multiple instances in 8 dimensions, all other SIC-POVMs arising from different nice error bases are projectively equivalent to known Weyl-Heisenberg covariant SIC-POVMs. The analytic fiducials in Appendix A, and Section 6.4, which were independently discovered, can be used to verify Zhu's numerical claims.

As for the Hoggar lines, while they are not projectively equivalent to Weyl-Heisenberg SIC-POVMs, they do not stray far from the Weyl-Heisenberg tree. By looking to the normaliser of the Weyl-Heisenberg group (Clifford group), one finds canonical abstract error groups as subgroups, which give the Hoggar lines (see Section A.3 in Appendix A). In fact, several abstract error groups are realised as subgroups of the Clifford group. See Table A.2 of Appendix A.

This leads to the following conjecture.

**Conjecture 6.5.1.**

Let $\Phi$ be a SIC-POVM covariant to a group $G$. Then $\Phi$ is projectively unitarily equivalent to a SIC-POVM that is covariant to a subgroup of the Clifford group.

# Appendix A

# Nice Error Groups

## A.1 Tables of canonical abstract error groups and index groups

TABLE A.1: Nice error bases for $d < 14$, $d \neq 8$. Here $H$ is the canonical abstract error group, $G$ is the index group, and `sic` indicates that a SIC-POVM exists numerically.

| $d$ | $H$ | $G$ |
|---|---|---|
| 1 | $\langle 1, 1\rangle$ | $\langle 1, 1\rangle = \mathbb{Z}_1$ sic |
| 2 | $\langle 8, 4\rangle$ | $\langle 4, 2\rangle = \mathbb{Z}_2^2$ sic |
| 3 | $\langle 27, 3\rangle$ | $\langle 9, 2\rangle = \mathbb{Z}_3^2$ sic |
| 4 | $\langle 64, 19\rangle$ | $\langle 16, 2\rangle = \mathbb{Z}_4^2$ sic |
| | $\langle 64, 94\rangle$ | $\langle 16, 3\rangle$ |
| | $\langle 64, 256\rangle$ | $\langle 16, 11\rangle$ |
| | $\langle 64, 266\rangle$ | $\langle 16, 14\rangle = (\mathbb{Z}_2 \times \mathbb{Z}_2)^2$ |
| 5 | $\langle 125, 3\rangle$ | $\langle 25, 2\rangle = \mathbb{Z}_5^2$ sic |
| 6 | $\langle 216, 42\rangle$ | $\langle 36, 11\rangle$ sic |
| | $\langle 216, 66\rangle$ | $\langle 36, 13\rangle$ |
| | $\langle 216, 80\rangle$ | $\langle 36, 14\rangle = \mathbb{Z}_6^2$ sic |
| 7 | $\langle 343, 3\rangle$ | $\langle 49, 2\rangle = \mathbb{Z}_7^2$ sic |
| 9 | $\langle 729, 24\rangle$ | $\langle 81, 2\rangle = \mathbb{Z}_9^2$ sic |
| | $\langle 729, 30\rangle$ | $\langle 81, 4\rangle$ |
| | $\langle 729, 405\rangle$ | $\langle 81, 9\rangle$ sic |
| | $\langle 729, 489\rangle$ | $\langle 81, 12\rangle$ |
| | $\langle 729, 503\rangle$ | $\langle 81, 15\rangle = (\mathbb{Z}_3 \times \mathbb{Z}_3)^2$ |
| 10 | $\langle 1000, 70\rangle$ | $\langle 100, 15\rangle$ |
| | $\langle 1000, 84\rangle$ | $\langle 100, 16\rangle = \mathbb{Z}_{10}^2$ sic |
| 11 | $\langle 1331, 3\rangle$ | $\langle 121, 2\rangle = \mathbb{Z}_{11}^2$ sic |
| 12 | $\langle 1728, 1294\rangle$ | $\langle 144, 68\rangle$ sic |
| | $\langle 1728, 2011\rangle$ | $\langle 144, 92\rangle$ |
| | $\langle 1728, 2079\rangle$ | $\langle 144, 101\rangle = \mathbb{Z}_{12}^2$ sic |
| | $\langle 1728, 2983\rangle$ | $\langle 144, 132\rangle$ |
| | $\langle 1728, 10718\rangle$ | $\langle 144, 95\rangle$ |
| | $\langle 1728, 10926\rangle$ | $\langle 144, 100\rangle$ |
| | $\langle 1728, 11061\rangle$ | $\langle 144, 102\rangle$ |
| | $\langle 1728, 13457\rangle$ | $\langle 144, 136\rangle$ |
| | $\langle 1728, 20393\rangle$ | $\langle 144, 170\rangle$ |
| | $\langle 1728, 20436\rangle$ | $\langle 144, 172\rangle$ |
| | $\langle 1728, 20556\rangle$ | $\langle 144, 177\rangle$ |
| | $\langle 1728, 20771\rangle$ | $\langle 144, 179\rangle$ |
| | $\langle 1728, 30353\rangle$ | $\langle 144, 184\rangle$ |
| | $\langle 1728, 30562\rangle$ | $\langle 144, 189\rangle$ |
| | $\langle 1728, 30928\rangle$ | $\langle 144, 193\rangle$ |
| | $\langle 1728, 30953\rangle$ | $\langle 144, 194\rangle$ |
| | $\langle 1728, 31061\rangle$ | $\langle 144, 196\rangle$ |
| | $\langle 1728, 31093\rangle$ | $\langle 144, 197\rangle = (\mathbb{Z}_2 \times \mathbb{Z}_6)^2$ |
| 13 | $\langle 2197, 3\rangle$ | $\langle 169, 2\rangle = \mathbb{Z}_{13}^2$ sic |

TABLE A.2: The nice error bases for $d = 8$. Those which are subgroups of the Clifford group are labelled with an *. All SIC-POVMs are the Hoggar lines, except for $H = \langle 512, 451\rangle$, $G = Z_8^2$.

| $H$ | $G$ | $\langle 512, 400223\rangle$ | $\langle 64, 90\rangle$ sic |
|---|---|---|---|
| $\langle 512, 451\rangle$ | $\langle 64, 2\rangle = \mathbb{Z}_8^2$ sic* | $\langle 512, 400443\rangle$ | $\langle 64, 123\rangle$ * |
| $\langle 512, 452\rangle$ | $\langle 64, 3\rangle$ sic* | $\langle 512, 401215\rangle$ | $\langle 64, 91\rangle$ sic* |
| $\langle 512, 35969\rangle$ | $\langle 64, 8\rangle$ sic* | $\langle 512, 402896\rangle$ | $\langle 64, 128\rangle$* |
| $\langle 512, 36083\rangle$ | $\langle 64, 10\rangle$ | $\langle 512, 402951\rangle$ | $\langle 64, 138\rangle$ sic |
| $\langle 512, 59117\rangle$ | $\langle 64, 34\rangle$ * | $\langle 512, 402963\rangle$ | $\langle 64, 138\rangle$ |
| $\langle 512, 59133\rangle$ | $\langle 64, 35\rangle$ * | $\langle 512, 403139\rangle$ | $\langle 64, 162\rangle$ * |
| $\langle 512, 260804\rangle$ | $\langle 64, 58\rangle$ * | $\langle 512, 406850\rangle$ | $\langle 64, 174\rangle$ * |
| $\langle 512, 261506\rangle$ | $\langle 64, 67\rangle$ sic * | $\langle 512, 406879\rangle$ | $\langle 64, 167\rangle$ * |
| $\langle 512, 261511\rangle$ | $\langle 64, 67\rangle$ sic* | $\langle 512, 406902\rangle$ | $\langle 64, 179\rangle$ * |
| $\langle 512, 261518\rangle$ | $\langle 64, 67\rangle$ * | $\langle 512, 6276980\rangle$ | $\langle 64, 192\rangle = (\mathbb{Z}_2 \times \mathbb{Z}_4)^2$ * |
| $\langle 512, 262018\rangle$ | $\langle 64, 60\rangle$ sic* | $\langle 512, 6277027\rangle$ | $\langle 64, 193\rangle$ sic* |
| $\langle 512, 262052\rangle$ | $\langle 64, 62\rangle$ sic* | $\langle 512, 6278298\rangle$ | $\langle 64, 195\rangle$ sic* |
| $\langle 512, 265618\rangle$ | $\langle 64, 69\rangle$ sic* | $\langle 512, 6279917\rangle$ | $\langle 64, 202\rangle$ sic |
| $\langle 512, 265839\rangle$ | $\langle 64, 68\rangle$ sic* | $\langle 512, 6279938\rangle$ | $\langle 64, 202\rangle$ sic |
| $\langle 512, 265911\rangle$ | $\langle 64, 71\rangle$ sic* | $\langle 512, 6280116\rangle$ | $\langle 64, 203\rangle$ |
| $\langle 512, 266014\rangle$ | $\langle 64, 72\rangle$ * | $\langle 512, 6291080\rangle$ | $\langle 64, 226\rangle$ |
| $\langle 512, 266267\rangle$ | $\langle 64, 73\rangle$ | $\langle 512, 6339777\rangle$ | $\langle 64, 211\rangle$ |
| $\langle 512, 266357\rangle$ | $\langle 64, 75\rangle$ sic | $\langle 512, 6339869\rangle$ | $\langle 64, 207\rangle$ |
| $\langle 512, 266373\rangle$ | $\langle 64, 74\rangle$ sic | $\langle 512, 6375318\rangle$ | $\langle 64, 236\rangle$ |
| $\langle 512, 266477\rangle$ | $\langle 64, 78\rangle$ sic | $\langle 512, 6376278\rangle$ | $\langle 64, 216\rangle$ |
| $\langle 512, 266583\rangle$ | $\langle 64, 77\rangle$ sic | $\langle 512, 7421157\rangle$ | $\langle 64, 242\rangle$ |
| $\langle 512, 266616\rangle$ | $\langle 64, 82\rangle$ | $\langle 512, 10481364\rangle$ | $\langle 64, 261\rangle$ |
| $\langle 512, 400195\rangle$ | $\langle 64, 90\rangle$ sic* | $\langle 512, 10494180\rangle$ | $\langle 64, 267\rangle = (\mathbb{Z}_2^3)^2$ sic |

TABLE A.3: The canonical abstract error groups and index groups for the first few nice error frames, which are not bases, for $2 \leq d \leq 7$.

| $d = 2$ | | $d = 3$ | | $d = 4$ | |
|---|---|---|---|---|---|
| $\langle 12, 1\rangle$ | $\langle 6, 1\rangle$ | $\langle 36, 11\rangle$ | $\langle 12, 3\rangle$ | $\langle 80, 28\rangle$ | $\langle 20, 3\rangle$ |
| $\langle 16, 9\rangle$ | $\langle 8, 3\rangle$ | $\langle 54, 8\rangle$ | $\langle 18, 4\rangle$ | $\langle 96, 157\rangle$ | $\langle 24, 8\rangle$ |
| $\langle 20, 1\rangle$ | $\langle 10, 1\rangle$ | $\langle 63, 3\rangle$ | $\langle 21, 1\rangle$ | $\langle 96, 215\rangle$ | $\langle 24, 14\rangle$ |
| $\langle 24, 3\rangle$ | $\langle 12, 3\rangle$ | $\langle 72, 42\rangle$ | $\langle 24, 12\rangle$ | $\langle 128, 523\rangle$ | $\langle 32, 27\rangle$ |
| $\langle 24, 4\rangle$ | $\langle 12, 4\rangle$ | $\langle 81, 9\rangle$ | $\langle 27, 3\rangle$ | $\langle 128, 545\rangle$ | $\langle 32, 24\rangle$ |
| $\langle 28, 1\rangle$ | $\langle 14, 1\rangle$ | $\langle 108, 15\rangle$ | $\langle 36, 9\rangle$ | $\langle 128, 749\rangle$ | $\langle 32, 34\rangle$ |
| $\langle 32, 20\rangle$ | $\langle 16, 7\rangle$ | $\langle 108, 22\rangle$ | $\langle 36, 11\rangle$ | $\langle 128, 782\rangle$ | $\langle 32, 31\rangle$ |
| $\langle 36, 1\rangle$ | $\langle 18, 1\rangle$ | $\langle 117, 3\rangle$ | $\langle 39, 1\rangle$ | $\langle 128, 864\rangle$ | $\langle 32, 6\rangle$ |
| $\langle 40, 4\rangle$ | $\langle 20, 4\rangle$ | $\langle 144, 68\rangle$ | $\langle 48, 3\rangle$ | $\langle 128, 880\rangle$ | $\langle 32, 9\rangle$ |
| $\langle 44, 1\rangle$ | $\langle 22, 1\rangle$ | $\langle 162, 14\rangle$ | $\langle 54, 5\rangle$ | $\langle 128, 1750\rangle$ | $\langle 32, 27\rangle$ |
| $\langle 48, 8\rangle$ | $\langle 24, 6\rangle$ | $\langle 171, 4\rangle$ | $\langle 57, 1\rangle$ | $\langle 128, 1799\rangle$ | $\langle 32, 28\rangle$ |
| $\langle 48, 28\rangle$ | $\langle 24, 12\rangle$ | $\langle 189, 8\rangle$ | $\langle 63, 3\rangle$ | $\langle 128, 2146\rangle$ | $\langle 32, 39\rangle$ |

| $d = 5$ | | $d = 6$ | | $d = 7$ | |
|---|---|---|---|---|---|
| $\langle 250, 8\rangle$ | $\langle 50, 4\rangle$ | $\langle 252, 16\rangle$ | $\langle 42, 1\rangle$ | $\langle 392, 39\rangle$ | $\langle 56, 11\rangle$ |
| $\langle 275, 3\rangle$ | $\langle 55, 1\rangle$ | $\langle 288, 230\rangle$ | $\langle 48, 3\rangle$ | $\langle 686, 8\rangle$ | $\langle 98, 4\rangle$ |
| $\langle 375, 2\rangle$ | $\langle 75, 2\rangle$ | $\langle 288, 896\rangle$ | $\langle 48, 48\rangle$ | $\langle 1029, 12\rangle$ | $\langle 147, 5\rangle$ |
| $\langle 400, 213\rangle$ | $\langle 80, 49\rangle$ | $\langle 288, 982\rangle$ | $\langle 48, 49\rangle$ | $\langle 1176, 219\rangle$ | $\langle 168, 43\rangle$ |
| $\langle 500, 25\rangle$ | $\langle 100, 12\rangle$ | $\langle 288, 986\rangle$ | $\langle 48, 50\rangle$ | $\langle 1372, 14\rangle$ | $\langle 196, 8\rangle$ |
| $\langle 625, 7\rangle$ | $\langle 125, 3\rangle$ | $\langle 324, 20\rangle$ | $\langle 54, 7\rangle$ | | |

## A.2 Analytic Solutions For Dimension 8

**Theorem A.2.1.**

An abstract error group $H$ with abelian index group $G \simeq H/Z(H)$ is nilpotent of class at most 2.

**Theorem A.2.2** (Lemma 1, [Bro73])**.**

Let $G$ be a finite nilpotent group. Then every irreducible $\mathbb{C}$–representation of $G$ is a monomial representation.

All of the canonical abstract error groups in 8 dimensions are of order 512 (finite $p$–group) and are hence nilpotent groups. By Theorem A.2.2, their irreducible representations in $\mathbb{C}$ can be monomial. This simplifies the angle equations significantly and makes them easier to solve (see also [ABB+12]).

**Remark A.2.3.**
In 6 dimensions, some of the abstract error groups were not nilpotent, but still admitted monomial representations.

Analytic fiducials for SIC-POVMs arising from canonical abstract error groups in 8 dimensions will now be presented along with their error groups. The tedious calculations are omitted.

For the following matrices, let $\zeta_k = e^{2\pi i/k}$.

### A.2.1 One Canonical Abstract Error Group

The groups presented here have a unique canonical abstract error groups for their associated index group. They are organised into their index groups.

#### A.2.1.1 SmallGroup(64, 3)

**Canonical Abstract Error Group:** `SmallGroup(512,452)`. Let

$$\sigma_1 = \begin{bmatrix} 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ -\zeta_4 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -\zeta_4 & 0 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}, \sigma_2 = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ -\zeta_8^3 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & \zeta_8^3 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & -\zeta_8^3 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & -\zeta_8^3 \\ 0 & 0 & 0 & 0 & -\zeta_4 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & \zeta_4 & 0 & 0 \end{bmatrix}.$$

Then $\{\sigma_i\}$ is a minimal set of generators for this group.

Let $s = \frac{\sqrt{3+3\sqrt{5}}}{6}$, $b = \frac{\sqrt{3-\sqrt{5}}}{2\sqrt{6}}$, $c = \frac{\sqrt{12+6\sqrt{-2+2\sqrt{5}}}}{12}$, $g = \frac{\sqrt{12-6\sqrt{-2+2\sqrt{5}}}}{12}$.

$$v = \left(0, 0, sI, -b(1+I), c - gI, b + \frac{s}{\sqrt{2}}I, \sqrt{2}c, g(1+I)\right)^T$$

is a fiducial vector giving rise to a SIC-POVM.

#### A.2.1.2 SmallGroup(64, 8)

**Canonical Abstract Error Group:** `SmallGroup(512,35969).`

Let $\sigma_1 = \zeta_8 I_8$, and

$$\sigma_2 = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ \zeta_4 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & \zeta_4 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & \zeta_4 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \zeta_4 & 0 & 0 & 0 \end{bmatrix}, \sigma_3 = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & -\zeta_8^3 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & \zeta_8 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & \zeta_8^3 \\ 0 & -\zeta_8 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}.$$

Then $\{\sigma_i\}$ is a set of generators for this group.

Let $\alpha = \frac{\sqrt{6+3\sqrt{2}}}{6}$, $\beta = \frac{\sqrt{6-3\sqrt{2}}}{6}$.

$$v = \left(-\beta \boldsymbol{i}, 0, -\alpha \boldsymbol{i}, \alpha, \beta, \frac{1}{2}\sqrt{2}\alpha + \frac{1}{2}\sqrt{2}\beta \boldsymbol{i}, 0, \frac{1}{2}\sqrt{2}\alpha + \frac{1}{2}\sqrt{2}\beta \boldsymbol{i}\right)^T$$

is a fiducial vector giving rise to a SIC-POVM.

#### A.2.1.3 SmallGroup(64, 60)

**Canonical Abstract Error Group:** `SmallGroup(512,262018).`

Let $\sigma_1 = \zeta_8 I_8$, and

$$\sigma_2 = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \end{bmatrix}; \sigma_3 = \begin{bmatrix} 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & \zeta_4 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & \zeta_4 \\ \zeta_4 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & \zeta_4 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \end{bmatrix};$$

$$\sigma_4 = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix};$$

Then $\{\sigma_i\}$ is a set of generators for this group.

$$v = \left(0, 0, 0, \frac{\sqrt{3}}{3}, -\frac{\sqrt{3}}{6} + \frac{\sqrt{3}}{6}\boldsymbol{i}, -\frac{\sqrt{3}}{6} - \frac{\sqrt{3}}{6}\boldsymbol{i}, \frac{\sqrt{6}}{6}, \frac{\sqrt{6}}{6}\right)^T.$$

is a fiducial vector giving rise to a SIC-POVM.

#### A.2.1.4 SmallGroup(64, 62)

**Canonical Abstract Error Group:** `SmallGroup(512, 262052).`

Let $\sigma_1 = \zeta_8 I_8$, and

$$\sigma_2 = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \end{bmatrix}; \sigma_3 = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ \zeta_4 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & \zeta_4 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & \zeta_4 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & \zeta_4 & 0 \end{bmatrix};$$

$$\sigma_4 = \begin{bmatrix} 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \end{bmatrix};$$

Then $\{\sigma_i\}$ is a set of generators for this group.

$$v = \left(0, 0, -\frac{\sqrt{6}}{6}, \frac{\sqrt{3}}{6} - \frac{\sqrt{3}}{6}\boldsymbol{i}, 0, \frac{\sqrt{3}}{3}, \frac{\sqrt{6}}{6}, \frac{\sqrt{3}}{6} + \frac{\sqrt{3}}{6}\boldsymbol{i}\right)^T.$$

is a fiducial vector giving rise to a SIC-POVM.

#### A.2.1.5 SmallGroup(64, 68)

**Canonical Abstract Error Group:** `SmallGroup(512, 265839).`

Let $\sigma_1 = \zeta_8 I_8$, and

$$\sigma_2 = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \end{bmatrix}; \sigma_3 = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & \zeta_4 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & -\zeta_4 & 0 \\ 0 & -\zeta_4 & 0 & 0 & 0 & 0 & 0 & 0 \\ -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 \\ 0 & 0 & \zeta_4 & 0 & 0 & 0 & 0 & 0 \end{bmatrix};$$

$$\sigma_4 = \begin{bmatrix} 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & -\zeta_4 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & -\zeta_4 \\ -\zeta_4 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & -\zeta_4 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 \end{bmatrix};$$

Then $\{\sigma_i\}$ is a set of generators for this group.

$$v = \left( -\frac{\sqrt{6}}{6}\boldsymbol{i}, 0, 0, \frac{\sqrt{3}}{6} - \frac{\sqrt{3}}{6}\boldsymbol{i}, \frac{\sqrt{3}}{3}, -\frac{\sqrt{6}}{6}\boldsymbol{i}, -\frac{\sqrt{3}}{6} - \frac{\sqrt{3}}{6}\boldsymbol{i}, 0 \right)^T.$$

is a fiducial vector giving rise to a SIC-POVM.

#### A.2.1.6 SmallGroup(64, 69)

**Canonical Abstract Error Group:** `SmallGroup(512,265618).`

Let $\sigma_1 = \zeta_8 I_8$, and

$$\sigma_2 = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \end{bmatrix}; \sigma_3 = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ -\zeta_4 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -\zeta_4 & 0 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & \zeta_4 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & -\zeta_4 & 0 \end{bmatrix};$$

$$\sigma_4 = \begin{bmatrix} 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & \zeta_4 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & \zeta_4 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & -\zeta_4 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -\zeta_4 & 0 & 0 & 0 & 0 \end{bmatrix};$$

Then $\{\sigma_i\}$ is a set of generators for this group.

$$v = \left(0, -\frac{\sqrt{6}}{6}, 0, \frac{\sqrt{3}}{6} - \frac{\sqrt{3}}{6}\boldsymbol{i}, -\frac{\sqrt{3}}{6} + \frac{\sqrt{3}}{6}\boldsymbol{i}, 0, \frac{\sqrt{6}}{6}\boldsymbol{i}, \frac{\sqrt{3}}{3}\right)^T.$$

is a fiducial vector giving rise to a SIC-POVM.

#### A.2.1.7 SmallGroup(64, 71)

**Canonical Abstract Error Group:** `SmallGroup(512, 265911).`

Let $\sigma_1 = \zeta_8 I_8$, and

$$\sigma_2 = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \end{bmatrix}; \sigma_3 = \begin{bmatrix} 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & \zeta_4 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & \zeta_4 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & -\zeta_4 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -\zeta_4 & 0 & 0 & 0 & 0 \end{bmatrix};$$

$$\sigma_4 = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix};$$

Then $\{\sigma_i\}$ is a set of generators for this group.

$$v = \left(0, 0, 0, \frac{\sqrt{3}}{3}, \frac{\sqrt{6}}{6}\boldsymbol{i}, \frac{\sqrt{6}}{6}\boldsymbol{i}, \frac{\sqrt{3}}{6} + \frac{\sqrt{3}}{6}\boldsymbol{i}, \frac{\sqrt{3}}{6} - \frac{\sqrt{3}}{6}\boldsymbol{i}\right)^T.$$

is a fiducial vector giving rise to a SIC-POVM.

#### A.2.1.8 SmallGroup(64, 74)

**Canonical Abstract Error Group:** `SmallGroup(512, 266373).`

Let $\sigma_1 = \zeta_8 I_8$, and

$$\sigma_2 = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ -\zeta_4 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & -\zeta_4 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & \zeta_4 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & \zeta_4 & 0 \end{bmatrix}; \sigma_3 = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 \\ -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \end{bmatrix};$$

$$\sigma_4 = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \end{bmatrix};$$

Then $\{\sigma_i\}$ is a set of generators for this group.

$$v = \left(0, 0, \frac{\sqrt{3}}{3}, \frac{\sqrt{3}}{6}, \frac{\sqrt{3}}{6}\boldsymbol{i}, 0, \frac{\sqrt{6}}{6}, \frac{\sqrt{3}}{6}, -\frac{\sqrt{3}}{6}\boldsymbol{i}, \frac{\sqrt{6}}{6}\right)^T.$$

is a fiducial vector giving rise to a SIC-POVM.

#### A.2.1.9 SmallGroup(64, 75)

**Canonical Abstract Error Group:** `SmallGroup(512,266357).`

Let $\sigma_1 = \zeta_8 I_8$, and

$$\sigma_2 = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ \zeta_4 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & \zeta_4 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & \zeta_4 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & \zeta_4 & 0 \end{bmatrix}; \sigma_3 = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \zeta_4 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & -\zeta_4 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & \zeta_4 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & -\zeta_4 & 0 & 0 \end{bmatrix};$$

$$\sigma_4 = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & -\zeta_4 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ -\zeta_4 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & -\zeta_4 \\ 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & \zeta_4 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \end{bmatrix};$$

Then $\{\sigma_i\}$ is a set of generators for this group.

$$v = \left( -\frac{\sqrt{6}}{6}\boldsymbol{i}, 0, -\frac{\sqrt{6}}{6}\boldsymbol{i}, 0, 0, -\frac{\sqrt{3}}{6} - \frac{\sqrt{3}}{6}\boldsymbol{i}, \frac{\sqrt{3}}{3}, -\frac{\sqrt{3}}{6} + \frac{\sqrt{3}}{6}\boldsymbol{i} \right)^T.$$

is a fiducial vector giving rise to a SIC-POVM.

#### A.2.1.10 SmallGroup(64, 77)

**Canonical Abstract Error Group:** `SmallGroup(512, 266583).`

Let $\sigma_1 = \zeta_8 I_8$, and

$$\sigma_2 = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 \end{bmatrix}; \sigma_3 = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & -\zeta_4 & 0 & 0 & 0 \\ -\zeta_4 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & \zeta_4 \\ 0 & 0 & 0 & \zeta_4 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 \end{bmatrix};$$

$$\sigma_4 = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \end{bmatrix};$$

Then $\{\sigma_i\}$ is a set of generators for this group.

$$v = \left( \frac{\sqrt{6}}{6}, 0, 0, -\frac{\sqrt{6}}{6}\boldsymbol{i}, -\frac{\sqrt{3}}{6} + \frac{\sqrt{3}}{6}\boldsymbol{i}, 0, \frac{\sqrt{3}}{3}, \frac{\sqrt{3}}{6} - \frac{\sqrt{3}}{6}\boldsymbol{i} \right)^T.$$

is a fiducial vector giving rise to a SIC-POVM.

#### A.2.1.11 SmallGroup(64, 78)

**Canonical Abstract Error Group:** `SmallGroup(512, 266477).`

Let $\sigma_1 = \zeta_8 I_8$, and

$$\sigma_2 = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ \zeta_4 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & \zeta_4 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & \zeta_4 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & \zeta_4 & 0 \end{bmatrix}; \sigma_3 = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \zeta_4 & 0 & 0 & 0 \\ -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & \zeta_4 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & \zeta_4 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & -\zeta_4 & 0 & 0 \end{bmatrix};$$

$$\sigma_4 = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 \\ -\zeta_4 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 \\ 0 & \zeta_4 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & -\zeta_4 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \zeta_4 & 0 & 0 & 0 \end{bmatrix};$$

Then $\{\sigma_i\}$ is a set of generators for this group.

$$v = \left(0, \frac{\sqrt{6}}{6}\boldsymbol{i}, -\frac{\sqrt{3}}{6} - \frac{\sqrt{3}}{6}\boldsymbol{i}, 0, 0, \frac{\sqrt{3}}{6} + \frac{\sqrt{3}}{6}\boldsymbol{i}, -\frac{\sqrt{6}}{6}\boldsymbol{i}, \frac{\sqrt{3}}{3}\right)^T.$$

is a fiducial vector giving rise to a SIC-POVM.

#### A.2.1.12 SmallGroup(64, 91)

**Canonical Abstract Error Group:** `SmallGroup(512, 401215)`.

Let $\sigma_1 = \zeta_8 I_8$, and

$$\sigma_2 = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}; \sigma_3 = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -\zeta_4 & 0 & 0 & 0 & 0 \\ 0 & 0 & \zeta_4 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & \zeta_4 \\ 0 & 0 & 0 & 0 & 0 & 0 & -\zeta_4 & 0 \end{bmatrix};$$

$$\sigma_4 = \begin{bmatrix} -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 \end{bmatrix};$$

Then $\{\sigma_i\}$ is a set of generators for this group.

$$v = \left(0, 0, \frac{\sqrt{6}}{6}\boldsymbol{i}, -\frac{\sqrt{6}}{6}, 0, \frac{\sqrt{3}}{3}, -\frac{\sqrt{6}}{6}\boldsymbol{i}, \frac{\sqrt{6}}{6}\boldsymbol{i}\right)^T.$$

is a fiducial vector giving rise to a SIC-POVM.

#### A.2.1.13 SmallGroup(64, 193)

**Canonical Abstract Error Group:** `SmallGroup(512, 6277027)`.

Let $\sigma_1 = \zeta_8 I_8$, and

$$\sigma_2 = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ \zeta_4 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & \zeta_4 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & \zeta_4 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & \zeta_4 & 0 & 0 \end{bmatrix}; \sigma_3 = \begin{bmatrix} 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \end{bmatrix};$$

$$\sigma_4 = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix}; \sigma_5 = \begin{bmatrix} -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix};$$

Then $\{\sigma_i\}$ is a set of generators for this group.

$$v = \left(0, 0, \frac{\sqrt{6}}{6}, -\frac{\sqrt{6}}{6}\boldsymbol{i}, 0, \frac{\sqrt{3}}{3}, \frac{\sqrt{6}}{6}, -\frac{\sqrt{6}}{6}\right)^T.$$

is a fiducial vector giving rise to a SIC-POVM.

#### A.2.1.14 SmallGroup(64, 195)

**Canonical Abstract Error Group:** `SmallGroup(512, 6278298)`.

Let $\sigma_1 = \zeta_8 I_8$, and

$$\sigma_2 = \begin{bmatrix} 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 \end{bmatrix}; \sigma_3 = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & -\zeta_4 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & -\zeta_4 \\ 0 & 0 & 0 & 0 & \zeta_4 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & \zeta_4 & 0 & 0 \end{bmatrix};$$

$$\sigma_4 = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix}; \sigma_5 = \begin{bmatrix} -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix};$$

Then $\{\sigma_i\}$ is a set of generators for this group.

$$v = \left(0, 0, \frac{\sqrt{6}}{6}, -\frac{\sqrt{6}}{6}\boldsymbol{i}, 0, \frac{\sqrt{3}}{3}, \frac{\sqrt{3}}{6}, \frac{\sqrt{3}}{6} + \frac{\sqrt{3}}{6}\boldsymbol{i}, -\frac{\sqrt{3}}{6} - \frac{\sqrt{3}}{6}\boldsymbol{i}\right)^T.$$

is a fiducial vector giving rise to a SIC-POVM.

### A.2.2 Two or More Canonical Abstract Error Groups

#### A.2.2.1 SmallGroup(64, 90)

**Canonical Abstract Error Group:** `SmallGroup(512, 400195).`

Let $\sigma_1 = \zeta_8 I_8$, and

$$\sigma_2 = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ \zeta_4 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 \\ 0 & 0 & -\zeta_4 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 0 & -\zeta_4 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 \\ 0 & 0 & 0 & 0 & 0 & 0 & -\zeta_4 & 0 \end{bmatrix}; \sigma_3 = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix};$$

$$\sigma_4 = \begin{bmatrix} -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix};$$

Then $\{\sigma_i\}$ is a set of generators for this group.

$$v = \left(0, -\frac{\sqrt{6}}{6}\boldsymbol{i}, 0, \frac{\sqrt{6}}{6}, 0, \frac{\sqrt{6}}{6}\boldsymbol{i}, \frac{\sqrt{3}}{3}, -\frac{\sqrt{6}}{6}\boldsymbol{i}\right)^T.$$

is a fiducial vector giving rise to a SIC-POVM.

**Canonical Abstract Error Group:** `SmallGroup(512, 400223)`.

Let $\sigma_1 = \zeta_8 I_8$, and

$$\sigma_2 = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & \zeta_4 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & \zeta_4 \\ 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ \zeta_4 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -\zeta_4 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}; \sigma_3 = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix};$$

$$\sigma_4 = \begin{bmatrix} -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix};$$

Then $\{\sigma_i\}$ is a set of generators for this group.

$$v = \left(0, -\frac{\sqrt{6}}{6}\boldsymbol{i}, 0, \frac{\sqrt{6}}{6}, 0, \frac{\sqrt{6}}{6}, \frac{\sqrt{3}}{3}, \frac{\sqrt{6}}{6}\right)^T.$$

is a fiducial vector giving rise to a SIC-POVM.

#### A.2.2.2 SmallGroup(64, 202)

**Canonical Abstract Error Group:** `SmallGroup(512, 6279917)`.

Let $\sigma_1 = \zeta_8 I_8$, and

$$\sigma_2 = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & -\zeta_4 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & -\zeta_4 \\ 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 \\ 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\ \zeta_4 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -\zeta_4 & 0 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}; \sigma_3 = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \end{bmatrix};$$

$$\sigma_4 = \begin{bmatrix} 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \end{bmatrix}; \sigma_5 = \begin{bmatrix} -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix};$$

Then $\{\sigma_i\}$ is a set of generators for this group.

$$v = \left( \frac{\sqrt{6}}{6}, \frac{\sqrt{3}}{3}, 0, -\frac{\sqrt{3}}{6} + \frac{\sqrt{3}}{6}\boldsymbol{i}, \frac{\sqrt{6}}{6}, 0, 0, -\frac{\sqrt{3}}{6} - \frac{\sqrt{3}}{6}\boldsymbol{i} \right)^T.$$

is a fiducial vector giving rise to a SIC-POVM.

**Canonical Abstract Error Group:** `SmallGroup(512, 6279938)`.

Let $\sigma_1 = \zeta_8 I_8$, and

$$\sigma_2 = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ \zeta_4 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 \\ 0 & 0 & -\zeta_4 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 0 & -\zeta_4 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 \\ 0 & 0 & 0 & 0 & 0 & 0 & -\zeta_4 & 0 \end{bmatrix}; \sigma_3 = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \end{bmatrix};$$

$$\sigma_4 = \begin{bmatrix} 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \end{bmatrix}; \sigma_5 = \begin{bmatrix} -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix};$$

Then $\{\sigma_i\}$ is a set of generators for this group.

$$v = \left(0, 0, 0, \frac{\sqrt{3}}{3}, \frac{\sqrt{6}}{6}, \frac{\sqrt{3}}{6} - \frac{\sqrt{3}}{6}\boldsymbol{i}, -\frac{\sqrt{6}}{6}, -\frac{\sqrt{3}}{6} - \frac{\sqrt{3}}{6}\boldsymbol{i}\right)^T.$$

is a fiducial vector giving rise to a SIC-POVM.

### A.2.3 Canonical Abstract Error Group With No Solutions

The fiducial searching method used was to start with a random vector and perturb it in ways that attempt to minimise the second frame potential. The number of perturbation cycles were around 2000. The program was then repeated 5000 times. When a group gave rise to a numerical SIC-POVMs, it often occured very early on (less than 5 repeats). In some of the following cases, no numerical SIC-POVMs were found. While it maybe possible that SIC-POVM fiducials exist in the groups, for example if they existed with very specific parameter values that were hard to

randomly converge on, it seems unlikely. The implication of the SIC-POVMs not existing with certain canonical error groups but existing with others (with the same index group), is that it demonstrates the importance of a detailed analysis of different canonical abstract error groups, and not just the index groups.

#### A.2.3.1 SmallGroup(64, 67)

**Canonical Abstract Error Group:** `SmallGroup(512,261506).`

Let $\sigma_1 = \zeta_8 I_8$, and

$$\sigma_2 = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & -\zeta_4 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & \zeta_4 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\ \zeta_4 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & \zeta_4 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}; \sigma_3 = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \end{bmatrix};$$

$$\sigma_4 = \begin{bmatrix} 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \end{bmatrix};$$

Then $\{\sigma_i\}$ is a set of generators for this group.

$$v = \left(0, \frac{\sqrt{6}}{6}, \frac{\sqrt{3}}{6} + \frac{\sqrt{3}}{6}\boldsymbol{i}, 0, \frac{\sqrt{3}}{3}, -\frac{\sqrt{6}}{6}, -\frac{\sqrt{3}}{6} + \frac{\sqrt{3}}{6}\boldsymbol{i}, 0\right)^T.$$

is a fiducial vector giving rise to a SIC-POVM.

**Canonical Abstract Error Group:** `SmallGroup(512,261511)`.

Let $\sigma_1 = \zeta_8 I_8$, and

$$\sigma_2 = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & -\zeta_4 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & \zeta_4 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\ \zeta_4 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & \zeta_4 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}; \sigma_3 = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \end{bmatrix};$$

$$\sigma_4 = \begin{bmatrix} 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \end{bmatrix};$$

Then $\{\sigma_i\}$ is a set of generators for this group.

$$v = \left(0, 0, -\frac{\sqrt{6}}{6}, \frac{\sqrt{3}}{6} - \frac{\sqrt{3}}{6}\boldsymbol{i}, 0, \frac{\sqrt{3}}{3}, \frac{\sqrt{6}}{6}\boldsymbol{i}, \frac{\sqrt{3}}{6} - \frac{\sqrt{3}}{6}\boldsymbol{i}\right)^T.$$

is a fiducial vector giving rise to a SIC-POVM.

**Canonical Abstract Error Group:** `SmallGroup(512,261518)`.

NO SIC-POVM FOUND.

Let $\sigma_1 = \zeta_8 I_8$, and

$$\sigma_2 = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & \zeta_4 \\ 0 & 0 & 0 & 0 & 0 & 0 & -\zeta_4 & 0 \\ 0 & -\zeta_4 & 0 & 0 & 0 & 0 & 0 & 0 \\ \zeta_4 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}; \sigma_3 = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \end{bmatrix};$$

$$\sigma_4 = \begin{bmatrix} 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \end{bmatrix};$$

### A.2.4 SmallGroup(64, 138)

**Canonical Abstract Error Group:** `SmallGroup(512,402951)`.

Let $\sigma_1 = \zeta_8 I_8$, and

$$\sigma_2 = \begin{bmatrix} 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ \zeta_4 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & \zeta_4 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & \zeta_4 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & \zeta_4 & 0 & 0 & 0 & 0 \end{bmatrix}; \sigma_3 = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \end{bmatrix};$$

$$\sigma_4 = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix};$$

Then $\{\sigma_i\}$ is a set of generators for this group.

$$v = \left(0, 0, 0, \frac{\sqrt{3}}{3}, -\frac{\sqrt{6}}{6}\boldsymbol{i}, \frac{\sqrt{6}}{6}\boldsymbol{i}, -\frac{\sqrt{6}}{6}\boldsymbol{i}, -\frac{\sqrt{6}}{6}\right)^T.$$

is a fiducial vector giving rise to a SIC-POVM.

**Canonical Abstract Error Group:** `SmallGroup(512,402963).`

NO SIC-POVM FOUND.

Let $\sigma_1 = \zeta_8 I_8$, and

$$\sigma_2 = \begin{bmatrix} 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ \zeta_4 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & \zeta_4 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & \zeta_4 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & \zeta_4 & 0 & 0 & 0 & 0 \end{bmatrix}; \sigma_3 = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 \end{bmatrix};$$

$$\sigma_4 = \begin{bmatrix} 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\ -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & -\zeta_4 & 0 & 0 \\ 0 & 0 & 0 & 0 & \zeta_4 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & -\zeta_4 \\ 0 & 0 & 0 & 0 & 0 & 0 & \zeta_4 & 0 \end{bmatrix};$$

## A.3 Hoggar SIC-POVM in the Clifford group

Canonical abstract error groups can appear as subgroups of the Clifford group (see Table A.2). Some of these give rise to Hoggar lines. Here is one explicit example.

Let $\omega := e^{2\pi i/d}$ and $\mu := e^{\pi i/d}$. The canonical abstract error group `(512,6278298)` with index group `(64,195)` is generated by $\omega I$ and the four matrices

$$\begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & i & 0 & 0 \\ 0 & 0 & i & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 \\ 0 & -i & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & -i & 0 \\ 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 \end{pmatrix}, \quad \begin{pmatrix} 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & i \\ 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 \\ 0 & -i & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & i & 0 & 0 & 0 & 0 \\ -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & -i & 0 & 0 \end{pmatrix},$$

$$\begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & \mu \\ 0 & 0 & \mu^3 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & -\mu & 0 & 0 \\ \mu^3 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & \mu & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & \mu^3 & 0 \\ 0 & -\mu & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \mu^3 & 0 & 0 & 0 \end{pmatrix}, \quad \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 \end{pmatrix}.$$

One fiducial vector is

$$\begin{pmatrix} \frac{\sqrt{6}}{12} + \left(\frac{\sqrt{3}}{6} - \frac{\sqrt{6}}{12}\right) i \\ 0 \\ \frac{\sqrt{3}}{6}(-1+i) \\ \frac{1}{6}\sqrt{6-3\sqrt{2}}\, i \\ \frac{\sqrt{6}}{12} - \left(\frac{\sqrt{3}}{6} + \frac{\sqrt{6}}{12}\right) i \\ 0 \\ \frac{\sqrt{3}}{6}(1-i) \\ \frac{1}{6}\sqrt{6+3\sqrt{2}} \end{pmatrix}.$$

## A.4 Dimension 16

In 16 dimensions, it is no longer possible to look for nice error groups using the Small Groups Library since canonical nice error groups are of order $16^3 = 4096$. Theorem 5 of [KR02] gives one general class of abstract error groups of order $2^{2n-2}$.

**Theorem A.4.1** (Theorem 5 of [KR02])**.**
Let $f, g : \mathbb{Z}_{2^n} \to \mathbb{Z}_{2^n}$, be maps such that

$$f := (x \mapsto x + 1 \bmod 2^n), \qquad g := (x \mapsto 5x \bmod 2^n).$$

Let $H_n := \langle f, g \rangle$ be the group generated by compositions of $f$ and $g$. Then $H_n$ is an abstract error group of order $2^{2n-2}$. The index group $H_n/Z(H_n)$ is nonabelian provided that $n \geq 5$.

A numerical SIC-POVM search was conducted on $H_7$ (of order 4096) and after more than 2000 searches, was unable to find any SIC-POVMs. However, due to other considerations (e.g., it may just be harder to hit the Welch bound with higher dimensions), it is not yet wise to conclude this group does not yield SIC-POVMs.

# Appendix B

# Projective symmetry groups of cyclic harmonic frames

The tables summarise the calculations on projective symmetry groups of the cyclic harmonic frames $\Phi = \Phi_J$ of $n$ vectors for $\mathbb{C}^d$, $d = 2, \ldots, 7$.

**Column legend:**

1. $d$ – the dimension $\mathbb{C}^d$.
2. $n$ – the number of vectors in $\Phi$.
3. real – whether $\Phi$ is projectively real.
4. orth – whether any vectors in $\Phi$ are orthogonal.
5. reps – whether the vectors in $\Phi$ projectively repeated, i.e., are not projectively distinct.
6. $\mathrm{Sym}_P(\Phi)$ – the projective symmetry group of $\Phi$, when the vectors of $\Phi$ are projectively distinct.
7. $\mathrm{Sym}_{EP}(\Phi)$ – the extended projective symmetry group of $\Phi$, when $\Phi$ is projectively complex with projectively distinct vectors.
8. $J$ – a subset of $\mathbb{Z}_n$ giving $\Phi_J$ (up to projective equivalence and reordering).
9. $4$-cycle – whether $4$–cycles were needed to define the projective equivalence class.

| d | n | real | orth | reps | $\mathrm{Sym}_P(\Phi)$ | $\mathrm{Sym}_{EP}(\Phi)$ | $J$ | 4-cycle | erasures |
|---|---|---|---|---|---|---|---|---|---|
| 2 | 2 | yes | y | | <2,1> | | $\{0,1\}$ | | 0 |
| 2 | 3 | y | | | <6,1> | | $\{0,1\}$ | | 1 |
| 2 | 4 | y | y | | <8,3> | | $\{0,1\}$ | y | 2 |
| | | y | y | y | | | $\{1,3\}$ | | 1 |
| 2 | 5 | y | | | <10,1> | | $\{0,1\}$ | | 3 |
| 2 | 6 | y | | y | | | $\{1,3\}$ | | 3 |
| | | y | y | | <12,4> | | $\{1,2\}$ | | 4 |
| | | y | y | y | | | $\{1,4\}$ | | 2 |
| 2 | 7 | y | | | <14,1> | | $\{1,4\}$ | | 5 |
| 2 | 8 | y | y | | <16,7> | | $\{1,6\}$ | | 6 |
| | | y | y | y | | | $\{1,3\}$ | | 5 |
| | | y | y | y | | | $\{1,5\}$ | | 3 |
| 2 | 9 | y | | | <18,1> | | $\{1,3\}$ | | 7 |
| | | y | | y | | | $\{1,4\}$ | | 5 |
| 2 | 10 | y | | y | | | $\{1,5\}$ | | 7 |
| | | y | y | | <20,4> | | $\{1,4\}$ | | 8 |
| | | y | y | y | | | $\{1,6\}$ | | 4 |
| 2 | 11 | y | | | <22,1> | | $\{1,10\}$ | | 9 |
| 2 | 12 | y | y | y | | | $\{1,10\}$ | | 8 |
| | | y | y | | <24,6> | | $\{1,8\}$ | | 10 |
| | | y | | y | | | $\{1,9\}$ | | 7 |
| | | y | y | y | | | $\{1,7\}$ | | 5 |
| | | y | y | y | | | $\{1,3\}$ | | 9 |
| 2 | 13 | y | | | <26,1> | | $\{1,4\}$ | | 11 |
| 2 | 14 | y | y | | <28,3> | | $\{1,2\}$ | | 12 |
| | | y | | y | | | $\{1,5\}$ | | 11 |
| | | y | y | y | | | $\{1,8\}$ | | 6 |
| 2 | 15 | y | | | <30,3> | | $\{0,1\}$ | | 13 |
| | | y | | y | | | $\{1,10\}$ | | 11 |
| | | y | | y | | | $\{1,6\}$ | | 9 |

10. erasures – the number of vectors that can be removed from $\Phi$ so that those remaining still span $\mathbb{C}^d$.[1]

[1]The erasures data was obtained from the catalogue of harmonic frames compiled by N. Hay and S. Waldron at https://www.math.auckland.ac.nz/~waldron/Harmonic-frames/framesN, where N in the url is a natural number from 2 to 7.

| d | n | real | orth | reps | $\mathrm{Sym}_P(\Phi)$ | $\mathrm{Sym}_{EP}(\Phi)$ | $J$ | 4-cycle | erasures |
|---|---|---|---|---|---|---|---|---|---|
| 3 | 3 | y | y | | <6,1> | | $\{0,1,2\}$ | | 0 |
| 3 | 4 | y | | | <24,12> | | $\{1,2,3\}$ | | 1 |
| 3 | 5 | y | | | <10,1> | | $\{0,1,3\}$ | | 2 |
| 3 | 6 | | | | <18,3> | <36,10> | $\{0,1,4\}$ | | 2 |
| | | y | y | | <12,4> | | $\{1,2,3\}$ | y | 3 |
| | | y | y | y | | | $\{1,3,5\}$ | | 1 |
| 3 | 7 | | | | <21,1> | <42,1> | $\{1,2,6\}$ | | 4 |
| | | y | | | <14,1> | | $\{1,3,5\}$ | | 4 |
| 3 | 8 | | | | <16,8> | <32,43> | $\{1,3,4\}$ | | 5 |
| | | | | | <32,11> | <64,134> | $\{0,1,4\}$ | | 3 |
| | | y | | | <16,7> | | $\{0,1,2\}$ | | 5 |
| | | y | | y | | | $\{1,3,5\}$ | | 3 |
| 3 | 9 | | | | <9,1> | <18,1> | $\{1,4,6\}$ | | 5 |
| | | y | y | | <18,1> | | $\{0,1,2\}$ | | 6 |
| | | y | y | y | | | $\{1,4,7\}$ | | 2 |
| 3 | 10 | | | | <50,3> | <100,13> | $\{0,1,5\}$ | | 4 |
| | | y | | | <20,4> | | $\{0,1,9\}$ | | 7 |
| | | | | | <10,2> | <20,4> | $\{0,1,8\}$ | | 7 |
| | | y | | y | | | $\{1,5,7\}$ | | 5 |
| 3 | 11 | | | | <11,1> | <22,1> | $\{0,1,3\}$ | | 8 |
| | | y | | | <22,1> | | $\{1,2,3\}$ | | 8 |
| 3 | 12 | | | | <12,2> | <24,6> | $\{1,2,11\}$ | | 8 |
| | | y | y | | <24,6> | | $\{1,2,3\}$ | | 9 |
| | | y | | y | | | $\{1,4,10\}$ | | 5 |
| | | | | | <12,2> | <24,6> | $\{0,3,4\}$ | | 7 |
| | | | | y | | | $\{1,5,7\}$ | | 5 |
| | | | y | | <24,5> | <48,38> | $\{0,1,8\}$ | | 7 |
| | | y | y | y | | | $\{1,3,5\}$ | | 7 |
| | | | | | <72,30> | <144,154> | $\{2,3,8\}$ | | 5 |
| | | y | y | y | | | $\{1,5,9\}$ | | 3 |
| 3 | 13 | y | | | <26,1> | | $\{0,1,12\}$ | | 10 |
| | | | | | <13,1> | <26,1> | $\{0,1,3\}$ | | 10 |
| | | | | | <39,1> | <78,1> | $\{1,2,11\}$ | | 10 |

| d | n | real | orth | reps | $\mathrm{Sym}_P(\Phi)$ | $\mathrm{Sym}_{EP}(\Phi)$ | $J$ | 4-cycle | erasures |
|---|---|---|---|---|---|---|---|---|---|
| 4 | 4 | y | y | | <24,12> | | $\{0,1,2,3\}$ | | 0 |
| 4 | 5 | y | | | <120,34> | | $\{0,1,2,3\}$ | | 1 |
| 4 | 6 | y | y | | <12,4> | | $\{1,2,3,4\}$ | | 2 |
| | | y | | | <48,48> | | $\{1,2,3,5\}$ | | 1 |
| | | y | y | | <72,40> | | $\{0,1,3,4\}$ | | 1 |
| 4 | 7 | | | | <21,1> | <42,1> | $\{0,1,2,4\}$ | | 3 |
| | | y | | | <14,1> | | $\{1,2,3,4\}$ | | 3 |
| 4 | 8 | y | y | | <16,7> | | $\{1,2,4,7\}$ | y | 4 |
| | | | | | <8,1> | <16,7> | $\{1,2,3,5\}$ | | 3 |
| | | y | y | | <128,928> | | $\{0,1,4,5\}$ | y | 2 |
| | | y | y | | <32,43> | | $\{0,1,3,4\}$ | | 3 |
| | | y | y | y | | | $\{1,3,5,7\}$ | | 1 |
| 4 | 9 | | | | <9,1> | <18,1> | $\{1,2,3,8\}$ | | 5 |
| | | | | | <162,10> | <324,39> | $\{0,1,4,7\}$ | | 2 |
| | | y | | | <18,1> | | $\{1,2,4,5\}$ | | 4 |
| | | y | | | <18,1> | | $\{0,1,2,3\}$ | | 5 |
| 4 | 10 | | | | <10,2> | <20,4> | $\{0,1,3,7\}$ | | 5 |
| | | y | y | | <20,4> | | $\{1,2,4,9\}$ | | 4 |
| | | y | y | | <20,4> | | $\{0,1,2,3\}$ | | 6 |
| | | y | y | | <40,12> | | $\{0,1,3,4\}$ | | 5 |
| | | | | | <10,2> | <20,4> | $\{1,2,4,6\}$ | | 4 |
| | | | | | <10,2> | <20,4> | $\{0,1,5,8\}$ | | 4 |
| | | y | | | <20,4> | | $\{0,2,5,8\}$ | | 4 |
| | | y | y | | <200,43> | | $\{1,2,6,7\}$ | | 3 |
| | | y | | y | | | $\{1,3,5,7\}$ | | 3 |
| 4 | 11 | y | | | <22,1> | | $\{1,2,5,9\}$ | | 7 |
| | | | | | <11,1> | <22,1> | $\{0,1,2,7\}$ | | 7 |
| | | | | | <11,1> | <22,1> | $\{0,1,2,8\}$ | | 7 |
| | | y | | | <22,1> | | $\{1,3,7,8\}$ | | 7 |
| 4 | 12 | | | | <12,2> | <24,6> | $\{1,3,6,9\}$ | | 5 |
| | | | | | <384,5557> | <768,1088009> | $\{1,2,6,10\}$ | | 3 |
| | | | | | <24,5> | <48,38> | $\{3,4,6,8\}$ | | 7 |
| | | y | y | | <48,38> | | $\{2,3,9,10\}$ | | 5 |
| | | y | y | | <24,6> | | $\{1,4,8,9\}$ | | 6 |
| | | | | | <12,2> | <24,6> | $\{0,1,2,8\}$ | | 5 |
| | | | | | <36,6> | <72,23> | $\{0,2,3,8\}$ | | 5 |
| | | y | y | | <24,6> | | $\{1,4,8,11\}$ | | 7 |
| | | y | y | | <24,6> | | $\{1,4,6,11\}$ | | 8 |
| | | y | | | <12,2> | <24,6> | $\{1,3,6,11\}$ | | 7 |
| | | | y | | <24,10> | <48,38> | $\{2,3,6,9\}$ | | 5 |
| | | | y | | <12,2> | <24,6> | $\{0,1,4,11\}$ | | 7 |
| | | | y | | <72,42> | <144,183> | $\{1,2,8,11\}$ | | 5 |
| | | y | y | | <48,38> | | $\{0,1,4,9\}$ | | 5 |
| | | y | y | | <48,38> | | $\{1,2,4,11\}$ | | 6 |
| | | y | y | | <48,38> | | $\{0,1,6,11\}$ | | 5 |
| | | y | y | y | | | $\{1,4,7,10\}$ | | 2 |
| | | y | y | y | | | $\{1,5,7,11\}$ | | 3 |
| | | y | y | y | | | $\{1,3,7,11\}$ | | 3 |
| | | y | y | | <288,889> | | $\{0,1,6,7\}$ | | 4 |
| | | y | y | y | | | $\{1,3,5,7\}$ | | 5 |

| d | n | real | orth | reps | $\mathrm{Sym}_P(\Phi)$ | $\mathrm{Sym}_{EP}(\Phi)$ | $J$ | 4-cycle | erasures |
|---|---|---|---|---|---|---|---|---|---|
| 5 | 5 | y | y | | <120,34> | | $\{0,1,2,3,4\}$ | | 0 |
| 5 | 6 | y | | | <720,763> | | $\{0,1,2,3,4\}$ | | 1 |
| 5 | 7 | y | | | <14,1> | | $\{0,1,2,3,5\}$ | | 2 |
| 5 | 8 | | | | <32,11> | <64, 134> | $\{1,2,3,5,6\}$ | | 2 |
| | | y | | | <16,7> | | $\{0,1,2,6,7\}$ | | 3 |
| | | | | | <16,8> | <32,43> | $\{0,1,2,3,6\}$ | | 3 |
| | | y | | | <384,5602> | | $\{1,2,3,5,7\}$ | | 1 |
| 5 | 9 | | | | <9,1> | <18,1> | $\{0,1,2,3,7\}$ | | 4 |
| | | | | | <162,10> | <324,39> | $\{0,1,3,4,7\}$ | | 2 |
| | | y | | | <18,1> | | $\{0,1,2,3,8\}$ | | 4 |
| | | y | | | <18,1> | | $\{1,2,3,4,7\}$ | | 2 |
| 5 | 10 | | | | <10,2> | <20,4> | $\{1,2,5,6,9\}$ | | 4 |
| | | | | | <10,2> | <20,4> | $\{1,2,4,8,9\}$ | | 4 |
| | | y | y | | <20,4> | | $\{1,2,4,5,8\}$ | y | 5 |
| | | y | | | <200,43> | | $\{1,2,3,6,8\}$ | | 3 |
| | | | | | <10,2> | <204,> | $\{1,3,4,5,8\}$ | | 4 |
| | | | | | <50,3> | <100,13> | $\{1,2,3,7,8\}$ | | 3 |
| | | | | | <10,2> | <20,4> | $\{0,1,3,5,9\}$ | | 3 |
| | | y | y | | <240,189> | | $\{0,1,2,4,8\}$ | | 3 |
| | | y | y | y | | | $\{1,3,5,7,9\}$ | | 1 |
| 5 | 11 | | | | <11,1> | <22,1> | $\{0,1,2,4,5\}$ | | 6 |
| | | | | | <11,1> | <22,1> | $\{0,1,2,5,9\}$ | | 6 |
| | | | | | <55,1> | <110,1> | $\{1,2,3,5,8\}$ | | 6 |
| | | | | | <11,1> | <22,1> | $\{1,2,3,4,10\}$ | | 6 |
| | | y | | | <22,1> | | $\{0,1,2,3,10\}$ | | 6 |
| | | y | | | <22,1> | | $\{0,1,2,4,9\}$ | | 6 |

| d | n | real | orth | reps | $\mathrm{Sym}_P(\Phi)$ | $\mathrm{Sym}_{EP}(\Phi)$ | $J$ | 4-cycle |
|---|---|---|---|---|---|---|---|---|
| 6 | 6 | y | y | | <720,763> | | $\{0,1,2,3,4,5\}$ | |
| 6 | 7 | y | | | 5040 | | $\{0,1,2,3,4,5\}$ | |
| 6 | 8 | y | y | | <16,7> | | $\{0,1,2,3,5,6\}$ | |
| | | y | y | | <128,928> | | $\{0,1,2,4,6,7\}$ | |
| | | y | y | | 1152 | | $\{0,1,2,4,5,6\}$ | |
| 6 | 9 | | | | <9,1> | <18,1> | $\{0,1,2,3,4,6\}$ | |
| | | y | y | | <18,1> | | $\{0,1,2,3,4,8\}$ | |
| | | y | y | | <1296,3490> | | $\{0,1,3,4,6,7\}$ | |
| 6 | 10 | | | | <10,2> | <20,4> | $\{0,1,2,5,7,9\}$ | |
| | | | y | | <10,2> | <20,4> | $\{0,1,3,4,5,8\}$ | |
| | | y | | | 3840 | | $\{1,2,3,5,7,9\}$ | |
| | | y | y | | <20,4> | | $\{0,1,2,3,6,7\}$ | |
| | | | | | <10,2> | <20,4> | $\{1,2,3,4,5,9\}$ | |
| | | y | y | | <200,43> | | $\{0,1,4,5,6,9\}$ | |
| | | y | y | | <20,4> | | $\{0,1,3,4,7,8\}$ | |
| | | y | y | | <40,12> | | $\{0,1,2,5,8,9\}$ | |
| | | y | | | <20,4> | | $\{1,2,3,5,6,7\}$ | |
| 6 | 11 | | | | <11,1> | <22,1> | $\{1,2,3,4,5,9\}$ | |
| | | | | | <11,1> | <22,1> | $\{0,1,2,3,5,10\}$ | |
| | | y | | | <22,1> | | $\{0,1,2,8,9,10\}$ | |
| | | y | | | <22,1> | | $\{1,2,3,8,9,10\}$ | |
| | | | | | <11,1> | <22,1> | $\{1,2,3,5,6,10\}$ | |
| | | | | | <55,1> | <110,1> | $\{0,1,3,4,5,9\}$ | |

TABLE B.1: The erasure column is omitted. For 6 vectors the number of erasures is 0 and for 7 vectors it is 1. Past this point, no computational data exists.

| d | n | real | orth | reps | $\mathrm{Sym}_P(\Phi)$ | $\mathrm{Sym}_{EP}(\Phi)$ | $J$ | 4-cycle |
|---|---|---|---|---|---|---|---|---|
| 7 | 7 | y | y | | 5040 | | $\{0,1,2,3,4,5,6\}$ | |
| 7 | 8 | y | | | 40320 | | $\{0,1,2,3,4,5,7\}$ | |
| 7 | 9 | y | | | <1296,3490> | | $\{0,1,2,4,5,7,8\}$ | |
| | | y | | | <18,1> | | $\{0,1,2,3,5,7,8\}$ | |
| 7 | 10 | | | | <10,2> | <20,4> | $\{1,2,3,4,5,6,8\}$ | |
| | | | | | <50,3> | <100,13> | $\{0,1,2,5,6,7,8\}$ | |
| | | y | | | <320,1636> | | $\{1,2,3,5,6,7,9\}$ | |
| | | y | | | <20,4> | | $\{1,2,4,5,6,8,9\}$ | |
| | 11 | | | | <11,1> | <22,1> | $\{0,1,2,3,5,6,9\}$ | |
| | | | | | <11,1> | <22,1> | $\{0,1,2,6,7,8,9\}$ | |
| | | y | | | <22,1> | | $\{1,2,3,4,5,6,8,10\}$ | |
| | | y | | | <22,1> | | $\{1,2,3,4,5,8,9\}$ | |
| 7 | 12 | y | y | | <24,6> | | $\{0,1,2,4,8,10,11\}$ | |
| | | | | | <12,2> | <24,6> | $\{1,2,3,4,6,7,10\}$ | |
| | | | | | <72,30> | <144,154> | $\{1,2,3,4,7,8,10\}$ | |
| | | | | | <12,2> | <24,6> | $\{1,3,4,5,7,8,9\}$ | |
| | | | | | <12,2> | <24,6> | $\{0,1,2,4,6,10,11\}$ | |
| | | | | | <12,2> | <24,6> | $\{0,1,2,3,6,10,11\}$ | |
| | | | | | <12,2> | <24,6> | $\{0,1,2,3,4,8,9\}$ | |
| | | | | | <12,2> | <24,6> | $\{1,2,3,4,5,6,11\}$ | |
| | | y | | | <24,6> | | $\{0,1,2,3,4,10,11\}$ | |
| | | | | | <12,2> | <24,6> | $\{1,2,3,4,6,8,9\}$ | |
| | | y | | | <96,187> | | $\{0,1,2,4,5,6,9\}$ | |
| | | | | | <72,30> | <144,154> | $\{1,2,3,5,8,9,11\}$ | |
| | | | | | <12,2> | <24,6> | $\{1,2,4,5,6,10,11\}$ | |
| | | | | | <72,30> | <144,154> | $\{1,2,3,7,8,9,10\}$ | |
| | | | | | <12,2> | <24,6> | $\{1,2,3,4,5,7,10\}$ | |
| | | | | | <24,10> | <48,38> | $\{1,2,3,4,6,7,9\}$ | |
| | | | | | <12,2> | <24,6> | $\{0,1,2,3,5,6,11\}$ | |
| | | y | | | <144,183> | | $\{0,1,2,3,4,7,9\}$ | |
| | | y | | | 46080 | | $\{1,3,4,5,7,9,11\}$ | |
| | | | | | <24,5> | <48,38> | $\{1,2,3,4,6,7,11\}$ | |
| | | | y | | <24,5> | <48,38> | $\{1,2,3,5,7,10,11\}$ | |
| | | y | | | <288,889> | | $\{1,2,3,5,7,8,9\}$ | |
| | | | | | <384,5557> | <768, 1088009> | $\{1,2,3,6,7,10,11\}$ | |
| | | y | y | | <192, 1472> | | $\{0,1,3,4,7,8,9\}$ | y |
| | | | | | <1944,3876> | 3888 | $\{0,1,3,4,6,7,9\}$ | |

TABLE B.2: The erasure column is omitted. For 7 vectors the number of erasures is 0 and for 8 vectors it is 1. Past this point, no computational data exists.

# Appendix C

# SIC-POVM in 17 dimensions

Using the Weyl-Heisenberg displacement operators (see Definition 2.3.1), the following $v$ is a new analytic SIC-POVM fiducial vector corresponding to the numerical 17c SIC-POVM in [SG10]. It was reverse engineered using the techniques from Chapter 3. The Scott-Grassl numerical solution ($SGv$) is first presented for comparison.

## C.1 The Scott-Grassl numerical solution for 17c

$SGv :=$ (

$+.48382980061662116217682602441506776537e + 0 + .000000000000000000000000000000000000e + 0 * I,$

$+ .27126484005469393538676883137444152309e + 0 + .69654983976200813223024153094325067504e - 1 * I,$

$+ .18658653094610710051164653413508437529e - 1 + .23700687707779005149550572962460233826e + 0 * I,$

$+ .11319309510366287709138835914195952421e + 0 + .32636202824922905980362245531983853 41e - 1 * I,$

$- .84225229290413586558883729975732400897e - 1 - .90360180246105915429383967536823361227e - 1 * I,$

$+ .13227984962902189203796538799252464632e + 0 + .13432103559521274434810260359045039985e + 0 * I,$

$+ .14004538045056275183072523113362579132e + 0 + .22296185348823112246409465467506215810e + 0 * I,$

$+ .82603198698614458128664900745377769187e - 1 + .33365677423215239918728902391140880142e - 1 * I,$

$+ .10926637036618627945875152106606807213e + 0 + .14440203852655326959578110616076824445e + 0 * I,$

$+ .18325895308228673672789111304629521170e + 0 - .70434780649141597277247665615861683213e - 2 * I,$

$+ .21065522597967223869239046620008904046e + 0 - .27476696758770999364612862635894575720e + 0 * I,$

$+ .16744513641220197618902743846181827091e + 0 - .89188124779113258848800204496925350890e - 1 * I,$

$+ .24359388765298324677017073863545845111e + 0 - .98778919986922507345035476599550316894e - 1 * I,$

$- .15216662789091992219769080826760384307e - 2 + .10096782395236138821142045805553955699e + 0 * I,$

$+ .31930652975339555818323482188870297478e - 2 - .35047984081599375367709829247485453467e + 0 * I,$

$+ .12948876712345079950472823771660306613e + 0 + .22926607189219006777123769404536112165e - 2 * I,$

$- .20814795522891847091328065519554133217e + 0 - .66991642102649847248234875056855152662e - 1 * I)^T.$

## C.2 Analytic solution for 17c

Let $a := \sqrt{7}, i := \sqrt{-1}, r1 := \sqrt{17}, t := \sin(\pi/17)$,

$s := 3757 + 1411 * a - 1479 * r1 - 561 * a * r1 - 68 * t - 119 * a * t - 2428 * r1 * t - 923 * a * r1 * t - 25500 * t^2 - 9588 * a * t^2 +$

$10572 * r1 * t^2 + 4012 * a * r1 * t^2 + 20808 * t^3 + 8840 * a * t^3 + 16744 * r1 * t^3 + 6464 * a * r1 * t^3 + 45696 * t^4 + 17136 * a * t^4 -$

$20640 * r1 * t^4 - 7856 * a * r1 * t^4 - 68544 * t^5 - 28288 * a * t^5 - 33600 * r1 * t^5 - 13120 * a * r1 * t^5 - 26112 * t^6 - 9792 * a * t^6 +$

$11904 * r1 * t^6 + 4544 * a * r1 * t^6 + 47872 * t^7 + 19584 * a * t^7 + 19200 * r1 * t^7 + 7552 * a * r1 * t^7,$

$b1 := 8 * \sqrt{s},$

$b2 := 17 * ( \sqrt[3]{\frac{1}{17}(10 + 3 * i * \sqrt{21})})^{-1} + 17^{2/3}\ \sqrt[3]{10 + 3 * i * \sqrt{21}} \approx 32.338,$

$w := [w_1, w_2, w_3, w_4, w_5, w_6, w_7, w_8, w_9, w_{10}, w_{11}, w_{12}, w_{13}, w_{14}, w_{15}, w_{16}, w_{17}]^T,$

$v := w/\,||w||,$

$w_1 := (1/436968 * (5 * a - 7) * r1 + 1/436968 * (13 * a - 7)) * b2^2 + (1/436968 * (-71 * a + 259) * r1 + 1/25704 * (a + 35)) * b2 +$

$1/6426 * (-53 * a + 154) * r1 + 1/378 * (-17 * a + 56),$

$w_2 := ((((1/40251494791862835497604 * (2390569005052745227 * a - 6274715912124044301) * i + 1/40251494791862835497604 *$

$(3140356510548608341*a-8182805022494548537))*r1+(1/40251494791862835497604*(-2544929372131190022 7*a+67878053605415835929)*$

$i + 1/40251494791862835497604 * (6095408318187223927 * a - 14412281383608493903))) * t^7 + ((1/40251494791862835497604 *$

$(-5174616347507284263*a+13722555477385500193)*i+1/9583689236157817975 62*(-2675388956621709*a+5224174754906894))*$

$r1+(1/2012574739593141774880 2*(5592126147766100912*a-1473642377662638143)*i+1/5750213541694690785372*(747704864310938143*$

$a - 2040229956970773211))) * t^6 + ((1/161005979167451341990416 * (-20589722611921746273 * a + 54214481831474310199) *$

$i + 1/230008541666778763141488 * (-2838305488391356931 * a + 7417129235112744479)) * r1 + (1/161005979167451341990416 *$

$(1749875848397722367 53*a-466717831483708329791)*i+1/161005979167451341990416*(-3181594507423232764 7*a+72554432115740794835)))*$

$t^5 + ((1/161005979167451341990416 * (36423739573932214129 * a - 96541942747232286689) * i + 1/805029895837256709952 08 *$

$(743216710732381766*a-171975891482675118 3))*r1+(1/161005979167451341990416*(-8977620895653092437 5*a+23693684585398911614 3)*$

$i + 1/805029895837256709952 08 * (-17512882541262701580 * a + 47823297015739991669))) * t^4 + ((1/64402391666980536796166 4 *$

$(5696156021358365092 5*a-150519805180552211341)*i+1/21467463888993512265388 8*(109233455208755310 77*a-28833442994936151971))*$

$r1 + (1/64402391666980536796166 4 * (-345334687402183256611 * a + 920978128808018754011) * i + 1/92003416667115052565952 *$

$(365360795098655089 * a - 6655761637970056023))) * t^3 + ((1/64402391666980536796166 4 * (-75369662159440958095 * a +$

$$19961090078808645 9658)*i+1/64402391666980536796 1664*(-6079477439371846044*a+1547829479789712 9331))*r1+(1/64402391666980536796 1664*(22626889793411132 2586*a-5982322366955230574 25)*i+1/21467463888993512265 3888*(21337810960597229949*a-5810741969547931 5896)))*t^2+((1/25760956666792214718 46656*(-45995906681322685837*a+12179669353341698 7645)*i+1/36801366666846021026 3808*(-1707854581829808547*a+4660701707985975529))*r1+(1/25760956666792214718 46656*(18550631964232553007 3*a-49517341765382458695 7)*i+1/25760956666792214718 46656*(17355961955320234181*a-54779880690635077915)))*t+((1/51521913333584429436 93312*(87844634076506120143*a-232405072641424794293)*i+1/73602733333692042052 7616*(18161165391439251 49*a-4828133107193657553))*r1+(1/51521913333584429436 93312*(-33228297223686029924 3*a+879613385494378101189)*i+1/51521913333584429436 93312*(-57638587923446442463*a+155627165414523147583))))*b1+(((1/54621*(64*a-196)*i+1/54621*(-24*a-28))*r1+(1/7803*(28*a-56)*i+1/7803*(-12*a-8)))*t^7+((1/18207*(a-7)*i+1/54621*(53*a-133))*r1+(1/18207*(-a+7)*i+1/18207*(-39*a+119)))*t^6+((1/54621*(-101*a+315)*i+1/18207*(12*a+14))*r1+(1/54621*(-255*a+455)*i+1/18207*(40*a+28)))*t^5+((1/31212*(-3*a+19)*i+1/218484*(-373*a+959))*r1+(1/218484*(33*a-217)*i+1/218484*(961*a-2891)))*t^4+((1/109242*(81*a-266)*i+1/72828*(-17*a-21))*r1+(1/218484*(191*a-63)*i+1/12138*(-8*a-7)))*t^3+((1/62424*(3*a-17)*i+1/109242*(97*a-259))*r1+(1/436968*(-57*a+329)*i+1/218484*(-620*a+1827)))*t^2+((1/1747872*(-117*a+427)*i+1/249696*(3*a+5))*r1+(1/1747872*(359*a-1421)*i+1/1747872*(-3*a+119)))*t+((1/582624*(-4*a+21)*i+1/1747872*(-230*a+651))*r1+(1/102816*(3*a-14)*i+1/1747872*(919*a-2646)))))*b2^2+((((1/40251494791862835497604*(-16419990613303051759*a+43309923507881279823)*i+1/40251494791862835497604*(-39036517189970381107*a+107266424458446531313))*r1+(1/23677349877566373822 12*(247925677780431515*a+85322904020035913)*i+1/23677349877566373822 12*(57943931022623518 99*a-14844217839616884397)))*t^7+((1/40251494791862835497604*(7949743185508015389*a-23346951138992052943)*i+1/95836892361578179756 2*(625838116311685329*a-1692592185169597297))*r1+(1/11838674938781869110 6*(-1742075389857139495*a+4893185444080982629)*i+1/33824785539380534031 6*(-1223658786509308601*a+3295886034896043137)))*t^6+((1/16100597916745134199041 6*(10629452806304219524 5*a-27928339660710344281 3)*i+1/23000854166778763141488*(39225985944970408445*a-10737124790372543527 1))*r1+(1/94709399510265495288 48*(-3697573137526200041*a+474867292660308261 7)*i+1/94709399510265495288 48*(-45161689581211108883*a+11712945508533079976 9)))*t^5+((1/16100597916745134199041 6*(-66163554460053438733*a+192973493098683209171)*i+1/80502989583725670995208*(-10029621879733333654 4*a+271477067867290334049))*r1+(1/94709399510265495288 48*(25438094502765231765*a-71383200556498479103)*i+1/47354699755132747644 24*(30775933732129850853*a-82872019162047834544)))*t^4+((1/64402391666980536796 1664*(-188293644903635050455*a+487763071828579352617)*i+1/21467463888993512265 3888*(-188680951967755323301*a+51279373040705082 7189))*r1+(1/37883759804106198115392*(15617168515589965829*a-32342874558948948211)*i+1/54119656863008854450 56*(15761886385783396771*a-41580615253727777523)))*t^3+((1/64402391666980536796 1664*(17469109186633450314 7*a-50230514199166036674 4)*i+1/64402391666980536796 1664*(47196438978923557628 4*a-12781609323030642699 45))*r1+(1/37883759804106198115392*(-55396560404244926596*a+155740791494158130967)*i+1/12627919934702066038464*(-44117991672526563397*a+118731738122500538448)))*t^2+((1/25760956666792214718 46656*(84721322039805509971*a-209636787681486392745)*i+1/36801366666846021026 3808*(46496810401572529915*a-125045756777728369315))*r1+(1/15153503921642479246 1568*(-16106802193492220211*a+39691207455329431757)*i+1/15153503921642479246 1568*(-79121879086010585305*a+212497605428668169477)))*

$$t+((1/5152191333358442943693312*(-2709374125141737492 49*a+76319736968356225 5785)*i+1/7360273333692042052 7616*(-92560898387575866853*a+249917962165261701051))*r1+(1/3030700784328495849231 36*(69116402936180655637*a-195536499863446746753)*i+1/3030700784328495849231 36*(163246324692661647239*a-439827851689933386365))))*b1+(((1/54621*(536*a-1988)*i+1/3213*(24*a-28))*r1+(1/459*(4*a-32)*i+1/459*(12*a-8)))*t^7+((1/1071*(-a-7)*i+1/54621*(-635*a+1645))*r1+(1/1071*(a+7)*i+1/1071*(25*a-63)))*t^6+((1/54621*(-817*a+2961)*i+1/1071*(-12*a+14))*r1+(1/3213*(-39*a+329)*i+1/1071*(-40*a+28)))*t^5+((1/1836*(3*a+19)*i+1/218484*(4255*a-11123))*r1+(1/12852*(-33*a-217)*i+1/12852*(-611*a+1519)))*t^4+((1/109242*(639*a-2128)*i+1/4284*(17*a-21))*r1+(1/12852*(47*a-441)*i+1/714*(8*a-7)))*t^3+((1/3672*(-3*a-17)*i+1/109242*(-1033*a+2737))*r1+(1/25704*(57*a+329)*i+1/12852*(382*a-945)))*t^2+((1/1747872*(-825*a+1673)*i+1/14688*(-3*a+5))*r1+(1/102816*(-37*a+217)*i+1/102816*(3*a+119)))*t+((1/34272*(4*a+21)*i+1/1747872*(2300*a-5985))*r1+(1/6048*(-3*a-14)*i+1/102816*(-527*a+1344)))))*b2+(((1/1183867493878318691106*(-2371590684277684552 7*a+60404882084096784591)*i+1/1183867493878318691106*(40846349726057211121*a-107683110140227104511))*r1+(1/1183867493878318691106*(380961147325584773939*a-101794340482217661222 7)*i+1/1183867493878318691106*(-314519136276578053 97*a+815896341453286449251)))*t^7+((1/2367734987756637382212*(139783450988127571095*a-373204046344411604917)*i+1/3758309504375614892 4*(-2029427772856309473*a+5351750079406275181))*r1+(1/2367734987756637382212*(-204378311956405861819*a+555865666042652701981)*i+1/3382478553938053403 16*(56904771520087183961*a-146995915227768170177)))*t^6+((1/4735469975513274764424*(231056182372870097829*a-597258702027556072553)*i+1/6764957107876106806 32*(-46687017854385570497*a+122862457095634699091))*r1+(1/4735469975513274764424*(-260031228882560166856 1*a+694575010473995368188 1)*i+1/4735469975513274764424*(221345864587191852401 5*a-573999793668937291492 9)))*t^5+((1/473546997551327476442 4*(-47736076914417412609 4*a+127534119349589037927 1)*i+1/473546997551327476442 4*(463200162530399946133*a-121976548550537425277 4))*r1+(1/473546997551327476442 4*(875492815353328163046*a-237520200395542783753 3)*i+1/473546997551327476442 4*(-150933069482479463055 9*a+390592107763915059524 2)))*t^4+((1/947093995102654952884 8*(-362119114824784684887*a+948416374354187530750)*i+1/350775553741724056624*(1523557836736856482 6*a-39958181802015058449))*r1+(1/947093995102654952884 8*(250574390538909056043 7*a-668745263229091580441 0)*i+1/135299142157522136126 4*(-328054953051039747994*a+851197867504469771463)))*t^3+((1/378837598041061981153 92*(188218877734362815829 7*a-503529737028406808757 1)*i+1/378837598041061981153 92*(-200176127619042069671 7*a+525586786568586276852 5))*r1+(1/378837598041061981153 92*(-473030283591283384092 5*a+127867758353670895412 79)*i+1/126279199347020660384 64*(236558652607230853352 3*a-614548355176027672582 7)))*t^2+((1/757675196082123962307 84*(642172092388719221339*a-169535080987626705264 9)*i+1/108239313726017708901 12*(-86558820246891040453*a+226014644041152485089))*r1+(1/757675196082123962307 84*(-254241303459757192647 9*a+678335929164819882338 9)*i+1/757675196082123962307 84*(271837584796107611416 7*a-706902658781744482806 7)))*t+((1/151535039216424792461568*(-102365632829986507894 1*a+274412726191105565893 9)*i+1/216478627452035417802 24*(175853581999552236703*a-460070660528379149667))*r1+(1/151535039216424792461568*(365161579024176947500 9*a-983406651478048019558 7)*i+1/151535039216424792461568*(-490098484171061351436 1*a+127901469150112780824 73))))*b1+((1/3213*(-2596*a+7084)*i+1/3213*(1908*a-2324))*r1+(1/27*(-68*a+124)*i+1/27*(36*a-20)))*t^7+((1/1071*(-174*$

$a+658)*i+1/3213*(-1382*a+3766))*r1+(1/63*(30*a-98)*i+1/63*(50*a-98)))*t^6+((1/3213*(4064*a-11340)*i+1/1071*$
$(-996*a+1288))*r1+(1/189*(636*a-1036)*i+1/63*(-108*a+28)))*t^5+((1/918*(279*a-1007)*i+1/6426*(4913*a-13363))*$
$r1+(1/378*(-369*a+1225)*i+1/378*(-625*a+1379)))*t^4+((1/12852*(-6453*a+19033)*i+1/476*(177*a-259))*r1+(1/756*$
$(-571*a+315)*i+1/84*(25*a+49)))*t^3+((1/1836*(-315*a+1081)*i+1/12852*(-5147*a+13979))*r1+(1/756*(477*a-1589)*$
$i+1/756*(841*a-2079)))*t^2+((1/25704*(1194*a-3997)*i+1/3672*(-117*a+232))*r1+(1/1512*(-148*a+679)*i+1/1512*$
$(117*a-406)))*t+(1/672*(19*a-63)*i+1/102816*(5867*a-15981))*r1+1/6048*(-711*a+2303)*i+1/6048*(-1355*a+3633),$

$w_3 := ((((1/40251494791862835497604 * (2032708796993678841 * a - 5439126824113408727) * i + 1/40251494791862835497604 *$
$(196209660521616893 1*a-5416407653427664859))*r1+(1/44723883102069817219 56*(32980255387477122 7*a-912196215169921297)*$
$i + 1/40251494791862835497604 * (5828224036634168909 * a - 1642855630958354 8809))) * t^7 + ((1/40251494791862835497604 *$
$(-48258912780223168*a+137033677318130397)*i+1/40251494791862835497604*(-44011331449658765*a+279378079766194766))*$
$r1 + (1/40251494791862835497604 * (6103412458394036241 * a - 16256670600286958450) * i + 1/40251494791862835497604 *$
$(2685089790805721786 * a - 7084358370056268217))) * t^6 + ((1/161005979167451341990416 * (-12913423133103850087 * a +$
$3465161166809469 9523)*i+1/230008541667787631414 88*(-172520200613756064 7*a+476180233965266368 9))*r1+(1/947093995102654952 8848*$
$(-777301094064947733*a+2211627514221275773)*i+1/161005979167451341990416*(-3463547845712580112 3*a+97560711179423862657)))*$
$t^5+((1/805029895837256709952 08*(685341135520214680*a-1889814418869966001)*i+1/536686597224837806 63472*(230918402050263551*$
$a - 974635666182973235)) * r1 + (1/805029895837256709952 08 * (-19992063931713224061 * a + 53167005662377091023) * i +$
$1/161005979167451341990416 * (-17800670278932845599 * a + 47469547276876861157))) * t^4 + ((1/644023916669805367961664 *$
$(21579291599702136803*a-58070474435484769909)*i+1/644023916669805367961664*(18287987611966595037*a-50435052587635048445))*$
$r1 + (1/644023916669805367961664 * (-3602593737165311033 * a + 4523257084745413035) * i + 1/644023916669805367961664 *$
$(46141891588337301261 * a - 13006976843681386961))) * t^3 + ((1/644023916669805367961664 * (-6660777524916047175 * a +$
$18111043082487485396)*i+1/644023916669805367961664*(-31430231020132473 09*a+10367588372094423274))*r1+(1/920034166671150525 65952*$
$(10424619878541837916*a-27696300630832638309)*i+1/644023916669805367961664*(32283908567177680778*a-87925191244381131319)))*$
$t^2+((1/25760956666792214718466 56*(-819611731462977350 3*a+219361018454557219 03)*i+1/25760956666792214718466 56*$
$(-472908637059756633 1*a+128115006667130773 29))*r1+(1/85869855555974049061 5552*(10365384534228766985*a-26999856410250381525)*$
$i+1/25760956666792214718466 56*(-48778422280085063 01*a+13129439189228456899)))*t+((1/51521913333584429436 93312*$
$(13419961957682462559*a-36030095515019888009)*i+1/73602733333692042052 7616*(943509530710860337*a-2753827852345367769))*$
$r1 + (1/73602733333692042052 7616 * (-9962928804965238465 * a + 26553379369306596007) * i + 1/17173971111194809812 31104 *$
$(-10157647553349389273 * a + 29034274351818420769)))) * b1 + (((1/54621 * (-76 * a + 140) * i + 1/54621 * (12 * a - 28)) * r1 +$
$(1/54621*(-152*a+392)*i+1/54621*(48*a+56)))*t^7+((1/54621*(-3*a-7)*i+1/54621*(-25*a+77))*r1+(1/18207*(a-$
$35) * i+1/54621 * (153 * a - 413))) * t^6 + ((1/54621 * (122 * a - 217) * i+1/54621 * (-18 * a + 49)) * r1 + (1/54621 * (188 * a - 539) * i +$
$1/54621 * (-72 * a - 133))) * t^5 + ((1/31212 * (3 * a + 5) * i+1/218484 * (193 * a - 595)) * r1 + (1/218484 * (-33 * a + 665) * i + 1/31212 *$
$(-155 * a + 433))) * t^4 + ((1/218484 * (-208 * a + 343) * i + 1/218484 * (27 * a - 98)) * r1 + (1/72828 * (-43 * a + 210) * i + 1/218484 *$
$(108 * a + 385))) * t^3 + ((1/62424 * (-3 * a - 1) * i + 1/218484 * (-116 * a + 357)) * r1 + (1/145656 * (19 * a - 189) * i + 1/109242 * (293 *$
$a - 854))) * t^2 + ((1/1747872 * (159 * a - 203) * i + 1/582624 * (-5 * a + 35)) * r1 + (1/1747872 * (-213 * a - 91) * i + 1/249696 * (-9 * a -$

$$85))) * t + ((1/582624 * (4 * a - 7) * i + 1/582624 * (55 * a - 168)) * r1 + (1/102816 * (-3 * a + 14) * i + 1/1747872 * (-730 * a + 2233))))) *$$
$$b2^2 + ((((1/40251494791862835497604 * (-26668614955502333757 * a + 67460687263396561505) * i + 1/40251494791862835497604 *$$
$$(9781467804429808467 * a - 2990858376852847 4401)) * r1 + (1/78924499591887912 7404 * (82112678395257719 9 * a - 2273240715981201575) *$$
$$i + 1/23677349877566373822 12 * (4083789220144608977 * a - 1138711308187510350 3))) * t^7 + ((1/40251494791862835497604 *$$
$$(20977637993648674015 * a - 5357695331667489581 4) * i + 1/40251494791862835497604 * (1294789486801990511 0 * a - 3553133259256954341 5)) *$$
$$r1 + (1/23677349877566373822 12 * (-5796923552222173722 * a + 15860277085904774131) * i + 1/23677349877566373822 12 * (-8329117713166221 55 *$$
$$a + 1924553985850228544))) * t^6 + ((1/161005979167451341990416 * (184297711718893317925 * a - 468243968023893533311) *$$
$$i + 1/230008541666778763141488 * (-7726909798866667313 * a + 24212808102084260609)) * r1 + (1/9470939951026549528848 *$$
$$(-20672135897191806465 * a + 56139453957315145487) * i + 1/9470939951026549528848 * (-25133226571996146421 * a + 70035603053968116873))) *$$
$$t^5 + ((1/10062873697965708874401 * (-9932815368237247334 * a + 25693163503862509571) * i + 1/53668659722483780663472 *$$
$$(-29598191191311698257 * a + 80334491546648266921)) * r1 + (1/47354699755132747644 24 * (21485603120563117365 * a - 58303269454142468714) *$$
$$i + 1/9470939951026549528848 * (7793201801646646169 * a - 194722961551503018 55))) * t^4 + ((1/64402391666980536796166 4 *$$
$$(-37077159049571430254 9 * a + 95010365248397018646 1) * i + 1/64402391666980536796166 4 * (52159185916895900943 * a - 18122845914548074044 1)) *$$
$$r1 + (1/37883759804106198115392 * (56147535445649437879 * a - 149077628707238221011) * i + 1/37883759804106198115392 *$$
$$(38915677456985855847 * a - 107243491022275371869))) * t^3 + ((1/64402391666980536796166 4 * (368687628088719543258 * a -$$
$$969062928835078184135) * i + 1/64402391666980536796166 4 * (174588099820872102360 * a - 468686774806364875789)) * r1 +$$
$$(1/54119656863008854450 56 * (-13689330232157816399 * a + 36704317980666791478) * i + 1/37883759804106198115392 * (-23743038330148504837 *$$
$$a + 62029181537750565992))) * t^2 + ((1/25760956666792214718466 56 * (212541558500496948827 * a - 551162345517534750013) *$$
$$i + 1/25760956666792214718466 56 * (25292147183569401133 * a - 50005068649491275481)) * r1 + (1/50511679738808264153856 *$$
$$(-15184766560512792353 * a + 39680132184911937915) * i + 1/15153503921642479246 1568 * (-11688164992237182009 * a + 31319753064529686841))) *$$
$$t + ((1/51521913333584429436933 12 * (-49513417794776030348 1 * a + 13132580815144783530 65) * i + 1/73602733333692042052 7616 *$$
$$(-26549512686497407405 * a + 70445865326367419955)) * r1 + (1/43295725490407083560448 * (17317654394062472379 * a - 46041866837208406835) *$$
$$i + 1/10102335947761652830771 2 * (13285809660310242585 * a - 35291833820492824463)))) * b1 + (((1/54621 * (340 * a - 1484) * i +$$
$$1/3213 * (-12 * a - 28)) * r1 + (1/3213 * (40 * a + 56) * i + 1/3213 * (-48 * a + 56))) * t^7 + ((1/3213 * (3 * a - 7) * i + 1/54621 * (-65 * a -$$
$$77)) * r1 + (1/1071 * (-a - 35) * i + 1/3213 * (-111 * a + 301))) * t^6 + ((1/54621 * (-548 * a + 2611) * i + 1/3213 * (18 * a + 49)) * r1 +$$
$$(1/3213 * (-34 * a - 203) * i + 1/3213 * (72 * a - 133))) * t^5 + ((1/1836 * (-3 * a + 5) * i + 1/218484 * (317 * a + 1015)) * r1 + (1/12852 * (33 *$$
$$a + 665) * i + 1/1836 * (97 * a - 269))) * t^4 + ((1/218484 * (862 * a - 5089) * i + 1/12852 * (-27 * a - 98)) * r1 + (1/4284 * (-13 * a + 252) *$$
$$i + 1/12852 * (-108 * a + 385))) * t^3 + ((1/3672 * (3 * a - 1) * i + 1/218484 * (26 * a - 1071)) * r1 + (1/8568 * (-19 * a - 189) * i + 1/6426 *$$
$$(-139 * a + 406))) * t^2 + ((1/1747872 * (-393 * a + 4907) * i + 1/34272 * (5 * a + 35)) * r1 + (1/102816 * (171 * a - 1309) * i + 1/14688 * (9 *$$
$$a - 85))) * t + ((1/34272 * (-4 * a - 7) * i + 1/582624 * (-123 * a + 742)) * r1 + (1/6048 * (3 * a + 14) * i + 1/102816 * (212 * a - 707))))) *$$
$$b2 + (((1/1183867493878318691106 * (-3304388629547666184 9 * a + 88425032761705684261) * i + 1/1183867493878318691106 *$$
$$(-73216733661406961457 * a + 194184122150006571163)) * r1 + (1/394622497959439563702 * (-64651790880738418829 * a +$$
$$174793685139920192633) * i + 1/1183867493878318691106 * (-29234002164904149358 3 * a + 782195292827846440725))) * t^7 +$$
$$((1/23677349877566373822 12 * (-49526843796303232067 * a + 13011090257329303820 1) * i + 1/23677349877566373822 12 * (-44421956170224168133 *$$

$a + 113504353829151151993)) * r1 + (1/2367734987756637382212 * (-26606413841192490129 * a + 80206998292782108251) * i +$

$1/2367734987756637382212 * (-143340865101000888695 * a + 378431337229287886411))) * t^6 + ((1/4735469975513274764424 *$

$(186674649688499237687*a-500937783543777401255)*i+1/6764957107876106806 32*(625526847824398122 65*a-165700975223659928399))*$

$r1 + (1/4735469975513274764424 * (119580152575081554317 7 * a - 3240358921401062075293) * i + 1/4735469975513274764424 *$

$(175312069931034422678 9 * a - 4689704649663121217199))) * t^5 + ((1/2367734987756637382212 * (83170139402601768403 * a -$

$217160091690094295092)*i+1/7892449959188791274 04*(22550044224142729709*a-57486085638833442654))*r1+(1/1183867493878318691106*$

$(-281995940207693606 4*a-252965450991318751)*i+1/2367734987756637382212*(20895283976995539507 7*a-55610059305064822063 1)))*$

$t^4 + ((1/947093995102654952 8848 * (-104044872396228041378 * a + 280853513326931848429) * i + 1/947093995102654952 8848 *$

$(-301677897249956231577 * a + 796869297535656562616)) * r1 + (1/947093995102654952 8848 * (-900747771500318789596 * a +$

$245148021115852004 4153) * i + 1/947093995102654952 8848 * (-123323765607665911 5657 * a + 330197749449649872 3710))) * t^3 +$

$((1/378837598041061981 15392 * (-616903704204500614299 * a + 159656756040355056 0469) * i + 1/378837598041061981 15392 *$

$(-394605773031132993081 * a + 995449846265571941321)) * r1 + (1/541196568630088544 5056 * (108926902080010549733 * a -$

$271848734784861578775) * i + 1/378837598041061981 15392 * (-101831208175844761 6475 * a + 276868958190509804 1463))) * t^2 +$

$((1/757675196082123962 30784*(-389722957102215242 81*a+103940361014413569415)*i+1/757675196082123962 30784*(844427121400898689 97*$

$a - 2158908307854723725 37))) * r1 + (1/841861328980137735 8976 * (514799638570202371 21 * a - 140871783894345144151) * i +$

$1/757675196082123962 30784 * (453786801244929492435 * a - 12157118086453310760 23))) * t + ((1/151535039216424792461568 *$

$(306010806009230971533*a-788216557847512798445)*i+1/216478627452035417 80224*(108396710215180632 43*a-256385992091776221 69))*$

$r1 + (1/216478627452035417 80224 * (-141456680858966762847 * a + 363191954197234799275) * i + 1/505116797388082641 53856 *$

$(-8341734540848507287 * a - 544715424553655371 1))))) * b1 + ((1/3213 * (3508 * a - 9716) * i + 1/3213 * (-828 * a + 2212)) * r1 +$

$(1/189 * (220 * a - 700) * i + 1/189 * (-180 * a + 140))) * t^7 + ((1/3213 * (522 * a - 1022) * i + 1/3213 * (1102 * a - 2870)) * r1 + (1/63 *$

$(-30 * a + 154) * i + 1/189 * (-222 * a + 742))) * t^6 + ((1/3213 * (-5828 * a + 15946) * i + 1/3213 * (1242 * a - 3556)) * r1 + (1/189 * (-208 *$

$a + 742) * i + 1/189 * (270 * a - 112))) * t^5 + ((1/918 * (-279 * a + 599) * i + 1/6426 * (-4121 * a + 11291)) * r1 + (1/378 * (369 * a - 1673) *$

$i + 1/54 * (119 * a - 397))) * t^4 + ((1/12852 * (10747 * a - 28777) * i + 1/12852 * (-1863 * a + 6167)) * r1 + (1/252 * (-61 * a + 63) * i +$

$1/756 * (-405 * a - 203))) * t^3 + ((1/1836 * (315 * a - 775) * i + 1/12852 * (4637 * a - 13503)) * r1 + (1/84 * (-53 * a + 203) * i + 1/756 *$

$(-983 * a + 3227))) * t^2 + ((1/25704 * (-2517 * a + 6503) * i + 1/8568 * (81 * a - 413)) * r1 + (1/1512 * (285 * a - 721) * i + 1/216 * (9 * a +$

$29))) * t + (1/2016 * (-57 * a + 161) * i + 1/34272 * (-2045 * a + 6335)) * r1 + 1/6048 * (711 * a - 2303) * i + 1/6048 * (1355 * a - 4361),$

$w_4 := ((((1/40251494791862835497604 * (8380315164608538279 * a - 22366597102958454563) * i + 1/40251494791862835497604 *$

$(5435985856599437933*a-14513734320895808355))*r1+(1/57502135416946907853 72*(49063067635231512 07*a-13026151848980975083)*$

$i + 1/40251494791862835497604 * (21985414533050555375 * a - 577191660677297886 01))) * t^7 + ((1/11500427083389381570744 *$

$(783039152120492155*a-2071637405661193221)*i+1/47354699755132747644 24*(17469571720207406 9*a-463993767105792301))*$

$r1 + (1/805029895837256709952 08 * (19932584642733319981 * a - 52556809040762058151) * i + 1/805029895837256709952 08 *$

$(87238633640954446 97 * a - 23022845586309193205))) * t^6 + ((1/161005979167451341990416 * (-50157088747399926179 * a +$

$133977617936173260431) * i + 1/161005979167451341990416 * (-32821220569229978407 * a + 877521263647730253 37)) * r1 +$

$(1/230008541667787631414 88*(-28952545846974243387*a+76834510043688670651)*i+1/161005979167451341990416*(-130872699284884695781*$

$a + 342139666213163046655))) * t^5 + ((1/16100597916745134199041 6 * (-16253244734770420611 * a + 43024471525909459726) *$

$i + 1/16100597916745134199041 6 * (-91495772901510861 25 * a + 24308938383571033864)) * r1 + (1/16100597916745134199041 6 *$

$(-58167387594748854593 * a + 153306351232951930916) * i + 1/16100597916745134199041 6 * (-24015444435941802859 * a +$

$63430955627994598966))) * t^4 + ((1/21467463888993512265388 8 * (23405051481658292591 * a - 62660324955092784409) * i +$

$1/64402391666980536796166 4 * (46993467568913142215 * a - 126188066235378311199)) * r1 + (1/21467463888993512265388 8 *$

$(90464671722043415319*a-239654461617409622665)*i+1/64402391666980536796166 4*(18030945492325846561 5*a-46541623400125035034 3)))*$

$t^3 + ((1/64402391666980536796166 4 * (22482844411475276875 * a - 59606586748969961225) * i + 1/80502989583725670995208 *$

$(1726693643224866822*a-459939888345870063))*r1+(1/32201195833490268398083 2*(37168778387633002693*a-97785633823987027051)*$

$i+1/64402391666980536796166 4 * (24970365596935869605 * a - 66273499047346451049))) * t^2 + ((1/12880478333396107359233 28 *$

$(-6229419319272303827*a+16869392026923726206)*i+1/12880478333396107359233 28*(-4843829085385891142*a+13435263380913215709))*$

$r1 + (1/12880478333396107359233 28 * (-15106337790916477241 * a + 38920513679675258554) * i + 1/12880478333396107359233 28 *$

$(-14936337418450473570 * a + 35300904236470902911))) * t + ((1/5152191333358442943693312 * (-7774038834413195759 * a +$

$208727607673697699 65)*i+1/5152191333358442943693312*(-6341203240114484233*a+17062630561299896685))*r1+(1/5152191333358442943693312*$

$(-44473686884655783205*a+107189294518144442 59)*i+1/5152191333358442943693312*(14561349558998591121*a-38446456426898195473))))*$

$b1 + ((1/54621 * (-4 * a + 28) * i * r1 + (1/54621 * (248 * a - 504) * i + 1/54621 * (-12 * a + 140))) * t^7 + ((1/18207 * (a + 7) * i +$

$1/18207 * (-3 * a - 7)) * r1 + (1/18207 * (-a - 7) * i + 1/7803 * (-a + 19))) * t^6 + ((1/109242 * (15 * a - 91) * i + 1/109242 * (3 *$

$a + 7)) * r1 + (1/109242 * (-891 * a + 1855) * i + 1/36414 * (19 * a - 161))) * t^5 + ((1/10404 * (-a - 7) * i + 1/218484 * (69 * a +$

$133)) * r1 + (1/10404 * (a + 11) * i + 1/218484 * (-69 * a - 721))) * t^4 + ((1/31212 * (-4 * a + 15) * i + 1/72828 * (-2 * a - 7)) * r1 +$

$(1/31212 * (133 * a - 286) * i + 1/218484 * (-75 * a + 490))) * t^3 + ((1/62424 * (3 * a + 22) * i + 1/436968 * (-80 * a - 119)) * r1 +$

$(1/436968 * (-27 * a - 364) * i + 1/436968 * (234 * a + 497))) * t^2 + ((1/1747872 * (81 * a - 203) * i + 1/1747872 * (3 * a + 49)) * r1 +$

$(1/1747872 * (-1019 * a + 2205) * i + 1/1747872 * (75 * a - 595))) * t + ((1/166464 * (-a - 9) * i + 1/3495744 * (121 * a + 91)) * r1 +$

$(1/205632 * (3 * a + 35) * i + 1/3495744 * (-423 * a - 413))))) * b2^2 + ((((1/40251494791862835497604 * (30807661735178699799 *$

$a - 85731372270206942461) * i + 1/40251494791862835497604 * (35284312758355798975 * a - 87894591055764425583)) * r1 +$

$(1/3382478553938053403 16*(-968877002837125493*a+2348320438354559255)*i+1/2367734987756637382212*(6519345082271921825*$

$a - 16750854902062742881))) * t^7 + ((1/11500427083389381570744 * (-3270596659169097211 * a + 9471098440191305343) * i +$

$1/80502989583725670995208*(-79680116746792401 79*a+16051952586388246913))*r1+(1/47354699755132747644 24*(70966351907024549 95*$

$a-16982064644454395899)*i+1/47354699755132747644 24*(96716275069301068 73*a-25566976631056449407)))*t^6+((1/16100597916745134199041 6*$

$(-22723996869075428735 5 * a + 624127963118160353797) * i + 1/16100597916745134199041 6 * (-21074087158706376268 1 * a +$

$52354751586816333638 9))*r1+(1/13529914215752213612 64*(73024013833287709 05*a-18026837494017632267)*i+1/94709399510265495288 48*$

$(-38708418830387061127 * a + 100605952881110204143))) * t^5 + ((1/16100597916745134199041 6 * (80599962787556169708 * a -$

$22923907234864664389 9)*i+1/16100597916745134199041 6*(42736787776107830296*a-95033772914290026607))*r1+(1/94709399510265495288 48*$

$(-24047930777703114104*a+58116626822317420553)*i+1/94709399510265495288 48*(-33038394001035451720*a+86948300155900133293)))*$

$t^4 + ((1/21467463888993512265388 8 * (16750776871675795171 9 * a - 450066740295488710219) * i + 1/64402391666980536796166 4 *$

$(296344274628820659217 * a - 728784983587224027051)) * r1 + (1/12627919934702066038464 * (-38907941691794378621 * a +$

$$9871551643998874988 9)*i+1/37883759804106198115392*(52260289119018424893*a-140121890951680541407)))*t^3+((1/32201195833490268398083 2*(-86448074755979524013*a+237082738236626397010)*i+1/64402391666980536796166 4*(-142441985031666684531*a+339538386265475188183))*r1+(1/37883759804106198115392*(47730457565061553523*a-118347387593948704709)*i+1/18941879902053099057696*(32220824260360816291*a-84068803511147469462)))*t^2+((1/12880478333396107359233 28*(-153897267029538411502*a+402299785950350937985)*i+1/12880478333396107359233 28*(-27009108636344698981*a+61509914068433529006))*r1+(1/75767519608212396230784*(37436179435380238558*a-97750756674900337421)*i+1/75767519608212396230784*(-3606367432277986167*a+12224453181959684498)))*t+((1/51521913333584429436933 12*(219862431684114959165*a-577782670317269627833)*i+1/51521913333584429436933 12*(259660982292601655317*a-652008134752865498895))*r1+(1/30307007843284958492313 6*(-54974856042951657445*a+141798107807861506309)*i+1/30307007843284958492313 6*(-68476717320754676433*a+174799311240282076943))))*b1+((1/54621*(-212*a+308)*i*r1+(1/3213*(-136*a+504)*i+1/3213*(12*a+140)))*t^7+((1/1071*(-a+7)*i+1/18207*(233*a-581))*r1+(1/1071*(a-7)*i+1/459*(-13*a+25)))*t^6+((1/109242*(711*a-581)*i+1/6426*(-3*a+7))*r1+(1/6426*(429*a-1631)*i+1/2142*(-19*a-161)))*t^5+((1/612*(a-7)*i+1/218484*(-5079*a+12551))*r1+(1/612*(-a+11)*i+1/12852*(783*a-1603)))*t^4+((1/31212*(-86*a-57)*i+1/4284*(2*a-7))*r1+(1/1836*(-53*a+212)*i+1/12852*(75*a+490)))*t^3+((1/3672*(-3*a+22)*i+1/436968*(5546*a-13447))*r1+(1/25704*(27*a-364)*i+1/25704*(-1032*a+2261)))*t^2+((1/1747872*(429*a+1967)*i+1/102816*(-3*a+49))*r1+(1/102816*(361*a-1617)*i+1/102816*(-75*a-595)))*t+((1/9792*(a-9)*i+1/3495744*(-6943*a+16709))*r1+(1/12096*(-3*a+35)*i+1/205632*(1473*a-3395)))))*b2+(((1/11838674938783186911 06*(-224622182243001417069*a+596235361853125562155)*i+1/11838674938783186911 06*(-2214958461483411835*a+8262259922987438235))*r1+(1/16912392769690267015 8*(-93264090957705162461*a+245424509679682429727)*i+1/11838674938783186911 06*(-58010317928075194921*a+141459703365608814389)))*t^7+((1/84561963848451335079*(-2649285923679505757*a+7074882934392015510)*i+1/59193374693915934555 3*(436757852402827841*a-119556178925496698 42))*r1+(1/59193374693915934555 3*(-130481410306157773775*a+346922125605080731013)*i+1/59193374693915934555 3*(-5317758823880746300*a+16615213391546515270)))*t^6+((1/47354699755132747644 24*(139428142250868121793 5*a-370405215037790816961 7)*i+1/47354699755132747644 24*(2167294152811874693*a-23557784327034005909))*r1+(1/67649571078761068063 2*(524843285455817243691*a-137887268749259227054 1)*i+1/47354699755132747644 24*(42917165744554758830 3*a-103547307191545894264 7)))*t^5+((1/47354699755132747644 24*(203350727387663999061*a-543491207610719333672)*i+1/47354699755132747644 24*(-60562494426831547042*a+167243485741599499267))*r1+(1/47354699755132747644 24*(159232740132523104675 1*a-423454600146277151926 6)*i+1/47354699755132747644 24*(10101791735495611502 0*a-306245439844945968899)))*t^4+((1/31569799836755165096 16*(-360885623522930466545*a+960806446356755786907)*i+1/94709399510265495288 48*(21930978233975155249*a-377064187679605036 59))*r1+(1/31569799836755165096 16*(-689753090541346842749*a+1799705432382374021827)*i+1/94709399510265495288 48*(-472262593834681393767*a+112347139980110567293 3)))*t^3+((1/37883759804106198115392*(-402437395822793134681*a+108017929759533040757 3)*i+1/37883759804106198115392*(248708450605054413087*a-694512984257071703297))*r1+(1/37883759804106198115392*(-4790424598055087136995*a+12739918114034615452067)*i+1/37883759804106198115392*(-629983276864865116895*a+184765120997094951568 5)))*t^2+((1/37883759804106198115392*(321848447737188643141*a-863475058251496035352)*

$$i + 1/37883759804106198115392 * (-29850111279244445516 * a + 64263272018296691445)) * r1 + (1/37883759804106198115392 *$$
$$(-24194630229940378589 3*a+690813906598386552856)*i+1/37883759804106198115392*(3227133155348787910 44*a-769760346935915308813)))*$$
$$t+((1/151535039216424792461568*(-153465333892354632695*a+402118264634914899379)*i+1/151535039216424792461568*$$
$$(-164690730267401534659*a+460581418424528611077))*r1+(1/151535039216424792461568*(122092734731332259 0219*a-$$
$$323873078499925174 9823)*i+1/151535039216424792461568*(546850208593781953047*a-1559888331136511808649))))*b1+$$
$$((1/3213*(388*a-532)*i+1/51*(-4*a+12))*r1+(1/189*(-580*a+1764)*i+1/189*(108*a-532)))*t^7+((1/1071*(-174*$$
$$a+182)*i+1/1071*(-38*a+490))*r1+(1/63*(30*a-70)*i+1/27*(14*a-98)))*t^6+((1/3213*(-843*a+1631)*i+1/3213*$$
$$(369*a-1379))*r1+(1/189*(1059*a-3199)*i+1/63*(-75*a+329)))*t^5+((1/306*(87*a-91)*i+1/6426*(423*a-5705))*$$
$$r1+(1/18*(-15*a+31)*i+1/378*(-267*a+2233)))*t^4+((1/1836*(379*a-945)*i+1/4284*(-219*a+1099))*r1+(1/108*$$
$$(-323*a+965)*i+1/756*(549*a-2177)))*t^3+((1/1836*(-270*a+283)*i+1/12852*(-509*a+6916))*r1+(1/756*(342*a-$$
$$581)*i+1/756*(165*a-2072)))*t^2+((1/25704*(-1392*a+3773)*i+1/25704*(279*a-1330))*r1+(1/1512*(646*a-1911)*i+$$
$$1/1512*(-153*a+644)))*t+(1/48*(a-1)*i+1/51408*(137*a-4123))*r1+1/3024*(-225*a+301)*i+1/3024*(-51*a+1141),$$
$$w_5 := ((((1/40251494791862835497604*(-674033425632271521*a+1900198024318121773)*i+1/40251494791862835497604*$$
$$(72657211641582544 39*a-193383920533841928 17))*r1+(1/57502135416946907853 72*(-113287310298741508 1*a+306456183654416403 3)*$$
$$i+1/40251494791862835497604*(40619384595179013573*a-108400130546350466327)))*t^7+((1/201257473959314177488 02*$$
$$(-110440365744694896*a+300694141821905671)*i+1/40251494791862835497604*(39698726244151701 05*a-1034042178216298664 7))*$$
$$r1+(1/40251494791862835497604*(150030014016208258 9*a-4024995918687960097)*i+1/20125747395931417748802*(4497476319659542848*$$
$$a-12135703201784797691)))*t^6+((1/53668659722483780663472*(1063315432791931359*a-3061481008961538443)*i+$$
$$1/76669513889262543804 96*(-194878090472263290 5*a+5186176767239431639))*r1+(1/16100597916745134199041 6*(54227905191839469943*$$
$$a-14644744097273906 4627)*i+1/16100597916745134199041 6*(-24867043196952287329 9*a+663754729510800397981)))*$$
$$t^5+((1/16100597916745134199041 6*(1431629784237178489*a-3867523982580944315)*i+1/80502989583725670995208*$$
$$(-12329932737987081961*a+32017123813304168028))*r1+(1/16100597916745134199041 6*(-9917617575125054089*a+$$
$$26799777970317582895)*i+1/80502989583725670995208*(-24032539020479908722*a+65350561394417463353)))*t^4+((1/64402391666980536796166 4*$$
$$(-68854726266768134 9*a+29222509928509499 07)*i+1/64402391666980536796166 4*(46198892424600344067*a-12282510765848001076 3))*$$
$$r1+(1/64402391666980536796166 4*(-10446040435324099356 9*a+28075306781993072599 9)*i+1/64402391666980536796166 4*$$
$$(37280706010399600211 1*a-995420638845636114503)))*t^3+((1/64402391666980536796166 4*(-2647399668083339259*a+$$
$$7138334686256546500)*i+1/64402391666980536796166 4*(38597364672695304976*a-99470285944966822319))*r1+(1/64402391666980536796166 4*$$
$$(18643438653960264888*a-50591327090447808371)*i+1/64402391666980536796166 4*(42334632061148764675*a-12017866269877998843 6)))*$$
$$t^2+((1/85869855555974049061 5552*(-1558759685040474471*a+4085779251770619613)*i+1/25760956666792219471846656*$$
$$(64447890982294633 89*a-17340562131907523341))*r1+(1/36801366666846021026 3808*(79110070393106330 71*a-21014462814708170145)*$$
$$i+1/85869855555974049061 5552*(-31383403182697820647*a+83821260210955143275)))*t+((1/51521913333584429436933 12*$$
$$(260706818449797473 1*a-718060437579085477 1)*i+1/51521913333584429436933 12*(-23267505224017332479*a+58489400923944796449))*$$
$$r1+(1/51521913333584429436933 12*(-17944133300586717491*a+48108370785736123415)*i+1/51521913333584429436933 12*$$
$$(48180668699473834679*a-115934911358154878045))))*b1+(((1/54621*(-64*a+196)*i+1/54621*(24*a+28))*r1+$$

$$(1/7803*(-28*a+56)*i+1/7803*(12*a+8)))*t^7+((1/18207*(a-7)*i+1/54621*(53*a-133))*r1+(1/18207*(-a+7)*i+1/18207*(-39*a+119)))*t^6+((1/54621*(101*a-315)*i+1/18207*(-12*a-14))*r1+(1/54621*(255*a-455)*i+1/18207*(-40*a-28)))*t^5+((1/31212*(-3*a+19)*i+1/218484*(-373*a+959))*r1+(1/218484*(33*a-217)*i+1/218484*(961*a-2891)))*t^4+((1/109242*(-81*a+266)*i+1/72828*(17*a+21))*r1+(1/218484*(-191*a+63)*i+1/12138*(8*a+7)))*t^3+((1/62424*(3*a-17)*i+1/109242*(97*a-259))*r1+(1/436968*(-57*a+329)*i+1/218484*(-620*a+1827)))*t^2+((1/1747872*(117*a-427)*i+1/249696*(-3*a-5))*r1+(1/1747872*(-359*a+1421)*i+1/1747872*(3*a-119)))*t+((1/582624*(-4*a+21)*i+1/1747872*(-230*a+651))*r1+(1/102816*(3*a-14)*i+1/1747872*(919*a-2646)))))*b2^2+((((1/40251494791862835497604*(-75682478385681058569*a+194384932148779404515)*i+1/40251494791862835497604*(39093768640790472125*a-97838307524241123973))*r1+(1/3382478553938053403 16*(100755574352845 2643*a-2733427259083459485)*i+1/2367734987756637382212*(1923569781793618275*a-251975030446629835 1)))*t^7+((1/10062873697965708874401*(9402802630314783828*a-24758494939608500546)*i+1/40251494791862835497604*(-6099602394957498443*a+12341466215653865997))*r1+(1/2367734987756637382212*(-12089833773116559755*a+3161044383498785 7875)*i+1/11838674938783186911 06*(283718343664305 1539*a-7689968710366341 47)))*t^6+((1/53668659722483780663472*(17636835452689534 5107*a-45340760040050629 7385)*i+1/76669513889262543804 96*(-11924733825503476919*a+30003375797873185359))*r1+(1/9470939951026549528848*(-60531312736678275785*a+162670954415410480851)*i+1/9470939951026549528848*(-6469185517498630609*a+109188432524739505)))*t^5+((1/16100597916745 1341990416*(-28501208346554537 0089*a+748141417125576405 0513)*i+1/80502895837256709952 08*(29219048603610737332*a-64394128521869479515))*r1+(1/9470939951026549528848*(86445072201625782779*a-22623595804441366 2515)*i+1/47354699755132747644 24*(-19448201810627135736*a+52599103284613058495)))*t^4+((1/64402391666980536796166 4*(-10771346625173419537 99*a+27764075693248262713 59)*i+1/64402391666980536796166 4*(42311288959158457334 7*a-10853266472929696870 1))*r1+(1/37883759804106198115392*(16583851856019590648 57*a-44056461586821318431 3)*i+1/37883759804106198115392*(-13349091739789173813*a+64421257060111146475)))*t^3+((1/64402391666980536796166 4*(66473779559828334690 97*a-17412492377710281873 70)*i+1/64402391666980536796166 4*(-17676548586111542846 8*a+42357198139390872304 7))*r1+(1/37883759804106198115392*(-18376490247149494331 4*a+48162066995760466821 1)*i+1/37883759804106198115392*(76278793310361647815*a-20398220895763827 4632)))*t^2+((1/85869855555974049061555 2*(20480712268331980282 7*a-52989505267777317017 1)*i+1/25760956666792214718466 56*(-15935062213654628262 3*a+43499929954679632321 3))*r1+(1/21647862745203541780224*(-18831609471271637279*a+49627439071557450435)*i+1/50511679738808264153856*(10112859229628056777*a-30195854621708950559)))*t+((1/51521913333584429436933 12*(-89971503873042958683 9*a+23532537804986104551 65)*i+1/51521913333584429436933 12*(30023158375705136933 3*a-76452470283640633570 5))*r1+(1/30307007843284958492313 6*(22187208214137324300 7*a-58140152227788234748 9)*i+1/30307007843284958492313 6*(-81250230635238337261*a+211836334457166972973))))*b1+(((1/54621*(-536*a+1988)*i+1/3213*(-24*a+28))*r1+(1/459*(-4*a+32)*i+1/459*(-12*a+8)))*t^7+((1/1071*(-a-7)*i+1/54621*(-635*a+1645))*r1+(1/1071*(a+7)*i+1/1071*(25*a-63)))*t^6+((1/54621*(817*a-2961)*i+1/1071*(12*a-14))*r1+(1/3213*(39*a-329)*i+1/1071*(40*a-28)))*t^5+((1/1836*(3*a+19)*i+1/218484*(4255*a-11123))*r1+(1/12852*(-33*a-217)*i+1/12852*(-611*a+1519)))*t^4+((1/109242*(-639*a+2128)*i+1/4284*(-17*a+21))*r1+(1/12852*(-47*a+441)*$$

$$i+1/714*(-8*a+7)))*t^3+((1/3672*(-3*a-17)*i+1/109242*(-1033*a+2737))*r1+(1/25704*(57*a+329)*i+1/12852*(382*a-945)))*t^2+((1/1747872*(825*a-1673)*i+1/14688*(3*a-5))*r1+(1/102816*(37*a-217)*i+1/102816*(-3*a-119)))*t+((1/34272*(4*a+21)*i+1/1747872*(2300*a-5985))*r1+(1/6048*(-3*a-14)*i+1/102816*(-527*a+1344)))))*b2+(((1/1183867493878318691106*(-50115412346638069983*a+128430153952905160825)*i+1/1183867493878318691106*(-202165113215354068193*a+535955793835177542001))*r1+(1/169123927696902670158*(-90099641321111347427*a+237826346594969026125)*i+1/1183867493878318691106*(-761681081852264191563*a+2044233329704240729075)))*t^7+((1/23677349877566373822 12*(-1883654630224033632 09*a+498271150436731064113)*i+1/23677349877566373822 12*(-93595921589653893701*a+240715384901364419907))*r1+(1/2367734987756637382212*(-4323509377046953822 3*a+116008143679503557251)*i+1/2367734987756637382212*(-606269534174930279103*a+1615972021096366971373)))*t^6+((1/15784899918377582548 08*(703075861483190340 17*a-176528146978021807171)*i+1/22549857026253689354 4*(57507282533045113821*a-152290226860461524041))*r1+(1/4735469975513274764424*(3993871822922867427013*a-105478754302042475106 27)*i+1/4735469975513274764424*(451693785011539779519 5*a-121449245767911862832 99)))*t^5+((1/4735469975513274764424*(623918917980593019112*a-165126020841473348167 7)*i+1/4735469975513274764424*(252336471226243011025*a-642730284805052428740))*r1+(1/4735469975513274764424*(-138123263369366414878*a+357353875156663427665)*i+1/4735469975513274764424*(188554253656951442888 7*a-503466803951393470164 2)))*t^4+((1/9470939951026549528848*(47533004406377040029*a-151878656694465979320)*i+1/9470939951026549528848*(-843111012384535007172*a+222378144017360304813 7))*r1+(1/9470939951026549528848*(-329323740574269933820 5*a+871362640307949523517 0)*i+1/9470939951026549528848*(-306846801818162146767 0*a+829302447046441004300 3)))*t^3+((1/3788375980410619811539 2*(-228164170991861080248 9*a+604042338093950021816 9)*i+1/3788375980410619811539 2*(-463485896501622610 45*a+112466902446436633295 9))*r1+(1/3788375980410619811539 2*(273007174247218058685 3*a-719453051307858679590 1)*i+1/3788375980410619811539 2*(-589017959195644348735 9*a+158211499162342140456 69)))*t^2+((1/25255839869404132076928*(-173772324333345263863*a+467123884449108480419)*i+1/7576751960821239623078 4*(278426322065480039571*a-709611689223206784385))*r1+(1/10823931372601770890112*(337966969633291555980 7*a-899626967165843702403)*i+1/8418613289801377358976*(92444769349882495561*a-260596006981851906939)))*t+((1/15153503921642479246156 8*(106329253321706313890 7*a-281476915041264306305 9)*i+1/15153503921642479246156 8*(-234476776400706743525*a+673277087651235615471))*r1+(1/15153503921642479246156 8*(-343866496520480302421 9*a+910271748896152592303 9)*i+1/15153503921642479246156 8*(177838016578010689669 7*a-489753256434179581379 9)))))*b1+((1/3213*(2596*a-7084)*i+1/3213*(-1908*a+2324))*r1+(1/27*(68*a-124)*i+1/27*(-36*a+20)))*t^7+((1/1071*(-174*a+658)*i+1/3213*(-1382*a+3766))*r1+(1/63*(30*a-98)*i+1/63*(50*a-98)))*t^6+((1/3213*(-4064*a+11340)*i+1/1071*(996*a-1288))*r1+(1/189*(-636*a+1036)*i+1/63*(108*a-28)))*t^5+((1/918*(279*a-1007)*i+1/6426*(4913*a-13363))*r1+(1/378*(-369*a+1225)*i+1/378*(-625*a+1379)))*t^4+((1/12852*(6453*a-19033)*i+1/476*(-177*a+259))*r1+(1/756*(571*a-315)*i+1/84*(-25*a-49)))*t^3+((1/1836*(-315*a+1081)*i+1/12852*(-5147*a+13979))*r1+(1/756*(477*a-1589)*i+1/756*(841*a-2079)))*t^2+((1/25704*(-1194*a+3997)*i+1/3672*(117*a-232))*r1+(1/1512*(148*a-679)*i+1/1512*(-117*a+406)))*t+(1/672*(19*a-63)*i+1/102816*(5867*a-15981))*r1+1/6048*(-711*a+2303)*i+1/6048*(-1355*a+3633),$$

$$w_6 := ((((1/40251494791862835497604*(3343857450724400027*a-8644965679114083921)*i+1/57502135416946907853 72*$$

$$(88165583614664017*a-199231673624024465))*r1+(1/40251494791862835497604*(38749558658278481025*a-101576180544448022035)*$$
$$i+1/40251494791862835497604*(-6115204175244660007*a+16641950197315174263)))*t^7+((1/11500427083389381570744*$$
$$(1103325360974906881*a-2969293885383787421)*i+1/80502989583725670995208*(-2614670925608197371*a+6688368021651876385))*$$
$$r1+(1/80502989583725670995208*(9924895821951194351*a-26800550910932083495)*i+1/80502989583725670995208*$$
$$(9930518079159531641*a-26393745230662573911)))*t^6+((1/16100597916745134199041 6*(-15016972322023903391*a+$$
$$38578790647798752113)*i+1/23000854166778763141488*(-712834623458405459*a+1673413739060413423))*r1+(1/16100597916745134199041 6*$$
$$(-248571091789732284081*a+65205462130368690 5843)*i+1/16100597916745134199041 6*(42062230711964926425*a-113465471058120080473)))*$$
$$t^5+((1/16100597916745134199041 6*(-24243025284109320032*a+65260502707658897849)*i+1/16100597916745134199041 6*$$
$$(971076524571287205 0*a-24881539594903321981))*r1+(1/23000854166778763141488*(-340314905482189946 8*a+917561028091510891 1)*$$
$$i+1/23000854166778763141488*(-5310103214461816862*a+14059273339268811169)))*t^4+((1/64402391666980536796166 4*$$
$$(-493381383445029365*a+260116424929159878 1)*i+1/64402391666980536796166 4*(12948981141504594327*a-31845474796133365801))*$$
$$r1+(1/64402391666980536796166 4*(41983897247241769439*a-110338505869934420938 3)*i+1/64402391666980536796166 4*$$
$$(-82828803435894230501*a+219932290146422901859)))*t^3+((1/32201195833490268398083 2*(19687282169303331536*a-$$
$$52999431179912315427)*i+1/64402391666980536796166 4*(-21612246536554767319*a+55548068420067428807))*r1+(1/64402391666980536796166 4*$$
$$(52593097017115227 91*a-13612885345635772133)*i+1/32201195833490268398083 2*(42703428080859221720*a-11245171532894301242 1)))*$$
$$t^2+((1/12880478333396107359233 28*(13344513170359051735*a-35220090412888949470)*i+1/12880478333396107359233 28*$$
$$(-5017909310546270736*a+12863931608288254333))*r1+(1/12880478333396107359233 28*(-79318495662303302837*a+$$
$$209392487529358959924)*i+1/12880478333396107359233 28*(21620671766189471070*a-55822614603167699231)))*t+((1/51521913333584429436933 12*$$
$$(-27803799254747154167*a+74992018697172634407)*i+1/51521913333584429436933 12*(27138278414784298913*a-70369382986735745759))*$$
$$r1+(1/51521913333584429436933 12*(71062726965108515447*a-19250394156160054863 5)*i+1/51521913333584429436933 12*$$
$$(-113808068335231588581*a+297102204537277500095))))*b1+((1/54621*(4*a-28)*i*r1+(1/54621*(-248*a+504)*i+$$
$$1/54621*(12*a-140)))*t^7+((1/18207*(a+7)*i+1/18207*(-3*a-7))*r1+(1/18207*(-a-7)*i+1/7803*(-a+19)))*t^6+$$
$$((1/109242*(-15*a+91)*i+1/109242*(-3*a-7))*r1+(1/109242*(891*a-1855)*i+1/36414*(-19*a+161)))*t^5+((1/10404*$$
$$(-a-7)*i+1/218484*(69*a+133))*r1+(1/10404*(a+11)*i+1/218484*(-69*a-721)))*t^4+((1/31212*(4*a-15)*i+$$
$$1/72828*(2*a+7))*r1+(1/31212*(-133*a+286)*i+1/218484*(75*a-490)))*t^3+((1/62424*(3*a+22)*i+1/436968*(-80*$$
$$a-119))*r1+(1/436968*(-27*a-364)*i+1/436968*(234*a+497)))*t^2+((1/1747872*(-81*a+203)*i+1/1747872*(-3*a-$$
$$49))*r1+(1/1747872*(1019*a-2205)*i+1/1747872*(-75*a+595)))*t+((1/166464*(-a-9)*i+1/3495744*(121*a+91))*$$
$$r1+(1/205632*(3*a+35)*i+1/3495744*(-423*a-413)))))*b2^2+((((1/40251494791862835497604*(53547195675643391413*$$
$$a-144774770711283634257)*i+1/5750213541694690785372*(4550287420828962221*a-11964067458158508547))*r1+$$
$$(1/23677349877566373822 12*(19489378077695657 1*a-23410195718519853 67)*i+1/23677349877566373822 12*(-21866206574138113 03*$$
$$a+5097599971689124977)))*t^7+((1/11500427083389381570744*(-2022135131245139743*a+5768602016356197209)*i+$$
$$1/80502989583725670995208*(-22593568839977772 45*a+9054093943560243557))*r1+(1/47354699755132747644 24*(13179022491628862903*$$
$$a-34402679261056308349)*i+1/47354699755132747644 24*(613669924265954721 5*a-16534352380173259107)))*t^6+((1/16100597916745134199041 6*$$
$$(-35880902280439557388 9*a+963393360891167129581)*i+1/23000854166778763141488*(-31267062823979257987*a+$$

$$8223713123778272522 9))*r1+(1/94709399510265495288 48*(8867155695713699781*a-1147181037351086938 9)*i+1/94709399510265495288 48*$$
$$(20814921194937243909 * a - 49590669993819800671))) * t^5 + ((1/16100597916745134199 0416 * (65846207735972776451 * a -$$
$$18257478326749957790 6)*i+1/16100597916745134199 0416*(11686571784886540911*a-42352679768646935030))*r1+(1/13529914215752213612 64*$$
$$(-640147198783741770 1*a+16743147974030208848)*i+1/13529914215752213612 64*(-277180908970950787 7*a+7500930568964872720)))*$$
$$t^4+((1/64402391666980536796 1664*(669616261627408281617*a-1770749073667483924819)*i+1/64402391666980536796 1664*$$
$$(428459506044401975931 * a - 1126915804622434629215)) * r1 + (1/37883759804106198115392 * (-56147049320902825847 * a +$$
$$127667331090605666885) * i + 1/37883759804106198115392 * (-62235046639395154613 * a + 151333993104008706985))) * t^3 +$$
$$((1/64402391666980536796 1664 * (-195508553770782020893 * a + 527758930433692452327) * i + 1/16100597916745134199 0416 *$$
$$(-10327707227081577020*a+33981975851317449991))*r1+(1/18941879902053099057696*(43061177672789603951*a-113148122967163226887)*$$
$$i+1/37883759804106198115392*(32849411673919334917*a-89489933525102514705))) * t^2 + ((1/12880478333396107359233 28 *$$
$$(-162125616051255697450 * a + 414647075408657519827) * i + 1/12880478333396107359233 28 * (-111325388005232768673 * a +$$
$$289736167182043455608))*r1+(1/75767519608212396230784*(32311471986117482128*a-81434733872863868805)*i+1/75767519608212396230784*$$
$$(25412481411698361375 * a - 63292043836349589926))) * t + ((1/51521913333584429436933 12 * (321877788804682633283 * a -$$
$$854834423788116801981) * i + 1/51521913333584429436933 12 * (81077839848797025061 * a - 244494783408850558045)) * r1 +$$
$$(1/30307007843284958492313 6 * (-868844802525482970 31 * a + 230499395709724120525) * i + 1/30307007843284958492313 6 *$$
$$(-26740350405773344581 * a + 754403404650688976 33))))) * b1 + (((1/54621 * (212 * a - 308) * i * r1 + (1/3213 * (136 * a - 504) * i +$$
$$1/3213 * (-12 * a - 140))) * t^7 + ((1/1071 * (-a + 7) * i + 1/18207 * (233 * a - 581)) * r1 + (1/1071 * (a - 7) * i + 1/459 * (-13 *$$
$$a + 25))) * t^6 + ((1/109242 * (-711 * a + 581) * i + 1/6426 * (3 * a - 7)) * r1 + (1/6426 * (-429 * a + 1631) * i + 1/2142 * (19 * a +$$
$$161))) * t^5 + ((1/612 * (a - 7) * i + 1/218484 * (-5079 * a + 12551)) * r1 + (1/612 * (-a + 11) * i + 1/12852 * (783 * a - 1603))) *$$
$$t^4 + ((1/31212 * (86 * a + 57) * i + 1/4284 * (-2 * a + 7)) * r1 + (1/1836 * (53 * a - 212) * i + 1/12852 * (-75 * a - 490))) * t^3 +$$
$$((1/3672 * (-3 * a + 22) * i + 1/436968 * (5546 * a - 13447)) * r1 + (1/25704 * (27 * a - 364) * i + 1/25704 * (-1032 * a + 2261))) * t^2 +$$
$$((1/1747872 * (-429 * a - 1967) * i + 1/102816 * (3 * a - 49)) * r1 + (1/102816 * (-361 * a + 1617) * i + 1/102816 * (75 * a + 595))) *$$
$$t + ((1/9792 * (a - 9) * i + 1/3495744 * (-6943 * a + 16709)) * r1 + (1/12096 * (-3 * a + 35) * i + 1/205632 * (1473 * a - 3395))))) *$$
$$b2 + (((1/1183867493878318691106 * (-154285129772087946577 * a + 404990438523276270189) * i + 1/169123927696902670158 *$$
$$(11878201992033712117*a-31600700882607192767))*r1+(1/1183867493878318691106*(-67617485645887523817 9*a+17648604877701441974 75)*$$
$$i+1/1183867493878318691106*(822683988083777726357*a-218794816580210583549 9))) * t^7 + ((1/16912392769690267015 8 *$$
$$(-73055035581934916 05*a+19762923021220999589)*i+1/1183867493878318691106*(81960240000014947923*a-21521639576813614909 3))*$$
$$r1 + (1/1183867493878318691106 * (-30290593491725952071 * a + 805023796995081229237) * i + 1/1183867493878318691106 *$$
$$(49857519420104324881 * a - 134953166516481684357))) * t^6 + ((1/4735469975513274764424 * (901905285437744400271 * a -$$
$$23707625941552100058 91)*i+1/6764957107876106806 32*(-58465301656707822647*a+15555723555753257122 9))*r1+(1/4735469975513274764424*$$
$$(403915303173581456574 9 * a - 10534649597922489478537) * i + 1/4735469975513274764424 * (-52347310004670280142 67 * a +$$
$$139254460996633995134 57))) * t^5 + ((1/4735469975513274764424 * (278622316118790345607 * a - 756547316563260740056) * i +$$
$$1/4735469975513274764424*(-52792531395624546895 8*a+13844790959928665804 69))*r1+(1/6764957107876106806 32*(269660968490546874773*$$
$$a - 715606248394436703328) * i + 1/676495710787610680632 * (-17707921067006898026 * a + 50358924519269706187))) * t^4 +$$

$$((1/4735469975513274764424*(-29417748295736224378 4*a+77709760200947994 2317)*i+1/4735469975513274764424*(46615602410863060989*$$

$$a-12440955563860382941))*r1+(1/4735469975513274764424*(-14028722578899293356 19*a+36511894925631134376 92)*$$

$$i+1/1183867493878318691106*(541840074270938356537*a-144241430581058969454 8)))*t^3+((1/3788375980410619811539 2*$$

$$(-559333025883038826499*a+154363678827780223378 5)*i+1/3788375980410619811539 2*(181439002895585124255 1*a-$$

$$474424279522447502567 9))*r1+(1/3788375980410619811539 2*(-590060944642940953412 5*a+155888401871283108686 75)*$$

$$i+1/3788375980410619811539 2*(-128973736875123184105 9*a+329450000343226380666 3)))*t^2+((1/3788375980410619811539 2*$$

$$(507481626221306519 92*a-144290632120237811257)*i+1/3788375980410619811539 2*(223696809951901807593*a-595074826148473058612))*$$

$$r1+(1/3788375980410619811539 2*(433000237234228209904*a-110511411474189997267 5)*i+1/3788375980410619811539 2*$$

$$(-155318623228114063312 5*a+413895013572371003570 8)))*t+((1/151535039216424792461568*(-161697977182036385057*$$

$$a+402493146264364877463)*i+1/151535039216424792461568*(-735157618555847734315*a+191438608910663310975 1))*r1+$$

$$(1/151535039216424792461568*(174904656316304696755 7*a-454361552904042252541 1)*i+1/151535039216424792461568*$$

$$(235712156113320157839 9*a-613660462401817307341 9))))*b1+((1/3213*(-388*a+532)*i+1/51*(4*a-12))*r1+$$

$$(1/189*(580*a-1764)*i+1/189*(-108*a+532)))*t^7+((1/1071*(-174*a+182)*i+1/1071*(-38*a+490))*r1+(1/63*$$

$$(30*a-70)*i+1/27*(14*a-98)))*t^6+((1/3213*(843*a-1631)*i+1/3213*(-369*a+1379))*r1+(1/189*(-1059*a+$$

$$3199)*i+1/63*(75*a-329)))*t^5+((1/306*(87*a-91)*i+1/6426*(423*a-5705))*r1+(1/18*(-15*a+31)*i+1/378*$$

$$(-267*a+2233)))*t^4+((1/1836*(-379*a+945)*i+1/4284*(219*a-1099))*r1+(1/108*(323*a-965)*i+1/756*(-549*$$

$$a+2177)))*t^3+((1/1836*(-270*a+283)*i+1/12852*(-509*a+6916))*r1+(1/756*(342*a-581)*i+1/756*(165*$$

$$a-2072)))*t^2+((1/25704*(1392*a-3773)*i+1/25704*(-279*a+1330))*r1+(1/1512*(-646*a+1911)*i+1/1512*$$

$$(153*a-644)))*t+(1/48*(a-1)*i+1/51408*(137*a-4123))*r1+1/3024*(-225*a+301)*i+1/3024*(-51*a+1141),$$

$$w_7:=((((1/40251494791862835497604*(-1300066503397755587*a+3403542804620583471)*i+1/40251494791862835497604*$$

$$(2061798249678041607*a-5344845882495937909))*r1+(1/40251494791862835497604*(-22563010430161697497*a+5944087658302400033 7)*$$

$$i+1/40251494791862835497604*(13654386462961995013*a-351274657555139204 67)))*t^7+((1/8050298958372567099520 8*$$

$$(-3543367090786939971*a+9445809115797793921)*i+1/8050298958372567099520 8*(2752388624548805915*a-7418620323072792631))*$$

$$r1+(1/8050298958372567099520 8*(-1360202791767310307*a+3779029165913299333)*i+1/8050298958372567099520 8*$$

$$(3834523421669499751*a-11163647755662936247)))*t^6+((1/1610059791674513419904 16*(42296519404049032 89*a-1103323573710275270 5)*$$

$$i+1/1610059791674513419904 16*(-11544924609244510651*a+30062422010789372701))*r1+(1/1610059791674513419904 16*$$

$$(149265738759585921387*a-393437617016660245295)*i+1/5366865972248378066347 2*(-285319926081693752 71*a+733759240958114979 65)))*$$

$$t^5+((1/8050298958372567099520 8*(56279250228427175 20*a-14983946856905228643)*i+1/1610059791674513419904 16*$$

$$(-8812511069960827429*a+2367491310832690359 1))*r1+(1/8050298958372567099520 8*(29231935365097629 3*a-988972039144824392)*$$

$$i+1/1610059791674513419904 16*(-8400142822772670131*a+25404053605563194821)))*t^4+((1/6440239166698053679616 64*$$

$$(985146791295737993 7*a-26210427387720171757)*i+1/6440239166698053679616 64*(125908310796386736 23*a-33424321769534233797))*$$

$$r1+(1/9200341666711505256595 2*(-38566024498989316529*a+101665970224050940661)*i+1/6440239166698053679616 64*$$

$$(136819261455273814623*a-351512040309474584365)))*t^3+((1/6440239166698053679616 64*(-188232407573088937 16*a+$$

$$501176660543986735 35)*i+1/9200341666711505256595 2*(21291065731030485 61*a-56674535954307100 30))*r1+(1/9200341666711505256595 2*$$

$$(210895750421680805*a-5476745635847011144)*i+1/64402391666980536796166 4*(-225310518563295719 8*a+1238769313430830041)))*$$
$$t^2+((1/12880478333396107359233 28*(-109077124010958041 91*a+28858336737850629436)*i+1/12880478333396107359233 28*$$
$$(12910936783887889 48*a-29979085283507080 49))*r1+(1/12880478333396107359233 28*(62922498520157254561*a-16586080498631509 0246)*$$
$$i+1/12880478333396107359233 28*(-22030143195312786958*a+55725571423750759839)))*t+((1/51521913333584429436933 12*$$
$$(13774615150544843951*a-36638942070520215209)*i+1/51521913333584429436933 12*(-12284326950567206407*a+32277903163103030811))*$$
$$r1+(1/51521913333584429436933 12*(-44363631982458322003*a+11715055550342665293 77)*i+1/51521913333584429436933 12*$$
$$(364220421151835415 75*a-933024899784213472 39))))*b1+(((1/54621*(76*a-168)*i+1/54621*(-24*a+28))*r1+$$
$$(1/54621*(320*a-980)*i-4/2023*a))*t^7+((1/54621*(-3*a+7)*i+1/54621*(-19*a+77))*r1+(1/18207*(a+$$
$$35)*i+1/54621*(-29*a-77)))*t^6+((1/109242*(-235*a+497)*i+1/109242*(75*a-91))*r1+(1/15606*(-143*a+$$
$$453)*i+1/36414*(115*a+21)))*t^5+((1/10404*(a-1)*i+1/218484*(111*a-497))*r1+(1/31212*(-3*a-97)*i+$$
$$1/218484*(193*a+581)))*t^4+((1/109242*(94*a-175)*i+1/18207*(-5*a+7))*r1+(1/72828*(267*a-917)*i+$$
$$1/218484*(-291*a-175)))*t^3+((1/62424*(-3*a-4)*i+1/436968*(-76*a+441))*r1+(1/436968*(27*a+602)*i+$$
$$1/436968*(-166*a-735)))*t^2+((1/194208*(-17*a+21)*i+1/1747872*(45*a-91))*r1+(1/1747872*(-625*a+$$
$$2513)*i+1/582624*(83*a+119)))*t+((1/166464*(a+5)*i+1/3495744*(25*a-385))*r1+(1/205632*(-3*a-35)*$$
$$i+1/499392*(27*a+145)))))*b2^2+((((1/40251494791862835497604*(-23695630189160737075*a+64566096785659537257)*$$
$$i+1/40251494791862835497604*(-17793635669035798335*a+45089379492126638683))*r1+(1/23677349877566373822 12*$$
$$(647501187025667591*a-528150204273477009)*i+1/23677349877566373822 12*(836020781748421699*a-2406551623686017451)))*$$
$$t^7+((1/80502895837256709952 08*(935344457598283095*a-5926017736059777643)*i+1/80502895837256709952 08*(-3277744336122305381*$$
$$a+6684377111628741343))*r1+(1/4735469975513274764424*(-391736235816895130 9*a+9670843201805365573)*i+1/4735469975513274764424*$$
$$(-513229578058252709*a+1361148176376823163)))*t^6+((1/16100597916745134199 0416*(167311738592957962593*a-$$
$$448649847460879409219)*i+1/16100597916745134199 0416*(125360638368091324951*a-319029479956151764723))*r1+$$
$$(1/94709399510265495288 48*(-976121064048099180 9*a+18690174604722728359)*i+1/35077555374172405662 4*(-360405991978640033*$$
$$a+992755953426741273)))*t^5+((1/40251494791862835497604*(-2620443257584634072*a+9197753527814126205)*i+$$
$$1/16100597916745134199 0416*(133236041535200654 17*a-29868215951341229623))*r1+(1/23677349877566373822 12*(335100811113120678 6*$$
$$a-838707344152068090 7)*i+1/94709399510265495288 48*(500318129765083813*a-1633623778793448911)))*t^4+((1/64402391666980536796166 4*$$
$$(-342431810594207344827*a+897184478822805398 21)*i+1/64402391666980536796166 4*(-255514608282382441115*a+$$
$$653264786798389365915))*r1+(1/54119656863008854450 56*(564398299705999903 9*a-1330168383418871325 7)*i+1/37883759804106198115392*$$
$$(337047112742366739 69*a-907642796583406362 01)))*t^3+((1/64402391666980536796166 4*(476152345212263509 55*a-$$
$$136518550376919981396)*i+1/92003416667115052565952*(-489027224647893187 6*a+1203887052296929966 9))*r1+(1/54119656863008854450 56*$$
$$(-372221727380676038 6*a+9465329341195775297)*i+1/37883759804106198115392*(360811439461524346 1*a-8123611803544842234)))*$$
$$t^2+((1/12880478333396107359233 28*(996240272660459145 33*a-254192191563275451556)*i+1/12880478333396107359233 28*$$
$$(73305952178897131982*a-186813007937186725591))*r1+(1/75767519608212396230784*(-204252363611781155 39*a+$$
$$51723499392778193750)*i+1/75767519608212396230784*(-151860355021795894 84*a+41146501886110194693)))*t+((1/51521913333584429436933 12*$$
$$(-104688511722207156713*a+276974774727682031945)*i+1/51521913333584429436933 12*(54089520346743435331*a-$$

$$14113995452570115188 1))*r1+(1/303070078432849584923136*(2869467733094213070 5*a-74569784277707992749)*i+1/303070078432849584923136*(-131882590567434891 27*a+33083699401039720937))))*b1+(((1/54621*(-172*a-336)*i+1/3213*(24*a+28))*r1+(1/3213*(-40*a-308)*i+4/119*a))*t^7+((1/3213*(3*a+7)*i+1/54621*(a+175))*r1+(1/1071*(-a+35)*i+1/3213*(127*a-287)))*t^6+((1/109242*(481*a+931)*i+1/6426*(-75*a-91))*r1+(1/918*(13*a+159)*i+1/2142*(-115*a+21)))*t^5+((1/612*(-a-1)*i+1/218484*(507*a-2443))*r1+(1/1836*(3*a-97)*i+1/12852*(-851*a+1967)))*t^4+((1/109242*(-184*a-119)*i+1/1071*(5*a+7))*r1+(1/4284*(-a-399)*i+1/12852*(291*a-175)))*t^3+((1/3672*(3*a-4)*i+1/436968*(-1466*a+4641))*r1+(1/25704*(-27*a+602)*i+1/25704*(824*a-1995)))*t^2+((1/582624*(181*a-511)*i+1/102816*(-45*a-91))*r1+(1/102816*(-145*a+1463)*i+1/34272*(-83*a+119)))*t+((1/9792*(-a+5)*i+1/3495744*(3257*a-8519))*r1+(1/12096*(3*a-35)*i+1/29376*(-141*a+367)))))*b2+(((1/1183867493878318691106*(26625704964214347157*a-70564656420628570893)*i+1/1183867493878318691106*(-18195804177587427819*a+46186125311620393757))*r1+(1/1183867493878318691106*(19876644207376779813 5*a-519201741842362435395)*i+1/1183867493878318691106*(-37842620480632407300 5*a+991167670657197954879)))*t^7+((1/1183867493878318691106*(11013583153752777174*a-29660765781598066045)*i+1/1183867493878318691106*(-33364292769601746923*a+88467669241997571274))*r1+(1/1183867493878318691106*(10804518201751498046*a-28117579125141848587)*i+1/1183867493878318691106*(-25731847051247051653*a+74913864382189952434)))*t^6+((1/4735469975513274764424*(-140916393044499649113*a+375896353367251487351)*i+1/4735469975513274764424*(508462356256837692 05*a-124137376706690774189))*r1+(1/4735469975513274764424*(-124992844789729565 6967*a+326392162073434097 3449)*i+1/157848999183775825 4808*(838372698378966266473*a-219714323885194566 1697)))*t^5+((1/4735469975513274764424*(-651590622602085826 99*a+175120194741622774056)*i+1/4735469975513274764424*(210018794898836527994*a-556205523319711588757))*r1+(1/4735469975513274764424*(-651879180373195464 57*a+168177490153820192318)*i+1/4735469975513274764424*(10227514428130447254 4*a-315077061302352507011)))*t^4+((1/9470939951026549528848*(55766827099369132671*a-152815012358190762482)*i+1/9470939951026549528848*(93285320745184206364*a-251337736665549565479))*r1+(1/1352991421575221361264*(14311366060747601854 1*a-372380573089876491326)*i+1/9470939951026549528848*(-229393486508864917011 6*a+601700625233119374904 9)))*t^3+((1/3788375980410619811539 2*(178524656512718179385*a-482350255715547592401)*i+1/5411965686300885445056*(-97483143330573193693*a+257234440857509745571))*r1+(1/5411965686300885445056*(24169331618481161417*a-63233890975164252449)*i+1/3788375980410619811539 2*(206756465090032076803*a-398841001381846669485)))*t^2+((1/3788375980410619811539 2*(41256993293616545472*a-105264788192651315249)*i+1/3788375980410619811539 2*(-181811421899087006123*a+479398517416157837302))*r1+(1/3788375980410619811539 2*(-35180682726063246466 4*a+905509030016714508581)*i+1/3788375980410619811539 2*(106273295650530030746 3*a-278093344430489165151 0)))*t+((1/151535039216424792461568*(-360980891298124471 9*a+11949171872517168079)*i+1/151535039216424792461568*(23633247757905631014 5*a-62010139746664316640 3))*r1+(1/151535039216424792461568*(-16474614922209030241*a+44054129124011808585)*i+1/151535039216424792461568*(-64036931511255929588 1*a+164038020578159019569 9))))*b1+((1/3213*(-2668*a+6468)*i+1/3213*(1404*a-2716))*r1+(1/189*(-388*a+1036)*i+1/21*(52*a-84)))*t^7+((1/3213*(522*a-1498)*i+1/3213*(394*a-2366))*r1+(1/63*(-30*a+14)*i+1/189*(-26*a+238)))*t^6+((1/3213*(4331*a-10297)*i+1/3213*(-2115*a+4067))*r1+(1/27*(83*a-225)*i+1/21*(-85*a+133)))*t^5+((1/306*(-87*a+227)*i+1/6426*(-1215*a+7777))*r1+(1/54*(45*a-29)*i+1/378*(59*a-833)))*t^4+((1/12852*(-7715*a+17591)*i+1/4284*

$$(1023*a-1981))*r1+(1/252*(-275*a+791)*i+1/756*(1359*a-1981)))*t^3+((1/1836*(270*a-589)*i+1/12852*(1019*a-7392))*$$
$$r1+(1/756*(-342*a+343)*i+1/756*(-23*a+924)))*t^2+((1/4284*(313*a-658)*i+1/12852*(-207*a+434))*r1+(1/756*(67*a-$$
$$245)*i+1/126*(-27*a+35)))*t+(1/144*(-3*a+5)*i+1/51408*(-475*a+4207))*r1+1/3024*(225*a-301)*i+1/432*(-3*a-83),$$
$$w_8 := ((((1/44723883102069817219 56*(171375640105817135*a-436848477143956045)*i+1/57502135416946907853 72*(785972079820285165*$$
$$a-2067414729694528041))*r1+(1/40251494791862835497604*(3488783610963010797*a-8838533567423925043)*i+1/13417164930620945165868*$$
$$(90439103167221882 19*a-23992560283134552067)))*t^7+((1/8050298583725670995208*(344280361718300753*a-1026880688931250377)*$$
$$i + 1/8050298583725670995208 * (4282011902369140153 * a - 11218558149857992815)) * r1 + (1/8050298583725670995208 *$$
$$(7205621387330183273*a-19610556932673027069)*i+1/8050298583725670995208*(780327239348126041 3*a-20464635469424233479)))*$$
$$t^6 + ((1/23000854166778763141488 * (-1327911677586397931 * a + 339383809104138 8149) * i + 1/23000854166778763141488 *$$
$$(-4586406526585571477 * a + 12045925635294199045)) * r1 + (1/16100597916745134199041 6 * (-20167989939433393 75 * a +$$
$$51529201231353784141) * i + 1/16100597916745134199041 6 * (-16541187989174463555 7 * a + 439203350645686188905))) * t^5 +$$
$$((1/26834329861241890331736*(-120374927090820817*a+370467266047938899)*i+1/53668659722483780663472*(-4448258793172824317*$$
$$a + 11670744003483219179)) * r1 + (1/805029895837256709952 08 * (-10780325850530551499 * a + 29268451908479423815) * i +$$
$$1/53668659722483780663472 * (-6353590940025210645 * a + 16672516004940046699))) * t^4 + ((1/64402391666980536796166 4 *$$
$$(13375520586536496183*a-34316523188574485929)*i+1/64402391666980536796166 4*(41265175269727040479*a-10773547129418389193 5))*$$
$$r1 + (1/64402391666980536796166 4 * (24246895916922644251 * a - 63238709058576642941) * i + 1/64402391666980536796166 4 *$$
$$(246478722125839240627 * a - 655628904385078039059))) * t^3 + ((1/64402391666980536796166 4 * (-130741661461866212 * a -$$
$$18707586217227089)*i+1/64402391666980536796166 4*(21223131022316790071*a-55862399147383538486))*r1+(1/21467463888993512265388 8*$$
$$(10931095509347574097*a-29517985142796951122)*i+1/64402391666980536796166 4*(81030321327750462 80*a-21323205582765140131)))*$$
$$t^2 + ((1/42934927777987024530777 6 * (-399311609707953833 * a + 102545519832306002 2) * i + 1/12880478333396107359233 28 *$$
$$(-119685381699894936 4*a+2629032736944748533))*r1+(1/12880478333396107359233 28*(-3703165864020602 45*a+165685456340359537 4)*$$
$$i+1/12880478333396107359233 28*(-32134917355184501960*a+85769178813170741119)))*t+((1/73602733333692042052761 6*$$
$$(321335394385677451*a-840552965843195499)*i+1/51521913333584429436933 12*(-13808242176642508043*a+365796860400758794 49))*$$
$$r1+(1/51521913333584429436933 12*(-17906847631117647021*a+47754573344447229169)*i+1/51521913333584429436933 12*$$
$$(43006418663612703079 * a - 113655787180545352169)))) * b1 + (((1/54621 * (-76 * a + 168) * i + 1/54621 * (24 * a - 28)) * r1 +$$
$$(1/54621 * (-320 * a + 980) * i + 4/2023 * a)) * t^7 + ((1/54621 * (-3 * a + 7) * i + 1/54621 * (-19 * a + 77)) * r1 + (1/18207 * (a + 35) *$$
$$i + 1/54621 * (-29 * a - 77))) * t^6 + ((1/109242 * (235 * a - 497) * i + 1/109242 * (-75 * a + 91)) * r1 + (1/15606 * (143 * a - 453) * i +$$
$$1/36414 * (-115 * a - 21))) * t^5 + ((1/10404 * (a - 1) * i + 1/218484 * (111 * a - 497)) * r1 + (1/31212 * (-3 * a - 97) * i + 1/218484 *$$
$$(193 * a + 581))) * t^4 + ((1/109242 * (-94 * a + 175) * i + 1/18207 * (5 * a - 7)) * r1 + (1/72828 * (-267 * a + 917) * i + 1/218484 * (291 *$$
$$a + 175))) * t^3 + ((1/62424 * (-3 * a - 4) * i + 1/436968 * (-76 * a + 441)) * r1 + (1/436968 * (27 * a + 602) * i + 1/436968 * (-166 * a -$$
$$735))) * t^2 + ((1/194208 * (17 * a - 21) * i + 1/1747872 * (-45 * a + 91)) * r1 + (1/1747872 * (625 * a - 2513) * i + 1/582624 * (-83 * a -$$
$$119))) * t + ((1/166464 * (a + 5) * i + 1/3495744 * (25 * a - 385)) * r1 + (1/205632 * (-3 * a - 35) * i + 1/499392 * (27 * a + 145))))) *$$
$$b2^2 + ((((1/13417164930620945165868 * (-11818778957208122841 * a + 30824690095498180189) * i + 1/57502135416946907853 72 *$$
$$(2971971984064302089*a-7337012539204089267))*r1+(1/23677349877566373822 12*(42956612909869233 51*a-11143163028311200999)*$$

$$i+1/789244995918879127404*(2075521268546996847*a-501972775417456380 1)))*t^7+((1/8050298958372567099520 8*(27051800362393422289*a-70586088246564050391)*i+1/8050298958372567099520 8*(17302714382629084859*a-51261822675061659171))*r1+(1/4735469975513274764424*(-8151357369461283475*a+2130136703975124711 3)*i+1/4735469975513274764424*(1629992466533902075*a-5834282907283084359)))*t^6+((1/23000854166778763141488*(36145224587857297895*a-94153971788006004307)*i+1/23000854166778763141488*(-17862967857152658781*a+44581593563163426119))*r1+(1/9470939951026549528848*(-35564599430944594729*a+92253335591956562065)*i+1/9470939951026549528848*(-36937444638472960489*a+89162519480168723855)))*t^5+((1/26834329861241890331736*(-16125693692176691918*a+41982672808463476845)*i+1/53668659722483780663472*(-17513418093839801575*a+51478572069886992093))*r1+(1/4735469975513274764424*(14067822485809963201*a-36787797873527779778)*i+1/31569799836755165096 16*(-13972182890363721 37*a+51981704892240902 03)))*t^4+((1/64402391666980536796166 4*(-53569636711793479974 3*a+13957687628473558300 03)*i+1/64402391666980536796166 4*(17687417144107030917 5*a-45610185063140737879 7))*r1+(1/37883759804106198115392*(93513414244158243553*a-24168445290168286807 7)*i+1/37883759804106198115392*(50386471963871683855*a-12088603767513822950 1)))*t^3+((1/64402391666980536796166 4*(20818109286020763704 3*a-54397599254815975160 8)*i+1/64402391666980536796166 4*(80388466737357089308*a-22922618761842135727 3))*r1+(1/12627919934702066038464*(-18837797325232891136*a+49062522376990503969)*i+1/37883759804106198115392*(27318755124121723 97*a-13137506968728731680)))*t^2+((1/42934927777987024530777 6*(54507122868754013009*a-14178027087125944460 6)*i+1/12880478333396107359233 28*(-20183716053348440318*a+58881131975292741933))*r1+(1/75767519608212396230784*(-36201053591074679171*a+93215002894370905898)*i+1/75767519608212396230784*(-29673445409069823 02*a+57572516337432961 27)))*t+((1/73602733336920420527616*(-35959982340499316395*a+94238167222132501797)*i+1/51521913333584429436933 12*(-46135311116636483263*a+123553910943014403859))*r1+(1/30307007843284958492313 6*(61423519055110686321*a-16012524976056870932 3)*i+1/30307007843284958492313 6*(73158728478756021 67*a-16329430340419942655))))*b1+(((1/54621*(172*a+336)*i+1/3213*(-24*a-28))*r1+(1/3213*(40*a+308)*i-4/119*a))*t^7+((1/3213*(3*a+7)*i+1/54621*(a+175))*r1+(1/1071*(-a+35)*i+1/3213*(127*a-287)))*t^6+((1/109242*(-481*a-931)*i+1/6426*(75*a+91))*r1+(1/918*(-13*a-159)*i+1/2142*(115*a-21)))*t^5+((1/612*(-a-1)*i+1/218484*(507*a-2443))*r1+(1/1836*(3*a-97)*i+1/12852*(-851*a+1967)))*t^4+((1/109242*(184*a+119)*i+1/1071*(-5*a-7))*r1+(1/4284*(a+399)*i+1/12852*(-291*a+175)))*t^3+((1/3672*(3*a-4)*i+1/436968*(-1466*a+4641))*r1+(1/25704*(-27*a+602)*i+1/25704*(824*a-1995)))*t^2+((1/582624*(-181*a+511)*i+1/102816*(45*a+91))*r1+(1/102816*(145*a-1463)*i+1/34272*(83*a-119)))*t+((1/9792*(-a+5)*i+1/3495744*(3257*a-8519))*r1+(1/12096*(3*a-35)*i+1/29376*(-141*a+367)))))*b2+(((1/3946224979594395637 02*(-33062241597029774973*a+86236340359392360151)*i+1/16912392769690267015 8*(-15241133959361338787*a+40302447974890106151))*r1+(1/11838674938783186911 06*(-64096377861483216749 7*a+16931611653218325287151)*i+1/3946224979594395637 02*(-16032299459401481514 1*a+42859101060967532249 3)))*t^7+((1/11838674938783186911 06*(-54854238241697229950*a+14627498689041944884 5)*i+1/11838674938783186911 06*(-37632749958759339817*a+98125410171112040682))*r1+(1/11838674938783186911 06*(-87240497393721179954*a+23362316448333237141 9)*i+1/11838674938783186911 06*(-68696640415454481079*a+17544248787043412880 6)))*t^6+((1/67649571078761068063 2*(77778717215440661647*a-20239186770572678867 9)*i+1/67649571078761068063 2*(91574441616985611883*

$a - 241606408007936919605)) * r1 + (1/4735469975513274764424 * (402545292870780573914 3 * a - 1065025834877413658805 5) *$

$i + 1/4735469975513274764424 * (28712122022366645512 43 * a - 768411535849838545657 3))) * t^5 + ((1/1578489991837758254808 *$

$(1144040643716679780 31*a-3050749196304644573 62)*i+1/1578489991837758254808*(7750499473557491909 2*a-2025858404989511059 31))*$

$r1 + (1/4735469975513274764424 * (402182785759110751085 * a - 1077776271791135166418) * i + 1/1578489991837758254808 *$

$(118112877300593799424 * a - 300341432713804205535))) * t^4 + ((1/9470939951026549528848 * (-271228441217180996721 * a +$

$697004572878614395024) * i + 1/9470939951026549528848 * (-453913775527045822462 * a + 1189386914984534522305)) * r1 +$

$(1/9470939951026549528848*(-323428504972041919740 7*a+858935284847252422321 4)*i+1/9470939951026549528848*(-201267413796865554092 8*$

$a+540436192731855187452 3)))*t^3+((1/3788375980410619811539 2*(-111602859257465001299 5*a+297474280767869506441 7)*$

$i+1/3788375980410619811539 2*(-716652400225657998605*a+188365604629187812242 5))*r1+(1/1262791993470206603846 4*$

$(-50027053517544044335 * a + 136162470809719411813) * i + 1/3788375980410619811539 2 * (-429896332023195701795 * a +$

$1039273234782722252551))) * t^2 + ((1/1262791993470206603846 4 * (-32008325098541565257 * a + 88173137727286768212) * i +$

$1/3788375980410619811539 2*(894462966578404681 10*a-224361973486735086303))*r1+(1/3788375980410619811539 2*(111876975821327133949 9*$

$a - 3001177671940105760200) * i + 1/3788375980410619811539 2 * (346931684436412897246 * a - 955071040849276708553))) *$

$t + ((1/21647862745203541780224 * (56682032184474599023 * a - 151001972758950349701) * i + 1/1515350392164247924615 68 *$

$(206011868639821032631 * a - 548861607488377092343)) * r1 + (1/1515350392164247924615 68 * (-117862003335638507374 5 * a +$

$313520854480269039659) * i + 1/1515350392164247924615 68 * (-621518666792602480655 * a + 167577353821135861159 9)))) * b1 +$

$((1/3213*(2668*a-6468)*i+1/3213*(-1404*a+2716))*r1+(1/189*(388*a-1036)*i+1/21*(-52*a+84)))*t^7+((1/3213*$

$(522*a-1498)*i+1/3213*(394*a-2366))*r1+(1/63*(-30*a+14)*i+1/189*(-26*a+238)))*t^6+((1/3213*(-4331*a+10297)*$

$i+1/3213*(2115*a-4067))*r1+(1/27*(-83*a+225)*i+1/21*(85*a-133)))*t^5+((1/306*(-87*a+227)*i+1/6426*(-1215*$

$a+7777))*r1+(1/54*(45*a-29)*i+1/378*(59*a-833)))*t^4+((1/12852*(7715*a-17591)*i+1/4284*(-1023*a+1981))*r1+$

$(1/252*(275*a-791)*i+1/756*(-1359*a+1981)))*t^3+((1/1836*(270*a-589)*i+1/12852*(1019*a-7392))*r1+(1/756*$

$(-342*a+343)*i+1/756*(-23*a+924)))*t^2+((1/4284*(-313*a+658)*i+1/12852*(207*a-434))*r1+(1/756*(-67*a+245)*$

$i+1/126*(27*a-35)))*t+(1/144*(-3*a+5)*i+1/51408*(-475*a+4207))*r1+1/3024*(225*a-301)*i+1/432*(-3*a-83),$

$w_9 := ((((1/40251494791862835497604 * (-5948386577558078403 * a + 15612112478919751375) * i + 1/44723883102069817219 56 *$

$(-855094633052519799*a+2286832874852413909))*r1+(1/40251494791862835497604*(-8740829761682819105*a+2252342909570026172 9)*$

$i + 1/40251494791862835497604 * (-30999517459558788073 * a + 82488286316072963431))) * t^7 + ((1/40251494791862835497604 *$

$(693746932438552386*a-180219136621700871 1)*i+1/40251494791862835497604*(-3480481111886776685*a+9247257139153169220))*$

$r1 + (1/40251494791862835497604 * (-16276556172220848717 * a + 43550714972590458920) * i + 1/40251494791862835497604 *$

$(-11517281310578318972 * a + 30583701576493588789))) * t^6 + ((1/16100597916745134199041 6 * (37723266232917452193 * a -$

$990642451971338575 43)*i+1/16100597916745134199041 6*(4563036674804078878 5*a-1221539535509020545 65))*r1+(1/16100597916745134199041 6*$

$(4145385395221467932 3*a-1058135234168253729 53)*i+1/16100597916745134199041 6*(1819681731274766508 35*a-4841409609877639482 11)))*$

$t^5+((1/5750213541694690785372*(-27798134721389269 3*a+731794795736927075)*i+1/16100597916745134199041 6*(2129177369126654782 3*$

$a - 56520678626226207435)) * r1 + (1/8050298958372567099520 8 * (55620082820088410013 * a - 14868084112082650043 0) * i +$

$1/1610059791674513419904 16 * (65355926019894241959 * a - 173170043076099235481))) * t^4 + ((1/92003416667115052565952 *$

$$(-8907996346076840823*a+23433866827506031615)*i+1/2146746388899351226538888*(-20534765267110511779*a+55114129476165192405))*$$

$$r1+(1/64402391666980536796166 4*(-8038198817350810741*a+15077891105423642321)*i+1/64402391666980536796166 4*$$

$$(-237712257844100210089*a+631683965632527162299)))*t^3+((1/64402391666980536796166 4*(26573320501844401695*a-$$

$$70574309146024195022)*i+1/64402391666980536796166 4*(-3187167241300849841 9*a+84559017639061696570))*r1+(1/64402391666980536796166 4*$$

$$(-216050216986099275166*a+576493989784922081845)*i+1/64402391666980536796166 4*(-77300009523708815146*a+$$

$$204045282318660374949)))*t^2+((1/25760956666792214718466 56*(22614114688593428669*a-59818926472163565763)*i+$$

$$1/25760956666792214718466 56*(71832381237782869 03*a-19886361860674752947))*r1+(1/25760956666792214718466 56*$$

$$(-65865569658195356369*a+176098490905536142755)*i+1/25760956666792214718466 56*(179750680853791514 33*a-46915751263524449985)))*$$

$$t+((1/51521913333584429436933 12*(-48399152229255242057*a+128904590236091213845)*i+1/51521913333584429436933 12*$$

$$(15922118811924504553*a-42316654463961708803))*r1+(1/51521913333584429436933 12*(226094659879850544561*a-$$

$$602498845422880072373)*i+1/73602733333692042052761 6*(-1324315753585547203*a+3641088792884973945))))*b1+(((1/54621*$$

$$(76*a-140)*i+1/54621*(-12*a+28))*r1+(1/54621*(152*a-392)*i+1/54621*(-48*a-56)))*t^7+((1/54621*(-3*a-7)*$$

$$i+1/54621*(-25*a+77))*r1+(1/18207*(a-35)*i+1/54621*(153*a-413)))*t^6+((1/54621*(-122*a+217)*i+1/54621*$$

$$(18*a-49))*r1+(1/54621*(-188*a+539)*i+1/54621*(72*a+133)))*t^5+((1/31212*(3*a+5)*i+1/218484*(193*a-595))*$$

$$r1+(1/218484*(-33*a+665)*i+1/31212*(-155*a+433)))*t^4+((1/218484*(208*a-343)*i+1/218484*(-27*a+98))*r1+$$

$$(1/72828*(43*a-210)*i+1/218484*(-108*a-385)))*t^3+((1/62424*(-3*a-1)*i+1/218484*(-116*a+357))*r1+(1/145656*$$

$$(19*a-189)*i+1/109242*(293*a-854)))*t^2+((1/1747872*(-159*a+203)*i+1/582624*(5*a-35))*r1+(1/1747872*$$

$$(213*a+91)*i+1/249696*(9*a+85)))*t+((1/582624*(4*a-7)*i+1/582624*(55*a-168))*r1+(1/102816*(-3*a+14)*i+$$

$$1/1747872*(-730*a+2233)))))*b2^2+((((1/402514947918628354976 04*(10515483055946737389*a-24018521550351907411)*$$

$$i+1/13417164930620945165868*(-2777084023194661127*a+4875037296424303463))*r1+(1/23677349877566373822 12*$$

$$(470747415385874491*a-480305723808409177)*i+1/23677349877566373822 12*(-1210323378206277466 3*a+30319382149320394459)))*$$

$$t^7+((1/402514947918628354976 04*(-6968650974348472233*a+19488144082448483092)*i+1/402514947918628354976 04*$$

$$(-3426731059900829324 8*a+93970933274028708849))*r1+(1/23677349877566373822 12*(27664646785254109 92*a-7884591787598543395)*$$

$$i+1/23677349877566373822 12*(2727948712399930303*a-7018531109226886886)))*t^6+((1/16100597916745134199 0416*$$

$$(-6584320755256883318 1*a+148573219795915245149)*i+1/16100597916745134199 0416*(22153966608148156401*a-13145540669886466555))*$$

$$r1+(1/94709399510265495288 48*(-2057680514142099103*a-37384558743548429)*i+1/94709399510265495288 48*(77967097664131528499*$$

$$a-196354037277700296029)))*t^5+((1/11500427083389381570744*(4187706697037316356*a-11621512398350973265)*i+$$

$$1/16100597916745134199 0416*(23633900795450128720 1*a-64718897420258209281 3))*r1+(1/47354699755132747644 24*$$

$$(-1027257867012145712 1*a+29335663879956955759)*i+1/94709399510265495288 48*(-24386172166865393391*a+63725227619104087729)))*$$

$$t^4+((1/92003416667115052565952*(14711731577831269275*a-32761350487357222789)*i+1/21467463888993512265388 8*$$

$$(32262553285022329893*a-107930052592703242237))*r1+(1/37883759804106198115392*(-65565861498188971 9*a+11700320053789962917)*$$

$$i+1/37883759804106198115392*(-13431747250325737130 5*a+34248087850874914438 1)))*t^3+((1/64402391666980536796166 4*$$

$$(-15146199609283715382 0*a+41984554217945624371 1)*i+1/64402391666980536796166 4*(-46994031421574560357 6*a+$$

$$128286309474054208679 3))*r1+(1/37883759804106198115392*(45445148110655785319*a-12950996184127825115 6)*i+$$

$$1/37883759804106198115392 * (69418178493217058843 * a - 184325412840999255168))) * t^2 + ((1/257609566667922147184665 6 *$$
$$(-30572248820944159187 * a + 72778925659656250195) * i + 1/257609566667922147184665 6 * (-184221378436043176723 * a +$$
$$500285194995627470345))*r1+(1/151535039216424792461568*(4774169938524672355*a-15068665750212393207)*i+1/151535039216424792461568*$$
$$(54840352510623684311 * a - 145051992208703150697))) * t + ((1/515219133335844294369331 2 * (227385598907410053989 * a -$$
$$634628463263546894023) * i + 1/515219133335844294369331 2 * (513623947237943387585 * a - 1388793345764747563165)) * r1 +$$
$$(1/303070078432849584923136*(-576654707907527133 45*a+162316746657120648523)*i+1/432957254904070835604 48*(-160729419400194533 59*$$
$$a+43355707591829678451))))*b1+(((1/54621*(-340*a+1484)*i+1/3213*(12*a+28))*r1+(1/3213*(-40*a-56)*i+1/3213*$$
$$(48*a-56)))*t^7+((1/3213*(3*a-7)*i+1/54621*(-65*a-77))*r1+(1/1071*(-a-35)*i+1/3213*(-111*a+301)))*t^6+$$
$$((1/54621*(548*a-2611)*i+1/3213*(-18*a-49))*r1+(1/3213*(34*a+203)*i+1/3213*(-72*a+133)))*t^5+((1/1836*$$
$$(-3*a+5)*i+1/218484*(317*a+1015))*r1+(1/12852*(33*a+665)*i+1/1836*(97*a-269)))*t^4+((1/218484*(-862*a+$$
$$5089)*i+1/12852*(27*a+98))*r1+(1/4284*(13*a-252)*i+1/12852*(108*a-385)))*t^3+((1/3672*(3*a-1)*i+1/218484*$$
$$(26*a-1071))*r1+(1/8568*(-19*a-189)*i+1/6426*(-139*a+406)))*t^2+((1/1747872*(393*a-4907)*i+1/34272*$$
$$(-5*a-35))*r1+(1/102816*(-171*a+1309)*i+1/14688*(-9*a+85)))*t+((1/34272*(-4*a-7)*i+1/582624*(-123*a+$$
$$742))*r1+(1/6048*(3*a+14)*i+1/102816*(212*a-707)))))*b2+(((1/11838674938783186911 06*(20161365587947819023 9*$$
$$a-531017755560292445669) * i + 1/394622497959439563702 * (150519701778524167 15 * a - 413831278428766810 15)) * r1 +$$
$$(1/11838674938783186911 06*(66414433477114182026 5*a-17457328264991981813 23)*i+1/11838674938783186911 06*(49665474315458443934 3*$$
$$a-132733970081602460326 7))) * t^7 + ((1/23677349877566373822 12 * (11951395858661816819 7 * a - 31574295922970227265 3) *$$
$$i + 1/23677349877566373822 12 * (17248215735575428002 5 * a - 453750530115963305463)) * r1 + (1/23677349877566373822 12 *$$
$$(757343915744143194807*a-202984367231093789102 3)*i+1/23677349877566373822 12*(-102355276375933154389*a+266902023504275710009 1)))*$$
$$t^6+((1/4735469975513274764424*(-122086171034939054189 1 * a + 3215905261995944897281) * i + 1/4735469975513274764424 *$$
$$(-215077237034624147361 * a + 598777193426641979869)) * r1 + (1/4735469975513274764424 * (-379801064544951453650 5 * a +$$
$$997177566185749945803 1) * i + 1/4735469975513274764424 * (-312028783713835615637 5 * a + 8329341107673224110759))) * t^5 +$$
$$((1/676495710787610680632*(-47271450554700409585*a+12446737476529829663 3)*i+1/4735469975513274764424*(-587715890934228313241*$$
$$a + 1545528705503267477643)) * r1 + (1/4735469975513274764424 * (-232420193281148246330 1 * a + 6235952374376381028155) *$$
$$i + 1/4735469975513274764424 * (617319619290968949927 * a - 1619604812308461525995))) * t^4 + ((1/135299142157522136126 4 *$$
$$(126267405216639680730*a-33277824947479035478 3)*i+1/315697998367551650961 6*(75988145294221160 81*a-28062096006868088942))*$$
$$r1 + (1/9470939951026549528848 * (224059657260486255186 4 * a - 5860949566092249922097) * i + 1/9470939951026549528848 *$$
$$(252037251655059000239 5 * a - 6706310790549192539276))) * t^3 + ((1/37883759804106198115392 * (699191823839258341407 * a -$$
$$1813909787835312411307) * i + 1/37883759804106198115392 * (227704268363594480059 1 * a - 5986748050032520494325)) * r1 +$$
$$(1/37883759804106198115392 * (709513351720375492858 9 * a - 1909171102243380064786 9) * i + 1/37883759804106198115392 *$$
$$(-441057987343122219052 7 * a + 11593783828757549246625))) * t^2 + ((1/75767519608212396230784 * (-363699319949847282301 *$$
$$a + 964263750255564078257) * i + 1/75767519608212396230784 * (323246229034951641631 * a - 838992978836971140665)) * r1 +$$
$$(1/75767519608212396230784*(383581530073604817949*a-10564235421174059808 33)*i+1/75767519608212396230784*(-175412046620297992179 1*$$
$$a+462165446908435563293 7))) * t + ((1/151535039216424792461568 * (159403849844533714291 * a - 456040574664408167609) * i +$$

$$1/151535039216424792461568*(-1177795630799971165355*a+309732653952693496 5691))*r1+(1/151535039216424792461568*(-197744624127072683323 5*a+539219214023003754 7069)*i+1/216478627452035417802 24*(592735449165230022353*a-155829627267181936408 5))))*b1+((1/3213*(-3508*a+9716)*i+1/3213*(828*a-2212))*r1+(1/189*(-220*a+700)*i+1/189*(180*a-140)))*t^7+((1/3213*(522*a-1022)*i+1/3213*(1102*a-2870))*r1+(1/63*(-30*a+154)*i+1/189*(-222*a+742)))*t^6+((1/3213*(5828*a-15946)*i+1/3213*(-1242*a+3556))*r1+(1/189*(208*a-742)*i+1/189*(-270*a+112)))*t^5+((1/918*(-279*a+599)*i+1/6426*(-4121*a+11291))*r1+(1/378*(369*a-1673)*i+1/54*(119*a-397)))*t^4+((1/12852*(-10747*a+28777)*i+1/12852*(1863*a-6167))*r1+(1/252*(61*a-63)*i+1/756*(405*a+203)))*t^3+((1/1836*(315*a-775)*i+1/12852*(4637*a-13503))*r1+(1/84*(-53*a+203)*i+1/756*(-983*a+3227)))*t^2+((1/25704*(2517*a-6503)*i+1/8568*(-81*a+413))*r1+(1/1512*(-285*a+721)*i+1/216*(-9*a-29)))*t+(1/2016*(-57*a+161)*i+1/34272*(-2045*a+6335))*r1+1/6048*(711*a-2303)*i+1/6048*(1355*a-4361),$$

$$w_10:=((((1/40251494791862835497604*(5948386577558078403*a-156121124789197513 75)*i+1/44723883102069817219 56*(855094633052519799*a-2286832874852413909))*r1+(1/40251494791862835497604*(874082976168281910 5*a-22523429095700261729)*i+1/40251494791862835497604*(30999517459558788073*a-824828631607296343 1)))*t^7+((1/40251494791862835497604*(-693746932438552386*a+180219136621700871 1)*i+1/40251494791862835497604*(3480481111886776685*a-9247257139153169220))*r1+(1/40251494791862835497604*(162765561722208487 17*a-435507149725904589 20)*i+1/40251494791862835497604*(115172813105783189 72*a-305837015764935887 89)))*t^6+((1/161005979167451341990416*(-377232662329174521 93*a+990624519713385754 3)*i+1/161005979167451341990416*(-456303667480407887 85*a+122153953550902054565))*r1+(1/161005979167451341990416*(-414538539522146793 23*a+105813523416825372953)*i+1/161005979167451341990416*(-181968173127476650835*a+484140960987763948211)))*t^5+((1/5750213541694690785372*(277981347213892693*a-731794795736927075)*i+1/161005979167451341990416*(-2129177369126654782 3*a+565206786262262074 35))*r1+(1/805029895837256709952 08*(-55620082820088410013*a+148680841120826500430)*i+1/161005979167451341990416*(-65355926019894241959*a+173170043076099235481)))*t^4+((1/9200341666711505256595 2*(8907996346076840823*a-2343386682750603161 5)*i+1/21467463888993512265388 8*(205347652671105117 79*a-55114129476165192405))*r1+(1/64402391666980536796166 4*(8038198817350810741*a-150778911054236423 21)*i+1/64402391666980536796166 4*(237712257844100210089*a-631683965632527162299)))*t^3+((1/64402391666980536796166 4*(-265733205018444016 95*a+705743091460241950 22)*i+1/64402391666980536796166 4*(318716724130084984 19*a-845590176390616965 70))*r1+(1/64402391666980536796166 4*(216050216986099275166*a-576493989784922081845)*i+1/64402391666980536796166 4*(773000095237088151 46*a-204045282318660374949)))*t^2+((1/25760956666792214718466 56*(-22614114688593428669*a+598189264721635657 63)*i+1/25760956666792214718466 56*(-7183238123778286903*a+19886361860674752947))*r1+(1/25760956666792214718466 56*(658655696581953563 69*a-176098490905361427 55)*i+1/25760956666792214718466 56*(-1797506808537915143 3*a+469157512635244499 85)))*t+((1/515219133335844294369331 2*(48399152229255242057*a-128904590236091213845)*i+1/515219133335844294369331 2*(-15922118811924504553*a+42316654463961708803))*r1+(1/515219133335844294369331 2*(-22609465987985054456 1*a+602498845422880072373)*i+1/736027333336920420527616*(1324315753585547203*a-3641088792884973945))))*b1+(((1/54621*(76*a-140)*i+1/54621*(-12*a+28))*r1+(1/54621*(152*a-392)*i+1/54621*(-48*a-56)))*t^7+((1/54621*(-3*a-7)*i+1/54621*(-25*a+77))*r1+(1/18207*(a-35)*i+1/54621*(153*a-413)))*t^6+((1/54621*(-122*a+217)*i+1/54621*(18*a-49))*

$$r1+(1/54621*(-188*a+539)*i+1/54621*(72*a+133)))*t^5+((1/31212*(3*a+5)*i+1/218484*(193*a-595))*r1+(1/218484*(-33*a+665)*i+1/31212*(-155*a+433)))*t^4+((1/218484*(208*a-343)*i+1/218484*(-27*a+98))*r1+(1/72828*(43*a-210)*i+1/218484*(-108*a-385)))*t^3+((1/62424*(-3*a-1)*i+1/218484*(-116*a+357))*r1+(1/145656*(19*a-189)*i+1/109242*(293*a-854)))*t^2+((1/1747872*(-159*a+203)*i+1/582624*(5*a-35))*r1+(1/1747872*(213*a+91)*i+1/249696*(9*a+85)))*t+((1/582624*(4*a-7)*i+1/582624*(55*a-168))*r1+(1/102816*(-3*a+14)*i+1/1747872*(-730*a+2233)))))*b2^2+((((1/40251494791862835497604*(-10515483055946737389*a+24018521550351907411)*i+1/13417164930620945165868*(2777084023194661127*a-487503729642430346 3))*r1+(1/23677349877566373822 12*(-470747415385874491*a+480305723808409177)*i+1/23677349877566373822 12*(12103233782062774663*a-30319382149320394459)))*t^7+((1/40251494791862835497604*(696865097434847223 3*a-19488144082448483092)*i+1/40251494791862835497604*(34267310599008293248*a-93970933274028708849))*r1+(1/23677349877566373822 12*(-276646467852541099 2*a+788459178759854339 5)*i+1/23677349877566373822 12*(-272794871239993030 3*a+701853110922688688 6)))*t^6+((1/16100597916745134199041 6*(65843207552568833181*a-14857321979591524514 9)*i+1/16100597916745134199041 6*(-22153966608148156401*a+13145540669886466555))*r1+(1/9470939951026549528848*(205768051414209910 3*a+3738455874354842 9)*i+1/9470939951026549528848*(-77967097664131528499*a+19635403727770029602 9)))*t^5+((1/11500427083389381570744*(-4187706697037316356*a+11621512398350973265)*i+1/16100597916745134199041 6*(-23633900795450128720 1*a+64718897420258209281 3))*r1+(1/47354699755132747644 24*(10272578670121457121*a-29335663879956955759)*i+1/9470939951026549528848*(24386172166865393391*a-63725227619104087729)))*t^4+((1/92003416667115052565952*(-14711731577831269275*a+32761350487357222789)*i+1/21467463888993512265388 8*(-32262553285022329893*a+107930052592703242237))*r1+(1/37883759804106198115392*(65565861498188971 9*a-11700320053789962917)*i+1/37883759804106198115392*(13431747250325737130 5*a-34248087850874914438 1)))*t^3+((1/64402391666980536796166 4*(15146199609283715382 0*a-41984554217945624371 1)*i+1/64402391666980536796166 4*(46994031421574560357 6*a-12828630947405420867 93))*r1+(1/37883759804106198115392*(-45445148110655785319*a+12950996184127825115 6)*i+1/37883759804106198115392*(-69418178493217058843*a+184325412840999255168)))*t^2+((1/25760956666792214718466 56*(30572248820944159187*a-72778925659656250195)*i+1/25760956666792214718466 56*(18422137843604317672 3*a-50028519499562747034 5))*r1+(1/15153503921642479246156 8*(-4774169938524672355*a+15068665750212393207)*i+1/15153503921642479246156 8*(-54840352510623684311*a+14505199220870315069 7)))*t+((1/51521913333584429436933 12*(-22738559890741005398 9*a+63462846326354689402 3)*i+1/51521913333584429436933 12*(-51362394723794338758 5*a+13887933457647475631 65))*r1+(1/30307007843284958492313 6*(57665470790752713345*a-16231674665712064852 3)*i+1/43295725490407083560448*(16072941940019453359*a-43355707591829678451))))*b1+(((1/54621*(-340*a+1484)*i+1/3213*(12*a+28))*r1+(1/3213*(-40*a-56)*i+1/3213*(48*a-56)))*t^7+((1/3213*(3*a-7)*i+1/54621*(-65*a-77))*r1+(1/1071*(-a-35)*i+1/3213*(-111*a+301)))*t^6+((1/54621*(548*a-2611)*i+1/3213*(-18*a-49))*r1+(1/3213*(34*a+203)*i+1/3213*(-72*a+133)))*t^5+((1/1836*(-3*a+5)*i+1/218484*(317*a+1015))*r1+(1/12852*(33*a+665)*i+1/1836*(97*a-269)))*t^4+((1/218484*(-862*a+5089)*i+1/12852*(27*a+98))*r1+(1/4284*(13*a-252)*i+1/12852*(108*a-385)))*t^3+((1/3672*(3*a-1)*i+1/218484*(26*a-1071))*r1+(1/8568*(-19*a-189)*i+1/6426*(-139*a+406)))*t^2+((1/1747872*(393*a-4907)*i+1/34272*(-5*a-35))*r1+(1/102816*(-171*a+1309)*i+1/14688*(-9*a+85)))*t+((1/34272*(-4*a-7)*i+1/582624*(-123*a+742))*r1+(1/6048*(3*a+14)*$$

$$i + 1/102816 * (212 * a - 707))))) * b2 + (((1/1183867493878318691106 * (-201613655879478190239 * a + 531017755560292445669) * i + 1/394622497959439563702 * (-15051970177852416715 * a + 41383127842876681015)) * r1 + (1/1183867493878318691106 * (-664144334771141820265 * a + 1745732826499198181323) * i + 1/1183867493878318691106 * (-496654743154584439343 * a + 132733970081602460326 7))) * t^7 + ((1/2367734987756637382212 * (-119513958586618168197 * a + 315742959229702272653) * i + 1/2367734987756637382212*(-172482157355754280025*a+453750530115963305463))*r1+(1/2367734987756637382212*(-757343915744143194807* a + 2029843672310937891023) * i + 1/2367734987756637382212 * (102355276375933154389 * a - 266920235042757100091))) * t^6 + ((1/4735469975513274764424*(122086171034939054189 1*a-321590526199594489728 1)*i+1/4735469975513274764424*(215077237034624147361* a - 598777193426641979869)) * r1 + (1/4735469975513274764424 * (3798010645449514536505 * a - 997177566185749945803 1) * i + 1/4735469975513274764424 * (312028783713835615637 5 * a - 832934110767322411075 9))) * t^5 + ((1/6764957107876106806 32 * (472714505547004095 85*a-124467374765298296633)*i+1/4735469975513274764424*(587715890934228313241*a-154552870550326747764 3))* r1 + (1/4735469975513274764424 * (2324201932811482463301 * a - 6235952374376381028155) * i + 1/4735469975513274764424 * (-617319619290968949927 * a + 161960481230846152599 5))) * t^4 + ((1/13529914215752213612 64 * (-126267405216639680730 * a + 332778249474790354783) * i + 1/3156979983675516509616 * (-7598814529422116081 * a + 28062096006868088942)) * r1 + (1/9470939951026549528848*(-224059657260486255186 4*a+586094956609224992209 7)*i+1/9470939951026549528848*(-252037251655059000239 5* a + 67063107905491925392 76))) * t^3 + ((1/3788375980410619811539 2 * (-699191823839258341407 * a + 181390978783531241130 7) * i+1/3788375980410619811539 2*(-227704268363594480059 1*a+598674805003252049432 5)) * r1 + (1/3788375980410619811539 2 * (-709513351720375492858 9 * a + 190917110224338006478 69) * i + 1/3788375980410619811539 2 * (441057987343122219052 7 * a - 115937838287575492466 25))) * t^2 + ((1/7576751960821239623078 4 * (363699319949847282301 * a - 964263750255564078257) * i + 1/7576751960821239623078 4 * (-323246229034951641631 * a + 838992978836971140665)) * r1 + (1/7576751960821239623078 4 * (-383581530073604817949 * a + 1056423542117405980833) * i + 1/7576751960821239623078 4 * (175412046620297992179 1 * a - 462165446908435563293 7))) * t + ((1/1515350392164247924615 68 * (-159403849844533714291 * a + 456040574664408167609) * i + 1/1515350392164247924615 68 * (117779563079997116535 5 * a - 309732653952693496569 1)) * r1 + (1/1515350392164247924615 68 * (197744624127072683323 5 * a - 539219214023003754706 9) * i + 1/2164786274520354178022 4 * (-592735449165230022353 * a + 155829627267181936408 5)))) * b1 + ((1/3213 * (-3508 * a + 9716) * i + 1/3213 * (828 * a - 2212)) * r1 + (1/189 * (-220 * a + 700) * i + 1/189 * (180 * a - 140))) * t^7 + ((1/3213 * (522 * a - 1022) * i + 1/3213 * (1102 * a - 2870)) * r1 + (1/63 * (-30 * a + 154) * i + 1/189 * (-222 * a + 742))) * t^6 + ((1/3213 * (5828 * a - 15946) * i + 1/3213 * (-1242 * a + 3556)) * r1 + (1/189 * (208 * a - 742) * i + 1/189 * (-270 * a + 112))) * t^5 + ((1/918 * (-279 * a + 599) * i + 1/6426 * (-4121 * a + 11291)) * r1 + (1/378 * (369 * a - 1673) * i + 1/54 * (119 * a - 397))) * t^4 + ((1/12852 * (-10747 * a + 28777) * i + 1/12852 * (1863 * a - 6167)) * r1 + (1/252 * (61 * a - 63) * i + 1/756 * (405 * a + 203))) * t^3 + ((1/1836 * (315 * a - 775) * i + 1/12852 * (4637 * a - 13503)) * r1 + (1/84 * (-53 * a + 203) * i + 1/756 * (-983 * a + 3227))) * t^2 + ((1/25704 * (2517 * a - 6503) * i + 1/8568 * (-81 * a + 413)) * r1 + (1/1512 * (-285 * a + 721) * i + 1/216 * (-9 * a - 29))) * t + (1/2016 * (-57 * a + 161) * i + 1/34272 * (-2045 * a + 6335)) * r1 + 1/6048 * (711 * a - 2303) * i + 1/6048 * (1355 * a - 4361),$$

$$w_1 1 := ((((1/4472388310206981721956 * (-171375640105817135 * a + 436848477143956045) * i + 1/57502135416946907853 72 * (-785972079820285165*a+206741472969452804 1))*r1+(1/4025149479186283549760 4*(-348878361096301079 7*a+883853356742392504 3)* i + 1/1341716493062094516586 8 * (-90439103167221882 19 * a + 239925602831345520 67))) * t^7 + ((1/8050298958372567099520 8 *$$

$$(-344280361718300753*a+1026880688931250377)*i+1/80502989583725670995208*(-428201190236914015 3*a+1121855814985799 2815))*$$

$$r1 + (1/80502989583725670995208 * (-7205621387330183273 * a + 19610556932673027069) * i + 1/80502989583725670995208 *$$

$$(-7803272393481260413*a+20464635469424233479)))*t^6+((1/23000854166778763141488*(1327911677586397931*a-3393838091041388149)*$$

$$i + 1/23000854166778763141488 * (4586406526585571477 * a - 12045925635294199045)) * r1 + (1/16100597916745134199041 6 *$$

$$(20167989939433393 75*a-51529201231353784141)*i+1/16100597916745134199041 6*(16541187989174463555 7*a-43920335064568618890 5)))*$$

$$t^5+((1/26834329861241890331736*(120374927090820817*a-370467266047938899)*i+1/53668659722483780663472*(4448258793172824317*$$

$$a - 11670744003483219179)) * r1 + (1/80502989583725670995208 * (10780325850530551499 * a - 29268451908479423815) * i +$$

$$1/53668659722483780663472*(63535909400252106 45*a-16672516004940046699)))*t^4+((1/64402391666980536796166 4*(-13375520586536496183*$$

$$a + 34316523188574485929) * i + 1/64402391666980536796166 4 * (-41265175269727040479 * a + 107735471294183891935)) * r1 +$$

$$(1/64402391666980536796166 4*(-24246895916922644251*a+63238709058576642941)*i+1/64402391666980536796166 4*(-2464787221258392406 27*$$

$$a + 655628904385078039059))) * t^3 + ((1/64402391666980536796166 4 * (130741661461866212 * a + 18707586217227089) * i +$$

$$1/64402391666980536796166 4 * (-21223131022316790071 * a + 55862399147383538486)) * r1 + (1/21467463888993512265388 8 *$$

$$(-10931095509347574097*a+29517985142796951122)*i+1/64402391666980536796166 4*(-81030321327750462 80*a+21323205582765140131)))*$$

$$t^2 + ((1/42934927777987024530777 6 * (399311609707953833 * a - 1025455198323060022) * i + 1/12880478333396107359233 28 *$$

$$(1196853816998949364*a-2629032736944748533))*r1+(1/12880478333396107359233 28*(37031658640206024 5*a-1656854563403595374)*$$

$$i + 1/12880478333396107359233 28 * (32134917355184501960 * a - 85769178813170741119))) * t + ((1/73602733333692042052761 6 *$$

$$(-321335394385677451*a+840552965843195499)*i+1/51521913333584429436933 12*(13808242176642508043*a-3657968604007587944 9))*$$

$$r1 + (1/51521913333584429436933 12 * (17906847631117647021 * a - 47754573344447229169) * i + 1/51521913333584429436933 12 *$$

$$(-43006418663612703079 * a + 113655787180545352169)))) * b1 + (((1/54621 * (-76 * a + 168) * i + 1/54621 * (24 * a - 28)) * r1 +$$

$$(1/54621 * (-320 * a + 980) * i + 4/2023 * a)) * t^7 + ((1/54621 * (-3 * a + 7) * i + 1/54621 * (-19 * a + 77)) * r1 + (1/18207 * (a + 35) *$$

$$i + 1/54621 * (-29 * a - 77))) * t^6 + ((1/109242 * (235 * a - 497) * i + 1/109242 * (-75 * a + 91)) * r1 + (1/15606 * (143 * a - 453) * i +$$

$$1/36414 * (-115 * a - 21))) * t^5 + ((1/10404 * (a - 1) * i + 1/218484 * (111 * a - 497)) * r1 + (1/31212 * (-3 * a - 97) * i + 1/218484 *$$

$$(193 * a + 581))) * t^4 + ((1/109242 * (-94 * a + 175) * i + 1/18207 * (5 * a - 7)) * r1 + (1/72828 * (-267 * a + 917) * i + 1/218484 * (291 *$$

$$a + 175))) * t^3 + ((1/62424 * (-3 * a - 4) * i + 1/436968 * (-76 * a + 441)) * r1 + (1/436968 * (27 * a + 602) * i + 1/436968 * (-166 * a -$$

$$735))) * t^2 + ((1/194208 * (17 * a - 21) * i + 1/1747872 * (-45 * a + 91)) * r1 + (1/1747872 * (625 * a - 2513) * i + 1/582624 * (-83 * a -$$

$$119))) * t + ((1/166464 * (a + 5) * i + 1/3495744 * (25 * a - 385)) * r1 + (1/205632 * (-3 * a - 35) * i + 1/499392 * (27 * a + 145))))) *$$

$$b2^2 + ((((1/13417164930620945165868 * (11818778957208122841 * a - 30824690095498180189) * i + 1/57502135416946907853 72 *$$

$$(-2971971984064302089*a+7337012539204089267))*r1+(1/23677349877566373822 12*(-4295661290986923351*a+1114316302831120099 9)*$$

$$i + 1/789244995918879127404 * (-2075521268546996847 * a + 5019727754174563801))) * t^7 + ((1/80502989583725670995208 *$$

$$(-27051800362393422289*a+70586088246564050391)*i+1/80502989583725670995208*(-17302714382629084859*a+51261822675061659171))*$$

$$r1+(1/47354699755132747644 24*(81513573694612834 75*a-21301367039751247113)*i+1/47354699755132747644 24*(-16299924665339020 75*$$

$$a + 5834282907283084359))) * t^6 + ((1/23000854166778763141488 * (-36145224587857297895 * a + 94153971788006004307) * i +$$

$$1/23000854166778763141488*(17862967857152658781*a-44581593563163426119))*r1+(1/94709399510265495288 48*(35564599430944594729*$$

$$a - 92253335591956562065) * i + 1/947093995102654952884 8 * (36937444638472960489 * a - 89162519480168723855))) * t^5 +$$

$$((1/26834329861241890331736*(16125693692176691918*a-41982672808463476845)*i+1/53668659722483780663472*(17513418093839801575*a - 51478572069886992093)) * r1 + (1/4735469975513274764424 * (-14067822485809963201 * a + 36787797873527779778) * i + 1/31569799836755165096 16*(1397218289036372137*a-5198170489224090203)))*t^4+((1/64402391666980536796166 4*(53569636711793479974 3*a - 139576876284735583000 3) * i + 1/64402391666980536796166 4 * (-17687417144107030917 5 * a + 45610185063140737879 7)) * r1 + (1/37883759804106198115392 * (-93513414244158243553 * a + 241684452901682868077) * i + 1/37883759804106198115392 * (-50386471963871683855 * a + 120886037675138229501))) * t^3 + ((1/64402391666980536796166 4 * (-20818109286020763704 3 * a + 543975992548159751608) * i + 1/64402391666980536796166 4 * (-80388466737357089308 * a + 229226187618421357273)) * r1 + (1/12627919934702066038464 * (18837797325232891136 * a - 49062522376990503969) * i + 1/37883759804106198115392 * (-2731875512412172397 * a + 13137506968728731680))) * t^2 + ((1/42934927777987024530777 6 * (-54507122868754013009 * a + 141780270871259444606)*i+1/12880478333396107359233 28*(20183716053348440318*a-58881131975292741933))*r1+(1/75767519608212396230784*(36201053591074679171*a-93215002894370905898)*i+1/75767519608212396230784*(296734454090698230 2*a-5757251633743296127)))*t + ((1/73602733333692042052761 6 * (35959982340499316395 * a - 94238167222132501797) * i + 1/51521913333584429436933 12 * (46135311116636483263 * a - 12355391094301440385 9)) * r1 + (1/30307007843284958492313 6 * (-61423519055110686321 * a + 160125249760568709323) * i + 1/30307007843284958492313 6 * (-73158728478756021 67 * a + 16329430340419942655)))) * b1 + (((1/54621*(172*a+336)*i+1/3213*(-24*a-28))*r1+(1/3213*(40*a+308)*i-4/119*a))*t^7+((1/3213*(3*a+7)*i+1/54621*(a+175))*r1+(1/1071*(-a+35)*i+1/3213*(127*a-287)))*t^6+((1/109242*(-481*a-931)*i+1/6426*(75*a+91))*r1+(1/918*(-13*a-159)*i+1/2142*(115*a-21)))*t^5+((1/612*(-a-1)*i+1/218484*(507*a-2443))*r1+(1/1836*(3*a-97)*i+1/12852*(-851*a+1967)))*t^4+((1/109242*(184*a+119)*i+1/1071*(-5*a-7))*r1+(1/4284*(a+399)*i+1/12852*(-291*a+175)))*t^3+((1/3672*(3*a-4)*i+1/436968*(-1466*a+4641))*r1+(1/25704*(-27*a+602)*i+1/25704*(824*a-1995)))*t^2+((1/582624*(-181*a+511)*i+1/102816*(45*a+91))*r1+(1/102816*(145*a-1463)*i+1/34272*(83*a-119)))*t+((1/9792*(-a+5)*i+1/3495744*(3257*a-8519))*r1+(1/12096*(3*a-35)*i+1/29376*(-141*a+367)))))*b2+(((1/3946224979594395637 02*(33062241597029774973*a-86236340359392360151)*i+1/16912392769690267015 8*(15241133959361338787*a-40302447974890106151))*r1+(1/11838674938783186911 06*(64096377861483216749 7*a-16931616532183252871 51)*i + 1/39462249795943956370 2 * (16032299459401481514 1 * a - 42859101060967532249 3))) * t^7 + ((1/11838674938783186911 06 * (54854238241697229950*a-14627498689041944845)*i+1/11838674938783186911 06*(37632749958759339817*a-98125410171112040682))*r1 + (1/11838674938783186911 06 * (87240497393721179954 * a - 233623164483332371419) * i + 1/11838674938783186911 06 * (68696640415454481079 * a - 175442487870434128806))) * t^6 + ((1/67649571078761068063 2 * (-77778717215440661647 * a + 202391867705726788679)*i+1/67649571078761068063 2*(-91574441616985611883*a+24160640800793691960 5))*r1+(1/47354699755132747644 24*(-40254529287078057391 43 * a + 10650258348774136588055) * i + 1/47354699755132747644 24 * (-28712122022366645512 43 * a + 76841153584983854565 73))) * t^5 + ((1/15784899918377582548 08 * (-11440406437166797803 1 * a + 305074919630464573 62) * i + 1/15784899918377582548 08 * (-77504994735574919092 * a + 202585840498951105931)) * r1 + (1/47354699755132747644 24 * (-40218278575911075108 5 * a + 1077776271791135166418) * i + 1/15784899918377582548 08 * (-11811287730059379942 4 * a + 300341432713804205535))) * t^4 + ((1/94709399510265495288 48 * (27122844121718099672 1 * a - 697004572878614395024) * i + 1/94709399510265495288 48*(45391377552704582246 2*a-11893869149845345223 05))*r1+(1/94709399510265495288 48*(32342850497204191974 07*$$

$a - 858935284847252422321 4) * i + 1/9470939951026549528848 * (2012674137968655540928 * a - 5404361927318551874523))) * t^3 +$

$((1/37883759804106198115392 * (111602859257465001 2995 * a - 297474280767869506 4417) * i + 1/37883759804106198115392 *$

$(716652400225657998605 * a - 188365604629187812 2425)) * r1 + (1/12627919934702066038464 * (50027053517544044335 * a -$

$136162470809719411813) * i + 1/37883759804106198115392 * (429896332023195701795 * a - 103927323478272225 2551))) * t^2 +$

$((1/12627919934702066038464*(32008325098541565257*a-881731377272867682 12)*i+1/37883759804106198115392*(-894462966578404681 10*$

$a + 224361973486735086303)) * r1 + (1/37883759804106198115392 * (-111876975821327133949 9 * a + 3001177671940105760200) *$

$i + 1/37883759804106198115392 * (-346931684436412897246 * a + 955071040849276708553))) * t + ((1/216478627452035417802 24 *$

$(-56682032184474599023 * a + 151001972758950349701) * i + 1/151535039216424792461568 * (-206011868639821032631 * a +$

$548861607488377092343)) * r1 + (1/151535039216424792461568 * (117862003335638507374 5 * a - 313520854480269039685 9) * i +$

$1/151535039216424792461568 * (621518666792602480655 * a - 167577353821135861159 9))))) * b1 + ((1/3213 * (2668 * a - 6468) * i +$

$1/3213 * (-1404 * a + 2716)) * r1 + (1/189 * (388 * a - 1036) * i + 1/21 * (-52 * a + 84))) * t^7 + ((1/3213 * (522 * a - 1498) * i + 1/3213 *$

$(394 * a - 2366)) * r1 + (1/63 * (-30 * a + 14) * i + 1/189 * (-26 * a + 238))) * t^6 + ((1/3213 * (-4331 * a + 10297) * i + 1/3213 * (2115 * a -$

$4067)) * r1 + (1/27 * (-83 * a + 225) * i + 1/21 * (85 * a - 133))) * t^5 + ((1/306 * (-87 * a + 227) * i + 1/6426 * (-1215 * a + 7777)) * r1 +$

$(1/54 * (45 * a - 29) * i + 1/378 * (59 * a - 833))) * t^4 + ((1/12852 * (7715 * a - 17591) * i + 1/4284 * (-1023 * a + 1981)) * r1 + (1/252 *$

$(275 * a - 791) * i + 1/756 * (-1359 * a + 1981))) * t^3 + ((1/1836 * (270 * a - 589) * i + 1/12852 * (1019 * a - 7392)) * r1 + (1/756 * (-342 *$

$a + 343) * i + 1/756 * (-23 * a + 924))) * t^2 + ((1/4284 * (-313 * a + 658) * i + 1/12852 * (207 * a - 434)) * r1 + (1/756 * (-67 * a + 245) * i +$

$1/126 * (27 * a - 35))) * t + (1/144 * (-3 * a + 5) * i + 1/51408 * (-475 * a + 4207)) * r1 + 1/3024 * (225 * a - 301) * i + 1/432 * (-3 * a - 83),$

$w_1 2 := ((((1/40251494791862835497604 * (1300066503397755587 * a - 3403542804620583471) * i + 1/40251494791862835497604 *$

$(-2061798249678041607*a+534484588249593790 9))*r1+(1/40251494791862835497604*(22563010430161697497*a-594408765830240003 37)*$

$i + 1/40251494791862835497604 * (-13654386462961995013 * a + 35127465755513920467))) * t^7 + ((1/80502989583725670995208 *$

$(3543367090786939971*a-9445809115797793921)*i+1/80502989583725670995208*(-2752388624548805915*a+7418620323072792631))*$

$r1+(1/80502989583725670995208*(1360202791767310307*a-3779029165913299333)*i+1/80502989583725670995208*(-3834523421669499751*$

$a + 11163647755662936247))) * t^6 + ((1/16100597916745134199 0416 * (-4229651940404903289 * a + 11033235737102752705) *$

$i + 1/16100597916745134199 0416 * (11544924609244510651 * a - 30062422010789372701)) * r1 + (1/16100597916745134199 0416 *$

$(-1492657387595859213 87*a+3934376170166602452 95)*i+1/5366865972248378066 3472*(2853199260816937527 1*a-7337592409581149796 5)))*$

$t^5 + ((1/80502989583725670995208 * (-5627925022842717520 * a + 14983946856905228643) * i + 1/16100597916745134199 0416 *$

$(8812511069960827429*a-2367491310832690359 1))*r1+(1/80502989583725670995208*(-292319353650976293*a+988972039144824392)*$

$i + 1/16100597916745134199 0416 * (8400142822772670131 * a - 25404053605563194821))) * t^4 + ((1/64402391666980536796 1664 *$

$(-9851467912957379937*a+2621042738772017175 7)*i+1/64402391666980536796 1664*(-1259083107963867362 3*a+3342432176953423 3797))*$

$r1 + (1/92003416667115052565952 * (38566024498989316529 * a - 10166597022405094066 1) * i + 1/64402391666980536796 1664 *$

$(-1368192614552738146 23 * a + 3515120403094745843 65))) * t^3 + ((1/64402391666980536796 1664 * (1882324075730889371 6 * a -$

$5011766605439867353 5)*i+1/92003416667115052565952*(-2129106573103048561*a+5667453595430710030))*r1+(1/92003416667115052565952*$

$(-2108957550421680805*a+5476745635847011144)*i+1/64402391666980536796 1664*(2253105185632957198*a-1238769313430830041)))*$

$t^2 + ((1/12880478333396107359233 28 * (10907712401095804191 * a - 28858336737850629436) * i + 1/12880478333396107359233 28 *$

$$(-1291093678388788948 * a + 2997908528350708049)) * r1 + (1/12880478333396107359233328 * (-62922498520157254561 * a +$$

$$16586080498631509024 6)*i+1/12880478333396107359233328*(22030143195312786958*a-55725571423750759839)))*t+((1/5152191333358442943693312*$$

$$(-13774615150544843951*a+36638942070520215209)*i+1/5152191333358442943693312*(12284326950567206407*a-32277903163103030811))*$$

$$r1+(1/5152191333358442943693312*(44363631982458322003*a-117150555034266529377)*i+1/5152191333358442943693312*$$

$$(-36422042115183541575*a+93302489978421347239)))) * b1 + (((1/54621 * (76 * a - 168) * i + 1/54621 * (-24 * a + 28)) * r1 +$$

$$(1/54621 * (320 * a - 980) * i - 4/2023 * a)) * t^7 + ((1/54621 * (-3 * a + 7) * i + 1/54621 * (-19 * a + 77)) * r1 + (1/18207 * (a +$$

$$35) * i + 1/54621 * (-29 * a - 77))) * t^6 + ((1/109242 * (-235 * a + 497) * i + 1/109242 * (75 * a - 91)) * r1 + (1/15606 * (-143 * a +$$

$$453) * i + 1/36414 * (115 * a + 21))) * t^5 + ((1/10404 * (a - 1) * i + 1/218484 * (111 * a - 497)) * r1 + (1/31212 * (-3 * a - 97) * i +$$

$$1/218484 * (193 * a + 581))) * t^4 + ((1/109242 * (94 * a - 175) * i + 1/18207 * (-5 * a + 7)) * r1 + (1/72828 * (267 * a - 917) * i +$$

$$1/218484 * (-291 * a - 175))) * t^3 + ((1/62424 * (-3 * a - 4) * i + 1/436968 * (-76 * a + 441)) * r1 + (1/436968 * (27 * a + 602) * i +$$

$$1/436968 * (-166 * a - 735))) * t^2 + ((1/194208 * (-17 * a + 21) * i + 1/1747872 * (45 * a - 91)) * r1 + (1/1747872 * (-625 * a +$$

$$2513) * i + 1/582624 * (83 * a + 119))) * t + ((1/166464 * (a + 5) * i + 1/3495744 * (25 * a - 385)) * r1 + (1/205632 * (-3 * a - 35) *$$

$$i + 1/499392 * (27 * a + 145))))) * b2^2 + ((((1/40251494791862835497604 * (23695630189160737075 * a - 64566096785659537257) *$$

$$i + 1/40251494791862835497604 * (17793635669035798335 * a - 45089379492126638683)) * r1 + (1/2367734987756637382212 *$$

$$(-647501187025667591*a+528150204273477009)*i+1/2367734987756637382212*(-836020781748421699*a+2406551623686017451)))*$$

$$t^7 + ((1/80502989583725670995208 * (-935344457598283095 * a + 5926017736059777643) * i + 1/80502989583725670995208 *$$

$$(327774436122305381*a-6684377111628741343))*r1+(1/47354699755132747644 24*(391736235816895130 9*a-9670843201805365573)*$$

$$i + 1/473546997551327476442 4 * (513229578058252709 * a - 1361148176376823163))) * t^6 + ((1/161005979167451341990416 *$$

$$(-167311738592957962593 * a + 448649847460879409219) * i + 1/161005979167451341990416 * (-125360638368091324951 * a +$$

$$319029479956151764723))*r1+(1/94709399510265495288 48*(976121064048099180 9*a-1869017460472272835 9)*i+1/350775553741724056624*$$

$$(360405991978640033*a-992755953426741273)))*t^5+((1/40251494791862835497604*(262044325758463407 2*a-9197753527814126205)*$$

$$i + 1/161005979167451341990416 * (-1332360415352006541 7 * a + 298682159513412296 23)) * r1 + (1/2367734987756637382212 *$$

$$(-3351008111131206786*a+8387073441520680907)*i+1/94709399510265495288 48*(-500318129765083813*a+1633623778793448911)))*$$

$$t^4 + ((1/64402391666980536796166 4 * (342431810594207344827 * a - 897184478822280539821) * i + 1/64402391666980536796166 4 *$$

$$(255514608282382441115*a-653264786798389365915))*r1+(1/54119656863008854450 56*(-564398299705999903 9*a+133016838341887132 57)*$$

$$i + 1/378837598041061981153 92 * (-33704711274236673969 * a + 907642796583406362 01))) * t^3 + ((1/64402391666980536796166 4 *$$

$$(-476152345212263509 55*a+136518550376919981396)*i+1/92003416667115052565952*(489027224647893187 6*a-1203887052296929966 9))*$$

$$r1+(1/54119656863008854450 56*(372221727380676038 6*a-946532934119577529 7)*i+1/378837598041061981153 92*(-360811439461524346 1*$$

$$a + 8123611803544842234))) * t^2 + ((1/12880478333396107359233 28 * (-996240272660459145 33 * a + 254192191563275451556) * i +$$

$$1/12880478333396107359233 28 * (-733059521788971319 82 * a + 186813007937186725591)) * r1 + (1/75767519608212396230784 *$$

$$(204252363611781155 39*a-517234949927781937 50)*i+1/75767519608212396230784*(151860355021795894 84*a-411465018861101946 93)))*$$

$$t+((1/5152191333358442943693312*(104688511722207156713*a-276974774727682031945)*i+1/5152191333358442943693312*$$

$$(-54089520346743435331 * a + 141139954525701151881)) * r1 + (1/30307007843284958492313 6 * (-28694677330942130705 * a +$$

$$74569784277707992749)*i+1/30307007843284958492313 6*(13188259056743489127*a-33083699401039720937))))*b1+(((1/54621*$$

$$(-172*a-336)*i+1/3213*(24*a+28))*r1+(1/3213*(-40*a-308)*i+4/119*a))*t^7+((1/3213*(3*a+7)*i+1/54621*(a+175))*r1+(1/1071*(-a+35)*i+1/3213*(127*a-287)))*t^6+((1/109242*(481*a+931)*i+1/6426*(-75*a-91))*r1+(1/918*(13*a+159)*i+1/2142*(-115*a+21)))*t^5+((1/612*(-a-1)*i+1/218484*(507*a-2443))*r1+(1/1836*(3*a-97)*i+1/12852*(-851*a+1967)))*t^4+((1/109242*(-184*a-119)*i+1/1071*(5*a+7))*r1+(1/4284*(-a-399)*i+1/12852*(291*a-175)))*t^3+((1/3672*(3*a-4)*i+1/436968*(-1466*a+4641))*r1+(1/25704*(-27*a+602)*i+1/25704*(824*a-1995)))*t^2+((1/582624*(181*a-511)*i+1/102816*(-45*a-91))*r1+(1/102816*(-145*a+1463)*i+1/34272*(-83*a+119)))*t+((1/9792*(-a+5)*i+1/3495744*(3257*a-8519))*r1+(1/12096*(3*a-35)*i+1/29376*(-141*a+367)))))*$$

$$b2+(((1/1183867493878318691106*(-26625704964214347157*a+70564656420628570893)*i+1/1183867493878318691106*(1819580417758742781 9*a-46186125311620393757))*r1+(1/1183867493878318691106*(-19876644207376779813 5*a+51920174184236243539 5)*i+1/1183867493878318691106*(378426204806324073005*a-991167670657197954879)))*t^7+((1/1183867493878318691106*(-11013583153752777174*a+29660765781598066045)*i+1/1183867493878318691106*(33364292769601746923*a-88467669241997571274))*r1+(1/1183867493878318691106*(-10804518201751498046*a+28117579125141848587)*i+1/1183867493878318691106*(257318470512470516 53*a-74913864382189952434)))*t^6+((1/4735469975513274764424*(14091639304449964911 3*a-37589635336725148735 1)*i+1/4735469975513274764424*(-50846235625683769205*a+124137376706690774189))*r1+(1/4735469975513274764424*(1249928447897295656967*a-3263921620734340973449)*i+1/1578489991837758254808*(-838372698378966266473*a+219714323885194566 1697)))*t^5+((1/4735469975513274764424*(65159062260208582699*a-17512019474162277 4056)*i+1/4735469975513274764424*(-21001879489883652799 4*a+55620552331971158875 7))*r1+(1/4735469975513274764424*(65187918037319546457*a-16817749015382019231 8)*i+1/4735469975513274764424*(-10227514428130447254 4*a+31507706130235270701 1)))*t^4+((1/9470939951026549528848*(-55766827099369132671*a+152815012358190762482)*i+1/9470939951026549528848*(-93285320745184206364*a+251337736665549565479))*r1+(1/1352991421575221361264*(-14311366060747601854 1*a+37238057308987649132 6)*i+1/9470939951026549528848*(22939348650886491701 16*a-60170062523311937490 49)))*t^3+((1/3788375980410619811539 2*(-17852465651271817938 5*a+48235025571554759240 1)*i+1/5411965686300885445056*(97483143330573193693*a-25723444085750974557 1))*r1+(1/5411965686300885445056*(-24169331618481161417*a+63233890975164252449)*i+1/3788375980410619811539 2*(-20675646509003207680 3*a+39884100138184666948 5)))*t^2+((1/3788375980410619811539 2*(-41256993293616545472*a+10526478819265131524 9)*i+1/3788375980410619811539 2*(18181142189908700612 3*a-47939851741615783730 2))*r1+(1/3788375980410619811539 2*(35180682726063246466 4*a-90550903001671450858 1)*i+1/3788375980410619811539 2*(-10627329565053003074 63*a+27809334443048916515 10)))*t+((1/1515350392164247924615 68*(36090890 12981244719*a-11949171872517168079)*i+1/1515350392164247924615 68*(-23633247757905631014 5*a+62010139746664316640 3))*r1+(1/1515350392164247924615 68*(16474614922209030241*a-44054129124011808585)*i+1/1515350392164247924615 68*(64036931511255929588 1*a-16403802057815901956 99))))*b1+((1/3213*(-2668*a+6468)*i+1/3213*(1404*a-2716))*r1+(1/189*(-388*a+1036)*i+1/21*(52*a-84)))*t^7+((1/3213*(522*a-1498)*i+1/3213*(394*a-2366))*r1+(1/63*(-30*a+14)*i+1/189*(-26*a+238)))*t^6+((1/3213*(4331*a-10297)*i+1/3213*(-2115*a+4067))*r1+(1/27*(83*a-225)*i+1/21*(-85*a+133)))*t^5+((1/306*(-87*a+227)*i+1/6426*(-1215*a+7777))*r1+(1/54*(45*a-29)*i+1/378*(59*a-833)))*t^4+((1/12852*(-7715*a+17591)*i+1/4284*(1023*a-1981))*r1+(1/252*(-275*a+791)*i+1/756*(1359*a-1981)))*t^3+((1/1836*(270*a-589)*i+1/12852*(1019*a-7392))*$$

$r1+(1/756*(-342*a+343)*i+1/756*(-23*a+924)))*t^2+((1/4284*(313*a-658)*i+1/12852*(-207*a+434))*r1+(1/756*(67*a-$

$245)*i+1/126*(-27*a+35)))*t+(1/144*(-3*a+5)*i+1/51408*(-475*a+4207))*r1+1/3024*(225*a-301)*i+1/432*(-3*a-83),$

$w_1 3 := ((((1/40251494791862835497604 * (-3343857450724400027 * a + 8644965679114083921) * i + 1/5750213541694690785372 *$

$(-88165583614664017*a+199231673624024465))*r1+(1/40251494791862835497604*(-3874955865827848 1025*a+10157618054444 8022035)*$

$i + 1/40251494791862835497604 * (6115204175244660007 * a - 16641950197315174263))) * t^7 + ((1/11500427083389381570744 *$

$(-1103325360974906881*a+2969293885383787421)*i+1/8050298958372567 0995208*(2614670925608197371*a-6688368021651876385))*$

$r1 + (1/805029895837256709952 08 * (-9924895821951194351 * a + 26800550910932083495) * i + 1/805029895837256709952 08 *$

$(-9930518079159531641 * a + 26393745230662573911))) * t^6 + ((1/16100597916745134199041 6 * (15016972322023903391 * a -$

$38578790647798752113)*i+1/23000854166778763141488*(7128346234584054 59*a-1673413739060413423))*r1+(1/16100597916745134199041 6*$

$(24857109178973228408 1*a-652054621303686905843)*i+1/16100597916745134199041 6*(-4206223071196492642 5*a+1134654710581200804 73)))*$

$t^5 + ((1/16100597916745134199041 6 * (24243025284109320032 * a - 65260502707658897849) * i + 1/16100597916745134199041 6 *$

$(-9710765245712872050*a+24881539594903321981))*r1+(1/23000854166778763141488*(3403149054821899468*a-9175610280915108911)*$

$i + 1/23000854166778763141488 * (5310103214461816862 * a - 14059273339268811169))) * t^4 + ((1/64402391666980536796166 4 *$

$(493381383445029365*a-2601164249291598781)*i+1/64402391666980536796166 4*(-1294898114150459432 7*a+3184547479613336 5801))*$

$r1+(1/64402391666980536796166 4*(-4198388972472417694 39*a+1103385058699344209383)*i+1/64402391666980536796166 4*$

$(82828803435894230501 * a - 219932290146422901859))) * t^3 + ((1/32201195833490268398083 2 * (-19687282169303331536 * a +$

$52999431179912315427)*i+1/64402391666980536796166 4*(2161224653655476731 9*a-5554806842006742 8807))*r1+(1/64402391666980536796166 4*$

$(-5259309701711522791*a+13612885345635772133)*i+1/32201195833490268398083 2*(-4270342808085922172 0*a+1124517153289430124 21)))*$

$t^2+((1/12880478333396107359233 28*(-13344513170359051735*a+35220090412888949470)*i+1/12880478333396107359233 28*$

$(501790931054627073 6*a-12863931608288254333))*r1+(1/12880478333396107359233 28*(7931849566230330283 7*a-2093924875293589599 24)*$

$i+1/12880478333396107359233 28*(-21620671766189471070*a+55822614603167699231)))*t+((1/51521913333584429436933 12*$

$(27803799254747154167*a-74992018697172634407)*i+1/51521913333584429436933 12*(-2713827841478429891 3*a+7036938298673574 5759))*$

$r1+(1/51521913333584429436933 12*(-71062726965108515447*a+19250394156160054 8635)*i+1/51521913333584429436933 12*$

$(113808068335231588581*a-297102204537277500095))))*b1+((1/54621*(4*a-28)*i*r1+(1/54621*(-248*a+504)*i+1/54621*$

$(12*a-140)))*t^7+((1/18207*(a+7)*i+1/18207*(-3*a-7))*r1+(1/18207*(-a-7)*i+1/7803*(-a+19)))*t^6+((1/109242*$

$(-15*a+91)*i+1/109242*(-3*a-7))*r1+(1/109242*(891*a-1855)*i+1/36414*(-19*a+161)))*t^5+((1/10404*(-a-7)*i+$

$1/218484*(69*a+133))*r1+(1/10404*(a+11)*i+1/218484*(-69*a-721)))*t^4+((1/31212*(4*a-15)*i+1/72828*(2*a+7))*$

$r1+(1/31212*(-133*a+286)*i+1/218484*(75*a-490)))*t^3+((1/62424*(3*a+22)*i+1/436968*(-80*a-119))*r1+(1/436968*$

$(-27*a-364)*i+1/436968*(234*a+497)))*t^2+((1/1747872*(-81*a+203)*i+1/1747872*(-3*a-49))*r1+(1/1747872*(1019*$

$a-2205)*i+1/1747872*(-75*a+595)))*t+((1/166464*(-a-9)*i+1/3495744*(121*a+91))*r1+(1/205632*(3*a+35)*i+$

$1/3495744*(-423*a-413)))))*b2^2+((((1/40251494791862835497604*(-5354719567564339141 3*a+1447747707112836342 57)*$

$i + 1/5750213541694690785372 * (-4550287420828962221 * a + 11964067458158508547)) * r1 + (1/23677349877566373822 12 *$

$(-1948937807769565 71*a+2341019571851985367)*i+1/23677349877566373822 12*(2186620657413811303*a-5097599971689124977)))*$

$t^7+((1/11500427083389381570744*(2022135131245139743*a-5768602016356197209)*i+1/80502989583725670995208*(2259356883997777245*$

$$a - 9054093943560243557)) * r1 + (1/4735469975513274764424 * (-13179022491628862903 * a + 34402679261056308349) * i +$$
$$1/4735469975513274764424 * (-6136699242659547215 * a + 16534352380173259107))) * t^6 + ((1/161005979167451341990416 *$$
$$(358809022804395573889*a-963393360891167129581)*i+1/2300085416677876314148 8*(31267062823979257987*a-82237131237782725229))*$$
$$r1+(1/9470939951026549528848*(-886715569571369978 1*a+11471810373510869389)*i+1/9470939951026549528848*(-20814921194937243909*$$
$$a + 49590669993819800671))) * t^5 + ((1/161005979167451341990416 * (-65846207735972776451 * a + 182574783267499577906) *$$
$$i + 1/161005979167451341990416 * (-11686571784886540911 * a + 42352679768646935030)) * r1 + (1/135299142157522136 1264 *$$
$$(640147198783741770 1*a-167431479740302088 48)*i+1/135299142157522136 1264*(277180908970950787 7*a-750093056896487272 0)))*$$
$$t^4+((1/644023916669805367961664*(-669616261627408281617*a+177074907366748392 4819)*i+1/644023916669805367961664*$$
$$(-428459506044401975931 * a + 1126915804622434629215)) * r1 + (1/37883759804106198115392 * (56147049320902825847 * a -$$
$$127667331090605666885)*i+1/37883759804106198115392*(62235046639395154613*a-151333993104008706985)))*t^3+((1/644023916669805367961664*$$
$$(195508553770782020893*a-527758930433692452327)*i+1/161005979167451341990416*(10327707227081577020*a-33981975851317449991))*$$
$$r1 + (1/18941879902053099057696 * (-43061177672789603951 * a + 113148122967163226887) * i + 1/37883759804106198115392 *$$
$$(-32849411673919334917 * a + 89489335251025147 05))) * t^2 + ((1/12880478333396107359233 28 * (16212561605125569745 0 * a -$$
$$414647075408657519827) * i + 1/12880478333396107359233 28 * (11132538800523276867 3 * a - 2897361671820434556 08)) * r1 +$$
$$(1/75767519608212396230784*(-32311471986117482128*a+81434733872863868805)*i+1/75767519608212396230784*(-25412481411698361375*$$
$$a + 63292043836349589926))) * t + ((1/51521913333584429436933 12 * (-32187778880468263328 3 * a + 85483442378811680198 1) *$$
$$i+1/51521913333584429436933 12*(-81077839848797025061*a+24449478340885055804 5))*r1+(1/30307007843284958492313 6*$$
$$(868844802525482970 31*a-230499395709724120525)*i+1/30307007843284958492313 6*(267403504057733445 81*a-754403404650688976 33))))*$$
$$b1 + ((1/54621 * (212 * a - 308) * i * r1 + (1/3213 * (136 * a - 504) * i + 1/3213 * (-12 * a - 140))) * t^7 + ((1/1071 * (-a + 7) * i +$$
$$1/18207 * (233 * a - 581)) * r1 + (1/1071 * (a - 7) * i + 1/459 * (-13 * a + 25))) * t^6 + ((1/109242 * (-711 * a + 581) * i + 1/6426 * (3 *$$
$$a - 7)) * r1 + (1/6426 * (-429 * a + 1631) * i + 1/2142 * (19 * a + 161))) * t^5 + ((1/612 * (a - 7) * i + 1/218484 * (-5079 * a + 12551)) *$$
$$r1 + (1/612 * (-a + 11) * i + 1/12852 * (783 * a - 1603))) * t^4 + ((1/31212 * (86 * a + 57) * i + 1/4284 * (-2 * a + 7)) * r1 + (1/1836 * (53 *$$
$$a - 212) * i + 1/12852 * (-75 * a - 490))) * t^3 + ((1/3672 * (-3 * a + 22) * i + 1/436968 * (5546 * a - 13447)) * r1 + (1/25704 * (27 * a -$$
$$364) * i + 1/25704 * (-1032 * a + 2261))) * t^2 + ((1/1747872 * (-429 * a - 1967) * i + 1/102816 * (3 * a - 49)) * r1 + (1/102816 * (-361 *$$
$$a + 1617) * i + 1/102816 * (75 * a + 595))) * t + ((1/9792 * (a - 9) * i + 1/3495744 * (-6943 * a + 16709)) * r1 + (1/12096 * (-3 * a + 35) *$$
$$i + 1/205632 * (1473 * a - 3395))))) * b2 + (((1/1183867493878318691106 * (1542851299772087946577 * a - 4049904385232762701 89) *$$
$$i + 1/169123927696902670158 * (-11878201992033712117 * a + 31600700882607192767)) * r1 + (1/1183867493878318691106 *$$
$$(676174856458875238179*a-176486048777014419 7475)*i+1/1183867493878318691106*(-82268398808377772635 7*a+21879481658021058354 99)))*$$
$$t^7+((1/169123927696902670158*(7305503558193491605*a-19762923021220999589)*i+1/1183867493878318691106*(-81960240000014947923*$$
$$a + 215216395768136149093)) * r1 + (1/1183867493878318691106 * (302905933491725952071 * a - 805023796995081229237) * i +$$
$$1/1183867493878318691106 * (-49857519420104324881 * a + 134953166516481684357))) * t^6 + ((1/4735469975513274764424 *$$
$$(-901905285437744400271*a+237076259415521000 5891)*i+1/676495710787610680632*(584653016567078226 47*a-155557235557532571229))*$$
$$r1 + (1/4735469975513274764424 * (-4039153031735814565749 * a + 105346495979224894785 37) * i + 1/4735469975513274764424 *$$
$$(5234731000467028014267 * a - 13925446099663399513457))) * t^5 + ((1/4735469975513274764424 * (-27862231611879034560 7 *$$

$a + 756547316563260740056) * i + 1/4735469975513274764424 * (527925313956245468958 * a - 1384479095992866580469)) * r1 +$

$(1/6764957107876106806 32*(-269660968490546874773*a+715606248394436703328)*i+1/6764957107876106806 32*(1770792106700689 8026*$

$a - 50358924519269706187))) * t^4 + ((1/4735469975513274764424 * (294177482957362243784 * a - 777097602009479942317) *$

$i + 1/4735469975513274764424 * (-46615602410863060989 * a + 12440995563860382941)) * r1 + (1/4735469975513274764424 *$

$(140287225788992933561 9 * a - 3651189492563113437692) * i + 1/1183867493878318691106 * (-541840074270938356537 * a +$

$144241430581058969454 8))) * t^3 + ((1/3788375980410619811539 2 * (559333025883038826499 * a - 154363678827780223378 5) * i +$

$1/3788375980410619811539 2 * (-181439002895585124255 1 * a + 474424279522447502567 9)) * r1 + (1/3788375980410619811539 2 *$

$(590060944642940953412 5 * a - 15588840187128310868675) * i + 1/3788375980410619811539 2 * (128973736875123184105 9 * a -$

$329450000343226380666 3))) * t^2 + ((1/3788375980410619811539 2 * (-50748162622130651992 * a + 144290632120237811257) * i +$

$1/3788375980410619811539 2 * (-223696809951901807593 * a + 595074826148473058612)) * r1 + (1/3788375980410619811539 2 *$

$(-433000237234228209904 * a + 1105114114741899972675) * i + 1/3788375980410619811539 2 * (155318623228114063312 5 * a -$

$413895013572371003570 8))) * t + ((1/151535039216424792461568 * (161697977182036385057 * a - 402493146264364877463) * i +$

$1/151535039216424792461568 * (735157618555847734315 * a - 191438608910663310975 1)) * r1 + (1/151535039216424792461568 *$

$(-174904656316304696755 7 * a + 4543615529040422525411) * i + 1/151535039216424792461568 * (-235712156113320157839 9 * a +$

$613660462401817307341 9))))*b1+((1/3213*(-388*a+532)*i+1/51*(4*a-12))*r1+(1/189*(580*a-1764)*i+1/189*(-108*a+$

$532)))*t^7+((1/1071*(-174*a+182)*i+1/1071*(-38*a+490))*r1+(1/63*(30*a-70)*i+1/27*(14*a-98)))*t^6+((1/3213*(843*$

$a-1631)*i+1/3213*(-369*a+1379))*r1+(1/189*(-1059*a+3199)*i+1/63*(75*a-329)))*t^5+((1/306*(87*a-91)*i+1/6426*$

$(423*a-5705))*r1+(1/18*(-15*a+31)*i+1/378*(-267*a+2233)))*t^4+((1/1836*(-379*a+945)*i+1/4284*(219*a-1099))*r1+$

$(1/108*(323*a-965)*i+1/756*(-549*a+2177)))*t^3+((1/1836*(-270*a+283)*i+1/12852*(-509*a+6916))*r1+(1/756*(342*$

$a-581)*i+1/756*(165*a-2072)))*t^2+((1/25704*(1392*a-3773)*i+1/25704*(-279*a+1330))*r1+(1/1512*(-646*a+1911)*$

$i+1/1512*(153*a-644)))*t+(1/48*(a-1)*i+1/51408*(137*a-4123))*r1+1/3024*(-225*a+301)*i+1/3024*(-51*a+1141),$

$w_14 := ((((1/40251494791862835497604 * (674033425632271521 * a - 1900198024318121773) * i + 1/40251494791862835497604 *$

$(-7265721164158254439*a+19338392053384192817))*r1+(1/57502135416946907853 72*(113287310298741508 1*a-30645618365441640 33)*$

$i+1/40251494791862835497604 * (-40619384595179013573 * a + 108400130546350466327))) * t^7 + ((1/20125747395931417748802 *$

$(110440365744694896*a-300694141821905671)*i+1/40251494791862835497604*(-3969872624415170105*a+1034042178216298664 7))*$

$r1 + (1/40251494791862835497604 * (-1500300140162082589 * a + 4024995918687960973) * i + 1/20125747395931417748802 *$

$(-4497476319659542848 * a + 12135703201784797691))) * t^6 + ((1/53668659722483780663472 * (-1063315432791931359 * a +$

$3061481008961538443)*i+1/76669513889262543804 96*(194878090472263290 5*a-51861767672394316 39))*r1+(1/16100597916745134199041 6*$

$(-54227905191839469943*a+146447440972739064627)*i+1/16100597916745134199041 6*(24867043196952287329 9*a-66375472951080039 7981)))*$

$t^5 + ((1/16100597916745134199041 6 * (-1431629784237178489 * a + 3867523982580944315) * i + 1/80502989583725670995208 *$

$(12329932737987081961*a-32017123813304168028))*r1+(1/16100597916745134199041 6*(99176175751250540 89*a-26799777970317582895)*$

$i + 1/80502989583725670995208 * (24032539020479908722 * a - 65350561394417463353))) * t^4 + ((1/64402391666980536796166 4 *$

$(68854726266768134 9*a-2922250992850949907)*i+1/64402391666980536796166 4*(-46198892424600344067*a+12282510765848001076 3))*$

$r1 + (1/64402391666980536796166 4 * (104460404353240993569 * a - 280753067819930725999) * i + 1/64402391666980536796166 4 *$

$$(-372807060103996002111 * a + 995420638845636114503))) * t^3 + ((1/64402391666980536796166 4 * (2647399668083339259 * a -$$

$$7138334686256546500)*i+1/64402391666980536796166 4*(-3859736467269530497 6*a+99470285944966822319))*r1+(1/64402391666980536796166 4*$$

$$(-18643438653960264888*a+50591327090447808371)*i+1/64402391666980536796166 4*(-42334632061148764675*a+12017866269877998843 6)))*$$

$$t^2 + ((1/8586985555597404906155 52 * (1558759685040474471 * a - 4085779251770619613) * i + 1/2576095666679221471846656 *$$

$$(-6444789098229463389 * a + 17340562131907523341)) * r1 + (1/3680136666684602102638 08 * (-7911007039310633071 * a +$$

$$210144628147081701 45)*i+1/8586985555597404906155 52*(31383403182697820647*a-83821260210955143275)))*t+((1/5152191333358442943693312*$$

$$(-260706818449797473 1*a+718060437579085477 1)*i+1/5152191333358442943693312*(23267505224017332479*a-58489400923944796449))*$$

$$r1 + (1/5152191333358442943693312 * (17944133300586717491 * a - 48108370785736123415) * i + 1/5152191333358442943693312 *$$

$$(-48180668699473834679 * a + 115934911358154878045)))) * b1 + (((1/54621 * (-64 * a + 196) * i + 1/54621 * (24 * a + 28)) * r1 +$$

$$(1/7803 * (-28 * a + 56) * i + 1/7803 * (12 * a + 8))) * t^7 + ((1/18207 * (a - 7) * i + 1/54621 * (53 * a - 133)) * r1 + (1/18207 * (-a +$$

$$7) * i + 1/18207 * (-39 * a + 119))) * t^6 + ((1/54621 * (101 * a - 315) * i + 1/18207 * (-12 * a - 14)) * r1 + (1/54621 * (255 * a - 455) *$$

$$i + 1/18207 * (-40 * a - 28))) * t^5 + ((1/31212 * (-3 * a + 19) * i + 1/218484 * (-373 * a + 959)) * r1 + (1/218484 * (33 * a - 217) * i +$$

$$1/218484 * (961 * a - 2891))) * t^4 + ((1/109242 * (-81 * a + 266) * i + 1/72828 * (17 * a + 21)) * r1 + (1/218484 * (-191 * a + 63) * i +$$

$$1/12138 * (8 * a + 7))) * t^3 + ((1/62424 * (3 * a - 17) * i + 1/109242 * (97 * a - 259)) * r1 + (1/436968 * (-57 * a + 329) * i + 1/218484 *$$

$$(-620 * a + 1827))) * t^2 + ((1/1747872 * (117 * a - 427) * i + 1/249696 * (-3 * a - 5)) * r1 + (1/1747872 * (-359 * a + 1421) * i +$$

$$1/1747872 * (3 * a - 119))) * t + ((1/582624 * (-4 * a + 21) * i + 1/1747872 * (-230 * a + 651)) * r1 + (1/102816 * (3 * a - 14) * i +$$

$$1/1747872 * (919 * a - 2646))))) * b2^2 + ((((1/4025149479186283549760 4 * (75682478385681058569 * a - 194384932148779404515) *$$

$$i + 1/4025149479186283549760 4 * (-39093768640790472125 * a + 97838307524241123973)) * r1 + (1/3382478555393805340316 *$$

$$(-100755574352845264 3*a+27334272590834594 85)*i+1/2367734987756637382212*(-192356978179361827 5*a+25197503044662983 51)))*$$

$$t^7 + ((1/10062873697965708874401 * (-9402802630314783828 * a + 24758494939608500546) * i + 1/4025149479186283549760 4 *$$

$$(60996023949574984 43*a-12341466215653865997))*r1+(1/2367734987756637382212*(12089833773116559755*a-31610443834987857875)*$$

$$i + 1/1183867493878318691106 * (-2837183436643051539 * a + 7689996871036634147))) * t^6 + ((1/5366865972248378066347 2 *$$

$$(-17636835452689534510 7*a+45340760040050629 7385)*i+1/7666951388926254380496*(11924733825503476919*a-30003375797873185359))*$$

$$r1 + (1/9470939951026549528848 * (60531312736678275785 * a - 162670954415410480851) * i + 1/9470939951026549528848 *$$

$$(64691855174986306 09*a-109188432524739505)))*t^5+((1/1610059791674513419904 16*(28501208346554537008 9*a-74841417125576405051 3)*$$

$$i + 1/8050298958372567099520 8 * (-29219048603610737332 * a + 64394128521869479515)) * r1 + (1/9470939951026549528848 *$$

$$(-86445072201625782779*a+22632559804441366251 5)*i+1/4735469975513274764424*(19448201810627135736*a-52599103284613058495)))*$$

$$t^4+((1/6440239166698053679616 64*(10771346625173419537 99*a-27764075693248262713 59)*i+1/6440239166698053679616 64*$$

$$(-42311288591584573347 * a + 108532664729296987 01)) * r1 + (1/3788375980410619811539 2 * (-16583515860195906485 7 *$$

$$a + 440564615868213184313) * i + 1/3788375980410619811539 2 * (13349091739789173813 * a - 64421257060111146475))) * t^3 +$$

$$((1/6440239166698053679616 64 * (-664737955982833469097 * a + 1741249237771028187370) * i + 1/6440239166698053679616 64 *$$

$$(176765485861115428468 * a - 423571981393908723047)) * r1 + (1/3788375980410619811539 2 * (183764902471494943314 * a -$$

$$481620669957604668211) * i + 1/3788375980410619811539 2 * (-76278793310361647815 * a + 203982208957638274632))) * t^2 +$$

$$((1/8586985555597404906155 52 * (-204807122683319802827 * a + 529895052677773170171) * i + 1/2576095666679221471846656 *$$

$$(159350622136546282623 * a - 434999299546796323213)) * r1 + (1/216478627452035417802224 * (18831609471271637279 * a - 49627439071557450435)*i+1/505116797388082641538 56*(-101128592296280567 77*a+30195854621708950559)))*t+((1/515219133335844294369331 2*(899715038730429586839 * a - 2353253780498610455165) * i + 1/515219133335844294369331 2 * (-300231583757051369333 * a + 764524702836406335705)) * r1 + (1/303070078432849584923136 * (-221872082141373243007 * a + 581401522277882347489) * i + 1/303070078432849584923136 * (81250230635238337261 * a - 211836334457166972973)))) * b1 + (((1/54621 * (-536 * a + 1988) * i + 1/3213 * (-24 * a + 28)) * r1 + (1/459 * (-4 * a + 32) * i + 1/459 * (-12 * a + 8))) * t^7 + ((1/1071 * (-a - 7) * i + 1/54621 * (-635 * a + 1645)) * r1 + (1/1071 * (a + 7) * i + 1/1071 * (25 * a - 63))) * t^6 + ((1/54621 * (817 * a - 2961) * i + 1/1071 * (12 * a - 14)) * r1 + (1/3213 * (39 * a - 329) * i + 1/1071 * (40 * a - 28))) * t^5 + ((1/1836 * (3 * a + 19) * i + 1/218484 * (4255 * a - 11123)) * r1 + (1/12852 * (-33 * a - 217) * i + 1/12852 * (-611 * a + 1519))) * t^4 + ((1/109242 * (-639 * a + 2128) * i + 1/4284 * (-17 * a + 21)) * r1 + (1/12852 * (-47 * a + 441) * i + 1/714 * (-8 * a + 7))) * t^3 + ((1/3672 * (-3 * a - 17) * i + 1/109242 * (-1033 * a + 2737)) * r1 + (1/25704 * (57 * a + 329) * i + 1/12852 * (382 * a - 945))) * t^2 + ((1/1747872 * (825 * a - 1673) * i + 1/14688 * (3 * a - 5)) * r1 + (1/102816 * (37 * a - 217) * i + 1/102816 * (-3 * a - 119))) * t + ((1/34272 * (4 * a + 21) * i + 1/1747872 * (2300 * a - 5985)) * r1 + (1/6048 * (-3 * a - 14) * i + 1/102816 * (-527 * a + 1344))))) * b2 + (((1/11838674938783186911 06 * (50115412346638069983 * a - 128430153952905160825) * i + 1/11838674938783186911 06 * (20216511321535406819 3 * a - 535955793835177542001)) * r1 + (1/16912392769690267015 8 * (90099641321111347427*a-237826346594969026125)*i+1/11838674938783186911 06*(76168108185226419156 3*a-204423332970424072907 5)))*t^7 + ((1/2367734987756637382212 * (188365463022403363209 * a - 498271150436731064113) * i + 1/2367734987756637382212 * (935959215896538937 01*a-240715384901364419907))*r1+(1/2367734987756637382212*(432350937704695382 23*a-116008143679503557251)*i + 1/2367734987756637382212 * (606269534174930279103 * a - 161597202109636697 1373))) * t^6 + ((1/15784899918377582548 08 * (-703075861483190340 17*a+176528146978021807171)*i+1/22549857026253689 3544*(-575078285330451138 21*a+152290226860461524041))*r1 + (1/4735469975513274764424 * (-3993871822922867427013 * a + 1054787543020424751 0627) * i + 1/4735469975513274764424 * (-4516937850115397795195 * a + 1214492457679118628 3299))) * t^5 + ((1/4735469975513274764424 * (-623918917980593019112 * a + 165126020841473348 1677) * i + 1/4735469975513274764424 * (-252336471226243011025 * a + 642730284805052428740)) * r1 + (1/4735469975513274764424 * (138123263369366414878 * a - 357353875156663427665) * i + 1/4735469975513274764424 * (-188554253656951442888 7 * a + 503466803951393470 1642))) * t^4 + ((1/9470939951026549528848 * (-47533004406377040029 * a + 151878656694465979320) * i + 1/9470939951026549528848 * (843111012384535007172 * a - 222378144017360304 8137)) * r1 + (1/9470939951026549528848*(32932374057426993382 05*a-871362640307949523 5170)*i+1/9470939951026549528848*(30684680181816214676 70*a - 829302447046441004 3003))) * t^3 + ((1/3788375980410619811 5392 * (228164170991861080 2489 * a - 604042338093950021 8169) * i + 1/3788375980410619811 5392 * (463485989650162261045 * a - 112466902446436633 2959)) * r1 + (1/3788375980410619811 5392 * (-273007174247218058 6853 * a + 719453051307858679 5901) * i + 1/3788375980410619811 5392 * (589017959195644348 7359 * a - 158211499162342140 45669))) * t^2 + ((1/2525583986940413207 6928 * (173772324333452638 63 * a - 467123884449108480419) * i + 1/7576751960821239623 0784 * (-278426322065480039571 * a + 709611689223206784385)) * r1 + (1/1082393137260177089 0112 * (-337996696332915559807*a+899626967165843702403)*i+1/8418613289801377358976*(-92444769349882495561*a+260596006981851906939)))*t+((1/1515350392164247924615 68*(-106329253321706313 8907*a+281476915041264306 3059)*i+1/1515350392164247924615 68*(23447677640070674352 5 * a - 673277087651235615471)) * r1 + (1/1515350392164247924615 68 * (34386649652048030242 19 * a -

$$910271748896152592303 9)*i+1/151535039216424792461568*(-1778380165780106896697*a+4897532564341795813799))))*b1+$$
$$((1/3213*(2596*a-7084)*i+1/3213*(-1908*a+2324))*r1+(1/27*(68*a-124)*i+1/27*(-36*a+20)))*t^7+((1/1071*(-174*$$
$$a+658)*i+1/3213*(-1382*a+3766))*r1+(1/63*(30*a-98)*i+1/63*(50*a-98)))*t^6+((1/3213*(-4064*a+11340)*i+1/1071*$$
$$(996*a-1288))*r1+(1/189*(-636*a+1036)*i+1/63*(108*a-28)))*t^5+((1/918*(279*a-1007)*i+1/6426*(4913*a-13363))*$$
$$r1+(1/378*(-369*a+1225)*i+1/378*(-625*a+1379)))*t^4+((1/12852*(6453*a-19033)*i+1/476*(-177*a+259))*r1+(1/756*$$
$$(571*a-315)*i+1/84*(-25*a-49)))*t^3+((1/1836*(-315*a+1081)*i+1/12852*(-5147*a+13979))*r1+(1/756*(477*a-1589)*$$
$$i+1/756*(841*a-2079)))*t^2+((1/25704*(-1194*a+3997)*i+1/3672*(117*a-232))*r1+(1/1512*(148*a-679)*i+1/1512*$$
$$(-117*a+406)))*t+(1/672*(19*a-63)*i+1/102816*(5867*a-15981))*r1+1/6048*(-711*a+2303)*i+1/6048*(-1355*a+3633),$$
$$w_15 := ((((1/4025149479186283549760 4*(-8380315164608538279*a+22366597102958454563)*i+1/4025149479186283549760 4*$$
$$(-5435985856599437933*a+14513734320895808355))*r1+(1/575021354169469078537 2*(-4906306763523151207*a+13026151848980975083)*$$
$$i+1/4025149479186283549760 4*(-21985414533050555375*a+57719166067729788601)))*t^7+((1/115004270833893815707 44*$$
$$(-783039152120492155*a+207163740566119322 1)*i+1/4735469975513274764424*(-174695717202074069*a+463993767105792301))*$$
$$r1+(1/805029895837256709952 08*(-1993258464273331998 1*a+52556809040762058151)*i+1/805029895837256709952 08*$$
$$(-8723863364095444697*a+23022845586309193205)))*t^6+((1/161005979167451341990416*(50157088747399926179*a-$$
$$133977617936173260431)*i+1/161005979167451341990416*(3282122056922997840 7*a-87752126364773025337))*r1+(1/230008541667787631414 88*$$
$$(28952545846974243387*a-76834510043688670651)*i+1/161005979167451341990416*(13087269928488469578 1*a-342139666213163046655)))*$$
$$t^5+((1/161005979167451341990416*(16253244734770420611*a-43024471525909459726)*i+1/161005979167451341990416*$$
$$(9149577290151086125*a-24308938383571033864))*r1+(1/161005979167451341990416*(58167387594748854593*a-153306351232951930916)*$$
$$i+1/161005979167451341990416*(24015444435941802859*a-63430955627994598966)))*t^4+((1/214674638889935122653888*$$
$$(-2340505148165829259 1*a+62660324955092784409)*i+1/644023916669805367961664*(-4699346756891314221 5*a+12618806623537831119 9))*$$
$$r1+(1/214674638889935122653888*(-90464671722043415319*a+23965446161740962265)*i+1/644023916669805367961664*$$
$$(-18030945492325846561 5*a+46541623400125035034 3)))*t^3+((1/644023916669805367961664*(-22482844411475276875*$$
$$a+59606586748969961225)*i+1/805029895837256709952 08*(-1726693643224866822*a+4599398888345870063))*r1+$$
$$(1/322011958334902683980832*(-3716877838763300269 3*a+97785633823987027051)*i+1/644023916669805367961664*(-2497036559693586960 5*$$
$$a+66273499047346451049)))*t^2+((1/128804783333961073592332 8*(6229419319272303827*a-16869392026923726206)*i+$$
$$1/128804783333961073592332 8*(4843829085385891142*a-13435263380913215709))*r1+(1/128804783333961073592332 8*$$
$$(151063377909164772 41*a-38920513679675258554)*i+1/128804783333961073592332 8*(14936337418450473570*a-35300904236470902911)))*$$
$$t+((1/515219133335844294369331 2*(7774038834413195759*a-20872760767369769965)*i+1/515219133335844294369331 2*$$
$$(6341203240114484233*a-17062630561299896685))*r1+(1/515219133335844294369331 2*(4447386884655783205*a-10718929451814444259)*$$
$$i+1/515219133335844294369331 2*(-14561349558998591121*a+38446456426898195473))))*b1+((1/54621*(-4*a+28)*$$
$$i*r1+(1/54621*(248*a-504)*i+1/54621*(-12*a+140)))*t^7+((1/18207*(a+7)*i+1/18207*(-3*a-7))*r1+$$
$$(1/18207*(-a-7)*i+1/7803*(-a+19)))*t^6+((1/109242*(15*a-91)*i+1/109242*(3*a+7))*r1+(1/109242*(-891*$$
$$a+1855)*i+1/36414*(19*a-161)))*t^5+((1/10404*(-a-7)*i+1/218484*(69*a+133))*r1+(1/10404*(a+11)*i+$$
$$1/218484*(-69*a-721)))*t^4+((1/31212*(-4*a+15)*i+1/72828*(-2*a-7))*r1+(1/31212*(133*a-286)*i+$$

$$1/218484*(-75*a+490)))*t^3+((1/62424*(3*a+22)*i+1/436968*(-80*a-119))*r1+(1/436968*(-27*a-364)*i+1/436968*(234*a+497)))*t^2+((1/1747872*(81*a-203)*i+1/1747872*(3*a+49))*r1+(1/1747872*(-1019*a+2205)*i+1/1747872*(75*a-595)))*t+((1/166464*(-a-9)*i+1/3495744*(121*a+91))*r1+(1/205632*(3*a+35)*i+1/3495744*(-423*a-413)))))*b2^2+((((1/40251494791862835497604*(-30807661735178699799*a+85731372270206942461)*i+1/40251494791862835497604*(-35284312758355798975*a+87894591055764425583))*r1+(1/338247855393805340316*(968877002837125493*a-2348320438354559255)*i+1/23677349877566373822 12*(-651934508227192 1825*a+1675085490206274 2881)))*t^7+((1/1150042708338938157074 4*(327059665916909721 1*a-947109844019130534 3)*i+1/8050298958372567099520 8*(7968011674679240179*a-1605195258638824 6913))*r1+(1/4735469975513274764424*(-709663519070245 4995*a+169820646444543 95899)*i+1/4735469975513274764424*(-9671627506930106873*a+25566976631056449407)))*t^6+((1/1610059791674513419904 16*(227239968690754287355*a-624127963118160353797)*i+1/1610059791674513419904 16*(210740871587063762681*a-523547515868163336389))*r1+(1/13529914215752213612 64*(-730240138332877090 5*a+180268374940176322 67)*i+1/9470939951026549528848*(38708418830387061127*a-1006059528811102041 43)))*t^5+((1/1610059791674513419904 16*(-80599962787556169708*a+229239072348646643899)*i+1/1610059791674513419904 16*(-42736787776107830296*a+95033772914290026607))*r1+(1/9470939951026549528848*(240479307777031141 04*a-581166268223174205 53)*i+1/9470939951026549528848*(33038394001035451720*a-86948300155900133293)))*t^4+((1/2146746388899351226538 88*(-167507768716757951719*a+450066740295488710219)*i+1/6440239166698053679616 64*(-296344274628820659217*a+728784983587224027051))*r1+(1/12627919934702066038464*(38907941691794378621*a-987155164399887498 89)*i+1/3788375980410619811539 2*(-522602891190184248 93*a+140121890951680541407)))*t^3+((1/3220119583349026839808 32*(864480747559795240 13*a-237082738236626397010)*i+1/6440239166698053679616 64*(142441985031666684531*a-339538386265475188183))*r1+(1/3788375980410619811539 2*(-477304575650615535 23*a+118347387593948704709)*i+1/1894187990205309905769 6*(-32220824260360816291*a+84068803511147469462)))*t^2+((1/1288047833339610735923328*(153897267029538411502*a-402299785950350937985)*i+1/1288047833339610735923328*(270091086363446989 81*a-615099140684335290 06))*r1+(1/7576751960821239623078 4*(-374361794353802385 58*a+977507566749003374 21)*i+1/7576751960821239623078 4*(3606367432277986167*a-1222445318195968449 8)))*t+((1/5152191333358442943693312*(-219862431684114959165*a+577782670317269627833)*i+1/5152191333358442943693312*(-259660982292601655317*a+652008134752865498895))*r1+(1/3030700784328495849231 36*(549748560429516574 45*a-141798107807861506309)*i+1/3030700784328495849231 36*(68476717320754676433*a-174799311240282076943))))*b1+((1/54621*(-212*a+308)*i*r1+(1/3213*(-136*a+504)*i+1/3213*(12*a+140)))*t^7+((1/1071*(-a+7)*i+1/18207*(233*a-581))*r1+(1/1071*(a-7)*i+1/459*(-13*a+25)))*t^6+((1/109242*(711*a-581)*i+1/6426*(-3*a+7))*r1+(1/6426*(429*a-1631)*i+1/2142*(-19*a-161)))*t^5+((1/612*(a-7)*i+1/218484*(-5079*a+12551))*r1+(1/612*(-a+11)*i+1/12852*(783*a-1603)))*t^4+((1/31212*(-86*a-57)*i+1/4284*(2*a-7))*r1+(1/1836*(-53*a+212)*i+1/12852*(75*a+490)))*t^3+((1/3672*(-3*a+22)*i+1/436968*(5546*a-13447))*r1+(1/25704*(27*a-364)*i+1/25704*(-1032*a+2261)))*t^2+((1/1747872*(429*a+1967)*i+1/102816*(-3*a+49))*r1+(1/102816*(361*a-1617)*i+1/102816*(-75*a-595)))*t+((1/9792*(a-9)*i+1/3495744*(-6943*a+16709))*r1+(1/12096*(-3*a+35)*i+1/205632*(1473*a-3395)))))*b2+(((1/1183867493878318691106*(22462218224300141706 9*a-596235361853125562155)*i+1/1183867493878318691106*(221495846148341183 5*a-826225992298743823 5))*r1+(1/1691239276969026701 58*(932640909577051624 61*a-245424509679682429727)*$$

$i + 1/1183867493878318691106 * (58010317928075194921 * a - 141459703365608814389))) * t^7 + ((1/84561963848451335079 *$

$(2649285923679505757*a-7074882934392015510)*i+1/591933746939159345553*(-4367578852402827841*a+11955617892549669842))*$

$r1+(1/591933746939159345553*(130481410306157773775*a-346922125605080731013)*i+1/591933746939159345553*(5317758823880746300*$

$a-16615213391546515270)))*t^6+((1/4735469975513274764424*(-139428142250868121793 5*a+370405215037790 8169617)*i+$

$1/4735469975513274764424*(-2167294152811874693*a+23557784327034005909))*r1+(1/6764957107876106806 32*(-524843285455817243691*$

$a + 137887268749259227 0541) * i + 1/4735469975513274764424 * (-429171657445547588303 * a + 103547307191545894 2647))) *$

$t^5 + ((1/4735469975513274764424 * (-203350727387663999061 * a + 543491207610719333672) * i + 1/4735469975513274764424 *$

$(60562494426831547042 * a - 167243485741599499267)) * r1 + (1/4735469975513274764424 * (-15923274013252310467 51 * a +$

$423454600146277151926 6) * i + 1/4735469975513274764424 * (-101017917354956115020 * a + 306245439844945968899))) * t^4 +$

$((1/31569799836755165096 16*(360885623522930466545*a-960806446356755786907)*i+1/947093995102654952 8848*(-219309782339751552 49*$

$a + 377064187679605036 59)) * r1 + (1/31569799836755165096 16 * (689753090541346842749 * a - 179970543238237402 1827) * i +$

$1/947093995102654952 8848 * (472262593834681393767 * a - 112347139980110567 2933))) * t^3 + ((1/378837598041061981153 92 *$

$(402437395822793134681 * a - 108017929759533040757 3) * i + 1/378837598041061981153 92 * (-248708450605441308 7 * a +$

$694512984257071703297)) * r1 + (1/378837598041061981153 92 * (479042459805508713699 5 * a - 127399181140346154520 67) * i +$

$1/378837598041061981153 92 * (629983276864865116895 * a - 184765120997094956 85))) * t^2 + ((1/378837598041061981153 92 *$

$(-321848447737188643141*a+863475058251496035352)*i+1/37883759804106198115392*(29850111279244445516*a-64263272018296691445))*$

$r1 + (1/378837598041061981153 92 * (241946302299403785893 * a - 690813906598386552856) * i + 1/378837598041061981153 92 *$

$(-322713315534878791044 * a + 769760346935915308813))) * t + ((1/151535039216424792461568 * (153465333892354632695 * a -$

$402118264634914899379) * i + 1/151535039216424792461568 * (164690730267401534659 * a - 460581418424528611077)) * r1 +$

$(1/151535039216424792461568 * (-12209273473133225902 19 * a + 323873078499925174982 3) * i + 1/151535039216424792461568 *$

$(-546850208593781953047 * a + 1559888331136511808649)))) * b1 + ((1/3213 * (388 * a - 532) * i + 1/51 * (-4 * a + 12)) * r1 +$

$(1/189 * (-580 * a + 1764) * i + 1/189 * (108 * a - 532))) * t^7 + ((1/1071 * (-174 * a + 182) * i + 1/1071 * (-38 * a + 490)) * r1 + (1/63 *$

$(30 * a - 70) * i + 1/27 * (14 * a - 98))) * t^6 + ((1/3213 * (-843 * a + 1631) * i + 1/3213 * (369 * a - 1379)) * r1 + (1/189 * (1059 * a -$

$3199) * i + 1/63 * (-75 * a + 329))) * t^5 + ((1/306 * (87 * a - 91) * i + 1/6426 * (423 * a - 5705)) * r1 + (1/18 * (-15 * a + 31) * i + 1/378 *$

$(-267 * a + 2233))) * t^4 + ((1/1836 * (379 * a - 945) * i + 1/4284 * (-219 * a + 1099)) * r1 + (1/108 * (-323 * a + 965) * i + 1/756 *$

$(549 * a - 2177))) * t^3 + ((1/1836 * (-270 * a + 283) * i + 1/12852 * (-509 * a + 6916)) * r1 + (1/756 * (342 * a - 581) * i + 1/756 *$

$(165 * a - 2072))) * t^2 + ((1/25704 * (-1392 * a + 3773) * i + 1/25704 * (279 * a - 1330)) * r1 + (1/1512 * (646 * a - 1911) * i + 1/1512 *$

$(-153 * a + 644))) * t + (1/48 * (a - 1) * i + 1/51408 * (137 * a - 4123)) * r1 + 1/3024 * (-225 * a + 301) * i + 1/3024 * (-51 * a + 1141),$

$w_1 6 := ((((1/40251494791862835497604 * (-2032078796993678841 * a + 5439126824113408727) * i + 1/40251494791862835497604 *$

$(-1962096605216186931*a+5416407653427664859))*r1+(1/44723883102069817219 56*(-32980255387477122 75*a+9121962151699219 97)*$

$i + 1/40251494791862835497604 * (-5828224036634168909 * a + 16428556309583548809))) * t^7 + ((1/40251494791862835497604 *$

$(48258912780223168*a-137033677318130397)*i+1/40251494791862835497604*(44011331449658765*a-279378079766194766))*$

$r1 + (1/40251494791862835497604 * (-6103412458394036241 * a + 16256670600286958450) * i + 1/40251494791862835497604 *$

$(-2685089790805721786 * a + 7084358370056268217))) * t^6 + ((1/161005979167451341990416 * (12913423133103850087 * a -$

$$3465161166809469952 3)*i+1/23000854166778763141488*(1725202006137560647*a-4761802339652663689))*r1+(1/9470939951026549528848*(777301094064947733*a-221162751422127577 3)*i+1/161005979167451341990416*(3463547845712580112 3*a-97560711179423862657)))*t^5+((1/80502989583725670995208*(-685341135520214680*a+1889814418869966001)*i+1/53668659722483780663472*(-230918402050263551*a+974635666182973235))*r1+(1/80502989583725670995208*(19992063931713224061*a-53167005662377091023)*i+1/161005979167451341990416*(17800670278932845599*a-47469547276876861157)))*t^4+((1/64402391666980536796166 4*(-215792915997021368 3*a+58070474435484769909)*i+1/64402391666980536796166 4*(-182879876119665950 37*a+50435052587635048445))*r1+(1/64402391666980536796166 4*(3602593737165311033*a-4523257084745413035)*i+1/64402391666980536796166 4*(-46141891588337301261*a+130069768643681386961)))*t^3+((1/64402391666980536796166 4*(6660777524916047175*a-18111043082487485396)*i+1/64402391666980536796166 4*(314302310201324730 9*a-10367588372094423274))*r1+(1/92003416667115052565952*(-104246198785418379 16*a+27696300630832638309)*i+1/64402391666980536796166 4*(-32283908567177680778*a+87925191244381131319)))*t^2+((1/25760956666792214718466 56*(819611731462977350 3*a-21936101845455721903)*i+1/25760956666792214718466 56*(4729086370597566331*a-12811500666713077329))*r1+(1/85869855555974049061555 2*(-10365384534228766985*a+26999856410250381525)*i+1/25760956666792214718466 56*(4877842228008506301*a-13129439189228456899)))*t+((1/51521913333584424936933 12*(-13419961957682462559*a+36030095515019888009)*i+1/73602733333692042052761 6*(-943509530710860337*a+2753827852345367769))*r1+(1/73602733333692042052761 6*(9962928804965238465*a-26553379369306596007)*i+1/17173971111194809812311 04*(10157647553349389273*a-29034274351818420769))))*b1+(((1/54621*(-76*a+140)*i+1/54621*(12*a-28))*r1+(1/54621*(-152*a+392)*i+1/54621*(48*a+56)))*t^7+((1/54621*(-3*a-7)*i+1/54621*(-25*a+77))*r1+(1/18207*(a-35)*i+1/54621*(153*a-413)))*t^6+((1/54621*(122*a-217)*i+1/54621*(-18*a+49))*r1+(1/54621*(188*a-539)*i+1/54621*(-72*a-133)))*t^5+((1/31212*(3*a+5)*i+1/218484*(193*a-595))*r1+(1/218484*(-33*a+665)*i+1/31212*(-155*a+433)))*t^4+((1/218484*(-208*a+343)*i+1/218484*(27*a-98))*r1+(1/72828*(-43*a+210)*i+1/218484*(108*a+385)))*t^3+((1/62424*(-3*a-1)*i+1/218484*(-116*a+357))*r1+(1/145656*(19*a-189)*i+1/109242*(293*a-854)))*t^2+((1/1747872*(159*a-203)*i+1/582624*(-5*a+35))*r1+(1/1747872*(-213*a-91)*i+1/249696*(-9*a-85)))*t+((1/582624*(4*a-7)*i+1/582624*(55*a-168))*r1+(1/102816*(-3*a+14)*i+1/1747872*(-730*a+2233)))))*b2^2+((((1/40251494791862835497604*(26668614955502333757*a-67460687263396561505)*i+1/40251494791862835497604*(-9781467804429808467*a+2990858376852847 401))*r1+(1/7892449959188791274 04*(-821126783952577199*a+2273240715981201575)*i+1/23677349877566373822 12*(-40837892201446089 77*a+11387113081875103503)))*t^7+((1/40251494791862835497604*(-20977637993648674015*a+53576953316674895814)*i+1/40251494791862835497604*(-1294789486801990511 0*a+35531332592569543415))*r1+(1/23677349877566373822 12*(57969235522221737 22*a-15860277085904774131)*i+1/23677349877566373822 12*(832911771316622155*a-1924553985850228544)))*t^6+((1/16100597916745134199041 6*(-184297711718893317925*a+468243968023893533311)*i+1/23000854166778763141488*(77269097988666673 13*a-24212808102084260609))*r1+(1/94709399510265495288 48*(20672135897191806465*a-56139453957315145487)*i+1/94709399510265495288 48*(25133226571996146421*a-70035603053968116873)))*t^5+((1/10062873697965708874401*(9932815368237247334*a-25693163503862509571)*i+1/53668659722483780663472*(29598191193116982 57*a-80334491546648266921))*r1+(1/47354699755132747644 24*(-21485603120563117365*a+58303269454142468714)*i+1/94709399510265495288 48*(-7793201801646646169*a+19472296155150301855)))*t^4+

((1/644023916669805367961664 * (370771590495714302549 * a − 950103652483970186461) * i + 1/644023916669805367961664 *
(−52159185916895900943 * a + 181228459145480740441)) * r1 + (1/378837598041061981153 92 * (−56147535445649437879 * a +
149077628707238221011) * i + 1/378837598041061981153 92 * (−38915677456985855847 * a + 107243491022275371869))) * t^3 +
((1/644023916669805367961664 * (−368687628088719543258 * a + 969062928835078184135) * i + 1/644023916669805367961664 *
(−174588099820872102360 * a + 46868677480636 4875789)) * r1 + (1/5411965686300885445056 * (13689330232157816399 * a −
36704317980666791478)*i+1/378837598041061981153 92*(23743038330148504837*a−62029181537750565992)))*t^2+((1/257609566667922147184 6656*
(−212541558500496948827 * a + 551162345517534750013) * i + 1/257609566667922147184 6656 * (−25292147183569401133 * a +
50005068649491275481))*r1+(1/505116797388082641538 56*(151847665605127923 53*a−39680132184911937915)*i+1/151535039216424792461568*
(116881649922371820 09 * a − 31319753064529686841))) * t + ((1/515219133335844294369 3312 * (495134177947760303481 * a −
131325808151447835 3065)*i+1/736027333336920420527616*(26549512686497407405*a−70445865326367419955))*r1+(1/432957254904070835 60448*
(−173176543940624723 79*a+46041866837208406835)*i+1/101023359477616528307712*(−13285809660310242585*a+35291833820492824463))))*
b1 + (((1/54621 * (340 * a − 1484) * i + 1/3213 * (−12 * a − 28)) * r1 + (1/3213 * (40 * a + 56) * i + 1/3213 * (−48 * a + 56))) * t^7 +
((1/3213 * (3 * a − 7) * i + 1/54621 * (−65 * a − 77)) * r1 + (1/1071 * (−a − 35) * i + 1/3213 * (−111 * a + 301))) * t^6 + ((1/54621 * (−548 *
a + 2611) * i + 1/3213 * (18 * a + 49)) * r1 + (1/3213 * (−34 * a − 203) * i + 1/3213 * (72 * a − 133))) * t^5 + ((1/1836 * (−3 * a + 5) * i +
1/218484 * (317 * a + 1015)) * r1 + (1/12852 * (33 * a + 665) * i + 1/1836 * (97 * a − 269))) * t^4 + ((1/218484 * (862 * a − 5089) * i +
1/12852 * (−27 * a − 98)) * r1 + (1/4284 * (−13 * a + 252) * i + 1/12852 * (−108 * a + 385))) * t^3 + ((1/3672 * (3 * a − 1) * i + 1/218484 *
(26 * a − 1071)) * r1 + (1/8568 * (−19 * a − 189) * i + 1/6426 * (−139 * a + 406))) * t^2 + ((1/1747872 * (−393 * a + 4907) * i + 1/34272 *
(5 * a + 35)) * r1 + (1/102816 * (171 * a − 1309) * i + 1/14688 * (9 * a − 85))) * t + ((1/34272 * (−4 * a − 7) * i + 1/582624 * (−123 * a +
742)) * r1 + (1/6048 * (3 * a + 14) * i + 1/102816 * (212 * a − 707))))) * b2 + (((1/1183867493878318691106 * (33043886295476661849 *
a − 88425032761705684261) * i + 1/1183867493878318691106 * (73216733661406961457 * a − 194184122150006571163)) * r1 +
(1/3946224979594395637 02*(64651790880738418829*a−17479368513992019263 3)*i+1/1183867493878318691106*(29234002164904149358 3*
a − 782195292827846440725))) * t^7 + ((1/2367734987756637382212 * (49526843796303232067 * a − 130110902573293038201) *
i + 1/2367734987756637382212 * (44421956170224168133 * a − 113504353829151151993)) * r1 + (1/2367734987756637382212 *
(26606413841192490129*a−80206998292782108251)*i+1/2367734987756637382212*(14334086510100088869 5*a−37843133722928788641 1)))*
t^6 + ((1/4735469975513274764424 * (−186674649688499237687 * a + 500937783543777401255) * i + 1/676495710787610680632 *
(−62552684782439812265 * a + 165700975223659928399)) * r1 + (1/4735469975513274764424 * (−119580152575081554317 7 * a +
324035892140106207529 3) * i + 1/4735469975513274764424 * (−175312069931034422678 9 * a + 468970464966312121719 9))) * t^5 +
((1/2367734987756637382212*(−83170139402601768403*a+21716009169009429509 2)*i+1/789244995918879127404*(−22550044224142729709*
a+57486085638833442654))*r1+(1/1183867493878318691106*(28199594020769360 64*a+25296545099131875 1)*i+1/2367734987756637382212*
(−20895283976995539507 7 * a + 556100593050648220631))) * t^4 + ((1/9470939951026549528848 * (104044872396228041378 * a −
28085351332693184842 9)*i+1/9470939951026549528848*(30167789724995623157 7*a−79686929753565656261 6))*r1+(1/9470939951026549528848*
(90074777150031878959 6*a−24514802111585200441 53)*i+1/9470939951026549528848*(12332376560766591156 57*a−33019774944964987237 10)))*
t^3 + ((1/3788375980410619811539 2 * (616903704204500614299 * a − 159656756040355056046 9) * i + 1/3788375980410619811539 2 *
(394605773031132993081 * a − 995449846265571941321)) * r1 + (1/5411965686300885445056 * (−108926902080010549733 * a +

$$27184873478486157877 5) * i + 1/37883759804106198115392 * (1018312081758447616475 * a - 2768689581905098041463))) * t^2 +$$
$$((1/75767519608212396230784*(38972295710221524281*a-10394036101441356941 5)*i+1/75767519608212396230784*(-84442712140089868997*$$
$$a + 215890830785472372537)) * r1 + (1/8418613289801377358976 * (-51479963857020237121 * a + 140871783894345144151) * i +$$
$$1/75767519608212396230784 * (-453786801244929492435 * a + 121571180864533107602 3))) * t + ((1/151535039216424792461568 *$$
$$(-306010806009230971533 * a + 788216557847512798445) * i + 1/21647862745203541780224 * (-10839671021518063243 * a +$$
$$256385992091776221 69))*r1+(1/21647862745203541780224*(1414566808589667628 47*a-363191954197234799275)*i+1/505116797388082641538 56*$$
$$(8341734540848507287*a+5447154245536553711))))*b1+((1/3213*(3508*a-9716)*i+1/3213*(-828*a+2212))*r1+(1/189*$$
$$(220*a-700)*i+1/189*(-180*a+140)))*t^7+((1/3213*(522*a-1022)*i+1/3213*(1102*a-2870))*r1+(1/63*(-30*a+$$
$$154)*i+1/189*(-222*a+742)))*t^6+((1/3213*(-5828*a+15946)*i+1/3213*(1242*a-3556))*r1+(1/189*(-208*a+$$
$$742)*i+1/189*(270*a-112)))*t^5+((1/918*(-279*a+599)*i+1/6426*(-4121*a+11291))*r1+(1/378*(369*a-1673)*$$
$$i+1/54*(119*a-397)))*t^4+((1/12852*(10747*a-28777)*i+1/12852*(-1863*a+6167))*r1+(1/252*(-61*a+63)*i+$$
$$1/756*(-405*a-203)))*t^3+((1/1836*(315*a-775)*i+1/12852*(4637*a-13503))*r1+(1/84*(-53*a+203)*i+1/756*$$
$$(-983*a+3227)))*t^2+((1/25704*(-2517*a+6503)*i+1/8568*(81*a-413))*r1+(1/1512*(285*a-721)*i+1/216*(9*a+$$
$$29)))*t+(1/2016*(-57*a+161)*i+1/34272*(-2045*a+6335))*r1+1/6048*(711*a-2303)*i+1/6048*(1355*a-4361),$$
$$w_1 7 := ((((1/40251494791862835497604*(-2390569005052745227*a+6274715912124044301)*i+1/40251494791862835497604*$$
$$(-3140356510548608341*a+8182805022494548537))*r1+(1/40251494791862835497604*(2544929372131190022 7*a-67878053605415835929)*$$
$$i+1/40251494791862835497604*(-6095408318187223927*a+14412281383608493903)))*t^7+((1/40251494791862835497604*$$
$$(5174616347507284263*a-13722555477385500193)*i+1/9583689236157817975 62*(2675388956621709*a-5224174754906894))*$$
$$r1+(1/20125747395931417748802*(-5592126147766100912*a+14733642377662638143)*i+1/57502135416946907853 72*$$
$$(-747704864310938143*a+2040229956970773211)))*t^6+((1/16100597916745134199041 6*(20589722611921746273*a-54214481831474310199)*$$
$$i+1/23000854166778763141488*(2838305488391356931*a-7417129235112744479))*r1+(1/16100597916745134199041 6*$$
$$(-17498758483977223675 3*a+466717831483708329791)*i+1/16100597916745134199041 6*(31815945074232327647*a-72554432115740794835)))*$$
$$t^5+((1/16100597916745134199041 6*(-364237395739322141 29*a+96541942747232286689)*i+1/80502989583725670995208*$$
$$(-743216710732381766*a+1719758914826751183))*r1+(1/16100597916745134199041 6*(89776208956530924375*a-23693684585398911 6143)*$$
$$i+1/80502989583725670995208*(17512882541262701580*a-47823297015739991669)))*t^4+((1/64402391666980536796166 4*$$
$$(-56961560213583650925*a+150519805180552211341)*i+1/21467463888993512265388 8*(-10923345520875531077*a+$$
$$28833442994936151971))*r1+(1/64402391666980536796166 4*(345334687402183256611*a-92097812880801875401 1)*i+$$
$$1/92003416667115052565952*(-3653360795098655089*a+665576163797005602 3)))*t^3+((1/64402391666980536796166 4*$$
$$(75369662159440958095*a-19961090078808645965 8)*i+1/64402391666980536796166 4*(607947743937184604 4*a-15478294797897129331))*$$
$$r1+(1/64402391666980536796166 4*(-226268897934111322586*a+598232236695523057425)*i+1/21467463888993512265388 8*$$
$$(-21337810960597229949*a+581074196954793158 96)))*t^2+((1/25760956666792214718466 56*(45995906681322685837*a-$$
$$121796693533416987645)*i+1/36801366666846021026380 8*(170785458182980854 7*a-466070170798597552 9))*r1+(1/25760956666792214718466 56*$$
$$(-185506319642325530073*a+495173417653824586957)*i+1/25760956666792214718466 56*(-17355961955320234181*a+$$
$$54779880690635077915)))*t+((1/51521913333584429436933 12*(-87844634076506120143*a+232405072641424794293)*i+$$

$$1/736027333336920420527616 * (-1816116539143925149 * a + 4828133107193657553)) * r1 + (1/5152191333358442943693312 *$$
$$(332282972236860299243*a-879613385494378101189)*i+1/5152191333358442943693312*(5763858792344644246 3*a-155627165414523147583))))*$$
$$b1+(((1/54621*(64*a-196)*i+1/54621*(-24*a-28))*r1+(1/7803*(28*a-56)*i+1/7803*(-12*a-8)))*t^7+((1/18207*(a-$$
$$7)*i+1/54621*(53*a-133))*r1+(1/18207*(-a+7)*i+1/18207*(-39*a+119)))*t^6+((1/54621*(-101*a+315)*i+1/18207*$$
$$(12*a+14))*r1+(1/54621*(-255*a+455)*i+1/18207*(40*a+28)))*t^5+((1/31212*(-3*a+19)*i+1/218484*(-373*a+$$
$$959))*r1+(1/218484*(33*a-217)*i+1/218484*(961*a-2891)))*t^4+((1/109242*(81*a-266)*i+1/72828*(-17*a-21))*r1+$$
$$(1/218484*(191*a-63)*i+1/12138*(-8*a-7)))*t^3+((1/62424*(3*a-17)*i+1/109242*(97*a-259))*r1+(1/436968*(-57*$$
$$a+329)*i+1/218484*(-620*a+1827)))*t^2+((1/1747872*(-117*a+427)*i+1/249696*(3*a+5))*r1+(1/1747872*(359*a-$$
$$1421)*i+1/1747872*(-3*a+119)))*t+((1/582624*(-4*a+21)*i+1/1747872*(-230*a+651))*r1+(1/102816*(3*a-14)*$$
$$i+1/1747872*(919*a-2646)))))*b2^2+((((1/40251494791862835497604*(16419990613303051759*a-4330992350788127982 3)*$$
$$i + 1/40251494791862835497604 * (39036517189970381107 * a - 107266424458446531313)) * r1 + (1/2367734987756637382212 *$$
$$(-247925677780431515*a-85322904020035913)*i+1/2367734987756637382212*(-5794393102262351899*a+14844217839616884397)))*$$
$$t^7 + ((1/40251494791862835497604 * (-7949743185508015389 * a + 23346951138992052943) * i + 1/9583689236157817975 62 *$$
$$(-625838116311685329*a+169259218516959729 7))*r1+(1/11838674938783186911 06*(174207538985713949 5*a-4893185444080982629)*$$
$$i + 1/338247855393805340316 * (1223658786509308601 * a - 3295886034896043137))) * t^6 + ((1/161005979167451341990416 *$$
$$(-106294528063042195245 * a + 279283396607103442813) * i + 1/230008541666778763141488 * (-39225985944970408445 * a +$$
$$107371247903725435271))*r1+(1/9470939951026549528848*(3697573137526200041*a-4748672926603082617)*i+1/9470939951026549528848*$$
$$(45161689581211108883 * a - 117129455085330799769))) * t^5 + ((1/161005979167451341990416 * (66163554460053438733 * a -$$
$$192973493098683209171)*i+1/805029895837256709952 08*(100296218797333336544*a-271477067867290334049))*r1+(1/9470939951026549528848*$$
$$(-2543809450276523176 5*a+71383200556498479103)*i+1/473546997551327476442 4*(-3077593373212985085 3*a+828720191620478345 44)))*$$
$$t^4 + ((1/64402391666980536796166 4 * (188293644903635050455 * a - 487763071828579352617) * i + 1/21467463888993512265388 8 *$$
$$(188680951967755323301 * a - 512793730407050827189)) * r1 + (1/378837598041061981153 92 * (-15617168515589965829 * a +$$
$$32342874558948948211)*i+1/5411965686300885445056*(-1576188638578339677 1*a+41580615253727777523)))*t^3+((1/64402391666980536796166 4*$$
$$(-174691091866334503147 * a + 502305141991660366744) * i + 1/64402391666980536796166 4 * (-471964389789235576284 * a +$$
$$127816093240306426994 5)) * r1 + (1/378837598041061981153 92 * (553965604042449265 96 * a - 155740791494158130967) * i +$$
$$1/126279199347020660384 64 * (441179916725265633 97 * a - 118731738122500538448))) * t^2 + ((1/25760956666792214718466 56 *$$
$$(-847213220398055099 71 * a + 209636787681486392745) * i + 1/36801366666846021026380 8 * (-464968104015725299 15 * a +$$
$$125045756777728369315))*r1+(1/15153503921642479246156 8*(16106802193492220211*a-396912074553294317 57)*i+1/15153503921642479246156 8*$$
$$(79121879086010585305 * a - 212497605428668169477))) * t + ((1/51521913333584429436933 12 * (270937412514173749249 * a -$$
$$763197369683562255785)*i+1/73602733333692042052761 6*(92560898387575866853*a-249917962165261701051))*r1+(1/30307007843284958492313 6*$$
$$(-69116402936180655637 * a + 195536499863446746753) * i + 1/30307007843284958492313 6 * (-163246324692661647239 * a +$$
$$439827851689933386365))))*b1+(((1/54621*(536*a-1988)*i+1/3213*(24*a-28))*r1+(1/459*(4*a-32)*i+1/459*(12*a-8)))*$$
$$t^7+((1/1071*(-a-7)*i+1/54621*(-635*a+1645))*r1+(1/1071*(a+7)*i+1/1071*(25*a-63)))*t^6+((1/54621*(-817*a+2961)*$$
$$i+1/1071*(-12*a+14))*r1+(1/3213*(-39*a+329)*i+1/1071*(-40*a+28)))*t^5+((1/1836*(3*a+19)*i+1/218484*(4255*a-$$

$$11123))*r1+(1/12852*(-33*a-217)*i+1/12852*(-611*a+1519)))*t^4+((1/109242*(639*a-2128)*i+1/4284*(17*a-21))*r1+(1/12852*(47*a-441)*i+1/714*(8*a-7)))*t^3+((1/3672*(-3*a-17)*i+1/109242*(-1033*a+2737))*r1+(1/25704*(57*a+329)*i+1/12852*(382*a-945)))*t^2+((1/1747872*(-825*a+1673)*i+1/14688*(-3*a+5))*r1+(1/102816*(-37*a+217)*i+1/102816*(3*a+119)))*t+((1/34272*(4*a+21)*i+1/1747872*(2300*a-5985))*r1+(1/6048*(-3*a-14)*i+1/102816*(-527*a+1344)))))*b2+(((1/1183867493878318691106*(23715906842776845527*a-6040488208409678459 1)*i+1/1183867493878318691106*(-40846349726057211121 * a + 107683110140227104511)) * r1 + (1/1183867493878318691106 * (-380961147325584773939 * a + 101794340482217661222 7) * i + 1/1183867493878318691106 * (314519136276657805397 * a - 815896341453286449251))) * t^7 + ((1/23677349877566373822 12*(-13978345098812757109 5*a+373204046344411604917)*i+1/375830950437561489 24*(202942777285630947 3*a - 5351750079406275181)) * r1 + (1/23677349877566373822 12 * (204378311956405861819 * a - 555865666042652701981) * i + 1/338247855393805340316*(-56904771520087183961*a+146995915227768170177)))*t^6+((1/473546997551327476442 4*(-231056182372870097829*a + 597258702027556072553) * i + 1/676495710787610680632 * (46687017854385570497 * a - 122862457095634699091)) * r1 + (1/473546997551327476442 4*(260031228882560166856 1*a-694575010473995368188 1)*i+1/473546997551327476442 4*(-221345864587191852401 5*a + 573999793668937291492 9))) * t^5 + ((1/473546997551327476442 4 * (477360769144174126094 * a - 127534119349589037927 1) * i + 1/473546997551327476442 4 * (-463200162530399946133 * a + 121976548550537425277 4)) * r1 + (1/473546997551327476442 4 * (-875492815353328163046 * a + 237520200395542783753 3) * i + 1/473546997551327476442 4 * (150933069482479463055 9 * a - 390592107763915059524 2))) * t^4 + ((1/947093995102654952884 8 * (362119114824784684887 * a - 948416374354187530750) * i + 1/350775553741724056624*(-15235578367368564826*a+39958181802015058449))*r1+(1/947093995102654952884 8*(-250574390538909056043 7*a + 668745263229091580441 0) * i + 1/135299142157522136126 4 * (328054953051039747994 * a - 851197867504469771463))) * t^3 + ((1/378837598041061981153 92 * (-188218877734362815829 7 * a + 503529737028406808757 1) * i + 1/378837598041061981153 92 * (200176127619042069671 7 * a - 525586786568586276852 5)) * r1 + (1/378837598041061981153 92 * (473030283591283384092 5 * a - 127867758353670895412 79) * i + 1/126279199347020660384 64 * (-236558652607230853352 3 * a + 614548355176027672582 7))) * t^2 + ((1/757675196082123962307 84 * (-642172092388719221339 * a + 169535080987626705264 9) * i + 1/108239313726017708901 12 * (86558820246891040453 * a - 226014644041152485089)) * r1 + (1/757675196082123962307 84 * (254241303459757192647 9 * a - 678335929164819882338 9) * i + 1/757675196082123962307 84 * (-271837584796107611416 7 * a + 706902658781744482806 7))) * t + ((1/151535039216424792461 568 * (102365632829986507894 1 * a - 274412726191105565893 9) * i + 1/216478627452035417802 24 * (-17585358199955223670 3 * a + 460070660528379149667)) * r1 + (1/151535039216424792461 568 * (-365161579024176947500 9 * a + 983406651478048019558 7) * i + 1/151535039216424792461 568 * (490098484171061351436 1 * a - 127901469150112780824 73)))) * b1 + ((1/3213*(-2596*a+7084)*i+1/3213*(1908*a-2324))*r1+(1/27*(-68*a+124)*i+1/27*(36*a-20)))*t^7+((1/1071*(-174*a+658)*i+1/3213*(-1382*a+3766))*r1+(1/63*(30*a-98)*i+1/63*(50*a-98)))*t^6+((1/3213*(4064*a-11340)*i+1/1071*(-996*a+1288))*r1+(1/189*(636*a-1036)*i+1/63*(-108*a+28)))*t^5+((1/918*(279*a-1007)*i+1/6426*(4913*a-13363))*r1+(1/378*(-369*a+1225)*i+1/378*(-625*a+1379)))*t^4+((1/12852*(-6453*a+19033)*i+1/476*(177*a-259))*r1+(1/756*(-571*a+315)*i+1/84*(25*a+49)))*t^3+((1/1836*(-315*a+1081)*i+1/12852*(-5147*a+13979))*r1+(1/756*(477*a-1589)*i+1/756*(841*a-2079)))*t^2+((1/25704*(1194*a-3997)*i+1/3672*(-117*a+232))*r1+(1/1512*(-148*a+679)*i+1/1512*(117*a-406)))*t+(1/672*(19*a-63)*i+1/102816*(5867*a-15981))*r1+1/6048*(-711*a+2303)*i+1/6048*(-1355*a+3633).$$

# Bibliography


[AN13] A. Abdollahi and H. Najafi. Frame graph. *Preprint*, 2013.

[All80] W. O. Alltop. Complex sequences with low periodic correlations. *IEEE Trans. Inform. Theory*, 26(3):350–354, 1980.

[App05] D. M. Appleby. Symmetric informationally complete-positive operator valued measures and the extended Clifford group. *J. Math. Phys.*, 46(5):052107, 29, 2005.

[App09] D. M. Appleby. Properties of the extended clifford group with applications to sic-povms and mubs. *arXiv:0909.5233 [quant-ph]*, 2009.

[ABB+12] D. M. Appleby, I. Bengtsson, S. Brierley, M. Grassl, D. Gross, and J. Larsson. The monomial representations of the Clifford group. *Quantum Inf. Comput.*, 12(5-6):404–431, 2012.

[AFF11] D. M. Appleby, S. T. Flammia, and C. A. Fuchs. The Lie algebraic significance of symmetric informationally complete measurements. *J. Math. Phys.*, 52(2):022202, 34, 2011.

[AYAZ13] D. M. Appleby, H. Yadsan-Appleby, and G. Zauner. Galois automorphisms of a symmetric measurement. *Quantum Inf. Comput.*, 13(7-8):672–720, 2013.

[AY02] H. Anai and K. Yokoyama. Radical representation of polynomial roots. *http://www.jssac.org/Editor/Suushiki/V09/No1/V9N1_108.pdf*, 2002. Accessed: 14-11-2014.

[BK73] L. Baggett and A. Kleppner. Multiplier representations of abelian groups. *J. Funct. Anal.*, 14(3):299–324, 1973.

[BZ04] S. Ball and M. Zieve. Symplectic spreads and permutation polynomials. In *Finite fields and applications*, volume 2948 of *Lecture Notes in Comput. Sci.*, pages 79–88. Springer, Berlin, 2004.

[BF03] J. J. Benedetto and M. Fickus. Finite normalized tight frames. *Adv. Comput. Math.*, 18(2-4):357–385, 2003.

[Ben07] I. Bengtsson. Three ways to look at mutually unbiased bases. In *Foundations of probability and physics—4*, volume 889 of *AIP Conf. Proc.*, pages 40–51. Amer. Inst. Phys., Melville, NY, 2007.

[Ben10] I. Bengtsson. From sics and mubs to eddington. In *J. Phys.: Conf. Ser.*, volume 254, page 012007. IOP Publishing, 2010.

[BC10] B. G. Bodmann and P. G. Casazza. The road to equal-norm Parseval frames. *J. Funct. Anal.*, 258(2):397–420, 2010.

[BP05] B. G. Bodmann and V. I. Paulsen. Frames, graphs and erasures. *Linear Algebra Appl.*, 404:118–146, 2005.

[Bla14] K. Blanchfield. Orbits of mutually unbiased bases. *J. Phys. A*, 47(13):135303, 15, 2014.

[BCP97] W. Bosma, J. Cannon, and C. Playoust. The Magma algebra system. I. The user language. *J. Symbolic Comput.*, 24(3-4):235–265, 1997. Computational algebra and number theory (London, 1993).

[BW07] L. Bos and S. Waldron. Some remarks on Heisenberg frames and sets of equiangular lines. *New Zealand J. Math.*, 36:113–137, 2007.

[Bro73] I. D. Brown. Representation of finitely generated nilpotent groups. *Pacific J. Math.*, 45:13–26, 1973.

[CCKS97] A. R. Calderbank, P. J. Cameron, W. M. Kantor, and J. J. Seidel. $Z_4$-Kerdock codes, orthogonal spreads, and extremal Euclidean line-sets. *Proc. London Math. Soc. (3)*, 75(2):436–480, 1997.

[CRSS97] A. R. Calderbank, E. M. Rains, P. W. Shor, and N. J. A. Sloane. Quantum error correction and orthogonal geometry. *Phys. Rev. Lett.*, 78(3):405–408, 1997.

[CK03] P. G. Casazza and J. Kovačević. Equal-norm tight frames with erasures. *Adv. Comput. Math.*, 18(2-4):387–430, 2003.

[CKE13] P. G. Casazza and G. Kutyniok (Eds.). *Finite Frames*. Springer-Verlag, New York, 2013.

[CW11] T. Y. Chien and S. Waldron. A classification of the harmonic frames up to unitary equivalence. *Appl. Comput. Harmon. Anal.*, 30(3):307–318, 2011.

[CW14a] T. Y. Chien and S. Waldron. A characterisation of projective unitary equivalence of finite frames. *arXiv:1312.5393 [math.FA]*, 2014.

[CW14b] T. Y. Chien and S. Waldron. The projective symmetry group of a finite frame. *Preprint*, 2014.

[Chr03] O. Christensen. *An introduction to frames and Riesz bases*. Applied and Numerical Harmonic Analysis. Birkhäuser Boston, Inc., Boston, MA, 2003.

[DGS77] P. Delsarte, J. M. Goethals, and J. J. Seidel. Spherical codes and designs. *Geometriae Dedicata*, 6(3):363–388, 1977.

[Die10] R. Diestel. *Graph theory*, volume 173 of *Graduate Texts in Mathematics*. Springer, Heidelberg, fourth edition, 2010.

[FMT12] M. Fickus, D. G. Mixon, and J. C. Tremain. Steiner equiangular tight frames. *Linear Algebra Appl.*, 436(5):1014–1027, 2012.

[Fra03] J. B. Fraleigh. *A first course in abstract algebra*. Addison-Wesley, fifth edition, 2003.

[Gal09] J. A. Gallian. *Contemporary abstract algebra*. Brooks/Cole, Belmont, CA, seventh edition, 2009.

[GR01] C. Godsil and G. Royle. *Algebraic graph theory*, volume 207 of *Graduate Texts in Mathematics*. Springer-Verlag, New York, 2001.

[GR09] C. Godsil and A. Roy. Equiangular lines, mutually unbiased bases, and spin models. *European J. Combin.*, 30(1):246–262, 2009.

[GKK01] V. K. Goyal, J. Kovačević, and J. A. Kelner. Quantized frame expansions with erasures. *Appl. Comput. Harmon. Anal.*, 10(3):203–233, 2001.

[GVT98] V. K. Goyal, M. Vetterli, and N. T. Thao. Quantized overcomplete expansions in $\mathbf{R}^N$: analysis, synthesis, and algorithms. *IEEE Trans. Inform. Theory*, 44(1):16–31, 1998.

[Gra05] M. Grassl. Tomography of quantum states in small dimensions. In *Proceedings of the Workshop on Discrete Tomography and its Applications*, volume 20 of *Electron. Notes Discrete Math.*, pages 151–164 (electronic). Elsevier, Amsterdam, 2005.

[Hal62] I. Halperin. On the Gram matrix. *Canad. Math. Bull.*, 5:265–280, 1962.

[Han07] D. Han. Classification of finite group-frames and super-frames. *Canad. Math. Bull.*, 50(1):85–96, 2007.

[Hig08] R. J. Higgs. Nice error bases and Sylow subgroups. *IEEE Trans. Inform. Theory*, 54(9):4199–4207, 2008.

[Hog98] S. G. Hoggar. $64$ lines from a quaternionic polytope. *Geom. Dedicata*, 69(3):287–289, 1998.

[HP04] R. B. Holmes and V. I. Paulsen. Optimal frames for erasures. *Linear Algebra Appl.*, 377:31–51, 2004.

[How05] R. Howe. Nice error bases, mutually unbiased bases, induced representations, the Heisenberg group and finite geometries. *Indag. Math. (N.S.)*, 16(3-4):553–583, 2005.

[Hug07] L. Hughston. $d = 3$ sic povms and elliptic curves. *http://pirsa.org/07100040/*, 2007. Accessed: 30-10-2013.

[Isa94] I. M. Isaacs. *Character theory of finite groups*. Dover Publications, Inc., New York, 1994. Corrected reprint of the 1976 original [Academic Press, New York].

[Iva81] I. D. Ivanović. Geometrical description of quantal state determination. *J. Phys. A*, 14(12):3241–3245, 1981.

[Jac85] N. Jacobson. *Basic algebra. I.* W. H. Freeman and Company, New York, second edition, 1985.

[JL01] G. James and M. Liebeck. *Representations and characters of groups*. Cambridge University Press, New York, second edition, 2001.

[Kan12] W. M. Kantor. MUBs inequivalence and affine planes. *J. Math. Phys.*, 53(3):032204, 9, 2012.

[Kha08] M. Khatirinejad. On Weyl-Heisenberg orbits of equiangular lines. *J. Algebraic Combin.*, 28(3):333–349, 2008.

[KR02] A. Klappenecker and M. Rötteler. Beyond stabilizer codes. I. Nice error bases. *IEEE Trans. Inform. Theory*, 48(8):2392–2395, 2002.

[Kni96b] E. Knill. Non-binary unitary error bases and quantum codes. *arXiv:quant-ph/9608048*, 1996.

[Kni96a] E. Knill. Group representations, error bases and quantum codes. *arXiv:quant-ph/9608049*, 1996.

[KT12] M. Korbelář and J. Tolar. Symmetries of finite Heisenberg groups for multipartite systems. *J. Phys. A*, 45(28):285305, 18, 2012.

[KP11] N. O. Kotelina and A. B. Pevnyi. The Venkov inequality with weights and weighted spherical half-designs. *J. Math. Sci. (N. Y.)*, 173(6):674–682, 2011. Problems in mathematical analysis. No. 55.

[KC07] J. Kovačević and A. Chebira. Life beyond bases: The advent of frames (part i). *Signal Processing Magazine, IEEE*, 24(4):86–104, 2007.

[LS73] P. W. H. Lemmens and J. J. Seidel. Equiangular lines. *J. Algebra*, 24:494–512, 1973.

[MACR01] N. Mukunda, Arvind, S. Chaturvedi, and Simon R. Bargmann invariants and off-diagonal geometric phases for multi-level quantum systems – a unitary group approach. *Phys. Rev. A*, 65(1):012102_1–012102_10, 2001.

[mag] The handbook of magma functions. https://magma.maths.usyd.edu.au/magma/documentation/. Accessed: 10-6-2014.

[Ozo08] M. Ozols. Clifford group. *http://home.lu.lv/ sd20008/papers/essays/Clifford group [paper].pdf*, 2008. Accessed: 14-11-2014.

[Ren04] J. M. Renes. *Frames, Designs, and Spherical Codes in Quantum Information Theory*. PhD thesis, University of New Mexico, 2004.

[RBKSC04] J. M. Renes, R. Blume-Kohout, A. J. Scott, and C. M. Caves. Symmetric informationally complete quantum measurements. *J. Math. Phys.*, 45(6):2171–2180, 2004.

[Rom06] S. Roman. *Field theory*, volume 158 of *Graduate Texts in Mathematics*. Springer, New York, second edition, 2006.

[SG10] A. J. Scott and M. Grassl. Symmetric informationally complete positive-operator-valued measures: a new computer study. *J. Math. Phys.*, 51(4):042203, 16, 2010.

[Sei01] J. J. Seidel. Definitions for spherical designs. *J. Statist. Plann. Inference*, 95(1-2):307–313, 2001. Special issue on design combinatorics: in honor of S. S. Shrikhande.

[ST92] J. H. Silverman and J. Tate. *Rational points on elliptic curves*. Undergraduate Texts in Mathematics. Springer-Verlag, New York, 1992.

[SH03] T. Strohmer and Robert W. Heath, Jr. Grassmannian frames with applications to coding and communication. *Appl. Comput. Harmon. Anal.*, 14(3):257–275, 2003.

[Stu08] B. Sturmfels. *Algorithms in invariant theory*. Texts and Monographs in Symbolic Computation. SpringerWienNewYork, Vienna, second edition, 2008.

[Tre08] J. C. Tremain. Concrete constructions of real equiangular line sets. *arXiv:0811.2779 [math.MG]*, 2008.

[Tre10] J. C. Tremain. Concrete constructions of equiangular line sets ii. *http://www.framerc.org/papers/ConcreteConstructionsofEquiangularLineSetsII.pdf*, 2010. Accessed: 14-11-2014.

[VW05] R. Vale and S. Waldron. Tight frames and their symmetries. *Constr. Approx.*, 21(1):83–112, 2005.

[VW08] R. Vale and S. Waldron. Tight frames generated by finite nonabelian groups. *Numer. Algorithms*, 48(1-3):11–27, 2008.

[VW10] R. Vale and S. Waldron. The symmetry group of a finite frame. *Linear Algebra Appl.*, 433(1):248–262, 2010.

[Wal03] S. Waldron. Generalized Welch bound equality sequences are tight frames. *IEEE Trans. Inform. Theory*, 49(9):2307–2309, 2003.

[Wal11] S. Waldron. Frames for vector spaces and affine spaces. *Linear Algebra Appl.*, 435(1):77–94, 2011.

[Wal15] S. Waldron. *An introduction to finite tight frames*. Springer-Verlag, New York, 2015. (In preparation.)

[WH06] S. Waldron and N. Hay. On computing all harmonic frames of $n$ vectors in $\mathbb{C}^d$. *Appl. Comput. Harmon. Anal.*, 21(2):168–181, 2006.

[WZ93] W. Watkins and J. Zeitlin. The minimal polynomial of $\cos(2\pi/n)$. *Amer. Math. Monthly*, 100(5):471–474, 1993.

[WF89] W. K. Wootters and B. D. Fields. Optimal state-determination by mutually unbiased measurements. *Ann. Physics*, 191(2):363–381, 1989.

[WS07] W. K. Wootters and D. M. Sussman. Discrete phase space and minimum-uncertainty states. *arXiv:0704.1277 [quant-ph]*, 2007.

[Zhu10] H. Zhu. SIC POVMs and Clifford groups in prime dimensions. *J. Phys. A*, 43(30):305305, 24, 2010.

[Zhu12] H. Zhu. *Quantum State Estimation and Symmetric Informationally Complete POMs*. PhD thesis, University of Vienna, 2012.