THE UNIVERSITY OF CHICAGO

HIGHER MOMENTS OF RANDOM DETERMINANTS AND PSEUDORANDOMNESS FOR STRUCTURED COMPUTATION

A DISSERTATION SUBMITTED TO
THE FACULTY OF THE DIVISION OF THE PHYSICAL SCIENCES
IN CANDIDACY FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY

DEPARTMENT OF COMPUTER SCIENCE

BY
ZELIN LV

CHICAGO, ILLINOIS
AUGUST 2026

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGMENTS

I feel deeply fortunate to have had Aaron Potechin and William Hoza as my advisors and professors.

Aaron, thank you for always being there for me, for your steadfast and unconditional support, and for your extraordinary patience. I have learned so much from the care and seriousness you bring to both research and teaching. I will always remember our meetings outdoors, and the sight of your notebook gleaming in the sunlight.

William, your exceptional ideas have been a constant source of inspiration. I find all of your research incredibly interesting, and every conversation with you leaves me with new ideas. Even before you joined the faculty here, I attended one of your talks in the department and immediately afterward told a friend how impressed I was by your presentation. Later, when I took your morning class, it may have been the first time in my life that I woke up early not because of an alarm, but because there was somewhere I genuinely wanted to be. I am nevertheless sorry for being late several times, since Chicago traffic was not always ideal.

I would also like to thank Janos Simon, not only for serving on my dissertation committee, but also for the wise advice he has given me over the years. I still remember visiting the department in January 2020, before I had even been admitted to the program, and already receiving many valuable insights from him.

I am grateful to the theoretical computer science community in the Chicago area, including Laci Babai, Sasha Razborov, Madhur Tulsiani, Avrim Blum, Lorenzo Orecchia, Neng Huang and many others, for the lectures, conversations, and inspiration that accompanied me throughout this journey.

I would also like to thank Dieter van Melkebeek, Jin-Yi Cai, and Shuchi Chawla for setting me on the path toward theoretical computer science. I was extremely fortunate to have opportunities to study the subject with such wonderful professors. I am also grateful to Yitong Yin at Nanjing University for giving me the opportunity to work on data-structure lower bounds during the summer of 2019. Although I did not produce any meaningful results, the experience was an important and wonderful one for me.

To all my friends: a Ph.D. is a long journey, and without you it would not have been possible. I treasure the time we spent playing basketball and doing so many other things together, and I will always remember our late nights at the now-closed restaurant Qibao. I feel very lucky to have played on two recreational basketball teams during my Ph.D. studies, and I am equally grateful for all the lunches we shared in the basement. To respect your privacy, I will leave your names unwritten here, but please know how much your friendship has meant to me.

Finally, I would like to thank my family. I am deeply grateful to my parents for their unconditional love and support. I wish my grandmother could be here to celebrate this moment with me; I am sure she would have been very happy to know that I completed my Ph.D. I also thank my partner for standing beside me and for all of her support, both emotionally and in our daily life.

To all of you, 谢谢.

ABSTRACT

This dissertation develops results in two research directions at the intersection of theoretical computer science, probability, and combinatorics. The first part studies moments of determinants of random matrices, with an emphasis on the combinatorial and analytic structures underlying their exact and asymptotic evaluation. The second part studies pseudorandomness for restricted models of computation, including branching programs and decision-tree-like models, with the goal of reducing the amount of randomness required to approximate or reproduce their behavior.

Although the two parts address different mathematical questions, they share a common methodological perspective. In both settings, an apparently complicated global quantity is understood by decomposing it into structured combinatorial components. For random determinants, this involves organizing the terms arising from determinant expansions and analyzing the resulting generating functions. For pseudorandomness, it involves identifying structural properties of computational models that allow truly random inputs to be replaced by distributions generated from short random seeds.

Let A be a random matrix. This dissertation studies exact formulas, generating functions, and asymptotic expansions for moments of its determinant. The central combinatorial object is a permutation table obtained by expanding several copies of the determinant simultaneously. The moment becomes a signed, weighted enumeration of these tables, and the dependence on the entry distribution is encoded by column types.

The first part develops an analytic-combinatorial method for this enumeration. Rather than count complete permutation tables one at a time, the method decomposes them into simpler components and uses exponential generating functions to count all ways of assigning labels and assembling those components. This gives a short derivation of the known second and fourth moments and a compact presentation of the sixth moment for mean-zero entries. For the centered sixth moment, the baseline consists of tables in which the six positions of

every column are grouped into three pairs of equal entries; the signed enumeration of these paired tables equals the sixth determinant moment for standard Gaussian entries. The paired part therefore provides a distribution-independent core. The remaining dependence on the third, fourth, and sixth entry moments is carried by explicit repeated-entry patterns that form isolated columns, closed cycles, or chains ending at pairs in the core.

The second part removes the mean-zero assumption. Write $A_n = B_n + m_1\mathbf{1}\mathbf{1}^\mathsf{T}$, where the entries of B_n are centered. Applying the matrix determinant lemma to each of the six determinant factors produces marked permutation tables in which each table row, one for each determinant factor, contains at most one mark; a mark records the choice of the scalar term m_1. Thus a table contains at most six marks, and the calculation treats separately every possible number of marks from zero through six. In a contributing column, each unmarked centered entry must occur at least twice, so the repeated entries form pairs, triples, or larger equality blocks. An inclusion–exclusion replacement separates the ordinary pairings from the corrections contributed by higher central moments.

A reduction then contracts arbitrarily long chains and leaves a finite shell containing the marks and exceptional columns, a compatible core containing only pairs, and closed components disjoint from the shell. The resulting shell formula combines local column weights, the complete centered sixth-moment series, symmetry factors that prevent duplicate counting, determinant signs, and a finite signed count of the smallest pair-only cores that complete the shell. To reuse calculations for the same row-support pattern, the implementation compares all $6! = 720$ relabelings of the six table rows and assigns row-equivalent patterns a common cache key. When a cached calculation is reused, a separately computed relative inversion parity determines whether it enters with the same or the opposite sign.

Finally, the same analytic-combinatorics viewpoint is applied to random symmetric, Wigner, and Hermitian matrices. In these ensembles, permutation tables are replaced by weighted multigraph structures that account for the dependence between transposed entries.

This yields explicit second-moment generating functions and places the classical symmetric result in a unified framework.

The second part of this dissertation studies pseudorandomness under strong structural restrictions. The first part concerns pseudorandom generators for space-bounded computation, with an emphasis on constant-width standard-order read-once branching programs. The second part concerns pseudorandom generators for adaptive local computation, most notably decision trees and related models.

The technical core of the dissertation consists of two chapters based on joint work with William M. Hoza. In the first, we investigate a new "XOR of INW" paradigm: starting from the Impagliazzo–Nisan–Wigderson generator, we analyze the bitwise XOR of several independent copies, prove that this construction fools constant-width branching programs, and establish matching limitations showing that this approach alone cannot break the classical $O(\log^2 n)$ seed-length barrier. In the second, we introduce the notion of k-wise probable uniformity, construct generators with seed length essentially $(1+\alpha)k$, and use them to fool near-maximal decision trees. Also, we build hitting sets for linear-algebraic and circuit classes.

Taken together, these results illustrate a common theme: useful pseudorandomness can be obtained by exploiting the exact structure of the target model rather than relying only on generic primitives such as small-bias spaces. The introduction and preliminaries are written to present both chapters as part of a single narrative about derandomization for restricted computation.

Although the two parts concern different mathematical objects, they share a common methodological perspective: complex global behavior is analyzed by identifying and exploiting underlying combinatorial structure.

CHAPTER 1

INTRODUCTION

This dissertation consists of two largely self-contained parts. Part I concerns the moments of determinants of random matrices, while Part II concerns pseudorandom generators and related derandomization questions for restricted models of computation. These two directions originated from separate research programs and use different technical tools. They are brought together in this dissertation as the two principal components of my doctoral research.

1.1 Part I: Higher Moments of Random Determinants

The determinant assigns a single number to a square matrix. Its absolute value measures how the corresponding linear transformation changes volume, while a zero determinant means that the transformation collapses some direction and the matrix is singular. If the entries of the matrix are random, then the determinant is also random. It may be very large, very small, positive, negative, or zero, depending on how the entries interact.

One way to understand this random quantity is through its moments. The second moment measures its typical squared size, while higher even moments place increasing emphasis on unusually large values. These moments are difficult to calculate because the determinant combines many products of matrix entries with alternating signs. Although the entries of the matrix may be independent, the products appearing in the determinant reuse them in overlapping ways. Raising the determinant to a power creates still more overlaps, together with extensive cancellation between positive and negative contributions.

This dissertation studies the moments

$$f_k(n) = \mathbb{E}\Big[\det(A_n)^k\Big] \tag{1.1}$$

of an $n \times n$ random matrix A_n. Here $\mathbb{E}$ denotes expectation, so $f_k(n)$ is the average value of

the kth power of the determinant. Most of the dissertation concerns matrices whose entries are independent and have the same probability distribution. The final part treats symmetric, Wigner, and Hermitian ensembles, where entries on opposite sides of the diagonal are related. The aim is not only to obtain formulas, but also to explain the combinatorial structures that make those formulas possible.

The Gaussian case provides a useful benchmark. If A_n has independent standard normal entries, then every even moment has the closed form

$$\mathbb{E}[\det(A_n)^{2r}] = \prod_{j=0}^{r-1} \frac{(n+2j)!}{(2j)!}. \tag{1.2}$$

Here $2r$ is the order of the moment. The product formula appears in early work of Forsythe and Tukey, Nyquist, Rice, and Riordan, and Prékopa [FT52, NRR54, Pré67]. It can be derived from special symmetries of the Gaussian distribution. For a general entry distribution, those symmetries are no longer available. The second moment is still simple, but the fourth moment already depends on more than the variance of an entry. At the sixth moment, contributions of both signs become essential. Allowing the entries to have nonzero mean adds another layer of interactions.

The first step is to turn the moment calculation into a counting problem. Each term in a determinant chooses one entry from every row and every column of the matrix; such a choice is encoded by a permutation. A term in the kth power of the determinant therefore consists of k permutations. Placing these permutations in k rows produces a *permutation table* with n columns.

The table shows where the same matrix entry is used more than once. If a label appears q times in one column, it contributes the qth moment $m_q = \mathbb{E}[X^q]$ of the common entry distribution. Since different matrix entries are independent, the weight of the whole table is the product of the weights of its columns. The table also records the positive or negative

sign of each permutation row. Consequently,

$$f_k(n) = \sum_{T \in \mathcal{T}_{k,n}} w(T)\,\mathrm{sgn}(T), \tag{1.3}$$

where $\mathcal{T}_{k,n}$ is the set of permutation tables, $w(T)$ is a product of entry moments, and $\mathrm{sgn}(T)$ is the product of the row signs. In this way, an expectation involving a complicated polynomial becomes a signed counting problem. The remaining task is to organize the tables so that their contributions can be evaluated without listing them one at a time.

1.1.1 From pairs to the shell–core structure

The structures that can occur in a permutation table become richer as the order of the moment increases. The lower moments show this progression.

For centered entries with variance one, the $k = 2$ calculation forces the two table rows to coincide, and the answer is $n!$. Under the same normalization, a nonzero $k = 4$ column contains either two pairs or one block of four equal entries. The 4-columns are isolated components, while the $2 + 2$ columns decompose into dependency cycles; every such table has positive determinant sign. This decomposition leads to the known generating function for the fourth moment.

For $k = 6$, even a centered entry distribution allows four kinds of nonzero column: one entry may occur six times; one may occur four times and another twice; two entries may each occur three times; or three entries may each occur twice. These are abbreviated by the multiplicity patterns 6, $4 + 2$, $3 + 3$, and $2 + 2 + 2$. Pairings of the six row positions provide the Gaussian benchmark, but the pairings themselves are combinatorial objects. For a general distribution, the difference from the paired contribution is recorded by *known* columns. Known 6-columns are isolated components; known 4-columns form closed cycles or paths ending at pairs in the remaining table; and $3+3$ columns form closed cycle components.

After removing the closed components, the remaining paired table is the universal core. Exponential generating functions then assemble the isolated columns, cycles, paths, and core.

When the entries have nonzero mean, write $A_{ij} = m_1 + B_{ij}$, where $\mathbb{E}[B_{ij}] = 0$, or in matrix form $A_n = B_n + m_1\mathbf{1}\mathbf{1}^{\mathsf{T}}$. The adjugate form of the matrix determinant lemma shows that this determinant is affine in m_1 [Bec23, Sec. 2.2]. Choosing its scalar term is recorded by a mark. Thus each row of the six-row table has at most one mark, so the total number of marks is between zero and six [BLP25, Definition 6 and Proposition 8]. Only finitely many column types can occur near a mark, although these columns can interact through paths of arbitrary length.

The central contribution of this dissertation is a *shell–core decomposition* that controls these interactions. A deterministic reduction gathers the marks and all exceptional behavior connected to them into a finite *shell*. What remains is a universal paired *core*, together with zero-mark components—isolated known 6-columns and closed cycles—that do not touch the shell. Thus the part of the calculation that depends on the pattern of marks and on the entry distribution is reduced to a finite enumeration, while the unbounded part is handled by universal generating functions.

The shell–core method has two complementary faces.

First, it is a structural proof technique. The contribution of a table separates into local factors from the shell, a universal factor from the paired core, and a factor from the components that float outside the shell. Symmetry factors prevent equivalent configurations from being counted repeatedly, while determinant signs retain the required cancellations.

Second, the decomposition gives a route to computing the sixth moment, but it does not by itself supply a practical implementation. For each possible shell, one must still determine its multiplicity and determinant sign and count the paired cores that can complete it. This is a finite problem, but it is too large for a naive program that generates every marked table and performs each remaining count independently. The implementation developed later

therefore recognizes repeated subcomputations and reuses their results, while retaining the shell-specific data needed to assemble the final contribution.

1.1.2 Contributions of the dissertation

This dissertation brings together our work on four related determinant-moment problems. Our work on i.i.d. entries with mean zero determines the sixth determinant moment: it gives an exact formal generating function, an explicit coefficient formula for every matrix size, and an asymptotic expansion to arbitrary fixed order [BLP23]. Our analytic-combinatorics work gives substantially shorter derivations of both the arbitrary-mean fourth moment and the mean-zero sixth moment [BLP25]. Our work on structured ensembles gives exact second-moment and characteristic-correlation generating functions for random symmetric, Wigner, and Hermitian matrices [BLP24].

Our work on the arbitrary-mean sixth moment develops marked tables, replacement rules, and the shell–core framework [BLP26]. The dissertation makes this project the center of the exposition and adds an extended account of its computational implementation.

1.1.2.1 A common analytic-combinatorics language

Our analytic-combinatorics work shows that two earlier determinant-moment calculations—the fourth moment for arbitrary i.i.d. entries and the sixth moment for mean-zero i.i.d. entries—can be recovered by decomposing permutation tables into labeled components and assembling those components with generating functions [BLP25]. For the fourth moment, this gives a direct component proof of the exact arbitrary-mean formula. For the centered sixth moment, it replaces much of the original recurrence calculation by a decomposition into isolated columns, closed cycles, paths, and a paired core.

Chapters 3 and 4 expand this method into a common language for the dissertation. Permutation tables are treated as labeled weighted structures, and the constructions SET,

SEQ, and CYC are interpreted directly on their components. Worked examples derive cycle-weighted derangements, paths ending at a core pair, marked atoms, and derivatives that select attachment sites. This component-first viewpoint exposes which generating-function factors are universal and which record the entry distribution.

1.1.2.2 The exact centered sixth moment

Our centered sixth-moment work proves that, after normalizing the entry variance to one, the exact answer depends on the entry distribution through its third, fourth, and sixth moments. The entries need not have a symmetric distribution: a nonzero third moment is allowed. In addition to the formal generating function, this work extracts a finite coefficient formula and an asymptotic expansion to arbitrary fixed order; in particular,

$$f_6(n) \sim \frac{e^{3m_4-9}}{48}(n!)^3 n^6 \qquad (n \to \infty)$$

for mean-zero, variance-one entries [BLP23, Corollary 1.11 and Theorem 1.13].

Chapter 6 presents these established results concisely because the longer recurrence proof was already developed in the author's Master's thesis. It follows the component proof of Beck, Potechin, and Lv [BLP25, Theorem 18], which separates the tables into isolated known 6-columns, closed cycles of known 4-columns, closed cycles formed from $3+3$ columns, and a paired core with paths of known 4-columns ending at its pairs. The paired-core series is exactly the standard-Gaussian sixth-moment series.

1.1.2.3 Marked tables for arbitrary entry distributions

The arbitrary-mean project begins with the exact decomposition

$$F_6(t) = \sum_{r=0}^{6} m_1^r T_6^{(r)}(t), \tag{1.4}$$

where $T_6^{(r)}(t)$ collects tables with exactly r marks. The matrix determinant identity makes each determinant factor affine in m_1, so each of the six table rows has at most one mark and all seven possible mark counts are included [BLP26].

The resulting closed form for the sixth-moment generating function is the following. Here

$$F_6(t) = \sum_{n=0}^{\infty} \frac{f_6(n)}{(n!)^2} t^n, \qquad N(t) = \frac{1}{48} \sum_{n=0}^{\infty} (n+1)(n+2)(n+4)!\, t^n,$$

and $\mu_j = \mathbb{E}[(X - m_1)^j]$ denotes the jth central moment of the entry distribution.

Theorem 1.1.1 (General sixth-moment formula)**.** *For any entry distribution Ω with $\mu_2 = 1$,*

$$F_6(t)\big|_{\mu_2=1} = \sum_{r=0}^{6} m_1^r \left(1 + (\mu_5 - 10\mu_3)\, m_1 t\right)^{6-r} Z_r(t). \tag{1.5}$$

Set

$$x = \frac{1}{1 - (\mu_4 - 3)t}, \qquad y = \frac{\mu_3^2}{1 + \mu_3^2 t}.$$

Then

$$\begin{aligned} Z_r(t) = \mu_3^{r \bmod 2} \left(1 + t\mu_3^2\right)^{10-(r \bmod 2)} & e^{t(\mu_6 - 15\mu_4 - 10\mu_3^2 + 30)} x^{21-2r} \\ & \times \sum_{s=0}^{2\lfloor r/2 \rfloor} y^s \left(p_{rs}(t) N(tx^3) + q_{rs}(t) N'(tx^3)\right), \end{aligned}$$

where

$$p_{00}(t) = \frac{1}{x^6}, \qquad q_{00}(t) = 0, \qquad p_{10}(t) = \frac{60t}{x^3}, \qquad q_{10}(t) = 12t^2,$$

and the remaining polynomials are as follows:

$$p_{20}(t) = \frac{15(x-1)}{x^2}, \qquad p_{30}(t) = 20t(21x^2 - 42x + 4),$$

$$p_{21}(t) = 630t^2, \qquad p_{31}(t) = 480t^2(7tx^3 + 3x - 5),$$

$$p_{22}(t) = -630t^3, \qquad p_{32}(t) = 480t^3(2 - 7tx^3),$$

$$q_{20}(t) = 3tx^2, \qquad q_{30}(t) = 12tx^2\big(7t(x-2)x^2 + 2\big),$$

$$q_{21}(t) = 6t^2(21tx^3 - 2x + 10), \qquad q_{31}(t) = 16t^2\big(42t^2x^6 + 18tx^4 - 11tx^3 - 6x + 10\big),$$

$$q_{22}(t) = -6t^3(21tx^3 + 8), \qquad q_{32}(t) = -32t^3(21t^2x^6 + 5tx^3 + 2).$$

$$p_{40}(t) = 15x^2\big((4t-6)x + 3\big),$$

$$q_{40}(t) = 3x^4\big(4t^2x^2 - 2t(3x+2)x + 1\big),$$

$$p_{41}(t) = 60tx^2\left\{\begin{array}{l} 84tx^3 - 147tx^2 \\ \quad + (44t+6)x - 18 \end{array}\right\},$$

$$q_{41}(t) = 12tx^2\left\{\begin{array}{l} 84t^2x^6 - 147t^2x^5 + t(44t+6)x^4 \\ \quad + 18tx^3 - 50tx^2 - 2x + 6 \end{array}\right\},$$

$$p_{42}(t) = 180t^2\left\{\begin{array}{l} 42t^2x^6 - 28tx^5 + 105tx^4 \\ \quad - 108tx^3 + 24x - 20 \end{array}\right\},$$

$$q_{42}(t) = 12t^2\left\{\begin{array}{l} 126t^3x^9 - 84t^2x^8 + 315t^2x^7 - 282t^2x^6 \\ \quad - 44tx^5 + 192tx^4 - 84tx^3 - 24x + 20 \end{array}\right\},$$

$$p_{43}(t) = -720t^3\left\{\begin{array}{l} 21t^2x^6 + 14tx^4 \\ \quad - 38tx^3 + 4x - 4 \end{array}\right\},$$

$$q_{43}(t) = -48t^3\left\{\begin{array}{l} 63t^3x^9 + 42t^2x^7 - 93t^2x^6 \\ \quad + 22tx^4 + 2tx^3 - 4x + 4 \end{array}\right\},$$

$$p_{44}(t) = 2520t^5x^3(3tx^3 - 4),$$

$$q_{44}(t) = 168t^5x^3(9t^2x^6 - 9tx^3 + 4).$$

$$p_{50}(t) = -60x^4\big(t(21x^2 - 43x + 4) + 3\big),$$

$$p_{51}(t) = 720tx^2\big(42t^2x^6 - 70t^2x^5 + 7t(4t+1)x^4 - 31tx^3 + 18tx^2 + 2x - 2\big),$$

$$p_{52}(t) = -720t^2\big(42t^2x^8 - 126t^2x^7 + 98t^2x^6 - 21tx^5 + 4tx^4 + 8tx^3 + 4x^2 - 8x + 4\big),$$

$$p_{53}(t) = -20160t^4x^3(2tx^4 - 4tx^3 + x - 1),$$

$$p_{54}(t) = -30240t^6x^6,$$

$$q_{50}(t) = -12x^4\big(t^2(21x^3 - 43x^2 - 8x - 4)x^2 + t(x^2 + 10x + 4)x - 1\big),$$

$$q_{51}(t) = 48tx^2\left\{\begin{array}{l} 126t^3x^9 - 210t^3x^8 + 21t^2(4t+1)x^7 - 51t^2x^6 \\ \quad - 35t^2x^5 + t(20t+11)x^4 + 13tx^3 - 18tx^2 - 2x + 2 \end{array}\right\},$$

$$q_{52}(t) = -48t^2\left\{\begin{array}{l} 126t^3x^{11} - 378t^3x^{10} + 294t^3x^9 - 21t^2x^8 - 162t^2x^7 \\ \quad + 94t^2x^6 + 57tx^5 - 76tx^4 + 28tx^3 - 4x^2 + 8x - 4 \end{array}\right\},$$

$$q_{53}(t) = -1344t^4x^3\big(6t^2x^7 - 12t^2x^6 + 7tx^4 - 5tx^3 - x + 1\big),$$

$$q_{54}(t) = -2016t^6x^6(3tx^3 - 1).$$

$$p_{60}(t) = \frac{x^6}{t}\big(4t^2(105x^2 - 255x + 64) + 15t(15x - 1) - 15\big),$$

$$p_{61}(t) = -10x^4\big(504t^2x^5 - 777t^2x^4 - 3t(64t - 81)x^3 + 4t(46t - 27)x^2 - 18(4t+1)x + 18\big),$$

$$p_{62}(t) = 30tx^2\left\{\begin{array}{l} 2520t^3x^9 - 4284t^3x^8 + 56t^2(32t+15)x^7 - 2919t^2x^6 \\ \quad + 3t(648t+35)x^5 - 8t(28t-27)x^4 - 504tx^3 \\ \quad + 24(9t-1)x^2 + 48x - 24 \end{array}\right\},$$

$$p_{63}(t) = -240t^2\left\{\begin{array}{l} 630t^3x^{11} - 1449t^3x^{10} + 7t^2(109t+12)x^9 - 525t^2x^8 \\ \quad + 408t^2x^7 + (59 - 48t)tx^6 - 93tx^5 + 36tx^4 \\ \quad - 2(t+2)x^3 + 12x^2 - 12x + 4 \end{array}\right\},$$

$$p_{64}(t) = 840t^4x^3\big(90t^2x^8 - 369t^2x^7 + 274t^2x^6 - 51tx^5 + 12tx^4 + 20tx^3 + 12x^2 - 24x + 12\big),$$

$$p_{65}(t) = 5040t^6x^6(18tx^4 - 25tx^3 + 6x - 6),$$

$$p_{66}(t) = 25200t^8x^9,$$

$$q_{60}(t) = x^6\big(12t^2(7x-17)x^4 + t(45x^2+44x+36)x^2 + t^{-1} - (8x^2+3x+12)x\big),$$

$$q_{61}(t) = 2x^4\left\{\begin{array}{l} -504t^3x^8 + 777t^3x^7 + 3t^2(64t-81)x^6 + 2t^2(18t-155)x^5 \\ \quad + 6t(37t+9)x^4 + 3t(8t+9)x^3 + 12t(2t-3)x^2 \\ \quad - 6(4t+1)x + 6 \end{array}\right\},$$

$$q_{62}(t) = 2tx^2\left\{\begin{array}{l} 7560t^4x^{12} - 12852t^4x^{11} + 168t^3(32t+15)x^{10} - 6237t^3x^9 \\ \quad + 9t^2(172t+35)x^8 + 4t^2(71t+564)x^7 - 1968t^2x^6 \\ \quad - 3t(72t+107)x^5 + 8t(34t+27)x^4 + 288tx^3 \\ \quad - 24(9t-1)x^2 - 48x + 24 \end{array}\right\},$$

$$q_{63}(t) = -16t^2\left\{\begin{array}{l} 1890t^4x^{14} - 4347t^4x^{13} + 21t^3(109t+12)x^{12} - 945t^3x^{11} \\ \quad - 225t^3x^{10} + t^2(389t+351)x^9 - 24t^2x^8 - 372t^2x^7 \\ \quad + t(126t-95)x^6 + 201tx^5 - 144tx^4 + (38t+4)x^3 \\ \quad - 12x^2 + 12x - 4 \end{array}\right\},$$

$$q_{64}(t) = 56t^4x^3\left\{\begin{array}{l} 270t^3x^{11} - 1107t^3x^{10} + 822t^3x^9 - 63t^2x^8 - 333t^2x^7 \\ \quad + 242t^2x^6 + 159tx^5 - 228tx^4 + 88tx^3 - 12x^2 + 24x - 12 \end{array}\right\},$$

$$q_{65}(t) = 336t^6x^6\big(54t^2x^7 - 75t^2x^6 + 36tx^4 - 29tx^3 - 6x + 6\big),$$

$$q_{66}(t) = 1680t^8x^9(3tx^3 - 1).$$

Chapter 7 develops the local combinatorics needed to evaluate these terms. It gives a systematic central-moment expansion, classifies the nonvanishing marked column types, and uses inclusion–exclusion to replace their repeated centered entries by known blocks, explicit triples, and anonymous pairs.

The chapter also makes explicit two bookkeeping rules needed for the nonzero-mean calculation. First, a mark represents choosing the constant term m_1 in the decomposition $X_{ij} = m_1 + B_{ij}$. It contributes one factor of m_1, but it is not a random matrix entry, so

it cannot be paired with another occurrence or included in an equality block. Pairings and blocks involve only the unmarked centered variables B_{ij}, and their repeated occurrences produce the central moments μ_j. Second, variance normalization is performed only after this separation. If $B_{ij} = \sqrt{\mu_2}\, C_{ij}$, then a sixth-moment table of order n with r marks contains $6n - r$ centered factors and therefore acquires the factor $\mu_2^{(6n-r)/2}$. Scaling before centralization would incorrectly assign variance factors to the scalar mean and mix the powers of m_1 with those of the central moments.

1.1.2.4 The shell–core theorem

Our general-mean work shows how a marked table can be reduced to a finite exceptional shell and a remaining pair-only core [BLP26]. Closed zero-mark components that do not meet the shell are handled by the centered generating functions. Consequently, once a shell is fixed, all of its unbounded extensions can be summed without enumerating larger matrices one size at a time.

Chapters 8 and 9 formulate this decomposition as a theorem about generating functions. For each shell S, its contribution factors into

$$\operatorname{sgn}(S)\frac{6!}{|\operatorname{Aut}(S)|}\cdot\frac{\mathcal{O}(t)}{\mathcal{N}(x(t))}\cdot\prod_{c\in S}G(c)\cdot\frac{x(t)^{k_S}\mathcal{N}^{(k_S)}(x(t))W_{\mathrm{core}}(S)}{\prod_{j=1}^{k_S}j(j+2)(j+4)}. \tag{1.6}$$

Here $\operatorname{Aut}(S)$ is the stabilizer under row, column, and element relabelings that preserve every mark, block, pairing, and distinguished root in the shell. The factor $\operatorname{sgn}(S)$ is the relative parity comparing the chosen shell representative with its canonical comparison table. Further, $G(c)$ is a local column factor, k_S is the number of shell elements minus the number of shell columns, and $W_{\mathrm{core}}(S)$ is the net signed number of minimal paired cores compatible with the shell. The series $\mathcal{N}$ enumerates paired cores and equals the standard-Gaussian sixth-moment

series, while $\mathcal{O}$ is the complete zero-mark series. The substitution

$$x(t) = \frac{t}{(1-(\mu_4-3)t)^3}$$

accounts for chains of known 4-columns attached to any of the three pairs in a core column.

Equation (1.6) is the organizing structural result of the dissertation.

1.1.2.5 The implementation pipeline and reuse of computations

Chapter 10 gives a substantially extended account of the calculation introduced in our general-mean work. Its input is a supplied feasible mark-position table P, a six-row record of the mark locations and the equality pattern of the entry labels associated with them. The program records the symmetry group $\mathrm{Aut}(P)$, places the allowed triples and blocks of four equal centered entries associated with those labels, and retains the branches on which some copies remain unresolved. It then pairs the remaining unmarked positions in the shell columns and records each local column factor. The result is a paired partial shell: its columns and local weights are known, but some exposed occurrences have not yet been decided to represent the same matrix entry.

Each such partial shell inherits the literal enumeration factor

$$\frac{6!}{|\operatorname{Aut}(P)|\, n_{\underline{3}}!},$$

where $n_{\underline{3}}$ is the number of separately generated triple columns. Separately generated four-block columns are emitted in a prescribed order and therefore introduce no additional factorial divisor. Chapter 10 derives this factor by orbit–stabilizer and explains why it is equivalent to the theorem-level factor $6!/|\operatorname{Aut}(S)|$ in (1.6); the implementation does not need to compute the automorphism group of every final shell.

The next stage emits the allowed set-partition choices for identifying the remaining

exposed occurrences, including choices that make no identification or only some of the possible identifications. After a choice is fixed, each surviving anonymous occurrence receives a fresh label. The program then computes the branch multiplicity, determinant sign, shell excess k_S, and the net signed count $W_{\mathrm{core}}(S)$ of pair-only cores that complete the shell. These data are combined with the inherited symmetry factor, local column factors, and universal generating-function factors. The working symbolic contribution of each recorded case is kept separate before any sum over the possible numbers of marks.

Two stored lookups make this calculation feasible. The first is used while the shell is still partially specified. Different construction paths can leave the same pattern of fixed and unfinished occurrences across the six rows, and row relabelings give further descriptions of that same remaining problem. The program compares all $6! = 720$ row relabelings and chooses a standard support description. On the first occurrence it computes and stores the full sum over the subsequent identification branches and compatible cores; the local column factors and case-level symmetry factor remain outside this stored quantity. A later occurrence reuses the sum and applies one relative determinant sign to the whole stored result, rather than correcting each branch separately.

The second lookup is used after the identifications have been fixed. Different fully specified shells can leave the same residual requirement for a pair-only core. The program computes the corresponding integer $W_{\mathrm{core}}(S)$ once and reuses it whenever that residual requirement reappears. Thus the first lookup reuses an entire branch-and-core sum for a partial shell, whereas the second reuses the compatible-core count for one fully specified shell.

1.1.2.6 Exact results for structured ensembles

Our structured-ensemble work proves exact generating functions for the second determinant moment and, in the Wigner and Hermitian settings, for more general characteristic-correlation functions [BLP24, Theorems 9, 10, and 12]. The symmetric result allows arbitrary i.i.d.

entries on and above the diagonal. The Hermitian model has i.i.d. real diagonal entries, i.i.d. complex entries above the diagonal with real means, and conjugate entries below the diagonal; the Wigner formulas arise from the corresponding specialization.

Because entries in positions (i, j) and (j, i) are now related, ordinary permutation-table columns are no longer independent components. That work instead represents the two determinant permutations by a two-colored directed multigraph and decomposes it into fixed points, mussels, loops, necklaces, chains, and dumbbells. Labeled sets, cycles, and sequences then assemble their weights into the exact generating functions. Chapter 11 presents these results in the common notation of the dissertation and relates their component proof to the analytic-combinatorics method used for asymmetric matrices.

1.1.3 Why the generating functions are formal

A generating function packages the moments for all matrix sizes into a single formal series. For i.i.d. asymmetric matrices, we use the normalization

$$F_k(t) = \sum_{n \geq 0} \frac{f_k(n)}{(n!)^2} t^n. \tag{1.7}$$

For even k, forgetting the common order of the n table columns removes one factor of $n!$ from the ordered table sum. The remaining denominator $n!$ is the standard exponential-generating-function normalization for the labeled columns. Even after this normalization the sixth-moment coefficients grow factorially, so $F_6(t)$ generally has radius of convergence zero. The series is therefore used formally. Labeled combinatorial operations remain valid coefficientwise, and the calculus of factorially divergent series later extracts asymptotic expansions from the same formal identities.

For symmetric and Hermitian ensembles, the natural normalization is different: the underlying graph structures carry one set of labels rather than independent row and column

labels, and the generating functions use $n!$ in the denominator. The distinction is not cosmetic; it reflects the number of label sets in the combinatorial model.

1.1.4 Organization

The dissertation is organized as follows.

Chapter 2 places determinant moments among neighboring problems in random matrix theory and summarizes the exact results most relevant to this project. Chapter 3 fixes notation for moments, central moments, table signs, marked tables, and the various generating functions. Chapter 4 gives the expanded background on labeled analytic combinatorics.

Chapter 5 treats the second and fourth moments as warm-ups. Chapter 6 gives the concise analytic-combinatorics proof of the mean-zero sixth-moment formula and its asymptotics.

The work of general sixth moment begins in Chapter 7. Chapter 8 defines the reduction to shell, core, and floating components. Chapter 9 states the shell contribution formula, gives the current proof sketch, and records the conditions needed for its use. Chapter 10 gives a substantially extended account of the calculation introduced in the previous chapters.

Chapter 11 treats symmetric, Wigner, and Hermitian matrices. The appendices record the mark-pattern tables with their multiplicities and the local column-signature factors.

1.2 Part II: Pseudorandomness for Structured Computation

Randomness is a computational resource. In many settings, a randomized algorithm can be viewed as a deterministic computation that receives a long random string as advice. A pseudorandom generator (PRG) attempts to replace that long string by the output of a much shorter seed without substantially changing the behavior of the computation. This viewpoint lies at the center of derandomization, and it is especially fruitful when the target computations have strong structural restrictions.

This dissertation focuses on two such restrictions. The first is the restriction of *space*:

randomized log-space algorithms are naturally modeled by read-once branching programs, and classical work of Nisan and of Impagliazzo, Nisan, and Wigderson showed that space-bounded computation admits remarkably efficient PRGs [Nis92, INW94]. The second is the restriction of *adaptivity with bounded local complexity*: decision trees, subcube partitions, and related models only inspect a limited amount of information from the input, but they may do so adaptively. In both settings, the central challenge is to identify pseudorandom objects that are much weaker than full randomness and yet still strong enough for the target model.

A useful way to organize the modern theory is to ask which structural feature of the distinguisher is being exploited. For branching programs, one can use the sequential and low-memory nature of the computation. For decision trees, one can use the fact that any particular execution path reveals only a bounded amount of information. The present thesis adopts that viewpoint: although the two main chapters address different models, both are best understood as attempts to push beyond generic bounded-independence techniques by exploiting finer structure.

1.2.1 Two themes of Structured Pseudorandomness

1.2.1.1 Space-bounded pseudorandomness

The first theme is pseudorandomness for space-bounded computation. A standard-order read-once branching program (ROBP) reads the input from left to right, one bit at a time, while maintaining a state of bounded width. This model captures the behavior of a space-bounded randomized algorithm on a fixed input, viewed as a function of its random bits. For polynomial width, Nisan's generator yields seed length $O(\log n \cdot \log(wn/\varepsilon))$ for width-w, length-n programs [Nis92], and the INW generator gives a closely related recursive construction based on expander graphs [INW94]. For constant width, the known landscape is more delicate: width 2 is essentially optimal already by classical work of Saks and Zuckerman [SZ95], while width 3 admits substantially better constructions due to Meka, Reingold, and

Tal [MRT19]. The width-4 case, however, remains a major barrier.

Chapter 13 studies a new approach to this barrier. Instead of trying to analyze a single INW generator instantiated with ever-better expanders, we analyze the bitwise XOR of several INW outputs. This construction is inspired by hardness amplification and XOR lemmas, but the resulting questions are specific to branching programs and to the INW recursion. The chapter proves both positive and negative results: on the positive side, sums of INW generators do fool constant-width ROBPs; on the negative side, this paradigm still cannot circumvent the classical $\Omega(\log^2 n)$ barrier if one reasons only from spectral expansion.

1.2.1.2 Adaptive pseudorandomness and near-maximal decision trees

The second theme is pseudorandomness for adaptive observers that reveal only limited information about the input. Classical bounded-independence and small-bias spaces are powerful tools in this regime [NN93, AGHP92], but they lose strength when the observer is adaptive or when the relevant parameter is very close to the ambient dimension. This becomes particularly visible for near-maximal decision trees, whose size can be as large as $2^{0.9n}$ while still being far from arbitrary Boolean functions.

Chapter 14 introduces a new notion, *k-wise probable uniformity*, designed for exactly this regime. It is stronger than almost k-wise uniformity in the direction that matters for lower-bounding the acceptance probability of junta-like tests, and it leads to generators with seed length essentially $(1+\alpha)k$. From this, we obtain PRGs for near-maximal decision trees and subcube partitions, as well as hitting sets for systems of linear equations over $\mathbb{F}_2$ and applications to linear-size Boolean circuits via work of Chen and Kabanets [CK16].

1.2.2 Contributions

The dissertation is organized around two research contributions.

Chapter 13. The chapter proves that the bitwise XOR of t independent INW generators fools constant-width ROBPs with an error bound that remains meaningful even when each constituent generator uses relatively mild expanders. The same chapter also proves that, despite this amplification phenomenon, no amount of XORing can by itself produce a seed-length improvement below $\Theta(\log^2 n)$ if the analysis only uses spectral expansion. Finally, it compares "increase t" and "decrease λ" as competing ways of strengthening INW, and shows that the two paradigms are incomparable in general.

Chapter 14. The chapter introduces k-wise probable uniformity, constructs explicit generators achieving this property with near-optimal seed length, and derives pseudorandomness consequences for decision trees, subcube partitions, and related circuit models. It also develops a linear-algebraic hitting-set construction for systems of equations over $\mathbb{F}_2$ and proves limitations showing that standard k-wise γ-biased generators are inherently too weak to recover the new results in the most interesting parameter regimes.

1.2.3 Related work

The literature surrounding both chapters is extensive, so this dissertation only highlights the works most directly relevant to the narrative.

Space-bounded pseudorandomness. The classical baseline consists of Nisan's generator [Nis92] and the recursive generator of Impagliazzo, Nisan, and Wigderson [INW94]. We compare three increasingly weak pseudorandom objects. An ε-PRG approximates every program's acceptance probability; an ε-hitting-set generator (HSG) only hits programs whose uniform acceptance probability is at least ε; and a weighted PRG (WPRG) uses a signed weighted average of its outputs [BCG20]. The support of a WPRG is an HSG after a constant-factor adjustment to the error.

Table 1.1 concerns explicit, standard-order, binary ROBPs of length n. Let w denote the

width and let $L_\varepsilon = \log(1/\varepsilon)$. The notation $\widetilde{O}$ suppresses polylogarithmic factors in logarithms of the parameters. For compactness, write $S_{\mathrm{reg}} = O(\log n(\log w + \sqrt{L_\varepsilon} + \log\log n) + L_\varepsilon)$ and $S_{\mathrm{perm}} = O(\log n(\log\log n + \sqrt{L_\varepsilon}) + L_\varepsilon)$.

Table 1.1: Best known explicit seed-length bounds for standard-order binary ROBPs, as of 2026. The permutation row has unbounded width and one accepting vertex.

Model	HSG	PRG	WPRG
Width 3	$\widetilde{O}(\log(n/\varepsilon))$ [GMR$^+$12]	$\widetilde{O}(\log n\, L_\varepsilon)$ [MRT19]	$\widetilde{O}(\log n\sqrt{L_\varepsilon} + L_\varepsilon)$ [CHL$^+$23]
General constant width	$O(\log^2 n + L_\varepsilon)$ [HZ20]	$O(\log n(\log n + L_\varepsilon))$ [Nis92]	$O(\log^2 n + L_\varepsilon)$ [Hoz21, CCD$^+$26]
Regular, width w	$\min\{O(w\log n), S_{\mathrm{reg}}\}$ [BRRY14, BHPP22]	$O(\log n(\log w + \log\log n + L_\varepsilon))$ [BRRY14]	S_{reg} [CW26, CT25]
Permutation	$\min\{O(\varepsilon^{-4}L_\varepsilon\log n), S_{\mathrm{perm}}\}$ [BHPP22, CW26]	$O(\log n(\log\log n + L_\varepsilon))$ [HPV21]	S_{perm} [CW26]

The width-3 row exhibits the clearest separation among the three notions. For $\varepsilon = 1/\operatorname{poly}(n)$, the respective seed lengths are $\widetilde{O}(\log n)$ for HSGs, $\widetilde{O}(\log^2 n)$ for PRGs, and $\widetilde{O}(\log^{3/2} n)$ for WPRGs [GMR$^+$12, MRT19, CHL$^+$23]. For extremely small ε, one should take the minimum of the displayed width-3 bounds and the general constant-width bounds. The next width is already a major frontier: no explicit HSG, PRG, or WPRG with $o(\log^2 n)$ seed length is known even for width 4 in the standard low-error regimes [DH25].

Regularity and reversibility permit further improvements. For regular ROBPs, the classical PRG analysis of Braverman, Rao, Raz, and Yehudayoff saves a logarithmic factor when w and $1/\varepsilon$ are small [BRRY14], while recent error-reduction methods give the square-root dependence on L_ε appearing in S_{reg} [CHL$^+$23, CL24, CW26, CT25]. For permutation ROBPs, an unbounded-width PRG can be independent of the width when there is one accepting

vertex [HPV21], and the best WPRG improves the low-error dependence to S_{perm} [CW26]. The one-accept condition is essential: an unbounded-width permutation program with an unrestricted accepting set can encode an arbitrary Boolean function. With width w and an arbitrary accepting set, the same constructions apply after replacing ε by ε/w. Finally, the latest general WPRG error reduction preserves the binary bound $O(\log^2 n + L_\varepsilon)$ for polynomial width while making the dependence on the input alphabet optimal [CCD$^+$26]. For broader context, see the surveys of Hoza and of Hatami and Hoza [Hoz22, HH24].

The XOR operation has appeared in related forms of hardness amplification. Assadi and N proved an XOR lemma for streaming algorithms; see also related work of Lee, Pyne, and Vadhan [AN21, LPV23]. Their setting presents several instances sequentially to a streaming algorithm, whereas Chapter 13 studies the bitwise XOR of several generator outputs; consequently, those general XOR lemmas do not directly analyze the construction considered here. There is also a line of work proving limitations on sums of small-bias distributions against various tests [BV10a, LT09, MZ09, BDVY13, LV17]. The lower bound in Chapter 13 is close in spirit to those results and, more specifically, builds on the noisy-codeword construction of Lee and Viola [LV17], together with earlier lower-bound ideas for space-bounded generators [BV10b, HPV24].

Approximate independence. For adaptive local models, the natural baseline is bounded independence. Naor and Naor's small-bias spaces and the almost k-wise uniform generators of Alon, Goldreich, Håstad, and Peralta are foundational examples [NN93, AGHP92]. Several inequivalent notions of approximate k-wise uniformity have been studied: one may control total variation distance on every k coordinates, the bias of every small parity, or the pointwise error of every local marginal. Another line of work asks whether a distribution is close to some exactly k-wise uniform distribution [AGM03, AAK$^+$07, OZ18]. The k-wise probable uniformity introduced in Chapter 14 is deliberately one-sided: it allows a small probability of failure but, outside that failure event, provides the lower bound needed for rare local events.

In this sense it is also reminiscent of Huber's contamination model in robust statistics, except that here the uncontaminated reference distribution on every local view is uniform [Hub64].

Universal sets and pseudorandom restrictions. A closely related combinatorial object is a k-universal set: a subset of $\{0,1\}^n$ whose projection onto every k coordinates contains every binary pattern. Universal sets have been studied for more than half a century [KS73, CKMZ83, TW83, Alo86, SB88, BS88, ABN$^+$92, NN93, NSS95, Bsh14]. The best explicit construction of Naor, Schulman, and Srinivasan has size $2^{k+O(\log^2 k)} \log n$ [NSS95]. Probable uniformity is stronger than this support-only guarantee: the support of a k-wise probably uniform distribution is k-universal, but the distribution additionally assigns enough probability to every local pattern. The construction in Chapter 14 was inspired in part by the universal-set construction of Naor, Schulman, and Srinivasan and is also related to a biased variant of universal sets studied by Harel, Hoza, Vardi, Evron, Srebro, and Soudry [HHV$^+$24].

The use of pseudorandom partitions to break a set of relevant variables into small buckets is another recurring theme in PRG constructions. Related partitioning ideas appear in work of Lovett, Reingold, Trevisan, and Vadhan; Meka and Zuckerman; and Gopalan, Kane, and Meka [LRTV09, MZ13, GKM18]. Chapter 14 uses such a partition together with reusable pairwise-uniform hashing, tailored to the one-sided guarantee above. Bshouty obtained strong nontrivial guarantees for juntas by the method of conditional probabilities [Bsh16]; those constructions provide an important comparison point, although their running time is much higher in the large-k regime considered here.

Circuits, correlation bounds, and hitting sets. Chen and Kabanets proved the first correlation bounds for general unbounded-depth circuit models and related near-maximal decision trees to linear-size circuits over the U_2 and B_2 bases [CK16]. Golovnev, Kulikov, Smal, and Tamaki subsequently strengthened these correlation bounds [GKST18]. These connections transfer the decision-tree results of Chapter 14 to circuit pseudorandomness and hitting sets.

The distinction between a PRG, a black-box hitting set, and a SAT or counting algorithm is important here: hitting sets apply even when the circuit is accessible only through queries, a property that has separate consequences for space-bounded derandomization [CH22, PRZ23].

1.2.4 Organization

Chapter 12 collects the shared notation and background used in both technical chapters. Chapter 13 develops the theory of sums of INW pseudorandom generators for constant-width ROBPs. Chapter 14 studies k-wise probable uniformity and its applications to decision trees, subcube partitions, linear systems, and linear-size circuits. The appendix records a form of the Gilbert–Varshamov bound for binary linear codes that is used in the branching-program chapter.

Part I

Higher Moments of Random Determinants

CHAPTER 2

BACKGROUND AND RELATED WORK

This chapter surveys the results most directly connected to the dissertation. The emphasis is on the point at which each result enters the permutation-table program rather than on a comprehensive history of random matrix theory.

2.1 Moments of determinants with independent entries

Let $A_n = (X_{ij})_{i,j\in[n]}$ have independent and identically distributed entries, and let

$$m_r = \mathbb{E}[X_{11}^r].$$

The earliest exact moment results concern $k = 2$. For centered entries, only terms in which the two determinant permutations agree survive, and therefore

$$\mathbb{E}[\det(A_n)^2] = n!\, m_2^n.$$

In the variance-one normalization this is $n!$. The result appears in early work of Fortet and Turán [For51, Tur55].

The fourth moment was determined by Nyquist, Rice, and Riordan [NRR54]. In the centered, variance-one case they showed that

$$f_4(n) = n! y_n, \qquad y_n = (n + m_4 - 1)y_{n-1} + (3 - m_4)(n - 1)y_{n-2}, \tag{2.1}$$

with $y_0 = 1$ and $y_1 = m_4$. The exponential generating function of y_n is

$$\sum_{n\geq 0} y_n \frac{t^n}{n!} = (1 - t)^{-3} e^{(m_4-3)t}. \tag{2.2}$$

The recurrence is already a table recurrence: one exposes the locations of the largest label and distinguishes whether it belongs to a 4-column or to two 2-columns. Beck later derived the fourth moment of the determinant of a square matrix for arbitrary i.i.d. entry distributions and, separately, the second moment of the Gram determinant associated with a rectangular matrix [Bec23]. The analytic-combinatorics rederivation in Beck, Potechin, and Lv [BLP25] is particularly close to the approach used here.

Beck, Potechin, and Lv determined the sixth moment for i.i.d. real entries with mean zero, without assuming a symmetric entry distribution; in particular, the third moment may be nonzero [BLP23]. They also gave a full asymptotic expansion. The original proof uses a detailed recurrence for paired permutation tables. The later paper Beck, Potechin, and Lv [BLP25] replaces much of that recurrence by a decomposition into labeled sets, cycles, and a paired core. This dissertation uses the latter presentation for the established mean-zero result and develops the marked-table and shell–core framework for the arbitrary-mean problem. The implementation organizes that calculation by the number of marks in the table and separates symbolic output from case-by-case verification.

2.2 The Gaussian benchmark

When the entries are standard real Gaussians, every even entry moment is a sum over pairings. This is precisely the situation in which permutation tables become pairing tables. Forsythe and Tukey, Nyquist–Rice–Riordan, and Prékopa obtained the product formula

$$\mathbb{E}[\det(A_n)^{2r}] = \prod_{j=0}^{r-1} \frac{(n+2j)!}{(2j)!} \tag{2.3}$$

for $r \geq 1$ [FT52, NRR54, Pré67]. Odd moments vanish by symmetry.

For the sixth moment, (2.3) gives

$$P_n = \frac{n!(n+2)!(n+4)!}{48}, \qquad P_n = n(n+2)(n+4)P_{n-1} \qquad (n \geq 1). \tag{2.4}$$

Here P_n is also the signed total of six-row permutation tables equipped with a valid pairing in each column. After the exceptional columns have been separated, the remaining paired subtable is counted by this same sequence. The associated normalized formal series is

$$\mathcal{N}(t) = \sum_{n\geq 0} \frac{P_n}{(n!)^2} t^n = \frac{1}{48} \sum_{n\geq 0} (n+1)(n+2)(n+4)!\, t^n. \tag{2.5}$$

Rather than regarding the Gaussian case only as a distribution to be substituted at the end, the centered sixth-moment calculation uses its paired table series as a combinatorial base. Deviations of the relevant central moments from their paired values are recorded by exceptional columns. The arbitrary-mean shell–core framework reuses this paired core.

2.3 Rectangular and Gram determinants

For a genuinely rectangular $p \times n$ matrix M with $p < n$, the determinant of M is not defined, but the $p \times p$ Gram determinant $\det(MM^{\mathsf{T}})$ is. Dembo determined its second moment for centered, variance-one entries [Dem89]. Beck extended this second-moment calculation to arbitrary i.i.d. entry distributions [Bec23].

Extending the sixth-moment shell–core method to rectangular Gram determinants is a possible direction for future work; no such formula is established here.

2.4 Symmetric, Wigner, and Hermitian ensembles

If $A = A^{\mathsf{T}}$, the entries above and below the diagonal are no longer independent. A pair of determinant permutations can use A_{ij} and A_{ji} in different columns, yet these are the same

random variable. Consequently, the asymmetric table factorization must be replaced by a graph factorization.

Zhurbenko determined the second moment of the determinant of a centered random symmetric matrix [Zhu68]. Götze and Kösters studied second-order correlations of characteristic polynomials for Hermitian Wigner matrices, while Mehta and Normand studied moments of determinants in the Gaussian unitary ensemble; these are related but distinct problems [GK09, MN98]. Beck, Potechin, and Lv gave a unified analytic-combinatorics derivation for symmetric, Wigner, and Hermitian matrices [BLP24]. Their symmetric model allows arbitrary i.i.d. entry distributions. In their Hermitian model, the diagonal entries are i.i.d. real variables, the entries above the diagonal are i.i.d. complex variables, the lower entries are their conjugates, and the relevant entry means are real. That work is integrated in Chapter 11.

The structured ensembles also explain why the analytic-combinatorics chapter is broader than the minimum needed for i.i.d. matrices. The cited analysis superimposes the two permutation graphs as a two-colored directed multigraph. Its component families are called fixed points, mussels, loops, necklaces, chains, and dumbbells; marked edges record mean contributions. Labeled sets, cycles, and sequences assemble these components.

2.5 Distributional results and singularity

Moment formulas are only one way to study random determinants. Another line of work concerns the distribution of $\log|\det(A_n)|$. Girko proved central limit theorems under suitable assumptions, and Nguyen and Vu later gave a streamlined and strengthened treatment [Gir80, NV14]. These results operate on a different scale from the exact fixed-order moments considered here: the logarithm is governed by the singular values collectively, whereas a fixed moment can be expanded into finitely many permutation patterns.

A third line of work studies the probability that a Bernoulli matrix is singular. The first

exponential upper bounds were developed by Kahn, Komlós, and Szemerédi, followed by a long sequence of improvements; Tikhomirov established the asymptotically sharp exponential rate [KKS95, TV06, Tik20]. For symmetric Bernoulli matrices, Costello, Tao, and Vu proved asymptotic nonsingularity [CTV06]. The survey of Vu gives a broad account of this combinatorial random-matrix literature [Vu20].

These singularity questions are related to determinant moments but are not reducible to them. A high even moment can be dominated by rare matrices with unusually large determinant, while singularity is a small-ball event. The table method is designed for exact polynomial expectations and therefore complements, rather than replaces, inverse Littlewood–Offord and singular-value techniques.

2.6 Analytic combinatorics and factorial divergence

The combinatorial framework follows Flajolet and Sedgewick [FS09]. The crucial extension for the sixth-moment asymptotics is that the relevant ordinary formal series are factorially divergent. Borinsky developed a differential calculus for such series, including product and composition rules for an asymptotic-derivation operator [Bor18]. This calculus allows one to start from a formal identity such as

$$F_6(t) = h(t)\mathcal{N}(g(t))$$

and extract the complete asymptotic expansion of the coefficients without first finding a convergent analytic continuation.

The dissertation uses analytic combinatorics in two distinct senses. In the exact enumeration, it is a symbolic language for labeled structures and is valid coefficientwise. In the asymptotic analysis, it is supplemented by the factorial-divergence calculus. Keeping these roles separate avoids a common confusion: the exponential generating functions in the

sixth-moment problem need not converge in order for the combinatorial identities or the asymptotic derivation to be meaningful.

2.7 Position of the present work

The results above reveal a sequence of increasing difficulty:

The first two stages are complete and admit compact formulas. The structured second-moment problem is also complete. The arbitrary-mean sixth moment is combinatorially much larger, but the shell–core reduction turns it into a finite computation, which is completed in this work.

CHAPTER 3

PRELIMINARIES

This chapter fixes notation used throughout the dissertation. The i.i.d. nonsymmetric model, called the asymmetric model in the cited papers, is the default. Here "asymmetric" means that no relation such as $X_{ij} = X_{ji}$ is imposed; it does not describe the entry distribution. Definitions specific to symmetric, Wigner, and Hermitian matrices are deferred to Chapter 11.

3.1 Random matrices and entry moments

Let Ω be a probability distribution on $\mathbb{R}$ with moments through order six. We write

$$A_n = (X_{ij})_{i,j\in[n]}, \qquad X_{ij} \overset{\text{i.i.d.}}{\sim} \Omega,$$

and use $\mathcal{M}_{n\times n}(\Omega)$ for the distribution of A_n. The raw moments are

$$m_r = \mathbb{E}[X_{11}^r]. \tag{3.1}$$

Set $B_{ij} = X_{ij} - m_1$. The central moments are

$$\mu_r = \mathbb{E}[B_{11}^r], \qquad \mu_1 = 0. \tag{3.2}$$

Most of our formulas use the normalization $\mu_2 = 1$. If $\mu_2 > 0$, normalize only after centralization by writing $B = \sqrt{\mu_2}\, C$. A sixth-moment table with n columns and r marks has $6n - r$ unmarked centered positions. Consequently, the general-variance formula is recovered from the normalized formula by the simultaneous substitutions

$$t \longmapsto \mu_2^3 t, \qquad m_1 \longmapsto \frac{m_1}{\sqrt{\mu_2}}, \qquad \mu_j \longmapsto \frac{\mu_j}{\mu_2^{j/2}} \quad (3 \le j \le 6). \tag{3.3}$$

Unless stated otherwise, the sixth-moment quantities below are written after this normalization, with $\mu_2 = 1$. If $\mu_2 = 0$, then $X_{11} = m_1$ almost surely, so

$$F_6(t) = 1 + m_1^6 t.$$

For every k for which $\mathbb{E}[|X_{11}|^k] < \infty$—in particular, for $0 \le k \le 6$ under the standing assumptions—define the determinant and permanent moments

$$f_k(n) = \mathbb{E}[\det(A_n)^k], \qquad p_k(n) = \mathbb{E}[\mathrm{per}(A_n)^k]. \tag{3.4}$$

For i.i.d. asymmetric matrices, the normalized formal generating function is

$$F_k(t) = \sum_{n \ge 0} \frac{f_k(n)}{(n!)^2} t^n. \tag{3.5}$$

The normalization reflects two label sets: row positions and entry labels. We use the word "exponential" for this normalization even when the coefficients remain factorially divergent.

For the sixth moment it is convenient to set

$$q_3 = \mu_3^2, \qquad q_4 = \mu_4 - 3, \qquad q_6 = \mu_6 - 10\mu_3^2 - 15\mu_4 + 30. \tag{3.6}$$

The constants 3 and 15 are the numbers of pairings of four and six labeled positions, respectively. We also define

$$D(t) = 1 - q_4 t, \qquad E(t) = 1 + q_3 t, \qquad x(t) = \frac{t}{D(t)^3}. \tag{3.7}$$

These quantities package the chain factors that recur in the sixth-moment decomposition. As

formal series at $t = 0$,

$$\frac{1}{D(t)} = \sum_{\ell \geq 0} (q_4 t)^{\ell}, \qquad \frac{1}{E(t)} = \sum_{\ell \geq 0} (-q_3 t)^{\ell}.$$

In the inverse shell reconstruction defined later, $1/D(t)$ records a possibly empty chain of known 4-columns attached to a pair, while $1/E(t)$ records the signed continuation associated with a $3+3$-column chain. A column of the paired core has three pairs, each of which may independently carry a known-4 chain; the substitution $x(t) = t/D(t)^3$ incorporates these three attachments into the core variable. Since $D(t)$ and $E(t)$ have constant term 1 and $x(t)$ has constant term 0, the reciprocals and subsequent compositions are well defined as formal power series.

3.2 Permutations and signs

Let S_n be the symmetric group on $[n] = \{1, \ldots, n\}$. The sign of $\pi \in S_n$ is

$$\operatorname{sgn}(\pi) = (-1)^{\operatorname{inv}(\pi)},$$

where $\operatorname{inv}(\pi)$ is the number of inversions. We regard a permutation as the row vector $(\pi(1), \ldots, \pi(n))$.

Swapping two entries within one permutation row reverses that row's sign. Reordering the k table rows merely reorders the factors in $\operatorname{sgn}(T)$. If k is even, a common permutation ρ of the columns, or a common relabeling ρ of $[n]$ in every row, multiplies the table sign by $\operatorname{sgn}(\rho)^k = 1$. These observations are used repeatedly in orbit reductions.

3.3 Permutation tables

Definition 3.3.1 (Permutation table)**.** A permutation k-table with n columns is a matrix

$$T = (T_{rj})_{r\in[k],\, j\in[n]}$$

whose rth row is a permutation $\pi_r \in S_n$. We denote the set of such tables by $\mathcal{T}_{k,n} = S_n^k$.

Definition 3.3.2 (Table sign)**.** The sign of a permutation table is

$$\operatorname{sgn}(T) = \prod_{r=1}^{k} \operatorname{sgn}(\pi_r). \tag{3.8}$$

For a column $c = (a_1, \ldots, a_k)^{\mathsf{T}}$, let $\nu_x(c)$ be the multiplicity of the label x. Its weight is

$$w(c) = \prod_{x\in[n]} m_{\nu_x(c)}, \tag{3.9}$$

where $m_0 = 1$. The table weight is the product of its column weights.

Proposition 3.3.3 (Table expansion)**.** *For every $k, n \geq 0$ for which $\mathbb{E}[|X_{11}|^k] < \infty$,*

$$f_k(n) = \sum_{T\in\mathcal{T}_{k,n}} w(T)\operatorname{sgn}(T), \qquad p_k(n) = \sum_{T\in\mathcal{T}_{k,n}} w(T). \tag{3.10}$$

Proof. Expand each determinant by the Leibniz formula:

$$\det(A_n)^k = \sum_{\pi_1,\ldots,\pi_k\in S_n} \left(\prod_{r=1}^{k} \operatorname{sgn}(\pi_r)\right) \prod_{j=1}^{n}\prod_{r=1}^{k} X_{j,\pi_r(j)}.$$

For a fixed k-tuple of permutations, independence across matrix entries factors the expectation by the row index j. The resulting factor is exactly the weight of column j. The permanent identity is the same without signs. □

When $m_1 = 0$, any column containing a label with multiplicity one has weight zero. For $k = 6$ and $\mu_2 = 1$, the only multiplicity partitions that can have nonzero weight are

$$6, \qquad 4+2, \qquad 3+3, \qquad 2+2+2.$$

We call them 6-, 4-, 3-, and 2-columns, respectively. Their weights are μ_6, μ_4, μ_3^2, and 1.

3.4 Forgetting column order

Analytic combinatorics is most convenient when the order of connected components is irrelevant. For even k, let $\widehat{\mathcal{T}}_{k,n}$ be the quotient of $\mathcal{T}_{k,n}$ by simultaneous permutations of the n columns. A simultaneous column permutation preserves both weight and sign. The action is free because every table row is a permutation, so every orbit has $n!$ representatives and

$$f_k(n) = n! \sum_{T \in \widehat{\mathcal{T}}_{k,n}} w(T)\,\mathrm{sgn}(T). \tag{3.11}$$

The remaining factor of $n!$ in (3.5) is the ordinary labeled-structure normalization.

We shall often suppress the hat and say that a class of tables has an exponential generating function when column order has been forgotten.

3.5 Pairings and the Gaussian benchmark

Definition 3.5.1 (Pairing)**.** A pairing of a sixth-moment column is a perfect matching of its six row positions such that matched positions carry equal labels. A pairing of a table is a choice of a valid pairing in every column.

A $2+2+2$ column has one valid pairing, a $4+2$ column has three, and a 6-column has fifteen. A $3+3$ column has no valid pairing. Consequently, if every column of a table T has

type 6, $4+2$, or $2+2+2$, and T contains a 6-columns and b $4+2$ columns, then

$$|\mathcal{P}(T)| = 15^a 3^b. \tag{3.12}$$

If T contains a $3+3$ column, then $\mathcal{P}(T) = \varnothing$.

For i.i.d. standard-normal entries, Wick's formula says that the expectation of a table monomial is $|\mathcal{P}(T)|$; both sides are zero when some column is unpairable. Therefore the signed enumeration of paired tables equals the sixth determinant moment of a standard Gaussian matrix:

$$P_n = \sum_{T \in \mathcal{T}_{6,n}} \operatorname{sgn}(T)|\mathcal{P}(T)| = \frac{n!(n+2)!(n+4)!}{48}. \tag{3.13}$$

This is the paired-table enumeration used in Beck, Potechin, and Lv [BLP23]. We write

$$\mathcal{N}(t) = \sum_{n \geq 0} \frac{P_n}{(n!)^2} t^n = \frac{1}{48} \sum_{n \geq 0} (n+1)(n+2)(n+4)!\, t^n. \tag{3.14}$$

This series is formal and factorially divergent.

3.6 Marked permutations and centralization

To treat nonzero mean, let $\mathbf{1}$ be the all-ones column vector and write

$$A_n = B_n + m_1 \mathbf{1}\mathbf{1}^{\mathsf{T}}.$$

The adjugate form of the rank-one determinant identity gives

$$\det(A_n) = \det(B_n) + m_1 \mathbf{1}^{\mathsf{T}} \operatorname{adj}(B_n)\mathbf{1}. \tag{3.15}$$

This polynomial identity does not require B_n to be invertible. It shows that one determinant is affine in m_1. Expanding $\det(B_n)$ and the cofactors in the second term produces permutations with no marked position and permutations with exactly one marked position. Equivalently, in the columnwise multilinear expansion, choosing the mean vector in two or more columns produces repeated columns and hence a zero determinant. This is the rank-one form of the marked-permutation expansion used in Beck, Potechin, and Lv [BLP25].

Definition 3.6.1 (Marked permutation)**.** A marked permutation of order n is obtained from a permutation $\pi \in S_n$ by replacing at most one of its entries by the mark $\times$. The sign of the marked permutation is defined to be $\operatorname{sgn}(\pi)$ before replacement. We denote the set of marked permutations by $S_n^{\times}$.

Definition 3.6.2 (Marked table)**.** A marked k-table is a table whose rows are marked permutations. We denote the set of marked k-tables with n columns by

$$\mathcal{T}_{k,n}^{\times} = (S_n^{\times})^k.$$

A marked position has weight m_1, while an unmarked label of multiplicity r in a column contributes the central moment μ_r. The table sign is the product of the underlying permutation signs.

Although a mark hides a label in the drawing, the underlying permutation determines that label uniquely. In a sixth-moment table, an element is *covered* if at least one of its six occurrences is replaced by a mark; otherwise it is *uncovered*.

Proposition 3.6.3 (Marked-table expansion)**.** *For every $k, n \geq 0$ for which $\mathbb{E}[|X_{11}|^k] < \infty$, the determinant moment is the signed weight sum of all marked permutation tables:*

$$f_k(n) = \sum_{T \in \mathcal{T}_{k,n}^{\times}} w_{\times}(T) \operatorname{sgn}(T). \tag{3.16}$$

Proof. Apply (3.15) independently to the k determinant copies and expand the determinant and cofactor terms. Taking expectations then factors the unmarked entries column by column into central moments, giving exactly the marked-table weights. □

If a table contains r marks, its weight contains m_1^r. For the sixth moment we therefore decompose

$$F_6(t) = \sum_{r=0}^{6} m_1^r T_6^{(r)}(t), \tag{3.17}$$

where $T_6^{(r)}$ depends only on the central moments. The zero-mark term is exactly the centered sixth-moment series.

3.7 Known blocks, triples, and anonymous pairs

The shell–core method replaces literal equality constraints by named repeated structures—known blocks and explicit triples—and anonymous pairs.

Definition 3.7.1 (Known block)**.** A known block records that four, five, or six indicated positions contain the same named element. A repeated triple is recorded explicitly as a triple rather than as a block.

Definition 3.7.2 (Anonymous pair)**.** An anonymous pair records only that two positions carry the same as-yet-unnamed element. Different pairs are initially anonymous and may later be identified if the row-support rules allow it.

For example, under the normalization $\mu_2 = 1$, an unmarked $4 + 2$ column of weight μ_4 is replaced by one known four-block of weight $\mu_4 - 3$ together with the three possible paired columns of weight 1. A 6-column is replaced by one known block of weight $\mu_6 - 15$ together with its fifteen pairings. A block-of-five convention exists as an alternative description of the exceptional one-mark column, but the production shell construction uses the pair-rooted

factor of Section 9.2.3 and does not place explicit covered five-blocks. The exact marked replacement rules are given in Chapter 7.

3.8 Row supports and the plus/minus encoding

Let

$$\mathcal{R}_2 = \{12, 13, 14, 15, 16, 23, 24, 25, 26, 34, 35, 36, 45, 46, 56\} \tag{3.18}$$

be the lexicographically ordered set of row pairs.

If an element occurs in exactly rows i, j, we represent it by a minus fragment a^{-}_{ij}. If it occurs in the complementary four rows, we represent it by a plus fragment a^{+}_{ij}. The superscripts describe support, not determinant sign. A plus and a minus with the same subscript have complementary supports and together form a completed six-row element.

The program packages the row-support fragments of a paired partial shell $\widetilde{S}$ into the four-block interface key

$$K_{\mathrm{par}}(\widetilde{S}) = \big(K_{\ell,-}, K_{u,-}, K_{\ell,+}, K_{u,+}\big), \tag{3.19}$$

where ℓ and u stand for labeled and unlabeled (anonymous), respectively. Every component stores only row-pair supports; the labeled components distinguish labeled fragments from anonymous ones but do not retain literal label values or their equality partition. Those labels remain available separately in the branch state while admissible identifications are selected. A partial-shell interface key may have nonempty anonymous components. Once an identification branch has been fixed, each surviving anonymous fragment receives a distinct fresh label; the resulting shell interface key has $K_{u,-} = K_{u,+} = \varnothing$. These support-only records are the dictionary keys defined precisely in Theorem 10.4.1. Thus the final all-labeled shell key is

$$K(S) = \big(K_{\ell,-}, \varnothing, K_{\ell,+}, \varnothing\big).$$

3.9 Shells, cores, and compatibility

The formal definitions of the reduction appear in Chapter 8; here we record the terminology.

A *shell* contains every reduced column that has a mark, a triple of a covered element, or a known block of four belonging to a covered element. A *core* contains only pairs. A core is compatible with a shell if their union contains exactly one copy of each element in every row. A *minimal compatible core* uses no element not already present in the shell.

If a shell has a columns and $a + k$ distinct elements, then a minimal compatible core has k columns. This is the origin of the derivative $\mathcal{N}^{(k)}$ in the shell formula.

3.10 Formal series conventions

All identities involving F_6, $\mathcal{N}$, and $\mathcal{O}$ are identities in a formal power-series ring over a polynomial or rational-function coefficient field. Differentiation, multiplication, and composition are interpreted coefficientwise. When a denominator such as $1 - q_4 t$ occurs, it is expanded at $t = 0$.

For later use, define the zero-mark sixth-moment series

$$\begin{aligned} \mathcal{O}(t) &= e^{(\mu_6 - 15)t} \frac{e^{-15 q_4 t}}{D(t)^{15}} E(t)^{10} e^{-10 q_3 t} \mathcal{N}(x(t)) \\ &= E(t)^{10} \frac{e^{q_6 t}}{D(t)^{15}} \mathcal{N}(x(t)). \end{aligned} \tag{3.20}$$

The two forms are equal by (3.6). The first displays the combinatorial components; the second is algebraically compact.

CHAPTER 4

ANALYTIC COMBINATORICS OF PERMUTATION TABLES

This chapter develops the analytic-combinatorics tools used throughout the dissertation. The presentation is self-contained at the level required for the determinant applications and follows the labeled framework of [FS09]. The essential principle is simple:

> A structural decomposition of labeled objects becomes an algebraic identity of exponential generating functions.

The value of the principle lies in choosing the correct objects and the correct notion of connected component. For permutation tables, connectedness is not geometric at first sight; it is induced by repeated labels, known-column dependencies, pairings, or graph edges. Once that dependency relation is made explicit, the standard constructions apply.

4.1 Weighted labeled structures

A weighted labeled combinatorial class $\mathcal{A}$ is a collection of finite objects such that every object α has:

- a finite label set $L(\alpha)$;
- a size $|\alpha| = |L(\alpha)|$; and
- a weight $w(\alpha)$ in a commutative ring, usually a polynomial or rational function in entry moments.

We assume that the total weight at each size is finite. Let

$$a_n = \sum_{\substack{\alpha \in \mathcal{A} \\ |\alpha| = n}} w(\alpha).$$

The exponential generating function is

$$A(t) = \sum_{n \geq 0} a_n \frac{t^n}{n!}. \tag{4.1}$$

Weights may be negative or complex. This is indispensable: determinant signs and the $3 + 3$ path factor introduce negative weights.

4.1.1 Relabeling and the star product

Suppose $\alpha \in \mathcal{A}$ has labels $[r]$ and $\beta \in \mathcal{B}$ has labels $[s]$. Their ordinary ordered pair does not yet define an object on $[r + s]$, because the two label sets overlap. A *standard relabeling* chooses an r-subset $R \subseteq [r + s]$, transports the relative order of $[r]$ to R, and transports the relative order of $[s]$ to the complement.

The labeled product $\mathcal{A} \star \mathcal{B}$ consists of all such relabeled pairs. For fixed sizes r, s, there are $\binom{r+s}{r}$ relabelings, so

$$\begin{aligned} [t^{r+s}]\, A(t)B(t) &= \frac{a_r}{r!}\frac{b_s}{s!}, \\ (r+s)![t^{r+s}]\, A(t)B(t) &= \binom{r+s}{r} a_r b_s. \end{aligned}$$

Thus

$$\mathcal{C} = \mathcal{A} \star \mathcal{B} \quad \Longleftrightarrow \quad C(t) = A(t)B(t). \tag{4.2}$$

Example 4.1.1 (A labeled product). Let $\mathcal{A}$ be a single red atom and $\mathcal{B}$ a single blue atom. Both have EGF t. The product has EGF t^2, so its total weight at size two is $2![t^2]t^2 = 2$. The two objects are obtained by assigning label 1 to red and 2 to blue, or the other way around.

This elementary example is exactly the mechanism by which columns from different table components are interleaved. The binomial or multinomial coefficients never need to be

inserted by hand; they are built into multiplication of EGFs.

4.2 Sequences, sets, and cycles

Three constructions appear repeatedly.

4.2.1 Sequences

A sequence of r objects of $\mathcal{A}$ has EGF $A(t)^r$. Allowing any length gives

$$\mathrm{SEQ}(\mathcal{A}) = \bigsqcup_{r\geq 0} \mathcal{A}^{\star r}, \qquad \mathrm{EGF}(\mathrm{SEQ}(\mathcal{A})) = \frac{1}{1-A(t)}. \tag{4.3}$$

If the sequence must be nonempty, the EGF is $A/(1-A)$.

In the determinant application, a directed path of known 4-columns is a nonempty sequence. If each path vertex contributes $q_4 t$, then the path factor is

$$\frac{q_4 t}{1-q_4 t}. \tag{4.4}$$

A chain attached to a pair may have length zero, giving $1/(1-q_4 t)$ instead.

4.2.2 Sets

A finite set of $\mathcal{A}$-objects has EGF

$$\mathrm{EGF}(\mathrm{SET}(\mathcal{A})) = e^{A(t)}. \tag{4.5}$$

The reason is that a set of r labeled components is a sequence modulo the $r!$ possible component orders. Hence its contribution is $A(t)^r/r!$, and summing over r gives the exponential.

Example 4.2.1 (Known 6-columns). A known 6-column contains one label, has weight $\mu_6 - 15$,

and contributes one column. Its EGF is $(\mu_6 - 15)t$. Such columns are disjoint components, so a set of them has EGF

$$\exp((\mu_6 - 15)t).$$

This factor appears unchanged in every centered sixth-moment formula.

4.2.3 Cycles

A directed cycle of r labeled $\mathcal{A}$-objects has $(r-1)!$ cyclic orderings, so its EGF contribution is $A(t)^r/r$. Therefore

$$\mathrm{EGF}(\mathrm{CYC}(\mathcal{A})) = \sum_{r\geq 1} \frac{A(t)^r}{r} = -\log(1 - A(t)). \tag{4.6}$$

Restricting to cycles of length at least two subtracts the $r = 1$ term:

$$\mathrm{EGF}(\mathrm{CYC}_{\geq 2}(\mathcal{A})) = -\log(1 - A(t)) - A(t). \tag{4.7}$$

A set of such cycles has EGF

$$\exp(-u\log(1 - A) - uA) = \frac{e^{-uA}}{(1 - A)^u}, \tag{4.8}$$

when every cycle receives an additional weight u.

4.3 A fundamental example: cycle-weighted derangements

Let D_n be the set of derangements of $[n]$, and let $c(\pi)$ be the number of cycles of π. A derangement is a set of cycles of length at least two. Giving each cycle weight u yields

$$\sum_{n\geq 0} \frac{t^n}{n!} \sum_{\pi\in D_n} u^{c(\pi)} = \frac{e^{-ut}}{(1 - t)^u}. \tag{4.9}$$

This identity appears in three forms in the dissertation.

First, in the fourth moment, each dependency cycle of 2-columns has three row arrangements, so $u = 3$. Second, cycles of known 4-columns in the sixth moment have fifteen row-pair choices, so $u = 15$ and the atom has weight $q_4 t$. Third, cycles of $3 + 3$ columns carry a signed multiplicity -10 and atom weight $-\mu_3^2 t$, which after simplification produces $E(t)^{10} e^{-10 q_3 t}$.

Example 4.3.1 (Checking small sizes). For $n = 2$, there is one derangement, a single 2-cycle, so the coefficient is u. Expanding the right-hand side,

$$\frac{e^{-ut}}{(1-t)^u} = 1 + \frac{u}{2} t^2 + O(t^3),$$

and multiplying by 2! gives u.

For $n = 3$, there are two 3-cycles, each with one cycle, so the total is $2u$. The coefficient of t^3 is $u/3$, and $3!(u/3) = 2u$.

4.4 Substitution and attached structures

Suppose each atom of an outer structure $\mathcal{A}$ is replaced by an inner nonempty structure $\mathcal{B}$. The resulting class is the labeled composition $\mathcal{A} \circ \mathcal{B}$, whose EGF is

$$A(B(t)). \tag{4.10}$$

The condition $B(0) = 0$ ensures that a finite outer object has only finitely many realizations at each total size.

A particularly important case is the replacement of each core column by itself together with three independent chains. A core column contributes t before expansion. Each of its three pairs may carry a possibly empty chain of known 4-columns, with factor $1/D(t)$. Thus

the decorated column contributes

$$t\left(\frac{1}{D(t)}\right)^3 = x(t). \tag{4.11}$$

Consequently, decorating every column of the Gaussian core replaces $\mathcal{N}(t)$ by $\mathcal{N}(x(t))$.

Example 4.4.1 (Why the exponent is three). A paired sixth-moment column contains exactly three pairs. The chains attached to distinct pairs are independent and use disjoint labels. Each pair gives $1 + q_4 t + (q_4 t)^2 + \cdots = 1/D(t)$. Multiplication of the three labeled chain classes gives $D(t)^{-3}$.

4.5 Pointing, derivatives, and prescribed atoms

Differentiation has a combinatorial meaning. If

$$A(t) = \sum_{n\geq 0} a_n \frac{t^n}{n!},$$

then

$$A'(t) = \sum_{n\geq 0} a_{n+1} \frac{t^n}{n!}.$$

The derivative counts an $\mathcal{A}$-object with one distinguished label removed from the size count. More generally, $A^{(k)}$ corresponds to k ordered distinguished labels.

In the shell–core formula, a shell with k more elements than columns requires a minimal compatible core with k prescribed columns. The Gaussian recurrence implies that the normalized contribution is

$$\frac{x^k \mathcal{N}^{(k)}(x)}{\prod_{j=1}^{k} j(j+2)(j+4)}. \tag{4.12}$$

The denominator removes the universal multiplicative factors that would otherwise be introduced while growing the prescribed core from size zero to size k.

Example 4.5.1 (One prescribed core column). Since $P_n = n(n+2)(n+4)P_{n-1}$, choosing a distinguished column in a Gaussian paired table of size n produces the factor n from its position and the additional factors $(n+2)(n+4)$ from the paired-row extension. Dividing by $1 \cdot 3 \cdot 5 = 15$ normalizes the base one-column core. Thus the one-column compatible-core series is

$$\frac{x\mathcal{N}'(x)}{15}.$$

If the shell has three possible pairings for its minimal core, this factor is multiplied by 3.

4.6 Signs as weights

Analytic combinatorics does not require positive weights. A component of odd permutation parity simply receives a negative weight. Two examples are especially important.

A path formed from $3+3$ columns changes sign at every added vertex: undoing one gluing step requires three transpositions. If one vertex contributes $\mu_3^2 t$, the signed nonempty sequence is

$$-\mu_3^2 t + (\mu_3^2 t)^2 - (\mu_3^2 t)^3 + \cdots = -\frac{\mu_3^2 t}{1+\mu_3^2 t}. \tag{4.13}$$

Similarly, the cycle multiplicity for pure $3+3$ columns is -10 at the structural level. Using (4.8) with the corresponding signed atom yields

$$(1+\mu_3^2 t)^{10} e^{-10\mu_3^2 t}. \tag{4.14}$$

The positivity of the final expression does not mean that every table contribution was positive; the cancellation has already been performed by the cycle construction.

4.7 Permutation tables as labeled structures

We now make the connection to the determinant normalization explicit. A table in $\widehat{\mathcal{T}}_{k,n}$ has an unordered set of n columns but still uses the labels $[n]$. Its total signed weight is

$$\widehat{f}_k(n) = \sum_{T \in \widehat{\mathcal{T}}_{k,n}} w(T)\,\mathrm{sgn}(T),$$

and $f_k(n) = n!\widehat{f}_k(n)$. Therefore

$$F_k(t) = \sum_{n \geq 0} \widehat{f}_k(n) \frac{t^n}{n!}. \tag{4.15}$$

This is an ordinary labeled EGF for the unordered-column table class.

4.7.1 Worked example: the second moment

Assume $m_1 = 0$ and $m_2 = 1$. Every nonzero 2-table has two identical rows. After forgetting column order, there is exactly one object at each size: the identity table

$$\begin{pmatrix} 1 & 2 & \cdots & n \\ 1 & 2 & \cdots & n \end{pmatrix}.$$

Its sign and weight are one, so $\widehat{f}_2(n) = 1$ and

$$F_2(t) = \sum_{n \geq 0} \frac{t^n}{n!} = e^t. \tag{4.16}$$

Restoring the forgotten column order gives $f_2(n) = n!$.

This example illustrates why the table EGF uses one factorial after quotienting by columns, even though the determinant moment was normalized by $(n!)^2$.

4.7.2 Worked example: the fourth-moment cycle class

In a centered fourth-moment table, each label appears either in one block of four or in two pairs. Remove the 4-columns temporarily. The remaining 2-columns decompose into dependency cycles: following the other occurrence of the label in the first row eventually returns to the starting label. For each cycle there are three row-pair patterns, and every pattern has positive sign. Hence the class of 2-column components is

$$\mathrm{SET}\big(3\,\mathrm{CYC}_{\geq 2}(\mathcal{Z})\big),$$

where $\mathcal{Z}$ is a labeled atom. Its EGF is

$$\frac{e^{-3t}}{(1-t)^3}. \tag{4.17}$$

A known 4-column carries the correction $m_4 - 3$ and forms an independent set, contributing $e^{(m_4-3)t}$. Combining and simplifying yields the fourth-moment generating function. This derivation is given formally in Chapter 5.

4.7.3 Worked example: a sixth-moment core with paths

Consider a Gaussian paired core with r columns. It contains $3r$ pairs. To each pair attach a chain of $\ell \geq 0$ known 4-columns. Across all pairs, this is equivalent to replacing each core atom t by $x(t) = t/D(t)^3$. The total contribution of all expanded cores is therefore

$$\mathcal{N}(x(t)).$$

No separate multinomial coefficient is needed to interleave the path columns with the core columns; labeled composition performs the relabeling.

4.8 Automorphisms and orbit factors

Analytic-combinatorics formulas count fully labeled objects. In the shell enumeration, however, we list one representative up to row, column, and element permutations. Let S be a shell with a columns and $a + k$ elements. The group

$$S_6 \times S_a \times S_{a+k}$$

acts by permuting rows, columns, and element names. Its orbit has size

$$\frac{6!\, a!\, (a+k)!}{|\operatorname{Aut}(S)|}. \tag{4.18}$$

After the label-placement factors from expanded shells and cores are combined, the surviving shell factor is $6!/|\operatorname{Aut}(S)|$, as in (1.6). This is a direct use of orbit–stabilizer, not an analytic-combinatorics construction, but it is essential for translating the finite shell list back to labeled tables.

4.9 Formal versus analytic use of generating functions

The exact identities in this dissertation take place in a formal power-series ring. For example,

$$\mathcal{N}(t) = \frac{1}{48} \sum_{n \geq 0} (n+1)(n+2)(n+4)! t^n$$

has radius of convergence zero. Nevertheless, multiplication and composition with a series $g(t) = t + O(t^2)$ are well-defined: the coefficient of t^n depends on only finitely many coefficients of the inputs.

The term "analytic combinatorics" therefore refers first to the structural calculus, not to singularity analysis of a convergent function. Asymptotics require a different analytic device,

reviewed next.

4.10 Factorially divergent series

A formal series $A(t) = \sum_{n\geq 0} a_n t^n$ is factorially divergent of type (α, β) if, for every fixed R,

$$a_n = \sum_{j=0}^{R-1} c_j \alpha^{n+\beta-j} \Gamma(n+\beta-j) + O(\alpha^n \Gamma(n+\beta-R)) \,. \tag{4.19}$$

Following Borinsky, define the asymptotic-derivation operator

$$\mathcal{A}_\beta^\alpha A(t) = \sum_{j\geq 0} c_j t^j. \tag{4.20}$$

For analytic series this operator vanishes. On factorially divergent series it satisfies product and composition rules analogous to ordinary differentiation. In the form used later,

$$\mathcal{A}_\beta^\alpha(AB) = (\mathcal{A}_\beta^\alpha A)B + A(\mathcal{A}_\beta^\alpha B), \tag{4.21}$$

$$\mathcal{A}_\beta^\alpha(A \circ g) = A'(g)\mathcal{A}_\beta^\alpha g + \left(\frac{t}{g(t)}\right)^\beta \exp\left(\frac{1/t - 1/g(t)}{\alpha}\right)(\mathcal{A}_\beta^\alpha A)(g(t)), \tag{4.22}$$

provided $g(t) = t + O(t^2)$ in the convention used here.

For the Gaussian core, the coefficient satisfies

$$\frac{(n+1)(n+2)(n+4)!}{48} = \frac{\Gamma(n+7) - 8\Gamma(n+6) + 12\Gamma(n+5)}{48}. \tag{4.23}$$

so the relevant type is $(1, 7)$. If the centered sixth-moment generating function is written as $h(t)\mathcal{N}(g(t))$ with h and g analytic, (4.22) produces the complete coefficient expansion. This is carried out in Section 6.6.

4.11 A practical translation dictionary

For reference, Table 4.1 summarizes the constructions used later.

Table 4.1: Combinatorial constructions used in the determinant calculations.

Structure	EGF operation	Determinant-table use
Labeled product $\mathcal{A} \star \mathcal{B}$	$A(t)B(t)$	Combine disjoint components and interleave their labels.
Set $\mathrm{SET}(\mathcal{A})$	$e^{A(t)}$	Known 6-columns; sets of dependency cycles.
Nonempty sequence $\mathrm{SEQ}_{\geq 1}(\mathcal{A})$	$A/(1-A)$	Paths of known 4-columns.
Possibly empty sequence $\mathrm{SEQ}(\mathcal{A})$	$1/(1-A)$	Chains attached to a pair.
Cycle $\mathrm{CYC}(\mathcal{A})$	$-\log(1-A)$	Dependency cycles of columns or graph edges.
Cycles of length ≥ 2	$-\log(1-A)-A$	Derangements and cycle components without fixed points.
Set of cycles, weight u per cycle	$e^{-uA}(1-A)^{-u}$	Fourth-moment cycles; known-4 and $3+3$ cycles.
Composition $\mathcal{A} \circ \mathcal{B}$	$A(B(t))$	Decorate each Gaussian core column by three chains.
Pointing k ordered labels	$A^{(k)}(t)$	Prescribe the minimal compatible core of a shell.
Signed atom	replace A by $-A$	Determinant parity and alternating $3+3$ paths.

Road map. The exact mean-zero sixth-moment proof in Chapter 6 uses only sets, cycle-weighted derangements, and the substitution $t \mapsto x(t)$. The general shell formula additionally uses derivatives, automorphism factors, and signed local column series.

CHAPTER 5

SECOND AND FOURTH MOMENTS

The second and fourth moments serve as test cases for the table method. They display all of the basic ingredients—column restrictions, signs, cycles, and distribution corrections—without the shell complexity of the sixth moment.

5.1 The second moment

Assume $m_1 = 0$. In a nonzero 2-table, every column must contain an equal pair. Because each row is a permutation, the second row is identical to the first. Thus every table has positive sign and weight m_2^n.

Theorem 5.1.1 (Fortet–Turán)**.** *For i.i.d. centered entries,*

$$f_2(n) = n! m_2^n. \tag{5.1}$$

In particular, if $m_2 = 1$, then $f_2(n) = n!$.

Proof. There are $n!$ choices for the first row, after which the second row is forced. The product of the row signs is one, and each of the n columns contributes m_2. □

After the normalization (3.5),

$$F_2(t) = \sum_{n \geq 0} \frac{(m_2 t)^n}{n!} = e^{m_2 t}. \tag{5.2}$$

5.2 The fourth moment by recurrence

Assume $m_1 = 0$ and $m_2 = 1$. A nonzero column is either a 4-column or a $2+2$ column. Every contributing 4-table has positive sign. One way to see this is to remove the largest label by a sequence of an even number of within-row swaps; induction reduces to a smaller table.

Fix the first row to be the identity and let y_n be the total weight of the resulting tables. Then $f_4(n) = n!y_n$. Exposing the occurrences of the label n yields two cases.

If all four copies of n lie in the final column, removing that column gives the contribution $m_4 y_{n-1}$. Otherwise, n occurs as two pairs in two columns. There are $n-1$ choices for the second column and three choices for the pair of non-n rows in the final column. Introducing an auxiliary quantity for the displaced label and eliminating it gives the classical recurrence.

Theorem 5.2.1 (Nyquist–Rice–Riordan)**.** *For centered variance-one entries,*

$$f_4(n) = n!y_n, \qquad y_n = (n + m_4 - 1)y_{n-1} + (3 - m_4)(n-1)y_{n-2}, \tag{5.3}$$

where $y_0 = 1$ *and* $y_1 = m_4$.

Let

$$Y(t) = \sum_{n \geq 0} y_n \frac{t^n}{n!}.$$

Multiplying (5.3) by $t^{n-1}/(n-1)!$ and summing gives

$$Y'(t) = tY'(t) + m_4 Y(t) + (3 - m_4)tY(t).$$

Solving with $Y(0) = 1$ gives

$$Y(t) = \frac{e^{(m_4-3)t}}{(1-t)^3}. \tag{5.4}$$

Consequently,

$$f_4(n) = \frac{(n!)^2}{2} \sum_{j=0}^{n} \frac{(n-j+1)(n-j+2)}{j!} (m_4 - 3)^j. \tag{5.5}$$

5.3 The fourth moment by analytic combinatorics

The recurrence proof is useful as a warm-up, but the component proof better anticipates the sixth moment.

Represent every 4-column as either known, with weight $m_4 - 3$, or Gaussian-paired. There are three pairings of four positions. After this replacement, all unknown columns are paired $2 + 2$ columns. Their dependency graph is a disjoint union of cycles of length at least two, and each cycle has three possible row-pair patterns. Therefore the unknown part is

$$\mathrm{SET}\big(3\,\mathrm{CYC}_{\geq 2}(\mathcal{Z})\big) \quad \Longrightarrow \quad \frac{e^{-3t}}{(1-t)^3}. \tag{5.6}$$

The known 4-columns form a set with EGF $e^{(m_4-3)t}$. The product is again (5.4), after matching the normalization.

This proof contains the core pattern used later:

1. choose a Gaussian reference multiplicity;
2. put deviations from the reference into known columns;
3. decompose the unknown paired part into connected components; and
4. translate the component decomposition into an EGF.

5.4 Arbitrary mean and marked tables

For completeness, when $m_1 \neq 0$ one centralizes and uses marked 4-tables. The full fourth-moment formula can still be derived by a finite analysis of the number and positions of marks; see [Bec23, BLP25]. The sixth-moment treatment in Chapter 7 follows the same principle but has many more local column types and requires the shell–core reduction.

5.5 Determinants versus permanents

For $k = 2$ and $k = 4$, every nonzero table has positive sign, so

$$p_k(n) = f_k(n)$$

for centered Bernoulli entries. At the sixth moment negative tables appear. For example, with uniform $\{\pm 1\}$ entries,

$$f_6(3) = 1536, \qquad p_6(3) = 2976.$$

The strict inequality is the first visible sign that the sixth moment cannot be reduced to an unsigned pairing count. In the later shell–core implementation, the product of the row signs must therefore be tracked whenever row-equivalent configurations are replaced by one standard representative or separate fragments are declared to represent the same matrix entry.

CHAPTER 6

THE SIXTH MOMENT FOR CENTERED ENTRIES

Source and status. This chapter presents the earlier sixth-moment result of Beck, Potechin, and Lv [BLP23]. The original proof used recurrence relations for paired permutation tables and was developed at greater length in the author's Master's thesis. The presentation here is adapted from the shorter analytic-combinatorics proof in Beck, Potechin, and Lv [BLP25].

Let A_n have i.i.d. real entries with $m_1 = 0$ and $m_2 = 1$. We first allow m_3 to be arbitrary. The nonzero sixth-moment columns have types

$$6, \qquad 4+2, \qquad 3+3, \qquad 2+2+2,$$

with weights $m_6, m_4, m_3^2, 1$. The goal is to express the full table class in terms of the Gaussian paired class $\mathcal{N}$.

6.1 Known and unknown columns

A Gaussian 4-column is counted by its three pairings, and a Gaussian 6-column by its fifteen pairings. We therefore replace a literal column by a sum of known and unknown versions:

$$m_4 = (m_4 - 3) + 3, \tag{6.1}$$

$$m_6 = (m_6 - 15) + 15. \tag{6.2}$$

A known 4-column records a block of four equal labels, has weight $m_4 - 3$, and retains its remaining pair. An unknown column carries a specified pairing and has weight one. A known 6-column has weight $m_6 - 15$. This is an inclusion–exclusion identity at the level of weighted structures: summing over all replacements recovers the original column weight.

The $3+3$ columns need no Gaussian correction, since their weight vanishes for centered

Gaussian entries. Their global structure nevertheless has a signed cycle multiplicity, as described below.

6.2 The four component classes

After the replacement, every table decomposes uniquely into four kinds of components.

6.2.1 Known six-columns

Each known 6-column is an isolated labeled component of weight $(m_6 - 15)t$. A set of such components contributes

$$\exp((m_6 - 15)t). \tag{6.3}$$

6.2.2 Cycles of known four-columns

A known 4-column has one remaining pair. If that pair belongs to another known 4-column, following the displaced labels produces a directed cycle. Cycles have length at least two, and each cycle has fifteen row-support realizations. Hence their class is

$$\mathrm{SET}\big(15\,\mathrm{CYC}_{\geq 2}(q_4\mathcal{Z})\big)\,, \qquad q_4 = m_4 - 3,$$

with EGF

$$\frac{e^{-15q_4t}}{(1-q_4t)^{15}}. \tag{6.4}$$

6.2.3 Cycles of three-columns

A 3+3 column connects two labels. The associated dependency permutation is a derangement. For each cycle there are ten row arrangements, and the determinant parity contributes the

sign encoded by the cycle weight. The resulting factor is

$$(1 + m_3^2 t)^{10} e^{-10 m_3^2 t}. \tag{6.5}$$

A useful check is that this factor has no linear term: a single $3 + 3$ column cannot form a valid table because each label must occur once in every row.

6.2.4 The unknown core and attached paths

Remove the three component types above. The remaining unknown columns form a Gaussian paired table, the *core*. A pair in the core may be reached by a path of known 4-columns. Each core column has three pairs, and each pair supports a possibly empty path with EGF $1/(1 - q_4 t)$. Consequently, each core atom t is replaced by

$$x(t) = \frac{t}{(1 - q_4 t)^3},$$

and the core factor is

$$\mathcal{N}(x(t)). \tag{6.6}$$

6.3 The centered sixth-moment generating function

Combining disjoint component classes by the labeled product gives the main formula.

Theorem 6.3.1 (Centered sixth moment)**.** *Let A_n have i.i.d. real entries with $m_1 = 0$ and $m_2 = 1$. Then*

$$\boxed{F_6(t) = (1 + m_3^2 t)^{10} \frac{e^{(m_6 - 10 m_3^2 - 15 m_4 + 30)t}}{(1 - (m_4 - 3)t)^{15}} \mathcal{N}\left(\frac{t}{(1 - (m_4 - 3)t)^3} \right)} \tag{6.7}$$

where

$$\mathcal{N}(t) = \frac{1}{48}\sum_{i\geq 0}(i+1)(i+2)(i+4)!t^i.$$

Equivalently,

$$\boxed{F_6(t) = \frac{(1+m_3^2 t)^{10}e^{(m_6-10m_3^2-15m_4+30)t}}{48(1+3t-m_4t)^{15}}\sum_{i\geq 0}\frac{(i+1)(i+2)(i+4)!t^i}{(1+3t-m_4t)^{3i}}.} \tag{6.8}$$

Proof. Multiply the four component factors (6.3), (6.4), (6.5), and (6.6). The exponent simplifies as

$$(m_6-15)-15(m_4-3)-10m_3^2 = m_6-10m_3^2-15m_4+30.$$

The second form follows by substituting the definition of $\mathcal{N}$ and observing that $1-(m_4-3)t = 1+3t-m_4t$. □

Remark 6.3.2 (The Gaussian specialization). For standard Gaussian entries, $m_3 = 0$, $m_4 = 3$, and $m_6 = 15$. All correction factors equal one, so $F_6(t) = \mathcal{N}(t)$ and

$$f_6(n) = \frac{n!(n+2)!(n+4)!}{48},$$

as required by (1.2).

Remark 6.3.3 (The symmetric-entry specialization). If the entry distribution is symmetric about zero, then $m_3 = 0$ and the factor (6.5) disappears. This specialization is also treated in the author's Master's thesis [Lv23].

6.4 An exact coefficient formula

Set

$$q_3 = m_3^2, \qquad q_4 = m_4 - 3, \qquad q_6 = m_6 - 10m_3^2 - 15m_4 + 30.$$

Expanding the three factors in (6.8) gives the following finite expression.

Corollary 6.4.1. *For $n \geq 0$,*

$$f_6(n) = (n!)^2 \sum_{j=0}^{n} \sum_{i=0}^{j} \sum_{k=0}^{n-j} \frac{(i+1)(i+2)(i+4)!}{48(n-j-k)!} \binom{10}{k} \times \binom{14+j+2i}{j-i} q_6^{n-j-k} q_4^{j-i} q_3^k. \tag{6.9}$$

Proof. Expand

$$(1+q_3 t)^{10}, \qquad e^{q_6 t}, \qquad \frac{1}{(1-q_4 t)^{15+3i}} = \sum_{r \geq 0} \binom{14+3i+r}{r} q_4^r t^r,$$

and set $r = j - i$. The binomial coefficient becomes $\binom{14+j+2i}{j-i}$. □

If $m_3 = 0$, only $k = 0$ remains. The formula then agrees with the statement in the author's Master's thesis [Lv23].

6.5 Small values and cancellation

For uniform $\{\pm 1\}$ entries,

$$m_3 = 0, \qquad m_4 = m_6 = 1, \qquad q_4 = -2, \qquad q_6 = 16.$$

The first values are

n	1	2	3	4	5
$f_6(n)$	1	32	1536	282624	66846720
$p_6(n)$	1	32	2976	513024	157854720

The gap between determinant and permanent moments begins at $n = 3$ and records genuine sign cancellation.

6.6 All-orders asymptotics

Write

$$F_6(t) = h(t)\,\Phi(g(t)), \tag{6.10}$$

where

$$\begin{aligned}
\Phi(t) &= \sum_{i\geq 0}(i+1)(i+2)(i+4)!\,t^i,\\
g(t) &= \frac{t}{(1-q_4 t)^3},\\
h(t) &= \frac{(1+q_3 t)^{10}e^{q_6 t}}{48(1-q_4 t)^{15}}.
\end{aligned}$$

The coefficients of Φ satisfy

$$(i+1)(i+2)(i+4)! = \Gamma(i+7) - 8\Gamma(i+6) + 12\Gamma(i+5), \tag{6.11}$$

so Φ is factorially divergent of type $(1, 7)$. Applying the composition rule (4.22) yields a formal series

$$C(t) = \sum_{r\geq 0} c_r t^r = \mathcal{A}_7^1 F_6(t)$$

that encodes the complete asymptotic expansion.

A convenient closed form is

$$\begin{aligned} C(t) &= e^{(q_6-3q_4^2)t+q_4^3t^2}(1+q_3t)^{10} \\ &\quad\times\big[1-2(3q_4+4)t+3(5q_4^2+8q_4+4)t^2 \\ &\qquad -4q_4^2(5q_4+6)t^3+q_4^3(15q_4+8)t^4-6q_4^5t^5+q_4^6t^6\big]. \end{aligned} \tag{6.12}$$

Theorem 6.6.1 (All-orders expansion)**.** *For every fixed $R \geq 0$,*

$$f_6(n) = \frac{e^{3q_4}(n!)^2}{48}\left(\sum_{r=0}^{R} c_r(n+6-r)!\right) + O\Big((n!)^2(n+5-R)!\Big). \tag{6.13}$$

The leading term is

$$f_6(n) \sim \frac{e^{3m_4-9}}{48}(n!)^3n^6. \tag{6.14}$$

Higher coefficients are explicit polynomials in m_3, m_4, m_6. For Bernoulli entries, the expansion begins

$$\begin{aligned} f_6(n) \sim \frac{(n!)^3}{48e^6}\Big(&n^6+29n^5+335n^4+\frac{5861}{3}n^3 \\ &+\frac{17944}{3}n^2+\frac{44036}{5}n+\frac{167536}{45}-\frac{210176}{63n}+\cdots\Big). \end{aligned} \tag{6.15}$$

6.7 What must change for nonzero mean

The centered formula succeeds because every nonzero column can be organized around pairings, known blocks, and $3+3$ cycles. For $m_1 \neq 0$, marked columns create partial row supports that need not close into components locally. A triple may connect to a mark, a block of four may be covered by marks, and anonymous pairs can later be identified with labeled fragments. The zero-mark series (3.20) remains universal, but it becomes only one factor in a finite shell sum. The next chapter develops the replacement rules needed to isolate that shell.

CHAPTER 7

MARKED TABLES AND REPLACEMENT RULES

Source and status. This chapter is adapted from the replacement-rule portion of the general sixth-moment paper [BLP26]. The paper's table definitions, replacement rules, and component factorization are retained in dissertation notation.

7.1 Permutation tables decomposition

Let us set the stage for the proof of the main result. For that, we need to recall some constructions on permutation tables. Let S_n be the set of all permutations on $[n] = \{1, 2, 3, \ldots, n\}$, then by definition of determinant,

$$\det A = \sum_{\pi \in S_n} \operatorname{sgn} \pi \prod_{i=1}^{n} X_{i\pi(i)}. \tag{7.1}$$

Taking the 6-th power and applying the expectation operator, we get

$$f_6(n) = \sum_{T \in \mathcal{T}_{6,n}} w(T) \operatorname{sgn} T, \tag{7.2}$$

where $\mathcal{T}_{6,n} := S_n \times S_n \times S_n \times S_n \times S_n \times S_n$ is the set of all *permutation tables* with 6 rows and n columns and where

$$\operatorname{sgn} T = \operatorname{sgn} \pi_1 \operatorname{sgn} \pi_2 \cdots \operatorname{sgn} \pi_6 \tag{7.3}$$

is the *sign* and

$$w(T) = \mathbb{E} \prod_{j=1}^{6} \prod_{i=1}^{n} X_{i\pi_j(i)} \tag{7.4}$$

is the *weight* of a given permutation table $T = \pi_1 \times \pi_2 \times \pi_3 \times \pi_4 \times \pi_5 \times \pi_6 \in \mathcal{T}_{6,n}$. Due to independence, the weight $w(T)$ can be written as a product of weights $w(\mathbf{c})$ of individual columns $\mathbf{c}$ of T. An example of a table T is shown below in Fig. 7.1 together with its column

weights. Its seventh column corresponds to the term $X_{79}^2 X_{78}^4$ in the expansion of $(\det A)^6$ with *weight of a column* given by the expectation $w(\mathbf{c}) = \mathbb{E}[X_{79}^2 X_{78}^4] = m_2 m_4$.

8	5	1	10	4	7	9	11	3	2	6	−
10	9	11	7	2	5	8	1	4	3	6	+
10	9	11	7	3	5	8	4	2	1	6	−
10	9	11	7	1	5	8	2	4	3	6	−
10	5	9	7	11	4	8	3	2	1	6	−
10	5	2	8	7	4	9	1	3	11	6	−
$m_1 m_5$	m_3^2	$m_1^3 m_3$	$m_1^2 m_4$	m_1^6	$m_1 m_2 m_3$	$m_2 m_4$	$m_1^4 m_2$	m_2^3	$m_1^2 m_2^2$	m_6	

Figure 7.1: A permutation table $T \in \mathcal{T}_{6,11}$ with $w(T) = m_1^{19} m_2^8 m_3^4 m_4^2 m_5 m_6$ and $\operatorname{sgn} T = -1$.

7.1.1 Marked permutation tables

It turns out that we can drastically reduce the number of tables by centering the random entries X_{ij} in A. Below we state the basic results. For more details, see [Lv23] and Beck, Potechin, and Lv [BLP25]. Let $Y_{ij} := X_{ij} - m_1$ and let $\mu_q = \mathbb{E}[Y_{11}^q]$ be their moments (central moments of the entry distribution). Let $S_n^\times$ be the set of *marked permutations* where at most one element of a permutation on n elements is decorated by a mark ($\times$). Then, by the matrix determinant lemma,

$$\det A = \sum_{\pi \in S_n^\times} \operatorname{sgn} \pi \prod_{i=1}^{n} Y_{i\pi(i)}^\times, \tag{7.5}$$

where $Y_{i\pi(i)}^\times = m_1$ if the element i is marked and $Y_{i\pi(i)}^\times = Y_{i\pi(i)}$ otherwise. Again by raising to the 6 power, expanding and applying the expectation operator, we get

$$f_6(n) = \sum_{T \in \mathcal{T}_{6,n}^\times} w(T) \operatorname{sgn} T, \tag{7.6}$$

where $\mathcal{T}_{6,n}^{\times} := (S_n^{\times})^6$ is the set of all *marked permutation tables* with 6 rows and n columns and where the sign $\operatorname{sgn} T$ corresponds to the sign of the underlying unmarked permutation table (equals the product of signs of the individual permutations) and where

$$w(T) = \mathbb{E} \prod_{j=1}^{6} \prod_{i=1}^{n} Y_{i\pi_j(i)} \tag{7.7}$$

is again the *weight* of a given marked permutation table $T \in \mathcal{T}_{6,n}^{\times}$. We also denote $\mathcal{T}_6^{\times} = \bigcup_{n\in\mathbb{N}} \mathcal{T}_{6,n}^{\times}$ as the set of all marked permutation tables with 6 rows irrespective of the number of columns. The weight $w(T)$ can be written as a product of weights $w(\mathbf{c})$ of individual columns $\mathbf{c}$ of T. Note that the column weight $w(\mathbf{c})$ is independent of what element is being covered by the mark $\times$. However, even if not shown next to a mark, this marked element is always uniquely deducible from other elements in the same row since each row contains at most one mark and uses up all elements from $[n]$ (no element repeats in a row of T). The types of all columns $\mathbf{c}$ with nonzero weight which can appear in a table T (up to permutation of rows) are shown below (including their weights $w(\mathbf{c})$). Elements $a, b, c, \ldots$ are placeholders representing *different* numbers (that is, $a \neq b, a \neq c, b \neq c$, etc.). The column types have names (the first row with $\mathbf{c}$) which we will use often. The notation $\times_s^r$ means a column with r marks and, in case of ambiguity, with s the size of the largest uncovered tuple.

$\mathcal{T}_{6,n}^{\times}$:

$\mathbf{c}$:	6	4	3	2	$\times_5^1$	$\times_3^1$	$\times_4^2$	$\times_2^2$	$\times^3$	$\times^4$	$\times^6$
	a	a	a	a	$\times$	$\times$	$\times$	$\times$	$\times$	$\times$	$\times$
	a	a	a	a	a	a	$\times$	$\times$	$\times$	$\times$	$\times$
	a	a	a	b	a	a	a	a	$\times$	$\times$	$\times$
	a	a	b	b	a	a	a	a	a	$\times$	$\times$
	a	b	b	c	a	b	a	b	a	a	$\times$
	a	b	b	c	a	b	a	b	a	a	$\times$
$w(\mathbf{c})$:	μ_6	μ_4	μ_3^2	1	$m_1\mu_5$	$m_1\mu_3$	$m_1^2\mu_4$	m_1^2	$m_1^3\mu_3$	m_1^4	m_1^6

An example of a marked permutation table T is shown below in Fig. 7.2. Its sixth column corresponds to the term $m_1 Y_{68}^2 Y_{62}^3$ in the expansion of $(\det A)^6$ with *weight of a column* given

by the expectation $w(\mathbf{c}) = \mathbb{E}[m_1 Y_{68}^2 Y_{62}^3] = m_1\mu_3$ (recall we assume $\mu_2 = 1$).

5	3	1	2	$\times 4$	8	6	7	+
5	3	1	8	4	$\times 2$	6	7	−
5	3	6	8	4	2	7	$\times 1$	−
5	3	6	2	4	8	7	$\times 1$	+
5	8	1	3	4	2	6	7	+
5	8	6	3	4	2	$\times 1$	7	−
μ_6	μ_4	μ_3^2	1	$m_1\mu_5$	$m_1\mu_3$	$m_1\mu_3$	$m_1^2\mu_4$	

Figure 7.2: A permutation table $T \in \mathcal{T}_{6,8}^{\times}$ with columns $6, 4, 3, 2, \times\frac{1}{5}, \times\frac{1}{3}, \times\frac{1}{3}, \times\frac{2}{4}$, carrying 5 marks in distinct rows, $w(T) = m_1^5\mu_3^4\mu_5\mu_4^2\mu_6$, and $\operatorname{sgn} T = -1$.

7.1.2 Decomposition based on the number of marks

Definition 7.1.1. We say a marked permutation table is r-marked if it contains exactly r marks. We denote $\mathcal{T}_{k,n}^{(r)}$ as the set of all r-marked permutation k-tables consisting of n columns, after the common factor m_1^r has been pulled out. Their column weights are $(1, \mu_2, \ldots, \mu_k)$.

$$F_k(t) = \sum_{r=0}^{k} m_1^r T_k^{(r)}(t), \tag{7.8}$$

where $T_k^{(r)}$ are functions dependent only on $t, \mu_2, \ldots, \mu_k$. Let

$$t_k^{(r)}(n) = \sum_{T \in \mathcal{T}_{k,n}^{(r)}} w(T) \operatorname{sgn} T, \tag{7.9}$$

then

$$T_k^{(r)}(t) = \sum_{n=0}^{\infty} \frac{t_k^{(r)}(n)}{n!^2} t^n. \tag{7.10}$$

For the sixth moment this gives

$$F_6(t) = \sum_{r=0}^{6} m_1^r T_6^{(r)}(t). \tag{7.11}$$

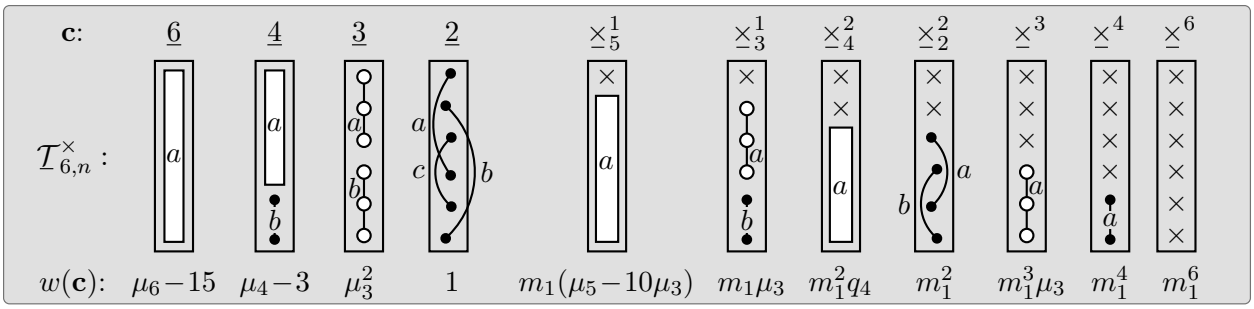
7.2 Paired/Blocked tables

Here now on, we assume $\mu_2 = 1$. To utilize the correspondence with the known machinery for the Gaussian case (in Beck, Potechin, and Lv [BLP23]) for which $\mu_6 = 15, \mu_5 = 0, \mu_4 = 3$ and $\mu_3 = 0$, we introduce another set of tables.

Definition 7.2.1. Let the *paired/blocked tables* $\underline{\mathcal{T}}_{6,n}^{\times}$ be marked permutation tables $\mathcal{T}_{6,n}^{\times}$ in which the largest tuple of at least four identical elements in some columns are *blocked* and all remaining elements are *paired* (each pair must have identical elements). The paired/blocked tables consist of *known* (blocked) columns and *unknown* (paired) columns. Their weight is carefully chosen via replacements (inspired by inclusion/exclusion in Beck, Potechin, and Lv [BLP23]) such that in the end we have

$$f_6(n) = \sum_{T \in \mathcal{T}_{6,n}^{\times}} w(T) \operatorname{sgn} T = \sum_{T \in \underline{\mathcal{T}}_{6,n}^{\times}} w(T) \operatorname{sgn} T. \tag{7.12}$$

Columns of $\underline{\mathcal{T}}_{6}^{\times} = \bigcup_{n \in \mathbb{N}} \underline{\mathcal{T}}_{6,n}^{\times}$ with their weights are shown below. To distinguish them from columns of $\mathcal{T}_6^{\times}$, we use an underline, so an $\underline{6}$-column is a column of a paired/blocked table. If a column has no blocked elements nor marks, we say it is an *unknown* column (this is the column $\underline{2}$). Note that now some of the elements $a, b, c, \ldots$ may represent the *same* number (that is, we may have $a = b, a = c, b = c$, etc.)—we are still able to distinguish the columns even if some elements are the same because of pairings and blocks.



An example of a paired/blocked table is shown below in Fig. 7.3.

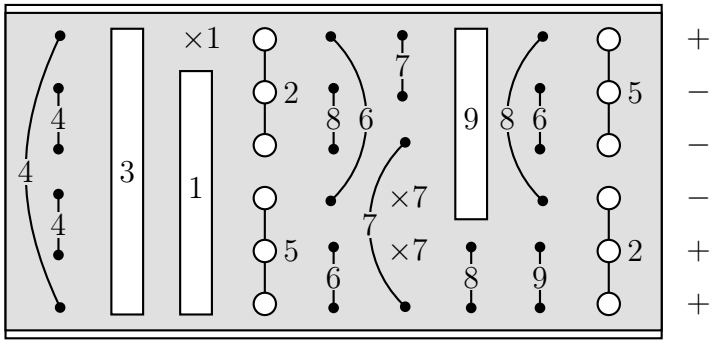


Figure 7.3: A paired/blocked table $T \in \mathcal{T}^{\times}_{6,9}$ with $w(T) = m_1^3\mu_3^4(\mu_4-3)(\mu_5-10\mu_3)(\mu_6-15)$ and $\operatorname{sgn} T = -1$.

Finally, we show the replacement rules for columns. For a given table $T \in \mathcal{T}^{\times}_{6,n}$, we generate a set of tables $T' \in \underline{\mathcal{T}}^{\times}_{6,n}$ in which each column $\mathbf{c}$ in T was replaced by blocked/paired columns $\mathbf{c}'$. This replacement is not a bijection since for a given column, we may replace it by many possible blocked/paired columns, so the set of T' tables may be large. However, Eq. (7.12) stays preserved if

- for any $T' \in \underline{\mathcal{T}}^{\times}_{6}$, there is a unique map from T' to T and
- the sum of weights of all columns $\mathbf{c}'$ equals $w(\mathbf{c})$.

Both conditions above can be satisfied by replacement rules shown in Figs. 7.4 to 7.7. The remaining correspondences of columns not shown here are simply bijectively paired up, so for example 2-columns map to $\underline{2}$ by pairing up the identical elements (only one pairing exists among those columns), the same goes with 3-columns, that is 3-columns match $\underline{3}$-columns,

etc. Those columns do not have to be called known/unknown.

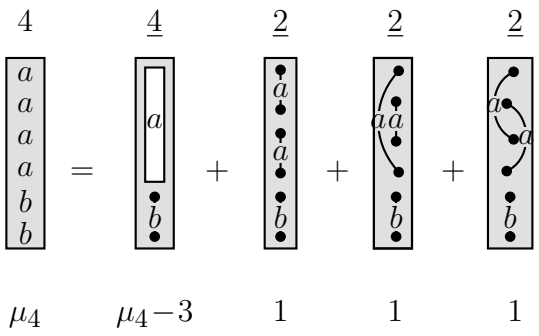


Figure 7.4: Replacement of 4-columns

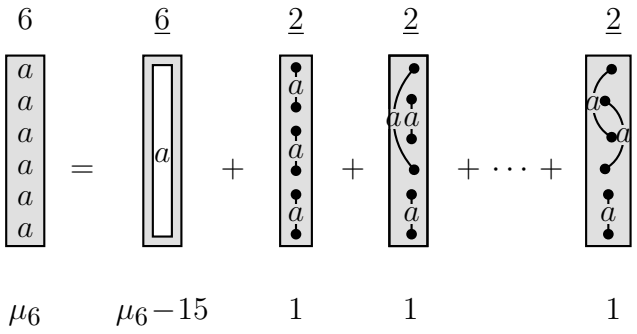


Figure 7.5: Replacement of 6-columns

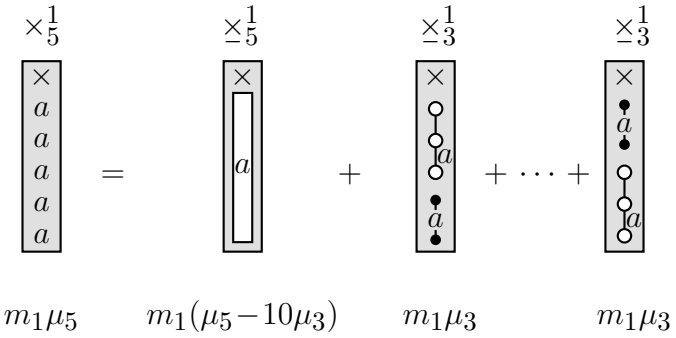


Figure 7.6: Replacement of $\times\frac{1}{5}$-columns

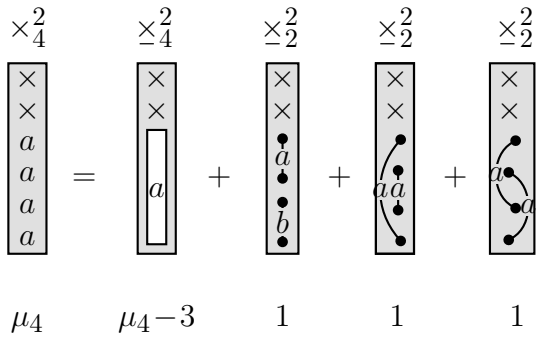


Figure 7.7: Replacement of $\times_4^2$-columns

Throughout the rest of the chapter, we assume T is a paired/blocked marked permutation table.

7.2.1 Connection formula

The $\times_{=5}^1$ columns are always disjoint from each other and the rest of the table. The fact that there cannot be a single uncovered element together with the rule that there is at most one mark per row forces the covered element in a $\times_{=5}^1$ column to always coincide with the block element. Hence, in terms of generating functions, one gets, by collecting those $\times_{=5}^1$ columns,

$$F_6(t) = \sum_{r=0}^{6} m_1^r \left(1 + (\mu_5 - 10\mu_3)\, m_1 t\right)^{6-r} Z_r(t), \tag{7.13}$$

where $Z_r(t)$ are the generating functions of pair/block tables with exactly r marks which do not contain $\times_{=5}^1$ columns. We further set $m_1 = 1$ in those tables since m_1^r has been already pulled out.

7.2.2 Factorisation of disjoint components

We utilize the fact the exponential generating function (EGF) of permutation tables (of any kind) factorizes over disjoint columns (see [Lv23]). Already in the case $m_1 = 0$ treated

in Beck, Potechin, and Lv [BLP25], we identified the three types of sub-components of paired/blocked tables which are disjoint from everything else, namely $\underline{6}$-columns, cycles of $\underline{4}$ columns and cycles of $\underline{3}$ columns. We call the union of these three sub-components the *floating component* of the table. In terms of EGFs, they contribute a factor

$$\begin{aligned}\mathrm{Fl}(t) = \underbrace{e^{(\mu_6-15)t}}_{\text{known 6-columns}} \underbrace{\frac{e^{-15(\mu_4-3)t}}{(1-(\mu_4-3)t)^{15}}}_{\text{cycles of known 4-columns}} \\ \times \underbrace{(1+\mu_3^2 t)^{10} e^{-10\mu_3^2 t}}_{\text{cycles of 3-columns}} = \frac{\mathcal{O}(t)}{\mathcal{N}(x(t))}.\end{aligned} \tag{7.14}$$

where the last equality comes from Eq. (3.20). This factor must appear also in the general case. More concretely, the generating function $F_6(t)$ corresponding to all paired/blocked tables equals to the product of $\mathcal{O}(t)/\mathcal{N}(x(t))$ with a generating function corresponding to all paired/blocked tables **lacking** $\underline{6}$-columns, cycles of $\underline{4}$ columns and cycles of $\underline{3}$ columns.

CHAPTER 8

THE SHELL–CORE DECOMPOSITION

Source and status. This chapter retains the shell–core development and reconstruction examples from the general sixth-moment paper [BLP26]. The paper's implementation subsection is omitted because Chapter 10 gives the extended dissertation treatment.

8.1 Decomposition of a Table Based on its Shell and Core

Given a table T, we can perform the following operations to reduce T to a simpler table.

Definition 8.1.1. Given a table T, we say that an element a is uncovered if it appears 6 times in T (i.e., it is never replaced by an $\times$). Otherwise, we say that the element a is covered.

Definition 8.1.2. Given a table T, we define the reduced table T_{red} to be the table which is obtained from T after applying the following steps:

1. If there is an uncovered element a which appears as a triple or block of size 4 in a column which also contains an $\times$, we can delete the entries containing a and then merge the two columns which contained these entries. After doing this merge, we put the resulting column in the same place as the column which contained an a but did not have an a in its first row (deleting the column which had an a in its first row). We repeat this step until there are no such uncovered elements a.

2. If there is an uncovered element b which appears as a block of size 4 in one column and appears 2 times in a different column with an $\times$, we can delete the entries containing b and then merge the two columns which contained these entries. Similarly, if there is an uncovered element b which appears as a triple in two different columns, we can delete the entries containing b and then merge the two columns which contained these entries.

After doing such a merge, we put the resulting column in the same place as the column which contained a b but did not have a b in its first row (deleting the column which had a b in its first row). Finally, if there is an uncovered b which appears as a block of size 6 or appears 6 times in a column due to a previous merge (in which case it will either appear as a block of size 4 and a pair or appear as two triples), we can delete this column. Note that if T starts with an unknown 6-column then we do not delete this column as this was present from the start rather than being due to a merger of columns.

We repeat these steps until there are no such uncovered elements b.

Remark 8.1.3. When merging columns, it doesn't really matter which column is deleted as long as we have a consistent procedure for picking this column.

For our analysis, it is very helpful to split the columns of T_{red} into two parts which we call the shell and core of T.

Definition 8.1.4 (Shells and Cores of a Table)**.** Given a table T, we define the shell $\text{shell}(T)$ of T to be the columns of T_{red} such that at least one of the following is true.

1. The column contains at least one $\times$.
2. The column contains a triple of a covered element (in which case it must also contain at least one $\times$ or a triple of a second covered element).
3. The column contains a block of size 4 or 5 of a covered element.

We define the core $\text{core}(T)$ of T to be the columns of T_{red} which are not in the shell.

Example 8.1.5. If T is the following table

$\times a$	b	d	c	e	f	g
$\times b$	d	g	c	a	e	f
c	$\times a$	d	f	e	b	g
c	b	d	a	e	f	g
c	b	d	f	a	e	g
c	d	g	a	e	b	f

where the first, third, and fifth columns are all known 4-columns, then in the first step of the decomposition, we merge the columns containing c which gives us the following table:

$\times a$	b	d	e	f	g
$\times b$	d	g	a	e	f
f	$\times a$	d	e	b	g
a	b	d	e	f	g
f	b	d	a	e	g
a	d	g	e	b	f

In the second stage of the decomposition, we merge the columns containing d, merge the columns containing e, and merge the columns containing g. This gives the following table T_{red}:

$\times a$	b	f
$\times b$	f	a
f	$\times a$	b
a	b	f
f	b	a
a	f	b

shell(T) and core(T) are as follows:

$\times a$	b
$\times b$	f
f	$\times a$
a	b
f	b
a	f

f
a
b
f
a
b

Example 8.1.6. If T is the following table

j	e	$\times a$	d	n	r	k	f	i	l	c	o	m	s	p	q	g	h	b
h	e	$\times a$	d	o	r	k	f	i	l	c	q	m	p	s	n	g	j	b
j	e	h	d	o	$\times b$	a	r	i	l	c	q	f	s	p	n	g	m	k
f	e	j	$\times c$	n	d	a	r	i	l	b	o	g	p	s	q	m	h	k
f	$\times c$	h	l	n	d	a	r	i	e	b	o	g	s	p	q	m	j	k
h	$\times c$	j	l	o	d	a	r	i	e	b	q	f	s	p	n	g	m	k

where the column with six i is a known 6-column, the columns with four a, e, g, l, p, r, and s are known four columns, and the column with four k is an unknown 4-column with the given pairing then in the first step of the decomposition, we merge the columns containing d, we merge the columns containing e, and we merge the columns containing l (this must be done after merging the columns containing e) which gives us the following table:

j	$\times a$	n	r	k	f	i	c	o	m	s	p	q	g	h	b
h	$\times a$	o	r	k	f	i	c	q	m	p	s	n	g	j	b
j	h	o	$\times b$	a	r	i	c	q	f	s	p	n	g	m	k
f	j	n	$\times c$	a	r	i	b	o	g	p	s	q	m	h	k
f	h	n	$\times c$	a	r	i	b	o	g	s	p	q	m	j	k
h	j	o	$\times c$	a	r	i	b	q	f	s	p	n	g	m	k

In the second step, we merge the columns containing r and we merge the columns containing g. We also delete the column containing i (as this is a known 6-column), the columns containing

p and s (as this is a cycle of known 4-columns) and the columns containing n, o, and q (as this is a cycle of columns with two unmarked triples). This gives the following table T_{red}:

j	$\times a$	k	f	c	m	h	b
h	$\times a$	k	f	c	m	j	b
j	h	a	$\times b$	c	f	m	k
f	j	a	$\times c$	b	m	h	k
f	h	a	$\times c$	b	m	j	k
h	j	a	$\times c$	b	f	m	k

shell(T) and core(T) are as follows:

$\times a$	k	f	c
$\times a$	k	f	c
h	a	$\times b$	c
j	a	$\times c$	b
h	a	$\times c$	b
j	a	$\times c$	b

j	m	h	b
h	m	j	b
j	f	m	k
f	m	h	k
f	m	j	k
h	f	m	k

For our analysis, we will enumerate possible shells and cores so it is useful to note the properties which shells and cores must have.

Definition 8.1.7. We say that T is a partial table if T satisfies the following conditions:

1. Each row of T contains at most one $\times$ and most one of each element.
2. Each element which is present in T appears a total of 2, 4, or 6 times in T (including the times where it is marked).
3. The columns of T do not contain any singletons. In other words, each column of T consists of marks, pairs, triples, and/or blocks of size 4, 5, or 6.

Whenever a partial table T contains an unknown 4-column or 6-column, it must specify how these elements are paired up with each other.

Note that any table with nonzero expected value must be a partial table.

Definition 8.1.8 (Shells)**.** We say that a partial table S is a shell if every column of S contains a mark, a triple of a covered element, and/or a block of size 4 of a covered element.

Definition 8.1.9. We say that a partial table C is a core if it only contains pairs (i.e., C does not have any marks, triples, or blocks of size 4, 5, or 6).

8.2 Reversing the Shell/Core Decomposition

We now describe what data is needed in order to recover a table T from shell(T) and core(T). For this process, it is convenient to keep track of each component separately and then combine them at the end. In particular, we keep track of an expanded shell, an expanded core, and a floating component consisting of columns which were deleted.

Definition 8.2.1. Given a table T, we define the expanded shell of T to be the set of columns of T which are present in shell(T) or become part of the columns of shell(T) through merges. Similarly, we define the expanded core of T to be the set of columns of T which are present in core(T) or become part of the columns of core(T) through merges. We define the floating component of T to be the columns of T which are not part of the expanded shell or expanded core of T.

To reverse the second step of the decomposition, we need the following data:

1. To obtain the floating component of T, we need to know which known 6-columns, cycles of known 4-columns, and cycles of pairs of triples were deleted and the locations of these deleted columns in relation to each other.

2. To obtain the expanded shell after the first step of the decomposition, we need to know which paths of known 4-columns were merged into the columns of shell(T) and the locations of the columns which were deleted during these merges. Similarly, to obtain

the expanded core, we need to know which paths of known 4-columns were merged into the columns of core(T) and the locations of the columns which were deleted during these merges.

Remark 8.2.2. To specify a path of known 4-columns which are merged into a column of the shell or core without specifying the locations of these columns, we can give the following data:

1. The pair of elements which starts in one of these known 4-columns and ends up as a pair in the resulting column of the shell or core. We call this pair the surviving pair as it is the only part remaining from these columns after the merges.
2. The final location of the surviving pair. This will be a pair in the shell or core and we consider this pair to be the destination of the path.
3. The sequence of elements which are deleted if at each step, we take the element which appears as a block of size 4 in the column with the surviving pair and merge the two columns which contain this element.

As in [BLP23], we represent this path of known 4-columns by the sequence of elements which are deleted followed by the element in the surviving pair (we put this element last because it ends up at the destination of the path).

For example, we would represent the merges

f	d	e	a
f	d	e	a
b	f	d	e
b	f	d	e
c	f	d	e
c	f	d	e

$\rightarrow$

a
a
b
b
c
c

by the path $e \to d \to f \to a$. and note the destination of the path (i.e., the final position of the surviving pair of a).

To reverse the first step of the decomposition, we need the following data:

1. For each column in the shell with at least one mark, we need to know how the column was originally partitioned. As described in Section 9.2.2, this partition can be described by a set of splits, each of which is a 4-2 split or a 3-3 split.

2. For each 4-2 split, we need to know about the sequence of uncovered elements which were originally in between the two parts (starting from the part of size 2) and the locations of the columns which were deleted during the merges.

3. For each 3-3 split, we need to know about the sequence of uncovered elements which were originally in between the two parts (starting from the part of size 3 which includes the first row) and the locations of the columns which were deleted during the merges.

Example 8.2.3. Recall that the shell and core for Example 8.1.5 are as follows:

$\times a$	b	f
$\times b$	f	a
f	$\times a$	b
a	b	f
f	b	a
a	f	b

For Example 8.1.5, there is no floating component. To recover the expanded shell after the first stage of the decomposition, we need to know that there was a path $g \to d \to f$ of known 4-columns leading to the pair of f in the second column of the shell, the known 4-column with 4 d was to the right of the second column of the shell, and the known 4-column with 4 g was to the right of the known 4-column with 4 d.

To recover the expanded core of T, we need to know that there was a path $e \to a$ of a known 4-column leading to the pair of a in the core and that the known 4-column with 4 e was to the left of the column in the core.

Using this data, we can deduce that the expanded shell and core after the first step of the decomposition were as follows.

$\times a$	b	d	g
$\times b$	d	g	f
f	$\times a$	d	g
a	b	d	g
f	b	d	g
a	d	g	f

e	f
a	e
e	b
e	f
a	e
e	b

To reverse the effect of the first step of the decomposition on the expended shell of T, we need to know that the first column originally had a 4-2 split between the top two rows and the bottom four rows, that the uncovered element c was originally between these two parts, and that the column with elements c, c, f, a, f, a which was deleted was between the column with a block of 4 d and the column with a block of 4 g. With this data, we can deduce that the expanded shell of T is

$\times a$	b	d	c	g
$\times b$	d	g	c	f
c	$\times a$	d	f	g
c	b	d	a	g
c	b	d	f	g
c	d	g	a	f

We can complete the recovery of T by specifying which columns of T are in the shell, which columns of T are in the core, and which columns of T are in the floating component (which is empty in this example).

Example 8.2.4. Recall that the shell and core for Example 8.1.6 are as follows:

$\times a$	k	f	c
$\times a$	k	f	c
h	a	$\times b$	c
j	a	$\times c$	b
h	a	$\times c$	b
j	a	$\times c$	b

j	m	h	b
h	m	j	b
j	f	m	k
f	m	h	k
f	m	j	k
h	f	m	k

We first reverse the second step of the decomposition. In order to recover the floating component of T, we need the following data:

1. We need to know about the known 6-column with a block of six i.
2. We need to know about the cycle $s \to p \to s$ of known 4-columns where the blocks of size 4 are in rows $1, 3, 5, 6$.
3. We need to know about the cycle $n \to o \to q \to n$ of columns with two pairs of triples where the first triple is in rows $1, 4, 5$ and the second triple is in rows $2, 3, 6$.
4. We need to know that the columns of the floating component are ordered so that the order of the elements in the first row is n, i, o, s, p, q.

From this data, we can deduce that the floating component is

n	i	o	s	p	q
o	i	q	p	s	n
o	i	q	s	p	n
n	i	o	p	s	q
n	i	o	s	p	q
o	i	q	s	p	n

In order to recover the expanded shell and core after the first step of the decomposition, we need the following data:

1. There is a path $r \to f$ of known 4-columns whose destination is the pair of f in the third column of the shell. The column which contained the elements $r, r, \times b, \times c, \times c, \times c$ (after the first stage of the decomposition) was originally between the first and second columns of shell(T).

2. There is a path $g \to m$ of known 4-columns whose destination is the pair of m in rows 4 and 5 of the second column of the core. The column which contained a block of size 4 of g and a pair of m was originally in between the second and third columns of core(T).

Using this data, we can deduce that the expanded shell and core after the first step of the decomposition were as follows:

$\times a$	r	k	f	c
$\times a$	r	k	f	c
h	$\times b$	a	r	c
j	$\times c$	a	r	b
h	$\times c$	a	r	b
j	$\times c$	a	r	b

j	m	g	h	b
h	m	g	j	b
j	f	g	m	k
f	g	m	h	k
f	g	m	j	k
h	f	g	m	k

To reverse the effect of the first step of the decomposition on the expended shell of T, we need the following data:

1. For the column with elements $r, r, \times b, \times c, \times c, \times c$, we originally had a 4-2 split between rows $1, 2, 3, 4$ and $5, 6$ and a 3-3 split between rows $1, 2, 3$ and $4, 5, 6$.

2. For the 4-2 split, we originally had the uncovered elements e and l between the part of size 2 in rows 5 and 6 and the part of size 4 (which was originally split by the 3-3 split).

3. The column with elements $e, e, e, e, \times c, \times c$ was the first column of the expanded shell of T and the column with elements l, l, l, l, e, e was between the column with elements f, f, r, r, r, r and the column with elements c, c, c, b, b, b.

4. For the 3-3 split, we originally had the uncovered element d in between the two parts of size 3.

5. The column with elements $d, d, d, \times c, l, l$ was originally in between the column with elements $\times a, \times a, h, j, h, j$ and the column with elements $r, r, \times b, d, d, d$.

With this data, we can deduce that the expanded shell of T is

e	$\times a$	d	r	k	f	l	c
e	$\times a$	d	r	k	f	l	c
e	h	d	$\times b$	a	r	l	c
e	j	$\times c$	d	a	r	l	b
$\times c$	h	l	d	a	r	e	b
$\times c$	j	l	d	a	r	e	b

We can complete the recovery of T by specifying which columns of T are in the shell, which columns of T are in the core, and which columns of T are in the floating component.

8.3 Sketch for Computing $F_6(t)$

We now give a high-level sketch for how this decomposition of tables into an expanded shell, an expanded core, and a floating component can be used to compute $F_6(t)$. We need the following definition.

Definition 8.3.1. We say that a partial table C is compatible with a shell S if the union of S and C contains six copies of each element which is present in S or C, one for each row.

We say that a core C is a minimum compatible core for S if C is compatible with S and C does not contain any elements which are not in S.

As described in more detail in Section 9, we can determine the contribution to the exponential generating function $F_6(t)$ from all tables with a given shell S as follows:

1. Given a shell S, we compute the exponential generating function corresponding to all of the possible expanded shells S' which result in S.

2. We compute the net number of minimum compatible cores for S (see Definition 9.2.8) and use this to compute the exponential generating function corresponding to all of the possible expanded cores C' which are compatible with S.

3. The exponential generating function for the floating component can be computed using the same analysis as [BLP25].

4. Combining these exponential generating functions gives the contribution to the exponential generating function $F_6(t)$ from all tables T such that $\text{shell}(T) = S$.

We obtain $F_6(t)$ by summing this contribution over all possible shells S.

CHAPTER 9

THE SHELL GENERATING FUNCTION

Source and status. This chapter incorporates the expanded generating-function construction from the general sixth-moment paper [BLP26], including its row-split enumeration, explicit local factors, minimal-core example, factorial bookkeeping, and compatible-core recurrence.

In this chapter, we implement the high-level plan described above and obtain an exact formula for $T_6^{(r)}(t)$, and hence for $F_6(t)$.

9.1 Overview of the formula for $T_6^{(r)}(t)$

We can find $T_6^{(r)}(t)$ by summing the contributions from all of the possible shells with r marks (up to permutations of the rows, columns, and elements). Marks have weight one in this chapter; the factor m_1^r is restored after $T_6^{(r)}(t)$ has been formed. For a given shell S (note that S includes pairings for unknown 4-columns), we have the following factors:

1. In order to take permutations of the rows, columns, and elements into account while avoiding double counting, we have a factor of $\frac{6!}{|\operatorname{Aut}(S)|}$, where $\operatorname{Aut}(S)$ is the set of automorphisms of S (see Definition 9.2.2).

2. For each column $\mathbf{c}$ of S, we have a factor $G(\mathbf{c})$ (see Definition 9.2.6) which captures factors coming from the elements of $\mathbf{c}$, paths of known 4-columns and/or paths of pairs of triples of elements between pieces of $\mathbf{c}$ if $\mathbf{c}$ is split up, and chains of known 4-columns attached to pairs of elements of $\mathbf{c}$.

3. If the shell has k more elements than columns, letting $x(t) = \frac{t}{(1-(\mu_4-3)t)^3}$, letting $\mathcal{N}^{(k)}(t) = \frac{d^k\mathcal{N}(t)}{dt^k}$ be the kth derivative of $\mathcal{N}(t)$, and letting $W_{\text{core}}(S)$ be the total weight of the minimal cores compatible with S (see Definition 9.2.16), we have a factor

of

$$x(t)^k \mathcal{N}^{(k)}(x(t)) \cdot \frac{W_{\text{core}}(S)}{\prod_{j=1}^{k} j(j+2)(j+4)}$$

to account for the contribution from the possible cores and chains of known 4-columns attached to the cores.

4. We have a factor of $\frac{\mathcal{O}(t)}{\mathcal{N}(x(t))}$ to account for the floating component which consists of known 6-columns, cycles of known 4-columns, and cycles of columns with two unmarked triples of elements.

5. In order to ensure that we count tables with the correct sign, we have a sign $\text{sgn}(S)$ for S which we obtain by comparing S to a canonical table for S (see Definition 9.2.11).

Thus, the total contribution from shells which are isomorphic to S is

$$\text{sgn}(S) \frac{6!}{|\operatorname{Aut}(S)|} \cdot \frac{\mathcal{O}(t)}{\mathcal{N}(x(t))} \cdot \left(\prod_{\text{columns } c \text{ of } S} G(\mathbf{c}) \right) \cdot \frac{x(t)^k \mathcal{N}^{(k)}(x(t)) W_{\text{core}}(S)}{\prod_{j=1}^{k} j(j+2)(j+4)}.$$

9.2 Precise description of the formula

We now give a more precise description of the terms in our formula.

9.2.1 Shell automorphisms

We start by defining shell automorphisms.

Definition 9.2.1. We define the following sets of permutations on $6 \times j$ tables T.

1. We define $Sym(Rows) \cong Sym(6)$ to be the group of permutations of the rows.

2. We define $Sym(Columns) \cong Sym(j)$ to be the group of permutations of the columns.

3. We define $Sym(Elements)$ to be the group of permutations of the elements of the table.

Note that $Sym(Rows) \cong Sym(6)$ and $Sym(Columns) \cong Sym(j)$. If the table T contains j elements than $Sym(Elements) \cong Sym(j)$, but we will also consider shells which have more

elements than columns.

Definition 9.2.2 (Shell automorphisms)**.** Given a shell S, we define the automorphism group Aut(S) of S to be the subgroup of $Sym(Rows) \times Sym(Columns) \times Sym(Elements)$ which preserves S. We define the orbit $orbit(S)$ of S to be the set of shells which can be obtained from S by applying an element of $Sym(Rows) \times Sym(Columns) \times Sym(Elements)$ to S.

9.2.2 Row splits of a column **c**

Before describing the column factor $G(\mathbf{c})$, we need to define the set $Splits(c)$ of possible sets of row splits a column can have.

Definition 9.2.3 (Row splits)**.** Given a column **c** of a shell S, we define $Splits(c)$ to be the set of possible sets RS of partitions (R_1, R_2) of the rows of **c** which have the following properties:

1. Either $|R_1| = 4$ and $|R_2| = 2$ (in which case we call (R_1, R_2) a 4-2 split) or $|R_1| = 3$ and $|R_2| = 3$ (in which case we call (R_1, R_2) a 3-3 split).
2. For each 4-2 split $(R_1, R_2) \in RS$, **c** has an $\times$ in the rows in R_2.
3. For each 3-3 split $(R_1, R_2) \in RS$, one of the following three cases holds for both R_1 and R_2:
    1. **c** contains an unmarked triple in these rows.
    2. **c** contains an $\times$ and an unmarked pair in these rows.
    3. **c** contains three $\times$'s in these rows.
4. For any distinct pair of partitions $(R_1, R_2), (R_1', R_2') \in RS$, either $R_2' \subseteq R_1$ or $R_2' \subseteq R_2$.

Example 9.2.4. If **c** is a column which has marks in rows 1, 2, and 3 and a triple in rows 4, 5, and 6, then there are eight possible sets RS of partitions:

1. $RS = \varnothing$.

2. $RS = \{(\{1,2,3\},\{4,5,6\})\}$.

3.

$$RS = \{(\{1,2\},\{3,4,5,6\})\},$$
$$RS = \{(\{1,3\},\{2,4,5,6\})\}, \quad \text{and}$$
$$RS = \{(\{2,3\},\{1,4,5,6\})\}.$$

4.

$$RS = \{(\{1,2\},\{3,4,5,6\}),(\{1,2,3\},\{4,5,6\})\},$$
$$RS = \{(\{1,3\},\{2,4,5,6\}),(\{1,2,3\},\{4,5,6\})\}, \quad \text{and}$$
$$RS = \{(\{2,3\},\{1,4,5,6\}),(\{1,2,3\},\{4,5,6\})\}.$$

9.2.3 Column factors $G(\mathbf{c})$

We now describe the column factor $G(\mathbf{c})$ for each column $\mathbf{c}$ of the shell. Before giving the definition of $G(\mathbf{c})$, we describe the various factors in $G(\mathbf{c})$.

For each column $\mathbf{c}$ of the shell, we sum over the possible sets of row splits $RS \in Splits(c)$. Unless the column has five unmarked copies of an element (we handle this case separately), we have the following factors:

1. We have a factor of t from the column itself.

2. The factors m_1 from the marks have already been extracted from $T_6^{(r)}(t)$.

3. We have factors from the unmarked elements in a column. If a column has a block of size 4, we have a factor of $(\mu_4 - 3)$. If a column has unmarked triple(s) of elements, we have a factor of μ_3 for each such triple.

4. For each 4-2 row split in RS, we have a factor of $\frac{(\mu_4-3)t}{1-(\mu_4-3)t}$ to account for the fact that there is a path of known 4-columns of length at least 1 between the split columns.

5. If there is a 3-3 row split in RS, we have a factor of $\frac{-\mu_3^2 t}{1+\mu_3^2 t}$ to account for the fact that there is a path of pairs of triples of an element of length at least 1 between the split columns.

6. For each unmarked pair of elements, we have a factor of $\frac{1}{1-(\mu_4-3)t}$ to account for the fact that a chain of known 4-columns may be attached to this pair.

For columns which have five unmarked copies of an element and one marked copy of this element, we have a factor of

$$G_5(t) = \frac{\mu_5 t}{10} + \frac{-\mu_3^3 t^2}{\left(1+\mu_3^2 t\right)\left(1-(\mu_4-3)t\right)} + \frac{\mu_3(\mu_4-3)t^2}{1-(\mu_4-3)t} \tag{9.1}$$

to account for the following observations:

1. If there is no 3-3 split or chain of known 4-columns attached to the given pair, we should obtain a total of $\mu_5 t$. This is divided by 10, as there are 10 choices for which pair of the 5 unmarked copies of the element in the column is given to us.

2. If there is a 3-3 split, this gives a factor of $\frac{-\mu_3^2 t}{1+\mu_3^2 t}$. In this case, a chain of known 4-columns may be attached to the given pair, which gives us a factor of $\frac{1}{1-(\mu_4-3)t}$. Finally, we have a factor of $\mu_3 t$ from the column itself.

3. The contribution from the terms where there is no 3-3 split and there is a chain of known 4-columns of length at least 1 attached to the given pair is $\frac{\mu_3(\mu_4-3)t^2}{1-(\mu_4-3)t}$.

Based on this, we define $G(\mathbf{c})$ as follows.

Definition 9.2.5. 1. Given an $RS \in Splits(c)$, we define $Num_{4,2}(RS)$ to be the number of 4-2 splits in RS (which will be 0, 1, 2, or 3), and we define $Num_{3,3}(RS)$ to be the number of 3-3 splits in RS (which will be 0 or 1).

2. We define $1_{\text{known } 4}(c)$ to be 1 if $\mathbf{c}$ is a known 4-column and 0 otherwise.

3. We define $Num_{triples}(c)$ to be the number of unmarked triples $\mathbf{c}$ contains.

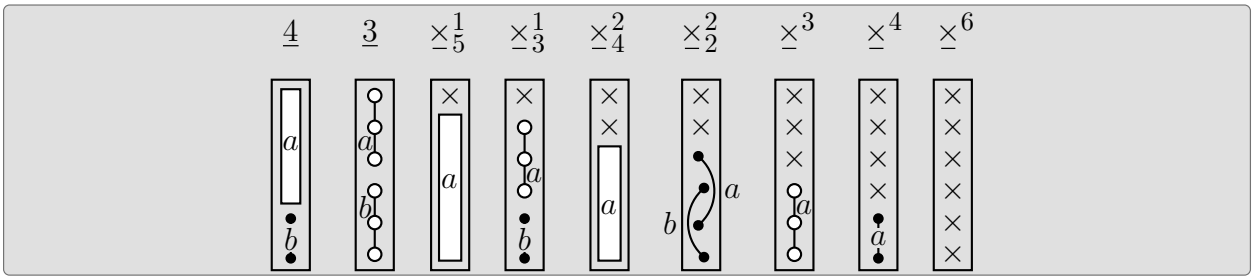
4. We define $Num_{pairs}(c)$ to be the number of unmarked pairs $\mathbf{c}$ contains.

Definition 9.2.6 (Column factors)**.** Given a column $\mathbf{c}$ of S, unless $\mathbf{c}$ is a column with a block of size 5 of an element and one marked copy of this element, we define

$$G(\mathbf{c}) = t(\mu_4 - 3)^{1_{\text{known } 4}(c)} \mu_3^{Num_{triples}(c)} \left(\frac{1}{1-(\mu_4-3)t}\right)^{Num_{pairs}(c)} \sum_{RS \in Splits(c)} \left(\frac{(\mu_4-3)t}{1-(\mu_4-3)t}\right)^{Num_{4,2}(RS)} \left(\frac{-\mu_3^2 t}{1+\mu_3^2 t}\right)^{Num_{3,3}(RS)}. \quad (9.2)$$

If $\mathbf{c}$ is a column with a block of size 5 of an element and one marked copy of this element, we define $G(\mathbf{c}) = G_5(t)$.

We recall the different types of columns in the shell.



Lemma 9.2.7. *The column factors for the various types of columns are as follows:*

1. $\times^1_{\underline{5}}$: $G(\mathbf{c}) = G_5(t)$.

2. $\times^1_{\underline{3}}$: $G(\mathbf{c}) = \frac{\mu_3 t}{(1-(\mu_4-3)t)(1+\mu_3^2 t)}$.

3. $\underline{4}$: $G(\mathbf{c}) = \frac{(\mu_4-3)t}{1-(\mu_4-3)t}$.

4. $\times^2_{\underline{4}}$: $G(\mathbf{c}) = \frac{(\mu_4-3)t}{1-(\mu_4-3)t}$.

5. $\underline{\times}_2^2$:

$$G(\mathbf{c}) = t\left(\frac{1-\mu_3^2 t+2\mu_3^2(\mu_4-3)t^2}{(1-(\mu_4-3)t)^3(1+\mu_3^2 t)}\right).$$

6. $\underline{3}$: $G(\mathbf{c}) = \frac{\mu_3^2 t}{1+\mu_3^2 t}$.

7. $\underline{\times}^3$:

$$G(\mathbf{c}) = \frac{\mu_3(1+2(\mu_4-3)t)t}{(1-(\mu_4-3)t)(1+\mu_3^2 t)}.$$

8. $\underline{\times}^4$:

$$G(\mathbf{c}) = t\left(\frac{1+4(\mu_4-3)t-2(\mu_4-3)^2t^2-3\mu_3^2 t+6\mu_3^2(\mu_4-3)^2t^3}{(1-(\mu_4-3)t)^3(1+\mu_3^2 t)}\right).$$

9. $\underline{\times}^6$:

$$G(\mathbf{c}) = t\Bigg(\frac{1+12(\mu_4-3)t+18(\mu_4-3)^2t^2-16(\mu_4-3)^3t^3}{(1-(\mu_4-3)t)^3(1+\mu_3^2 t)} + \frac{-9\mu_3^2 t-18\mu_3^2(\mu_4-3)t^2+18\mu_3^2(\mu_4-3)^2t^3+24\mu_3^2(\mu_4-3)^3t^4}{(1-(\mu_4-3)t)^3(1+\mu_3^2 t)}\Bigg).$$

Proof. For the $\underline{\times}_3^1$ column shown, there are two possible sets RS of partitions:

1. $RS = \varnothing$.

2. $RS = \{(\{1,5,6\},\{2,3,4\})\}$.

Thus,

$$\sum_{RS\in Splits(c)} \left(\frac{(\mu_4-3)t}{1-(\mu_4-3)t}\right)^{Num_{4,2}(RS)} \left(\frac{-\mu_3^2 t}{1+\mu_3^2 t}\right)^{Num_{3,3}(RS)}$$
$$= 1+\left(\frac{-\mu_3^2 t}{1+\mu_3^2 t}\right) = \frac{1}{1+\mu_3^2 t}.$$

Multiplying this by $\mu_3 t\left(\frac{1}{1-(\mu_4-3)}\right)$ gives

$$G(\mathbf{c}) = \frac{\mu_3 t}{(1-(\mu_4-3))(1+\mu_3^2 t)}.$$

For the $\underline{4}$ column shown, the only possible set RS of partitions is $RS = \varnothing$, so

$$G(\mathbf{c}) = \frac{(\mu_4-3)t}{1-(\mu_4-3)t}.$$

For the $\times_{\underline{4}}^2$ column shown, there are two possible sets RS of partitions:

1. $RS = \varnothing$.
2. $RS = \{(\{1,2\},\{3,4,5,6\})\}$.

Thus,

$$\sum_{RS \in Splits(c)} \left(\frac{(\mu_4-3)t}{1-(\mu_4-3)t}\right)^{Num_{4,2}(RS)} \left(\frac{-\mu_3^2 t}{1+\mu_3^2 t}\right)^{Num_{3,3}(RS)}$$
$$= 1 + \left(\frac{(\mu_4-3)t}{1-(\mu_4-3)t}\right) = \frac{1}{1-(\mu_4-3)t}.$$

Multiplying this by $(\mu_4-3)t$ gives

$$G(\mathbf{c}) = \frac{(\mu_4-3)t}{1-(\mu_4-3)}.$$

For the $\times_{\underline{2}}^2$ column shown, there are four possible sets RS of partitions:

1. $RS = \varnothing$.
2. $RS = \{(\{1,2\},\{3,4,5,6\})\}$.
3. $RS = \{(\{1,3,5\},\{2,4,6\})\}$.
4. $RS = \{(\{2,4,6\},\{2,4,6\})\}$.

Thus,

$$\begin{aligned}
&\sum_{RS\in Splits(c)} \left(\frac{(\mu_4-3)t}{1-(\mu_4-3)t}\right)^{Num_{4,2}(RS)} \left(\frac{-\mu_3^2 t}{1+\mu_3^2 t}\right)^{Num_{3,3}(RS)} \\
&= 1 + \left(\frac{(\mu_4-3)t}{1-(\mu_4-3)t}\right) + 2\left(\frac{-\mu_3^2 t}{1+\mu_3^2 t}\right) \\
&= \frac{(1-(\mu_4-3)t)(1+\mu_3^2 t) + (\mu_4-3)t(1+\mu_3^2 t) - 2\mu_3^2 t(1-(\mu_4-3)t)}{(1-(\mu_4-3)t)(1+\mu_3^2 t)} \\
&= \frac{1-\mu_3^2 t + 2\mu_3^2(\mu_4-3)t^2}{(1-(\mu_4-3)t)(1+\mu_3^2 t)}.
\end{aligned}$$

Multiplying this by $t\left(\frac{1}{1-(\mu_4-3)t}\right)^2$ gives

$$G(\mathbf{c}) = t\left(\frac{1-\mu_3^2 t + 2\mu_3^2(\mu_4-3)t^2}{(1-(\mu_4-3)t)^3(1+\mu_3^2 t)}\right).$$

For the $\underline{3}$ column shown, there are two possible sets RS of partitions:

1. $RS = \varnothing$.
2. $RS = \{(\{1,5,6\},\{2,3,4\})\}$.

Thus,

$$\begin{aligned}
&\sum_{RS\in Splits(c)} \left(\frac{(\mu_4-3)t}{1-(\mu_4-3)t}\right)^{Num_{4,2}(RS)} \left(\frac{-\mu_3^2 t}{1+\mu_3^2 t}\right)^{Num_{3,3}(RS)} \\
&\quad = 1 + \left(\frac{-\mu_3^2 t}{1+\mu_3^2 t}\right) = \frac{1}{1+\mu_3^2 t}.
\end{aligned}$$

Multiplying this by $\mu_3^2 t$ gives

$$G(\mathbf{c}) = \frac{\mu_3^2 t}{1+\mu_3^2 t}.$$

As noted in Example 9.2.4, for the $\underline{\times}^3$ column shown, there are eight possible sets RS of partitions:

1. $RS = \varnothing$.

2. $RS = \{(\{1,2,3\},\{4,5,6\})\}$.

3.

$$RS = \{(\{1,2\},\{3,4,5,6\})\},$$
$$RS = \{(\{1,3\},\{2,4,5,6\})\}, \quad \text{and}$$
$$RS = \{(\{2,3\},\{1,4,5,6\})\}.$$

4.

$$RS = \{(\{1,2\},\{3,4,5,6\}),(\{1,2,3\},\{4,5,6\})\},$$
$$RS = \{(\{1,3\},\{2,4,5,6\}),(\{1,2,3\},\{4,5,6\})\}, \quad \text{and}$$
$$RS = \{(\{2,3\},\{1,4,5,6\}),(\{1,2,3\},\{4,5,6\})\}.$$

Thus,

$$\begin{aligned}
&\sum_{RS \in Splits(c)} \left(\frac{(\mu_4-3)t}{1-(\mu_4-3)t}\right)^{Num_{4,2}(RS)} \left(\frac{-\mu_3^2 t}{1+\mu_3^2 t}\right)^{Num_{3,3}(RS)} \\
&= 1 + \left(\frac{-\mu_3^2 t}{1+\mu_3^2 t}\right) + 3\left(\frac{(\mu_4-3)t}{1-(\mu_4-3)t}\right) \\
&\qquad + 3\left(\frac{(\mu_4-3)t}{1-(\mu_4-3)t}\right)\left(\frac{-\mu_3^2 t}{1+\mu_3^2 t}\right) \\
&= \frac{1+2(\mu_4-3)t}{(1-(\mu_4-3)t)(1+\mu_3^2 t)}.
\end{aligned}$$

Multiplying this by $\mu_3 t$ gives

$$G(\mathbf{c}) = \frac{\mu_3(1+2(\mu_4-3)t)t}{(1-(\mu_4-3)t)(1+\mu_3^2 t)}.$$

For the $\underline{\times}^4$ column shown, there are 26 possible sets RS of partitions:

1. $RS = \varnothing$.
2. There are 4 different ways for RS to contain a single 3-3 split.
3. There are 6 different ways for RS to contain a single 4-2 split.
4. There are 12 different ways for RS to contain a 4-2 split and a 3-3 split. For example, we can have
$$RS = \{(\{1,2\},\{3,4,5,6\}),(\{1,2,3\},\{4,5,6\})\}.$$
5. There are 3 different ways for RS to contain two 4-2 splits. For example, we can have
$$RS = \{(\{1,2\},\{3,4,5,6\}),(\{3,4\},\{1,2,5,6\})\}.$$

Thus,

$$\begin{aligned}
&\sum_{RS \in Splits(c)} \left(\frac{(\mu_4-3)t}{1-(\mu_4-3)t}\right)^{Num_{4,2}(RS)} \left(\frac{-\mu_3^2 t}{1+\mu_3^2 t}\right)^{Num_{3,3}(RS)} \\
&= 1 + 4\left(\frac{-\mu_3^2 t}{1+\mu_3^2 t}\right) + 6\left(\frac{(\mu_4-3)t}{1-(\mu_4-3)t}\right) \\
&\quad + 12\left(\frac{(\mu_4-3)t}{1-(\mu_4-3)t}\right)\left(\frac{-\mu_3^2 t}{1+\mu_3^2 t}\right) + 3\left(\frac{(\mu_4-3)t}{1-(\mu_4-3)t}\right)^2 \\
&= \frac{(1-(\mu_4-3)t)^2(1+\mu_3^2 t) - 4(1-(\mu_4-3)t)^2\mu_3^2 t + 6(\mu_4-3)t(1-(\mu_4-3)t)(1+\mu_3^2 t)}{(1-(\mu_4-3)t)^2(1+\mu_3^2 t)} \\
&\quad + \frac{-12(\mu_4-3)(1-(\mu_4-3)t)\mu_3^2 t^2 + 3(\mu_4-3)^2(1+\mu_3^2 t)t^2}{(1-(\mu_4-3)t)^2(1+\mu_3^2 t)} \\
&= \frac{1+4(\mu_4-3)t - 2(\mu_4-3)^2 t^2 - 3\mu_3^2 t + 6\mu_3^2(\mu_4-3)^2 t^3}{(1-(\mu_4-3)t)^2(1+\mu_3^2 t)}.
\end{aligned}$$

Multiplying this by $\frac{t}{1-(\mu_4-3)t}$ gives

$$G(\mathbf{c}) = t\left(\frac{1+4(\mu_4-3)t-2(\mu_4-3)^2t^2-3\mu_3^2t+6\mu_3^2(\mu_4-3)^2t^3}{(1-(\mu_4-3)t)^3(1+\mu_3^2t)}\right).$$

Finally, for an $\underline{\times}^6$ column, there are 236 possible sets RS of partitions:

1. $RS = \varnothing$.
2. There are 10 different ways for RS to contain a single 3-3 split.
3. There are 15 different ways for RS to contain a single 4-2 split.
4. There are 60 different ways for RS to contain a 4-2 split and a 3-3 split. For example, we can have
$$RS = \{(\{1,2\},\{3,4,5,6\}),(\{1,2,3\},\{4,5,6\})\}.$$
5. There are 45 different ways for RS to contain two 4-2 splits. For example, we can have
$$RS = \{(\{1,2\},\{3,4,5,6\}),(\{3,4\},\{1,2,5,6\})\}.$$
4. There are 90 different ways for RS to contain two 4-2 splits and a 3-3 split. For example, we can have
$$RS = \{(\{1,2\},\{3,4,5,6\}),(\{1,2,3\},\{4,5,6\}),(\{1,2,3,4\},\{5,6\})\}.$$
5. There are 15 different ways for RS to contain three 4-2 splits. For example, we can have
$$RS = \{(\{1,2\},\{3,4,5,6\}),(\{3,4\},\{1,2,5,6\}),(\{5,6\},\{1,2,3,4\})\}.$$

Thus,

$$\begin{aligned}
&\sum_{RS\in Splits(c)} \left(\frac{(\mu_4-3)t}{1-(\mu_4-3)t}\right)^{Num_{4,2}(RS)} \left(\frac{-\mu_3^2 t}{1+\mu_3^2 t}\right)^{Num_{3,3}(RS)} \\
&= 1 + 10\left(\frac{-\mu_3^2 t}{1+\mu_3^2 t}\right) + 15\left(\frac{(\mu_4-3)t}{1-(\mu_4-3)t}\right) \\
&\quad + 60\left(\frac{(\mu_4-3)t}{1-(\mu_4-3)t}\right)\left(\frac{-\mu_3^2 t}{1+\mu_3^2 t}\right) + 45\left(\frac{(\mu_4-3)t}{1-(\mu_4-3)t}\right)^2 \\
&\quad + 90\left(\frac{(\mu_4-3)t}{1-(\mu_4-3)t}\right)^2\left(\frac{-\mu_3^2 t}{1+\mu_3^2 t}\right) + 15\left(\frac{(\mu_4-3)t}{1-(\mu_4-3)t}\right)^3 \\
&= \frac{(1-(\mu_4-3)t)^3(1+\mu_3^2 t) - 10(1-(\mu_4-3)t)^3\mu_3^2 t + 15(\mu_4-3)t(1-(\mu_4-3)t)^2(1+\mu_3^2 t)}{(1-(\mu_4-3)t)^3(1+\mu_3^2 t)} \\
&\quad + \frac{-60(\mu_4-3)(1-(\mu_4-3)t)^2\mu_3^2 t^2 + 45(\mu_4-3)^2(1-(\mu_4-3)t)(1+\mu_3^2 t)t^2}{(1-(\mu_4-3)t)^3(1+\mu_3^2 t)} \\
&\quad + \frac{-90(\mu_4-3)^2(1-(\mu_4-3)t)\mu_3^2 t^3 + 15(\mu_4-3)^3(1+\mu_3^2 t)t^3}{(1-(\mu_4-3)t)^3(1+\mu_3^2 t)} \\
&= \frac{1+12(\mu_4-3)t+18(\mu_4-3)^2t^2-16(\mu_4-3)^3t^3}{(1-(\mu_4-3)t)^3(1+\mu_3^2 t)} \\
&\quad + \frac{-9\mu_3^2 t - 18\mu_3^2(\mu_4-3)t^2 + 18\mu_3^2(\mu_4-3)^2t^3 + 24\mu_3^2(\mu_4-3)^3t^4}{(1-(\mu_4-3)t)^3(1+\mu_3^2 t)}.
\end{aligned}$$

Multiplying this by t gives the given formula for $G(\mathbf{c})$. □

9.2.4 Counting minimal compatible cores for a shell S

Definition 9.2.8. Recall that a core C is a minimal compatible core for a shell S if

1. C does not contain any elements which are not in S.
2. If an element is contained in rows R of S, then it is contained in rows $[6] \setminus R$ of C. In other words, if we combine S and C into a table T, then each element which appears in S and/or C appears exactly once in each row of T (including appearances where it is marked).

Given a shell S, we define $MinCores(S)$ to be the set of minimal compatible cores of S.

Example 9.2.9. If S is the shell

$\times a$
$\times a$
b
b
c
c

then the minimum compatible cores for S can be obtained by starting with the tables shown below

b	c
b	c
a	c
a	c
a	b
a	b

b	c
b	c
a	c
a	c
b	a
b	a

b	c
b	c
c	a
c	a
a	b
a	b

b	c
b	c
c	a
c	a
b	a
b	a

b	c
c	b
a	c
c	a
a	b
b	a

b	c
c	b
c	a
c	a
b	a
a	b

b	c
c	b
c	a
a	c
a	b
b	a

b	c
c	b
c	a
a	c
b	a
a	b

and making the following modifications:

1. For the tables which have two 4-columns, we need to specify the pairings for these columns to obtain a core C.

2. For each of these cores, we can swap the columns of the table to obtain a different core.

Thus, there are a total of

$$2(9+9+1+9+1+1+1+1) = 64$$

minimum cores for S. However, we have to be careful as some of these minimum cores have negative sign.

In order to have the right sign for our tables, we make the following definitions.

Definition 9.2.10 (Canonical tables)**.** Given a shell S, we construct a canonical table $canon(S)$ for S as follows.

1. We assign a label to the elements of S based on the set of rows R they are contained in. If $|R| = 2$, then we assign the label R. If $|R| = 4$, then we assign the label $[6] \setminus R$. If $|R| = 6$, then we assign the label $\varnothing$. We then assign an order to the elements so that the labels of the elements are in lexicographic order. Note that while the order may not be unique, the sign of the resulting table is fixed.

2. We sort the rows so that the elements are in ascending order according to the order they were given, not their order in $[n]$. Note that this makes the canonical table invariant under permutations of $[n]$.

Given a core C, we construct a canonical table $canon(C)$ using the same procedure.

Definition 9.2.11 (Signs of shells and cores)**.** Given a shell S, we define $\mathrm{sgn}(S)$ to be 1 if it takes an even number of swaps (where each swap takes two elements in one row and swaps them) to transform S into $canon(S)$, and -1 otherwise. Similarly, given a core C, we define $\mathrm{sgn}(C)$ to be 1 if it takes an even number of swaps (where each swap takes two elements in one row and swaps them) to transform C into $canon(C)$, and -1 otherwise.

Example 9.2.12. If C is the core

b	d
b	c
e	d
e	a
f	a
f	c

then the order of the elements would be b, d, c, e, a, f, as b appears in rows $1, 2$, d appears in rows $1, 3$, c appears in rows $2, 6$, e appears in rows $3, 4$, a appears in rows $4, 5$, and f appears in rows $5, 6$. Thus, $canon(C)$ is

b	d
b	c
d	e
e	a
a	f
c	f

Since it takes 3 swaps to go from C to $canon(C)$, $\text{sgn}(C) = -1$.

Remark 9.2.13. While $canon(C)$ is not a valid partial table, as a, c, d, and e do not appear as pairs, this is okay because we are only using $canon(C)$ as a tool to determine the signs of our tables.

Lemma 9.2.14. *If C is a minimum compatible core for S, then the table formed by merging $canon(S)$ and $canon(C)$ has positive sign.*

Proof. This proof uses the fragments which are described later on in Section 10.4.

Consider the fragments of S and C. $canon(S)$ orders the fragments of S by the lexicographic order of their supports. Similarly, $canon(C)$ orders the fragments of C (which are the same as the fragments of S but with opposite polarity) by the lexicographic order of their supports.

Observe that we can reach a table with positive sign by swapping the order of the fragments of S and C so that each fragment of S is adjacent to the corresponding fragment of C (i.e., the fragment with the same element and opposite polarity). We can do this by going through the fragments F of S in reverse order and swapping each fragment F with all of the fragments in C which correspond to earlier fragments of S.

When we swap the order of two fragments, this gives a factor of -1 if the row supports of the fragments have an odd number of elements in common (note that the row support is equal to the stored support for a minus fragment and the complement of the stored support for a plus fragment). Otherwise, this swap leaves the sign unchanged. Thus, the net sign for these swaps is the same as the sign for swapping each pair of fragments in S an odd number of times. This implies that the overall sign when we merge $canon(C)$ and $canon(S)$ is

$$(-1)^{\sum_{i=1}^{6} \binom{\text{number of times } i \text{ appears in the row support of a fragment } F \text{ of } S}{2}}$$
$$= (-1)^{6\binom{\#\text{ of columns of } S - \#\text{ of elements appearing 6 times in } S}{2}} = 1.$$

as for all $i \in [6]$, the number of fragments F of S such that i is in the row support of F is equal to the number of columns of S minus the number of elements which appear 6 times in S. □

Corollary 9.2.15. *If C is a minimum compatible core for S, then the table formed by merging S and C has sign* $\operatorname{sgn}(S)\operatorname{sgn}(C)$.

Definition 9.2.16. Given a shell S, we define

$$W_{\text{core}}(S) = \sum_{C \in MinCores(S)} \operatorname{sgn}(C).$$

Example 9.2.17. If S is the shell

$\times a$
$\times a$
b
b
c
c

then for each of the minimum compatible cores C for S, $canon(C)$ is

b	c
b	c
a	c
a	c
a	b
a	b

To see this, observe that the order of the elements is a, b, c, as a is missing from rows $1, 2$, b is missing from rows $3, 4$, and c is missing from rows $5, 6$.

For the tables shown in Example 9.2.9, the first four tables give minimum compatible cores with positive sign while the last four tables give minimum compatible cores with negative sign. Thus,

$$W_{\text{core}}(S) = 2(9 + 9 + 1 + 9 - 1 - 1 - 1 - 1) = 48.$$

9.3 The generating function $T_6^{(r)}(t)$

We are now ready to state and prove our formula for computing $T_6^{(r)}(t)$.

Definition 9.3.1. For each shell S, we define k_S to be the number of elements in S minus the number of columns of S.

Definition 9.3.2. We define $\mathcal{S}_r$ to be the set of possible shells with exactly r marks up to permutations of the rows, columns, and elements.

Theorem 9.3.3 (Formula for $T_6^{(r)}(t)$)**.** *Recall that*

$$\mathcal{N}(t) = \frac{1}{48} \sum_{n=0}^{\infty} (n+1)(n+2)(n+4)!\, t^n$$

and

$$\mathcal{O}(t) = (1+\mu_3^2 t)^{10} \frac{\exp\big(t(\mu_6 - 15\mu_4 - 10\mu_3^2 + 30)\big)}{(1+3t-\mu_4 t)^{15}} \mathcal{N}\left(\frac{t}{(1+3t-\mu_4 t)^3}\right).$$

Letting $x(t) = \frac{t}{(1-(\mu_4-3)t)^3}$,

$$T_6^{(r)}(t) = \sum_{S \in \mathcal{S}_r} \operatorname{sgn}(S) \frac{6!}{|\operatorname{Aut}(S)|} \cdot \frac{\mathcal{O}(t)}{\mathcal{N}(x(t))} \cdot \left(\prod_{\text{columns } c \text{ of } S} G(\mathbf{c}) \right) \cdot \frac{x(t)^{k_S} \mathcal{N}^{(k_S)}(x(t)) W_{\text{core}}(S)}{\prod_{j=1}^{k_S} j(j+2)(j+4)}. \tag{9.3}$$

Consequently,

$$F_6(t) = \sum_{r=0}^{6} m_1^r T_6^{(r)}(t).$$

Remark 9.3.4. Note that $\operatorname{sgn}(S)$ may be affected by permutations of the rows. The reason that this is not an issue is that if $\operatorname{sgn}(S)$ is flipped by a permutation of the rows, for each minimal compatible core C for S, $\operatorname{sgn}(C)$ will also be flipped by the permutation of the rows.

Proof. We show this by handling the expanded shell, the expanded core, and the floating cycles of known 4-columns and/or columns with two unmarked triples separately. In particular, we show the following:

1. Lemma 9.4.1 shows that if a shell S has a columns and $a + k_S$ elements, then taking

$$F_S(t) = \operatorname{sgn}(S) \prod_{\text{columns } c \text{ of } S} G(\mathbf{c}),$$

if we write $F_S(t) = \sum_{x=0}^{\infty} s_x t^x$, then the total weight of the partial tables S' with x columns such that

1. The shell of S' is S, the core of S' is empty, and S' does not have a floating component. In other words, S' is a possible expanded shell for S.
2. The $x - a$ elements of S' which are not in S are labeled only by their relative

order. In particular, instead of giving the final labels of the elements which are in S' but not S, we only specify how these elements compare with each other, and we do not specify how these elements compare to the elements in S.

is $\frac{x!(x-a)!}{a!}s_x$.

2. Corollary 9.4.3 shows that for each shell S, if we take

$$C_S(t) = \frac{x(t)^{k_S}\mathcal{N}^{(k_S)}(x(t))W_{\text{core}}(S)}{\prod_{j=1}^{k_S} j(j+2)(j+4)}$$

and write $C_S(t) = \sum_{y=0}^{\infty} c_y t^y$, then the total weight of the partial tables C' with y columns such that

(a) C' is compatible with S, $shell(C')$ is empty, and C' does not have a floating component. In other words, C' is a possible expanded core for a table T such that $\text{shell}(T) = S$.

(b) The $y - k_S$ elements of C' which are not in S are labeled only by their relative order.

is $y!(y-k_S)!c_y$.

3. As noted in Section 7.2.2, if we take

$$Fl(t) = \frac{\mathcal{O}(t)}{\mathcal{N}(x(t))}$$

and write $Fl(t) = \sum_{z=0}^{\infty} fl_z t^z$, then the total weight of tables with z columns consisting only of known $\underline{6}$-columns, cycles of known $\underline{4}$-columns, and cycles of columns with two unmarked triples where the elements are labeled only by their relative order is $(z!)^2 fl_z$.

Let $n = x + y + z$. We consider a shell $S_0 \in \mathcal{S}_r$ and compute the total weight of the tables T such that $S = \text{shell}(T)$ is isomorphic to S_0, the expanded shell of T has x columns, the

expanded core of T has y columns, and the floating component of T has z columns. We can choose such a table T as follows:

1. Letting a be the number of columns of S_0, there are

$$\frac{6!(k_S+a)!a!}{\mathrm{Aut}(S_0)}$$

different shells S which are isomorphic to S_0 (where elements are labeled by their relative order), as there are 6! permutations of the rows, $(k_S+a)!$ permutations of the elements, and $a!$ permutations of the columns, but we need to divide by $\mathrm{Aut}(S_0)$ to avoid double counting.

2. We choose an expanded shell S' with x columns, an expanded core C' compatible with S, and a floating component with z columns satisfying the conditions described above. The total weight of the choices for S' is $\frac{x!(x-a)!}{a!}s_x$, the total weight of the choices for C' is $y!(y-k_S)!c_y$, and the total weight of the choices for the floating component is $(z!)^2 fl_z$.

3. There are

$$\binom{n}{k_S+a} = \frac{n!}{(k_S+a)!(n-(k_S+a))!}$$

ways to choose the actual elements of the shell S (since we have specified the relative order of the elements in S and their positions, specifying the k_S+a elements which appear in S uniquely determines S).

4. There are

$$\frac{(n-(k_S+a))!}{(x_S-a)!(y-k_S)!z!}$$

ways to order the elements which are not in S, as we have already chosen the relative order for the $x-a$ elements in S' which are not in S, the relative order for the $y-k_S$ elements of C' which are not in S, and the relative order for the z elements in the

floating component.

5. There are $\frac{n!}{x!y!z!}$ ways to complete the description of T by choosing which x columns of T are in S' and which y columns of T are in S' (the remaining z columns are the floating component).

Combining these factors, the total weight of all such tables T is

$$\begin{aligned}&\frac{6!(k_S+a)!a!}{\mathrm{Aut}(S_0)}\cdot\frac{x!(x-a)!}{a!}s_x\cdot y!(y-k_S)!c_y\cdot(z!)^2fl_z\\&\qquad\cdot\frac{n!}{(k_S+a)!(n-(k_S+a))!}\cdot\frac{(n-(k_S+a))!}{(x-a)!(y-k_S)!z!}\cdot\frac{n!}{x!y!z!}\\&=\frac{6!}{\mathrm{Aut}(S_0)}(n!)^2s_xc_yfl_z.\end{aligned}$$

Thus, for each $S\in\mathcal{S}_r$, the contribution to $T_6^{(r)}(t)$ from tables T whose shell is isomorphic to S is

$$\frac{6!}{\mathrm{Aut}(S)}\sum_{x=0}^{\infty}\sum_{y=0}^{\infty}\sum_{z=0}^{\infty}s_Xc_yfl_zt^{x+y+z}=\frac{6!}{\mathrm{Aut}(S)}F_S(t)C_S(t)Fl(t).$$

The result follows by summing over all $S\in\mathcal{S}_r$, and multiplying the contribution from tables with r marks by m_1^r gives the stated formula for $F_6(t)$. □

9.4 Generating functions for the expanded shells and cores

We now complete the proof of Theorem 9.3.3 by computing the exponential generating functions corresponding to the expanded shell and the expanded core.

Lemma 9.4.1. *If a shell S has a columns and $a+k$ elements, then taking*

$$F_S(t)=\mathrm{sgn}(S)\prod_{\text{columns } c \text{ of } \mathrm{shell}(T)}G(\mathbf{c}),$$

if we write $F_S(t)=\sum_{x=0}^{\infty}s_xt^x$, the total weight of the expanded tables S' with x columns

such that

1. *The shell of S' is S, the core of S' is empty, and S' does not have a floating component. In other words, S' is a possible expanded shell for S.*

2. *The $x-a$ elements of S' which are not in* shell(T) *are labeled by their relative order.*

is $\frac{x!(x-a)!}{a!}s_x$.

Proof. The key idea is that whenever we add a column due to a 4-2 split, a 3-3 split, or attaching a chain to an unmarked pair, if we already have x' columns, then there are $x'+1$ places where the new column can fit in and there are $x'-a+1$ choices for where the new element is inserted in the relative order of the added elements. This gives a factor of

$$\prod_{j=a+1}^{x} j(j-a) = \frac{x!(x-a)!}{a!}.$$

We now make the following observations:

1. For each 4-2 row split, we add l columns/elements for some $l \geq 1$. If we add l elements and columns, then this gives a factor of $((\mu_4-3)t)^l$. Summing over the possible l gives a factor of
$$\sum_{l=1}^{\infty}((\mu_4-3)t)^l = \frac{(\mu_4-3)t}{1-(\mu_4-3)t}.$$

2. For each 3-3 row split, we add l columns/elements for some $l \geq 1$. If we add l elements and columns, then this gives a factor of $(-\mu_3^2 t)^l$ (we have a minus sign as it takes 3 swaps to undo each step of the split). Summing over the possible l gives a factor of
$$\sum_{l=1}^{\infty}(-\mu_3^2 t)^l = \frac{-\mu_3^2 t}{1+\mu_3^2 t}.$$

3. For each unmarked pair of elements, we attach a chain of known 4-columns of length l for some $l \geq 0$. If we attach a chain of length l, then this gives a factor of $((\mu_4-3)t)^l$.

Summing over the possible l gives a factor of

$$\sum_{l=0}^{\infty}((\mu_4-3)t)^l = \frac{1}{1-(\mu_4-3)t}.$$

The result follows by combining these factors with $\operatorname{sgn}(S)$ and the existing factors of t, μ_3, and (μ_4-3) from the columns, triples, and blocks of size 4 in S. □

Lemma 9.4.2. *For each shell S, for all $y' \geq k_S$, the total weight of the cores C with y' columns such that*

1. *C is compatible with S.*

2. *The $y'-k_S$ elements of C which are not in S are labeled only by their relative order.*

is

$$\left(\prod_{j=k+1}^{y'} j(j+2)(j+4)\right) W_{\text{core}}(S).$$

Proof. Let $W_S(y')$ be the total weight of the cores C with y' columns such C is compatible with S and the $y'-k_S$ elements of C which are not in S are labeled only by their relative order. To prove this result, it is sufficient to show that for all $y' > k_S$,

$$W_S(y') = y'(y'+2)(y'+4)W_S(y'-1).$$

We prove this using the techniques in Appendix A of Beck, Potechin, and Lv [BLP23].

Let $n'-y'-k_S$ and consider the cores C with y' columns which are compatible with S such that the elements of C which are not in S are $1, 2, \ldots, n'$. Observe that

1. The total weight of the cores C such that all six n' are in the same column is $15y'W_S(y'-1)$, as each such core C can be obtained by starting from a core C_0 with $y'-1$ columns which is compatible with S, choosing one of y' positions to insert the column with six n', and then choosing one of 15 different pairings for the six n'.

2. The total weight of the cores C such that four n' are in one column and the remaining two n' are in a different column is $9n(n-1)W_S(y'-1)$. To see this, observe that each such core can be obtained by starting from a core C_0 with $y'-1$ columns which is compatible with S, choosing one of the $3(y'-1)$ pairs of elements x in C_0, attaching a chain $n \to x$ of length 1 to this pair of elements, choosing one of y' possible locations for the column with four n', and then choosing one of 3 possible pairings for this column.

Analyzing the total weight of the cores C such that the six n' are in three different columns is trickier. One way to do this is to consider the following procedure for choosing such a core C and compare it with the following procedure for choosing a core C_2 with $y'-1$ columns which is compatible with S.

Procedure for choosing C:

1. Start with a partial table T which has all of the elements of C except for the six n' and twelve other elements.
2. Choose how to arrange these elements into three columns such that each column contains a pair of n' and the other elements are paired up as well.
3. Insert these 3 columns into T by choosing which 3 of the y' columns will be these columns. There are $\binom{y'}{3}$ choices for this.

Procedure for choosing C_2:

1. Start with a partial table T which has all but twelve of the elements of C_2.
2. Choose how to arrange these twelve elements into two columns so that these elements are paired up.
3. Choose two of the $y'-1$ positions for these columns. There are $\binom{y'-1}{2}$ choices for this.

We make the following observations about these procedures:

1. By Lemma A.2 of Beck, Potechin, and Lv [BLP23], after taking signs into account, the total weight of the choices for the second step for choosing C is 6 times the total weight

of the choices for the second step for choosing C_2.

2. For each such core C, there is a unique way to choose C via this procedure, as T must be the $y' - 3$ columns of C which do not contain n'. On the other hand, for each such core C_2, there are $\binom{y'-1}{2}$ different ways of choosing C_2, as we can take T to be any $y' - 3$ of the $y' - 1$ columns of C.

Putting these observations together, the total weight of the columns C where the six n' appear in 3 different columns is

$$6\binom{y'}{3} = y'(y'-1)(y'-2)W_S(y'-1),$$

as the total weight of the columns C_2 with $y' - 1$ columns which are compatible with S is $W_S(y' - 1)$.

Adding these contributions together, we have that

$$W_S(y') = \left(15y' + 9y'(y'-1) + y'(y'-1)(y'-2)\right) W_S(y'-1) = y'(y'+2)(y'+4)W_S(y'-1),$$

as needed. □

Corollary 9.4.3. *For each shell S, if we take*

$$C_S(t) = \frac{x(t)^{k_S}\mathcal{N}^{(k_S)}(x(t))W_{\text{core}}(S)}{\prod_{j=1}^{k_S} j(j+2)(j+4)}$$

and write $C_S(t) = \sum_{y=0}^{\infty} c_y t^y$, then the total weight of the partial tables C' with y columns such that

1. *C' is compatible with S, shell(C') is empty, and C' does not have a floating component. In other words, C' is a possible expanded core for a table T such that* $\text{shell}(T) = S$.

2. *The $y - k_S$ elements of C' which are not in S are labeled only by their relative order.*

is $y!(y-k_S)!c_y$.

Proof. We first show that if we write

$$\frac{t^{k_S}\mathcal{N}^{(k_S)}(t)W_{\text{core}}(S)}{\prod_{j=1}^{k_S} j(j+2)(j+4)} = \sum_{y'=0}^{\infty} c_{y',\text{ no chains}} t^{y'},$$

then the total weight of the cores C with y' columns which are compatible with S, where the elements of C which are not in S are labeled only by their relative order, is

$$y'!(y'-k_S)!c_{y',\text{ no chains}}.$$

To see this, recall that

$$\mathcal{N}(t) = \sum_{y'=0}^{\infty} n_{y'} t^{y'} \quad \text{where} \quad n_{y'} = \frac{y'!(y'+2)!(y'+4)!}{(y'!)^2},$$

so

$$t^{k_S}\mathcal{N}^{k_S}(t) = \sum_{y'=0}^{\infty} \frac{y'!}{(y'-k_S)!} n_{y'} t^{y'} = \sum_{y'=0}^{\infty} \frac{y'!(y'+2)!(y'+4)!}{y'!(y'-k_S)!} t^{y'},$$

and thus

$$c_{y',\text{ no chains}} = \frac{y'!(y'+2)!(y'+4)!}{y'!(y'-k_S)!} \cdot \frac{W_{\text{core}}(S)}{\prod_{j=1}^{k_S} j(j+2)(j+4)}.$$

By Lemma 9.4.2, the total weight of the cores C with y' columns which are compatible with S, where the elements of C which are not in S are labeled only by their relative order, is

$$\begin{aligned}\left(\prod_{j=k+1}^{y'} j(j+2)(j+4)\right) W_{\text{core}}(S) &= \frac{y'!(y'+2)!(y'+4)!}{\prod_{j=1}^{k_S} j(j+2)(j+4)} W_{\text{core}}(S) \\ &= y'!(y'-k_S)!c_{y',\text{ no chains}}.\end{aligned}$$

The reason why we replace t with

$$x(t) = \frac{t}{(1-(\mu_4-3)t)^3}$$

is the same reason why we have $\mathcal{N}(x(t))$ rather than $\mathcal{N}(t)$ for the case when $m_1 = 0$. For each column in the core, each of the three pairs in the column may have a chain of known 4-columns leading to it. Thus, for each t, we have a factor of

$$\frac{1}{(1-(\mu_4-3)t)^3},$$

which corresponds to replacing t by $x(t) = \frac{t}{(1-(\mu_4-3)t)^3}$.

Since our setting is more subtle, we give a more careful verification that replacing t with $x(t)$ is correct. To do this, we show that the total weight of the expanded cores C' such that C' is compatible with S, C' has y columns, and $core(C') = C$ has y' columns is the same as the coefficient of t^y in

$$y!(y-k_S)!c_{y',\text{ no chains}}x(t)^{y'}.$$

Observe that

1. The coefficient of t^y in $y!(y-k_S)!c_{y',\text{ no chains}}x(t)^{y'}$ is

$$y!(y-k_S)!(\mu_4-3)^{y-y'}\binom{3y'+(y-y')-1}{3y'-1}c_{y',\text{ no chains}},$$

as we have to choose $y-y'$ factors of $(\mu_4-3)t$ from $3y'$ factors of

$$\frac{1}{1-(\mu_4-3)t} = \sum_{j=0}^{\infty}(\mu_4-3)^j,$$

which corresponds to putting $y-y'$ unlabeled balls into $3y'$ labeled bins.

2. The total weight of the expanded cores C' such that C' is compatible with S, C' has y

columns, and $C = core(C')$ has y' columns is

$$(\mu_4 - 3)^{y-y'} \binom{3y' + (y - y') - 1}{3y' - 1} \left(\prod_{j=y'+1}^{y} j(j - k_S) \right) y'!(y' - k_S)! c_{y', \text{ no chains}}$$
$$= y!(y - k_S)!(\mu_4 - 3)^{y-y'} \binom{3y' + (y - y') - 1}{3y' - 1} c_{y', \text{ no chains}}$$

as

1. In order to add $y - y'$ known 4-columns to a core C with y' columns, we need to have a total length of $y - y'$ for the $3y'$ chains (possibly of length 0) leading to pairs of elements of C.

2. Whenever we increase the length of a chain by 1, if we currently have $j-1$ columns, then we have a factor of $j(j - k_S)$, as there are j choices for the position of the new column and there are $j - k_S$ choices for where the new element is inserted in the ordering of the added elements (i.e., the elements which are in C' but not in C).

□

CHAPTER 10

IMPLEMENTATION OF THE SHELL–CORE FRAMEWORK

When enumerating the possible shells, it is useful first to make the local placements that do not require global identities for the remaining pairs. Starting from a fixed mark-position table P, the program enumerates every admissible placement of covered triples and known four-blocks, including branches on which some covered copies remain unplaced. As described in Section 9.2.3, the pair-rooted factor $G_5(t)$ eliminates explicit covered-five-block placement without eliminating the fifth-moment term $\mu_5 t/10$; the distinguished root pair remains in the literal record. The program then partitions the remaining unmarked vacancies into anonymous pairs without giving them persistent global names. We call the resulting literal record a *paired partial shell* and denote it by $\widetilde{S}$. It records the covered labels and the row supports of the anonymous pair occurrences, but it does not yet decide which anonymous occurrences represent the same eventual shell element as one another or as a covered element.

The contributions descending from a fixed $\widetilde{S}$ separate at a natural cache boundary. The inherited factor $\mathrm{SymFac}(P, \widetilde{S})$ and the local column-factor product $\prod_c G(c)$ remain attached to the literal partial-shell record. The remaining signed branch/core aggregate is organized by the exposed boundary data—row supports, support polarities, multiplicities, and the labeled/anonymous ownership information. The mathematical interface record retains these data. The MATLAB implementation uses its canonicalized four-block support projection as the key of an *outer partial-shell dictionary*; the required row-orbit transport property is discussed in Section 10.6. On a miss, the program computes and stores the complete aggregate over all identification branches, while on a hit it reuses that aggregate with the required relative determinant sign.

To compute the stored aggregate, the program expresses the exposed occurrences of covered and anonymous elements as *fragments*, as formalized in Section 10.4. A fragment occupies either two rows or four rows and records its support, polarity, and whether it already

belongs to a named shell element. These fragments form the *interface* in the mathematical gluing sense: the shell supplies the occurrences on the recorded rows, and a compatible core supplies the complementary occurrences. Data already absorbed into the local column factors are internal to the shell and do not enter this boundary record. The program then branches over the admissible set-partition identifications of the fragment occurrences. Each branch b materializes a shell S_b: a completed six-row element is removed, while every surviving anonymous fragment receives a distinct fresh label. At that point the branch multiplicity, accumulated determinant sign, shell excess, local factors, and inherited partial-shell data are known. The remaining structural integer is $W_{\mathrm{core}}(S_b)$, the net signed number of minimal compatible cores.

A second, *inner compatible-core dictionary* avoids recomputing this integer for repeated residual fragment configurations. It is queried for each materialized branch during an outer-dictionary miss. If the inner query also misses, the exact compatible-core routine completes the residual fragment demands, consults a separate fixed lookup table of pair-demand decompositions, and stores the resulting $W_{\mathrm{core}}(S_b)$. Thus the outer dictionary reuses a complete sum over identification branches, whereas the inner dictionary reuses the minimal-core weight needed by one branch. The calculations for this project are implemented in MATLAB.

At the key level, every identification branch follows the roadmap

$$K_{\mathrm{par}}(\widetilde{S}) = K_0 \longrightarrow K_i \longrightarrow K_{\mathrm{par},b} \xrightarrow{\text{label}} K(S_b). \tag{10.1}$$

Here K_0 is the initial partial-shell interface key, K_i stands schematically for any intermediate partial-shell key along the branch, and $K_{\mathrm{par},b}$ is the residual partial-shell key after the selected identifications. The labeling arrow gives each remaining unlabeled fragment a distinct fresh label and produces the shell S_b and its shell interface key $K(S_b)$. A branch may have no intermediate key; for the unchanged identification branch, $K_{\mathrm{par},b} = K_0$ before

the fresh labels are assigned. The search also carries the accumulated branch multiplicity and parity. Intermediate keys describe a possible evaluation path, not the identity of a branch; Section 10.5 defines a branch by its set partition of the original fragment occurrences. This roadmap describes the work performed on an outer-cache miss. The residual polarity lists derived from the final shell key supply the input to the inner compatible-core lookup.

We first state the six-stage executable pipeline, including both dictionary lookups. The subsequent sections follow this pipeline in order before describing the two cache-miss calculations.

10.1 Computational pipeline and dictionary hierarchy

The computation begins with a feasible mark-position table P, specified in Section 10.2, and then follows six stages:

1. Initialize the case from its mark-position table P and record $|\operatorname{Aut}(P)|$ for the multiplicity factor;

2. Enumerate every admissible placement, compatible with P, of triples and known blocks of size 4 of covered elements, retaining branches on which some covered copies remain unplaced and emitting separate $\underline{4}$-columns in a prescribed covered-element order;

3. Pair all remaining unmarked vacancies, thereby producing the literal paired partial shells, record $n_{\underline{3}}(\widetilde{S})$, and attach the symmetry factor defined in Section 10.2.1;

4. Compute the local factor $G(c)$ of every shell column using the definitions in Section 9.2.3, extract the four-block partial-shell support key, and query the outer partial-shell dictionary;

5. On an outer miss, enumerate every admissible identification branch, give each surviving anonymous fragment a distinct fresh shell label, compute the determinant parity and excess k_S, and obtain the signed compatible-core weight from the inner dictionary; and

6. Sum the branch/core terms and store the outer aggregate, or transport the relative sign on an outer hit, before combining the reused or newly computed result with the current local and case-level factors.

The column patterns and corresponding local factors used by the implementation are collected for reference in Chapter B.

Even after organizing the computation by this pipeline and exploiting the available symmetries, the number of possible resulting shells is too large for every associated calculation to be performed independently. The executable therefore inserts an outer cache between the fourth and fifth stages.

After item 4 constructs the partial-shell support key, query the partial-shell result dictionary. On a hit, reuse the cached branch-summed compatible-core expression, transport the relative determinant sign, and bypass item 5. On a miss, perform item 5, compute the core contributions, and store the aggregate result. This hit–miss control flow is summarized in Fig. 10.1.

The inner dictionary is reached only along the outer miss branch. For each materialized shell S_b, it either returns the previously computed value of $W_{\mathrm{core}}(S_b)$ or invokes the exact compatible-core calculation described in Section 10.7 and stores the result, as displayed later in Fig. 10.2. The two dictionaries therefore remove repetition at different scales.

The first literal partial shell encountered under each canonical support key must follow the miss branch, so the dictionary avoids *re-enumerating* identifications rather than removing them from the mathematical calculation. Within that miss calculation, the inner dictionary similarly avoids reevaluating the same compatible-core weight. The local column-factor product and the later case-level normalization remain outside both lookups. The sections below describe the branch and core calculations carried out on a miss; they do not move the outer lookup after identification.

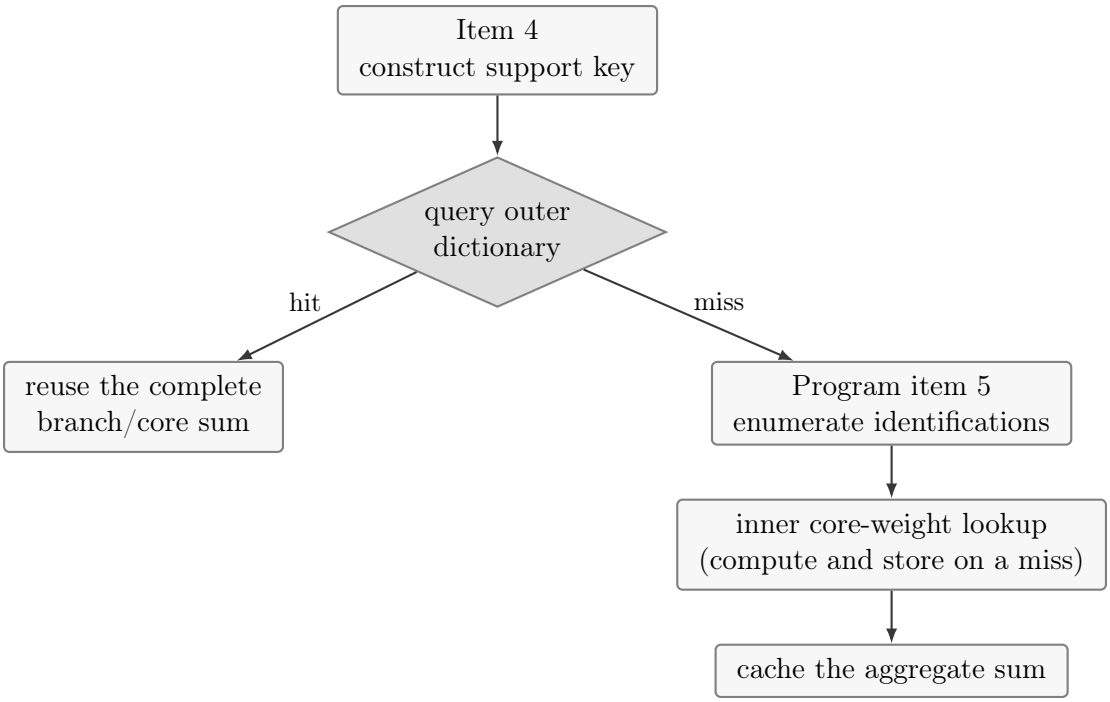


Figure 10.1: The outer partial-shell lookup; the inner compatible-core lookup is reached only on an outer miss.

10.2 Mark-position inputs, symmetry factors, and term counts

The first pipeline stage is organized by one representative mark-position table. We now specify that input record, its symmetry factor, and the number of cases for each possible number of marks in the table. This section is adapted from [BLP26, Section 4], with the symmetry normalization adjusted to match the fixed ordering of separate $\underline{4}$-columns in the MATLAB enumeration.

10.2.1 Mark-position tables and symmetry factors

Symmetry bookkeeping in this enumeration must be handled carefully. The shell formula in Theorem 9.3.3 is stated as a sum over one representative of each final-shell orbit, with factor $6!/|\operatorname{Aut}(S)|$, whereas the implementation below enumerates literal partial shells from a fixed mark-position table.

A mark-position table P is a 6-by-j array whose nonzero entries record both the locations of the marks and the equality pattern of their covered elements. Each marked column contains at least one mark and each row contains at most one mark. A shell is *consistent* with P if its marks occur in exactly these positions with exactly these equalities; P is *feasible* if at least one such shell exists. Let $\mathcal{P}$ contain one representative of each isomorphism class of feasible tables. The group $\mathrm{Aut}(P)$ is the subgroup of permutations of the rows, marked columns, and covered-element labels that preserves P.

For a literal paired partial shell $\widetilde{S}$ generated from P, let $n_{\underline{3}}(\widetilde{S})$ and $n_{\underline{4}}(\widetilde{S})$ denote the numbers of columns in its separate $\underline{3}$- and $\underline{4}$-column blocks. A triple or four-block placed inside a marked column contributes to neither count. The first count is computed from P in the enumeration convention used here and is constant across its generated descendants. The second count is a property of the generated literal partial shell, not of P alone: the mark-position table does not determine all of the four-block placements made during shell generation.

The implementation lists marked columns first, separate $\underline{3}$-columns second, and separate $\underline{4}$-columns third. The separate $\underline{4}$-columns are emitted in a prescribed covered-element order. Consequently their permutations are not generated as distinct literal records and must not be divided out a second time. The symmetry factor attached to $\widetilde{S}$ is therefore

$$\mathrm{SymFac}(P, \widetilde{S}) = \frac{6!}{|\,\mathrm{Aut}(P)|\, n_{\underline{3}}(\widetilde{S})!}. \tag{10.2}$$

The fixed $\underline{4}$-column order is part of the enumeration convention; it is not an additional factorial correction applied afterward. Uncovered pair occurrences remain anonymous. Equality within each pair is retained, while equalities between distinct occurrences are recorded by the later identification branches, so no arbitrary global pair labels are introduced.

For completeness, the reason that (10.2) replaces the final-shell factor is a short orbit–stabilizer calculation. If a generated shell S has final automorphism group $\mathrm{Aut}(S)$, then this

enumeration contains

$$\frac{|\operatorname{Aut}(P)|\, n_{\underline{3}}(\widetilde{S})!}{|\operatorname{Aut}(S)|}$$

copies of its isomorphism class. Dividing the orbit-representative factor $6!/|\operatorname{Aut}(S)|$ among these copies gives (10.2); the order $|\operatorname{Aut}(S)|$ cancels. Every identification branch descending from $\widetilde{S}$ inherits this factor, and no further automorphism factor is applied after the shell is materialized.

Examples. For $P = [a, a, a, a, 0, 0]^{\mathsf{T}}$, row permutations of the four equal marked rows and the two blank rows give $|\operatorname{Aut}(P)| = 4!\,2! = 48$; when $n_{\underline{3}}(\widetilde{S}) = 0$, the factor is $720/48 = 15$. For

$$P = \begin{bmatrix} a & 0 \\ a & 0 \\ 0 & b \\ 0 & b \\ 0 & 0 \\ 0 & 0 \end{bmatrix},$$

we have $|\operatorname{Aut}(P)| = 2!\,2!\,2!\,2! = 16$, and the illustrated descendant has $n_{\underline{3}}(\widetilde{S}) = 0$, so its factor is $720/16 = 45$. Finally, for $P = [a, b, c, d, e, f]^{\mathsf{T}}$, every row permutation can be undone by relabeling the six covered elements, so $|\operatorname{Aut}(P)| = 720$. Every completion has three separate $\underline{3}$-columns, and hence its factor is $1/3! = 1/6$. Separate $\underline{4}$-columns, when present, remain in their prescribed order and introduce no further factorial denominator. Thus two descendants with the same P and $n_{\underline{3}}$ have the same symmetry factor even if their values of $n_{\underline{4}}$ differ.

10.2.2 Input records and term counts

The theorem-level case record augments the table P defined above with its number of marks, case identifier, $|\operatorname{Aut}(P)|$, and the table-determined value of $n_{\underline{3}}$. Thus the exact symmetry factor in (10.2) can be attached at the table level, and every generated literal partial shell inherits it. The value $n_{\underline{4}}(\widetilde{S})$ may be retained as literal-shell construction metadata, since it is not fully determined by P, but it is not an input to the symmetry factor. These construction fields are not part of the outer cached expression.

The input records grouped by number of marks and summarized in Table 10.1 assign each listed term to one recorded mark-position case. These input matrices appear among the tables in the mark-position-table appendix.

Table 10.1: Numbers of recorded cases by the number of marks in the table.

Number of marks	Listed cases
1	1
2	3
3	8
4	24
5	51
6	159

With the mark-position record fixed, stages two and three generate the literal paired partial shells.

10.3 Generating paired partial shells

Throughout the implementation, rows are numbered by $1, 2, \ldots, 6$. A row pair is the sorted tuple (i, j) with $i < j$. Its complement is the four-element subset $[6] \setminus \{i, j\}$, but only the pair itself is stored.

Fix one of the mark-position records counted above. Stages two and three proceed in two passes: first enumerate the covered triples and four-blocks; then pair the vacancies. The first pass does not attempt to place all six copies of every covered element.

10.3.1 Placing covered triples and four-blocks

For every covered label a, record the rows in which the marks cover a and the complementary rows available to its unmarked copies. At this stage the generator places only triples and

known blocks of size 4 of covered elements:

1. If a is covered in two rows, retain the branch on which no covered four-block is placed and enumerate every branch on which one covered four-block occupies either a new $\underline{4}$-column or a compatible four-position vacancy of an existing column.

2. If a is covered in one or three rows, enumerate every compatible placement of one covered triple. A placement is rejected if it creates an unmarked singleton or violates a known pairing. In the one-mark case, the triple accounts for only three of the five unmarked copies, so two copies remain unresolved at this stage.

Thus some copies of a covered element may remain unplaced and later enter the anonymous-pair and identification steps. Each admissible choice produces a separate branch, including the no-four-block choice above. Branch states must be independent so that extending one placement cannot alter any of its siblings.

10.3.2 Pairing the vacancies

After the selected covered triples and four-blocks have been placed, every empty unmarked position must belong to a pair. A two-position vacancy has one pairing. Four positions $a < b < c < d$ have the three pairings

$$ab|cd, \qquad ac|bd, \qquad ad|bc. \tag{10.3}$$

No shell column has six vacancies: such a column is wholly unmarked and belongs to the Gaussian core rather than the shell. Thus the two- and four-position cases above exhaust the shell pairing step. Choices in distinct shell columns are independent, and the Cartesian product of the local pairing sets gives the partial-shell list.

Each anonymous pair is initially local to its column. It receives no global identifier; after its row support is extracted, its only retained attributes are polarity and support.

10.4 Interface keys

10.4.1 Fragments

After the local column factors have been recorded, each support piece exposed to the identification and compatible-core calculations is represented by a fragment. At the language-independent specification level, a fragment has three fields:

Field	Meaning
`support`	one of the fifteen row pairs;
`polarity`	`MINUS` for a two-row support, `PLUS` for its four-row complement;
`label`	an immutable shell-element identifier, or `None` in a partial-shell key.

The word "plus" does not mean positive determinant sign. It means that the element occupies the four rows complementary to the stored pair. Multiplicity is not a field of a fragment: every pair occurrence in a partial shell gives a separate fragment, even when several fragments have the same three field values.

A labeled fragment is written (ℓ, σ, ij); an anonymous fragment is (u, σ, ij). Anonymous names used temporarily while pairing vacancies are discarded when the initial partial-shell interface key is formed. The value `None` occurs only in partial-shell and intermediate keys; every fragment in a shell key has a label. If j distinct anonymous fragments have row-pair index ij and polarity $s \in \{-, +\}$, we use the display shorthand

$$j u_{ij}^{s}. \tag{10.4}$$

The leading coefficient j abbreviates repeated entries in the surrounding fragment multiset; it is not part of any one fragment. An implementation may run-length encode that multiset internally, but every choice that distinguishes fragments must behave exactly as the occurrence-level enumeration.

Example. Suppose the exposed support records contain a labeled occurrence of element a on rows $1, 2$, a labeled occurrence of a different element b on the complementary rows $3, 4, 5, 6$, and two distinct anonymous occurrences on rows $3, 4$. The labeled fragment records are

$$a_{12}^{-} = \texttt{Fragment}((1,2), \texttt{MINUS}, a),$$
$$b_{12}^{+} = \texttt{Fragment}((1,2), \texttt{PLUS}, b).$$

The shorthand $2u_{34}^{-}$ abbreviates the two separate records

$$\texttt{Fragment}((3,4), \texttt{MINUS}, \texttt{None}), \qquad \texttt{Fragment}((3,4), \texttt{MINUS}, \texttt{None}).$$

Thus the plus fragment stores the pair $(1, 2)$ that it omits, even though it occupies rows $3, 4, 5, 6$. The fragments a_{12}^{-} and b_{12}^{+} cannot be identified because their existing labels are distinct. If the plus fragment instead carried label a, the two records would already describe one completed label class and would be removed when the interface key is normalized. The two anonymous records remain separate candidates for later identifications; their common shorthand does not assert that they belong to the same eventual element.

10.4.2 Mathematical interface record

Literal fragment names are immaterial, but their supports, polarities, multiplicities, and labeled/anonymous status are not. The following definition packages these data for the mathematical description of a cache miss.

Definition 10.4.1 (Interface key)**.** Let T be a paired partial shell or a shell after the local factors of its columns have been recorded and after any selected identifications have been applied. First normalize its exposed support records by omitting every pre-existing or newly identified element class whose fragments together occupy all six rows. The *interface key* is

the four-block record

$$K(T) = \big(K_{\ell,-}, K_{u,-}, K_{\ell,+}, K_{u,+}\big), \tag{10.5}$$

in which

- $K_{\ell,-}$ is the multiset of indices ij for labeled fragments that occupy exactly rows i and j;
- $K_{u,-}$ is the multiset of indices ij for anonymous fragments that occupy exactly rows i and j;
- $K_{\ell,+}$ is the multiset of indices ij for labeled fragments that occupy the four complementary rows $[6] \setminus \{i, j\}$; and
- $K_{u,+}$ is the multiset of indices ij for anonymous fragments that occupy the four complementary rows $[6] \setminus \{i, j\}$.

Here ℓ and u mean labeled and unlabeled, while $-$ and $+$ encode two-row and complementary four-row support. Repeated entries are retained as separate fragment occurrences.

For a paired partial shell $\widetilde{S}$, we write $K_{\mathrm{par}}(\widetilde{S})$ and call this a *partial-shell interface key*; its fragments may be unlabeled. For a shell S, every surviving fragment belongs to a shell element and is labeled, and we call $K(S)$ a *shell interface key*. In particular, the distinction concerns the stage of the computation, not a different mathematical data type.

Proposition 10.4.2. *For all shells S, the quantity*

$$\sum_{\substack{\text{fragments } F \text{ with} \\ + \text{ polarity}}} \mathbf{1}_{\{i \in \mathrm{supp}(F)\}} - \sum_{\substack{\text{fragments } F \text{ with} \\ - \text{ polarity}}} \mathbf{1}_{\{i \in \mathrm{supp}(F)\}}$$

is the same for all row indices $i \in [6]$.

Proof. Observe that the total number of times an element appears in row i is

$$\sum_{\substack{\text{fragments } F \text{ with} \\ - \text{ polarity}}} \mathbf{1}_{\{i \in \operatorname{supp}(F)\}} + \sum_{\substack{\text{fragments } F \text{ with} \\ + \text{ polarity}}} \left(1 - \mathbf{1}_{\{i \in \operatorname{supp}(F)\}}\right).$$

Since this number is the same for all $i \in [6]$,

$$\sum_{\substack{\text{fragments } F \text{ with} \\ + \text{ polarity}}} \mathbf{1}_{\{i \in \operatorname{supp}(F)\}} - \sum_{\substack{\text{fragments } F \text{ with} \\ - \text{ polarity}}} \mathbf{1}_{\{i \in \operatorname{supp}(F)\}}$$

must be the same for all row indices $i \in [6]$. □

Consequently, when enumerating shell interface keys, it is enough to retain records that satisfy this row-balance condition.

Literal positive label values do not occur in the interface key. They remain available in the branch state so that the identification step can forbid merging distinct labeled elements. Notation such as ju^s_{ij} may compress repeated entries on the page, but the record contains j occurrences. For a shell interface key,

$$K_{u,-} = K_{u,+} = \varnothing,$$

although the empty blocks are retained so that both kinds of key use the same serialization.

For example, suppose the literal paired partial shell is

$$\widetilde{S} = \begin{pmatrix} a & \alpha \\ a & \alpha \\ b & \beta \\ b & \beta \\ b & c \\ b & c \end{pmatrix},$$

where a, b, c are labeled and α, β are anonymous. Its partial-shell interface key is

$$K_{\ell,-} = \{12, 56\}, \qquad K_{u,-} = \{12, 34\}, \qquad K_{\ell,+} = \{12\}, \qquad K_{u,+} = \varnothing.$$

The plus at 12 records the four-row occurrence of b in rows $3, 4, 5, 6$. Although the interface key suppresses the literal labels, the branch state retains them: the plus cannot be identified with the labeled minus belonging to a because a and b are distinct existing labels, but it may be identified with the anonymous minus at 12. On the unchanged identification branch, the two anonymous fragments receive distinct fresh labels, so the corresponding shell key is

$$K_{\ell,-} = \{12, 12, 34, 56\}, \qquad K_{u,-} = \varnothing, \qquad K_{\ell,+} = \{12\}, \qquad K_{u,+} = \varnothing.$$

Implementation invariant. The four blocks of a partial-shell interface key must remain semantically distinguishable. Labeled versus anonymous and plus versus minus are not cosmetic tags. A shell interface key uses the same four-block serialization with empty anonymous blocks. If a faster dictionary omits a distinction after a preprocessing stage, its key equivalence must be proved for that stage.

During a cache-miss calculation, the implementation carries literal label information internally, but the interface key suppresses positive label values and serializes only the four

support blocks. At this pre-identification point, each positive element that remains on the boundary is represented by at most one aggregate two- or four-row fragment, while every anonymous pair contributes one separate fragment. The program key retains row-pair multiplicities, labeled/anonymous status, and polarity, but it does not retain the equality partition of the positive labels.

The executable then examines all $6! = 720$ row images of this support key and uses the least serialization as its row-orbit key, denoted $\widehat{K}_{\mathrm{par}}$. The stored value is not merely a list of identification templates or a single compatible-core count. It is the complete Mathematica expression for the identification-branch and compatible-core sum computed from the first literal representative on a miss:

$$\widehat{K}_{\mathrm{par}} \longmapsto \big(\widetilde{S}_{\mathrm{ref}}, \mathcal{A}_{\mathrm{miss}}(\widetilde{S}_{\mathrm{ref}})\big). \tag{10.6}$$

The aggregate $\mathcal{A}_{\mathrm{miss}}$ is defined explicitly in Section 10.7.3.

The cache key does not include the originating mark-position table, $n_{\underline{3}}(\widetilde{S})$, the partial-shell symmetry factor, literal column incidence, the literal determinant sign, or the local column-factor product. Nor does it separately retain $n_{\underline{4}}(\widetilde{S})$, which belongs to the literal placement record and is not a denominator in the symmetry factor. The reference sign is already included in the stored expression, and a hit supplies the one relative sign described in Section 10.6. The other omitted quantities belong to the surrounding literal-shell or later case-level calculation.

The four-block support projection is the key used by the MATLAB implementation. Before any identification branches are enumerated, this key is canonicalized over the 720 row permutations and queried in the outer cache. A hit reuses the stored complete branch/core aggregate with the required relative sign, whereas a miss computes that aggregate and stores it for later reuse.

The same four-block data type represents every intermediate identification key.

10.5 Admissible identifications and shell materialization

Let $K_0 = K_{\mathrm{par}}(\widetilde{S})$ be the initial partial-shell interface key. Before materializing a shell, the program enumerates all admissible identifications of its fragments.

Implementation invariant. At no point may the implementation identify two distinct labeled shell elements. Anonymous pairs may be identified when their row supports allow it, and an anonymous fragment may be absorbed into one labeled element, but two existing labels encode distinct entries of the original permutation table.

Identification branches. Let $\mathcal{F}_0$ be the occurrence set underlying K_0, with the literal labels retained in the branch state. Regard each fragment as one initial class. An *identification branch* is an admissible set partition $\mathcal{P}$ of $\mathcal{F}_0$ that coarsens these initial classes. A singleton block leaves its fragment unchanged. Every nonsingleton block must have one of the three support patterns described below: a complementary minus–plus pair, two minuses on disjoint row pairs, or three minuses on disjoint pairs that cover all six rows. Moreover, the defined labels in one block must all be equal. Consequently, fragments already carrying distinct labels can never lie in the same block.

The residual partial-shell key is obtained from all blocks simultaneously. A complementary minus–plus block and a three-minus block form a completed element and are omitted. A two-minus block becomes one plus fragment on the complementary pair and inherits the block's label if it has one; otherwise it remains anonymous. Singleton blocks are carried forward. This set-partition description makes a branch independent of the order in which a recursive program happens to apply the displayed operations. In particular, first joining two minuses and then joining the resulting plus to the third minus is the same three-minus block, not an additional branch.

10.5.1 Complementary plus/minus identification

A minus and a plus with the same row-pair index have complementary supports:

$$(-_{ij}, +_{ij}) \longmapsto \varnothing. \tag{10.7}$$

Such a block occupies all six rows and disappears from the residual partial-shell key. Before branch enumeration begins, every plus fragment in the initial key K_0 is labeled. Indeed, every anonymous name in a freshly generated literal partial shell comes from a paired vacancy and therefore occupies exactly two rows. An anonymous plus fragment can occur only later, for example as the output of a two-minus identification in (10.8).

Consequently, a complementary minus–plus block of $\mathcal{F}_0$ is admissible only when the minus is anonymous. The plus and minus cannot both carry the same existing label: such a pair already forms a completed label class and is omitted during the normalization in Definition 10.4.1, before branch enumeration begins. A pair with distinct labels is forbidden.

10.5.2 Two-minus identification

Let $ij, k\ell, rs$ be a partition of [6] into three disjoint pairs. A block containing two minus fragments produces a plus on the complementary pair:

$$(-_{ij}, -_{k\ell}) \longmapsto +_{rs}. \tag{10.8}$$

If one input is labeled, the output inherits that label. Two distinct labeled inputs are forbidden.

10.5.3 Three-minus identification

A block containing three minus fragments on disjoint row pairs forms a complete element:

$$(-_{ij}, -_{k\ell}, -_{rs}) \longmapsto \varnothing. \tag{10.9}$$

Again, at most one distinct pre-existing label may occur among the inputs.

In the implementation, on an outer-cache miss we examine these merge choices in the fixed row-pair order. The unchanged choice is included, so the emitted terms include nonmaximal as well as maximal identifications. Mathematically, an emitted term is interpreted by the set partition of the original occurrences defined above, rather than by an ordered history of rewrites. Accordingly, until duplicate-freeness is proved, the output is treated as an indexed family of emitted branch terms. Repeated support types are handled by the exact multiplicities in the next subsection.

10.5.4 Counting repeated identification choices

Every fragment is a distinct unit entity, but several fragments may have the same support, polarity, and label status. Suppose two eligible classes contain u and v individual fragments. The number of ways to choose q disjoint pair identifications is

$$\frac{(u)_q (v)_q}{q!}. \tag{10.10}$$

Here

$$(v)_q = (v)_q := v(v-1)\cdots(v-q+1) = \frac{v!}{(v-q)!}, \qquad (v)_0 := 1,$$

is the falling factorial; it counts ordered selections of q distinct objects from v available objects. For three input classes the numerator has three falling factorials and the denominator is again $q!$. These formulas count occurrence-level blocks in the branch partition, not sequences of

elementary rewrites. Every q from zero to the smallest available class size must be included. Branches may be combined into one coefficient only when they have the same parity and the same materialized-shell contribution. Choices with different parities or different compatible-core contributions remain separate. Every branch from the same literal partial shell inherits the factor in (10.2). Taking only the maximal q is incorrect: fragments left unmerged represent distinct shell elements on another branch and may have different compatible-core completions.

Example. Consider the partial-shell interface key

$$K_{\mathrm{ex}} = \big(\varnothing, \{12, 12, 12, 34, 34\}, \varnothing, \varnothing\big), \tag{10.11}$$

which contains three distinct anonymous minuses on 12 and two distinct anonymous minuses on 34—in the shorthand of (10.4), $3u_{12}^{-}$ and $2u_{34}^{-}$. One fragment from each class may form an anonymous plus on the complementary pair 56. Grouped by their residual key, the structural templates have the counts

q	key-level count n_q	residual partial-shell key $K_{\mathrm{ex},q}$
0	1	$(\varnothing, \{12, 12, 12, 34, 34\}, \varnothing, \varnothing)$
1	$(3)_1(2)_1 = 6$	$(\varnothing, \{12, 12, 34\}, \varnothing, \{56\})$
2	$(3)_2(2)_2/2! = 6$	$(\varnothing, \{12\}, \varnothing, \{56, 56\})$

(10.12)

Thus retaining only the maximal choice $q = 2$ would lose the unchanged partition and the six occurrence-level partitions with exactly one identification. The displayed n_q records how many abstract occurrence-level templates have the same residual partial-shell key. Because the example specifies only the key and not the columns from which its occurrences arose, n_q is not yet a materialized-shell multiplicity. After the choices are applied to a literal partial shell, only those with the same parity and the same materialized-shell contribution are combined into the branch multiplicity m in (10.13). Before a residual key is sent to the core calculation,

each of its anonymous fragments receives a distinct fresh label.

Interpreting a branch as a shell. Fix a literal paired partial shell $\widetilde{S}$ and one admissible partition $\mathcal{P}_b$ of its fragment occurrences. Write $K_{\mathrm{par},b}$ for the residual partial-shell key. After applying exactly the blocks of $\mathcal{P}_b$, omit every element class completed on all six rows. Each remaining fragment inherits the label of its class if that class already has one; every remaining anonymous fragment receives its own fresh label. This produces the shell S_b and its all-labeled interface key

$$K' := K(S_b) = (K'_{\ell,-}, \varnothing, K'_{\ell,+}, \varnothing).$$

The unchanged identification branch has $K_{\mathrm{par},b} = K_0$, but its shell key $K(S_b)$ is the all-labeled materialization of K_0 and need not have the same serialization. A nonmaximal branch similarly declares the anonymous fragments that it does not identify to be distinct shell elements.

This is the mathematical interpretation used in the shell formula. The MATLAB helper does not allocate a separate all-labeled $K(S_b)$ object: after the identifications have been encoded in its branch state, it passes the remaining polarity arrays to the compatible-core routine. Giving the survivors fresh distinct labels states explicitly which ownership convention that computation represents.

A branch is summarized by its resulting shell key, multiplicity, and parity:

$$b = (K', m, \varepsilon), \tag{10.13}$$

where $K' = K(S_b)$ is the shell interface key, m the combinatorial branch multiplicity, and ε the parity change induced by the corresponding row swaps. The materialized shell is used immediately to compute its excess, sign, and compatible-core contribution. Every admissible identification choice, including the choice to make no identifications, is included in the sum.

Example. Consider the one-column partial shell

$$\widetilde{S} = \begin{pmatrix} \alpha \\ \alpha \\ \beta \\ \beta \\ a \\ a \end{pmatrix},$$

where a is labeled and α, β are anonymous. The fragment a occupies rows 56, while α and β occupy rows 12 and 34. Thus the initial partial-shell interface key is

$$K_0 = (\{56\}, \{12, 34\}, \varnothing, \varnothing).$$

These three minus fragments occupy disjoint row pairs that together cover all six rows. They may therefore be identified as one completed element, which inherits the label a. Since completed elements are omitted from an interface key, both the residual partial-shell key and the resulting shell key are

$$K' = (\varnothing, \varnothing, \varnothing, \varnothing).$$

There is one occurrence-level choice and its parity is $+1$. Thus (10.13) gives

$$b = (K', 1, +1).$$

The corresponding identification is

$$(\alpha \text{ on } 12,\ \beta \text{ on } 34,\ a \text{ on } 56) \longmapsto a.$$

For each branch, m and ε multiply the contribution computed from its materialized shell.

The quantities k_{S_b} and $W_{\mathrm{core}}(S_b)$ belong to that shell, whereas $\mathrm{SymFac}(P, \widetilde{S})$ belongs to the originating literal partial shell and is inherited unchanged by every branch. The next section describes the determinant-sign comparison used when the complete branch sum is reused from the outer dictionary.

10.6 Row-orbit cache and sign transport

10.6.1 Sign convention and cache transport

The underlying table sign is defined in (3.8). Definitions 9.2.10 and 9.2.11 then define the canonical comparison tables $\mathrm{canon}(S)$ and $\mathrm{canon}(C)$ and the relative signs $\mathrm{sgn}(S)$ and $\mathrm{sgn}(C)$. Their factorization is stated in Corollary 9.2.15.

The implementation evaluates the same signs by applying one recorded common ordering to the element names and multiplying the six row inversion signs. A common relabeling or common column permutation changes every row by the same parity and hence contributes a sixth power, so it does not change the completed-table sign. The symbols + and − on fragments remain support polarities and have no determinant-sign meaning.

For a materialized branch shell with interface key K and a compatible completed paired core C, write

$$\varepsilon(C \mid K) \in \{\pm 1\}$$

for the relative core sign in the common element-ordering convention. The literal-shell parity is denoted by $\varepsilon_{\widetilde{S}}$, while the branch parity is ε in (10.13). These signs are computed on a cache miss and are included in the branch sum.

10.6.2 Relative sign under row-orbit reuse

The outer support key is row-canonicalized before any identification branch is enumerated. Suppose a current literal partial shell $\widetilde{S}$ and the first stored literal representative $\widetilde{S}_{\mathrm{ref}}$ have

the same row-orbit key $\widehat{K}_{\mathrm{par}}$. The dictionary routine chooses a row permutation π carrying the current support key to the stored support key and compares the two literal tables in the common element-ordering convention. The data used by the lookup are

$$(\widehat{K}_{\mathrm{par}}, \rho_{\mathrm{rel}}, \pi), \qquad \rho_{\mathrm{rel}} \in \{\pm 1\}, \quad \pi \in S_6, \tag{10.14}$$

where ρ_{rel} is the relative dictionary sign computed after row transport and conversion to the stored element-ordering convention. On a cache hit, the driver defines the expression used for the current partial shell by applying this one factor to the complete cached branch/core aggregate:

$$\mathcal{A}_{\mathrm{use}}(\widetilde{S}; \widetilde{S}_{\mathrm{ref}}) := \rho_{\mathrm{rel}}\, \mathcal{A}_{\mathrm{miss}}(\widetilde{S}_{\mathrm{ref}}). \tag{10.15}$$

Thus the relative sign is part of a cache hit, not another identification branch or another coefficient. It is applied exactly once. The case calculation continues to carry its local factors and later normalization outside this reuse.

10.6.3 Exhaustive row-orbit normalization

The row group has only $6! = 720$ elements, so the executable examines the complete row orbit of each newly encountered support serialization. For each $\pi \in S_6$, it:

1. Applies π to every stored row pair;
2. Sorts the two entries of each resulting pair;
3. Retains the four labeled/anonymous and minus/plus blocks; and
4. Sorts the entries within each block and serializes the result.

The lexicographically least string is $\widehat{K}_{\mathrm{par}}$. The auxiliary map `key_dict` stores the association between an encountered support string and this row-orbit key. For the first literal partial shell under that key, the executable evaluates the complete aggregate and stores both that

value and the literal reference table, as displayed in (10.6). Later literal partial shells with the same row-orbit key use (10.15). No automorphism group of a materialized final shell is computed here. The theorem-level coefficient uses $\mathrm{SymFac}(P, \widetilde{S})$, while the MATLAB cache does not compute $\mathrm{Aut}(S_b)$.

10.6.4 Example: a negative row-transport sign

Four-mark case 17 supplies two different literal partial shells,

$$\widetilde{S}_{\mathrm{ref}} = \begin{pmatrix} -1 & 3 & -4 \\ -1 & 3 & 2 \\ 1 & 3 & -3 \\ 2 & -2 & -4 \\ 2 & -2 & 3 \\ 2 & 1 & -3 \end{pmatrix}, \qquad \widetilde{S} = \begin{pmatrix} -1 & -2 & 2 \\ -1 & 3 & -4 \\ 1 & 3 & -3 \\ 2 & -2 & 3 \\ 2 & 1 & -3 \\ 2 & 3 & -4 \end{pmatrix}. \tag{10.16}$$

Positive integers denote labeled covered elements. Negative integers are temporary names for anonymous pairs, not determinant signs. Suppressing the equality-class identifiers of labeled fragments, their support projections are

$$K_{\mathrm{par}}(\widetilde{S}_{\mathrm{ref}}) = \big(\{36\}, \{12, 14, 36, 45\}, \{13, 46\}, \varnothing\big), \tag{10.17}$$

$$K_{\mathrm{par}}(\widetilde{S}) = \big(\{35\}, \{12, 14, 26, 35\}, \{15, 23\}, \varnothing\big). \tag{10.18}$$

The regression implementation selects the row reordering

$$\pi = (2, 6, 3, 1, 4, 5), \qquad T_\pi A = A[\pi, :]. \tag{10.19}$$

Thus an occurrence on old row r moves to position $\pi^{-1}(r)$. On the three nonempty support blocks of $K_{\mathrm{par}}(\widetilde{S})$, respectively, this gives

$$35 \longmapsto 36, \qquad \{12,14,26,35\} \longmapsto \{12,14,36,45\}, \qquad \{15,23\} \longmapsto \{13,46\}.$$

Hence T_π carries the current support key to the stored reference key. This fact alone does not determine the relative sign, because each literal table is first put into its own canonical element convention.

After canonicalizing element names within the two literal tables, the comparison arrays produced by the implementation are

$$A_{\widetilde{S}} = \begin{pmatrix} -1 & -2 & 2 \\ -1 & 3 & -4 \\ 3 & -3 & 1 \\ -2 & 3 & 2 \\ 2 & -3 & 1 \\ 3 & 2 & -4 \end{pmatrix}, \qquad A_{\mathrm{ref}} = \begin{pmatrix} -1 & -4 & 3 \\ -1 & 2 & 3 \\ -3 & 1 & 3 \\ 2 & -4 & -2 \\ 2 & -2 & 3 \\ 2 & -3 & 1 \end{pmatrix}. \tag{10.20}$$

Applying π to $A_{\widetilde{S}}$ and then relabeling in the reference convention gives

$$A_{\widetilde{S}\to\mathrm{ref}} = \begin{pmatrix} -4 & 3 & -1 \\ 3 & 2 & -1 \\ 3 & -3 & 1 \\ -4 & -2 & 2 \\ -2 & 3 & 2 \\ 2 & -3 & 1 \end{pmatrix}. \tag{10.21}$$

For any six-row array B written in a fixed element-ordering convention, let

$$\chi(B) = (-1)^{\sum_{r=1}^{6} \operatorname{inv}(B_r)}$$

be the product of its six row inversion signs. For the two literal records in (10.16), this gives

$$\chi(\widetilde{S}_{\mathrm{ref}}) = (-1)^{2+1+2+3+1+3} = (-1)^{12} = +1,$$
$$\chi(\widetilde{S}) = (-1)^{1+2+2+1+3+2} = (-1)^{11} = -1.$$

Thus their displayed-array inversion parities are opposite. These are auxiliary parities of the literal partial-shell arrays, not yet signs of individual compatible cores. Once an identification branch and a compatible paired core C have been fixed, the relative core sign is computed by the same row-inversion comparison:

$$\varepsilon(C \mid K) = \chi(C)\,\chi\big(\operatorname{canon}(C)\big) = (-1)^{\sum_{r=1}^{6}(\operatorname{inv}(C_r)+\operatorname{inv}(\operatorname{canon}(C)_r))},$$

with both arrays written in the common element-ordering convention. The complete row-by-row calculation for the two literal arrays and the comparison arrays used in the cache transport is shown in Table 10.2.

Table 10.2: Row inversion calculation for the negative case-17 transport sign.

Array	Row inversion counts	Row signs	χ
$A_{\widetilde{S}}$	$(1,2,2,1,2,3)$	$(-,+,+,-,+,-)$	-1
$\widetilde{S}$	$(1,2,2,1,3,2)$	$(-,+,+,-,-,+)$	-1
$\widetilde{S}_{\mathrm{ref}}$	$(2,1,2,3,1,3)$	$(+,-,+,-,-,-)$	$+1$
A_{ref}	$(1,0,0,2,1,2)$	$(-,+,+,+,-,+)$	$+1$
$A_{\widetilde{S}\to\mathrm{ref}}$	$(1,3,2,0,1,2)$	$(-,-,+,+,-,+)$	-1

The current shell agrees in relative sign with its own comparison array, since $\chi(A_{\widetilde{S}})\chi(\widetilde{S}) = (-1)(-1) = +1$. Likewise, $\chi(A_{\mathrm{ref}})\chi(\widetilde{S}_{\mathrm{ref}}) = +1$. The transported comparison contributes the

only minus sign:

$$\chi(A_{\mathrm{ref}})\chi(A_{\widetilde{S}\to\mathrm{ref}}) = (+1)(-1) = -1.$$

Multiplying these three comparisons, the two occurrences of $\chi(A_{\mathrm{ref}})$ cancel because their square is 1. Therefore the four-factor expression used by the dictionary routine is

$$\begin{aligned}\rho_{\mathrm{rel}} &= \chi(A_{\widetilde{S}})\,\chi(\widetilde{S})\,\chi(\widetilde{S}_{\mathrm{ref}})\,\chi(A_{\widetilde{S}\to\mathrm{ref}}) \\ &= (-1)(-1)(+1)(-1) = -1.\end{aligned} \tag{10.22}$$

Because each record in (10.16) emits several identification branches and compatible cores, neither record has one standalone core sign. The corresponding statement at the level of the complete signed core calculation is nevertheless exact. A cache-free run on each record emits twenty branches and gives

$$\begin{aligned}\mathcal{A}_{\mathrm{miss}}(\widetilde{S}_{\mathrm{ref}}) &= 2R_1(t) + 64R_2(t) + 1200R_3(t) + 11520R_4(t), \\ \mathcal{A}_{\mathrm{miss}}(\widetilde{S}) &= -2R_1(t) - 64R_2(t) - 1200R_3(t) - 11520R_4(t),\end{aligned} \tag{10.23}$$

where the derivative blocks $R_k(t)$ are defined in (10.27). In particular, the single unchanged branch with $k = 4$ has net signed compatible-core weights $+11520$ and -11520, respectively. Thus the directly recomputed branch/core aggregates have opposite signs, exactly as $\rho_{\mathrm{rel}} = -1$ predicts.

The two row-isomorphic records may therefore share one outer row-orbit cache entry, but the contribution of $\widetilde{S}$ must acquire a minus sign relative to the stored aggregate.

The displayed matrices are initial literal partial-shell records, and $\widetilde{S}_{\mathrm{ref}}$ is the first cached representative rather than an asserted lexicographic minimum among literal tables. The executable applies the relative sign -1 to the complete cached branch/core aggregate; it does not repeat this comparison after each identification branch. The example therefore isolates the sign calculation performed on an outer-cache hit.

The next section describes the compatible-core calculation performed for each materialized branch on a cache miss.

10.7 Computing the signed compatible-core weight

For each emitted identification branch, it remains to compute the structural integer $W_{\mathrm{core}}(S_b)$ defined in Definition 9.2.16. At this point every surviving fragment represents a distinct shell element, as explained in Section 10.5. The implementation therefore passes only the fifteen lists of support polarities to the compatible-core calculation and introduces one fresh placeholder for each fragment. The branch multiplicity, accumulated shell sign, inherited symmetry factor, and local factors $G(c)$, as well as the derivative block, remain outside this calculation.

Before carrying out that calculation, the accelerated implementation serializes the fifteen residual polarity lists and queries the inner compatible-core dictionary. Because all surviving fragments now represent distinct elements, their temporary placeholder names are immaterial to this lookup. A hit reuses the stored net weight; a miss evaluates $W_{\mathrm{core}}(S_b)$ and stores it for later branches with the same residual support configuration. This is distinct from the outer dictionary, whose value is the complete sum over every identification branch of a partial shell.

The core-weight dictionary is the runtime cache; its miss evaluator consults a separate fixed table of pair-demand decompositions. The compatible-core machinery therefore has an offline precomputation and a per-branch evaluation stage. The precomputation tabulates how multisets of row-pair demands can be divided into paired-core columns. On a core-weight miss, the per-branch stage completes the complementary rows of the fresh placeholders, queries the fixed table, sums the signed multiplicities, and returns the value to be stored in the runtime cache.

This inner hit–miss flow is summarized in Fig. 10.2.

10.7.1 Precomputing pair-demand decompositions

A paired core column is a perfect matching of the six rows: it consists of three disjoint row pairs whose union is [6]. There are $(6-1)!! = 15$ such matching types. Denote them by

$$P_1 < P_2 < \cdots < P_{15},$$

where the three pairs within each P_j are first sorted in the fixed row-pair order and the resulting triples are then ordered lexicographically. For example,

$$P_1 = \{12, 34, 56\}, \qquad P_2 = \{12, 35, 46\}, \qquad P_3 = \{12, 36, 45\},$$
$$P_4 = \{13, 24, 56\}, \qquad \ldots, \qquad P_{15} = \{16, 25, 34\}.$$

Thus P_1, P_2, P_3 exhaust the matchings beginning with the pair 12; the lexicographic ordering then continues with the matchings beginning with 13, and so on.

In the implementation, these matching types are used to build a finite lookup table before any compatible-core weight is evaluated. Adjoin a blank symbol $P_{16} = \varnothing$, placed after the fifteen matching types. Six nested loops make the nondecreasing choices

$$1 \le i_1 \le i_2 \le \cdots \le i_6 \le 16,$$

with the all-blank choice omitted. After the blank symbols are deleted, one obtains a multiset of h matching types with $1 \le h \le 6$. The nondecreasing order ensures that every such multiset is generated exactly once.

For each choice, concatenate the three row pairs in every selected matching and sort the resulting $3h$ pair occurrences in the fixed row-pair order. The resulting untagged multiset Q is the lookup key. Let q_p be the multiplicity of $p \in \mathcal{R}_2$ in Q, and let $d = (d_1, \ldots, d_{15}) \in \mathbb{Z}_{\ge 0}^{15}$ record how many selected columns have each matching type. The entries stored under Q are

exactly the vectors d satisfying

$$q_p = \sum_{j=1}^{15} d_j \, \mathbf{1}_{\{p \in P_j\}} \quad (p \in \mathcal{R}_2), \qquad 1 \le h = \sum_{j=1}^{15} d_j \le 6. \tag{10.24}$$

Write $\mathcal{D}(Q)$ for this stored list of decomposition vectors. Every stored key necessarily satisfies $|Q| = 3h$, and every row occurs in exactly h of its pair occurrences:

$$\sum_{p \ni r} q_p = h \qquad (r \in [6]).$$

Consequently, a demand multiset failing either condition has no lookup-table entry.

Alongside each decomposition, the precomputation stores the multiplicity

$$\nu(d, Q) = \frac{h!}{\prod_{j=1}^{15} d_j!} \prod_{p \in \mathcal{R}_2} q_p!.$$

The first factor counts the distinct placements of the multiset of matching types into h ordered core-column positions. The second counts permutations among repeated occurrences of the same row-pair support. After the fresh placeholders are introduced, these become their assignments to the corresponding slots. Thus a later sign calculation may evaluate one representative of such a class and multiply by its stored exact cardinality.

Example: one support key with two decompositions. The following $h = 4$ example illustrates every step of the precomputation.

1. Two choices made by the six nondecreasing loops are

$$(1, 3, 5, 9, 16, 16) \qquad \text{and} \qquad (1, 2, 6, 8, 16, 16).$$

After the two blank entries are deleted, their matching multisets are

$$\mathcal{P}_A = \big\{\{12,34,56\},\{12,36,45\},\{13,25,46\},\{14,26,35\}\big\},$$
$$\mathcal{P}_B = \big\{\{12,34,56\},\{12,35,46\},\{13,26,45\},\{14,25,36\}\big\}.$$

2. Concatenating the pairs in either line and sorting them gives the same support multiset

$$Q = \{12,12,13,14,25,26,34,35,36,45,46,56\}.$$

Its serialized lookup key is

$$\texttt{121213142526343536454656}.$$

3. The entry under this key retains two decomposition vectors. Their nonzero coordinates are

$$d_1^{(A)} = d_3^{(A)} = d_5^{(A)} = d_9^{(A)} = 1, \qquad d_1^{(B)} = d_2^{(B)} = d_6^{(B)} = d_8^{(B)} = 1.$$

Equivalently, the stored split strings are

$$s_A = \texttt{123456123645132546142635},$$
$$s_B = \texttt{123456123546132645142536}.$$

4. In either decomposition the four matching types are distinct, so the column-placement factor is $4! = 24$. The pair 12 occurs twice in Q, while every other pair occurs once, so the repeated-support factor is $2! = 2$. Therefore

$$\nu(d^{(A)}, Q) = \nu(d^{(B)}, Q) = 24 \cdot 2 = 48.$$

A direct run of the precomputation returned the displayed lookup key, exactly the two split strings s_A, s_B, and the value 48 for each. Thus sorting the pair demands does not discard alternative ways of grouping them into core columns; it places both decompositions under one key for the signed calculation in the next subsection.

This lookup deliberately omits placeholder names; it records only which three row pairs share each core column. The all-blank case is not stored, and the caller handles the empty core separately by setting $W_{\mathrm{core}}(\varnothing) = 1$. The present table covers only $1 \le h \le 6$; a computation requiring more than six core columns would require the precomputation to be extended. The next stage introduces the fresh placeholders and evaluates the corresponding core signs.

10.7.2 The signed compatible-core weight

After an identification branch has been fixed, scan its fifteen row-pair lists in the order $12, 13, \ldots, 56$ and give every residual fragment a fresh consecutive placeholder. Literal element names are no longer needed. A plus fragment a^{+}_{ij}, which occupies the four rows complementary to ij, has the single forced core demand ij. A minus fragment a^{-}_{ij}, which occupies rows ij, must be completed on the other four rows; their three perfect matchings give the three possible pairs of core demands. Let Ω_b be the Cartesian product of these three choices over the minus fragments of branch b. Plus-fragment demands introduce no choice.

For $\omega \in \Omega_b$, concatenate all selected pair demands and sort them in the fixed row-pair order. Denote the resulting untagged multiset by Q_ω. The lookup from Section 10.7.1 returns every $d \in \mathcal{D}(Q_\omega)$ together with $\nu(d, Q_\omega)$. For each returned decomposition, the implementation constructs one representative paired core $C_{\omega,d}$: for each support p, queue the fresh placeholders whose chosen completion contains p; then traverse the pair slots of the stored column decomposition and consume the next placeholder from the corresponding queue.

Independently, the same fresh placeholders form a fixed comparison array $C_{0,b}$. Each

placeholder is placed once in every row missing from its shell fragment, with row positions filled in the fixed scan order. As in Definition 9.2.10, this array is used only for parity and need not itself display the selected pairings. With χ denoting the product of the six row inversion signs from Section 10.6, the representative relative sign is

$$\varepsilon_{\omega,d} = \chi(C_{\omega,d})\chi(C_{0,b}).$$

The compatible-core routine returns

$$W_{\mathrm{core}}(S_b) = \sum_{\omega \in \Omega_b} \sum_{d \in \mathcal{D}(Q_\omega)} \nu(d, Q_\omega)\, \varepsilon_{\omega,d}. \tag{10.25}$$

If Q_ω has no dictionary entry, that completion contributes zero. The caller assigns the empty residual record the value $W_{\mathrm{core}}(\varnothing) = 1$.

Only the representative $C_{\omega,d}$ is constructed. The factor $\nu(d, Q_\omega)$ counts the other ordered column placements and equal-support placeholder assignments represented by the same decomposition. They have the same relative sign: a common permutation of the core columns acts in all six rows, while exchanging two placeholders on the same pair support exchanges them in two rows. Both changes preserve the total parity. Thus (10.25) is the compressed evaluation of the net minimal-core weight defined in Definition 9.2.16; the routine does not enumerate the individual cores counted by $\nu(d, Q_\omega)$.

Example. Suppose a branch leaves one minus fragment α^{-}_{12} and one plus fragment b^{+}_{12}, with distinct fresh placeholders α and b. The plus fragment forces the core demand 12, while the four missing rows of α have the three pairings

$$\{34, 56\}, \qquad \{35, 46\}, \qquad \{36, 45\}.$$

The three completion choices therefore give

$$\{(b,12),(\alpha,34),(\alpha,56)\},$$

$$\{(b,12),(\alpha,35),(\alpha,46)\},$$

$$\{(b,12),(\alpha,36),(\alpha,45)\}.$$

After the placeholders are suppressed, each line is one perfect matching of the six rows. Its dictionary entry has one single-column decomposition with $\nu = 1$, and the representative sign is $+1$. Hence

$$W_{\mathrm{core}}(S_b) = 1 + 1 + 1 = 3.$$

The three paired cores have the same displayed element-name column $(b, b, \alpha, \alpha, \alpha, \alpha)^{\mathsf{T}}$, but they remain distinct because, as specified in Definition 9.2.8, the pairing data are part of the core. As a larger check, three minus fragments on $12, 34, 56$ give $W_{\mathrm{core}} = 48$, agreeing with the direct signed count of 64 minimal cores in Example 9.2.9.

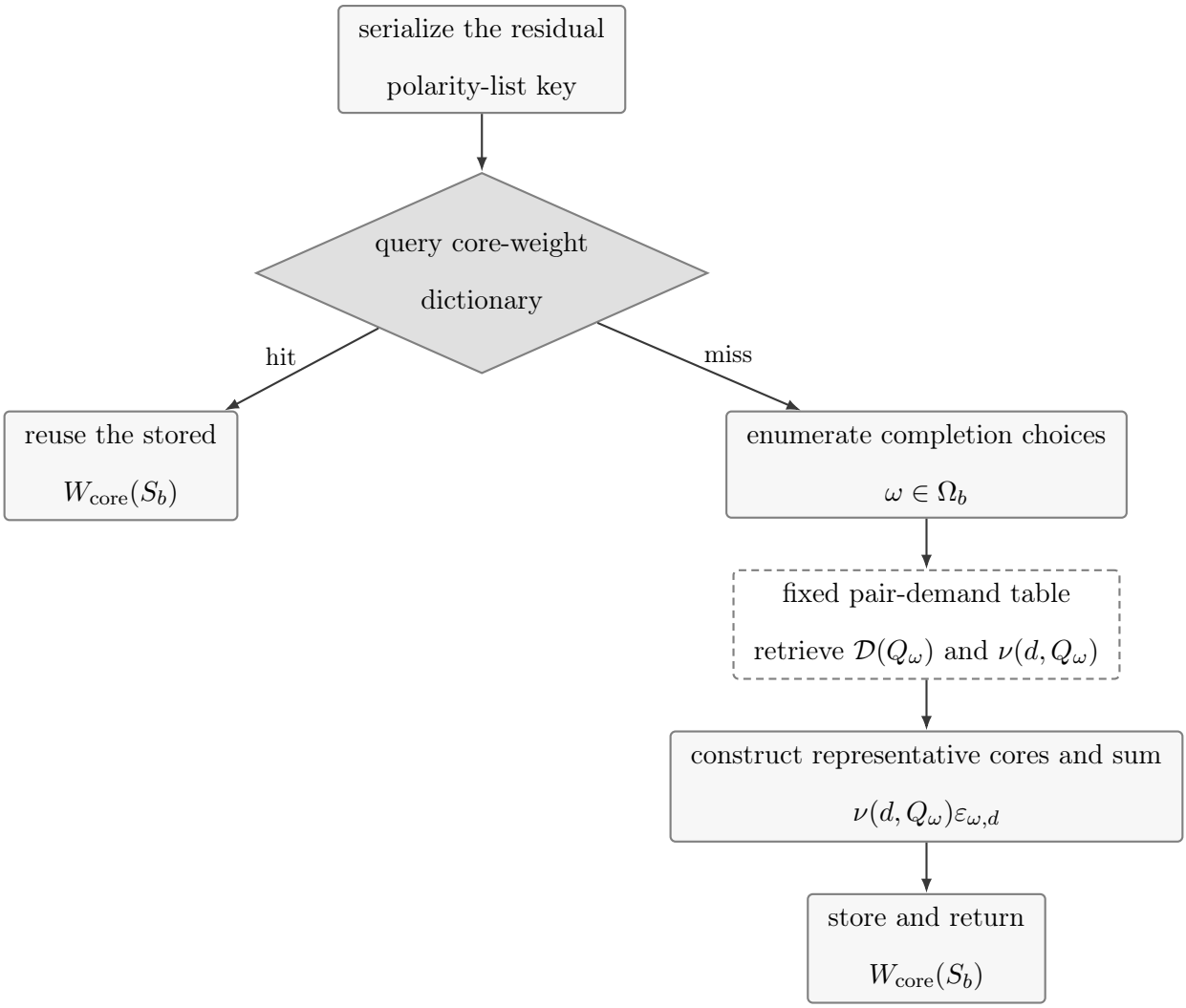


Figure 10.2: Inner core-weight dictionary lookup for a materialized shell S_b. A hit returns the stored net signed number of minimal compatible cores. On a miss, the routine queries the fixed pair-demand table, sums the signed decomposition multiplicities, and stores $W_{\mathrm{core}}(S_b)$. The empty residual record is handled separately by $W_{\mathrm{core}}(\varnothing) = 1$.

10.7.3 Assembling and caching the partial-shell expression

After the identification stage, the implementation visits each generated branch state once. Let a be the number of columns of the originating literal partial shell, and for a branch b let e_b be its current number of shell elements. The stored branch fields also contain its identification multiplicity m_b and one accumulated sign σ_b. This sign is initialized with the

literal shell sign and is updated by the signs of the selected identification operations. Thus the implementation carries their product rather than two separate sign objects. The derivative order is

$$k_b = e_b - a. \tag{10.26}$$

This is the shell excess from Definition 9.3.1.

For each branch, the program discards the fragment-label column and passes its fifteen polarity lists to the compatible-core routine. If every list is empty, it sets the core weight to 1; otherwise it queries the inner compatible-core dictionary. A cache miss evaluates $W_{\mathrm{core}}(S_b)$ by (10.25) and stores the result, whereas a hit reuses that exact integer. Write

$$R_k(t) = \frac{\mathcal{O}(t)}{\mathcal{N}(x(t))} \frac{x(t)^k \mathcal{N}^{(k)}(x(t))}{\displaystyle\prod_{j=1}^{k} j(j+2)(j+4)}, \qquad R_0(t) = \mathcal{O}(t), \tag{10.27}$$

where the product is 1 when $k = 0$. This is the floating-factor and derivative block in the shell formula of Theorem 9.3.3 and Lemma 9.4.2. The program appends the branch term

$$m_b \, \sigma_b \, W_{\mathrm{core}}(S_b) \, R_{k_b}(t)$$

to the partial-shell expression. Hence the complete value computed on an outer-cache miss is

$$\mathcal{A}_{\mathrm{miss}}(\widetilde{S}) = \sum_{b \in \mathcal{B}(\widetilde{S})} m_b \, \sigma_b \, W_{\mathrm{core}}(S_b) \, R_{k_b}(t). \tag{10.28}$$

Here $\mathcal{B}(\widetilde{S})$ is the list of branch states produced by the identification loops. Thus the returned parenthesized expression already contains every branch multiplicity, accumulated sign, signed core weight, and excess-dependent derivative block.

On the first occurrence of a row-orbit support key, the caller stores this complete expression

together with the corresponding literal partial shell. On a later occurrence it bypasses the identification and core routines, retrieves the stored expression, and applies the single relative sign from (10.15). The current product of local column factors $\prod_c G(c)$ is attached afterward, and repeated output records with the same constructed key are grouped by an integer multiplicity. The mark-position and partial-shell symmetry factor from Section 10.2.1, which realizes the orbit normalization in Theorem 9.3.3, belongs to the later aggregation over cases with the same number of marks and is not part of the cached compatible-core expression.

CHAPTER 11

SYMMETRIC, WIGNER, AND HERMITIAN ENSEMBLES

Source and status. This chapter is adapted from Beck, Potechin, and Lv [BLP24].

The asymmetric i.i.d. model has two independent label sets: matrix rows and matrix columns. Symmetry identifies X_{ij} with X_{ji}, so independence no longer factors one permutation-table column at a time. The correct replacement is a colored directed multigraph whose vertices are matrix indices and whose edges record permutation arrows. Equal random variables correspond to an undirected edge type rather than to entries in a single table column.

This change of representation does not change the analytic-combinatorics principle developed in Chapter 4. Once the connected multigraph components have been classified, sets, cycles, chains, and marked components translate into products and exponentials of generating functions. The structured-ensemble normalization uses one label set:

$$F_k^{\mathrm{str}}(t) = \sum_{n\geq 0} \frac{\mathbb{E}[\det(X_n)^k]}{n!} t^n, \tag{11.1}$$

rather than the $(n!)^{-2}$ normalization used for asymmetric determinant moments.

11.1 Symmetric matrices

Let $X = X^{\mathsf{T}}$ be an $n \times n$ random matrix for which $\{X_{ij} : 1 \leq i \leq j \leq n\}$ are i.i.d., and write

$$m_q = \mathbb{E}[X_{ij}^q], \qquad f_k(n) = \mathbb{E}[\det(X)^k], \qquad F_k(t) = \sum_{n\geq 0} f_k(n)\frac{t^n}{n!}.$$

Thus the notation m_q initially denotes raw moments. After centralization below, μ_q denotes the corresponding central moments, consistently with Chapter 3.

In this section, we analyze the second moment of the determinant of symmetric matrices.

We start by showing how the problem can be reduced to counting weighted permutation tables and how these permutation tables can be represented using multigraphs. We then illustrate these techniques by computing the first moment of the determinant of a random symmetric matrix.

To analyze the second moment, we observe that the determinant of $X = Y + m_1 J$ is linear in m_1. Representing each factor of m_1 by a marked edge, this means that it is sufficient to consider marked multigraphs which have at most two marked edges. All such marked multigraphs can be split into components of a few different types which allows us to determine the second moment of the determinant of a random symmetric matrix for arbitrary m_1.

11.1.1 Permutation tables

By definition, we have

$$f_k(n) = \mathbb{E}\left[\prod_{j=1}^{k} \sum_{\pi_j \in S_n} \operatorname{sgn}(\pi_j) \prod_{i=1}^{n} X_{i\pi_j(i)}\right] = \sum_{t \in F_{k,n}} w(t) \operatorname{sgn}(t), \tag{11.2}$$

where

1. $F_{k,n}$ is the set of k by n tables whose rows are permutations $\pi_j \in S_n$ (where S_n is the set of all permutations of $[n]$).
2. $\operatorname{sgn}(t) = \prod_{j=1}^{k} \operatorname{sgn}(\pi_j)$. In other words, the sign of t is the product of the signs of the permutations corresponding to the rows of t.
3. The weight $w(t)$ of t is $w(t) = \mathbb{E}\left[\prod_{j=1}^{k} \prod_{i=1}^{n} X_{i\pi_j(i)}\right]$. In other words, $w(t)$ is the expected value of the product of the entries of X corresponding to the entries of t.

Note that, unlike in the asymmetric case, the weight $w(t)$ cannot be expressed simply as the product of the weights of the individual columns of t. Consider the example in Figure 11.1 – to keep track of the first indices of the variables X_{ij}, we have added the identity permutation

as the *zeroth* row as a reference.

1	2	3	4	5	6	7	8	9
4	9	1	7	5	6	8	3	2
1	9	8	7	6	3	4	5	2

Figure 11.1: An example of a table $t \in F_{2,9}$ with weight $w(t) = m_1^9 m_2 m_3 m_4$

Ignoring the sign, the corresponding term is

$$X_{14}X_{29}X_{31}X_{47}X_{55}X_{66}X_{78}X_{83}X_{92}X_{11}X_{29}X_{38}X_{47}X_{56}X_{63}X_{74}X_{85}X_{92}. \tag{11.3}$$

However, by symmetry, this is equal to

$$X_{11}X_{13}X_{14}X_{29}^4X_{36}X_{38}^2X_{47}^3X_{55}X_{56}X_{58}X_{66}X_{78}. \tag{11.4}$$

Thus, we have that $w(t) = m_1^9 m_2 m_3 m_4$.

11.1.2 Multigraphs

As observed in the author's Master's thesis [Lv23], we can represent tables $t \in F_{k,n}$ by multigraphs with colored edges. That work also rederived $F_2(t)$ for the case $m_1 = 0$.

Definition 11.1.1. Let $\mathcal{S}_n$ be the set of all directed graphs on n vertices (where loops are allowed) where each vertex has indegree and outdegree 1. In other words, $\mathcal{S}_n$ is the set of graphs on n vertices which are disjoint unions of cycles. Given a permutation $\pi \in S_n$, we define the corresponding graph $G_\pi \in \mathcal{S}_n$ to be the graph with vertices $V(G_\pi) = [n]$ and edges $E(G_\pi) = \{(i, \pi(i)) : i \in [n]\}$.

Definition 11.1.2. We define $\mathcal{F}_{k,n}$ to be the set of all k-tuples of graphs in $\mathcal{S}_n$. Given $t \in F_{k,n}$, we take the corresponding element $\widehat{t} \in \mathcal{F}_{k,n}$ to be $\widehat{t} = (G_{\pi_1}, \ldots, G_{\pi_k})$ where

$\pi_1, \ldots, \pi_k$ are the permutations corresponding to the rows of t. Visually, we represent $\widehat{t}$ as a multigraph with edges $E(\widehat{t}) = \cup_{j=1}^{k} E(G_{\pi_j})$ where the edges in $E(G_{\pi_j})$ have color j. Since we only consider $k = 1$ and $k = 2$ in this chapter, we represent the edges in G_{π_1} with solid lines and we represent the edges in G_{π_2} with dashed lines.

It is not hard to check that there is a bijection between $F_{k,n}$ and $\mathcal{F}_{k,n}$. This bijection allows us to work with $\mathcal{F}_{k,n}$ rather than $F_{k,n}$.

Definition 11.1.3. Given $t \in F_{k,n}$,

1. We define $\operatorname{sgn}(\widehat{t}) = \operatorname{sgn}(t)$. Note that

$$\operatorname{sgn}(\widehat{t}) = \prod_{j=1}^{k} (-1)^{C_E(\pi_j)} = \prod_{j=1}^{k} (-1)^{n - C(\pi_j)} = (-1)^{kn - C(\widehat{t})}$$

   where $C_E(\pi_j)$ is the number of cycles of π_j of even length, $C(\pi_j)$ is the total number of cycles of π_j, and $C(\widehat{t})$ is the total number of cycles in $\widehat{t}$ such that all edges have the same color.

2. We define $w(\widehat{t}) = w(t) = \mathbb{E}\left[\prod_{(i,j)\in E(\widehat{t})} X_{ij}\right]$. Equivalently, $w(\widehat{t})$ is the product of $m_{e_{ij}}$ over all pairs of vertices $i \geq j \in [n]$ where e_{ij} is the number of edges between i and j (in both directions).

Example 11.1.4. The multigraph $\widehat{t}$ for the table t from Figure 11.1 is as follows.

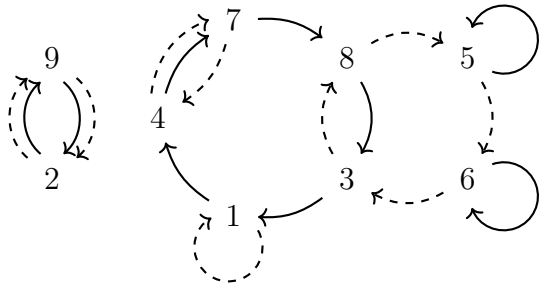


Figure 11.2: A multigraph $\widehat{t} \in \mathcal{F}_{2,9}$ with weight $w(\widehat{t}) = m_1^9 m_2 m_3 m_4$

Proposition 11.1.5. *For any distribution of X_{ij}, we have*

$$f_k(n) = \sum_{\widehat{t} \in \mathcal{F}_{k,n}} \operatorname{sgn}(\widehat{t}) w(\widehat{t}). \tag{11.5}$$

and

$$F_k(t) = \sum_{n=0}^{\infty} \sum_{\widehat{t} \in \mathcal{F}_{k,n}} \frac{\operatorname{sgn}(\widehat{t}) w(\widehat{t}) t^n}{n!}. \tag{11.6}$$

To analyze the exponential generating function $F_k(t)$, it is convenient to consider $\mathcal{F}_{k,n}$ for all n simultaneously so we make the following definition.

Definition 11.1.6. We define $\mathcal{F}_k = \bigcup_{n=0}^{\infty} \mathcal{F}_{k,n}$.

11.1.2.1 The first moment of the determinant of a random symmetric matrix

As an example, we find $F_1(t)$ (and $f_1(n)$). Here, the structure of all graphs in $\mathcal{F}_1$ is simple (a permutation graph consists of disjoint cycles). See Figure 11.3 for an example of such a graph.

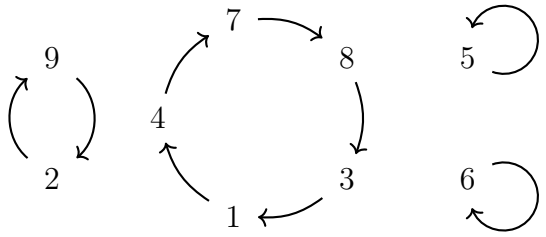


Figure 11.3: A graph $\widehat{t} \in \mathcal{F}_{1,9}$ with weight $w(\widehat{t}) = m_1^7 m_2$

We analyze $F_1(t)$ by decomposing the graphs in $\mathcal{F}_1$ into a few different types of structures. To each structure, we assign its exponential generating function (EGF). The overall $F_1(t)$ (which is also an EGF) is then found by multiplying all of these partial EGF's. For $F_1(t)$, there are three types of structures:

- Fixed points (cycles of length one) with weight $W = m_1 t$.

- Mussels (cycles of length two) with weight $W = -m_2 t^2$.
- Loops (cycles of length at least 3) where a loop of length l has weight $W = -m_1^l (-t)^l$.

Remark 11.1.7. We use W rather than w for these weights as we are including the factors of t and the signs in these weights.

Remark 11.1.8. Note that in order to verify the results by computer for small matrices, it is better to compute the logarithm of the EGF tather than the EGF as this makes the contribution of each structure additive rather than multiplicative. This allows us to heuristically identify missing structures we have not taken into account yet.

Fixed points Let $\mathcal{X}_1$ be the set of fixed points in $\mathcal{F}_1$. Then

$$\mathcal{X}_1 = \textsc{Set}\Bigg(\begin{array}{c}\circlearrowleft\\ 1\end{array}\Bigg). \tag{11.7}$$

For the EGF, this gives a factor of $\exp(m_1 t)$.

Mussels Next, let $\mathcal{M}_1$ be the set of mussels in $\mathcal{F}_1$. Then

$$\mathcal{M}_1 = \textsc{Set}\Bigg(\begin{array}{c}2\\ \circlearrowleft\\ 1\end{array}\Bigg). \tag{11.8}$$

The EGF of the inner factor is $-m_2 t^2/2!$ so the EGF of $\mathcal{M}_1$ is $\exp(-\frac{1}{2} m_2 t^2)$.

Loops Finally, let $\mathcal{L}_1$ be the set of loops in $\mathcal{F}_1$. Then

$$\mathcal{L}_1 = \textsc{Set}\Bigg(\textsc{Cyc}_{\geq 3}\Bigg(\begin{array}{c}\curvearrowright\\ 1\end{array}\Bigg)\Bigg). \tag{11.9}$$

The EGF for cycles of length $n \geq 3$ is $-m_1 t + \frac{m_1^2 t^2}{2} - \ln\left(\frac{1}{1+m_1 t}\right)$ so the EGF of $\mathcal{L}_1$ is equal to

$$\exp\left(-m_1 t + \frac{m_1^2 t^2}{2} - \ln\left(\frac{1}{1+m_1 t}\right)\right) = (1+m_1 t)e^{-m_1 t + \frac{1}{2}m_1^2 t^2}.$$

Conclusion Since $\mathcal{F}_1 = \mathcal{X}_1 \star \mathcal{M}_1 \star \mathcal{L}_1$, by multiplying the EGFs of the structures we obtain that

$$F_1(t) = e^{m_1 t} e^{-\frac{1}{2}m_2 t^2}(1+m_1 t)e^{-m_1 t + \frac{1}{2}m_1^2 t^2} = (1+m_1 t)e^{\frac{1}{2}(m_1^2 - m_2)t^2}. \tag{11.10}$$

11.1.3 Techniques for handling nonzero m_1

To analyze the second moment of the determinant of X when $m_1 \neq 0$, it is more convenient to write $X = Y + m_1 J$ and carry out our analysis in terms of the entries of Y.

Definition 11.1.9. We take $Y = X - m_1 J$ so we have that $Y_{ij} = X_{ij} - m_1$. We define $\mu_q = \mathbb{E}[Y_{ij}^q]$ to be the qth moment of the entries of Y. Note that $\mu_1 = 0$ and $\mu_2 = m_2 - {m_1}^2$.

Proposition 11.1.10.

$$\det X = \sum_{\pi \in S_n} \operatorname{sgn}(\pi) \prod_{i \in [n]} Y_{i\pi(i)} + m_1 \sum_{j \in [n]} \sum_{\pi \in S_n} \operatorname{sgn}(\pi) \prod_{i \in [n]\setminus\{j\}} Y_{i\pi(i)}. \tag{11.11}$$

Proof. Using the fact that $\det X = \sum_{\pi \in S_n} \operatorname{sgn}(\pi) \prod_{i \in [n]} X_{i\pi(i)}$ and multiplying everything out,

$$\begin{aligned}
\det X &= \sum_{\pi \in S_n} \operatorname{sgn}(\pi) \prod_{i \in [n]} X_{i\pi(i)} \\
&= \sum_{\pi \in S_n} \operatorname{sgn}(\pi) \prod_{i \in [n]} \left(Y_{i\pi(i)} + m_1\right) \\
&= \sum_{\pi \in S_n} \operatorname{sgn}(\pi) \sum_{A \subseteq [n]} m_1^{|A|} \prod_{i \in [n]\setminus A} Y_{i\pi(i)}.
\end{aligned}$$

We claim that the terms with $|A| \geq 2$ vanish. To see this, given an A such that $|A| \geq 2$, let a_1 and a_2 be the first two elements of A. For each permutation $\pi \in S_n$, let swap_π be the permutation such that $\text{swap}_\pi(a_1) = \pi(a_2)$, $\text{swap}_\pi(a_2) = \pi(a_1)$, and $\text{swap}_\pi(i) = \pi(i)$ for all $i \in [n] \setminus \{a_1, a_2\}$. Since $\text{sgn}(\text{swap}_\pi) = -\text{sgn}(\pi)$ for all $\pi \in S_n$ and $\text{swap}_\pi(i) = \pi(i)$ for all $i \in [n] \setminus A$, we have that

$$\sum_{\substack{\pi \in S_n \\ \pi(a_1) < \pi(a_2)}} \left(\text{sgn}(\pi) \prod_{i \in [n] \setminus A} m_1^{|A|} Y_{i\pi(i)} + \text{sgn}(\text{swap}_\pi) \prod_{i \in [n] \setminus A} m_1^{|A|} Y_{i\,\text{swap}_\pi(i)} \right) = 0.$$

□

Remark 11.1.11. Theorem 11.1.10 is a special case of the Matrix Determinant Lemma which says that $\det(A + uv^T) = (1 + v^T A^{-1} u) \det(A)$. We gave an alternative direct proof as it helps motivate the definition of marked permutations below.

Definition 11.1.12. We say σ is a *marked permutation* if it is either in S_n or it is formed from some $\pi \in S_n$ by setting $\sigma(i) = \times$ for some $i \in [n]$ and $\sigma(j) = \pi(j)$ for all $j \in [n] \setminus \{i\}$. In this case, we think of $\times$ as a mark and we call i the marked element of σ.

We define $\text{sgn}(\sigma) = \text{sgn}(\pi)$ and we take $Y^{\times}_{i\sigma(i)} = m_1$ if i is marked and $Y^{\times}_{i\sigma(i)} = Y_{i\pi(i)}$ otherwise. We write $S_n^{\times}$ for the set of all marked permutations.

Restating Theorem 11.1.10 in terms of marked permutations, we have the following equation.

Proposition 11.1.13.

$$\det X = \sum_{\sigma \in S_n^{\times}} \text{sgn}(\sigma) \prod_{i=1}^{n} Y^{\times}_{i\sigma(i)}.$$

We can easily modify tables and multigraphs to take these marks into account.

Definition 11.1.14. We define $G^{\times}_{k,n}$ to be the set of marked tables formed from $F_{k,n}$ by marking at most one element in each row. Note that $G^{\times}_{k,n}$ includes the tables where no elements are marked.

Given a table $t \in G^{\times}_{k,n}$ we define its weight to be $w(t) = \mathbb{E}\left[\prod_{j=1}^{k}\prod_{i=1}^{n} Y^{\times}_{i\sigma_j(i)}\right]$ where σ_j is the marked permutation corresponding to row j of t (i.e., $\sigma_j(i)$ is the element in row j and column i of t if this element is unmarked and $\times$ if this element is marked.).

Definition 11.1.15. Given a marked table $t \in G^{\times}_{k,n}$, we construct a marked multigraph $\widehat{t}$ by first taking the multigraph corresponding to t when we ignore the marks. We then mark the edges (if any) corresponding to marked elements in t. Note that at most one edge of each color is marked.

As before, we define $w(\widehat{t}) = w(t)$ and $\mathrm{sgn}(\widehat{t}) = \mathrm{sgn}(t)$. We define $\mathcal{G}^{\times}_{k,n}$ to be the set of all marked multigraphs corresponding to a marked table $t \in G^{\times}_{k,n}$.

Definition 11.1.16. Given $t \in G^{\times}_{k,n}$, we say that t and $\widehat{t}$ are trivial if there is an $i \in [n]$ such that there is exactly one unmarked loop incident to i in $\widehat{t}$ or there exist $i, j \in [n]$ such that there is exactly one unmarked edge (in either direction) between i and j in $\widehat{t}$. Equivalently, t and $\widehat{t}$ are trivial if there exists an $i \in [n]$ such that i appears exactly once in column i of t or there exist $i, j \in [n]$ such that the number of times j appears in column i of t plus the number of times i appears in column j of t is exactly 1.

Otherwise, we say that t and $\widehat{t}$ are non-trivial.

We observe that trivial tables/multigraphs have weight 0 so it is sufficient to restrict our attention to non-trivial tables/multigraphs.

Proposition 11.1.17. *If t and $\widehat{t}$ are trivial then $w(t) = w(\widehat{t}) = 0$.*

We will split our analysis into cases depending on the number of marks in the table/multigraph.

Definition 11.1.18. We define $G^r_{k,n}$ to be the set of tables in $G^\times_{k,n}$ which have exactly r marks. Similarly, we define $\mathcal{G}^r_{k,n}$ to be the set of multigraphs in $\mathcal{G}^r_{k,n}$ which have exactly r marked edges. Note that we can have $r = 0$, in that case, we often omit the superscript.

Definition 11.1.19. We define

$$g^r_k(n) = \sum_{\widehat{t} \in \mathcal{G}^r_{k,n}} w(\widehat{t}) \operatorname{sgn}(t) \qquad \text{and} \qquad G^r_k(t) = \sum_{n=0}^{\infty} \frac{t^n}{n!} g^r_k(n). \tag{11.12}$$

Proposition 11.1.20. *For any distribution of X_{ij},*

$$f_k(n) = \sum_{r=0}^{k} g^r_k(n) \qquad \textit{and} \qquad F_k(t) = \sum_{r=0}^{k} G^r_k(t). \tag{11.13}$$

In order to apply the techniques of Flajolet and Sedgewick [FS09], it is convenient to absorb $\operatorname{sgn}(\widehat{t})$ and t^n into the weight of $\widehat{t}$.

Definition 11.1.21. We define $\mathcal{G}^\times_k$ to be the (disjoint) union $\bigsqcup_{n=0}^{\infty} \mathcal{G}^\times_{k,n}$. For a given graph $\widehat{g} \in \mathcal{G}^\times_k$, such that $\widehat{g} \in \mathcal{G}^\times_{k,n}$ for some n, we define the corresponding weight $W(\widehat{g}) = w(\widehat{g}) \operatorname{sgn}(\widehat{g}) t^n$, where $\operatorname{sgn}(\widehat{g}) = (-1)^{kn}(-1)^{C(\widehat{g})}$ and we define the order $|\widehat{g}|$ of $\widehat{g}$ to be $|\widehat{g}| = n$. Similarly, we define $\mathcal{G}^r_k = \bigsqcup_{n=0}^{\infty} \mathcal{G}^r_{k,n}$.

Proposition 11.1.22. *For any distribution,*

$$F_k(t) = \sum_{\widehat{g} \in \mathcal{G}^\times_k} \frac{W(\widehat{g})}{|\widehat{g}|!}. \tag{11.14}$$

Remark 11.1.23. Note that $\mathcal{G}_k = \mathcal{G}^0_k$ tables (with no marks) and $\mathcal{F}_n$ tables are equivalent when $m_1 = 0$. More precisely, $f_k(n)|_{m_1=0} = g_k(n)|_{\mu_q \to m_q}$ and thus $F_k(t)|_{m_1=0} = G_k(t)|_{\mu_q \to m_q}$.

Example 11.1.24. An example of nontrivial table $t \in G^\times_{2,9}$ and its corresponding multigraph.

1	2	3	4	5	6	7	8	9
3	7	1	×	5	2	6	9	8
3	6	1	9	5	7	2	×	4

Figure 11.4: $t \in G^{\times}_{2,9}$ with weight $w(t) = \mu_2^6 \mu_4 m_1^2$,

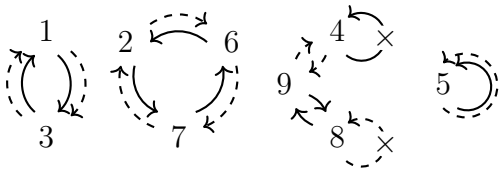


Figure 11.5: The corresponding $\widehat{t} \in \mathcal{G}^{\times}_{2,9}$

11.1.4 Rederivation of the first symmetric moment

To demonstrate the technique of using marked tables, we rederive $F_1(t)$.

11.1.4.1 Zero marks

Consider the set of all nontrivial 1-graphs in $\mathcal{G}_1 = \mathcal{G}_1^0$. This set includes only the permutations which decompose into mussels (2-cycles). Thus,

$$\mathcal{G}_1 = \mathcal{G}_1^0 = \textsc{Set}\left(\begin{array}{c} 2 \\ \circlearrowleft \quad \circlearrowright \\ 1 \end{array} \right). \tag{11.15}$$

In terms of the EGF,

$$G_1^0(t) = e^{-\frac{1}{2}\mu_2 t^2}. \tag{11.16}$$

11.1.4.2 One mark

It turns out there is only one nontrivial structure which can have one mark:

- A marked fixed point (marked 1-cycle) with weight $W = m_1 t$

Note that there are no marked mussels as they would have weight equal to zero.

Marked fixed points As there is only one mark per multigraph $\widehat{t}$ in $\mathcal{G}_1^1$, there must be exactly one structure containing it. The only nontrivial structure of $\mathcal{G}_1^1$ is precisely the

marked fixed point $\mathcal{X}_1$, whose EGF is $m_1 t$.

Conclusion Since other structures of $\mathcal{G}_1^1$ are taken from $\mathcal{G}_1$, we have $\mathcal{G}_1^1 = \mathcal{X}_1 \star \mathcal{G}_1$. Hence, in terms of the EGFs, $G_1^1(t) = m_1 t\, G_1(t)$.

11.1.4.3 Full generating function

By Theorem 11.1.20,

$$F_1(t) = G_1^0(t) + G_1^1(t) = (1+m_1 t)G_1(t) = (1+m_1 t)e^{-\frac{1}{2}\mu_2 t^2} = (1+m_1 t)e^{\frac{1}{2}(m_1^2 - m_2)t^2}, \quad (11.17)$$

which matches Equation (11.10).

11.1.5 Second symmetric moment

Next, we move to the case of $k = 2$. To derive the full generating function $F_2(t)$, we must analyze all of the possible structures of nontrivial graphs in $\mathcal{G}_2^{\times}$.

11.1.5.1 Zero marks

Proposition 11.1.25. *For any distribution of X_{ij} with $m_1 = 0$,*

$$F_2(t) = \frac{\exp\left(\frac{(m_4 - 3m_2^2)t^2}{2} - m_2 t\right)}{(1 - m_2 t)^{\,2}\sqrt{1 - m_2^2 t^2}} \quad (11.18)$$

Remark 11.1.26. For the special case when $m_2 = 1$, we obtain

$$F_2(t) = \frac{\exp\left(\frac{(m_4 - 3)t^2}{2} - t\right)}{(1-t)^2\sqrt{1 - t^2}} \quad (11.19)$$

Proof. First, in the spirit of Remark 11.1.23, we consider the structure of nontrivial graphs in $\mathcal{G}_2$. An example of such graph is in Figure 11.7.

Example 11.1.27. An example of a nontrivial table $t \in G^0_{2,10}$ and its corresponding multigraph.

1	2	3	4	5	6	7	8	9	10
3	7	1	9	8	2	6	5	4	10
3	6	1	5	4	7	2	9	8	10

Figure 11.6: $t \in G_{2,10}$ with weight $w(t) = \mu_2^8\mu_4$,

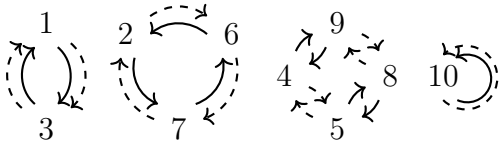
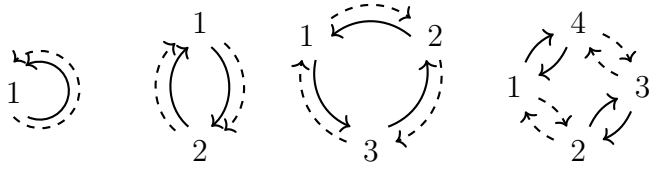


Figure 11.7: A corresponding $\widehat{t} \in \mathcal{G}_{2,10}$

This example was purposely created to show all types of structures a graph $\widehat{t}$ from $\mathcal{G}_2$ can have. For $\mathcal{G}_2$, we have the following structures.

- Fixed points (two 1-cycles on the same element) with weight $W = \mu_2 t$.
- Mussels (two 2-cycles on two elements) with weight $W = \mu_4 t^2$.
- Loops (double cycles on $n \geq 3$ vertices) with weight $W = \mu_2^n t^n$.
- Necklaces (closed chains of alternating dashed/undashed 2-cycles on n vertices where $n \geq 4$ is even) with weight $W = \mu_2^n t^n$.

Representatives of each type are also shown in Table 11.1.



Fixed point	Mussel	Loop	Necklace
$\mathcal{X}_2$	$\mathcal{M}_2$	$\mathcal{L}_2$	$\mathcal{N}_2$

Table 11.1: Types of structures in $\mathcal{G}_2$

Fixed points Let $\mathcal{X}_2$ be the set of fixed points in $\mathcal{G}_2$. Then

$$\mathcal{X}_2 = \textsc{Set}\left(\begin{array}{c}1\\ \circlearrowleft\end{array}\right). \tag{11.20}$$

In terms of the EGF, this gives a factor of $\exp(\mu_2 t)$.

Mussels Next, let $\mathcal{M}_2$ be the set of mussels in $\mathcal{G}_2$. Then

$$\mathcal{M}_2 = \textsc{Set}\left(\begin{array}{c}1\\ \circlearrowleft\\ 2\end{array}\right). \tag{11.21}$$

The EGF of the inner factor is $\mu_4 t^2/2!$ so the EGF of $\mathcal{M}_2$ is $\exp(\frac{1}{2}\mu_4 t^2)$.

Loops There are two ways how we can make a loop. Either the undashed and dashed cycles have the same orientation, or they have opposite orientations. Letting $\mathcal{L}_2$ be the set of loops in $\mathcal{G}_2$, we have

$$\mathcal{L}_2 = \textsc{Set}\left(\textsc{Cyc}_{\geq 3}\left(\begin{array}{c}\curvearrowright\\ 1\end{array}\right)\right) \star \textsc{Set}\left(\textsc{Cyc}_{\geq 3}\left(\begin{array}{c}\curvearrowleft\\ 1\end{array}\right)\right). \tag{11.22}$$

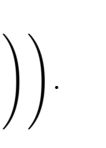
The corresponding EGF of $\mathcal{L}_2$ is equal to

$$\left[\exp\left(-\mu_2 t - \frac{\mu_2^2 t^2}{2} + \ln\left(\frac{1}{1-\mu_2 t}\right)\right)\right]^2 = \frac{e^{-2\mu_2 t - \mu_2^2 t^2}}{(1-\mu_2 t)^2}. \tag{11.23}$$

Necklaces Finally, the set of necklaces $\mathcal{N}_2$ is formed as cycles of two alternating 2-cycles joined together. Note that it depends on the order of the elements in consecutive 2-cycles. However, every necklace is counted twice as the mirror image of a necklace is counted as the

same necklace. Overall, we have that

$$\mathcal{N}_2 = \textsc{Set}\left(\frac{1}{2}\,\textsc{Cyc}_{\geq 2}\left(\right)\right). \tag{11.24}$$



The resulting EGF is

$$\exp\left(\frac{1}{2}\left(-2\mu_2^2\frac{t^2}{2!} + \ln\left(\frac{1}{1-2\mu_2^2\frac{t^2}{2!}}\right)\right)\right) = \frac{e^{-\frac{1}{2}\mu_2^2 t^2}}{\sqrt{1-\mu_2^2 t^2}}. \tag{11.25}$$

Conclusion Since $\mathcal{G}_2 = \mathcal{X}_2 \star \mathcal{M}_2 \star \mathcal{L}_2 \star \mathcal{N}_2$, by multiplying the EGFs we obtain that

$$G_2(t) = e^{\mu_2 t} e^{\frac{1}{2}\mu_4 t^2} \frac{e^{-2\mu_2 t - \mu_2^2 t^2}}{(1-\mu_2 t)^2} \frac{e^{-\frac{1}{2}\mu_2^2 t^2}}{\sqrt{1-\mu_2^2 t^2}} = \frac{\exp\left(\frac{(\mu_4 - 3\mu_2^2)t^2}{2} - \mu_2 t\right)}{(1-\mu_2 t)^{\,2}\sqrt{1-\mu_2^2 t^2}}. \tag{11.26}$$

□

11.1.5.2 One mark

Note that all unmarked structures are also part of every graph $\mathcal{G}_2^r$ for $r \geq 1$. Thus, we only need to find the exponential generating function of connected structures having marks and add them up (as they are disjoint). It turns out that the only marked structure of each $\widehat{t} \in \mathcal{G}_2^1$ is a mussel with exactly one mark (1-marked mussel).

Example 11.1.28. An example of a nontrivial table $t \in G_{2,9}^1$ and its corresponding nontrivial multigraph $\widehat{t} \in \mathcal{G}_{2,9}^1$.

1	2	3	4	5	6	7	8	9
3	7	1	9	8	2	6	5	4
3	6	×	5	4	7	2	9	8

Figure 11.8: $t \in G_{2,9}$ with $w(t) = m_1\mu_2^7\mu_3$,

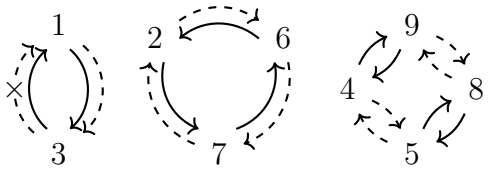


Figure 11.9: The corresponding $\widehat{t} \in \mathcal{G}_{2,10}$

The EGF of a mussel with exactly one mark is equal to $\frac{1}{2}m_1\mu_3t^2$. However, there are four places where we can put our mark so the full EGF is $2m_1\mu_3t^2$ and

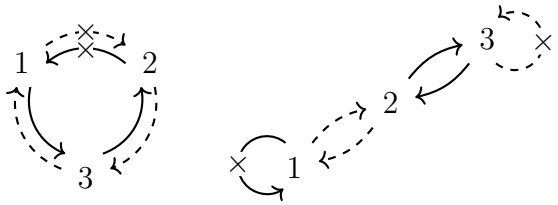
$$G_2^1(t) = 2m_1\mu_3t^2G_2(t) \tag{11.27}$$

as we fill the remaining space with unmarked structures.

11.1.5.3 Two marks

The following Table 11.2 shows representatives of all types of the remaining (connected) structures having exactly two marks. A structure with two marks can be

- A marked loop (double cycle on $n \geq 2$ vertices with two marks) with weight $W = m_1^2\mu_2^{n-1}t^n$.
- A marked chain (alternating dashed/undashed 2-cycle on an odd number n of vertices, provided we attach two marked 1-cycles to the endpoints) with weight $W = m_1^2\mu_2^{n-1}t^n$.



2-Marked loop	2-Marked chain
$\mathcal{L}_2^2$	$\mathcal{C}_2^2$

Table 11.2: Types of structures in $\mathcal{G}_2^2$ with two marks

Remark 11.1.29. There are no 2-marked necklaces as they would require marking two arrows of the same type, which is forbidden. Also note that there are 2-marked fixed points, but they are just a special case of chains with a single vertex. Similarly, we absorb 2-marked mussels into loops, allowing the number of their vertices to start from $n = 2$.

Marked mussels The EGF of a mussel marked by two marks is equal to $\frac{1}{2}m_1^2\mu_2 t^2$. However, there are 4 ways how we can mark one dashed and one undashed arrow so the total EGF is $2m_1^2\mu_2 t^2$.

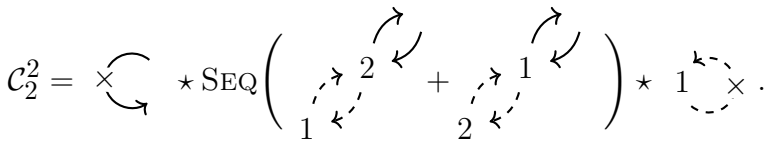
Marked loops The set of marked loops $\mathcal{L}_2^2$ with two marks is created by starting with a parallel pair of marked arrows and attaching a sequence of $n-1$ parallel pairs of arrows (where $n \geq 3$) to this marked pair of arrows. For each pair of arrows, one of the arrows is dashed and the other is undashed. We either have that all of the dashed and undashed arrows point in the same direction or we have that all of the dashed arrows point in one direction while all of the undashed arrows point in the opposite direction. Thus,

$\mathcal{L}_2^2 =$  (11.28)

In terms of the EGF,

$$2m_1^2 t\left(-1 + \frac{1}{1-\mu_2 t}\right) = \frac{2m_1^2\mu_2 t^2}{1-\mu_2 t}.$$

Marked chains Let $\mathcal{C}_2^2$ be the set of all marked chains. We can write

$\mathcal{C}_2^2 =$  . (11.29)

In terms of the EGF,

$$m_1 \left(\frac{1}{1 - 2\mu_2^2 \frac{t^2}{2!}} \right) m_1 t = \frac{m_1^2 t}{1 - \mu_2^2 t^2}.$$

Pair of one-marked structures Finally, we can create a multigraph with two marks by combining a multigraph with no marks, a 1-marked mussel with a mark on a dashed arrow, and a 1-marked mussel with a mark on an undashed arrow. Since both of these mussels have EGF equal to $m_1 \mu_3 t^2$, we get that the contribution to $G_2^2(t)$ is $(m_1 \mu_3 t^2)^2 G_2(t)$.

Conclusion Adding the contributions together, we get

$$G_2^2(t) = \left(\frac{2m_1^2 \mu_2 t^2}{1 - \mu_2 t} + \frac{m_1^2 t}{1 - \mu_2^2 t^2} + (m_1 \mu_3 t^2)^2 \right) G_2(t). \tag{11.30}$$

11.1.5.4 Full generating function

Proposition 11.1.30. *For any distribution of X_{ij},*

$$F_2(t) = \left(1 + 2m_1 \mu_3 t^2 + m_1^2 \left(\mu_3^2 t^4 + \frac{t}{1 - \mu_2^2 t^2} + \frac{2\mu_2 t^2}{1 - \mu_2 t} \right) \right) \frac{\exp\left(\frac{(\mu_4 - 3\mu_2^2) t^2}{2} - \mu_2 t \right)}{(1 - \mu_2 t)^2 \sqrt{1 - \mu_2^2 t^2}}. \tag{11.31}$$

Proof. By adding up the contributions,

$$\begin{aligned} F_2(t) &= G_2^0(t) + G_2^1(t) + G_2^2(t) \\ &= \left[1 + 2m_1 \mu_3 t^2 + \left(\frac{2m_1^2 \mu_2 t^2}{1 - \mu_2 t} + \frac{m_1^2 t}{1 - \mu_2^2 t^2} + (m_1 \mu_3 t^2)^2 \right) \right] G_2(t). \end{aligned} \tag{11.32}$$

□

Corollary 11.1.31. *For any distribution X_{ij} with $\mu_2 = 1$,*

$$F_2(t) = \left(1 + 2m_1\mu_3 t^2 + m_1^2\left(\mu_3^2 t^4 + \frac{t}{1-t^2} + \frac{2t^2}{1-t}\right)\right)\frac{\exp\left(\frac{(\mu_4-3)t^2}{2} - t\right)}{(1-t)^2\sqrt{1-t^2}}. \tag{11.33}$$

11.1.5.5 Asymptotics

Expanding the EGFs around their singular point and by singular asymptotics (cf. [FS09]),

$$f_2(n) = \frac{4n^{3/2}n!}{3\sqrt{2\pi}}e^{\frac{1}{2}(\mu_4-5)}\left(m_1^2 n + 1 + 2m_1\mu_3 + \frac{m_1^2}{4}\left(43 + 4\mu_3^2 - 10\mu_4\right) + O\left(\frac{1}{n}\right)\right). \tag{11.34}$$

11.2 Wigner matrices

Let Z be Hermitian, with off-diagonal entries

$$Z_{ij} = R_{ij} + \mathbf{i}I_{ij}, \qquad Z_{ji} = R_{ij} - \mathbf{i}I_{ij} \quad (i < j).$$

Assume that the centered families $\{R_{ij}\}$, $\{I_{ij}\}$, and the real diagonal entries $\{Z_{ii}\}$ are mutually independent and i.i.d. within each family. Write

$$\mu_q = \mathbb{E}[R_{ij}^q], \qquad \nu_q = \mathbb{E}[I_{ij}^q], \qquad \kappa_q = \mathbb{E}[Z_{ii}^q].$$

For the all-ones matrix J and the identity I_n, define the characteristic correlation function and its EGF by

$$h_k(n; a_1, \ldots, a_k, c_1, \ldots, c_k) = \mathbb{E}\prod_{r=1}^{k}\det(Z + a_r J + c_r I_n),$$
$$H_k(t; a_1, \ldots, a_k, c_1, \ldots, c_k) = \sum_{n\geq 0} h_k(n; a_1, \ldots, a_k, c_1, \ldots, c_k)\frac{t^n}{n!}.$$

In this section, we analyze the EGFs $H_1(t, a_1, c_1)$ and $H_2(t, a_1, a_2, c_1, c_2)$ of the characteristic correlation functions $h_1(n, a_1, c_1)$ and $h_2(n, a_1, a_2, c_1, c_2)$ for Wigner matrices Z. For this analysis, we use the same techniques as before with the following modification. Due to the $c_1 I_n$ and $c_2 I_n$ terms in $h_2(n, a_1, a_2, c_1, c_2) = \mathbb{E}\left[\det\left(Z + a_1 J + c_1 I_n\right) \det\left(Z + a_2 J + c_2 I_n\right)\right]$, tables with exactly one i in column i may now have nonzero weight. Thus, we modify the definition of trivial marked tables and multigraphs as follows.

Definition 11.2.1. Given a marked $k \times n$ table t and the corresponding marked multigraph $\widehat{t}$, we say that t and $\widehat{t}$ are trivial if there exist $i, j \in [n]$ such that there is exactly one unmarked edge (in either direction) between i and j in $\widehat{t}$. Equivalently, t and $\widehat{t}$ are trivial if there exist $i, j \in [n]$ such that the number of times j appears in column i of t plus the number of times i appears in column j of t is exactly 1.

Otherwise, we say that t and $\widehat{t}$ are non-trivial.

In order to distinguish this case from our previous analysis where we had that $c_i = 0$ for all $i \in [k]$, we denote the sets of non-trivial marked tables and multigraphs with k rows/colors by $H_k^{\times}$ and $\mathcal{H}_k^{\times}$ rather than $G_k^{\times}$ and $\mathcal{G}_k^{\times}$. We have similar definitions and equations as in Section 11.1.3:

1. We define $\mathcal{H}_k^r$ to be the subset of $\mathcal{H}_k^{\times}$ of marked graphs which have exactly r marks (when $r = 0$, we sometimes omit writing r).

2. We define
$$h_k^r(n) = \sum_{\widehat{t} \in \mathcal{H}_{k,n}^r} w(\widehat{t}) \operatorname{sgn}(\widehat{t}), \qquad H_k^r(t) = \sum_{n=0}^{\infty} h_k^r(n) \frac{t^n}{n!}.$$

3. We have that $H_k(t, a_1, \ldots, a_k, c_1, \ldots, c_k) = \sum_{r=0}^{k} H_k^r(t)$

4. As before, for any $\widehat{g} \in \mathcal{H}_k$, we define the weight $W(\widehat{g})$ to be $W(\widehat{g}) = \operatorname{sgn}(\widehat{g}) w(\widehat{g}) t^n$ and we have that $H_k(t, a_1, \ldots, a_k, c_1, \ldots, c_k) = \sum_{\widehat{g} \in \mathcal{H}_k^{\times}} \frac{W(\widehat{g})}{|\widehat{g}|!}$

Example 11.2.2. An example of a nontrivial table $t \in H^{\times}_{2,9}$ and its corresponding nontrivial multigraph $\widehat{t} \in \mathcal{H}^{\times}_{2,9}$.

1	2	3	4	5	6	7	8	9
3	×	1	9	8	7	6	5	4
3	6	×	5	4	2	7	9	8

Figure 11.10: $t \in H^2_{2,9}$ with weight $w(t) = c_2(\mu_2 + \nu_2)^6(\mu_3 + \mathbf{i}\nu_3)a_1a_2$,

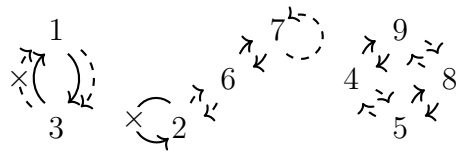


Figure 11.11: The corresponding $\widehat{t} \in \mathcal{H}^2_{2,9}$

11.2.1 First moment

11.2.1.1 Zero marks

Proposition 11.2.3.

$$H_1(t,0) = H^0_1(t) = e^{c_1 t - \frac{1}{2}(\mu_2+\nu_2)t^2}. \tag{11.35}$$

Proof. We have the following structures in $\mathcal{H}^0_1$:

- Fixed points (1-cycles) with weight $W = c_1 t$.
- Mussels (2-cycles) with weight $W = (\mu_2 + \nu_2)t^2$.

Thus,

$$\mathcal{H}^0_1 = \textsc{Set}\Big(\circlearrowright 1 \Big) \star \textsc{Set}\Big(\overset{2}{\underset{1}{\circ}} \Big). \tag{11.36}$$

In terms of the EGF,

$$H^0_1(t) = H^0_1(t) = e^{c_1 t - \frac{1}{2}(\mu_2+\nu_2)t^2}. \tag{11.37}$$

□

11.2.1.2 One mark

The only nontrivial marked structure in $\mathcal{H}_1^1$ is a marked fixed point (1-cycle with one mark) with weight $W = a_1 t$, thus

$$\mathcal{H}_1^1 = \; \times \circlearrowright 1 \; \star \mathcal{H}_1^0. \tag{11.38}$$

In terms of the EGF,

$$H_1^1(t) = a_1 t \, H_2^0(t) = a_1 t \, e^{c_1 t - \frac{1}{2}(\mu_2+\nu_2)t^2}. \tag{11.39}$$

11.2.1.3 Full generating function

Proposition 11.2.4.

$$H_1(t, a_1) = (1 + a_1 t) e^{c_1 t - \frac{1}{2}(\mu_2+\nu_2)t^2}. \tag{11.40}$$

Proof. Observe that $H_1(t, a_1) = H_1^0(t) + H_1^1(t) = (1 + a_1 t) e^{c_1 t - \frac{1}{2}(\mu_2+\nu_2)t^2}$. □

11.2.2 Second moment

11.2.2.1 Zero marks

Proposition 11.2.5.

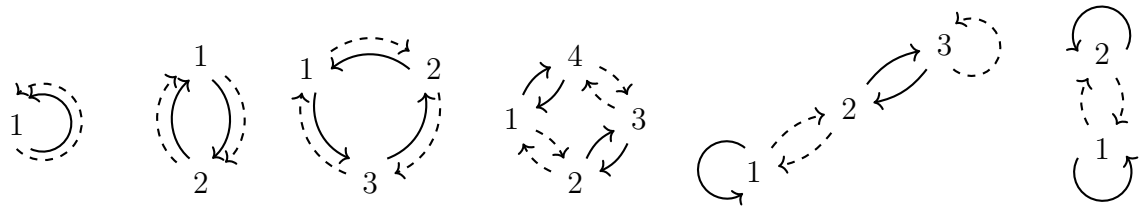
$$H_2(t, 0, 0) = H_2^0(t) = \frac{\exp\left(t\left(\kappa_2 - 2\mu_2\right) + \frac{1}{2}t^2\left(\mu_4 - 3\nu_2^2 - 3\mu_2^2 + \nu_4\right) - \frac{(\mu_2+\nu_2)\kappa_1^2 t^2}{1+t(\mu_2+\nu_2)} \right)}{\left(1 - 2\mu_2 t + (\mu_2^2 - \nu_2^2)t^2\right)\sqrt{1 - (\mu_2+\nu_2)^2 t^2}}. \tag{11.41}$$

Proof. For $\mathcal{H}_2^0$, we have the following types of structures:

- Fixed points (two 1-cycles on the same element) with weight $W = (c_1 c_2 + \kappa_2)t$
- Mussels (two 2-cycles on two elements) with weight $W = (\mu_4 + 2\mu_2\nu_2 + \nu_4)t^2$

- Loops (double cycles on $n \geq 3$ vertices) with weight $W = (\mu_2 + \nu_2)^n t^n$ if the arrows have the opposite direction and $W = (\mu_2 - \nu_2)^n t^n$ if they have the same direction
- Necklaces (closed chains of alternating dashed/undashed 2-cycles on n vertices, $n \geq 4$ even) with weight $W = (\mu_2 + \nu_2)^n t^n$.
- Chains (sequences of alternating dashed/undashed 2-cycles on n vertices with endpoints being 1-cycles, $n \geq 3$ odd) with weight $W = (\mu_2 + \nu_2)^{n-1} c_1 c_2 t^n$
- Dumbbells (sequences of alternating dashed/undashed 2-cycles on n vertices with endpoints being 1-cycles of same type, $n \geq 2$ even) with weight $W = -(\mu_2 + \nu_2)^{n-1} c_1^2 t^n$ if those 1-cycles are undashed. If they are dashed, we simply replace c_1^2 by c_2^2.

Representatives of each type are shown in Table 11.3.



Fixed point	Mussel	Loop	Necklace	Chain	Dumbbell
$\mathcal{X}_2$	$\mathcal{M}_2$	$\mathcal{L}_2$	$\mathcal{N}_2$	$\mathcal{C}_2$	$\mathcal{D}_2$

Table 11.3: Types of structures in $\mathcal{H}_2$

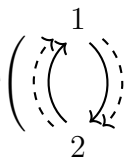
Fixed points Let $\mathcal{X}_2$ be the set of fixed points in $\mathcal{H}_2^0$. Then

$$\mathcal{X}_2 = \textsc{Set}\left(\begin{array}{c}1\\ \circlearrowleft\end{array}\right). \tag{11.42}$$

In terms of the EGF, this gives the factor of $\exp((c_1 c_2 + \kappa_2)t)$.

Mussels Next, let $\mathcal{M}_2$ be the set of mussels in $\mathcal{H}_2^0$. Then

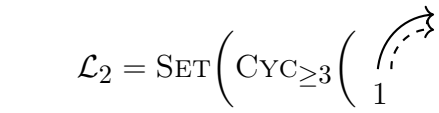
$$\mathcal{M}_2 = \textsc{Set}\left(\right). \tag{11.43}$$



The EGF of the inner factor is $(\mu_4 + 2\mu_2\nu_2 + \nu_4)t^2/2!$ so the EGF of $\mathcal{M}_2$ is $\exp(\frac{1}{2}(\mu_4 + 2\mu_2\nu_2 + \nu_4)t^2)$.

Loops Let $\mathcal{L}_2$ be the set of loops in $\mathcal{H}_2^0$. Note that $\mathcal{L}_2$ includes both possible orientations for the arrows. We have that

$$\mathcal{L}_2 = \textsc{Set}\Big(\textsc{Cyc}_{\geq 3}\Big(\Big) \Big) \star \textsc{Set}\Big(\textsc{Cyc}_{\geq 3}\Big(\Big)\Big). \tag{11.44}$$



The corresponding EGF of $\mathcal{L}_2$ is equal to

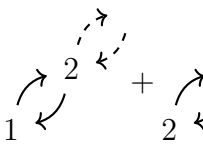
$$\exp\left(-(\mu_2-\nu_2)t - \frac{(\mu_2-\nu_2)^2t^2}{2} + \ln\left(\frac{1}{1-(\mu_2-\nu_2)t}\right)\right)$$
$$\times \exp\left(-(\mu_2+\nu_2)t - \frac{(\mu_2+\nu_2)^2t^2}{2} + \ln\left(\frac{1}{1-(\mu_2+\nu_2)t}\right)\right).$$

or after simplification,

$$\frac{e^{-2\mu_2 t-(\mu_2^2+\nu_2^2)t^2}}{1-2\mu_2 t+\left(\mu_2^2-\nu_2^2\right)t^2}.$$

Necklaces For the set of necklaces $\mathcal{N}_2$,

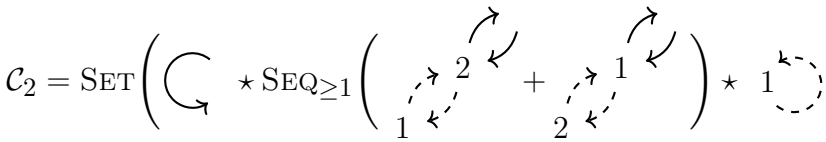
$$\mathcal{N}_2 = \textsc{Set}\left(\frac{1}{2}\,\textsc{Cyc}_{\geq 2}\Big(\quad + \quad \Big)\right). \tag{11.45}$$



Hence, its EGF is

$$\exp\left(\frac{1}{2}\left(-2(\mu_2+\nu_2)^2\frac{t^2}{2!}+\ln\left(\frac{1}{1-2(\mu_2+\nu_2)^2\frac{t^2}{2!}}\right)\right)\right)=\frac{e^{-\frac{1}{2}(\mu_2+\nu_2)^2t^2}}{\sqrt{1-(\mu_2+\nu_2)^2t^2}}. \tag{11.46}$$

Chains Let $\mathcal{C}_2$ denote the set of all chains. We can write

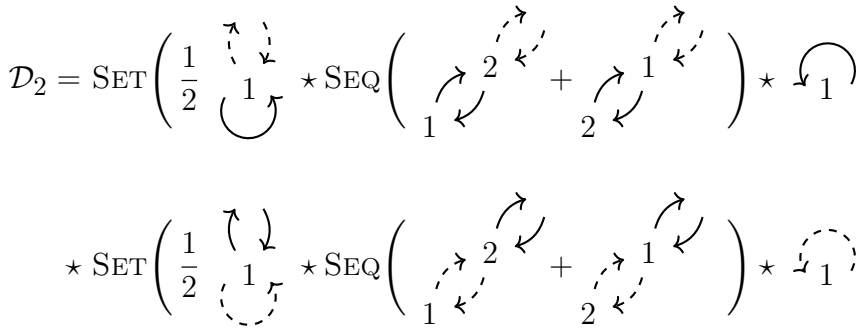


(11.47)

In terms of the EGF,

$$\exp\left[c_1\left(-1+\frac{1}{1-2(\mu_2+\nu_2)^2\frac{t^2}{2!}}\right)c_2t\right]=\exp\left(\frac{c_1c_2\left(\mu_2+\nu_2\right)^2t^3}{1-\left(\mu_2+\nu_2\right)^2t^2}\right).$$

Dumbbells There are two types of dumbbells as the endpoints are either undashed 1-cycles or dashed 1-cycles. Dumbbells are different from chains as they are symmetric and have the opposite sign and parity of the number of vertices. Hence, we have that for the set $\mathcal{D}_2$ of all dumbbells,



(11.48)

Its EGF is thus equal to

$$\begin{aligned}&\left[\exp\left(-\frac{1}{2}c_1(\mu_2+\nu_2)t\left(\frac{1}{1-2(\mu_2+\nu_2)^2\frac{t^2}{2!}}\right)c_1t\right)\right]\\&\left[\exp\left(-\frac{1}{2}c_2(\mu_2+\nu_2)t\left(\frac{1}{1-2(\mu_2+\nu_2)^2\frac{t^2}{2!}}\right)c_2t\right)\right]\\&=\exp\left(-\frac{\frac{1}{2}(c_1^2+c_2^2)(\mu_2+\nu_2)t^2}{1-(\mu_2+\nu_2)^2t^2}\right).\end{aligned}\tag{11.49}$$

Conclusion As $\mathcal{H}_2^0=\mathcal{X}_2\star\mathcal{M}_2\star\mathcal{L}_2\star\mathcal{N}_2\star\mathcal{C}_2\star\mathcal{D}_2$, by multiplying the EGFs we obtain that

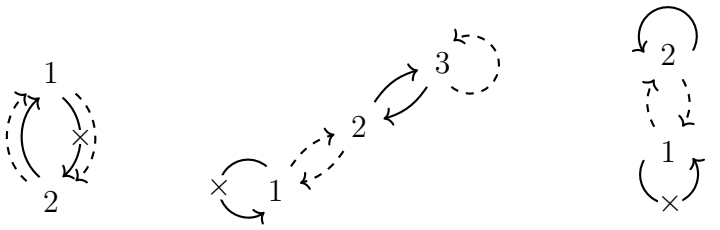
$$H_2^0(t)=\frac{\exp\left(t\left(c_1c_2+\kappa_2-2\mu_2\right)+\frac{1}{2}t^2\left(\mu_4-3\nu_2^2-3\mu_2^2+\nu_4\right)-\frac{t^2}{2}\frac{(c_1^2+c_2^2)(\mu_2+\nu_2)}{1-t^2(\mu_2+\nu_2)^2}+\frac{c_1c_2(\mu_2+\nu_2)^2t^3}{1-(\mu_2+\nu_2)^2t^2}\right)}{\left(1-2\mu_2t+\left(\mu_2^2-\nu_2^2\right)t^2\right)\sqrt{1-t^2\left(\mu_2+\nu_2\right)^2}}.$$

□

11.2.2.2 One mark

A structure with one mark can be

- A marked mussel (double 2-loop with one arrow marked) where W depends on both which type of arrow was dashed *and* also if the marked arrow points from the smaller to the bigger number or the other way around. Overall, W can attain one of the four values $a_1(\mu_3\pm\mathbf{i}\nu_3)t^2$ and $a_2(\mu_3\pm\mathbf{i}\nu_3)t^2$
- A marked chain (alternating dashed/undashed 2-cycle on an odd $n\geq 1$ number of vertices, provided we attach one marked 1-cycle and one unmarked 1-cycle to the endpoints) with weight $W=a_1\kappa_1(\mu_2+\nu_2)^{n-1}t^n$ or $W=a_2\kappa_1(\mu_2+\nu_2)^{n-1}t^n$ depending on which type of arrow is marked.
- A marked dumbbell (sequence alternating dashed/undashed 2-cycles on n vertices with endpoints being 1-cycles of same type, $n\geq 2$ even, with one mark at the endpoint) with weight $W=-a_1\kappa_1(\mu_2+\nu_2)^{n-1}t^n$ or $W=-a_2\kappa_1(\mu_2+\nu_2)^{n-1}t^n$.

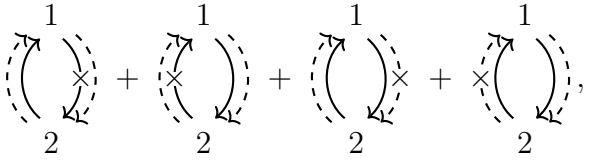


1-Marked mussel	1-Marked chain	1-Marked dumbbell
$\mathcal{M}_2^1$	$\mathcal{C}_2^1$	$\mathcal{D}_2^1$

Table 11.4: Types of structures in $\mathcal{H}_2^1$ with one mark

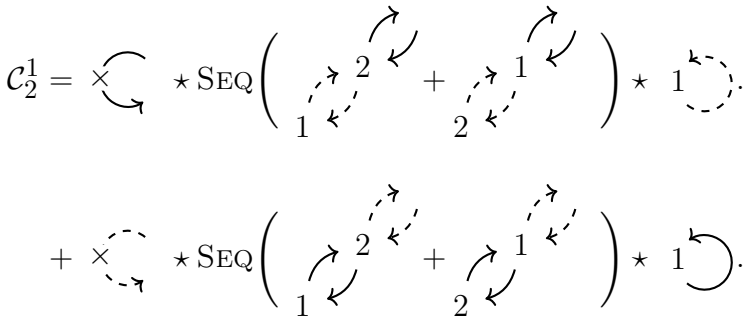
Remark 11.2.6. Here, we included 1-marked fixed points (one marked and one unmarked 1-cycle) into 1-marked chains as they have the same weight.

Marked mussels Next, let $\mathcal{M}_2^1$ be the set of 1-marked mussels. By adding up the contribution for the four different possibilities for which edge is marked, that is

$$\mathcal{M}_2^1 =$$  (11.50)

we obtain $2(a_1 + a_2)\mu_3 t^2/2!$ for its EGF.

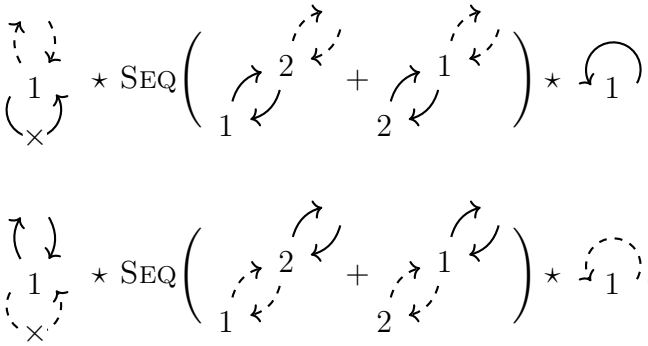
Marked chains For the set $\mathcal{C}_2^1$ of chains with one mark, we write

$$\mathcal{C}_2^1 =$$  (11.51)

In terms of the EGF,

$$a_1 \left(\frac{1}{1-2(\mu_2+\nu_2)^2 \frac{t^2}{2!}} \right) c_2 t + a_2 \left(\frac{1}{1-2(\mu_2+\nu_2)^2 \frac{t^2}{2!}} \right) c_1 t = \frac{(a_1 c_2 + a_2 c_1)t}{1-(\mu_2+\nu_2)^2 t^2}.$$

Marked dumbbells For the set $\mathcal{D}_2^1$ of all dumbbells with one mark,

$\mathcal{D}_2^1 =$  (11.52)

Its EGF is thus equal to

$$\begin{aligned} &- a_1(\mu_2+\nu_2)t \left(\frac{1}{1-2(\mu_2+\nu_2)^2 \frac{t^2}{2!}} \right) c_1 t - a_2(\mu_2+\nu_2)t \left(\frac{1}{1-2(\mu_2+\nu_2)^2 \frac{t^2}{2!}} \right) c_2 t \\ &= - \frac{(\mu_2+\nu_2)(a_1 c_1 + a_2 c_2)t^2}{1-(\mu_2+\nu_2)^2 t^2}. \end{aligned} \tag{11.53}$$

Note that the sign is negative as the total number of 2-cycles contained in a dumbbell (of either color) is odd.

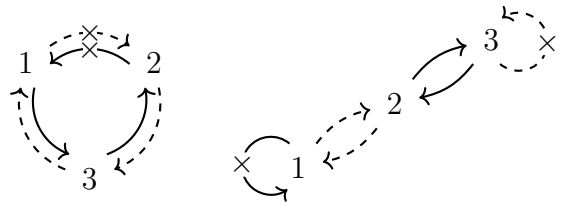
Conclusion Since $\mathcal{H}_2^1 = (\mathcal{M}_2^1 + \mathcal{C}_2^1 + \mathcal{D}_2^1) \star \mathcal{H}_2^0$, we obtain that

$$H_2^1(t) = \left((a_1+a_2)\mu_3 t^2 + \frac{(a_1 c_2 + a_2 c_1)t}{1-(\mu_2+\nu_2)^2 t^2} - \frac{(\mu_2+\nu_2)(a_1 c_1 + a_2 c_2)t^2}{1-(\mu_2+\nu_2)^2 t^2} \right) H_2^0(t). \tag{11.54}$$

11.2.2.3 Two marks

The following Table 11.5 shows representatives of all types of the remaining (connected) structures having exactly two marks. A structure with two marks can be

- A marked loop (double cycle on $n \geq 2$ vertices with two marks) with weight $W = a_1 a_2 (\mu_2 \pm \nu_2)^{n-1} t^n$ (plus sign if the arrows have the opposite direction and minus sign when they have the same direction)
- A marked chain (alternating dashed/undashed 2-cycle on an odd number $n \geq 1$ of vertices, provided we attach two marked 1-cycles to the endpoints) with weight $W = a_1 a_2 (\mu_2 + \nu_2)^{n-1} t^n$.



2-Marked loop	2-Marked chain
$\mathcal{L}_2^2$	$\mathcal{C}_2^2$

Table 11.5: Types of structures in $\mathcal{H}_2^2$ with two marks

Remark 11.2.7. There are no marked dumbbells as they would violate the assumption that there is at most one mark per one type of arrow. There are, however, 2-marked fixed points (two marked 1-cycle), but they are again counted as 2-marked chains with one vertex. Similarly, 2-marked mussels (two 2-cycles with two marks) are counted as 2-marked loops with two vertices.

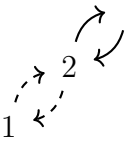
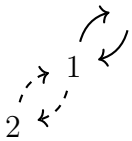
Marked loops The set of marked loops $\mathcal{L}_2^2$ is created as follows

$$\mathcal{L}_2^2 = \Big(\overset{\overset{+}{\square}}{1} \star \textsc{Seq}_{\geq 1}\Big(\overset{}{1} \Big) \Big) + \Big(\overset{\overset{+}{\square}}{1} \star \textsc{Seq}_{\geq 1}\Big(\overset{}{1} \Big) \Big). \tag{11.55}$$

In terms of the EGF,

$$a_1 a_2 t \left(-1 + \frac{1}{1-(\mu_2-\nu_2)t}\right) + a_1 a_2 t \left(-1 + \frac{1}{1-(\mu_2+\nu_2)t}\right) = \frac{2a_1 a_2 t^2 \left(\mu_2 - \mu_2^2 t + \nu_2^2 t\right)}{1 - 2\mu_2 t + (\mu_2^2 - \nu_2^2)t^2}.$$

Marked chains Letting $\mathcal{C}_2^2$ be the set of all marked chains, we can write

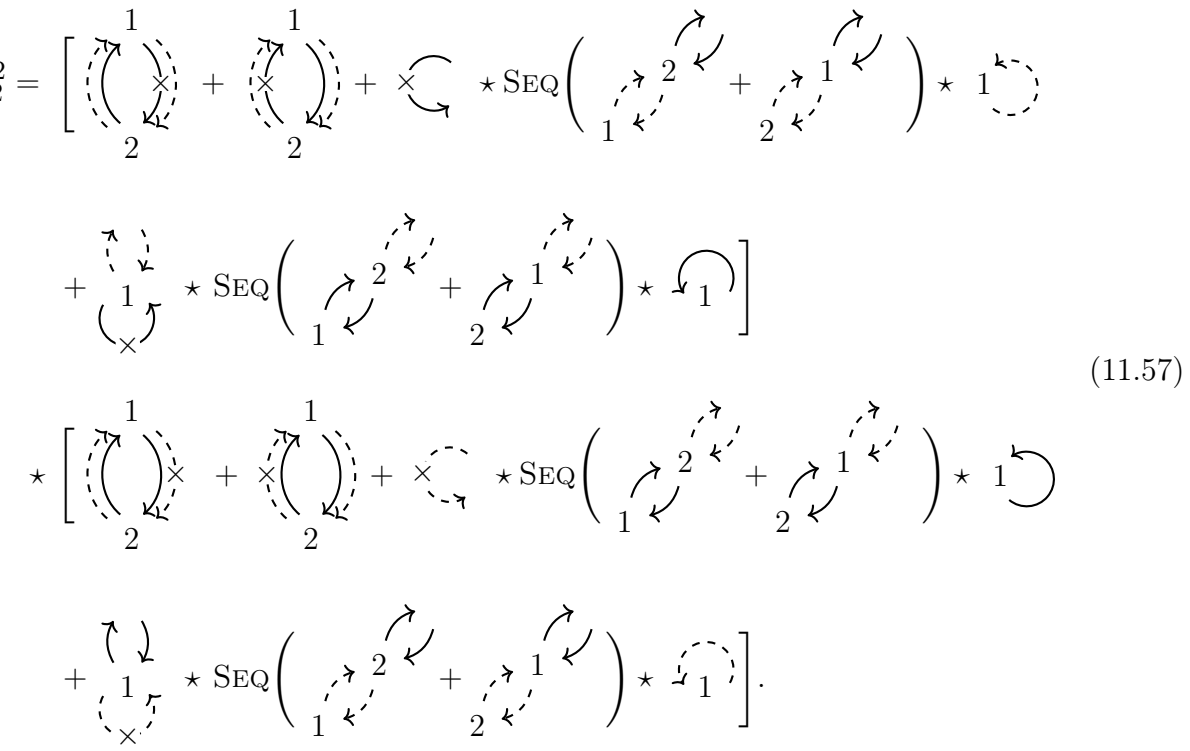
$\mathcal{C}_2^2 =$  $\star\,\textsc{Seq}\Big($  $+$  $\Big)\star$  . (11.56)

In terms of the EGF,

$$a_1 \left(\frac{1}{1 - 2(\mu_2+\nu_2)^2 \frac{t^2}{2!}}\right) a_2 t = \frac{a_1 a_2 t}{1-(\mu_2+\nu_2)^2 t^2}.$$

Pair of one-marked structures Finally, let $\mathcal{P}_2^2$ be the set of pairs of structures where one structure has an undashed arrow marked and the other structure has a dashed arrow

marked. In other words,



(11.57)

For its EGF, we have

$$\left[a_1\mu_3t^2 + a_1\left(\frac{1}{1-2(\mu_2+\nu_2)^2\frac{t^2}{2!}}\right)c_2t - a_1(\mu_2+\nu_2)t\left(\frac{1}{1-2(\mu_2+\nu_2)^2\frac{t^2}{2!}}\right)c_1t\right]$$
$$\left[a_2\mu_3t^2 + a_2\left(\frac{1}{1-2(\mu_2+\nu_2)^2\frac{t^2}{2!}}\right)c_1t - a_2(\mu_2+\nu_2)t\left(\frac{1}{1-2(\mu_2+\nu_2)^2\frac{t^2}{2!}}\right)c_2t\right],$$

which after simplification is equal to

$$a_1a_2\left(\mu_3t^2 + \frac{c_2t}{1-(\mu_2+\nu_2)^2t^2} - \frac{c_1(\mu_2+\nu_2)t^2}{1-(\mu_2+\nu_2)^2t^2}\right)$$
$$\times\left(\mu_3t^2 + \frac{c_1t}{1-(\mu_2+\nu_2)^2t^2} - \frac{c_2(\mu_2+\nu_2)t^2}{1-(\mu_2+\nu_2)^2t^2}\right).$$

Conclusion Adding the contributions together, since $\mathcal{H}_2^2 = (\mathcal{L}_2^2 + \mathcal{C}_2^2 + \mathcal{P}_2^2) \star \mathcal{H}_2^0$, we obtain that

$$H_2^2(t) = a_1 a_2 \left[\frac{2t^2\left(\mu_2 - \mu_2^2 t + \nu_2^2 t\right)}{1 - 2\mu_2 t + (\mu_2^2 - \nu_2^2)t^2} + \frac{t}{1-(\mu_2+\nu_2)^2 t^2} + \left(\mu_3 t^2 + \frac{c_2 t}{1-(\mu_2+\nu_2)^2 t^2} - \frac{c_1(\mu_2+\nu_2)t^2}{1-(\mu_2+\nu_2)^2 t^2} \right) \left(\mu_3 t^2 + \frac{c_1 t}{1-(\mu_2+\nu_2)^2 t^2} - \frac{c_2(\mu_2+\nu_2)t^2}{1-(\mu_2+\nu_2)^2 t^2} \right) \right] H_2^0(t).$$

11.2.2.4 Full generating function

Corollary 11.2.8. *Adding up the contributions, we obtain that for any distribution X_{ij}, Y_{ij} and Z_{ii},*

$$H_2(t, a_1, a_2) = H_2^0(t) + H_2^1(t) + H_2^2(t) = \left[1 + (a_1+a_2)\mu_3 t^2 + \frac{(a_1c_2 + a_2c_1)t}{1-(\mu_2+\nu_2)^2 t^2} - \frac{(\mu_2+\nu_2)(a_1c_1 + a_2c_2)t^2}{1-(\mu_2+\nu_2)^2 t^2} + a_1 a_2 \left(\frac{2t^2\left(\mu_2 - \mu_2^2 t + \nu_2^2 t\right)}{1 - 2\mu_2 t + (\mu_2^2 - \nu_2^2)t^2} + \frac{t}{1-(\mu_2+\nu_2)^2 t^2} + \left(\mu_3 t^2 + \frac{c_2 t}{1-(\mu_2+\nu_2)^2 t^2} - \frac{c_1(\mu_2+\nu_2)t^2}{1-(\mu_2+\nu_2)^2 t^2} \right) \left(\mu_3 t^2 + \frac{c_1 t}{1-(\mu_2+\nu_2)^2 t^2} - \frac{c_2(\mu_2+\nu_2)t^2}{1-(\mu_2+\nu_2)^2 t^2} \right) \right) \right] H_2^0(t).$$

11.3 Hermitian matrices

For a general Hermitian matrix, the real and imaginary parts of an off-diagonal entry need not be independent. For $i < j$, set

$$\lambda_{pq} = \mathbb{E}\left[Z_{ij}^p \overline{Z_{ij}}^q \right], \qquad \kappa_q = \mathbb{E}[Z_{ii}^q].$$

Then $\lambda_{pq} = \overline{\lambda_{qp}}$. In the marked second-moment calculation the base entries are centered, so $\lambda_{10} = \kappa_1 = 0$ and real mean shifts are carried by a_r and c_r. The first-moment warm-up temporarily retains λ_{10} and κ_1.

11.3.1 Eulerian numbers and polynomials

As we will see in the following subsections, there are new degrees of freedom from suppressing the condition that the real and imaginary parts of the matrix Z are independent. This increases the complexity of our generating functions. It is convenient to introduce here the so called *Eulerian numbers* $A(n,k)$, *Eulerian polynomials* $A_n(u)$, and their corresponding bivariate generating functions. Eulerian numbers can be defined via an explicit relation

$$A(n,k) = \sum_{i=0}^{n} (-1)^i \binom{n+1}{i} (k+1-i)^n \tag{11.58}$$

They appear ubiquitously in various combinatorial problems. For our purposes, we need to know that $A(n,k)$ counts the number of ways to order the numbers $[n] = \{1, 2, \ldots, n\}$ so that there are exactly k numbers which are greater than the previous number. As a consequence, $\sum_{k=0}^{n} A(n,k) = n!$, where by definition we take $A(n,n) = 0$ when $n \geq 1$. The Eulerian numbers may also be defined as the coefficients of the Eulerian polynomials

$$A_n(u) = \sum_{k=0}^{n} A(n,k) u^k, \tag{11.59}$$

which are themselves coefficients of the corresponding bivariate exponential generating function

$$E(u,t) = \sum_{n=0}^{\infty} A_n(u) \frac{t^n}{n!} = \frac{u-1}{u - e^{(u-1)t}}.$$

In our analysis, we will use the following bivariate generating function. The reason this bivariate generating function is useful is that it allows us to handle loops of length $n \geq 3$ where the weight of the loop depends on whether each edge is ascending or descending.

Definition 11.3.1. We define $B(u,t) = u \sum_{n=3}^{\infty} A_{n-1}(u) \frac{t^n}{n!}$.

Proposition 11.3.2. *Let $\mathcal{L}$ be the set of loops of length at least 3 where the vertices of each loop are labeled by the indices $[n]$ where n is the length of the loop.*

1. *If the weight of a loop $L \in \mathcal{L}$ of order n is*

$$w_L = (-1)^{n-1}\lambda^{|\{e=(j,j')\in E(L):j'>j\}|}\lambda'^{|\{e=(j,j')\in E(L):j'<j\}|},$$

so that ascending edges contribute λ, descending edges contribute λ', and the loop sign is $(-1)^{n-1}$, then the exponential generating function for $\mathcal{L}$ is $-B\left(\frac{\lambda}{\lambda'}, -\lambda' t\right)$.

2. *If the weight of a loop $L \in \mathcal{L}$ of order n is*

$$w_L = \lambda^{|\{e=(j,j')\in E(L):j'>j\}|}\lambda'^{|\{e=(j,j')\in E(L):j'<j\}|},$$

i.e., the same as the previous case but the sign of each loop is now positive, then the exponential generating function for $\mathcal{L}$ is $B\left(\frac{\lambda}{\lambda'}, \lambda' t\right)$.

Proof. Observe that in the first case, the exponential generating function for $\mathcal{L}$ is

$$\begin{aligned}\sum_{n=3}^{\infty}\sum_{k=0}^{n-2} A(n-1,k)\lambda^{k+1}\lambda'^{n-k-1}\frac{(-1)^{n-1}t^n}{n!} \\ = -\frac{\lambda}{\lambda'}\sum_{n=3}^{\infty} A_{n-1}\left(\frac{\lambda}{\lambda'}\right)\frac{(-\lambda' t)^n}{n!} = -B\left(\frac{\lambda}{\lambda'}, -\lambda' t\right).\end{aligned} \tag{11.60}$$

To see this, observe that for each loop $L \in \mathcal{L}$ of length n, we can consider n to be the starting and ending point of the loop. We are guaranteed to have a factor of $\lambda\lambda'$ from the two edges incident to n. The remaining edges are an ordering of $[n-1] = \{1, \ldots, n-1\}$ so for each $n \geq 3$ and $k \in [0, n-2]$, there are $A(n-1,k)$ loops of length n which have $k+1$ edges (j, j') where $j' > j$ and $n-k-1$ edges (j, j') where $j' < j$.

Following similar logic, in the second case the exponential generating function for $\mathcal{L}$ is

$$\sum_{n=3}^{\infty}\sum_{k=0}^{n-2} A(n-1,k)\lambda^{k+1}\lambda'^{n-k-1}\frac{t^n}{n!}$$

$$= \frac{\lambda}{\lambda'} \sum_{n=3}^{\infty} A_{n-1}\left(\frac{\lambda}{\lambda'}\right) \frac{(\lambda' t)^n}{n!} = B\left(\frac{\lambda}{\lambda'}, \lambda' t\right). \tag{11.61}$$

□

We can evaluate the bivariate generating function $B(u,t)$ by integrating $E(u,t)$ with respect to t.

Proposition 11.3.3. $B(u,t) = u\sum_{n=3}^{\infty} A_{n-1}(u)\frac{t^n}{n!} = \ln\left(\frac{1-u}{1-ue^{(1-u)t}}\right) - ut - \frac{ut^2}{2}$.

Proof.

$$\int E(u,t)dt = \sum_{n=1}^{\infty} A_{n-1}(u)\frac{t^n}{n!} = \frac{1}{u}\ln\left(\frac{1-u}{1-ue^{(1-u)t}}\right). \tag{11.62}$$

This implies that $t + \frac{t^2}{2} + \frac{1}{u}B(u,t) = \frac{1}{u}\ln\left(\frac{1-u}{1-ue^{(1-u)t}}\right)$. Rearranging and multiplying both sides of this equation by u, we have that $B(u,t) = \ln\left(\frac{1-u}{1-ue^{(1-u)t}}\right) - ut - \frac{ut^2}{2}$, as needed. □

Remark 11.3.4. While it is not obvious from this expression for $B(u,t)$, it is not hard to check that $B(\frac{\lambda}{\lambda'}, -\lambda' t) = B(\frac{\lambda'}{\lambda}, -\lambda t)$

11.3.2 First Moment

As a warm-up, we compute $\mathbb{E}[\det(Z)]$. For this computation, we do not assume that $\lambda_{10} = 0$ or $\kappa_1 = 0$ so we do not have a_1 or c_1 and we do not use marked tables or graphs. To indicate this, we write $\mathcal{F}_1$ and F_1 as we did in Section 11.1.2 rather than writing $\mathcal{H}_1$ and H_1.

Fixed points Let $\mathcal{X}_1$ be the set of fixed points in $\mathcal{F}_1$. Then

$$\mathcal{X}_1 = \textsc{Set}\left(1 \circlearrowleft\right). \tag{11.63}$$

In terms of the EGF, this gives a factor of $\exp(\kappa_1 t)$.

Mussels Next, let $\mathcal{M}_1$ be the set of mussels in $\mathcal{F}_1$. Then

$$\mathcal{M}_1 = \textsc{Set}\left(\circlearrowleft_{1}^{2}\right). \tag{11.64}$$

The EGF of the inner factor is $-\lambda_{11}t^2/2!$ so the EGF of $\mathcal{M}_1$ is $\exp(-\frac{1}{2}\lambda_{11}t^2)$.

Loops For the set of loops $\mathcal{L}_1$, observe that $\mathcal{L}_1 = \textsc{Set}(\mathcal{L})$ where $\mathcal{L}$ is the set of single loops of length $n \geq 3$ and the weight of a loop $L \in \mathcal{L}$ of order n is

$$w_L = (-1)^{n-1}\lambda^{|\{e=(j,j')\in E(L):j'>j\}|}\lambda'^{|\{e=(j,j')\in E(L):j'<j\}|}.$$

where $\lambda = \lambda_{10}$ and $\lambda' = \lambda_{01}$. By Theorem 11.3.2 and Theorem 11.3.3, the EGF for $\mathcal{L}$ is

$$-B\left(\frac{\lambda_{10}}{\lambda_{01}}, -\lambda_{01}t\right) = -\ln\left(\frac{\lambda_{01}-\lambda_{10}}{\lambda_{01}-\lambda_{10}e^{(\lambda_{10}-\lambda_{01})t}}\right) - \lambda_{10}t + \lambda_{10}\lambda_{01}\frac{t^2}{2}.$$

Exponentiating this gives the EGF for $\mathcal{L}_1$, which is

$$\frac{\lambda_{01}-\lambda_{10}e^{(\lambda_{10}-\lambda_{01})t}}{\lambda_{01}-\lambda_{10}}\exp\left(-\lambda_{10}t + \lambda_{10}\lambda_{01}\frac{t^2}{2}\right).$$

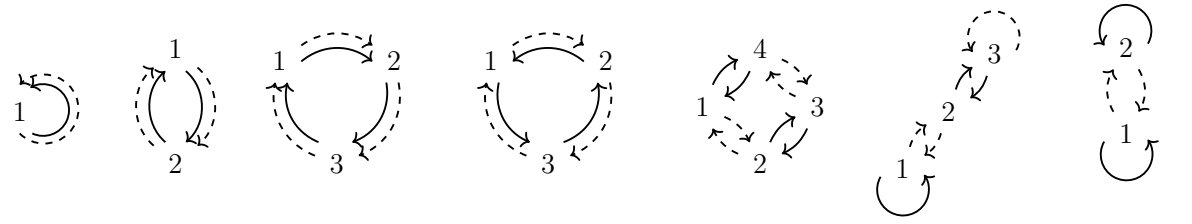
Conclusion Since $\mathcal{F}_1 = \mathcal{X}_1 \star \mathcal{M}_1 \star \mathcal{L}_1$, by multiplying the EGFs we obtain that

$$\begin{aligned} F_1(t) &= \exp(\kappa_1 t)\exp(-\tfrac{1}{2}\lambda_{11}t^2)\frac{\lambda_{01}-\lambda_{10}e^{(\lambda_{10}-\lambda_{01})t}}{\lambda_{01}-\lambda_{10}}\exp\left(-\lambda_{10}t+\lambda_{10}\lambda_{01}\frac{t^2}{2}\right) \\ &= \frac{e^{\frac{1}{2}t(2\kappa_1+(-2+t\lambda_{01})\lambda_{10}-t\lambda_{11})}\left(\lambda_{01}-e^{t(-\lambda_{01}+\lambda_{10})}\lambda_{10}\right)}{\lambda_{01}-\lambda_{10}}. \end{aligned}$$

11.3.3 Second moment

11.3.3.1 Zero marks

For the second moment, the Wigner case is very close to the more general Hermitian case described here. Hermitian multigraphs have the same connected structures as Wigner multigraphs. We no longer assume $\lambda_{20} = \lambda_{02}$ (both equal $\mu_2 - \nu_2$ in the Wigner case), however. This extra freedom changes the weights of loops whose solid and dashed arrows point in opposite directions; the other structures are obtained by replacing the real–imaginary moments with the corresponding λ's. It is therefore convenient to split $\mathcal{L}_2$ into $\mathcal{A}_2 \star \mathcal{O}_2$, where $\mathcal{A}_2$ denotes antiparallel loops and $\mathcal{O}_2$ denotes parallel loops. Table 11.6 shows representatives of all connected types.



Fixed point	Mussel	Parallel loop	Anti-parallel loop	Necklace	Chain	Dumbbell
$\mathcal{X}_2$	$\mathcal{M}_2$	$\mathcal{O}_2$	$\mathcal{A}_2$	$\mathcal{N}_2$	$\mathcal{C}_2$	$\mathcal{D}_2$

Table 11.6: Types of structures in $\mathcal{H}_2$ (Hermitian case)

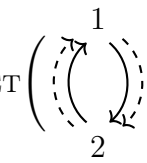
Fixed points Let $\mathcal{X}_2$ be the set of fixed points in $\mathcal{H}_2^0$. Then

$$\mathcal{X}_2 = \textsc{Set}\left(\begin{array}{c}1\\ \circlearrowleft\end{array}\right). \tag{11.65}$$

In terms of the EGF, this gives a factor of $\exp((c_1 c_2 + \kappa_2)t)$.

Mussels Next, let $\mathcal{M}_2$ be the set of mussels in $\mathcal{H}_2^0$. Then

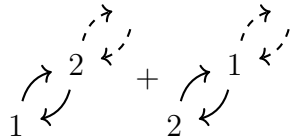
$$\mathcal{M}_2 = \textsc{Set}\left(\right). \tag{11.66}$$



The EGF of the inner factor is $\lambda_{22}t^2/2!$ so the EGF of $\mathcal{M}_2$ is $\exp(\frac{1}{2}\lambda_{22}t^2)$.

Necklaces For the set of necklaces $\mathcal{N}_2$,

$$\mathcal{N}_2 = \textsc{Set}\left(\frac{1}{2}\,\textsc{Cyc}_{\geq 2}\left(\right)\right). \tag{11.67}$$

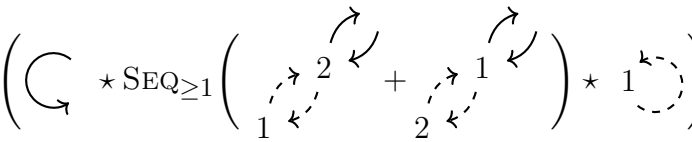


Hence, we have that its EGF is

$$\exp\left(\frac{1}{2}\left(-2\lambda_{11}^2\frac{t^2}{2!} + \ln\frac{1}{1-2\lambda_{11}^2\frac{t^2}{2!}}\right)\right) = \frac{e^{-\frac{1}{2}\lambda_{11}^2 t^2}}{\sqrt{1-\lambda_{11}^2 t^2}}. \tag{11.68}$$

Chains Let $\mathcal{C}_2$ be the set of all chains. We can write

$$\mathcal{C}_2 = \textsc{Set}\left(\star\,\textsc{Seq}_{\geq 1}\left(+ \right) \star \right). \tag{11.69}$$



In terms of the EGF,

$$\exp\left[c_1\left(-1+\frac{1}{1-2\lambda_{11}^2\frac{t^2}{2!}}\right)c_2 t\right] = \exp\left(\frac{c_1 c_2 \lambda_{11}^2 t^3}{1-\lambda_{11}^2 t^2}\right).$$

Dumbbells The set of dumbbells $\mathcal{D}_2$ can be constructed as

$$\mathcal{D}_2 = \textsc{Set}\left(\frac{1}{2} \; \circlearrowleft_1 \star \textsc{Seq}\left(\; \overset{1}{\underset{2}{\circlearrowright}} + \overset{2}{\underset{1}{\circlearrowright}} \; \right) \star \circlearrowleft_1 \right) \star \textsc{Set}\left(\frac{1}{2} \; \circlearrowleft_1 \star \textsc{Seq}\left(\; \overset{1}{\underset{2}{\circlearrowright}} + \overset{2}{\underset{1}{\circlearrowright}} \; \right) \star \circlearrowleft_1 \right). \tag{11.70}$$

Its EGF is thus equal to

$$\begin{aligned} &\left[\exp\left(-\frac{1}{2}c_1\lambda_{11}t\left(\frac{1}{1-2\lambda_{11}^2\frac{t^2}{2!}}\right)c_1 t\right)\right]\left[\exp\left(-\frac{1}{2}c_2\lambda_{11}t\left(\frac{1}{1-2\lambda_{11}^2\frac{t^2}{2!}}\right)c_2 t\right)\right] \\ &= \exp\left(-\frac{\frac{1}{2}(c_1^2+c_2^2)\lambda_{11}t^2}{1-\lambda_{11}^2t^2}\right). \end{aligned} \tag{11.71}$$

Anti-parallel Loops We can write

$$\mathcal{A}_2 = \textsc{Set}\left(\textsc{Cyc}_{\geq 3}\left(\; \overset{\curvearrowleft}{1} \; \right)\right). \tag{11.72}$$

As the element in the bracket has weight $W = \lambda_{11}t$, the corresponding EGF of $\mathcal{A}_2$ is

$$\exp\left(-\lambda_{11}t - \frac{\lambda_{11}^2t^2}{2} + \ln\frac{1}{1-\lambda_{11}t}\right) = \frac{e^{-\lambda_{11}t-\frac{1}{2}\lambda_{11}^2t^2}}{1-\lambda_{11}t}.$$

Parallel Loops For the set of parallel loops $\mathcal{O}_2$, observe that $\mathcal{O}_2 = \textsc{Set}(\mathcal{L})$ where each element $L \in \mathcal{L}$ is a single pair of parallel loops of length $n \geq 3$ and the weight of an element $L \in \mathcal{L}$ of order n is

$$w_L = (-1)^{n-1}\lambda^{|\{e=(j,j')\in E(L):j'>j\}|}\lambda'^{|\{e=(j,j')\in E(L):j'<j\}|}$$

where $\lambda = \lambda_{20}$ and $\lambda' = \lambda_{02}$. By Theorem 11.3.2 and Theorem 11.3.3, the EGF for $\mathcal{L}$ is

$$B\left(\frac{\lambda_{20}}{\lambda_{02}}, \lambda_{02}t\right) = \ln\left(\frac{\lambda_{02}-\lambda_{20}}{\lambda_{02}-\lambda_{20}e^{(\lambda_{02}-\lambda_{20})t}}\right) - \lambda_{20}t - \lambda_{20}\lambda_{02}\frac{t^2}{2} \tag{11.73}$$

Exponentiating this gives the EGF for $\mathcal{O}_2$, which is

$$\frac{\lambda_{02}-\lambda_{20}}{\lambda_{02}-\lambda_{20}e^{(\lambda_{02}-\lambda_{20})t}} \exp\left(-\lambda_{20}t - \lambda_{20}\lambda_{02}\frac{t^2}{2}\right).$$

Note that when $\lambda_{20} = \lambda_{02}$ (as a limit), this is equal to

$$\frac{1}{1-\lambda_{20}t} \exp\left(-\lambda_{20}t - \lambda_{20}^2\frac{t^2}{2}\right). \tag{11.74}$$

which is precisely the anti-parallel loop EGF with λ_{20} instead of λ_{11} as we would expect.

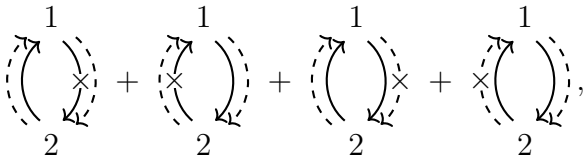
Conclusion As $\mathcal{H}_2^0 = \mathcal{X}_2 \star \mathcal{M}_2 \star \mathcal{O}_2 \star \mathcal{A}_2 \star \mathcal{N}_2 \star \mathcal{C}_2 \star \mathcal{D}_2$, by multiplying the EGFs we obtain that

$$H_2^0(t) = \frac{(\lambda_{02}-\lambda_{20}) \exp\left((c_1c_2+\kappa_2-\lambda_{11}-\lambda_{20})\,t + \frac{t^2}{2}\left(\lambda_{22}-2\lambda_{11}^2-\lambda_{02}\lambda_{20}\right) + \frac{c_1c_2\lambda_{11}^2t^3}{1-\lambda_{11}^2t^2} - \frac{t^2}{2}\frac{\left(c_1^2+c_2^2\right)\lambda_{11}}{1-\lambda_{11}^2t^2}\right)}{(1-\lambda_{11}t)\sqrt{1-\lambda_{11}^2t^2}\left(\lambda_{02}-\lambda_{20}e^{(\lambda_{02}-\lambda_{20})t}\right)}.$$

11.3.3.2 One mark

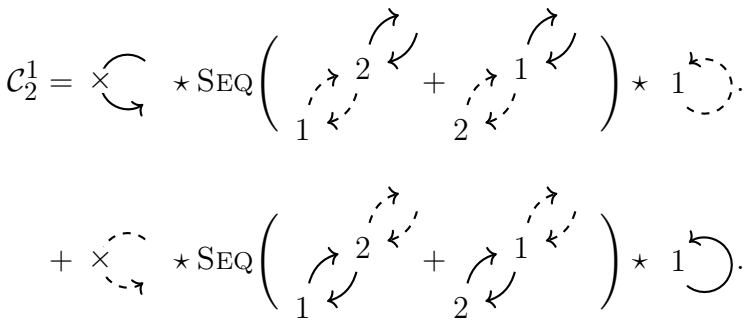
The one-marked structures are the same as in the case of Wigner matrices. There are 1-marked mussels, chains and dumbbells (see Table 11.4 in the section on Wigner matrices).

Marked mussels Let $\mathcal{M}_2^1$ be the set of 1-marked mussels. By adding up the contribution for the four different possibilities of marking, that is

$$\mathcal{M}_2^1 =$$  (11.75)

we obtain $(a_1 + a_2)(\lambda_{21} + \lambda_{12})t^2/2!$ for its EGF.

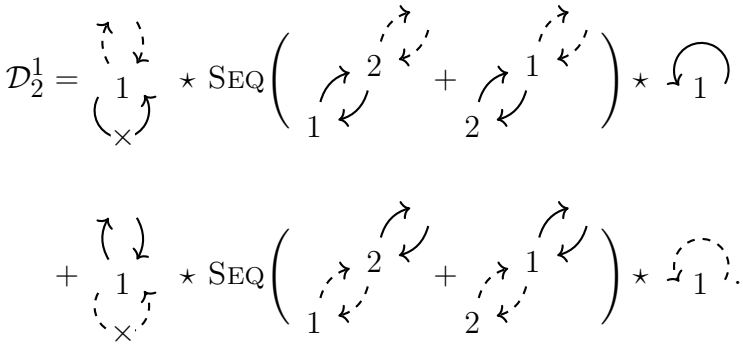
Marked chains For the set $\mathcal{C}_2^1$ of chains with one mark, we write

$$\mathcal{C}_2^1 =$$  (11.76)

In terms of the EGF,

$$a_1 \left(\frac{1}{1 - 2\lambda_{11}^2 \frac{t^2}{2!}} \right) c_2 t + a_2 \left(\frac{1}{1 - 2\lambda_{11}^2 \frac{t^2}{2!}} \right) c_1 t = \frac{(a_1 c_2 + a_2 c_1)t}{1 - \lambda_{11}^2 t^2}.$$

Marked dumbbells For the set $\mathcal{D}_2^1$ of all dumbbells with one mark,



(11.77)

Its EGF is thus equal to

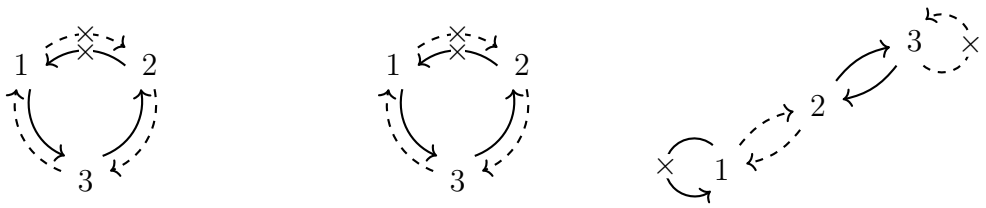
$$-a_1\lambda_{11}t\left(\frac{1}{1-2\lambda_{11}^2\frac{t^2}{2!}}\right)c_1t - a_2\lambda_{11}t\left(\frac{1}{1-2\lambda_{11}^2\frac{t^2}{2!}}\right)c_2t = -\frac{\lambda_{11}(a_1c_1+a_2c_2)t^2}{1-\lambda_{11}^2t^2}. \quad (11.78)$$

Conclusion As $\mathcal{H}_2^1 = (\mathcal{M}_2^1 + \mathcal{C}_2^1 + \mathcal{D}_2^1) \star \mathcal{H}_2^0$, we obtain that

$$H_2^1(t) = \left((a_1+a_2)(\lambda_{21}+\lambda_{12})\frac{t^2}{2} + \frac{(a_1c_2+a_2c_1)t}{1-\lambda_{11}^2t^2} - \frac{\lambda_{11}(a_1c_1+a_2c_2)t^2}{1-\lambda_{11}^2t^2}\right)H_2^0(t). \quad (11.79)$$

11.3.3.3 Two marks

The following Table 11.7 shows representatives of all types of the remaining (connected) structures having exactly two marks. Note that this table is the same as Table 11.5 in the Wigner case except for the fact we now distinguish between two types of loops.



2-Marked anti-parallel loop	2-Marked parallel loop	2-Marked chain
$\mathcal{O}_2^2$	$\mathcal{A}_2^2$	$\mathcal{C}_2^2$

Table 11.7: Types of structures in $\mathcal{H}_2^2$ with two marks (Hermitian case)

Remark 11.3.5. There are no marked dumbbells as they would violate the assumption that there is at most one mark per one type of arrow. There are, however, 2-marked fixed points (two marked 1-cycle), but they are again counted as 2-marked chains with one vertex. Similarly, 2-marked mussels (two 2-cycles with two marks) are counted as 2-marked loops with two vertices.

Marked anti-parallel loops The set of marked anti-parallel loops $\mathcal{A}_2^2$ (including mussels) is created as follows

$$\mathcal{A}_2^2 = \Bigg(\underset{1}{\circlearrowright} \star \textsc{Seq}_{\geq 1}\Bigg(\underset{1}{\circlearrowright} \Bigg)\Bigg). \tag{11.80}$$

Its EGF is then simply

$$a_1 a_2 t \left(-1 + \frac{1}{1-\lambda_{11} t}\right) = \frac{a_1 a_2 \lambda_{11} t^2}{1-\lambda_{11} t}.$$

Marked parallel loops We can analyze the set of marked parallel loops as follows. For a given marked parallel loop of length n with k increasing edges and $n-k$ decreasing edges, the weight before we mark a pair of parallel edges is $\lambda_{20}^k \lambda_{02}^{n-k}$

When we mark this parallel loop, if we mark an increasing edge (i.e., an edge j, j' such that $j' > j$) then this replaces a factor of λ_{20} with $a_1 a_2$. There are k ways to do this so the total contribution is $k a_1 a_2 \lambda_{20}^{k-1} \lambda_{02}^{n-k} = a_1 a_2 \frac{\partial\left(\lambda_{20}^k \lambda_{02}^{n-k}\right)}{\partial \lambda_{20}}$. Similarly, if we mark a decreasing

edge then this replaces a factor of λ_{02} with a_1a_2. There are $n-k$ ways to do this so the total contribution is $a_1a_2(n-k)\lambda_{20}^{k-1}\lambda_{02}^{n-k-1} = a_1a_2\frac{\partial\left(\lambda_{20}^k\lambda_{02}^{n-k}\right)}{\partial\lambda_{02}}$.

Thus, we can obtain the contribution from marking a parallel loop by computing the partial derivatives of its original contribution with respect to λ_{20} and λ_{02}, adding these partial derivates together, and multiplying the result by a_1a_2.

By Eq. (11.73), the EGF for a single pair of parallel loops of length at least 3 is

$$B\left(\frac{\lambda_{20}}{\lambda_{02}}, \lambda_{02}t\right) = \ln\left(\frac{\lambda_{02}-\lambda_{20}}{\lambda_{02}-\lambda_{20}e^{(\lambda_{02}-\lambda_{20})t}}\right) - \lambda_{20}t - \lambda_{20}\lambda_{02}\frac{t^2}{2}.$$

We now observe that

1. $\frac{\partial B\left(\frac{\lambda_{20}}{\lambda_{02}},\lambda_{02}t\right)}{\partial\lambda_{20}} = -\frac{1}{\lambda_{02}-\lambda_{20}} + \frac{(1-\lambda_{20}t)e^{(\lambda_{02}-\lambda_{20})t}}{\lambda_{02}-\lambda_{20}e^{(\lambda_{02}-\lambda_{20})t}} - t - \lambda_{02}\frac{t^2}{2}$.
2. $\frac{\partial B\left(\frac{\lambda_{20}}{\lambda_{02}},\lambda_{02}t\right)}{\partial\lambda_{02}} = \frac{1}{\lambda_{02}-\lambda_{20}} + \frac{-1+\lambda_{20}te^{(\lambda_{02}-\lambda_{20})t}}{\lambda_{02}-\lambda_{20}e^{(\lambda_{02}-\lambda_{20})t}} - \lambda_{20}\frac{t^2}{2}$.

Adding these contributions together and multiplying the result by a_1a_2 gives
$a_1a_2\left(\frac{e^{(\lambda_{02}-\lambda_{20})t}-1}{\lambda_{02}-\lambda_{20}e^{(\lambda_{02}-\lambda_{20})t}} - t - (\lambda_{20}+\lambda_{02})\frac{t^2}{2}\right)$.

We make one final adjustment to this expression. Since we want to include marked parallel loops of length 2 in $\mathcal{O}_2^2$, we add $a_1a_2(\lambda_{20}+\lambda_{02})\frac{t^2}{2}$ to the above expression. Thus, the EGF for $\mathcal{O}_2^2$ is

$$a_1a_2\left(\frac{e^{(\lambda_{02}-\lambda_{20})t}-1}{\lambda_{02}-\lambda_{20}e^{(\lambda_{02}-\lambda_{20})t}} - t\right).$$

Pair of one-marked structures Finally, let $\mathcal{P}_2^2$ be the set of pairs of structures one with an undashed arrow marked and the other with a dashed arrow marked. Similarly as in the

Wigner case,

$$\mathcal{P}_2^2 = \Bigg[\quad + \quad + \quad \star \textsc{Seq}\Big(\quad + \quad \Big) \star \quad + \quad \star \textsc{Seq}\Big(\quad + \quad \Big) \star \quad \Bigg] \tag{11.81}$$

$$\star \Bigg[\quad + \quad + \quad \star \textsc{Seq}\Big(\quad + \quad \Big) \star \quad + \quad \star \textsc{Seq}\Big(\quad + \quad \Big) \star \quad \Bigg].$$

For its EGF, we thus have

$$\left[a_1(\lambda_{21}+\lambda_{12})\frac{t^2}{2} + a_1 \left(\frac{1}{1-2\lambda_{11}^2 \frac{t^2}{2!}} \right) c_2 t - a_1 \lambda_{11} t \left(\frac{1}{1-2\lambda_{11}^2 \frac{t^2}{2!}} \right) c_1 t \right]$$
$$\left[a_2(\lambda_{21}+\lambda_{12})\frac{t^2}{2} + a_2 \left(\frac{1}{1-2\lambda_{11}^2 \frac{t^2}{2!}} \right) c_1 t - a_2 \lambda_{11} t \left(\frac{1}{1-2\lambda_{11}^2 \frac{t^2}{2!}} \right) c_2 t \right],$$

which is, after simplification, equal to

$$a_1 a_2 \left((\lambda_{21}+\lambda_{12})\frac{t^2}{2} + \frac{c_2 t}{1-\lambda_{11}^2 t^2} - \frac{c_1 \lambda_{11} t^2}{1-\lambda_{11}^2 t^2} \right)$$
$$\times \left((\lambda_{21}+\lambda_{12})\frac{t^2}{2} + \frac{c_1 t}{1-\lambda_{11}^2 t^2} - \frac{c_2 \lambda_{11} t^2}{1-\lambda_{11}^2 t^2} \right).$$

Conclusion Adding the contributions together, since $\mathcal{H}_2^2 = (\mathcal{A}_2^2 + \mathcal{O}_2^2 + \mathcal{C}_2^2 + \mathcal{P}_2^2) \star \mathcal{H}_2^0$,

$$H_2^2(t) = a_1 a_2 \Bigg[\frac{\lambda_{11} t^2}{1-\lambda_{11} t} + \frac{t}{1-\lambda_{11}^2 t^2} - t + \frac{e^{(\lambda_{02}-\lambda_{20})t} - 1}{\lambda_{02} - \lambda_{20} e^{(\lambda_{02}-\lambda_{20})t}} + \left((\lambda_{21}+\lambda_{12}) \frac{t^2}{2} + \frac{c_2 t}{1-\lambda_{11}^2 t^2} - \frac{c_1 \lambda_{11} t^2}{1-\lambda_{11}^2 t^2} \right) \times \left((\lambda_{21}+\lambda_{12}) \frac{t^2}{2} + \frac{c_1 t}{1-\lambda_{11}^2 t^2} - \frac{c_2 \lambda_{11} t^2}{1-\lambda_{11}^2 t^2} \right) \Bigg] H_2^0(t).$$

11.3.3.4 Full generating function

Corollary 11.3.6. *Adding up the contributions, we get for any distribution X_{ij} and Z_{ii},*

$$H_2(t, a_1, a_2) = H_2^0(t) + H_2^1(t) + H_2^2(t) = \Bigg[1 + (a_1+a_2)(\lambda_{21}+\lambda_{12}) \frac{t^2}{2} + \frac{(a_1 c_2 + a_2 c_1) t}{1-\lambda_{11}^2 t^2} - \frac{\lambda_{11}(a_1 c_1 + a_2 c_2) t^2}{1-\lambda_{11}^2 t^2} + a_1 a_2 \Bigg(\frac{\lambda_{11} t^2}{1-\lambda_{11} t} + \frac{t}{1-\lambda_{11}^2 t^2} - t + \frac{e^{(\lambda_{02}-\lambda_{20})t} - 1}{\lambda_{02} - \lambda_{20} e^{(\lambda_{02}-\lambda_{20})t}} + \Bigg((\lambda_{21}+\lambda_{12}) \frac{t^2}{2} + \frac{c_2 t}{1-\lambda_{11}^2 t^2} - \frac{c_1 \lambda_{11} t^2}{1-\lambda_{11}^2 t^2} \Bigg) \Bigg((\lambda_{21}+\lambda_{12}) \frac{t^2}{2} + \frac{c_1 t}{1-\lambda_{11}^2 t^2} - \frac{c_2 \lambda_{11} t^2}{1-\lambda_{11}^2 t^2} \Bigg) \Bigg) \Bigg] H_2^0(t).$$

11.4 Relation to the shell–core framework

The structured-ensemble problem and the arbitrary-mean sixth moment use different finite objects, but the same three design principles:

1. isolate connected exceptional structures;
2. sum arbitrary unmarked components by labeled set, cycle, and sequence constructions; and
3. group tables by their number of marks.

In the symmetric second moment, the exceptional marked components can be classified by

hand. In the general sixth moment, the number of row-support interactions is larger, and the shell serves as a finite interface to a universal core. The multigraph calculation therefore provides both a separate family of exact results and a simpler model of the architecture used in the dissertation's main shell–core contribution.

Part II

Pseudorandomness for Structured Computation

CHAPTER 12

PRELIMINARIES

This chapter consolidates the background that was originally spread across the two source papers. The first section introduces the most common notation and the computational models that recur throughout the dissertation. The remaining sections collect more specialized preliminaries on expanders, the INW generator, bounded-independence-type objects, decision trees, and Fourier-analytic tools.

12.1 Core definitions and notation

Definition 12.1.1 (Pseudorandom generators)**.** Let X be a distribution over $\{0,1\}^n$, let $f\colon \{0,1\}^n \to \{0,1\}$, and let $\varepsilon \in (0,1)$. We say that X *fools* f *with error* ε if

$$|\,\mathbb{E}[f(X)] - \mathbb{E}[f]| \leq \varepsilon.$$

We say that $G\colon \{0,1\}^s \to \{0,1\}^n$ is a *pseudorandom generator (PRG) that fools* f *with error* ε if $G(U_s)$ fools f with error ε. The parameter s is called the *seed length* of the PRG. If $\mathcal{F}$ is a class of Boolean functions on $\{0,1\}^n$, we say that X (respectively G) fools $\mathcal{F}$ with error ε if it fools every $f \in \mathcal{F}$ with error ε.

Definition 12.1.2 (Standard-order ROBP)**.** A width-w length-n standard-order *read-once branching program* (ROBP) is a directed acyclic multigraph whose vertices are arranged in layers $V_0, V_1, \ldots, V_n$, each of size w. Every vertex in a layer V_i with $i < n$ has two outgoing edges to V_{i+1} labeled 0 and 1. One vertex in V_0 is designated as the start vertex, and each vertex in V_n is labeled by an output bit. On input $x \in \{0,1\}^n$, the program starts at the start vertex and follows the edge labeled x_i in step i. The output is the label of the reached vertex in the final layer.

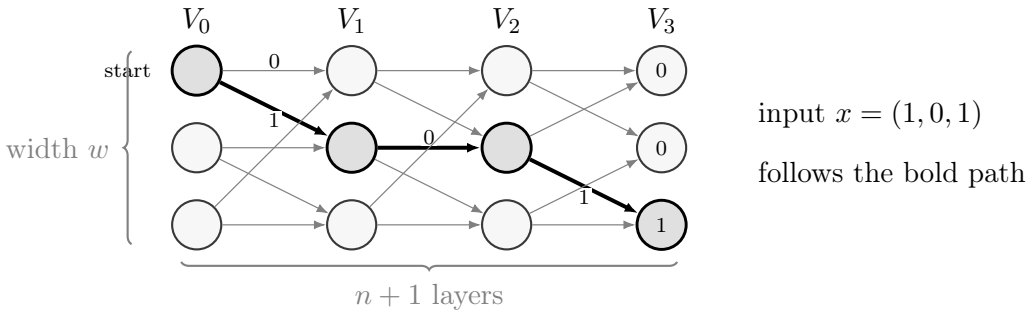


Figure 12.1: A width-3, length-3 standard-order ROBP. Each input bit selects one outgoing edge, so an input traces a single path from the start state to an output-labeled state. The bold path shows the computation on $x = (1, 0, 1)$.

Definition 12.1.3 (Decision trees)**.** An n-variate *decision tree* is a rooted tree in which each internal node queries one variable from among $x_1, \ldots, x_n$, each internal node has two outgoing edges labeled 0 and 1, and each leaf is labeled by an output bit. On input $x \in \{0,1\}^n$, evaluation begins at the root and follows the edge consistent with the queried variable until a leaf is reached. The *depth* is the maximum root-to-leaf path length, and the *size* is the number of leaves.

Definition 12.1.4 (Almost k-wise uniformity)**.** Let X be a distribution over $\{0,1\}^n$, let $k \in [n]$, and let $\varepsilon \in [0,1]$. We say that X is *ε-almost k-wise uniform* if, for every size-k set $S \subseteq [n]$, the total variation distance between X_S and the uniform distribution U_k on $\{0,1\}^k$ is at most ε.

12.2 Space-bounded computation, expanders, and codes

12.2.1 Read-once evaluation programs (ROEPs)

Ultimately, the model of computation we are interested in is the standard-order ROBP model (Theorem 12.1.2). At intermediate stages of our argument, we will use a slight generalization of the ROBP model called the "read-once evaluation program" (ROEP) model, introduced by Braveman, Rao, Raz, and Yehudayoff [BRRY14]. An ROEP is simply an ROBP with

fractional output values:

Definition 12.2.1 (Standard-order ROEP [BRRY14])**.** A width-w length-n standard-order *read-once evaluation program* (ROEP) is defined just like a width-w length-n standard-order ROBP (Theorem 12.1.2), except that the output values q_v are permitted to be any values in $[0, 1]$. Thus, the program computes a function $f\colon \{0,1\}^n \to [0,1]$. We say that a distribution X over $\{0,1\}^n$ fools f with error ε if $|\mathbb{E}[f(X)] - \mathbb{E}[f]| \leq \varepsilon$.

12.2.2 Graphs

For any graph H, we use $V(H)$ to denote the vertex set of H. As discussed in the introduction, we use the notation $H[x, y]$ to denote the y-th neighbor of the vertex x in the graph H. This is well-defined provided that H is a *labeled* graph, defined as follows.

Definition 12.2.2 (Graph labeling)**.** Let $H = (V, E)$ be a D-regular directed multigraph. We say that H is *labeled* if for every vertex x, the outgoing edges are labeled $1, \ldots, D$. In this case, we write $H[x, y]$ to denote the vertex reached from x by traversing the outgoing edge labeled y. If H is a D-regular undirected multigraph, then we identify H with the symmetric digraph obtained by replacing each undirected edge $\{x, x'\}$ with two directed edges: (x, x') and (x', x). If we say that H is "labeled," we allow the two edges (x, x') and (x', x) to have distinct labels.

We write K_R to denote the complete graph on R vertices without self-loops, and we write J_R to denote the complete graph on R vertices with self-loops. We write J or J_* if the number of vertices is clear from context. Several occurrences of "J_*" in a single equation might represent complete graphs on several different numbers of vertices.

If H is a D-regular undirected multigraph on R vertices, its *transition probability matrix* is the matrix $M \in [0,1]^{R \times R}$ defined by letting $M_{u,v} = \frac{e(u,v)}{D}$, where $e(u, v)$ denotes the number of edges from u to v. We often abuse notation by identifying a graph H with its

transition probability matrix. For example, we use J_R to denote the $R \times R$ matrix in which every entry is equal to $1/R$.

12.2.3 Spectral expansion

For a matrix $M \in \mathbb{R}^{R \times R}$, we use the notation $\|M\|_{\text{op}}$ to denote the *operator norm* of M, i.e.,

$$\|M\|_{\text{op}} = \max_{\substack{x \in \mathbb{R}^R \\ \|x\|_2 = 1}} \|xM\|_2.$$

Definition 12.2.3 (Expansion parameter)**.** Let H be a regular undirected multigraph. The *expansion parameter* $\lambda(H)$ is defined as

$$\lambda(H) = \|H - J\|_{\text{op}}.$$

Equivalently, one can define $\lambda(H)$ to be the second-largest eigenvalue of H in absolute value.

For example, $\lambda(J) = 0$. The complete graph without self-loops also has quite a good expansion parameter:

Fact 12.2.4 (Expansion parameter of the complete graph without self-loops)**.** *For every $R \in \mathbb{N}$, we have $\lambda(K_R) = 1/(R-1)$.*

Proof sketch. The following R vectors are linearly independent eigenvectors of K_R:

- The all-ones vector, which has eigenvalue 1;
- Vectors of the form $(1, 0, 0, \ldots, 0, -1, 0, 0, \ldots, 0)$, each of which has eigenvalue $-1/(R-1)$.

□

Cayley graphs are a more interesting class of expanders. The general definition is as follows.

Definition 12.2.5 (Cayley graph)**.** Let V be a group, let $\mathcal{R}$ be a finite set, and let $G\colon \mathcal{R} \to V$ be a function. The *Cayley graph* $\mathrm{Cay}(V, G)$ is a labeled $|\mathcal{R}|$-regular directed multigraph on the vertex set V defined by

$$\mathrm{Cay}(V, G)[x, y] = x \cdot G(y), \tag{12.1}$$

where $\cdot$ is the group operation.

We will only use this definition in the special case that $V = \mathbb{F}_2^n$, so Eq. (12.1) becomes

$$\mathrm{Cay}(\mathbb{F}_2^n, G)[x, y] = x \oplus G(y).$$

It is well-known that if G is a small-bias generator, then $\mathrm{Cay}(\mathbb{F}_2^n, G)$ is an expander. We include the proof for completeness' sake.

Lemma 12.2.6 (Expanders from small-bias generators)**.** *Let $n \in \mathbb{N}$ and let $G\colon \mathcal{R} \to \mathbb{F}_2^n$ be a λ-biased generator. Then $\lambda(\mathrm{Cay}(\mathbb{F}_2^n, G)) \leq \lambda$.*

Proof. Let A be the $2^n \times 2^n$ transition probability matrix of $\mathrm{Cay}(\mathbb{F}_2^n, G)$. Then

$$A = \frac{1}{|\mathcal{R}|} \sum_{y \in \mathcal{R}} A^{(y)},$$

where $A^{(y)}_{x,x'} = 1$ if $x' = x \oplus G(y)$ and 0 otherwise.

The 2^n eigenvectors of $A^{(y)}$ are the *character functions* χ_a (viewed as vectors indexed by $\mathbb{F}_2^n$). Indeed,

$$(\chi_a A^{(y)})_x = \sum_{x'} \chi_a(x')\, A^{(y)}_{x',x} = \chi_a(x \oplus G(y)) = \chi_a(x) \cdot \chi_a(G(y)),$$

because the only x' for which $A^{(y)}(x', x)$ is nonzero is $x' = x \oplus G(y)$. Hence χ_a is an eigenvector of $A^{(y)}$ with eigenvalue $\chi_a(G(y))$, and by linearity χ_a is an eigenvector of A with

eigenvalue

$$\lambda_a = \frac{1}{|\mathcal{R}|}\sum_{y\in\mathcal{R}} \chi_a(G(y)) = \mathbb{E}_{y\in\mathcal{R}}[\chi_a(G(y))].$$

Then we have all 2^n eigenvalues of A. When $a = 0$, one finds $\lambda_0 = 1$, and so

$$\lambda\big(\mathrm{Cay}(\mathbb{F}_2^n, G)\big) = \max_{a\neq 0}\big|\mathbb{E}_{y\in\mathcal{R}}[\chi_a(G(y))]\big|.$$

Since G is an λ-biased generator, $\big|\mathbb{E}_{y\sim\mathcal{R}}\big[\chi_a(G(y))\big]\big| \leq \lambda$, for all nonzero $a \in \mathbb{F}_2^n$. □

12.2.4 Lower bound on the degree of expander graphs

To prove our seed length lower bounds, we rely on the following standard fact. We include a proof for completeness' sake.

Proposition 12.2.7 (Expander graph degree lower bound)**.** *Let H be an undirected D-regular graph on R vertices. Then*

$$\lambda(H) \geq \sqrt{\frac{1}{D}\cdot\frac{R-D}{R-1}}.$$

In particular, $D \geq \min\{1/(2\cdot\lambda(H)^2), (R+1)/2\}$, and if $D = 1$, then $\lambda(H) = 1$.

Proof. Let H^2 denote the square of the transition probability matrix of H. Every entry on the diagonal of H^2 is at least $1/D$, because H is undirected. Therefore, $\mathrm{Tr}(H^2) \geq R/D$. On the other hand,

$$\mathrm{Tr}(H^2) = \sum_{i=1}^{R}\lambda_i^2 \leq 1 + (R-1)\cdot\lambda(H)^2,$$

where λ_i's are the eigenvalues of H, such that $1 = \lambda_1^2 \geq \lambda_2^2 \geq \cdots \geq \lambda_R^2$ and $\lambda(H) = \lambda_2$. Combining these together, we get

$$\lambda(H) \geq \sqrt{\frac{1}{D}\cdot\frac{R-D}{R-1}}.$$

It immediately follows that if $D = 1$, then $\lambda(H) = 1$. Finally, if $D < \min\{1/(2\cdot\lambda(H)^2), (R+$

1)/2}, then

$$\lambda(H)^2 \geq \frac{1}{D} \cdot \frac{R-D}{R-1} > 2 \cdot \lambda(H)^2 \cdot \frac{R - \frac{R+1}{2}}{R-1} = \lambda(H)^2,$$

which is a contradiction. □

12.2.5 Tensor products

Definition 12.2.8 (Tensor product of graphs)**.** Given a pair of labeled graphs H_1, H_2 on R_1, R_2 vertices with degrees D_1, D_2 respectively, define the tensor product $H_1 \otimes H_2$ to be the $(D_1 \cdot D_2)$-regular graph on $R_1 \cdot R_2$ vertices with neighbor relation $(H_1 \otimes H_2)[(u_1, u_2), (e_1, e_2)] = (H_1[u_1, e_1], H_2[u_2, e_2])$.

Proposition 12.2.9 (Spectral expansion of a tensor product)**.** *Let H_1, H_2 be undirected regular graphs. Then $\lambda(H_1 \otimes H_2) = \max(\lambda(H_1), \lambda(H_2))$.*

We also use the notation $M \otimes M'$ to denote the tensor product of matrices, aka the *Kronecker product*.

Fact 12.2.10 (Operator norm of tensor product)**.** *For any two matrices M, M', we have*

$$\|M \otimes M'\|_{\text{op}} = \|M\|_{\text{op}} \cdot \|M'\|_{\text{op}}.$$

12.2.6 Expander mixing lemma

We will use the following weak version of the famous "expander mixing lemma."

Lemma 12.2.11 (Expander mixing lemma)**.** *Let $H = (V, E)$ be a regular undirected multigraph. Let $f, g \colon V \to \{0, 1\}$. Sample a uniform random vertex X, then sample a uniform random neighbor Y of X. Then*

$$|\mathbb{E}[f(X) \cdot g(Y)] - \mathbb{E}[f] \cdot \mathbb{E}[g]| \leq \lambda(H).$$

For a proof, see, e.g., Vadhan's pseudorandomness survey [Vad12].

12.2.7 INW generators

We now present a more precise definition of the INW generator, in case the informal definition in the introduction was not sufficiently clear.

Definition 12.2.12 (Permissible graph families)**.** Let n be a power of two, let $D_1, \dots, D_{\log n} \in \mathbb{N}$, let H_i be a labeled D_i-regular undirected multigraph for every $i \in [\log n]$, and let $\vec{H} = (H_1, \dots, H_{\log n})$. We say that $\vec{H}$ is *permissible* if $V(H_1) = [2]$ and $V(H_{i+1}) = V(H_i) \times [D_i]$ for every $i \in [\log n - 1]$, where $V(H)$ denotes the vertex set of H.

More generally, suppose $\mathcal{H}$ is a $t \times \log n$ matrix of labeled regular undirected multigraphs:

$$\mathcal{H} = \begin{bmatrix} H_{1,1} & \dots & H_{1,\log n} \\ \vdots & \ddots & \vdots \\ H_{t,1} & \dots & H_{t,\log n} \end{bmatrix}.$$

We say that $\mathcal{H}$ is *permissible* if each row is permissible.

Definition 12.2.13 (INW generators)**.** Let $\vec{H} = (H_1, \dots, H_{\log n})$ be a permissible family of labeled regular undirected multigraphs. We define $\mathrm{INW}_{\vec{H}} \colon V(H_{\log n}) \to \{0,1\}^n$ recursively by the formulas

$$\begin{aligned} \mathrm{INW}_{()}(x) &= x \\ \mathrm{INW}_{(H_1,;H_j)}(x,y) &= (\mathrm{INW}_{(H_1,\dots,H_{j-1})}(x), \mathrm{INW}_{(H_1,\dots,H_{j-1})}(H_j[x,y])). \end{aligned}$$

More generally, let $\mathcal{H}$ be a $t \times \log n$ matrix of labeled regular undirected multigraphs, say

$$\mathcal{H} = \begin{bmatrix} H_{1,1} & \dots & H_{1,\log n} \\ \vdots & \ddots & \vdots \\ H_{t,1} & \dots & H_{t,\log n} \end{bmatrix},$$

and assume that $\mathcal{H}$ is permissible. We define $\mathrm{INW}_{\mathcal{H}}^{\oplus t} \colon V(H_{1,\log n}) \times \cdots \times V(H_{t,\log n}) \to \{0,1\}^n$ by the formula

$$\mathrm{INW}_{\mathcal{H}}^{\oplus t}(x_1, \dots, x_t) = \mathrm{INW}_{\mathcal{H}_1}(x_1) \oplus \cdots \oplus \mathrm{INW}_{\mathcal{H}_t}(x_t),$$

where $\mathcal{H}_1, \dots, \mathcal{H}_t$ are the rows of $\mathcal{H}$.

For a vector $\vec{\lambda} \in [0,1]^{\log n}$, we define

$$\mathrm{INW}(\vec{\lambda}) = \left\{ \mathrm{INW}_{\vec{H}} \colon \vec{H} \text{ is a } 1 \times \log n \text{ permissible family and } \lambda(H_j) \le \lambda_j \text{ for all } j \right\}.$$

When n is clear from context, we use $\mathrm{INW}(\lambda)$ as a shorthand for the case that $\lambda_j = \lambda$ for every j. Similarly, for a matrix $\Lambda \in [0,1]^{t \times \log n}$, we define

$$\mathrm{INW}^{\oplus t}(\Lambda) = \left\{ \mathrm{INW}_{\mathcal{H}} \colon \mathcal{H} \text{ is a } t \times \log n \text{ permissible family and } \lambda(H_{i,j}) \le \Lambda_{i,j} \text{ for all } i,j \right\}.$$

When n is clear from context, we use $\mathrm{INW}^{\oplus t}(\lambda)$ as a shorthand for the case that $\Lambda_{i,j} = \lambda$ for every i, j.

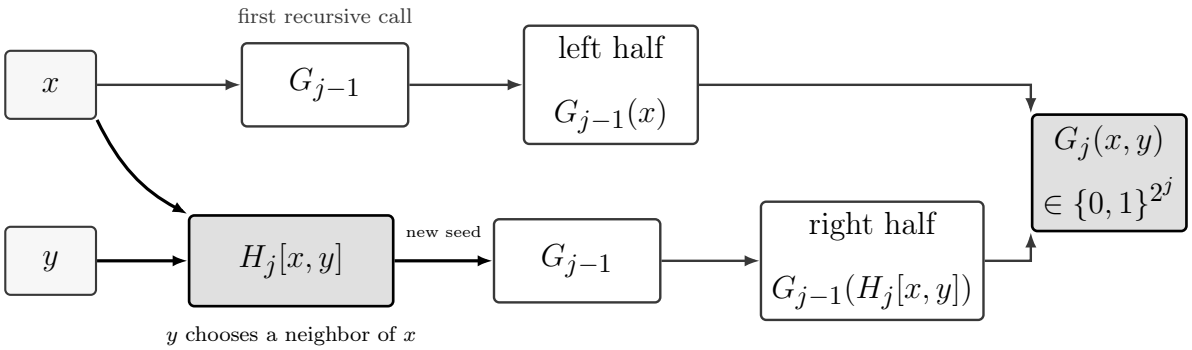


Figure 12.2: One recursive step of the INW generator. The first half uses the old seed x directly. The expander edge indexed by y produces the correlated seed $H_j[x, y]$ for the second half, allowing the generator to double its output while adding only the neighbor label y to its seed.

12.2.8 Binary linear code

In this section we briefly review the basic concepts of the coding theory.

Definition 12.2.14 (Binary Linear Code)**.** A *binary linear code* $\mathcal{C}$ of block length m is a subspace of $\mathbb{F}_2^m$, where $\mathbb{F}_2$ is the field with two elements. The *minimum distance* d of $\mathcal{C}$ is the smallest Hamming weight among all nonzero codewords in $\mathcal{C}$.

Definition 12.2.15 (Binary Entropy Function)**.** For $0 \leq \delta \leq 1$, the binary entropy function is defined as

$$H(\delta) = -\delta \log_2(\delta) - (1-\delta)\log_2(1-\delta),$$

with the convention that $0 \log_2 0 = 0$.

Theorem 12.2.16 (GV bound for binary linear codes)**.** *For every $m, k \in \mathbb{N}$ such that $k \leq m/2$, there exists a binary linear code of block length m with minimum distance $k+1$ and dimension at least $\lfloor (1 - H(k/m)) \cdot m \rfloor$.*

Since Theorem 12.2.16 is perhaps not the most common formulation of the GV bound, we include a proof for completeness' sake in Chapter C.

Corollary 12.2.17. *For every $\delta \in (0, 1/2)$, there is a subspace $\mathcal{C}^\perp \subseteq \mathbb{F}_2^m$ of dimension at most $\lceil H(\delta) \cdot m \rceil$ such that the uniform distribution over $\mathcal{C}^\perp$ is (δm)-wise uniform distribution, which means every $\lfloor \delta m \rfloor$ bits of this distribution are uniform over $\{0,1\}^{\lfloor \delta m \rfloor}$.*

Proof. Let $k = \lfloor \delta m \rfloor$. By Theorem 12.2.16, there exists a code $\mathcal{C}$ with minimum distance $k+1$ and dimension at least $\lfloor (1 - H(k/m)) \cdot m \rfloor$. It is well-known that in general, if a linear code has minimum distance $k+1$, then the uniform distribution over the dual code is k-wise uniform. (For a proof, see, e.g., Hatami and Hoza's survey [HH24].) Finally, we have

$$\begin{aligned}
\dim(\mathcal{C}^\perp) = m - \dim(\mathcal{C}) &\leq m - \lfloor (1 - H(k/m)) \cdot m \rfloor \\
&= \lceil m - (1 - H(k/m)) \cdot m \rceil \\
&= \lceil H(k/m) \cdot m \rceil \\
&\leq \lceil H(\delta) \cdot m \rceil. \qquad \square
\end{aligned}$$

Remark 12.2.18. Guruswami [Shp09] explicitly constructs a family of constant-rate binary linear codes whose primal and dual relative minimum distance are both constant with efficient decoding and encoding procedures. For our purposes, the existence of such family of codes is enough.

12.3 Decision trees, bounded independence, and Fourier tools

12.3.1 Decision tree models

Below we record the standard definitions of a decision tree and parity decision trees.

Definition 12.3.1 (Decision trees)**.** An n-variate *decision tree* is a rooted tree T in which each internal node is labeled with a variable from among $x_1, \ldots, x_n$; each internal node has two outgoing edges labeled 0 and 1; and each leaf is labeled either 0 or 1. The tree T computes a Boolean function $T \colon \{0,1\}^n \to \{0,1\}$ defined inductively as follows. If T consists

of a single leaf labeled $b \in \{0,1\}$, then we define $T(x) \equiv b$. Otherwise, let x_i be the variable labeling the root node. Given an input $x \in \{0,1\}^n$, we start at the root node and traverse the outgoing edge labeled with the value x_i. This leads to a vertex u, which is the root of a subtree T'. Then we set $T(x) = T'(x)$. The *depth* of the tree is the length of the longest path from the root to a leaf. The *size* of the tree is the total number of leaves.

Definition 12.3.2 (Parity decision trees [KM93])**.** A *parity decision tree* on variables $x_1, \ldots, x_n$ is a rooted tree T defined exactly as in Theorem 12.1.3, except that each internal node is labeled by a non-empty subset $S \subseteq [n]$. The internal node queries $\bigoplus_{i \in S} x_i$ and has two outgoing edges labeled 0 and 1 corresponding to the value of that parity. Leaves are labeled by output bits in $\{0,1\}$, and evaluation proceeds exactly as for ordinary decision trees. The *depth* of a parity decision tree is the length of the longest root-to-leaf path, and its *size* is the number of leaves. Equivalently, a parity decision tree computes a Boolean function

$$f(x_1, \ldots, x_n) = T\left(\bigoplus_{i \in S_1} x_i, \ldots, \bigoplus_{i \in S_m} x_i\right),$$

where T is an ordinary decision tree on m inputs and $S_1, \ldots, S_m \subseteq [n]$. computation.

12.3.2 Pairwise uniform hashing

We rely on the standard notion of a *pairwise uniform hashing*, aka "strongly universal hashing," introduced in Carter and Wegman's seminal papers [CW79, WC81].

Definition 12.3.3 (Pairwise uniform families of hash functions)**.** A family $\mathcal{H}$ of hash functions $h\colon \{0,1\}^q \to \{0,1\}^\ell$ is called *pairwise uniform* if, for every two distinct $x, x' \in \{0,1\}^q$, if we sample $h \sim \mathcal{H}$, then $(h(x), h(x'))$ is distributed uniformly at random over $\{0,1\}^{2\ell}$.

Theorem 12.3.4 (Explicit pairwise uniform families of hash functions)**.** *For every $q, \ell \in \mathbb{N}$,*

there exists an explicit[1] *pairwise uniform family* $\mathcal{H}$ *of hash functions* $h\colon \{0,1\}^q \to \{0,1\}^\ell$ *such that* $h \in \mathcal{H}$ *can be sampled using a seed of length* $O(q+\ell)$.

For example, if we define $h_{a,b}(x) = a * x + b$, where $*$ is convolution mod 2 and $+$ is bitwise XOR, then $\{h_{a,b} : a \in \{0,1\}^{q+\ell-1}$ and $b \in \{0,1\}^\ell\}$ is a pairwise uniform family [MNT93]. The reason pairwise uniform hashing is useful for us is given by the following relative-error sampling lemma.

Lemma 12.3.5 (Pairwise uniformity sampling lemma)**.** *Let* $\mathcal{H}$ *be a pairwise uniform family of hash functions* $h\colon \{0,1\}^q \to \{0,1\}^\ell$. *Let* $f\colon \{0,1\}^\ell \to \{0,1\}$ *and let* $\mu = \mathbb{E}[f]$. *Then for every* $\varepsilon \in (0,1)$,

$$\Pr_{h\sim\mathcal{H}}[h \text{ fools } f \text{ with error } \varepsilon\cdot\mu] \geq 1 - \frac{1}{2^q\cdot\varepsilon^2\cdot\mu}.$$

Proof. For each $x \in \{0,1\}^q$, define $Z_x = f(h(x))$, so Z_x is a random variable based on the choice of $h \sim \mathcal{H}$. Then $\mathbb{E}[Z_x] = \mu$ and $\mathrm{Var}[Z_x] = \mu\cdot(1-\mu) \leq \mu$. Furthermore, the variables Z_x are pairwise independent. Therefore, if we let $Z = \sum_x Z_x$, then $\mathbb{E}[Z] = 2^q\cdot\mu$ and $\mathrm{Var}[Z] \leq 2^q\cdot\mu$. Therefore, by Chebyshev's inequality, we have

$$\Pr[|Z - 2^q\cdot\mu| \geq \varepsilon\cdot\mu\cdot 2^q] \leq \frac{\mathrm{Var}[Z]}{\varepsilon^2\cdot\mu^2\cdot 2^{2q}} \leq \frac{1}{2^q\cdot\varepsilon^2\cdot\mu}. \qquad \square$$

12.3.3 Small-bias distributions

We also rely on asymptotically optimal constructions of k-wise γ-biased generators, introduced earlier in this dissertation.

Theorem 12.3.6 (Explicit k-wise γ-biased generators [NN93])**.** *For every* $n, k \in \mathbb{N}$ *and every* $\gamma \in (0,1)$, *there exists an explicit* k*-wise* γ*-biased generator* $G\colon \{0,1\}^s \to \{0,1\}^n$ *with seed length* $O(\log(k/\gamma) + \log\log n)$.

1. That is, given a seed $x \in \{0,1\}^{O(q+\ell)}$ and an input $y \in \{0,1\}^q$, the value $h_x(q)$ can be computed in $\mathrm{poly}(q,\ell)$ time, where h_x is the hash function corresponding to the seed x.

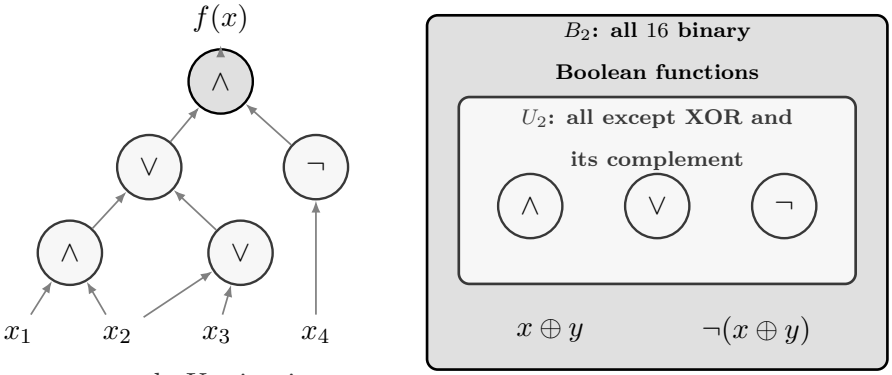
The reason k-wise γ-biased generators are useful for us is that they satisfy the following two properties.

Lemma 12.3.7 (Small-bias generators fool juntas and conjunctions of literals [NN93, AGHP92])**.** *Let X be a k-wise γ-biased distribution over $\{0,1\}^n$. Then X is ε-almost k-wise uniform, where $\varepsilon = \gamma \cdot 2^{k/2}$. Furthermore, X fools every conjunction of at most k literals with error γ.*

12.3.4 Parity circuits

To construct examples showing the weakness of k-wise γ-biased generators, we will rely on circuits computing the parity function.

Definition 12.3.8 (B_2- and U_2-circuits)**.** A *B_2-circuit* is a Boolean circuit in which every gate computes an arbitrary function $\phi\colon \{0,1\}^2 \to \{0,1\}$. A *$U_2$-circuit* is defined in the same way, except that a gate may not compute XOR or its complement. Thus, every U_2-circuit is a B_2-circuit. We measure size by the number of non-input gates.



an example U_2 circuit

(and hence a B_2 circuit)

Figure 12.3: Left: a small circuit using AND, OR, and NOT gates, hence a U_2-circuit. Right: the containment of gate bases. The B_2 basis contains every binary Boolean function, while U_2 excludes exactly XOR and its complement.

Proposition 12.3.9 (Parity circuits)**.** *For any integer $n \geq 2$, the function $f(x_1, \ldots, x_n) = x_1 \oplus x_2 \oplus \cdots \oplus x_n$ can be computed by a U_2-circuit of size $3n - 3$.*

Proof. When $n = 2$, we have $x_1 \oplus x_2 = (x_1 \wedge \overline{x}_2) \vee (\overline{x}_1 \wedge x_2)$. When $n > 2$, we perform a tree of binary $\oplus$ operations, each of which can be computed using three gates. □

12.3.5 Fourier analysis of Boolean functions

Our seed length lower bound for fooling decision trees using small-bias distributions uses Fourier analysis. For a set $S \subseteq [n]$, we use the notation $\chi_S \colon \{0,1\}^n \to \mathbb{R}$ to denote the function $\chi_S(x) = \prod_{i \in S} (-1)^{x_i}$. For a function $f \colon \{0,1\}^n \to \mathbb{R}$, we use the notation $\widehat{f}(S)$ to denote the Fourier coefficient of f at S:

$$\widehat{f}(S) = \mathop{\mathbb{E}}_{x \in \{0,1\}^n} [f(x) \cdot \chi_S(x)].$$

Parseval's theorem states that

$$\mathop{\mathbb{E}}_{x \in \{0,1\}^n} [f(x)^2] = \sum_{S \subseteq [n]} \widehat{f}(S)^2.$$

CHAPTER 13

SUMS OF INW PSEUDORANDOM GENERATORS

This chapter is based on the paper *On Sums of INW Pseudorandom Generators* [HL25b]. It studies a new way of using the Impagliazzo–Nisan–Wigderson construction to fool constant-width standard-order ROBPs. Rather than tuning a single instance of the INW generator ever more aggressively, we consider the bitwise XOR of several INW outputs and ask how much pseudorandomness amplification can be obtained from this operation.

13.1 Motivation and main results

The INW generator is one of the canonical tools in pseudorandomness for space-bounded computation. Its recursive structure makes it natural to ask whether "mild" instances of INW can be combined into a stronger object. The starting point for this chapter is the observation that, although instantiations of INW with relatively large expansion parameter λ are too weak to fool all constant-width ROBPs, such generators still have meaningful unpredictability properties for the next input bit. This suggests an XOR amplification approach in the spirit of Yao's XOR lemma.

The first theorem stated below captures the main obstruction to simply lowering the seed length of a single INW generator while reasoning only from spectral expansion.

Theorem 13.1.1 (INW generator limitations [BV10b, HPV24])**.** *Let* $\vec{\lambda} = (\lambda_1, \ldots, \lambda_{\log n}) \in [0,1]^{\log n}$*. If every PRG in the family* $\mathrm{INW}(\vec{\lambda})$ *fools width-*3 *standard-order ROBPs with error* 0.99*, then* $\lambda_j \leq 1/n^{\Omega(1)}$ *for* $\Omega(\log n)$ *values of* j*, and moreover every PRG in the family* $\mathrm{INW}(\vec{\lambda})$ *has seed length* $\Omega(\log^2 n)$*.*

In response, we define a family obtained by XORing several independent INW outputs.

Definition 13.1.2 (XOR of INW generators)**.** Let $n, t \in \mathbb{N}$ where n is a power of two, and let $\Lambda \in [0,1]^{t \times \log n}$. Let $\mathrm{INW}^{\oplus t}(\Lambda)$ denote the set of all PRGs $G\colon \mathcal{R}_1 \times \cdots \times \mathcal{R}_t \to \{0,1\}^n$

of the form

$$G(z^{(1)}, \ldots, z^{(t)}) = G^{(1)}(z^{(1)}) \oplus \cdots \oplus G^{(t)}(z^{(t)}),$$

where $G^{(i)} \in \mathrm{INW}(\Lambda_i)$ for every $i \in [t]$. Here Λ_i denotes the i-th row of Λ. As a shorthand, we also write $\mathrm{INW}^{\oplus t}(\lambda)$ if every entry of Λ is equal to λ and n is clear from context.

Our main positive result shows that this XOR operation genuinely strengthens the INW paradigm for constant-width ROBPs.

Theorem 13.1.3 ($\mathrm{INW}^{\oplus t}$ fools constant-width ROBPs)**.** *Let $n, w, t \in \mathbb{N}$ where n is a power of two, and let $\Lambda \in [0,1]^{t \times \log n}$. Every PRG in the family $\mathrm{INW}^{\oplus t}(\Lambda)$ fools width-w length-n standard-order ROBPs with error*

$$n^{\log(w+1)} \cdot \prod_{i=1}^{t} \sum_{j=1}^{\log n} \Lambda_{i,j}.$$

This theorem recovers the classical $O(\log^2 n)$ seed-length bound for constant-width ROBPs from a new angle. It is therefore natural to ask whether one can obtain an actual improvement by taking fewer copies with milder expanders. The next theorem shows that this is impossible within the present paradigm.

Theorem 13.1.4 (Limitations of $\mathrm{INW}^{\oplus t}$)**.** *Let n be a power of two, let $t \in \mathbb{N}$, and let $\Lambda \in [0,1]^{t \times \log n}$. If every PRG in the family $\mathrm{INW}^{\oplus t}(\Lambda)$ fools width-3 standard-order ROBPs with error 0.99, then every PRG in the family $\mathrm{INW}^{\oplus t}(\Lambda)$ has seed length $\Omega(\log^2 n)$.*

13.2 Increasing t versus decreasing λ

A natural interpretive question is whether XORing several mild INW generators is simply a disguised way of simulating one stronger INW generator with a smaller expansion parameter. The answer is no: the two strengthening strategies are incomparable in general. We record the two representative separation results here.

Theorem 13.2.1 (A case where XORing is cheaper than using heavy-duty expanders)**.**

1. *For every $\lambda \in (0,1)$, every PRG in $\mathrm{INW}^{\oplus 2}(\lambda)$ fools quadratic polynomials over $\mathbb{F}_2$ with error $O(\sqrt{\lambda})$.*
2. *Let $\vec{\lambda} \in [0,1]^{\log n}$ where n is a power of two. If every PRG in $\mathrm{INW}(\vec{\lambda})$ fools quadratic polynomials over $\mathbb{F}_2$ with error 0.49, then $\lambda_j \leq 2^{-\Omega(2^j)}$ for every $j \geq 4$, and moreover every PRG in $\mathrm{INW}(\vec{\lambda})$ has seed length $\Omega(n)$.*

Theorem 13.2.2 (A case where using heavy-duty expanders is cheaper than XORing)**.** *Let n be a power of two and let $\lambda \in (\frac{1000}{n}, \frac{1}{2})$. There exist $\lambda' = \Omega(\lambda^2)$ and $f\colon \{0,1\}^n \to \{0,1\}$ such that the following hold.*

1. *Every PRG in $\mathrm{INW}(\lambda')$ fools f with error 0.01.*
2. *Let $t \in \mathbb{N}$. If every PRG in $\mathrm{INW}^{\oplus t}(\lambda)$ fools f with error 0.99, then $t \geq \Omega(\lambda \cdot n / \log n)$, and moreover every PRG in $\mathrm{INW}^{\oplus t}(\lambda)$ has seed length $\Omega(n \cdot \lambda \cdot \log(1/\lambda))$.*

13.3 Guide to the chapter

The remainder of the chapter contains the technical development from the paper. It first proves the positive ROBP-fooling theorem, then establishes the seed-length lower bound, and finally studies the incomparability phenomena above. The chapter closes with a short discussion of future directions.

13.4 Proof that sums of INW generators fool constant-width ROBPs

In this section, we present the proof of Theorem 13.1.3, which says that $\mathrm{INW}^{\oplus t}(\Lambda)$ fools width-w programs with error $n^{\log(w+1)} \cdot \prod_{i=1}^{t} \sum_{j=1}^{\log n} \Lambda_{i,j}$. The proof is based on the notion of a *robust PRG*. Roughly speaking, a robust PRG is a multi-seed PRG that still works even

if some seeds are fixed to arbitrary values, with an error bound that depends on the number of random seeds. The precise definition is as follows.

Definition 13.4.1 (Robust PRG)**.** Let $\mathcal{R}_1, \dots, \mathcal{R}_t$ be finite sets, let $G\colon \mathcal{R}_1 \times \cdots \times \mathcal{R}_t \to \{0,1\}^n$, let $W \geq 0$, let $\vec{\varepsilon} = (\varepsilon_1, \dots, \varepsilon_t) \in (0,1)^t$, and let $f\colon \{0,1\}^n \to \mathbb{R}$. We say that G *robustly* $(W, \vec{\varepsilon})$*-fools* f if the following holds. For every $A \subseteq [t]$ and every $x_{[t]\setminus A} \in \prod_{i \in [t]\setminus A} \mathcal{R}_i$, if we sample $x_A \in \prod_{i \in A} \mathcal{R}_i$ uniformly at random, then

$$\left| \mathop{\mathbb{E}}_{x_A} [f(G(x_1, \dots, x_t))] - \mathbb{E}[f] \right| \leq W \cdot \prod_{i \in A} \varepsilon_i.$$

For example, if $\mathcal{R}_1 = \cdots = \mathcal{R}_t = \{0,1\}$ and $G(x_1, \dots, x_t) = x_1 \oplus \cdots \oplus x_t$, then G robustly $(1, \vec{0})$-fools every function $f\colon \{0,1\} \to [0,1]$. This example will serve as the base case of our analysis of $\mathrm{INW}^{\oplus t}$. The inductive step will be based on the following lemma, which shows how to double the output length of a robust PRG fooling ROEPs.

Lemma 13.4.2 (Inductive step in the analysis of $\mathrm{INW}^{\oplus t}$)**.** *Let $n, w \in \mathbb{N}$ where n is even. Let $\mathcal{R}_1, \dots, \mathcal{R}_t$ be finite sets. Let $G\colon \mathcal{R}_1 \times \cdots \times \mathcal{R}_t \to \{0,1\}^{n/2}$. Let $W \geq 0$ and $\vec{\varepsilon} \in (0,1)^t$, and assume that G robustly $(W, \vec{\varepsilon})$-fools all width-w length-$(n/2)$ standard-order ROEPs. For each $i \in [t]$, let $H^{(i)}$ be a D_i-regular multigraph on the vertex set $\mathcal{R}_i$. Let $\vec{\lambda} = (\lambda(H^{(1)}), \dots, \lambda(H^{(t)}))$. Define $G'\colon (\mathcal{R}_1 \times [D_1]) \times \cdots \times (\mathcal{R}_t \times [D_t]) \to \{0,1\}^n$ by the formula*

$$G'((x_1, y_1), \dots, (x_t, y_t)) = (G(x_1, \dots, x_t), G(H^{(1)}[x_1, y_1], \dots, H^{(t)}[x_t, y_t])).$$

Then G' robustly $(W \cdot (w+1), \vec{\varepsilon} + \vec{\lambda})$-fools all width-$w$ length-n standard-order ROEPs.

Proof. Let $f\colon \{0,1\}^n \to [0,1]$ be a width-w length-n standard-order ROEP. Fix any set $A \subseteq [t]$. For every $i \in [t] \setminus A$, let $x_i, x_i' \in \mathcal{R}_i$ be arbitrary fixed values. We also write $X_i = x_i$ and $X_i' = x_i'$ for every $i \in [t] \setminus A$. Meanwhile, for every $i \in A$, sample $X_i \in \mathcal{R}_i$

and $Y_i \in [D_i]$ independently and uniformly at random, and let $X'_i = H^{(i)}[X_i, Y_i]$. Let $X = (X_1, \ldots, X_t)$ and $X' = (X'_1, \ldots, X'_t)$. Our goal is to show that $(G(X), G(X'))$ fools f with error $W \cdot (w+1) \cdot \prod_{i \in A}(\varepsilon_i + \lambda_i)$.

For each vertex u in the middle layer of f, define $f_{\to u} \colon \{0,1\}^{n/2} \to \{0,1\}$ by letting $f_{\to u}(z)$ indicate whether f reaches u when it reads z. Furthermore, define $f_{u\to} \colon \{0,1\}^{n/2} \to [0,1]$ by letting $f_{u\to}(z)$ be the label of the vertex reached in the final layer if we start at u and read z. Let $\mu_{u\to} = \mathbb{E}[f_{u\to}]$ and $\overline{f}_{u\to} = f_{u\to} - \mu_{u\to}$. Then for any $z, z' \in \{0,1\}^{n/2}$, we have

$$f(z,z') = \sum_{u \in [w]} f_{\to u}(z) \cdot f_{u\to}(z') = \left(\sum_{u \in [w]} f_{\to u}(z) \cdot \overline{f}_{u\to}(z') \right) + \left(\sum_{u \in [w]} f_{\to u}(z) \cdot \mu_{u\to} \right).$$

The second sum above is computable by a width-w standard-order ROEP $f_{\text{left}} \colon \{0,1\}^{n/2} \to [0,1]$ that ignores the second half of its input. By assumption, $(G(X), G(X'))$ fools f_{left} with error $W \cdot \prod_{i \in A} \varepsilon_i$.

Now consider a single term in the first sum, $f_{\to u}(z) \cdot \overline{f}_{u\to}(z')$. Under the uniform distribution, we have

$$\mathbb{E}[f_{\to u} \cdot \overline{f}_{u\to}] = \mathbb{E}[f_{\to u}] \cdot \mathbb{E}[\overline{f}_{u\to}] = \mathbb{E}[f_{\to u}] \cdot (\mu_{u\to} - \mu_{u\to}) = 0.$$

Now we analyze the expectation under the pseudorandom distribution $(G(X), G(X'))$. We begin by writing the expectation as a sum. For convenience, for any set $S \subseteq [t]$, let us use the notation $\mathcal{R}_S$ to denote the Cartesian product $\prod_{i \in S} \mathcal{R}_i$. Furthermore, let us identify the graph $H^{(i)}$ with its transition probability matrix. Then we have

$$\begin{aligned}
\mathbb{E}[f_{\to u}(G(X)) \cdot \overline{f}_{u\to}(G(X'))] &= \sum_{x_A, x'_A \in \mathcal{R}_A} \Pr[X = x \text{ and } X' = x'] \cdot f_{\to u}(G(x)) \cdot \overline{f}_{u\to}(G(x')) \\
&= \sum_{x_A, x'_A \in \mathcal{R}_A} \left(\prod_{i \in A} \frac{H_{x_i, x'_i}}{|\mathcal{R}_i|} \right) \cdot f_{\to u}(G(x)) \cdot \overline{f}_{u\to}(G(x')),
\end{aligned}$$

where the notation x denotes the vector $x = (x_1, \ldots, x_t)$, and similarly $x' = (x'_1, \ldots, x'_t)$. Next, we use the decomposition $H^{(i)} = J_{|\mathcal{R}_i|} + E^{(i)}$, where $J_{|\mathcal{R}_i|}$ has $1/|\mathcal{R}_i|$ in every entry and $E^{(i)}$ is some matrix with operator norm λ_i. Applying this decomposition entrywise, we get

$$\begin{aligned}\mathbb{E}[f_{\to u}(G(X)) \cdot \overline{f}_{u\to}(G(X'))] &= \sum_{x_A, x'_A \in \mathcal{R}_A} \left(\prod_{i \in A} \left(\frac{1}{|\mathcal{R}_i|^2} + \frac{E^{(i)}_{x_i, x'_i}}{|\mathcal{R}_i|} \right) \right) \\ &\quad \cdot f_{\to u}(G(x)) \cdot \overline{f}_{u\to}(G(x')) \\ &= \sum_{A = S \sqcup T} \sum_{x_T, x'_T \in \mathcal{R}_T} \left(\prod_{i \in T} \frac{E^{(i)}_{x_i, x'_i}}{|\mathcal{R}_i|} \right) \\ &\quad \cdot \mathop{\mathbb{E}}_{x_S, x'_S \in \mathcal{R}_S} [f_{\to u}(G(x)) \cdot \overline{f}_{u\to}(G(x'))],\end{aligned}$$

where the outer sum is over all partitions of A into two disjoint sets, S and T. The product $\prod_{i \in T} E^{(i)}_{x_i, x'_i}$ is exactly the (x_T, x'_T) entry in the tensor product matrix $\bigotimes_{i \in T} E^{(i)}$. Meanwhile, the expectation

$$\mathop{\mathbb{E}}_{x_S, x'_S \in \mathcal{R}_S} [f_{\to u}(G(x)) \cdot \overline{f}_{u\to}(G(x'))]$$

splits as a product of expectations. Thus, we get

$$\begin{aligned}\mathbb{E}[f_{\to u}(G(X)) \cdot \overline{f}_{u\to}(G(X'))] &= \sum_{A = S \sqcup T} \frac{1}{|\mathcal{R}_T|} \cdot \sum_{x_T, x'_T \in \mathcal{R}_T} \left(\bigotimes_{i \in T} E^{(i)} \right)_{x_T, x'_T} \\ &\quad \cdot \mathop{\mathbb{E}}_{x_S \in \mathcal{R}_S} [f_{\to u}(G(x))] \\ &\quad \cdot \mathop{\mathbb{E}}_{x'_S \in \mathcal{R}_S} [\overline{f}_{u\to}(G(x'))].\end{aligned}$$

We can think of the first expectation, $\mathbb{E}_{x_S \in \mathcal{R}_S}[f_{\to u}(G(x))]$, as a single entry a_{x_T} in a long vector $a \in \mathbb{R}^{\mathcal{R}_T}$. Similarly, we think of the second expectation, $\mathbb{E}_{x_S \in \mathcal{R}_S}[\overline{f}_{u\to}(G(x'))]$, as a single entry $b_{x'_T}$ in a long vector $b \in \mathbb{R}^{\mathcal{R}_T}$. In this way, the inner sum above becomes a

vector-matrix-vector product:

$$\begin{aligned}
|\,\mathbb{E}[f_{\to u}(G(X)) \cdot \overline{f}_{u\to}(G(X'))]| &= \left| \sum_{A=S\sqcup T} \frac{1}{|\mathcal{R}_T|} \cdot a^{\mathsf{T}} \cdot E^{\otimes T} \cdot b \right| \\
&\le \sum_{A=S\sqcup T} \frac{1}{|\mathcal{R}_T|} \cdot \|a\|_2 \cdot \left\| \bigotimes_{i\in T} E^{(i)} \right\|_{\mathrm{op}} \cdot \|b\|_2 \\
&\le \sum_{A=S\sqcup T} \|a\|_\infty \cdot \|b\|_\infty \cdot \prod_{i\in T} \|E^{(i)}\|_{\mathrm{op}} \\
&= \sum_{A=S\sqcup T} \|a\|_\infty \cdot \|b\|_\infty \cdot \prod_{i\in T} \lambda_i.
\end{aligned}$$

The function $f_{\to u}$ is $\{0,1\}$-valued, so $\|a\|_\infty \le 1$. Meanwhile, for any fixing of $x'_T \in \mathcal{R}_T$, the entry $b_{x'_T}$ is precisely the error when we sample $x'_S \in \mathcal{R}_S$ uniformly at random and try to use $G(x')$ to fool $f_{u\to}$, a width-w length-$(n/2)$ standard-order ROEP. By assumption, $\|b\|_\infty \le W \cdot \prod_{i\in S} \varepsilon_i$. Therefore,

$$|\,\mathbb{E}[f_{\to u}(G(X)) \cdot \overline{f}_{u\to}(G(X'))]| \le W \cdot \sum_{A=S\sqcup T} \left(\prod_{i\in S} \varepsilon_i\right) \cdot \left(\prod_{i\in T} \lambda_i\right) = W \cdot \prod_{i\in A} (\varepsilon_i + \lambda_i).$$

Consequently, summing up all the errors, we get

$$|\,\mathbb{E}[f(G(X), G(X'))] - \mathbb{E}[f]| \le \left(\sum_{u\in[w]} W \cdot \prod_{i\in A} (\varepsilon_i + \lambda_i)\right) + W \cdot \prod_{i\in A} \varepsilon_i \le W \cdot (w+1) \cdot \prod_{i\in A} (\varepsilon_i + \lambda_i).$$

□

Proof of Theorem 13.1.3. Let $G_{\log n}$ be any PRG in the family $\mathrm{INW}^{\oplus t}(\Lambda)$. We will prove by induction on n that $G_{\log n}$ robustly $(W, \vec{\varepsilon})$-fools width-w length-n standard-order ROEPs, where $W = (w+1)^{\log n}$ and $\varepsilon_i = \sum_{j=1}^{\log n} \Lambda_{i,j}$. For the base case, if $n = 1$, then there is exactly one PRG in the family $\mathrm{INW}^{\oplus t}(\Lambda)$, namely $G_0 \colon \{0,1\}^t \to \{0,1\}$ is given by $G_0(x) = x_1 \oplus \cdots \oplus x_t$. This PRG indeed robustly $(1, \vec{0})$-fools all functions $f \colon \{0,1\} \to [0,1]$. Now, for the inductive step, suppose $n > 1$. By definition of $\mathrm{INW}^{\oplus t}(\Lambda)$, the PRG $G_{\log n}$ has

the form $G_{\log n}(a_1, \ldots, a_t) = G^{(1)}_{\log n}(a_1) \oplus \cdots \oplus G^{(t)}_{\log n}(a_t)$, where $G^{(i)}_{\log n} \in \mathrm{INW}(\Lambda_i)$ for every i. By definition of $\mathrm{INW}(\Lambda_i)$, the seed a_i is a pair (x_i, y_i), and the generator $G^{(i)}_{\log n}$ has the form

$$G^{(i)}_{\log n}(x_i, y_i) = (G^{(i)}_{\log n-1}(x_i), G^{(i)}_{\log n-1}(H^{(i)}[x_i, y_i]))$$

for some $G^{(i)}_{\log n-1} \in \mathrm{INW}((\Lambda_{i,1}, \ldots, \Lambda_{i,\log n-1}))$ and some $\Lambda_{i,\log n}$-spectral expander $H^{(i)}$. Define a PRG $G_{\log n-1}$ by the rule

$$G_{\log n-1}(x_1, \ldots, x_t) = G^{(1)}_{\log n-1}(x_1) \oplus \cdots \oplus G^{(t)}_{\log n-1}(x_t).$$

Then $G_{\log n-1} \in \mathrm{INW}^{\oplus t}(\Lambda')$, where Λ' consists of all but the last column of Λ. By induction, $G_{\log n-1}$ robustly $((w+1)^{\log n-1}, \vec{\alpha})$-fools width-$w$ length-n standard-order ROEPs, where $\alpha_i = \sum_{j=1}^{\log n-1} \Lambda_{i,j}$. Working through the definitions, we see that the PRG $G_{\log n}$ can be written as

$$G_{\log n}((x_1, y_1), \ldots, (x_t, y_t)) = (G_{\log n-1}(x_1, \ldots, x_t), G_{\log n-1}(H^{(1)}[x_1, y_1], \ldots, H^{(t)}[x_t, y_t])).$$

Applying Theorem 13.4.2 completes the inductive step.

Finally, since $G_{\log n}$ robustly $(W, \vec{\varepsilon})$-fools width-w length-n standard-order ROEPs, it follows that $G_{\log n}$ fools width-w length-n standard-order ROBPs with error ε, where

$$\varepsilon = W \cdot \prod_{i=1}^{t} \varepsilon_i = n^{\log(w+1)} \cdot \prod_{i=1}^{t} \sum_{j=1}^{\log n} \Lambda_{i,j}. \qquad \square$$

13.5 Seed length lower bound for fooling ROBPs

In this section, we prove Theorem 13.1.4, which states that no matter how many (or how few) copies of the INW generator we XOR together, and regardless of how we set the expansion parameters, if the resulting PRG fools width-3 programs, then the seed length is inevitably

$\Omega(\log^2 n)$.

13.5.1 Sums of small-bias distributions

We begin by analyzing a family of a small-bias distributions introduced by Lee and Viola [LV17]. This construction guarantees that summing independent samples from these distributions does not substantially increase the overall number of distinct strings.

Definition 13.5.1 (Sum of sets)**.** For $S, T \subseteq \{0,1\}^m$, $S + T = \{s \oplus t \mid s \in S, t \in T\}$.

Lemma 13.5.2 (Small-bias distributions with a small sum set)**.** *For every $m, t \in \mathbb{N}$ and every $\varepsilon_1, \ldots, \varepsilon_t \in (0,1]$ such that $\lceil \ln(1/\varepsilon_1) \rceil + \cdots + \lceil \ln(1/\varepsilon_t) \rceil < \frac{1}{625} \cdot m$, there exist distributions $D_1, \ldots, D_t$ over $\{0,1\}^m$ such that D_i is ε_i-biased for every i, and*

$$|\operatorname{Supp}(D_1) + \cdots + \operatorname{Supp}(D_t)| < 2 \cdot 2^{m/2}.$$

Furthermore, the probability mass function of each D_i only takes on rational values.

Proof. For each $i \in [t]$, we construct D_i by taking the bitwise XOR $X \oplus Y_i$, where X is distributed uniformly over $\mathcal{C}^\perp$ which is $(\frac{m}{25})$-wise uniform constructed by Theorem 12.2.17, and Y_i is an independent "noise vector" constructed as follows. Repeat the following process r_i times independently, where $r_i = \lceil 25 \ln(1/\varepsilon_i) \rceil$: choose a position uniformly at random from $[m]$, and set it to a uniform bit. The remaining bits of Y_i are zero. Note that X is uniform over a subspace of $\mathbb{F}_2^m$ and the probability of getting a particular noise vector y is a multiple of $(1/m)^{r_i}$, which is a rational number.

First we prove that each D_i is ε_i-biased. For any character function χ_S with $|S| < \frac{m}{25}$, since X and Y_i are independent and X is $(\frac{m}{25})$-wise uniform, χ_S is perfectly fooled by D_i. Next, consider any character function χ_S, where $|S| \geq \frac{m}{25}$. In this case, the bias is nonzero

only if none of the elements in S are selected by the random noise Y_i. So the bias is at most

$$(1 - |S|/m)^{r_i} \leq \exp(-r_i \cdot |S|/m) \leq \varepsilon_i.$$

Thus, we have shown that each D_i is ε_i-biased.

By the closure property of linear subspaces,

$$\begin{aligned} |\operatorname{Supp}(D_1) + \cdots + \operatorname{Supp}(D_t)| &= |\operatorname{Supp}(X) + \operatorname{Supp}(Y_1) + \cdots + \operatorname{Supp}(X) + \operatorname{Supp}(Y_t)| \\ &= |\mathcal{C}^\perp + \cdots + \mathcal{C}^\perp + \operatorname{Supp}(Y_1) + \cdots + \operatorname{Supp}(Y_t)| \\ &= |\mathcal{C}^\perp + \operatorname{Supp}(Y_1) + \cdots + \operatorname{Supp}(Y_t)|. \end{aligned}$$

The set $\mathcal{C}^\perp + \operatorname{Supp}(Y_1) + \cdots + \operatorname{Supp}(Y_t)$ is precisely the set of all binary strings within Hamming distance at most $r_1 + \cdots + r_t \leq \lceil 25\ln(1/\varepsilon_1)\rceil + \ldots + \lceil 25\ln(1/\varepsilon_t)\rceil < \frac{m}{25}$ from $\mathcal{C}^\perp$. Therefore,

$$\begin{aligned} |\mathcal{X} + \operatorname{Supp}(Y_1) + \cdots + \operatorname{Supp}(Y_t)| &\leq |\mathcal{C}^\perp| \cdot \binom{m}{\leq m/25} \leq 2^{\lceil H(\frac{1}{25})\cdot m\rceil} \cdot 2^{H(\frac{1}{25})\cdot m} \\ &< 2 \cdot 2^{m/2}. \end{aligned}$$

□

Remark 13.5.3 (The benefit of noisy codewords). In the proof of Theorem 13.5.2, we use small-bias distributions based on noisy codewords [LV17]. A more straightforward approach for minimizing $|\operatorname{Supp}(D_1) + \cdots + \operatorname{Supp}(D_t)|$ would be to simply make $|\operatorname{Supp}(D_i)|$ as small as possible and then use the trivial bound

$$|\operatorname{Supp}(D_1) + \cdots + \operatorname{Supp}(D_t)| \leq \prod_{i=1}^{t} |\operatorname{Supp}(D_i)|.$$

This approach is too weak to prove Theorem 13.5.2. Indeed, every small-bias distribution D_i satisfies $|\operatorname{Supp}(D_i)| \geq \Omega(m)$ [AGHP92], so we inevitably have $\prod_{i=1}^{t} |\operatorname{Supp}(D_i)| \geq \Omega(m)^t$, so

the bound is trivial when $t \geq 1.01 \cdot m/\log m$. In contrast, Theorem 13.5.2 is meaningful even when $t = \Theta(m)$.

13.5.2 Constructing an $\mathrm{INW}^{\oplus t}$ *generator that does not fool ROBPs*

Recall that the seed length of $\mathrm{INW}^{\oplus t}_{\mathcal{H}}$ is $\sum_{i\in[t],j\in[\log n]} \log(\deg(H_{i,j}))$. Our goal is to show that if every PRG in $\mathrm{INW}^{\oplus t}(\Lambda)$ fools width-3 programs, then every PRG in $\mathrm{INW}^{\oplus t}(\Lambda)$ has seed length $\Omega(\log^2 n)$. Indeed, we will show that for each PRG in $\mathrm{INW}^{\oplus t}(\Lambda)$, the seed length grows by $\Omega(\log n)$ in each of $\Omega(\log n)$ of the rounds of the INW recursion. The lemma below makes this precise, formulated in the contrapositive.

Lemma 13.5.4 (One-round seed length lower bound for fooling width-3 programs)**.** *Let n be a sufficiently large power of two, let $t \in \mathbb{N}$, let $\Lambda \in [0,1]^{t\times \log n}$, and let $j_* \in [\log\log n, \frac{1}{4}\log n]$. Assume that there exists a $t \times \log n$ permissible family of labeled regular undirected multigraphs $\mathcal{H} = \{H_{i,j}\}$ such that $\lambda(H_{i,j}) \leq \Lambda_{i,j}$ for every i, j and $\log(\deg(H_{1,j_*}))+\cdots+\log(\deg(H_{t,j_*})) < \frac{1}{20000}\cdot\log n$. Then there exists another $t \times \log n$ permissible family of labeled regular undirected multigraphs $\mathcal{H}' = \{H'_{i,j}\}$ such that $\lambda(H'_{i,j}) \leq \Lambda_{i,j}$ for every i, j, along with a length-n, width-3, standard-order ROBP B such that $\Pr\big[B(U_{\{0,1\}^n}) = 1\big] \geq 1 - \exp(-n^{1/4})$, but $B(\mathrm{INW}^{\oplus t}_{\mathcal{H}'}(x)) = 0$ for every seed x.*

Proof. We first partition $[t]$ based on whether $\deg(H_{i,j_*}) \geq 1/(2\Lambda^2_{i,j_*})$. Let $T_1 \subseteq [t]$ be the indices such that $\deg(H_{i,j_*}) \geq 1/(2\Lambda^2_{i,j_*})$ and let $T_2 = [t] \setminus T_1$, which means for all $i \in T_2$, $\deg(H_{i,j_*}) \geq (|H_{i,j_*}| + 1)/2$ by Theorem 12.2.7. Without loss of generality, we assume that $T_1 = [t_1]$ and $T_2 = [t] \setminus [t_1]$. Note that if $\Lambda_{i,j_*} = 0$ for some $i \in [t]$, then $i \in T_2$.

Let M be a power of two satisfying $n^{1/8} \leq M < n^{1/4}$, and let $m = \log M$. Now we can construct a new family of graphs $\mathcal{H}'$. For indices $i \in T_1$, we use Theorem 13.5.2 to construct a family of distributions $D_i, i \in T_1$ over $\{0,1\}^m$ such that each D_i is Λ_{i,j_*}-biased. By Theorem 12.2.7, we know that for any graph H_{i,j_*} such that $\deg(H_{i,j_*}) = 1$, we have

$\lambda(H_{i,j_*}) = \Lambda_{i,j_*} = 1$. Thus,

$$\begin{aligned}\sum_{i\in T_1} \lceil \log(1/\Lambda_{i,j_*})\rceil &= \sum_{i\in T_1, \Lambda_{i,j_*}\neq 1} \lceil \log(1/\Lambda_{i,j_*})\rceil + \sum_{i\in T_1, \Lambda_{i,j_*}=1} \lceil \log(1/\Lambda_{i,j_*})\rceil \\ &\leq \sum_{i\in T_1, \Lambda_{i,j_*}\neq 1} \lceil \log(2\cdot \deg(H_{i,j_*}))\rceil + 0 \\ &\leq 2\cdot \sum_{i\in[t]} \log(\deg(H_{i,j_*})) \leq \frac{1}{10000}\cdot \log n < \frac{1}{1000}\cdot m,\end{aligned}$$

where we used the fact that if $\Lambda_{i,j_*} \neq 1$, then $\deg(H_{i,j_*}) \geq 2$. Let $Z_i = \mathrm{Cay}(\mathbb{F}_2^m, \mathrm{SB}_i)$, where $\mathrm{SB}_i \colon K_i \to \{0,1\}^m$ is some generator such that $\mathrm{SB}_i(U_{K_i}) = D_i$. (Such a generator exists, because all probabilities under D_i are rational numbers.) This graph satisfies $\lambda(Z_i) \leq \Lambda_{i,j_*}$ by Theorem 12.2.6, and each vertex in Z_i has $|\operatorname{Supp}(D_i)|$ distinct neighbors. Let $q = 2^{j_*-1}$ and $Q = 2^q$, and we define

$$W_i = Z_i \otimes J_{Q/M}.$$

Recall that $\mathcal{H}'_1, \ldots, \mathcal{H}'_t$ denote the rows of $\mathcal{H}'$. For each $i \in T_1$, let

$$\mathcal{H}'_i = [J_2, \ldots, J_{2^{2^{j_*-2}}}, W_i, J_*, \ldots, J_*],$$

that is, for all indices $j' \neq j_*$, $H'_{i,j'}$ is the complete graph of appropriate size, and in j_*-th index, we use our W_i constructed before.

For each $i' \in T_2$, let

$$\mathcal{H}'_{i'} = [H_{i',1}, H_{i',2}, \ldots, H_{i',j_*-1}, H_{i',j_*}, J_*, \ldots J_*],$$

i.e. the graphs up to j_* are exactly same as the corresponding graphs in $\mathcal{H}$, and after j_*, we use the complete graphs of appropriate size. By combining $\mathcal{H}'_i, i \in T_1$ and $\mathcal{H}'_{i'}, i' \in T_2$ accordingly, we get a new family graphs $\mathcal{H}'$. Since $\mathcal{H}$ is a family of graphs that satisfy the

constraint Λ, so is $\mathcal{H}'$.

By the definition of $\mathrm{INW}^{\oplus t}_{\mathcal{H}'}$,

$$\begin{aligned}
\mathrm{INW}^{\oplus t}_{\mathcal{H}'}(x_1^0,\dots,x_t^0) &= \mathrm{INW}_{\mathcal{H}'_1}(x_1^0)\oplus\cdots\oplus\mathrm{INW}_{\mathcal{H}'_t}(x_t^0)\\
&= G^{(1)}_{\log n}(x_1^0)\oplus\cdots\oplus G^{(t)}_{\log n}(x_t^0)\\
&= \big(G^{(1)}_{\log n-1}(x'_1), G^{(1)}_{\log n-1}(H'_{1,\log n}[x'_1,y'_1])\big)\\
&\qquad\oplus\cdots\oplus\big(G^{(t)}_{\log n-1}(x'_t), G^{(t)}_{\log n-1}(H'_{t,\log n}[x'_t,y'_t])\big),
\end{aligned}$$

where x'_i and y'_i are the substrings of x_i of appropriate length, and each $G^{(i)}_{\log n}$ as a INW generator in $\mathrm{INW}(\Lambda_i)$, so that $G^{(i)}_j$, $j\in[\log n]$, is defined recursively. By recursively expanding these generators until the j_*-th round, where we can express the output of $\mathrm{INW}^{\oplus t}_{\mathcal{H}'}$ as $n/2^{j_*}$ independent copies, as $\mathcal{H}'_{i,j'}$ are complete graphs for $i\in[t]$ and $j_*<j'\le\log n$, of the following form

$$\begin{aligned}
&G^{(1)}_{j_*}(x_1,y_1)\oplus\cdots\oplus G^{(t)}_{j_*}(x_t,y_t)\\
&=\big(G^{(1)}_{j_*-1}(x_1),\, G^{(1)}_{j_*-1}(H'_{1,j_*}[x_1,y_1])\big)\oplus\cdots\oplus\big(G^{(t)}_{j_*-1}(x_t),\, G^{(t)}_{j_*-1}(H'_{1,j_*}[x_t,y_t])\big)\\
&=\Big(x_1\oplus\cdots\oplus x_{t_1}\oplus G^{(t_1+1)}_{j_*-1}(x_{t_1+1})\oplus\cdots\oplus G^{(t)}_{j_*-1}(x_t),\\
&\quad H'_{1,j_*}[x_1,y_1]\oplus\cdots\oplus H'_{t_1,j_*}[x_{t_1},y_{t_1}]\\
&\quad\oplus G^{(t_1+1)}_{j_*-1}(H'_{t_1+1,j_*}[x_{t_1+1},y_{t_1+1}])\oplus\cdots\oplus G^{(t)}_{j_*-1}(H'_{t,j_*}[x_t,y_t])\Big),
\end{aligned}$$

as $G^{(i)}_{j_*-1}$ is the identity function for $i\in T_1$, where x_i,y_i are the corresponding input strings.

Let R'_i be the set of all possible $G^{(i)}_{j_*-1}(x_i)\oplus G^{(i)}_{j_*-1}(H'_{i,j_*}[x_i,y_i])$ where we allow x_i to range over all vertices *and* we allow y_i to range over all edge labels, and let $R_i=\{x_{1,\dots,m}|x\in R'_i\}$, which is the set of first m bits of strings in R'_i. In the following, we are going to show that $|\sum_{i=1}^t R_i|<2^m$, hence there is at least one $v_*\in\{0,1\}^m$ such that $v_*\notin\sum_{i=1}^t R_i$. Our

approach is to bound

$$\left|\sum_{i=1}^{t} R_i\right| \leq \left|\sum_{i\in T_1} R_i\right| \cdot \prod_{i\in T_2} |R_i|.$$

We will bound $|\sum_{i\in T_1} R_i|$ and $\prod_{i\in T_2} |R_i|$ separately.

For each $i \in T_1$, let $x_i^{(1)} \in \{0,1\}^m$ be the first m bits of x_i, and $x_i^{(2)} \in \{0,1\}^{q-m}$ to be the substring of x_i after $x_i^{(1)}$, so that $x_i = (x_i^{(1)}, x_i^{(2)})$. For y_i, since $W_i = Z_i \otimes J$, we have $y_i = (y_i^{(1)}, y_i^{(2)})$ where $y_i^{(1)} \in K_i$ and $y_i^{(2)} \in \{0,1\}^{q-m}$. Thus, for $i \in T_1$, we have

$$\begin{aligned}
x_i \oplus H'_{i,j_*}[x_i, y_i] &= (x_i^{(1)}, x_i^{(2)}) \oplus H'_{i,j_*}[(x_i^{(1)}, x_i^{(2)}), (y_i^{(1)}, y_i^{(2)})] \\
&= (x_i^{(1)}, x_i^{(2)}) \oplus W_i[(x_i^{(1)}, x_i^{(2)}), (y_i^{(1)}, y_i^{(2)})] \\
&= (x_i^{(1)}, x_i^{(2)}) \oplus (Z_i[x_i^{(1)}, y_i^{(1)}], J_{Q/M}[x_i^{(2)}, y_i^{(2)}]) \\
&= (x_i^{(1)}, x_i^{(2)}) \oplus (Z_i[x_i^{(1)}, y_i^{(1)}], y_i^{(2)}).
\end{aligned}$$

By our construction of Z_i, for any choice of $x_i^{(1)}, y_i^{(1)}$, we have $x_i^{(1)} \oplus Z_i[x_i^{(1)}, y_i^{(1)}] \in \mathrm{Supp}(D_i)$, and

$$\bigoplus_{i\in T_1} x_i^{(1)} \oplus Z_i[x_i^{(1)}, y_i^{(1)}] \in \sum_{i\in T_1} \mathrm{Supp}(D_i).$$

Therefore, by Theorem 13.5.2, we have

$$\left|\sum_{i\in T_1} R_i\right| \leq \left|\sum_{i\in T_1} \mathrm{Supp}(D_i)\right| \leq 2 \cdot 2^{m/2}.$$

For $i \in T_2$, we have

$$G_{j_*-1}^{(i)}(x_i) \oplus G_{j_*-1}^{(i)}(H'_{i,j_*}[x_i, y_i]) = G_{j_*-1}^{(i)}(x_i) \oplus G_{j_*-1}^{(i)}(H_{i,j_*}[x_i, y_i]).$$

For $i \in T_2$, $\deg(H_{i,j_*}) \geq \frac{|H_{i,j_*}|+1}{2}$ and x_i is a vertex label in H_{i,j_*},

$$|R_i| \leq |H_{i,j_*}| \cdot \deg(H_{i,j_*}) \leq 2(\deg(H_{i,j_*}) - 1) \cdot \deg(H_{i,j_*}) \leq 2\deg(H_{i,j_*})^2.$$

We know that $\log(\deg(H_{1,j_*})) + \ldots + \log(\deg(H_{t,j_*})) < \frac{1}{20000} \cdot \log n < 0.001 \cdot \log n$. By Theorem 12.2.7, for any undirected graph h with degree 1, we know $\lambda(h) = 1$, so that for any $i \in T_2$, $\deg(H_{i,j_*}) \geq 2$, which means $|T_2| < 0.001 \cdot \log n$. Therefore,

$$\prod_{i \in T_2} |R_i| \leq 2^{|T_2|} \cdot \prod_{i \in T_2} \deg(H_{i,j_*})^2 \leq 2^{0.001 \cdot \log n} \cdot n^{0.01} \leq n^{0.02} \leq 2^{0.2m}.$$

By the bounds above, the number of possible choices for v is at most

$$\left| \sum_{i \in T_1} R_i \right| \cdot \prod_{i \in T_2} |R_i| \leq 2 \cdot 2^{m/2 + 0.2m} < 2^m.$$

Therefore, there is at least one $v_* \in \{0,1\}^m$ such that $v_* \notin \sum_{i=1}^{t} R_i$.

Encoding this missing element to a width-3 branching program B computing the function $f : \{0,1\}^n \to \{0,1\}$ as

$$f(x) = \bigvee_{i=0}^{n/2^{j_*}-1} (x_{2^{j_*} \cdot i+1 \ldots 2^{j_*} \cdot i+m} = 0^m \wedge x_{2^{j_*} \cdot i+q+1 \ldots 2^{j_*} \cdot i+q+m} = v_*),$$

we know that $B(\mathrm{INW}_{\mathcal{H}'}^{\oplus t}(x)) = 0$ for any input seed x. However, for a truly random input satisfies each clause with probability $2^{-2m} \geq 1/\sqrt{n}$, and since there are $n/2^{j_*} \geq n^{3/4}$ such clauses, the acceptance probability is

$$\Pr[B(U_{\{0,1\}^n}) = 1] = 1 - \Pr[B(U_{\{0,1\}^n}) = 0] \geq 1 - (1 - 1/\sqrt{n})^{n^{3/4}} \geq 1 - \exp(-n^{1/4}). \quad \square$$

We can then prove Theorem 13.1.4.

Theorem 13.5.5 (Restatement of Theorem 13.1.4)**.** *Let n be a power of two, let $t \in \mathbb{N}$, and let $\Lambda \in [0,1]^{t \times \log n}$. If every PRG in the family $\mathrm{INW}^{\oplus t}(\Lambda)$ fools width-3 standard-order ROBPs with error 0.99, then every PRG in the family $\mathrm{INW}^{\oplus t}(\Lambda)$ has seed length $\Omega(\log^2 n)$.*

Proof. For every $t \in \mathbb{N}$, Theorem 13.5.4 implies that for every PRG in the family $\mathrm{INW}^{\oplus t}(\Lambda)$ and for each $j \in [\log\log n, \log n]$,

$$\log(\deg(H_{1,j})) + \cdots + \log(\deg(H_{t,j})) \geq \frac{1}{20000} \cdot \log n.$$

Consequently, the overall seed length is at least

$$\begin{aligned}\sum_{j \in [\log\log n, \log n]} \big(\log(\deg(H_{1,j})) + \cdots + \log(\deg(H_{t,j}))\big) &\geq \sum_{j \in [\log\log n, \log n]} \frac{1}{20000} \cdot \log n \\ &= \Omega(\log^2 n).\end{aligned}$$

□

13.6 A case where XORing is cheaper than using heavy-duty expanders

In this section, we give the proof of the Theorem 13.2.1, which shows that $\mathrm{INW}^{\oplus 2}$ is indeed strictly better than INW at fooling certain tests, namely quadratic polynomials over $\mathbb{F}_2$. As discussed in Section 13.2, the proof that $\mathrm{INW}^{\oplus 2}$ does a good job of fooling such polynomials is an immediate consequence of prior work [KNP11, De11, Ste12, BV10a, Lov09, Vio09]. The main content of the theorem is the fact that INW alone does a *bad* job of fooling such polynomials. The proof is based on the *inner product* function $\mathsf{IP}_{2m} \colon \{0,1\}^m \times \{0,1\}^m \to \{0,1\}$, which is defined as follows for each positive integer m:

$$\mathsf{IP}_{2m}(x,y) = \bigoplus_{i=1}^{m} x_i \cdot y_i.$$

We rely on the following expander graph construction.

Proposition 13.6.1 (An expander contained in $\mathrm{IP}^{-1}(0)$)**.** *For every $m \in \mathbb{N}$, there exists a regular undirected multigraph $H^*_{2^m} = (V, E)$ on the vertex set $V = \{0,1\}^m$ such that $\lambda(H^*_{2^m}) \leq 5 \cdot 2^{-m/2}$ and $\mathsf{IP}_{2m}(x, y) = 0$ for every $(x, y) \in E$.*

Proof. Let G be a graph on $\{0,1\}^m$ in which (x, y) is an edge if and only if $\mathrm{IP}(x, y) = 0$. Let A be the adjacency matrix of G. This graph G is not regular, because 0^m is connected to all other vertices. To fix this, let $H^*_{2^m}$ be the graph with adjacency matrix $A + \Delta$, where

$$\Delta_{x,y} = \begin{cases} -1 & \text{if } \mathsf{PARITY}(x) = \mathsf{PARITY}(y) = 0 \text{ and } 0^m \in \{x, y\}, \\ +1 & \text{if } x = y \text{ and } x \neq 0^m \text{ and } \mathsf{PARITY}(x) = 0, \\ 0 & \text{otherwise.} \end{cases}$$

The adjacency matrix $A + \Delta$ is nonnegative, because $0^m \in \{x, y\}$ implies that $\mathrm{IP}(x, y) = 0$. Furthermore, $H^*_{2^m}$ is undirected, because Δ and A are symmetric. Now we show that H is regular of degree $D = 2^{m-1}$. For the vertex $x = 0^m$,

$$\deg(x) = \sum_{y \in \{0,1\}^m} (A_{x,y} + \Delta_{x,y}) = \sum_{y \in \{0,1\}^m} 1 + \sum_{\substack{y \in \{0,1\}^m \\ \mathsf{PARITY}(y)=0}} -1 = 2^{m-1}.$$

For a vertex $x \neq 0$ and $\mathsf{PARITY}(x) = 0$, we have

$$\deg(x) = \sum_{y \in \{0,1\}^m} (A_{x,y} + \Delta_{x,y}) = 2^{m-1} + 1 - 1 = 2^{m-1}.$$

Finally, for a vertex x such that $\mathsf{PARITY}(x) = 1$, we have

$$\deg(x) = \sum_{y \in \{0,1\}^m} (A_{x,y} + \Delta_{x,y}) = 2^{m-1} + 0 = 2^{m-1}.$$

We now move on to bounding $\lambda(H^*_{2^m})$. Let $M = 2^m$, and let $\mathbf{1}$ denote the all-ones column vector of length M. By the definition of expansion parameter and triangle inequality,

$$\lambda(H^*_{2^m}) = \|H^*_{2^m} - J_M\|_{\text{op}} = \left\| \frac{A+\Delta}{D} - \frac{\mathbf{1}\mathbf{1}^T}{M} \right\|_{\text{op}} \leq \left\| \frac{A}{D} - \frac{\mathbf{1}\mathbf{1}^T}{M} \right\|_{\text{op}} + \left\| \frac{\Delta}{D} \right\|_{\text{op}}$$
$$= \frac{1}{M} \|2A - \mathbf{1}\mathbf{1}^T\|_{\text{op}} + \frac{1}{D} \|\Delta\|_{\text{op}}.$$

Consider the first term. We claim that the matrix $2A - \mathbf{1}\mathbf{1}^T$ is a Hadamard matrix. Indeed, each entry of the matrix is given by

$$(2A - \mathbf{1}\mathbf{1}^T)_{x,y} = 1 - 2\,\mathsf{IP}(x,y) = (-1)^{\langle x,y \rangle},$$

so the dot product of two distinct rows $x, x' \in \{0,1\}^m$ is given by

$$\sum_{y \in \{0,1\}^m} (-1)^{\langle x,y \rangle} \cdot (-1)^{\langle x',y \rangle} = \sum_{y \in \{0,1\}^m} (-1)^{\langle x+x',\, y \rangle}.$$

Since $x + x' \neq 0$, the map $y \mapsto \langle x + x',\, y \rangle \bmod 2$ takes on equally many 0's and 1's as y varies; hence the sum of $(-1)^{\langle x+x',\, y \rangle}$ over all y is zero. Thus, $2A - \mathbf{1}\mathbf{1}^T$ is a matrix with ± 1 entries with mutually orthogonal rows, i.e., a Hadamard matrix. It follows that $\|2A - \mathbf{1}\mathbf{1}^T\|_{\text{op}} = \sqrt{M}$, because each row has norm $\sqrt{M}$.

Now consider the second term, $(1/D) \cdot \|\Delta\|_{\text{op}}$. For any unit vector π, the entries of $\pi\Delta$ are given by

$$(\pi\Delta)_y = \sum_x \pi_x \Delta_{x,y} = \begin{cases} -\sum_{x:\, \mathsf{PARITY}(x)=0} \pi_x, & \text{if } y = 0^m, \\ \pi_y - \pi_{0^m}, & \text{if } y \neq 0^m \text{ and } \mathsf{PARITY}(y) = 0, \\ 0, & \text{if } \mathsf{PARITY}(y) = 1. \end{cases}$$

Consequently,

$$\begin{aligned}
\|\pi\Delta\|_2^2 &= \left(\sum_{\substack{x\in\{0,1\}^m \\ \mathsf{PARITY}(x)=0}} \pi_x\right)^2 + \sum_{\substack{y\in\{0,1\}^m\setminus\{0^m\} \\ \mathsf{PARITY}(y)=0}} (\pi_y - \pi_{0^m})^2 \\
&\leq 2^{m-1} \sum_{x\colon \mathsf{PARITY}(x)=0} \pi_x^2 + 2 \sum_{\substack{y\neq 0 \\ \mathsf{PARITY}(y)=0}} \left(\pi_y^2 + \pi_{0^m}^2\right) \\
&\leq 2^{m-1}\cdot 1 + 2\cdot 1 + 2\left(2^{m-1}-1\right)\cdot 1 \\
&< 4M.
\end{aligned}$$

Thus $\|\Delta\|_{\mathrm{op}} \leq 2\sqrt{M}$, and so

$$\lambda(H) \;\leq\; \frac{1}{\sqrt{M}} \;+\; \frac{2\cdot 2\cdot\sqrt{M}}{M} \;=\; 5\cdot M^{-1/2}. \qquad \square$$

We can then prove Theorem 13.2.1:

Proof. To prove Item 1, recall that for any constant w, every generator in $\mathrm{INW}(\lambda)$ fools width-w standard-order "permutation ROBPs" with error $O(\lambda)$ [De11, Ste12, KNP11]. Every parity function can be computed by a width-2 standard-order permutation ROBP, so it follows that every generator in $\mathrm{INW}(\lambda)$ is $O(\lambda)$-biased. Finally, the XOR of any two λ-biased distributions fools quadratic polynomials over $\mathbb{F}_2$ with error $\lambda' \leq O(\sqrt{\lambda})$ [BV10a, Lov09, Vio09].

Now let us prove Item 2. Let $m_j = 2^{j-1}$ for each $j \in [\log n]$. Suppose $\lambda_j \geq 5\cdot 2^{-m_j/2}$ for some j. In this case, we will construct an $\mathrm{INW}(\vec{\lambda})$ generator that does not hit IP_{2m_j} applied to the first $2m_j$ bits of its output. We use the following family of expanders:

$$\mathcal{H} = (J_2, J_4, \ldots, J_{2^{m_{j-1}}}, H^*_{2^{m_j}}, J_*, \ldots, J_*).$$

Here $H^*_{2^{m_j}}$ denotes the graph constructed in Theorem 13.6.1. By Theorem 13.6.1, this family

$\mathcal{H}$ satisfies the constraint $\vec{\lambda}$. Furthermore, for any output $(x, y, z) \in \{0,1\}^{m_j} \times \{0,1\}^{m_j} \times \{0,1\}^{n-2m_j}$ of $\mathrm{INW}_{\mathcal{H}}$, we have $(x, y) \in E$, and hence $\mathrm{IP}_{2m_j}(x, y) = 0$ by Theorem 13.6.1.

The function IP_{2m} is a quadratic polynomial over $\mathbb{F}_2$, and under the uniform distribution, we have $\mathbb{E}[\mathsf{IP}_{2m}] = 1/2 \cdot (1 - 2^{-m})$, which is larger than 0.49 if $m \geq 6$. Therefore, if we assume that $\mathrm{INW}(\vec{\lambda})$ fools all quadratic polynomials with error 0.49, then it follows that $\lambda_j < 5 \cdot 2^{-m_j/2} < 2^{-m_j/8}$ for all $j \geq 4$.

Now we prove the seed length lower bound. Let $H_1, H_2, \ldots, H_{\log n}$ be any family of undirected graphs, where H_j is D_j-regular, H_j is on the vertex set $[2] \times [D_1] \times \cdots \times [D_{j-1}]$, and $\lambda(H_j) \leq \lambda_j$. We will prove by induction on j that $\log |H_j| \geq \frac{1}{16} \cdot m_j + 1$ for every $j \geq 5$. For the base case, note that trivially $|H_4| \geq 2$. By Theorem 12.2.7, we have $D_4 \geq \min\{1/(2 \cdot \lambda_4^2), (2+1)/2\}$. Both of those bounds are nontrivial, so $D_4 \geq 2$, and hence $|H_5| \geq 4$, and hence $\log |H_5| \geq 2 = \frac{1}{16} \cdot m_5 + 1$. For the inductive step, we apply Theorem 12.2.7 again to get

$$\begin{aligned}
\log |H_{j+1}| = \log |H_j| + \log D_j &\geq \min\left\{\log |H_j| + \frac{1}{4} \cdot m_j - 1, \log |H_j| + \log |H_j| - 1\right\} \\
&\geq \min\left\{\begin{array}{c} \frac{1}{16} \cdot m_j + 1 + \frac{1}{4} \cdot m_j - 1, \\ \frac{1}{16} \cdot m_j + 1 + \frac{1}{16} \cdot m_j + 1 - 1 \end{array}\right\} \\
&\geq \frac{1}{16} \cdot m_{j+1} + 1.
\end{aligned}$$

This completes the inductive step. Therefore, $\log |H_{\log n}| \geq \frac{1}{16} \cdot m_{\log n} + 1 \geq \Omega(n)$. □

13.7 A case where using heavy-duty expanders is cheaper than XORing

In this section, we prove Theorem 13.2.2. Our distinguisher f will be a member of the following class.

Definition 13.7.1 (The class $\mathcal{F}_{m,w,\ell}$)**.** For every $m, w, \ell \in \mathbb{N}$, we define $\mathcal{F}_{m,w,\ell}$ to be the class of functions $f\colon (\{0,1\}^m \times \{0,1\}^m)^\ell \to \{0,1\}$ of the form

$$f\left((x^{(1)}, y^{(1)}), \ldots, (x^{(\ell)}, y^{(\ell)})\right) = 1 \iff \forall k \in [\ell],\ h(x^{(k)}) \neq h(y^{(k)}),$$

for some function $h\colon \{0,1\}^m \to [w]$.

Let us begin by showing that INW(λ) fools $\mathcal{F}_{m,w,\ell}$, provided ℓ is large enough and λ is small enough.

Lemma 13.7.2 (Functions in $\mathcal{F}_{m,w,\ell}$ have low expectation under the INW generator)**.** *Let m and ℓ be powers of two, let $w \in \mathbb{N}$, and let $\lambda \in (0,1)$. For any function $f \in \mathcal{F}_{m,w,\ell}$ and any PRG $G\colon \mathcal{R} \to \{0,1\}^{2m\ell}$ such that $G \in \mathrm{INW}(\lambda)$, we have*

$$\mathbb{E}[f(G(U_{\mathcal{R}}))] \leq (1 - 1/w)^\ell + \lambda \cdot (w \cdot \ell + \ell - 1).$$

Proof. We will prove it by induction on ℓ. For the base case, suppose $\ell = 1$. Then f has the form $f(x,y) = 1 \iff h(x) \neq h(y)$ for some $h\colon \{0,1\}^m \to [w]$. We can write this function in the form

$$f(x,y) = \sum_{u \in [w]} 1[h(x) = u] \cdot 1[h(y) \neq u].$$

Any generator $G \in \mathrm{INW}(\lambda)$ has the form $G(a,b) = (g(a), g(H[a,b]))$, where $\lambda(H) \leq \lambda$. By the Expander Mixing Lemma (Theorem 12.2.11), we have

$$\Pr_{a,b}[h(g(a)) = u \wedge h(g(H[a,b])) \neq u] \leq \Pr_{a,a'}[h(g(a)) = u \wedge h(g(a')) \neq u] + \lambda.$$

Therefore, by the triangle inequality,

$$\mathop{\mathbb{E}}_{a,b}[f(G(a,b))] \leq \mathop{\mathbb{E}}_{a,a'}[f(g(a), g(a'))] + \lambda \cdot w.$$

When we pick two seeds a and a' independently, there is at least a $1/w$ chance that $h(g(a)) = h(g(a'))$, simply because every probability distribution over $[w]$ has collision probability at least $1/w$. Therefore,

$$\mathop{\mathbb{E}}_{a,b}[f(G(a,b))] \leq 1 - 1/w + \lambda \cdot w,$$

completing the base case.

Now, for the inductive step, suppose $\ell > 1$. Then f has the form

$$f(x,y) = f_1(x) \cdot f_2(y)$$

for some $f_1, f_2 \in \mathcal{F}_{m,w,\ell/2}$. Again, any generator $G \in \mathrm{INW}(\lambda)$ has the form $G(a,b) = (g(a), g(H[a,b]))$ where $\lambda(H) \leq \lambda$ and $g \in \mathrm{INW}(\lambda)$. By the Expander Mixing Lemma (Theorem 12.2.11), we have

$$\mathop{\mathbb{E}}_{a,b}[f(G(a,b))] \leq \mathop{\mathbb{E}}_{a}[f_1(g(a))] \cdot \mathop{\mathbb{E}}_{a}[f_2(g(a))] + \lambda.$$

Let $p = (1-1/w)^{\ell/2}$ and $\varepsilon = \lambda \cdot (w \cdot \ell/2 + \ell/2 - 1)$. By induction, we have $\mathbb{E}_a[f_1(g(a))] \leq p + \varepsilon$. Similarly, we have $\mathbb{E}_a[f_2(g(a))] \leq p + \varepsilon$, and we also trivially have $\mathbb{E}_a[f_2(g(a))] \leq 1$. Therefore,

$$\begin{aligned}
\mathop{\mathbb{E}}_{a,b}[f(G(a,b))] &\leq (p+\varepsilon) \cdot \min\{p+\varepsilon, 1\} + \lambda \\
&\leq p \cdot (p+\varepsilon) + \varepsilon \cdot 1 + \lambda \\
&\leq p^2 + 2\varepsilon + \lambda \\
&= p^2 + \lambda \cdot (w \cdot \ell + \ell - 2) + \lambda. \qquad \square
\end{aligned}$$

Now we move on to constructing an $\mathrm{INW}^{\oplus t}(\lambda)$ generator that does *not* fool $\mathcal{F}_{m,w,\ell}$. The construction is based on the existence of an injective sum of small-bias generators:

Lemma 13.7.3 (Injective sum of small-bias generators)**.** *For every $n \in \mathbb{N}$ and $\delta \in (0,1)$, there*

exist values $s > 2\log(1/\delta) + \log n$ *and* $t = \Omega(n/\log(n/\delta))$ *and there exist* δ*-biased generators* $G_1, \ldots, G_t \colon \{0,1\}^s \to \{0,1\}^n$ *such that the function* $G(x_1, \ldots, x_t) := G_1(x_1) \oplus \cdots \oplus G_t(x_t)$ *is injective.*

Proof. Pick $G_1, \ldots, G_t$ independently and uniformly at random from the set of all functions mapping $\{0,1\}^s$ to $\{0,1\}^n$. For any fixed $i \in [t]$ and any fixed set $S \subseteq [n]$, by Hoeffding's inequality, the probability that G does not fool the parity function on S with error $\delta/2$ is at most $2\exp(-\delta^2 \cdot 2^{s-1})$. Therefore, by the union bound, the probability that some G_i is not δ-biased is at most

$$t \cdot 2^{n+1} \cdot \exp(-\delta^2 \cdot 2^{s-1}).$$

Meanwhile, consider any fixed pair of distinct vectors $(x_1, \ldots, x_t), (y_1, \ldots, y_t) \in \{0,1\}^{st}$. Since the vectors are distinct, there is some i such that $x_i \neq y_i$; without loss of generality, assume $i = 1$. Observe that we have a collision $G(x_1, \ldots, x_t) = G(y_1, \ldots, y_t)$ if and only if

$$G_1(x_1) \oplus G_1(y_1) = G_2(x_2) \oplus G_2(y_2) \oplus G_3(x_3) \oplus G_3(y_3) \oplus \cdots \oplus G_t(x_t) \oplus G_t(y_t).$$

After sampling $G_2, \ldots, G_t$, the probability of the equation above with respect to the random choice of G_1 is precisely 2^{-n}. Therefore, by the union bound, the probability that G is not injective is at most

$$\binom{2^{st}}{2} \cdot 2^{-n} \leq 2^{2st-n}.$$

To complete the proof, choose $t = \Omega(n/\log(n/\delta))$ and $s = 2\log(1/\delta) + \log n + \log\log t + O(1)$ in such a way that

$$t \cdot 2^{n+1} \cdot \exp(-\delta^2 \cdot 2^{s-1}) + 2^{st-n} < 1.$$

(If δ is extremely small, then there is no positive *integer* t that is small enough to make the calculation above work, but in this case we can simply let $t = 1$ and let G_1 be the identity function on $\{0,1\}^n$.) □

Now, using Theorem 13.7.3, let us construct an $\mathrm{INW}^{\oplus t}(\lambda)$ generator whose outputs are always accepted by a certain $\mathcal{F}_{m,w,\ell}$ function.

Proposition 13.7.4. *Let $\lambda \in (0,1)$ and let m, ℓ be powers of two where $m \geq 8$. There exists $t = \Omega(m/\log(m/\lambda))$, there exists a PRG $G\colon \mathcal{R} \to \{0,1\}^{2m\ell}$, there exists $w \leq 4/\lambda$, and there exists $f \in \mathcal{F}_{m,w,\ell}$ such that $G \in \mathrm{INW}^{\oplus t}(\lambda)$ and $f(G(x)) = 1$ for every $x \in \mathcal{R}$.*

Proof. Let $G_1, \ldots, G_t\colon \{0,1\}^s \to \{0,1\}^{m/2}$ be the λ-biased generators from Theorem 13.7.3, where $s > 2\log(1/\lambda) + \log(m/2)$ and $t = \Omega(m/\log(m/\lambda))$. Let $H_i = \mathrm{Cay}(\mathbb{F}_2^{m/2}, G_i)$. Let w be a power of two in the interval $[2/\lambda, 4/\lambda]$, and note that $\log w \leq s$. Our $\mathrm{INW}^{\oplus t}(\lambda)$ generator is $\mathrm{INW}^{\oplus t}_{\mathcal{H}}$, where

$$\mathcal{H} = \begin{bmatrix} J_2 & J_4 & \cdots & J_{2^{m/4}} & H_1 & J_* \otimes K_w & J_* & J_* & \cdots & J_* \\ J_2 & J_4 & \cdots & J_{2^{m/4}} & H_2 & J_* \otimes K_w & J_* & J_* & \cdots & J_* \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ J_2 & J_4 & \cdots & J_{2^{m/4}} & H_t & J_* \otimes K_w & J_* & J_* & \cdots & J_* \end{bmatrix}.$$

This generator is indeed in $\mathrm{INW}^{\oplus t}(\lambda)$, because $\lambda(H_i) \leq \lambda$ (Theorem 12.2.6) and $\lambda(K_w) = 1/(w-1) \leq \lambda$ (Theorem 12.2.4). By construction, each output of this generator has the form

$$\Big(\bigoplus_{i=1}^{t} x_{i,1},\ \bigoplus_{i=1}^{t} (x_{i,1} \oplus G_i(z_{i,1})),$$
$$\bigoplus_{i=1}^{t} x_{i,2},\ \bigoplus_{i=1}^{t} (x_{i,2} \oplus G_i(z_{i,2})),$$
$$\ldots,\ \bigoplus_{i=1}^{t} x_{i,2\ell},\ \bigoplus_{i=1}^{t} (x_{i,2\ell} \oplus G_i(z_{i,2\ell})) \Big),$$

where, whenever j is odd, the last $\log w$ bits of $z_{i,j}$ do not all agree with the last $\log w$ bits of $z_{i,j+1}$.

By Theorem 13.7.3, there is some function $R\colon \{0,1\}^{m/2} \to \{0,1\}^{\log w}$ such that for every

$z_1, \dots, z_t \in \{0,1\}^s$, the output $R(G_1(z_1) \oplus \cdots \oplus G_t(z_t))$ consists of the last $\log w$ bits of z_1. Now define $h\colon \{0,1\}^m \to \{0,1\}^{\log w}$ by

$$h(x, y) = R(x \oplus y),$$

and let f be the corresponding $\mathcal{F}_{m,w,\ell}$ function:

$$f(x_1, x_2, \dots, x_{2\ell}) = 1 \iff \forall k \in [\ell],\ h(x_{2k-1}) \neq h(x_{2k}).$$

By construction, f accepts every output of $\mathrm{INW}_{\mathcal{H}}^{\oplus t}$. □

The final ingredient in the proof of Theorem 13.2.2 is the following seed length lower bound.

Lemma 13.7.5 (Seed length lower bound for INW(λ))**.** *Let $\lambda \in (2^{-n/32}, 1/2)$, let n be a power of two, and let $G\colon \mathcal{R} \to \{0,1\}^n$. If $G \in \mathrm{INW}(\lambda)$, then G has seed length at least*

$$\Omega\left(\log(1/\lambda) \cdot \log\left(\frac{n}{\log(1/\lambda)}\right)\right).$$

Proof. Let $H_1, \dots, H_{\log n}$ be undirected multigraphs, where H_j is a D_j-regular graph on the vertex set $[2] \times [D_1] \times \cdots \times [D_{j-1}]$ and $\lambda(H_j) \leq \lambda$. By Theorem 12.2.7, we have $D_j \geq \min\{1/(2 \cdot \lambda^2), (|H_j| + 1)/2\}$. Let j_* be the first value such that $1/\lambda^2 < |H_{j_*}| + 1$ (or $j_* = \infty$ if no such value exists).

For $2 \leq j \leq j_*$, we claim that $\log |H_j| \geq 2^{j-2} + 1$. Indeed, we start with $|H_1| = 2$. If $j_* \geq 2$, then $D_1 \geq (2+1)/2$, which implies $\log |H_2| \geq 2 = 2^{2-2} + 1$. From here, if $j \leq j_*$, then $|H_{j+1}| \geq |H_j| \cdot |H_j|/2$, so $\log |H_{j+1}| \geq 2 \log |H_j| - 1 \geq 2 \cdot (2^{j-2} + 1) - 1 = 2^{j-1} + 1$.

This shows that $\log |H_{j_*}| \geq 2^{j_*-2} + 1$, which shows that $j_* \leq \log\log(1/\lambda) + 4$. For all

$j \geq j_*$, we have $D_j \geq 1/(2 \cdot \lambda^2)$. Consequently, the final seed length $|H_{\log n}|$ is at least

$$(2\log(1/\lambda) - 1) \cdot (\log n - j_*) \geq \Omega\left(\log(1/\lambda) \cdot \log\left(\frac{n}{16\log(1/\lambda)}\right)\right). \qquad \square$$

Proof of Theorem 13.2.2. Let ℓ be a power of two in the interval $[19/\lambda, 38/\lambda]$, and let $m = n/(2\ell)$. Our assumption $\lambda \geq \frac{1000}{n}$ implies that $m \geq 8$. Therefore, we may apply Theorem 13.7.4.

By Theorem 13.7.4, there exists $t = \Omega(\lambda \cdot n/\log n)$, there exists $G \in \mathrm{INW}^{\oplus t}(\lambda)$, there exists $w \leq 4/\lambda$, and there exists $f \in \mathcal{F}_{m,w,\ell}$ such that $f(G(x)) = 1$ for every x. On the other hand, by Theorem 13.7.2, the expectation of f under any $\mathrm{INW}(\lambda')$ generator is at most

$$\begin{aligned}(1 - 1/w)^\ell + \lambda' \cdot (w+1) \cdot \ell &\leq (1 - \lambda/4)^{19/\lambda} + O(\lambda'/\lambda^2) \\ &\leq \exp(-19/4) + O(\lambda'/\lambda^2) \\ &< 0.01,\end{aligned}$$

provided we choose $\lambda' \leq c\lambda^2$ for a suitable constant c. In particular, taking $\lambda' = 0$, we see that $\mathbb{E}[f] \leq 0.01$ under the uniform distribution. Consequently, G does not fool f with error 0.99, but every generator in $\mathrm{INW}(c\lambda^2)$ fools f with error 0.01. Finally, the seed length lower bound follows from Theorem 13.7.5. $\square$

We find it interesting that the proof of Theorem 13.2.2 is based on the existence of an injective sum of small-bias generators (Theorem 13.7.3), whereas the proof of Theorem 13.1.4 is based on the existence of small-bias distributions with a small sumset (Theorem 13.5.2). These two properties are essentially opposites, and yet each is useful in its own way for establishing limitations of the $\mathrm{INW}^{\oplus t}$ generator.

13.8 Directions for future research

It would be very interesting to prove a general "XOR lemma" saying that taking the bitwise XOR of many copies of a distribution amplifies its unpredictability for bounded-width ROBPs. Proving such a general lemma might be the key to analyzing derandomized sums of INW generators.

There is a well-known variant of the INW generator in which we use an extractor to recycle the seed at each stage instead of using an expander, i.e., the recursive step is $G_{j+1}(x, y) = (G_j(x), G_j(\mathsf{Ext}(x, y)))$. This construction and its analysis are similar to the Nisan-Zuckerman PRG [NZ96]. Intriguingly, it is not clear how to carry out the proof of Theorem 13.1.3 using extractors instead of expanders. Conceivably, an extractor-based proof might be more amenable to derandomization.

We would also like to highlight the problem of determining the optimal dependence on w in Theorem 13.1.3. Can the $n^{\log(w+1)}$ term be improved to $\mathrm{poly}(n, w)$?

CHAPTER 14

FOOLING NEAR-MAXIMAL DECISION TREES

This chapter is based on the paper *Fooling Near-Maximal Decision Trees* [HL25a]. Its starting point is that standard approximate forms of bounded independence, while powerful, are not the right tool for fooling highly adaptive models when the relevant parameter is close to n. The chapter therefore introduces a new pseudorandomness notion tailored to that regime. William posted a single-author manuscript online. Then Zelin and William posted an updated, jointly-authored manuscript in July 2025.

14.1 The new notion of k-wise probable uniformity

A distribution that looks uniform on every set of k coordinates need not interact well with adaptive tests. The notion below strengthens almost k-wise uniformity in a way that is asymmetric but extremely useful for lower-bounding the acceptance probabilities of junta-like and decision-tree-like computations.

Definition 14.1.1 (k-wise probable uniformity)**.** Let X be a distribution over $\{0,1\}^n$, let $k \in [n]$, and let $\varepsilon \in [0,1]$. We say that X is *k-wise ε-probably uniform* if it satisfies either of the following two equivalent conditions.

1. For every size-k set $S \subseteq [n]$, there exists a distribution E over $\{0,1\}^k$ such that the distribution X_S can be written as the mixture distribution $X_S \equiv (1-\varepsilon) \cdot U_k + \varepsilon \cdot E$.
2. For every k-junta $f\colon \{0,1\}^n \to \{0,1\}$, we have $\mathbb{E}[f(X)] \geq (1-\varepsilon) \cdot \mathbb{E}[f]$.

We say that $G\colon \{0,1\}^s \to \{0,1\}^n$ is a *k-wise ε-probably uniform generator* if $G(U_s)$ is k-wise ε-probably uniform.

The chapter proves the existence of explicit generators with seed length essentially $(1+\alpha)k$ and then derives several consequences.

14.2 Main results

Theorem 14.2.1 (Explicit k-wise probably uniform generator)**.** *For every $n, k \in \mathbb{N}$ and $\varepsilon \in (0,1)$, there exists an explicit k-wise ε-probably uniform generator $G\colon \{0,1\}^s \to \{0,1\}^n$ with seed length*

$$s = k + O\left(k^{2/3} \cdot \log^{1/3}(k/\varepsilon) + \log(1/\varepsilon) + \log\log n\right).$$

Theorem 14.2.2 (Fooling near-maximal decision trees)**.** *Let $n, m \in \mathbb{N}$ and $\varepsilon \in (0,1)$. There exists an explicit PRG $G\colon \{0,1\}^s \to \{0,1\}^n$ that fools n-variate decision trees of size m with error ε and seed length*

$$s = \log m + O\left(\log^{2/3} m \cdot \log^{1/3}\left(\frac{\log m}{\varepsilon}\right) + \log(1/\varepsilon) + \log\log n\right).$$

Theorem 14.2.3 (Hitting set for systems of equations over $\mathbb{F}_2$)**.** *For every $n, k \in \mathbb{N}$, there exists $H \subseteq \mathbb{F}_2^n$ such that:*

1. *For every $A \in \mathbb{F}_2^{k \times n}$ and every $b \in \text{image}(A)$, there exists $x \in H$ such that $Ax = b$.*
2. *Given the parameters n and k, the set H can be enumerated in time T (and hence $|H| \leq T$), where $T = 2^{k + O((k \cdot \log k \cdot \log n)^{2/3} + \log n)}$.*

By combining these constructions with the simulations of Chen and Kabanets, we also obtain pseudorandomness consequences for linear-size Boolean circuits.

Corollary 14.2.4 (Fooling circuits over the U_2 basis)**.** *For every $n \in \mathbb{N}$ and $\alpha \in (0,3)$, there exists an explicit PRG $G\colon \{0,1\}^s \to \{0,1\}^n$ that fools n-variate U_2-circuits of size $(3-\alpha) \cdot n$ with error $n \cdot 2^{-\Omega(\alpha^6 n)}$ and seed length $s = (1 - \Omega(\alpha^2)) \cdot n$.*

Corollary 14.2.5 (A hitting set for circuits over the B_2 basis)**.** *For every $n \in \mathbb{N}$ and $\alpha \in (0, 2.5)$, there exists a value $\varepsilon = 2^{-\Omega(\alpha^2 n)}$ and a set $H \subseteq \{0,1\}^n$ such that:*

1. *H is an ε-hitting set for B_2-circuits of size $(2.5-\alpha)\cdot n$.*

2. *Given the parameters n and α, the set H can be enumerated in time $2^{(1-\Omega(\alpha^2))\cdot n+\widetilde{O}(n^{2/3})}$.*

The chapter also proves that standard small-bias technology cannot, by itself, yield these improvements in the most interesting parameter ranges.

Theorem 14.2.6 (Seed length lower bound for k-wise γ-biased generators)**.** *Let $G\colon \{0,1\}^s \to \{0,1\}^n$ be a k-wise γ-biased generator, where $k = \lfloor (1/2+\alpha)\cdot n \rfloor$ for some $\alpha \in (0,1/2]$. Then*

$$s \geq \min\{n, 2\log(1/\gamma)\} - \log(1/\alpha) - O(1).$$

14.3 Overview of the constructions

The generator construction begins from a simple but powerful observation about pairwise uniform hashing: for a fixed low-support test, most pairwise-uniform hash functions give good relative-error sampling. The difficulty is that sampling the hash function itself is too expensive. The core idea is therefore to reduce the effective dimension of the target problem by pseudorandomly partitioning the coordinates into buckets and reusing a single hash family across those buckets. This leads to the $(1+\alpha)k$-type seed length in Theorem 14.2.1. The later sections then convert this one-sided approximate uniformity guarantee into pseudorandomness for adaptive models and into hitting sets for linear-algebraic problems.

14.4 Characterizing k-wise probable uniformity

The following proposition shows the equivalence of three ways of defining k-wise probably uniform distributions.

Proposition 14.4.1 (Equivalence of three definitions of k-wise probable uniformity)**.** *Let X be a distribution over $\{0,1\}^n$, let $k \in [n]$, and let $\varepsilon \in [0,1]$. Then the following are equivalent.*

1. *For every k-junta $f\colon \{0,1\}^n \to \{0,1\}$, we have $\mathbb{E}[f(X)] \geq (1-\varepsilon)\cdot\mathbb{E}[f]$.*
2. *For every size-k set $S \subseteq [n]$ and every $z \in \{0,1\}^k$, we have $\Pr[X_S = z] \geq (1-\varepsilon)\cdot 2^{-k}$.*
3. *For every size-k set $S \subseteq [n]$, there exists a distribution E over $\{0,1\}^k$ such that one can sample from X_S by sampling from U_k with probability $1-\varepsilon$ and sampling from E with probability ε.*

Proof.

- ($1 \implies 2$) Consider the function $f(x) = 1 \iff x_S = z$.
- ($2 \implies 3$) If $\varepsilon = 0$, then for every $x \in \{0,1\}^k$, we have $\Pr[X_S = x] \geq 2^{-k}$, which implies that X_S is exactly uniform over $\{0,1\}^k$. If $\varepsilon > 0$, define $p\colon \{0,1\}^k \to \mathbb{R}$ by the formula
$$p(x) = \frac{\Pr[X_S = x] - (1-\varepsilon)\cdot 2^{-k}}{\varepsilon}.$$
Then $p(x)$ is a probability mass function: it is nonnegative because $\Pr[X_S = x] \geq (1-\varepsilon)\cdot 2^{-k}$, and it sums to 1 because X_S is a probability distribution. Let E be corresponding probability distribution.
- ($3 \implies 1$) If f is a k-junta, then there is some set $S \subseteq [n]$ of size k and some function $g\colon \{0,1\}^k \to \{0,1\}$ such that $f(x) = g(x_S)$ for all $x \in \{0,1\}^n$. Therefore,
$$\mathbb{E}[f(X)] = \mathbb{E}[g(X_S)] = (1-\varepsilon)\cdot\mathbb{E}[g(U_k)] + \varepsilon\cdot\mathbb{E}[g(E_S)] \geq (1-\varepsilon)\cdot\mathbb{E}[f]. \qquad \square$$

By definition, if X satisfies any of the three equivalent conditions in Theorem 14.4.1, then X is k-wise ε-probably uniform. The third condition in Theorem 14.4.1 motivates the name "k-wise probably uniform," but we find it more mathematically convenient to work with the first two conditions.

14.5 Constructing k-wise probably uniform generators

In this section, we present our new k-wise probably uniform generator, thereby proving Theorem 14.2.1. At the end of this section, for completeness' sake, we record the standard nonconstructive proof of the existence of nonexplicit k-wise probably uniform generators with excellent seed lengths.

14.5.1 A small family of generators, each with a good seed length

As a first step, we begin by constructing a family of generator $\mathcal{G}$, such that for any k_0-junta f, most generators $g \in \mathcal{G}$ satisfy $(1-\zeta)\cdot\mathbb{E}[f] \leq \mathbb{E}_x[f(g(x))] \leq (1+\zeta)\cdot\mathbb{E}[f]$. This construction is based on a combination of pairwise uniform hash functions and k-wise γ-biased generators.

Lemma 14.5.1 (Family of generators)**.** *For every $n, k_0 \in \mathbb{N}$ and $\zeta \in (0,1)$, there exists an explicit family $\mathcal{G}$ of PRGs $g\colon \{0,1\}^q \to \{0,1\}^n$ satisfying the following.*

1. *A generator $g \sim \mathcal{G}$ can be sampled using $O(k_0 + \log(1/\zeta) + \log\log n)$ truly random bits.*
2. *Each generator g in $\mathcal{G}$ has seed length $q = k_0 + O(\log(1/\zeta))$.*
3. *If $f\colon \{0,1\}^n \to \{0,1\}$ is a k_0-junta with expectation $\mathbb{E}[f] = \mu$, then*

$$\Pr_{g\sim\mathcal{G}}[g \text{ fools } f \text{ with error } \zeta\cdot\mu] \geq 1-\zeta.$$

Proof. Let $G_{\mathrm{sb}}\colon \{0,1\}^\ell \to \{0,1\}^n$ be a k-wise γ-biased generator where $\gamma = (\zeta/3)\cdot 2^{-3k_0/2}$ and

$$\ell = O(k_0 + \log(1/\zeta) + \log\log n).$$

Let $\mathcal{H}$ be a pairwise uniform family of hash functions $h\colon \{0,1\}^q \to \{0,1\}^\ell$. For each hash function h in $\mathcal{H}$, we define a generator $g(x) = G_{\mathrm{sb}}(h(x))$. By Theorems 12.3.4 and 12.3.6, this family is explicit and $\mathcal{G}$ can be sampled using $O(k_0 + \log(1/\zeta) + \log\log n)$ truly random bits.

For the correctness proof, define $f' \colon \{0,1\}^\ell \to \{0,1\}$ by $f'(y) = f(G_{\mathrm{sb}}(y))$ and let $\mu' = \mathbb{E}[f']$. The generator G_{sb} fools f with error $\gamma \cdot 2^{k_0/2}$ (see Theorem 12.3.7), so $|\mu - \mu'| \leq \zeta/3 \cdot \mu$. Furthermore, $\mu \geq 2^{-k_0}$ unless $f \equiv 0$, so $\mu' \geq 2^{-k_0-1}$. Therefore, by the pairwise uniformity sampling lemma (Theorem 12.3.5), we have

$$\Pr_{h \sim \mathcal{H}}[h \text{ fools } f' \text{ with error } (\zeta/3) \cdot \mu'] \geq 1 - \frac{9}{2^q \cdot \zeta^2 \cdot \mu'} \geq 1 - \frac{18 \cdot 2^{k_0}}{2^q \cdot \zeta^2} \geq 1 - \zeta,$$

provided we choose a suitable value $q = k_0 + O(\log(1/\zeta))$. Now fix an h such that the bad event above does not occur, and let g be the corresponding generator in $\mathcal{G}$, i.e., $g(x) = G_{\mathrm{sb}}(h(x))$. Then g fools f with error

$$(\zeta/3) \cdot \mu' + |\mu - \mu'| \leq \mu \cdot (\zeta/3) \cdot (2 + \zeta/3) \leq \zeta \cdot \mu. \qquad \square$$

14.5.2 Pseudorandomly partitioning the coordinates into buckets

In this subsection, we explain how to pseudorandomly partition the coordinates into buckets, $[n] = B_1 \cup \cdots \cup B_r$, such that no single bucket gets too many of the k coordinates we care about. To be more precise, we construct a *balanced partition generator*, defined as follows.

Definition 14.5.2 (Balanced partition generator [MZ13])**.** A (k, k_0, δ)-*balanced partition generator* is a function $G_{\mathrm{vars}} \colon \{0,1\}^a \to [r]^n$ such that for every set $S \subseteq [n]$ with $|S| \leq k$, with probability at least $1 - \delta$ over a uniform random choice of seed $x \in \{0,1\}^a$, for every bucket $j \in [r]$, we have $|\{i \in S : G_{\mathrm{vars}}(x)_i = j\}| \leq k_0$.

Theorem 14.5.2 is due to Meka and Zuckerman, who used the term "balanced hash family" [MZ13, Definition 4.9]. We use the term "balanced partition generator" to avoid confusion with the hash functions that appear in the proof of Theorem 14.5.1. Our balanced partition generator will essentially consist of a d-wise γ-biased generator for appropriate values d and γ. The analysis will be based on the following bound on the moments of a sum

of independent Bernoulli random variables [SSS95].[1]

Theorem 14.5.3 (Moment bound for independent Bernoulli sums [SSS95])**.** *Let $X_1, \ldots, X_k$ be independent $\{0,1\}$-valued random variables. Let $X = \sum_{i=1}^{k} X_i$, let $\mu_i = \mathbb{E}[X_i]$, and let $\mu = \sum_{i=1}^{k} \mu_i$. Then for every even positive integer t, we have*

$$\mathbb{E}[(X-\mu)^t] \leq \max\{t^t, (t\mu)^{t/2}\}.$$

Theorem 14.5.3 can be improved in some parameter regimes [Sko22], but the simple bound in Theorem 14.5.3 suffices for our purposes. Using Theorem 14.5.3, we now present a tail bound for sums of random variables that satisfy a certain "near t-wise independence" condition. Similar bounds were proven in several previous papers [LRTV09, CRSW13, SVW17], and our proof is almost identical to their proofs.

Corollary 14.5.4 (Tail bound for sums of nearly t-wise independent random variables)**.** *Let $X_1, \ldots, X_k$ be $\{0,1\}$-valued random variables and let $\mu_1, \ldots, \mu_k \in [0,1]$. Let $X = \sum_{i=1}^{k} X_i$ and $\mu = \sum_{i=1}^{k} \mu_i$. Let t be an even positive integer, let $\gamma \in (0,1)$, and assume that for every set $S \subseteq [k]$ with $|S| \leq t$, we have*

$$\left| \mathbb{E}\left[\prod_{i \in S} X_i\right] - \prod_{i \in S} \mu_i \right| \leq \gamma.$$

Then for every $\Delta > 0$, we have

$$\Pr[|X-\mu| \geq \Delta] \leq \left(\frac{t}{\Delta}\right)^t + \left(\frac{\sqrt{\mu t}}{\Delta}\right)^t + \gamma \cdot \left(\frac{2k}{\Delta}\right)^t.$$

Proof. Sample $X'_1, \ldots, X'_k \in \{0,1\}$ independently, where $\mathbb{E}[X'_i] = \mu_i$, and let $X' = \sum_{i=1}^{k} X'_i$.

1. The exact statement of Theorem 14.5.3 does not appear in Schmidt, Siegel, and Srinivasan's work [SSS95], but it follows from the proof of item "(III)" in their "Theorem 4."

Then

$$\begin{aligned}
\Pr[|X-\mu| \geq \Delta] &= \Pr[(X-\mu)^t \geq \Delta^t] \\
&\leq \Delta^{-t} \cdot \mathbb{E}[(X-\mu)^t] \\
&= \Delta^{-t} \cdot \sum_{i=0}^{t} \binom{t}{i} (-\mu)^{t-i} \cdot \mathbb{E}[X^i] \\
&= \Delta^{-t} \cdot \sum_{i=0}^{t} \binom{t}{i} (-\mu)^{t-i} \cdot \sum_{j_1,\ldots,j_i \in [k]} \mathbb{E}[X_{j_1} X_{j_2} \cdots X_{j_i}] \\
&\leq \Delta^{-t} \cdot \sum_{i=0}^{t} \binom{t}{i} \cdot \left(\begin{array}{c} (-\mu)^{t-i} \cdot \sum\limits_{j_1,\ldots,j_i \in [k]} \mu_{j_1} \cdots \mu_{j_i} \\ \\ + \mu^{t-i} \cdot k^i \cdot \gamma \end{array} \right) \\
&= \Delta^{-t} \cdot \left(\mathbb{E}[(X'-\mu)^t] + \gamma \cdot \sum_{i=0}^{t} \binom{t}{i} \mu^{t-i} \cdot k^i \right) \\
&= \Delta^{-t} \cdot \left(\mathbb{E}[(X'-\mu)^t] + \gamma \cdot (\mu + k)^t \right) \\
&\leq \left(\frac{t}{\Delta} \right)^t + \left(\frac{\sqrt{\mu t}}{\Delta} \right)^t + \gamma \cdot \left(\frac{2k}{\Delta} \right)^t .
\end{aligned}$$

Here we use Markov's inequality, the binomial theorem twice, and finally Theorem 14.5.3. □

Given Theorem 14.5.4, we are ready to construct our balanced partition generator.

Lemma 14.5.5 (Balanced partition generator)**.** *Let $n, k, r \in \mathbb{N}$ and $\delta \in (0,1)$. Assume r is a power of two and $r \leq k \leq n$. There exists an explicit (k, k_0, δ)-balanced partition generator $G_{\text{vars}} \colon \{0,1\}^a \to [r]^n$, where*

$$k_0 = k/r + O\left(\sqrt{k/r \cdot \log(r/\delta)} + \log(r/\delta) \right),$$

with seed length

$$a = O\left(\log(r/\delta) \cdot \log\left(2 \cdot \left\lceil \frac{rk}{\log(r/\delta)} \right\rceil \right) + \log\log n \right).$$

Proof. Identify $[r]^n$ with $\{0,1\}^{n \log r}$. We let G_{vars} be a $(t \log r)$-wise γ-biased generator for appropriate values

$$t = \log(3r/\delta)$$
$$\gamma = \frac{\delta}{3r} \cdot \left(\frac{t}{rk}\right)^{t/2}.$$

The seed length bound follows from Theorem 12.3.6. For the correctness proof, assume without loss of generality that $|S| = k$. Sample $Z \in [r]^n$ using the generator. Fix any bucket $j \in [r]$. For each $i \in S$, let X_i indicate whether $Z_i = j$. Then for any set $T \subseteq S$ with $|T| \le t$, the value $\prod_{i \in T} X_i$ can be expressed in terms of the underlying bits of Z as a conjunction of at most $t \log r$ literals. Therefore, by Theorem 12.3.7, we have $|\mathbb{E}[\prod_{i \in T} X_i] - r^{-|T|}| \le \gamma$. Therefore, by Theorem 14.5.4, for every $\Delta > 0$, we have

$$\Pr\left[\sum_{i \in S} X_i \ge k/r + \Delta\right] \le \left(\frac{t}{\Delta}\right)^t + \left(\frac{\sqrt{kt/r}}{\Delta}\right)^t + \gamma \cdot \left(\frac{2k}{\Delta}\right)^t.$$

We choose $\Delta = \max\left\{2t, 2\sqrt{kt/r}\right\}$. Then we get

$$\begin{aligned}\Pr\left[\sum_{i \in S} X_i \ge k/r + \Delta\right] &\le 2^{-t} + 2^{-t} + \gamma \cdot \left(\sqrt{\frac{rk}{t}}\right)^t \\ &\le \frac{\delta}{3r} + \frac{\delta}{3r} + \frac{\delta}{3r}\end{aligned}$$

due to our choices of t and γ. The union bound over r buckets completes the proof. □

For comparison, Lovett, Reingold, Trevisan, and Vadhan constructed an explicit (k, k_0, δ)-balanced partition generator for the special case $k = \Theta(r \cdot \log(1/\delta))$, with $k_0 = O(k/r)$ and seed length $a = O(\log n + \log(r/\delta) \cdot \log(r \cdot \log(1/\delta)))$ [LRTV09]. For any k, one can also use Gopalan, Kane, and Meka's PRG for Fourier shapes [GKM18] to construct a (k, k_0, δ)-

balanced partition generator with the same value of k_0 as in Theorem 14.5.5 and with seed length $a = \widetilde{O}(\log(n/\delta))$.

14.5.3 The full k-wise probably uniform generator

Proof of Theorem 14.2.1. Let $G_{\text{vars}} \colon \{0,1\}^a \to [r]^n$ be the (k, k_0, δ)-balanced partition generator from Theorem 14.5.5 with parameters $\delta = \varepsilon/3$ and $r = (k/\log(k/\varepsilon))^{1/3}$, or to be more precise, r is the largest power of two that is at most $(k/\log(k/\varepsilon))^{1/3}$. Let $\mathcal{G}$ be the family of generators $g \colon \{0,1\}^q \to \{0,1\}^n$ from Theorem 14.5.1, using $\zeta = \varepsilon/(3r)$ and using the value k_0 from G_{vars}. The final generator G is defined as follows.

1. Sample a partition $Z = (Z_1, \ldots, Z_n) \in [r]^n$ using G_{vars}.
2. Sample a generator $g \sim \mathcal{G}$.
3. Sample seeds $X^{(1)}, \ldots, X^{(r)} \in \{0,1\}^q$ independently and uniformly at random.
4. Output $Y \in \{0,1\}^n$, where
$$Y_i = g(X^{(Z_i)})_i$$
for every $i \in [n]$.

To prove that this works, let $f \colon \{0,1\}^n \to \{0,1\}$ be a conjunction of k literals, say

$$f(x) = \bigwedge_{i \in S} (x_i \oplus b_i)$$

where $S \subseteq [n]$, $|S| = k$, and $b_i \in \{0,1\}$ for every $i \in S$. We will prove that $\mathbb{E}[f(X)] \geq (1-\varepsilon) \cdot 2^{-k}$, which is sufficient by Theorem 14.4.1.

For each bucket $j \in [r]$, let $B_j = Z^{-1}(j)$. The definition of a balanced partition generator ensures that except with probability $\varepsilon/3$ over the choice of Z, we have $|S \cap B_j| \leq k_0$ for every $j \in [r]$. Let E_1 be this "good" event. Fix any choice of Z such that E_1 occurs.

For each $j \in [r]$, define $f_j \colon \{0,1\}^n \to \{0,1\}$ by

$$f_j(x) = \bigwedge_{i \in S \cap B_j} (x_i \oplus b_i),$$

so $f(x) = f_1(x) \wedge \cdots \wedge f_r(x)$. By Theorem 14.5.1 and the union bound over the r buckets, except with probability $\varepsilon/3$ over the choice of $g \sim \mathcal{G}$, we have

$$\mathop{\mathbb{E}}_{x \in \{0,1\}^q}[f_j(g(x))] \geq \left(1 - \frac{\varepsilon}{3r}\right) \cdot \mathbb{E}[f_j]$$

for every $j \in [r]$. Let E_2 be this "good" event. Fix any choice of g such that E_2 occurs.

For any such fixing of Z and g, with respect to the choice of $X^{(1)}, \ldots, X^{(r)}$ alone, we have

$$\begin{aligned}
\mathop{\mathbb{E}}_{X^{(1)},\ldots,X^{(r)}}[f(Y)] &= \prod_{j=1}^{r} \mathop{\mathbb{E}}_{X^{(j)}}[f_j(g(X^{(j)}))] \\
&\geq \prod_{j=1}^{r} \left(1 - \frac{\varepsilon}{3r}\right) \cdot \mathbb{E}[f_j] = \left(1 - \frac{\varepsilon}{3r}\right)^r \cdot 2^{-k} \geq (1 - \varepsilon/3) \cdot 2^{-k}
\end{aligned}$$

by Bernoulli's inequality. Therefore, with respect to all the randomness, we have

$$\begin{aligned}
\mathbb{E}[f(Y)] \geq \Pr[f(Y) = 1 \text{ and } E_1 \text{ and } E_2] &= \Pr[E_1] \cdot \Pr[E_2 \mid E_1] \cdot \Pr[f(Y) = 1 \mid E_1, E_2] \\
&\geq (1 - \varepsilon/3) \cdot (1 - \varepsilon/3) \cdot (1 - \varepsilon/3) \cdot 2^{-k} \\
&\geq (1 - \varepsilon) \cdot 2^{-k}
\end{aligned}$$

by another application of Bernoulli's inequality.

Now let us bound the seed length. By Theorem 14.5.5, the cost of sampling Z is

$$O\left(\log(r/\varepsilon) \cdot \log\left(2 \cdot \left\lceil \frac{rk}{\log(r/\varepsilon)} \right\rceil\right) + \log\log n\right)$$

$$
\begin{aligned}
&\leq O\left(\log(k/\varepsilon) \cdot \log\left(2 \cdot \left\lceil \frac{k}{\log(k/\varepsilon)} \right\rceil\right) + \log\log n\right) \\
&\leq O\left(\log(k/\varepsilon) \cdot \left(\frac{k}{\log(k/\varepsilon)}\right)^{2/3} + \log(k/\varepsilon) + \log\log n\right) \\
&= O(k^{2/3} \cdot \log^{1/3}(k/\varepsilon) + \log(k/\varepsilon) + \log\log n).
\end{aligned}
$$

Furthermore, the parameter k_0 is given by

$$
k_0 = k/r + O\left(\sqrt{k/r \cdot \log(r/\varepsilon)} + \log(r/\varepsilon)\right) \leq k/r + O\left(\sqrt{k/r \cdot \log(k/\varepsilon)} + \log(k/\varepsilon)\right).
$$

Therefore, by Theorem 14.5.1, the cost of sampling $g \sim \mathcal{G}$ is

$$
\begin{aligned}
&O\left(k_0 + \log(k/\varepsilon) + \log\log n\right) \\
&\quad = O(k^{2/3} \cdot \log^{1/3}(k/\varepsilon) + k^{1/3} \cdot \log^{2/3}(k/\varepsilon) + \log(k/\varepsilon) + \log\log n) \\
&\quad = O(k^{2/3} \cdot \log^{1/3}(k/\varepsilon) + \log(k/\varepsilon) + \log\log n).
\end{aligned}
$$

Finally, the cost of sampling $X^{(1)}, \ldots, X^{(r)}$ is

$$
\begin{aligned}
r \cdot q &= r \cdot k_0 + O(r \cdot \log(k/\varepsilon)) \\
&= k + O(k^{2/3} \cdot \log^{1/3}(k/\varepsilon) + k^{1/3} \log^{2/3}(k/\varepsilon) + \log(k/\varepsilon)) \\
&= k + O(k^{2/3} \cdot \log^{1/3}(k/\varepsilon) + \log(k/\varepsilon)).
\end{aligned}
$$

□

14.5.4 Nonexplicit k-wise probably uniform generators

At this point, we have completed our explicit k-wise uniform generator construction. We now use a standard probabilistic argument to show the existence of nonexplicit k-wise probably uniform generators with a very good seed length.

Proposition 14.5.6 (Nonexplicit k-wise probably uniform generator)**.** *For every* $n, k \in \mathbb{N}$

and every $\varepsilon \in (0,1)$, there exists a k-wise ε-probably uniform generator $G\colon \{0,1\}^s \to \{0,1\}^n$ with seed length

$$s = k + \log k + 2\log(1/\varepsilon) + \log\log(n/k) + O(1).$$

Proof. Pick G uniformly at random. For every function f that is a conjunction of k literals, let $Z_f = \sum_{x\in\{0,1\}^s} f(G(x))$. Then Z_f is a sum of 2^s independent $\{0,1\}$-valued random variables with mean $\mu := \mathbb{E}[Z_f] = 2^{s-k}$. Therefore, by the Chernoff bound,

$$\Pr[Z_f < (1-\varepsilon)\cdot\mu] \leq \exp(-\varepsilon^2\mu/2).$$

By the union bound, it follows that

$$\begin{aligned}\Pr[\text{there exists } f \text{ such that } Z_f < (1-\varepsilon)\cdot\mu] &\leq \binom{n}{k}\cdot 2^k \cdot \exp(-\varepsilon^2\mu/2)\\ &\leq (2en/k)^k \cdot \exp(-\varepsilon^2 2^{s-k}/2).\end{aligned}$$

This probability is less than 1 if we choose a suitable value $s = k + \log k + 2\log(1/\varepsilon) + \log\log(n/k) + O(1)$. Now suppose G is such that $Z_f \geq (1-\varepsilon)\cdot\mu$ for every f that is a conjunction of k literals. Let $g\colon \{0,1\}^n \to \{0,1\}$ be a k-junta. Then we can write $g = \sum_{i=1}^m f_i$ where each f_i is a conjunction of k literals, hence

$$\mathop{\mathbb{E}}_x[g(G(x))] = \sum_{i=1}^m 2^{-s}\cdot Z_{f_i} \geq \sum_{i=1}^m 2^{-s}\cdot(1-\varepsilon)\cdot 2^{s-k} = (1-\varepsilon)\cdot m\cdot 2^{-k} = (1-\varepsilon)\cdot\mathbb{E}[g]. \quad \square$$

14.6 Implications of k-wise probable uniformity

In this section, we will show that every k-wise probably uniform distribution fools decision trees. In fact, we will show that such distributions fool a more general model, called the *subcube partition model*.

Definition 14.6.1 (The subcube partition model)**.** A *subcube partition* f is a collection

of *terms* $f_1, \ldots, f_m$ and *values* $b_1, \ldots, b_m \in \{0,1\}$. Each term $f_i \colon \{0,1\}^n \to \{0,1\}$ is a conjunction of literals, and the sets $f_1^{-1}(1), \ldots, f_m^{-1}(1)$ must partition the domain $\{0,1\}^n$. That is, for every $x \in \{0,1\}^n$, we have $\sum_{i=1}^m f_i(x) = 1$. The subcube partition computes the function $f \colon \{0,1\}^n \to \{0,1\}$ defined by

$$f(x) = \sum_{i=1}^{m} f_i(x) \cdot b_i.$$

The *width* of a term f_i is the number of literals in the term. The width of the subcube partition is the maximum width of any term. The *size* of the subcube partition is the number of terms (m).

Every width-k subcube partition has size at most 2^k, because $1 = \sum_{i=1}^m \mathbb{E}[f_i] \geq m \cdot 2^{-k}$. A decision tree of depth k and size m can be simulated by a subcube partition of width k and size m: for each leaf u, we construct a term f_u that indicates whether the tree reaches the leaf u on a given input. The converse does not hold. In fact, there exist subcube partitions of width k that cannot be simulated by decision trees of depth $k^{1.99}$ [Sav02, KRDS15, GPW18, AKK16]. We now explain why k-wise probably uniform generators fool subcube partitions.

Lemma 14.6.2 (k-wise probable uniformity fools subcube partitions)**.** *Let X be a distribution over $\{0,1\}^n$ that is k-wise ε-probably uniform. Then:*

- *X fools width-k subcube partitions (hence also depth-k decision trees) with error ε.*
- *X fools size-m subcube partitions (hence also size-m decision trees) with error $\varepsilon + m \cdot 2^{-(k+1)}$.*

Proof. Let $f \colon \{0,1\}^n \to \{0,1\}$ be a function computed by a subcube partition with terms $f_1, \ldots, f_m$ and values $b_1, \ldots, b_m$. Let $S \subseteq [m]$ be the set of terms of width at most k. We will show that X fools f with error $\varepsilon + \sum_{i \notin S} \mathbb{E}[f_i]$. To prove it, sample $R \in \{0,1\}^n$ uniformly at

random. Then

$$\begin{aligned}
\mathbb{E}[f(X)] &= \sum_{i=1}^{m} b_i \cdot \mathbb{E}[f_i(X)] \geq \sum_{i \in S} b_i \cdot \mathbb{E}[f_i(X)] \\
&\geq \sum_{i \in S} b_i \cdot (1-\varepsilon) \cdot \mathbb{E}[f_i] = (1-\varepsilon) \cdot \mathbb{E}\left[\sum_{i \in S} b_i \cdot f_i(R)\right] \\
&\geq \mathbb{E}\left[\sum_{i \in S} b_i \cdot f_i(R)\right] - \varepsilon \\
&= \mathbb{E}\left[f(R) - \sum_{i \notin S} b_i \cdot f_i(R)\right] - \varepsilon \\
&\geq \mathbb{E}[f] - \sum_{i \notin S} \mathbb{E}[f_i] - \varepsilon.
\end{aligned}$$

Now we bound the expectation from above. Let $\overline{f} = 1 - f$. Since $\overline{f}$ can also be computed by a subcube partition with the same terms $f_1, \ldots, f_m$, we have

$$\mathbb{E}[f(X)] = 1 - \mathbb{E}\left[\, \overline{f}(X) \,\right] \leq 1 - \mathbb{E}\left[\, \overline{f} \,\right] + \varepsilon + \sum_{i \notin S} \mathbb{E}[f_i] = \mathbb{E}[f] + \varepsilon + \sum_{i \notin S} \mathbb{E}[f_i].$$

The lemma follows, because $\mathbb{E}[f_i] \leq 2^{-(k+1)}$ whenever $i \notin S$. □

By combining Theorem 14.2.1 (our k-wise probably uniform generator) with Theorem 14.6.2, we now prove the following theorem, which generalizes Theorem 14.2.2.

Theorem 14.6.3 (Fooling near-maximal subcube partitions)**.** *Let $n, m \in \mathbb{N}$ and $\varepsilon \in (0,1)$. There exists an explicit PRG $G\colon \{0,1\}^s \to \{0,1\}^n$ that fools n-variate subcube partitions of size m with error ε and seed length*

$$s = \log m + O\left(\log^{2/3} m \cdot \log^{1/3}\left(\frac{\log m}{\varepsilon}\right) + \log(1/\varepsilon) + \log\log n\right). \tag{14.1}$$

Proof. We use our k-wise $(\varepsilon/2)$-probably uniform generator, where $k = \log m + \log(2/\varepsilon)$. By

Theorem 14.6.2, the generator fools size-m subcube partitions with error $\varepsilon/2 + m \cdot 2^{-k} = \varepsilon$. By Theorem 14.2.1, the seed length is

$$k + O(k^{2/3} \cdot \log^{1/3}(k/\varepsilon) + \log(1/\varepsilon) + \log\log n),$$

which, after substituting the choice of k, simplifies to Eq. (14.1). □

14.7 Hitting sets for systems of equations over $\mathbb{F}_2$ and for B_2-circuits

In this section, we present our *hitting set* for systems of equations over $\mathbb{F}_2$, thereby proving Theorem 14.2.3. Next, we show that such hitting sets can hit circuits over the B_2 basis, thus proving Theorem 14.2.5. Finally, we present a more explicit construction of *hitting set generators* for systems of equations over $\mathbb{F}_2$, where given the seed, we can output the corresponding string in time $\text{poly}(n)$.

14.7.1 Rank condenser

First, we use a *rank condenser*, due to Forbes and Guruswami [FG15] to "condense" the number of variables from n to $O(k \cdot \log n)$.

Definition 14.7.1 (k-rank condenser)**.** Let $\mathbb{F}$ be a field and let $n \geq k \geq 1$. A collection of matrices $\mathcal{M} \subseteq \mathbb{F}^{n \times n'}$ is a *k-rank condenser* if, for every matrix $A \in \mathbb{F}^{k \times n}$ with $\text{rank}(A) = k$, there exists $M \in \mathcal{M}$ such that $\text{rank}(AM) = k$.

We say that $\mathcal{M}$ is *explicit* if, given an index $i \in [|\mathcal{M}|]$, the i-th matrix of $\mathcal{M}$ can be constructed in time $\text{poly}(n)$.

Remark 14.7.2. Stronger "lossless" variants—which bound how many matrices in $\mathcal{M}$ can cause rank loss or how much total rank loss can occur—have been studied (see, e.g., Forbes

and Guruswami [FG15]). The simpler notion above suffices for our purposes.

The following theorem, due to Forbes and Guruswami, shows that we can construct such condensers explicitly over $\mathbb{F}_2$ while keeping the output dimension only $O(k \cdot \log n)$.

Theorem 14.7.3. *Let $n \geq k \geq 1$. There is an explicit* k-rank condenser $\mathcal{M} \subseteq \mathbb{F}_2^{n \times 4k \log n}$ *with* $|\mathcal{M}| = \text{poly}(n)$.

Proof. This follows from Forbes and Guruswami's work [FG15, Corollary 8.7, preprint version], by setting the parameters appropriately. □

14.7.2 Partition the variables

For the sake of brevity, we define the following notation.

Definition 14.7.4. Let $H \subseteq \mathbb{F}_2^n$. We say that H *hits codimension* k if, for every affine subspace of codimension k, there exists $x \in H$ in this affine subspace. Equivalently, for every $A \in \mathbb{F}_2^{k \times n}$ and every $b \in \text{image}(A)$, there exists $x \in H$ such that $Ax = b$.

Our goal is to construct an H that hits codimension k. We can split the n variables into ℓ consecutive blocks of arbitrary size. For any $A \subseteq \mathbb{F}_2^{k \times n}$, this induces a column partition, giving a column partition $A = [A_1 \ A_2 \ \dots \ A_t]$, where $A \subseteq \mathbb{F}_2^{n_i}$ and $n_1 + \cdots + n_t = n$. Without loss of generality, we assume that A has full rank. Write k_i for the incremental rank contributed by block A_i, i.e., $k_i = \text{rank}([A_1 \ A_2 \ \dots \ A_i]) - \text{rank}([A_1 \ A_2 \ \dots \ A_{i-1}])$, so $k_1 + \cdots + k_t = k$. Andreev, Clementi and Rolim [ACR97] stated the result that, if $H_i \subseteq \mathbb{F}_2^{n_i}$ hits codimension k_i for every i, then there is some x in the Cartesian product $H_1 \times H_2 \times \cdots \times H_t$ such that $Ax = b$.

However, they skipped the proof, so we complete the proof in this subsection. We began by showing that after fixing the first $i-1$ blocks, the feasible assignments to the i-th block form an affine subspace of codimension k_i. In the following, we focus on the case of partitioning A into two blocks, which will turn out to be sufficient for analyzing the general case.

Lemma 14.7.5. *Let $\mathbb{F}$ be a field. Let $A_1 \in \mathbb{F}^{k\times n_1}$ and $A_2 \in \mathbb{F}^{k\times n_2}$. Let $b \in \text{image}([A_1\ A_2])$, and define $V = \left\{ y \in \mathbb{F}_2^{n_1} \;\middle|\; \exists\, z \in \mathbb{F}_2^{n_2} \text{ such that } A_1 y + A_2 z = b \right\}$. Then V is an affine space with codimension* $\text{rank}([A_1\ A_2]) - \text{rank}(A_2)$.

Proof. Since $b \in \text{image}([A_1\ A_2])$, we know there exists (y_*, z_*) such that $A_1 y_* + A_2 z_* = b$. Let $W = A_1^{-1}\big(\text{image}(A_2)\big)$. We claim that $V = W + y_*$. Indeed, if $y \in W + y_*$, then $y - y_* \in W$, so there is some z such that $A_1(y - y_*) = A_2 z$. Hence

$$A_1 y + A_2(z_* - z) = A_1 y_* + A_2 z_* = b,$$

so $y \in V$. Conversely, if $y \in V$, then there is some z such that $A_1 y + A_2 z = b$, and consequently

$$A_1(y - y_*) = b - A_2 z - A_1 y_* = A_2(z_* - z),$$

showing that $y - y_* \in W$, i.e. $y \in W + y_*$.

Now we are going to show that $\text{codim}(W) = \text{rank}([A_1\ A_2]) - \text{rank}(A_2)$. Let $b_1, \ldots, b_s$ be a basis of W. Extend this to a basis $b_1, \ldots, b_{n_1}$ of $\mathbb{F}^{n_1}$, and set $U = \text{span}(b_{s+1}, \ldots, b_{n_1})$, so $\mathbb{F}^{n_1} = U + W$. Because $\ker(A_1) \subseteq W$, the map A_1 is injective on U. Hence $\dim(A_1 U) = \dim(U) = \text{codim}(W)$. On the other hand, let us show that $\dim(A_1 U) = \text{rank}([A_1\ A_2]) - \text{rank}(A_2)$. Observe that $\text{image}(A_2) \cap A_1 U = \{0\}$, because otherwise U and W would have nontrivial intersection. Furthermore, clearly,

$$A_1 U + \text{image}(A_2) \subseteq \text{image}([A_1\ A_2]) = A_1 U + \text{image}(A_2).$$

Conversely, consider any point $A_1 y + A_2 z \in \text{image}(A_1) + \text{image}(A_2)$. We can decompose $y = y_U + y_W$ for some $y_U \in U$ and $y_W \in W$. By the definition of W, there is some z' such that $A_1 y_W = A_2 z'$. Therefore, $A_1 y + A_2 z = A_1 y_U + A_2(z + z') \in A_1 U + \text{image}(A_2)$. This

shows that $A_1U + \text{image}(A_2) = \text{image}([A_1\ A_2])$. Therefore,

$$\begin{aligned}\text{rank}([A_1\ A_2]) &= \dim(\text{image}([A_1\ A_2])) \\ &= \dim(A_1U) + \dim(\text{image}(A_2)) = \dim(A_1U) + \text{rank}(A_2).\end{aligned}$$ □

Corollary 14.7.6 ([ACR97])**.** *Let $A = [A_1\ A_2\ \cdots\ A_t] \in \mathbb{F}^{k\times n}$ be a matrix of rank k, where each block $A_i \in \mathbb{F}^{k\times n_i}$ and $n_1+\cdots+n_t = n$. For every $i \in [t]$, let $k_i = \text{rank}\big([A_i\ A_2\ \cdots\ A_t]\big) - \text{rank}\big([A_{i+1}\ A_2\ \cdots\ A_t]\big)$. For each $i \in [t]$, let $H_i \subseteq \mathbb{F}_2^{n_i}$, and assume H_i hits codimension k_i. Then for every $b \in \text{image}(A)$, there exists a vector x in the Cartesian product*

$$H_1 \times H_2 \times \cdots \times H_t \subseteq \mathbb{F}_2^n$$

such that $Ax = b$.

Proof. We prove it by induction on ℓ. In the base case, when $\ell = 1$, the corollary follows immediately from the definition of hitting rank k_1. Now suppose $\ell > 1$. Define $A_{>1} = [A_2\ A_3\ \cdots\ A_t]$, and define

$$V = \{y \in \mathbb{F}_2^{n_1} : \exists z \in \mathbb{F}_2^{n-n_1} \text{ such that } A_1y + A_{>1}z = b\}.$$

By Theorem 14.7.5, V is an affine space with codimension k_1. Therefore, there exists $y \in H_1 \cap V$. By the definition of V, $b - A_1y \in \text{image}(A_{>1})$. Therefore, by induction, there exists $z \in H_2 \times \cdots \times H_t$ such that $A_{>1}z = b - A_1y$. Let $x = (y, z)$. Then $x \in H_1 \times \cdots \times H_t$, and $Ax = A_1y + A_{>1}z = b$. □

14.7.3 Brute-force construction

In this subsection, we use a brute-force method to construct a hitting set that hits codimension k. Our method is similar to the one used in Naor, Schulman, and Srinivasan's work [NSS95].

On its own, this brute-force method is too slow to prove Theorem 14.2.3. However, we will only apply the brute-force method after reducing the length of binary strings we are searching, so we can afford the exponential time cost. Note the size of our construction matches that of the hitting set obtained by the standard probabilistic method.

Lemma 14.7.7 (Brute-force hitting set for systems of equations)**.** *For every $n, k \in \mathbb{N}$, there exists $H \subseteq \mathbb{F}_2^n$ of size $2^{k+O(\log(nk))}$ that hits codimension k, which can be constructed in time $O(nk \cdot 2^{kn+n+2k})$.*

Proof. Let H be the hitting set we are going to construct, and let $\mathcal{U}$ be the set of feasible systems of k linear equations over $\mathbb{F}_2$ that have not yet been satisfied by any element in H. Note that hitting codimension k is equivalent to intersecting every nonempty solution space defined by k linear equations. At the beginning, H is empty and $\mathcal{U}$ consists all the feasible systems of k linear equations over $\mathbb{F}_2$. Note that each feasible system of k linear equations can be written in the form $Ax = b$ where $A \in \mathbb{F}_2^{k \times n}$ and $b \in \text{image}(A) \subseteq \mathbb{F}_2^k$. Thus, $|\mathcal{U}| \leq |\mathbb{F}_2^{k \times n}| \cdot |\mathbb{F}_2^k| = 2^{k(n+1)}$.

The algorithm works as follows: in each round, we find an element $x \in \mathbb{F}_2^n$ such that x satisfies $1/2^k$ fraction of $\mathcal{U}$ and add this x to H, so the size of $\mathcal{U}$ is shrunk by a factor of $(1 - 1/2^k)$ in each round. The existence of such x is guaranteed by the averaging argument. After d rounds, the number of unsatisfied systems will be at most $2^{k(n+1)}\left(1 - 2^{-k}\right)^d$. Using the inequality $1 - 2^{-k} \leq e^{-2^{-k}}$, this quantity drops below 1 whenever $d > k(n+1)\, 2^k \ln 2$. Consequently, after $d = 2^{\,k+O(\log(nk))}$ rounds of searching, we obtain a hitting set H with $|H| = d = 2^{k+O(\log(nk))}$ such that for every matrix $A \in \mathbb{F}_2^{k \times n}$ and every vector $b \in \text{image}(A)$, there exists an $x \in H$ satisfying $Ax = b$.

In each round, we need to find an element $x \in \mathbb{F}_2^n$ and test if this x satisfies at least $1/2^k$ fraction of $\mathcal{U}$. Testing if a vector satisfies an equation takes at most $O(nk)$ time. So in the i-th round, the time required to find such a x is at most $O(nk \cdot 2^{n+k(n+1)} \cdot (1 - 1/2^k)^{i-1})$.

Thus, the total running time is at most

$$O\left(nk \cdot 2^{n+k(n+1)} \cdot \sum_{i=0}^{\infty}(1-1/2^k)^i\right) = O(nk \cdot 2^{n+kn+2k}). \qquad \square$$

14.7.4 Our final hitting set for systems of equations

Proof of Theorem 14.2.3. Without loss of generality, we assume that $k \geq \log n$; otherwise, we can simply use a small-bias distribution as described in the paragraph following the statement of Theorem 14.2.3. First we use Theorem 14.7.3 to construct a k-rank condenser $\mathcal{M} \subseteq \mathbb{F}_2^{n \times 4k \log n}$, where $|\mathcal{M}| = \text{poly}(n)$. Then we partition the variables into t blocks of equal size, where $t \approx k^{2/3}$ (the exact value will be specified later). Without loss of generality, we assume that n'/t is an integer. For each $i \in \{0, 1, \ldots, n'/t\}$, we use Theorem 14.7.7 to construct $H_i \subseteq \mathbb{F}_2^{n'/t}$ that hits codimension i, as defined in Theorem 14.7.4. Then we combine them by taking a Cartesian product. Thus, the overall construction is

$$H = \bigcup_{\substack{k_1,\ldots,k_t \in \mathbb{N} \\ k_1+\cdots+k_t=k}} \{Mx : M \in \mathcal{M} \text{ and } x \in H_{k_1} \times \cdots \times H_{k_t}\}.$$

We first prove the construction is efficient (Item 2) and then prove the construction is correct (Item 1).

(Item 2). By Theorem 14.7.7, we know the size of each H_i is $|H_i| = O(\frac{n'}{t} \cdot i2^i)$, and the total running time to construct these hitting sets over n'/t variables is $\sum_{i=1}^{n'/t} O(2^{n'/t+i(n'/t+2)}\frac{n'}{t}i) \leq O(2^{n'/t+(n'/t)^2+2(n'/t)}(\frac{n'}{t})^3) = 2^{O((n'/t)^2)}$.

For one partition of k, namely $k_1 + \cdots + k_t = k$, we have

$$\begin{aligned} |H_{k_1} \times \cdots \times H_{k_t}| &\leq \prod_{i=1}^{t} k_i \cdot 2^{k_i + O(\log(\frac{n'}{t}))} \\ &= 2^{t \cdot O(\log(\frac{n'}{t})) + \sum_{i=1}^{t} k_i} \cdot \prod_{i=1}^{t} k_i \end{aligned}$$

$$\leq 2^{t\cdot(O(\log \frac{n'}{t})+\log k)+k}$$
$$\leq 2^{O(t\cdot\log k)+k}.$$

Since each $0 \leq k_i \leq k$, the total number of partitions $k_1, \ldots, k_t$ is at most $(k+1)^t = 2^{t\log(k+1)} \leq 2^{O(t\cdot\log k)}$. Thus,

$$|H| \leq |\mathcal{M}| \cdot 2^{t\cdot O(\log k)} \cdot 2^{O(t\cdot\log k)+k}$$
$$\leq 2^{O(\log n)+O(t\cdot\log k)+k}.$$

Thus, the time used by our algorithm is at most

$$2^{O(\log n + t\cdot\log k + (k\log n)^2/t^2)+k}.$$

By choosing $t = O\left(\dfrac{k^2\log^2 n}{\log k}\right)^{1/3}$, we get the time used by our algorithm to be

$$2^{k+O\left((k\log n\log k)^{2/3}+\log n\right)}.$$

(Item 1). Now consider any $A \in \mathbb{F}_2^{k\times n}$ and any $b \in \text{image}(A)$. Without loss of generality, we assume that $\text{rank}(A) = k$. By Theorem 14.7.3, we know there is an $M \in \mathcal{M}$ such that $\text{rank}\, A = \text{rank}\, AM$, and we denote $A' := AM$. Since $\text{image}(AM) \subseteq \text{image}(A)$ and $\dim(\text{image}(AM)) = \text{rank}(AM) = \text{rank}(A) = \dim(\text{image}(A))$, we have $\text{image}(AM) = \text{image}(A)$, thus $b \in \text{image}(AM)$ as well. By partitioning A' into t blocks of size $k \times \frac{n'}{t}$, we get

$$A' = \left[A'_1\ A'_2\ \cdots\ A'_t\right].$$

Let $k_i = \text{rank}([A'_1\ A'_2\ \cdots\ A'_i]) - \text{rank}([A'_1\ A'_2\ \cdots\ A'_{i-1}])$. Thus, by Theorem 14.7.6, we know that there exists an $x \in H_{k_1} \times \cdots \times H_{k_t}$, such that $A'x = b$, which means $A(Mx) = b$

and $Mx \in H$. □

14.7.5 Hitting set for B_2-circuits

In this subsection, we will show that this hitting set can be used for B_2 circuits.

Corollary 14.7.8 (Restatement of Theorem 14.2.5)**.** *For every $n \in \mathbb{N}$ and $\alpha \in (0, 2.5)$, there exists a value $\varepsilon = 2^{-\Omega(\alpha^2 n)}$ and a set $H \subseteq \{0,1\}^n$ such that:*

1. *H is an ε-hitting set for B_2-circuits of size $(2.5 - \alpha) \cdot n$.*
2. *Given the parameters n and α, the set H can be enumerated in time $2^{(1-\Omega(\alpha^2))\cdot n + \widetilde{O}(n^{2/3})}$.*

Proof. Assume without loss of generality that the queries on every root-to-leaf path in f are linearly independent. First we note that any parity decision tree f can be written as $f = f_1 + \cdots + f_\ell$, where each f_i corresponds to a path from the root to an accepting leaf in f. Note that f_i's are disjoint and each f_i is a conjunction of parities function. The number of parity functions in the conjunction is the depth of this leaf. If f is a size-m parity decision tree with $\mathbb{E}[f] > \varepsilon$, then there is some accepting leaf that is reached with probability greater than ε/m. The depth of that leaf must be less than $\log(m/\varepsilon)$, because otherwise the probability of reaching it would be smaller. Consequently, if $H \subseteq \mathbb{F}_2^n$ hits codimension $\log(m/\varepsilon)$, then H is an ε-hitting set for size-m parity decision trees.

By Chen and Kabanets' work [CK16], we know every B_2-circuit of size $(2.5 - \alpha) \cdot n$ can be simulated by a parity decision tree of size $m = 2^{(1-c\cdot\alpha^2)\cdot n}$. Let $\varepsilon = 2^{-\frac{c}{2}\cdot\alpha^2 n}$. Then $\log(m/\varepsilon) = (1 - (c/2) \cdot \alpha^2)n$. By Theorem 14.7.7, a set that hits codimension $\log(m/\varepsilon)$ can be enumerated in time

$$2^{\log(m/\varepsilon)+O((\log(m/\varepsilon)\cdot\log\log(m/\varepsilon)\cdot\log n)^{2/3}+\log n)} = 2^{(1-\Omega(\alpha^2))\cdot n+\widetilde{O}(n^{2/3})}. \qquad \square$$

Remark 14.7.9. In this proof, we have used the fact hitting parity decision trees is equivalent to hitting system of equations. We further note that hitting DNF of parities is also equivalent

to these two problems.

14.7.6 Hitting sets that are more explicit

At this point, we have completed the proofs of Theorems 14.2.3 and 14.2.5. In this subsection, we prove a variant of Theorem 14.2.3 in which the hitting set is, in some sense, "more explicit," although its cardinality is slightly worse. Specifically, we show how to construct a hitting set generator $G : \{0,1\}^{k+o(n)} \to \mathbb{F}_2^n$ for systems of equations of $\mathbb{F}_2$, where given the seed x, we can output $G(x)$ in time $\text{poly}(n)$. The construction is similar to the construction in the proof of Theorem 14.2.3 but we skip the condensing step and choose t to be larger so we can afford the brute-force construction in Theorem 14.7.7. Note that most applications of hitting sets don't require this level of explicitness; however, we construct the following hitting set generator to match the level of explicitness of our PRGs.

Theorem 14.7.10 (More explicit hitting set for systems of equations over $\mathbb{F}_2$)**.** *For every $n, k \in \mathbb{N}$, there exists $G : \{0,1\}^s \to \mathbb{F}_2^n$, where $s = k + o(n)$, such that:*

1. *For every $A \in \mathbb{F}_2^{k \times n}$ and every $b \in \text{image}(A)$, there exists $x \in \{0,1\}^s$, such that $AG(x) = b$.*
2. *Given the parameters n and k and the seed x, $G(x)$ can be computed in time $\text{poly}(n)$.*

Proof Sketch. Let $t = \frac{n}{\sqrt{\log n}}$. Without loss of generality, we assume that n'/t is an integer. For each $i \in \{0, 1, \ldots, n/t\}$, we use Theorem 14.7.7 to construct $H_i \subseteq \mathbb{F}_2^{n/t}$ that hits codimension i, as defined in Theorem 14.7.4, where each H_i is on only n/t variables. Then we define $G_i : \{0,1\}^{d_i} \to \{0,1\}^{n/t}$ where $d_i = \lceil \log(|H_i|) \rceil$, and $G_i(y)$ output the y-th element in H_i. The overall construction can be seen as

$$G(k_1, \ldots, k_t, y_1, \ldots, y_t) = (G_{k_1}(y_1), \ldots, G_{k_t}(y_t)),$$

where $k_1 + \cdots + k_t = k$.

By Theorem 14.7.7, we know the size of each H_i is $|H_i| = O(\frac{n}{t} \cdot i2^i)$, and the total running time to construct these hitting sets over n/t variables is $\sum_{i=1}^{n/t} O(2^{n/t+i(n/t+2)}\frac{n}{t}i) \leq O(2^{n/t+(n/t)^2+2(n/t)}(\frac{n}{t})^3) = 2^{O((n/t)^2)} = n^{O(1)}$. Thus, to compute G, we first construct $\{G_i\}$ that hits codimension i, $i = 0, \ldots, n/t$, which takes time poly(n). Then for each $G_{k_i}(y_i)$, we just output the element indexed by y_i in H_{k_i}.

For each k_i, we denote its binary expansion as BIN(k_i). Let $|\cdot|$ denote the length of a string. Then, we can encode the input $(k_1, \ldots, k_t, y_1, \ldots, y_t)$ as

$$1^{|\mathrm{BIN}(k_1)|}0\mathrm{BIN}(k_1)\ldots 1^{|\mathrm{BIN}(k_t)|}0\mathrm{BIN}(k_t)y_1 \ \ldots \ y_t.$$

Note here we know the length for each y_i given k_i, since H_{k_i} has been constructed before.

By the construction above and the AM-GM inequality, the seed length required for $(k_1, \ldots, k_t)$ is at most

$$\begin{aligned}
\sum_{i=1}^{t}(2\log(k_i+1)+3) &= 3t + 2\log\left(\prod_{i=1}^{t}(k_i+1)\right) \\
&\leq 3t + 2\log\left((\frac{k+t}{t})^t\right) \\
&= 3t + 2t \cdot O\left(\log\left(\frac{n\sqrt{\log n}}{n}\right)\right) \\
&= O(t\log\log n) = o(n),
\end{aligned}$$

since when $k_i = 0$, we still need 3 bits to encode it.

The seed length required for $(y_1, \ldots, y_t)$ is at most

$$\begin{aligned}
\sum_{i=1}^{t}\left\lceil\log(|H_{k_i}|)\right\rceil &\leq t + \sum_{i=1}^{t}\log(|H_{k_i}|) \\
&\leq t + \sum_{i=1}^{t}\left(k_i + O\left(\log\left(\frac{n}{t}\cdot k_i\right)\right)\right) \\
&\leq t + k + t \cdot O(\log\log n) = k + o(n).
\end{aligned}$$

The proof of correctness is essentially same as the proof of Theorem 14.2.3. □

14.8 Limitations of k-wise γ-biased generators

In this section, we prove that our main results (Theorems 14.2.1, 14.2.2 and 14.2.4) cannot be proven by simply developing a better construction and/or analysis of k-wise γ-biased generators.

First, in Section 14.8.1, we present examples showing that if one wishes to use a generic k-wise γ-biased generator to get a universal set, or to hit near-maximal decision trees or B_2-circuits of size n or U_2-circuits of size $2n$, then one is forced to use a very large k and a very small γ. Then, in Section 14.8.2, we extend Karloff and Mansour's work [KM97] to show that when k is very large and γ is very small, every k-wise γ-biased generator has a very large seed length.

14.8.1 Counterexamples showing that k must be large and γ must be small

We begin by analyzing the parameter k. The argument is fairly trivial.

Proposition 14.8.1. *For every $n \in \mathbb{N}$ and $k \in [n-1]$, there exists a k-wise uniform distribution X over $\{0,1\}^n$ such that $\mathrm{Supp}(X)$ is not a 0.49-hitting set for $(k+1)$-juntas, or for B_2-circuits of size k, or for U_2-circuits of size $3k$.*

Proof. Let X be the uniform distribution over the set $\{x \in \{0,1\}^n : x_1 \oplus x_2 \oplus \cdots \oplus x_{k+1} = 0\}$. Let $f(x) = x_1 \oplus \cdots \oplus x_{k+1}$. Then f is a $(k+1)$-junta, and f can be computed by a B_2-circuit of size k, and f can be computed by a U_2-circuit of size $3k$ (Theorem 12.3.9). Furthermore, $\mathbb{E}[f] = 1/2$, whereas $\mathbb{E}[f(X)] = 0$. □

Now we move on to the bias parameter, γ. We begin by showing that a very small bias would be required to achieve k-universality.

Proposition 14.8.2. *For every $n \in \mathbb{N}$ and every $k \in [n]$, there exists a distribution X over $\{0,1\}^n$ such that X is n-wise $O(2^{-k})$-biased, but $\mathrm{Supp}(X)$ is not k-universal.*

Proof. Let X be the uniform distribution over the set $\{x \in \{0,1\}^n : (x_1, \ldots, x_k) \neq 0^k\}$. Clearly, $\mathrm{Supp}(X)$ is not k-universal. To show that X is n-wise $O(2^{-k})$-biased, let $S \subseteq [n]$ be any nonempty set of size at most k. If $S \not\subseteq [k]$, then $\mathbb{E}[\chi_S(X)] = 0$, because $(X_{k+1}, \ldots, X_n)$ is uniform over $\{0,1\}^{n-k}$ and independent of $(X_1, \ldots, X_k)$. If $S \subseteq [k]$, then

$$\begin{aligned} |\,\mathbb{E}[\chi_S(X)]| = \frac{1}{2^k - 1} \cdot \left| \sum_{x \in \{0,1\}^k \setminus \{0^k\}} \chi_S(x) \right| &= \frac{1}{2^k - 1} \cdot \left| \left(\sum_{x \in \{0,1\}^k} \chi_S(x) \right) - \chi_S(0^k) \right| \\ &= \frac{1}{2^k - 1} \\ &\leq \frac{2}{2^k}. \end{aligned}$$

□

Next, we show that a very small bias would be required to fool decision trees of depth $0.76 \cdot n$, or to hit B_2 circuits of size n or U_2-circuits of size $2n$. The proof is based on the "inner product mod 2" function. For each even positive integer n, we define $\mathsf{IP}_n \colon \{0,1\}^n \to \{0,1\}$ by the formula

$$\mathsf{IP}_n(x, y) = \bigoplus_{i=1}^{n/2} x_i y_i.$$

Proposition 14.8.3. *Let n be an even positive integer and let X be the uniform distribution over $\mathsf{IP}_n^{-1}(0)$. Then:*

1. *The distribution X is n-wise $(2^{-n/2})$-biased.*[2]
2. *The set $\mathrm{Supp}(X)$ is not a 0.49-hitting set for B_2-circuits of size $n-1$ or for U_2-circuits of size $2n-3$, assuming n is sufficiently large.*
3. *There is a value $k = \frac{3}{4} \cdot n + O(\sqrt{n})$ such that $\mathrm{Supp}(X)$ is not a 0.49-hitting set for*

2. For context, Bogdanov and Viola previously showed that X is n-wise $(2^{-\Omega(n)})$-biased, and they also showed a generalization of this statement to larger fields [BV10a].

depth-k decision trees, assuming n is sufficiently large.

Proof. Let $f(x,y) = (-1)^{\mathsf{IP}_n(x,y)}$. Let χ_S be any nontrivial character function. Sample $R \in \{0,1\}^n$ uniformly at random, and sample Y uniformly from $\mathsf{IP}^{-1}(1)$. For each $b \in \{0,1\}$, let $p_b = \Pr[\mathsf{IP}(R) = b]$. Note that $p_0 > 1/2$. Therefore,

$$\begin{aligned}
|\mathbb{E}[\chi_S(X)]| &< 2p_0 \cdot |\mathbb{E}[\chi_S(X)]| \\
&= |p_0 \cdot \mathbb{E}[\chi_S(X)] + p_1 \cdot \mathbb{E}[\chi_S(Y)] + p_0 \cdot \mathbb{E}[\chi_S(X)] - p_1 \cdot \mathbb{E}[\chi_S(Y)]| \\
&= |\mathbb{E}[\chi_S(R)] + \mathbb{E}[\chi_S(R) \cdot f(R)]| \\
&= |\widehat{f}(S)|.
\end{aligned}$$

(The second-to-last equation is an application of the law of total expectation.) It follows that X is n-wise $(2^{-n/2})$-biased, because the inner product mod 2 function is famously "bent," meaning that $|\widehat{f}(S)| = 2^{-n/2}$ for every S. For completeness, we include the calculation showing that $|\widehat{f}(S)| = 2^{-n/2}$ below:

$$\begin{aligned}
\widehat{f}(S) &= \mathop{\mathbb{E}}_{x,y}[f(x,y) \cdot \chi_S(x,y)] \\
&= \mathop{\mathbb{E}}_{x,y}\left[\left(\prod_{i=1}^{n/2}(-1)^{x_i y_i}\right) \cdot \left(\prod_{i=1}^{n/2}(-1)^{x_i u_i}\right) \cdot \left(\prod_{i=1}^{n/2}(-1)^{y_i v_i}\right)\right] \quad \text{for some } u, v \in \{0,1\}^{n/2} \\
&= \prod_{i=1}^{n/2} \mathop{\mathbb{E}}_{a,b \in \{0,1\}}[(-1)^{ab + a u_i + b v_i}] \\
&= \prod_{i=1}^{n/2} \frac{1 + (-1)^{v_i} + (-1)^{u_i} + (-1)^{1+u_i+v_i}}{4} \\
&= \prod_{i=1}^{n/2} \left(\pm \frac{1}{2}\right) \\
&= \pm 2^{-n/2}.
\end{aligned}$$

The set $\mathrm{Supp}(X)$ is not a 0.49-hitting set for B_2-circuits of size $n-1$, because $\mathbb{E}[\mathsf{IP}_n] =$

$1/2 - o(1)$, and IP_n can be computed by a B_2-circuit of size $n-1$. Similarly, $\mathrm{Supp}(X)$ is not a 0.49-hitting set for U_2-circuits of size $2n-3$, because IP_n can be computed by a U_2-circuit of size $2n-3$:

- We use $n/2$ "AND" gates to compute the bits $x_1y_1, \ldots, x_{n/2}y_{n/2}$.
- Then we use $3(n/2) - 3$ gates to compute the parity of those $n/2$ bits (Theorem 12.3.9).

Finally, we will show that $\mathrm{Supp}(X)$ is not a 0.49-hitting set for decision trees of depth $\frac{3}{4} \cdot n + O(\sqrt{n})$. Define

$$T(x,y) = \begin{cases} \mathsf{IP}(x,y) & \text{if } |x| \leq n/4 + 2\sqrt{n} \\ 0 & \text{if } |x| > n/4 + 2\sqrt{n}, \end{cases}$$

where $|x|$ denotes the Hamming weight of x and c is an appropriate constant. Then $T(x,y)$ can be computed by a decision tree of depth $\frac{3}{4} \cdot n + O(\sqrt{n})$, and $T \leq \mathsf{IP}_n$, so $\mathbb{E}[T(X)] = 0$. On the other hand, if we pick x and y uniformly at random:

- There is a $2^{-n/2}$ chance that $x = 0^{n/2}$.
- There is at most an $\exp(-16)$ chance that x has Hamming weight more than $n/4 + 2\sqrt{n}$, by Hoeffding's inequality.
- For any fixing of x such that neither of the two events above occur, we have $\mathbb{E}_y[T(x,y)] = 1/2$.

Therefore, $\mathbb{E}[T] \geq \frac{1}{2} - 2^{-n/2} - \exp(-16) \geq 0.49$. □

14.8.2 Seed length lower bound for k-wise γ-biased generators

In this section, we prove our seed length lower bound for k-wise γ-biased generators (Theorem 14.2.6). The proof is a straightforward extension of Karloff and Mansour's argument [KM97], which covers the case $\gamma = 0$. The approach is to bound the *collision probability*

of a k-wise γ-biased distribution.

Definition 14.8.4 (Collision probability)**.** Let X be a probability distribution over the space $\mathcal{X}$. The *collision probability* $\mathsf{CP}(X)$ is defined by

$$\mathsf{CP}(X) = \Pr_{\substack{x \sim X \\ x' \sim X}}[x = x'],$$

where x and x' are sampled independently from X. Equivalently, $\mathsf{CP}(X) = \sum_{x \in \mathcal{X}} \Pr[X = x]^2$.

Theorem 14.8.5 (Collision probability of k-wise γ-biased distributions)**.** *Let $n \in \mathbb{N}$, let $\gamma \in (0,1)$, let $\alpha \in (0, 1/2]$, let $k = \lfloor (\frac{1}{2} + \alpha) \cdot n \rfloor$, and let X be a distribution that is k-wise γ-biased. Then*

$$\mathsf{CP}(X) \leq \left(1 + \frac{1}{2\alpha}\right) \cdot \left(2^{-n} + \gamma^2\right).$$

Proof. Let $p \colon \{0,1\}^n \to [0,1]$ be the probability mass function of X, i.e., $p(x) = \Pr[X = x]$. Since X is a probability distribution, we have $\widehat{p}(\varnothing) = 2^{-n}$. Furthermore, since X is k-wise γ-biased, we have $|\widehat{p}(S)| \leq \gamma \cdot 2^{-n}$ whenever $1 \leq |S| \leq k$. Therefore, we can bound the collision probability of X as follows.

$$\begin{aligned}
\mathsf{CP}(X) = \sum_{x \in \{0,1\}^n} p(x)^2 &= 2^n \cdot \mathop{\mathbb{E}}_{x \in \{0,1\}^n}[p(x)^2] \\
&= 2^n \cdot \sum_{S \subseteq [n]} \widehat{p}(S)^2 && \text{(Parseval's theorem)} \\
&\leq 2^n \cdot \left(\frac{1}{2^{2n}} + \binom{n}{\leq k} \cdot \frac{\gamma^2}{2^{2n}} + \sum_{\substack{S \subseteq [n] \\ |S| > k}} \widehat{p}(S)^2 \right) \\
&\leq 2^{-n} + \gamma^2 + 2^n \cdot \sum_{\substack{S \subseteq [n] \\ |S| > k}} \widehat{p}(S)^2.
\end{aligned}$$

To bound the high-degree Fourier weight, let $x^{\oplus i}$ denote x with the i-th bit flipped. Identify

a set $T \subseteq [n]$ with its indicator function $T\colon [n] \to \{0,1\}$. Then

$$\begin{aligned}
0 \leq \sum_{i=1}^{n} \sum_{x\in\{0,1\}^n} p(x)\cdot p(x^{\oplus i}) &= \sum_{S\subseteq[n]} \sum_{T\subseteq[n]} \widehat{p}(S)\cdot\widehat{p}(T)\cdot \sum_{i=1}^{n} \sum_{x\in\{0,1\}^n} \chi_S(x)\cdot\chi_T(x^{\oplus i}) \\
&= \sum_{S\subseteq[n]} \sum_{T\subseteq[n]} \widehat{p}(S)\cdot\widehat{p}(T)\cdot \sum_{i=1}^{n} (-1)^{T(i)} \cdot \sum_{x\in\{0,1\}^n} \chi_S(x)\cdot\chi_T(x) \\
&= 2^n \cdot \sum_{S\subseteq[n]} \widehat{p}(S)^2 \cdot \sum_{i=1}^{n} (-1)^{S(i)} \\
&= 2^n \cdot \sum_{d=0}^{n} \sum_{|S|=d} \widehat{p}(S)^2 \cdot (n-2d) \\
&\leq 2^n \cdot \left(\left(n \cdot \sum_{\substack{S\subseteq[n] \\ |S|\leq k}} \widehat{p}(S)^2 \right) - (2k+2-n) \cdot \sum_{\substack{S\subseteq[n] \\ |S|>k}} \widehat{p}(S)^2 \right) \\
&\leq n\cdot(2^{-n}+\gamma^2) - 2^n\cdot(2k+2-n)\cdot \sum_{d=k+1}^{n} \sum_{|S|=d} \widehat{p}(S)^2.
\end{aligned}$$

Consequently,

$$\begin{aligned}
\mathsf{CP}(X) &\leq 2^{-n} + \gamma^2 + \frac{n\cdot(2^{-n}+\gamma^2)}{2k+2-n} = \frac{2k+2}{2k+2-n}\cdot(2^{-n}+\gamma^2) \\
&= \left(1 + \frac{n}{2k+2-n}\right)\cdot(2^{-n}+\gamma^2) \\
&\leq \left(1 + \frac{n}{(1+2\alpha)n-n}\right)\cdot(2^{-n}+\gamma^2) \\
&= \left(1 + \frac{1}{2\alpha}\right)\cdot(2^{-n}+\gamma^2).
\end{aligned}$$

□

Proof of Theorem 14.2.6. The output of G has collision probability at least 2^{-s}, since this is the chance of getting the same seed twice in a row. Therefore,

$$2^{-s} \leq \left(1 + \frac{1}{2\alpha}\right)\cdot(2^{-n}+\gamma^2) \leq \frac{2}{\alpha}\cdot\max\{2^{-n},\gamma^2\},$$

and consequently

$$s \geq \min\{n, 2\log(1/\gamma)\} - \log(2/\alpha). \qquad \square$$

By combining the results of this subsection with the counterexamples from the previous subsection, we get the following conclusion.

Corollary 14.8.6. *Let $n, k \in \mathbb{N}$ and $\gamma \in (0,1)$. Suppose that at least one of the following holds.*

1. *The support of every k-wise γ-biased distribution over $\{0,1\}^n$ is $(0.51 \cdot n)$-universal.*
2. *The support of every k-wise γ-biased distribution over $\{0,1\}^n$ is a 0.49-hitting set for decision trees of depth $0.76 \cdot n$, or for B_2-circuits of size n, or for U_2-circuits of size $2n$.*

Then every k-wise γ-biased generator has seed length $n - O(1)$.

Proof. First, we show that $k \geq \frac{1}{2} + \Omega(1)$. Case (1) implies that every k-wise uniform distribution is a 0-hitting set for $(0.51 \cdot n)$-juntas, hence $k \geq \lfloor 0.51 \cdot n \rfloor$ by Theorem 14.8.1. Similarly, case (2) implies that every k-wise uniform distribution is a 0.49-hitting set for $(0.76 \cdot n)$-juntas, or for B_2-circuits of size n, or for U_2-circuits of size $2n$. By Theorem 14.8.1, these three possibilities would imply $k \geq \lfloor 0.76n \rfloor$, $k \geq n$, and $k \geq \lfloor 2n/3 \rfloor$ respectively.

Next, we show that $\gamma \leq O(2^{-n/2})$. In case (1), this follows immediately from Theorem 14.8.2. Now suppose we are in case (2). Let Z be the distribution over $\{0,1\}^{n'}$ from Theorem 14.8.3, where $n' \in \{n, n-1\}$ and n' is even. By appending a uniform random bit to Z if necessary, we get a distribution Z' over $\{0,1\}^n$ such that (a) Z' is n-wise $(2^{-(n-1)/2})$-biased, but (b) $\mathrm{Supp}(Z')$ is not a 0.49-hitting set for B_2-circuits of size n, or for B_2-circuits of size $2n$, or for decision trees of depth $0.76n$. Therefore, $\gamma < 2^{-(n-1)/2}$.

Finally, because the parameters k and γ have such extreme values, Theorem 14.2.6 tells us that every k-wise γ-biased generator has seed length at least $\min\{n, 2\log(1/\gamma)\} - O(1) = n - O(1)$. $\square$

14.9 Open problems

- Find more applications of k-wise probably uniform generators.
- Improve the seed lengths of our constructions.
- Design an explicit PRG, with a seed length similar to that of our k-wise probably uniform generator, that samples a distribution X such that

$$(1-\varepsilon)\cdot\mathbb{E}[f] \leq \mathbb{E}[f(X)] \leq (1+\varepsilon)\cdot\mathbb{E}[f]$$

  for every k-junta f. This is equivalent to saying that every k coordinates of X are uniform to within ℓ_∞ error $\varepsilon \cdot 2^{-k}$. Such a PRG could be used to fool near-maximal unambiguous DNF formulas.
- Design an explicit PRG (not just a hitting set) that fools near-maximal parity decision trees and B_2-circuits of size $2.49 \cdot n$ with seed length $(1-\Omega(1)) \cdot n$.
- Improve the seed length in Theorem 14.5.5 (the balanced partition generator) to $O(\log(k/\delta) + \log\log n)$. This would not have any effect on our main theorems, but it is a natural problem in its own right.
- Prove tight bounds on the optimal nonexplicit seed length of PRGs fooling depth-k decision trees with error ε when k and $\log(1/\varepsilon)$ are both large. For example, does there exist a PRG that fools decision trees of depth $k = 0.9 \cdot n$ with error $\varepsilon = 2^{-0.4n}$ and seed length $(1-\Omega(1)) \cdot n$?
- Prove matching upper and lower bounds on the power of small-bias distributions to fool decision trees. For example, does there exist a constant $c < 1/2$ such that every n-wise (2^{-cn})-biased distribution fools decision trees of depth $n/2$ with error 0.1?

CHAPTER 15

CONCLUSION

The two technical chapters of this dissertation develop different pseudorandom objects for different restricted models, but they share a methodological lesson. Generic bounded independence and generic expander-based recursion remain indispensable tools, yet the strongest results here arise only after tailoring those tools to the exact structure of the model being fooled. In the ROBP setting, this leads to a careful analysis of XORs of INW generators and to matching limitations. In the decision-tree setting, it leads to the asymmetric notion of probable uniformity and to consequences that are out of reach for ordinary small-bias distributions.

Several natural directions remain. On the space-bounded side, the most compelling next step is to understand whether the positive XOR-of-INW theorem can itself be derandomized, for example by correlating the seeds rather than choosing them independently. On the adaptive side, it would be very interesting to sharpen the seed length for probable uniformity further, to improve the dependence on lower-order terms, and to extend the techniques to broader circuit classes. More broadly, these results suggest that there is still substantial room to discover new intermediate notions of pseudorandomness between generic bounded independence and fully model-specific PRGs.

APPENDIX A

MARK-PATTERN TABLES

Here we record the currently selected mark-pattern tables with the symmetry multiplicity inherited by each partial shell generated from them. Rows correspond to the six determinant copies, while columns distinguish marked-table columns. A zero denotes an unmarked position. Equal letters denote marks covering the same element, and different letters denote different covered elements.

For a table P, let $\mathrm{Aut}(P)$ denote its automorphism group. If $\widetilde{S}$ is a literal partial shell generated from P, let $n_{\underline{3}}(\widetilde{S})$ be its number of separate $\underline{3}$-columns. This number is determined by P and is constant across its generated descendants in the enumeration convention used here. The multiplicity recorded beside P is therefore

$$C_{\mathrm{NB}}(P) = \mathrm{SymFac}(P, \widetilde{S}) = \frac{6!}{|\,\mathrm{Aut}(P)|\, n_{\underline{3}}(\widetilde{S})!}. \tag{A.1}$$

By contrast, $n_{\underline{4}}(\widetilde{S})$, the number of separate $\underline{4}$-columns, is not completely determined by P and contributes no factorial to this multiplicity. The implementation emits those columns in a prescribed covered-element order, so their permutations are never generated as distinct records. Every table below displays the exact value of $\mathrm{SymFac}(P, \widetilde{S}) = C_{\mathrm{NB}}(P)$.

A.1 One-mark tables

Table A.1: Mark-pattern tables with $r = 1$ mark.

$\mathcal{M}_1^{(1)} = \begin{bmatrix} a \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$	
$C_{\mathrm{NB}}(P)$=6	

A.2 Two-mark tables

Table A.2: Mark-pattern tables with $r = 2$ marks.

$\mathcal{M}_1^{(2)} = \begin{bmatrix} a \\ a \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$	$\mathcal{M}_2^{(2)} = \begin{bmatrix} a \\ b \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$	$\mathcal{M}_3^{(2)} = \begin{bmatrix} a & 0 \\ 0 & b \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \end{bmatrix}$
$C_{\mathrm{NB}}(P)$=15	$C_{\mathrm{NB}}(P)$=15	$C_{\mathrm{NB}}(P)$=15

A.3 Three-mark tables

Table A.3: Mark-pattern tables with $r = 3$ marks.

$\mathcal{M}_1^{(3)} = \begin{bmatrix} a \\ a \\ a \\ 0 \\ 0 \\ 0 \end{bmatrix}$	$\mathcal{M}_2^{(3)} = \begin{bmatrix} a \\ a \\ b \\ 0 \\ 0 \\ 0 \end{bmatrix}$	$\mathcal{M}_3^{(3)} = \begin{bmatrix} a \\ b \\ c \\ 0 \\ 0 \\ 0 \end{bmatrix}$
$C_{\mathrm{NB}}(P)$=20	$C_{\mathrm{NB}}(P)$=60	$C_{\mathrm{NB}}(P)$=20
$\mathcal{M}_4^{(3)} = \begin{bmatrix} a & 0 \\ a & 0 \\ 0 & a \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \end{bmatrix}$	$\mathcal{M}_5^{(3)} = \begin{bmatrix} a & 0 \\ a & 0 \\ 0 & b \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \end{bmatrix}$	$\mathcal{M}_6^{(3)} = \begin{bmatrix} a & 0 \\ b & 0 \\ 0 & a \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \end{bmatrix}$
$C_{\mathrm{NB}}(P)$=60	$C_{\mathrm{NB}}(P)$=60	$C_{\mathrm{NB}}(P)$=120
$\mathcal{M}_7^{(3)} = \begin{bmatrix} a & 0 \\ b & 0 \\ 0 & c \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \end{bmatrix}$	$\mathcal{M}_8^{(3)} = \begin{bmatrix} a & 0 & 0 \\ 0 & b & 0 \\ 0 & 0 & c \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$	
$C_{\mathrm{NB}}(P)$=60	$C_{\mathrm{NB}}(P)$=20	

A.4 Four-mark tables

Table A.4: Mark-pattern tables with $r = 4$ marks.

$\mathcal{M}_1^{(4)} = \begin{bmatrix} a \\ a \\ a \\ a \\ 0 \\ 0 \end{bmatrix}$	$\mathcal{M}_2^{(4)} = \begin{bmatrix} a \\ a \\ a \\ b \\ 0 \\ 0 \end{bmatrix}$	$\mathcal{M}_3^{(4)} = \begin{bmatrix} a \\ a \\ b \\ c \\ 0 \\ 0 \end{bmatrix}$
$C_{\mathrm{NB}}(P)$=15	$C_{\mathrm{NB}}(P)$=60	$C_{\mathrm{NB}}(P)$=90
$\mathcal{M}_4^{(4)} = \begin{bmatrix} a \\ b \\ c \\ d \\ 0 \\ 0 \end{bmatrix}$	$\mathcal{M}_5^{(4)} = \begin{bmatrix} a & 0 \\ a & 0 \\ 0 & a \\ 0 & a \\ 0 & 0 \\ 0 & 0 \end{bmatrix}$	$\mathcal{M}_6^{(4)} = \begin{bmatrix} a & 0 \\ a & 0 \\ 0 & b \\ 0 & b \\ 0 & 0 \\ 0 & 0 \end{bmatrix}$
$C_{\mathrm{NB}}(P)$=$\frac{15}{2}$	$C_{\mathrm{NB}}(P)$=45	$C_{\mathrm{NB}}(P)$=45
$\mathcal{M}_7^{(4)} = \begin{bmatrix} a & 0 \\ b & 0 \\ 0 & a \\ 0 & b \\ 0 & 0 \\ 0 & 0 \end{bmatrix}$	$\mathcal{M}_8^{(4)} = \begin{bmatrix} a & 0 \\ a & 0 \\ 0 & b \\ 0 & c \\ 0 & 0 \\ 0 & 0 \end{bmatrix}$	$\mathcal{M}_9^{(4)} = \begin{bmatrix} a & 0 \\ b & 0 \\ 0 & a \\ 0 & c \\ 0 & 0 \\ 0 & 0 \end{bmatrix}$
$C_{\mathrm{NB}}(P)$=90	$C_{\mathrm{NB}}(P)$=90	$C_{\mathrm{NB}}(P)$=180
$\mathcal{M}_{10}^{(4)} = \begin{bmatrix} a & 0 \\ a & 0 \\ b & 0 \\ 0 & c \\ 0 & 0 \\ 0 & 0 \end{bmatrix}$	$\mathcal{M}_{11}^{(4)} = \begin{bmatrix} a & 0 \\ b & 0 \\ c & 0 \\ 0 & a \\ 0 & 0 \\ 0 & 0 \end{bmatrix}$	$\mathcal{M}_{12}^{(4)} = \begin{bmatrix} a & 0 \\ b & 0 \\ c & 0 \\ 0 & d \\ 0 & 0 \\ 0 & 0 \end{bmatrix}$
$C_{\mathrm{NB}}(P)$=180	$C_{\mathrm{NB}}(P)$=180	$C_{\mathrm{NB}}(P)$=60
$\mathcal{M}_{13}^{(4)} = \begin{bmatrix} a & 0 \\ b & 0 \\ 0 & c \\ 0 & d \\ 0 & 0 \\ 0 & 0 \end{bmatrix}$	$\mathcal{M}_{14}^{(4)} = \begin{bmatrix} a & 0 & 0 \\ a & 0 & 0 \\ 0 & b & 0 \\ 0 & 0 & c \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$	$\mathcal{M}_{15}^{(4)} = \begin{bmatrix} a & 0 & 0 \\ b & 0 & 0 \\ 0 & c & 0 \\ 0 & 0 & d \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$
$C_{\mathrm{NB}}(P)$=$\frac{45}{2}$	$C_{\mathrm{NB}}(P)$=90	$C_{\mathrm{NB}}(P)$=90
$\mathcal{M}_{16}^{(4)} = \begin{bmatrix} a & 0 & 0 \\ b & 0 & 0 \\ 0 & a & 0 \\ 0 & 0 & c \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$	$\mathcal{M}_{17}^{(4)} = \begin{bmatrix} a & 0 & 0 \\ 0 & a & 0 \\ 0 & 0 & b \\ 0 & 0 & c \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$	$\mathcal{M}_{18}^{(4)} = \begin{bmatrix} a & 0 & 0 \\ a & 0 & 0 \\ 0 & a & 0 \\ 0 & 0 & b \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$
$C_{\mathrm{NB}}(P)$=360	$C_{\mathrm{NB}}(P)$=90	$C_{\mathrm{NB}}(P)$=180

$\mathcal{M}_{19}^{(4)} = \begin{bmatrix} a & 0 & 0 \\ b & 0 & 0 \\ 0 & a & 0 \\ 0 & 0 & a \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$	$\mathcal{M}_{20}^{(4)} = \begin{bmatrix} a & 0 & 0 & 0 \\ 0 & b & 0 & 0 \\ 0 & 0 & c & 0 \\ 0 & 0 & 0 & d \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$	$\mathcal{M}_{21}^{(4)} = \begin{bmatrix} a & 0 \\ a & 0 \\ a & 0 \\ 0 & b \\ 0 & 0 \\ 0 & 0 \end{bmatrix}$
$C_{\mathrm{NB}}(P)$=180	$C_{\mathrm{NB}}(P)$=15	$C_{\mathrm{NB}}(P)$=60
$\mathcal{M}_{22}^{(4)} = \begin{bmatrix} a & 0 \\ a & 0 \\ b & 0 \\ 0 & a \\ 0 & 0 \\ 0 & 0 \end{bmatrix}$	$\mathcal{M}_{23}^{(4)} = \begin{bmatrix} a \\ a \\ b \\ b \\ 0 \\ 0 \end{bmatrix}$	$\mathcal{M}_{24}^{(4)} = \begin{bmatrix} a & 0 \\ a & 0 \\ 0 & a \\ 0 & b \\ 0 & 0 \\ 0 & 0 \end{bmatrix}$
$C_{\mathrm{NB}}(P)$=180	$C_{\mathrm{NB}}(P)$=45	$C_{\mathrm{NB}}(P)$=180

A.5 Five-mark tables

Table A.5: Mark-pattern tables with $r = 5$ marks.

$\mathcal{M}_{1}^{(5)} = \begin{bmatrix} a & 0 \\ a & 0 \\ a & 0 \\ a & 0 \\ 0 & b \\ 0 & 0 \end{bmatrix}$	$\mathcal{M}_{2}^{(5)} = \begin{bmatrix} a & 0 \\ a & 0 \\ 0 & a \\ 0 & a \\ 0 & b \\ 0 & 0 \end{bmatrix}$	$\mathcal{M}_{3}^{(5)} = \begin{bmatrix} a & 0 & 0 \\ a & 0 & 0 \\ 0 & a & 0 \\ 0 & a & 0 \\ 0 & 0 & b \\ 0 & 0 & 0 \end{bmatrix}$
$C_{\mathrm{NB}}(P)$=30	$C_{\mathrm{NB}}(P)$=180	$C_{\mathrm{NB}}(P)$=90
$\mathcal{M}_{4}^{(5)} = \begin{bmatrix} a & 0 \\ a & 0 \\ a & 0 \\ b & 0 \\ 0 & a \\ 0 & 0 \end{bmatrix}$	$\mathcal{M}_{5}^{(5)} = \begin{bmatrix} a & 0 & 0 \\ a & 0 & 0 \\ 0 & a & 0 \\ 0 & b & 0 \\ 0 & 0 & a \\ 0 & 0 & 0 \end{bmatrix}$	$\mathcal{M}_{6}^{(5)} = \begin{bmatrix} a & 0 \\ a & 0 \\ a & 0 \\ 0 & b \\ 0 & b \\ 0 & 0 \end{bmatrix}$
$C_{\mathrm{NB}}(P)$=120	$C_{\mathrm{NB}}(P)$=360	$C_{\mathrm{NB}}(P)$=60
$\mathcal{M}_{7}^{(5)} = \begin{bmatrix} a & 0 & 0 \\ 0 & a & 0 \\ 0 & a & 0 \\ 0 & 0 & b \\ 0 & 0 & b \\ 0 & 0 & 0 \end{bmatrix}$	$\mathcal{M}_{8}^{(5)} = \begin{bmatrix} a & 0 & 0 \\ b & 0 & 0 \\ 0 & a & 0 \\ 0 & b & 0 \\ 0 & 0 & a \\ 0 & 0 & 0 \end{bmatrix}$	$\mathcal{M}_{9}^{(5)} = \begin{bmatrix} a & 0 \\ a & 0 \\ b & 0 \\ b & 0 \\ 0 & a \\ 0 & 0 \end{bmatrix}$
$C_{\mathrm{NB}}(P)$=180	$C_{\mathrm{NB}}(P)$=360	$C_{\mathrm{NB}}(P)$=180
$\mathcal{M}_{10}^{(5)} = \begin{bmatrix} a & 0 \\ a & 0 \\ a & 0 \\ b & 0 \\ 0 & c \\ 0 & 0 \end{bmatrix}$	$\mathcal{M}_{11}^{(5)} = \begin{bmatrix} a & 0 & 0 \\ a & 0 & 0 \\ a & 0 & 0 \\ 0 & b & 0 \\ 0 & 0 & c \\ 0 & 0 & 0 \end{bmatrix}$	$\mathcal{M}_{12}^{(5)} = \begin{bmatrix} a & 0 & 0 \\ a & 0 & 0 \\ 0 & a & 0 \\ 0 & b & 0 \\ 0 & 0 & c \\ 0 & 0 & 0 \end{bmatrix}$
$C_{\mathrm{NB}}(P)$=120	$C_{\mathrm{NB}}(P)$=60	$C_{\mathrm{NB}}(P)$=360

$$\mathcal{M}_{13}^{(5)} = \begin{bmatrix} a & 0 & 0 & 0 \\ 0 & a & 0 & 0 \\ 0 & a & 0 & 0 \\ 0 & 0 & b & 0 \\ 0 & 0 & 0 & c \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

$C_{\mathrm{NB}}(P)$=180

$$\mathcal{M}_{14}^{(5)} = \begin{bmatrix} a & 0 & 0 \\ 0 & a & 0 \\ 0 & a & 0 \\ 0 & b & 0 \\ 0 & 0 & c \\ 0 & 0 & 0 \end{bmatrix}$$

$C_{\mathrm{NB}}(P)$=360

$$\mathcal{M}_{15}^{(5)} = \begin{bmatrix} a & 0 & 0 & 0 \\ 0 & a & 0 & 0 \\ 0 & 0 & a & 0 \\ 0 & 0 & b & 0 \\ 0 & 0 & 0 & c \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

$C_{\mathrm{NB}}(P)$=360

$$\mathcal{M}_{16}^{(5)} = \begin{bmatrix} a & 0 \\ a & 0 \\ a & 0 \\ 0 & b \\ 0 & c \\ 0 & 0 \end{bmatrix}$$

$C_{\mathrm{NB}}(P)$=60

$$\mathcal{M}_{17}^{(5)} = \begin{bmatrix} a & 0 \\ a & 0 \\ b & 0 \\ 0 & a \\ 0 & c \\ 0 & 0 \end{bmatrix}$$

$C_{\mathrm{NB}}(P)$=360

$$\mathcal{M}_{18}^{(5)} = \begin{bmatrix} a & 0 \\ a & 0 \\ b & 0 \\ c & 0 \\ 0 & b \\ 0 & 0 \end{bmatrix}$$

$C_{\mathrm{NB}}(P)$=360

$$\mathcal{M}_{19}^{(5)} = \begin{bmatrix} a & 0 \\ a & 0 \\ 0 & a \\ 0 & b \\ 0 & c \\ 0 & 0 \end{bmatrix}$$

$C_{\mathrm{NB}}(P)$=180

$$\mathcal{M}_{20}^{(5)} = \begin{bmatrix} a & 0 \\ 0 & a \\ 0 & a \\ 0 & b \\ 0 & c \\ 0 & 0 \end{bmatrix}$$

$C_{\mathrm{NB}}(P)$=360

$$\mathcal{M}_{21}^{(5)} = \begin{bmatrix} a & 0 & 0 \\ 0 & a & 0 \\ 0 & a & 0 \\ 0 & 0 & b \\ 0 & 0 & c \\ 0 & 0 & 0 \end{bmatrix}$$

$C_{\mathrm{NB}}(P)$=360

$$\mathcal{M}_{22}^{(5)} = \begin{bmatrix} a & 0 & 0 & 0 \\ 0 & a & 0 & 0 \\ 0 & 0 & a & 0 \\ 0 & 0 & 0 & b \\ 0 & 0 & 0 & c \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

$C_{\mathrm{NB}}(P)$=180

$$\mathcal{M}_{23}^{(5)} = \begin{bmatrix} a & 0 \\ a & 0 \\ b & 0 \\ b & 0 \\ 0 & c \\ 0 & 0 \end{bmatrix}$$

$C_{\mathrm{NB}}(P)$=180

$$\mathcal{M}_{24}^{(5)} = \begin{bmatrix} a & 0 & 0 \\ a & 0 & 0 \\ 0 & b & 0 \\ 0 & b & 0 \\ 0 & 0 & c \\ 0 & 0 & 0 \end{bmatrix}$$

$C_{\mathrm{NB}}(P)$=180

$$\mathcal{M}_{25}^{(5)} = \begin{bmatrix} a & 0 & 0 \\ b & 0 & 0 \\ 0 & a & 0 \\ 0 & b & 0 \\ 0 & 0 & c \\ 0 & 0 & 0 \end{bmatrix}$$

$C_{\mathrm{NB}}(P)$=60

$$\mathcal{M}_{26}^{(5)} = \begin{bmatrix} a & 0 & 0 \\ 0 & a & 0 \\ 0 & 0 & a \\ 0 & 0 & b \\ 0 & 0 & c \\ 0 & 0 & 0 \end{bmatrix}$$

$C_{\mathrm{NB}}(P)$=90

$$\mathcal{M}_{27}^{(5)} = \begin{bmatrix} a & 0 & 0 \\ 0 & a & 0 \\ 0 & b & 0 \\ 0 & 0 & a \\ 0 & 0 & c \\ 0 & 0 & 0 \end{bmatrix}$$

$C_{\mathrm{NB}}(P)$=90

$$\mathcal{M}_{28}^{(5)} = \begin{bmatrix} a & 0 \\ a & 0 \\ c & 0 \\ 0 & b \\ 0 & b \\ 0 & 0 \end{bmatrix}$$

$C_{\mathrm{NB}}(P)$=180

$$\mathcal{M}_{29}^{(5)} = \begin{bmatrix} a & 0 \\ b & 0 \\ c & 0 \\ 0 & a \\ 0 & b \\ 0 & 0 \end{bmatrix}$$

$C_{\mathrm{NB}}(P)$=180

$$\mathcal{M}_{30}^{(5)} = \begin{bmatrix} a & 0 \\ a & 0 \\ b & 0 \\ c & 0 \\ 0 & d \\ 0 & 0 \end{bmatrix}$$

$C_{\mathrm{NB}}(P)$=360

$$\mathcal{M}_{31}^{(5)} = \begin{bmatrix} a & 0 \\ b & 0 \\ c & 0 \\ d & 0 \\ 0 & a \\ 0 & 0 \end{bmatrix}$$

$C_{\mathrm{NB}}(P)$=180

$$\mathcal{M}_{32}^{(5)} = \begin{bmatrix} a & 0 \\ b & 0 \\ c & 0 \\ 0 & a \\ 0 & d \\ 0 & 0 \end{bmatrix}$$

$C_{\mathrm{NB}}(P)$=360

$$\mathcal{M}_{33}^{(5)} = \begin{bmatrix} a & 0 & 0 \\ a & 0 & 0 \\ 0 & b & 0 \\ 0 & c & 0 \\ 0 & 0 & d \\ 0 & 0 & 0 \end{bmatrix}$$

$C_{\mathrm{NB}}(P)$=180

$C_{\mathrm{NB}}(P)$=120

$$\mathcal{M}_{34}^{(5)} = \begin{bmatrix} a & 0 & 0 \\ b & 0 & 0 \\ 0 & a & 0 \\ 0 & c & 0 \\ 0 & 0 & d \\ 0 & 0 & 0 \end{bmatrix}$$

$C_{\mathrm{NB}}(P)$=360

$$\mathcal{M}_{35}^{(5)} = \begin{bmatrix} a & 0 & 0 \\ a & 0 & 0 \\ b & 0 & 0 \\ 0 & c & 0 \\ 0 & 0 & d \\ 0 & 0 & 0 \end{bmatrix}$$

$C_{\mathrm{NB}}(P)$=180

$$\mathcal{M}_{36}^{(5)} = \begin{bmatrix} a & 0 & 0 \\ b & 0 & 0 \\ c & 0 & 0 \\ 0 & a & 0 \\ 0 & 0 & d \\ 0 & 0 & 0 \end{bmatrix}$$

$C_{\mathrm{NB}}(P)$=360

$$\mathcal{M}_{37}^{(5)} = \begin{bmatrix} a & 0 & 0 & 0 \\ a & 0 & 0 & 0 \\ 0 & b & 0 & 0 \\ 0 & 0 & c & 0 \\ 0 & 0 & 0 & d \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

$C_{\mathrm{NB}}(P)$=60

$$\mathcal{M}_{38}^{(5)} = \begin{bmatrix} a & 0 & 0 & 0 \\ b & 0 & 0 & 0 \\ 0 & a & 0 & 0 \\ 0 & 0 & c & 0 \\ 0 & 0 & 0 & d \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

$C_{\mathrm{NB}}(P)$=360

$$\mathcal{M}_{39}^{(5)} = \begin{bmatrix} a & 0 & 0 & 0 \\ 0 & a & 0 & 0 \\ 0 & 0 & b & 0 \\ 0 & 0 & c & 0 \\ 0 & 0 & 0 & d \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

$C_{\mathrm{NB}}(P)$=60

$$\mathcal{M}_{40}^{(5)} = \begin{bmatrix} a & 0 \\ a & 0 \\ 0 & b \\ 0 & c \\ 0 & d \\ 0 & 0 \end{bmatrix}$$

$C_{\mathrm{NB}}(P)$=360

$$\mathcal{M}_{41}^{(5)} = \begin{bmatrix} a & 0 \\ a & 0 \\ b & 0 \\ 0 & c \\ 0 & d \\ 0 & 0 \end{bmatrix}$$

$C_{\mathrm{NB}}(P)$=180

$$\mathcal{M}_{42}^{(5)} = \begin{bmatrix} a & 0 & 0 \\ b & 0 & 0 \\ 0 & a & 0 \\ 0 & 0 & c \\ 0 & 0 & d \\ 0 & 0 & 0 \end{bmatrix}$$

$C_{\mathrm{NB}}(P)$=60

$$\mathcal{M}_{43}^{(5)} = \begin{bmatrix} a & 0 & 0 \\ 0 & a & 0 \\ 0 & 0 & b \\ 0 & 0 & c \\ 0 & 0 & d \\ 0 & 0 & 0 \end{bmatrix}$$

$C_{\mathrm{NB}}(P)$=180

$$\mathcal{M}_{44}^{(5)} = \begin{bmatrix} a & 0 \\ b & 0 \\ c & 0 \\ d & 0 \\ 0 & e \\ 0 & 0 \end{bmatrix}$$

$C_{\mathrm{NB}}(P)$=360

$$\mathcal{M}_{45}^{(5)} = \begin{bmatrix} a & 0 & 0 \\ b & 0 & 0 \\ c & 0 & 0 \\ 0 & d & 0 \\ 0 & 0 & e \\ 0 & 0 & 0 \end{bmatrix}$$

$C_{\mathrm{NB}}(P)$=60

$$\mathcal{M}_{46}^{(5)} = \begin{bmatrix} a & 0 & 0 \\ b & 0 & 0 \\ 0 & c & 0 \\ 0 & d & 0 \\ 0 & 0 & e \\ 0 & 0 & 0 \end{bmatrix}$$

$C_{\mathrm{NB}}(P)$=15

$$\mathcal{M}_{47}^{(5)} = \begin{bmatrix} a & 0 & 0 & 0 \\ b & 0 & 0 & 0 \\ 0 & c & 0 & 0 \\ 0 & 0 & d & 0 \\ 0 & 0 & 0 & e \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

$C_{\mathrm{NB}}(P)$=60

$$\mathcal{M}_{48}^{(5)} = \begin{bmatrix} a & 0 & 0 & 0 & 0 \\ 0 & b & 0 & 0 & 0 \\ 0 & 0 & c & 0 & 0 \\ 0 & 0 & 0 & d & 0 \\ 0 & 0 & 0 & 0 & e \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

$C_{\mathrm{NB}}(P)$=45

$$\mathcal{M}_{49}^{(5)} = \begin{bmatrix} a & 0 \\ b & 0 \\ c & 0 \\ 0 & d \\ 0 & e \\ 0 & 0 \end{bmatrix}$$

$C_{\mathrm{NB}}(P)$=60

$$\mathcal{M}_{50}^{(5)} = \begin{bmatrix} a & 0 & 0 \\ b & 0 & 0 \\ 0 & a & 0 \\ 0 & c & 0 \\ 0 & 0 & b \\ 0 & 0 & 0 \end{bmatrix}$$

$C_{\mathrm{NB}}(P)$=6

$$\mathcal{M}_{51}^{(5)} = \begin{bmatrix} a & 0 & 0 \\ a & 0 & 0 \\ 0 & b & 0 \\ 0 & c & 0 \\ 0 & 0 & b \\ 0 & 0 & 0 \end{bmatrix}$$

$C_{\mathrm{NB}}(P)$=30

$C_{\mathrm{NB}}(P)$=720

$C_{\mathrm{NB}}(P)$=360

A.6 Six-mark tables

Table A.6: Mark-pattern tables with $r = 6$ marks.

$\mathcal{M}_1^{(6)} = \begin{bmatrix} a \\ a \\ a \\ a \\ a \\ a \end{bmatrix}$	$\mathcal{M}_2^{(6)} = \begin{bmatrix} a & 0 \\ a & 0 \\ a & 0 \\ a & 0 \\ 0 & a \\ 0 & a \end{bmatrix}$	$\mathcal{M}_3^{(6)} = \begin{bmatrix} a & 0 & 0 \\ a & 0 & 0 \\ 0 & a & 0 \\ 0 & a & 0 \\ 0 & 0 & a \\ 0 & 0 & a \end{bmatrix}$
$C_{\mathrm{NB}}(P)$=1	$C_{\mathrm{NB}}(P)$=15	$C_{\mathrm{NB}}(P)$=15
$\mathcal{M}_4^{(6)} = \begin{bmatrix} a & 0 & 0 \\ a & 0 & 0 \\ 0 & a & 0 \\ 0 & a & 0 \\ 0 & 0 & b \\ 0 & 0 & b \end{bmatrix}$	$\mathcal{M}_5^{(6)} = \begin{bmatrix} a & 0 \\ a & 0 \\ a & 0 \\ a & 0 \\ 0 & b \\ 0 & b \end{bmatrix}$	$\mathcal{M}_6^{(6)} = \begin{bmatrix} a & 0 \\ a & 0 \\ b & 0 \\ b & 0 \\ 0 & a \\ 0 & a \end{bmatrix}$
$C_{\mathrm{NB}}(P)$=45	$C_{\mathrm{NB}}(P)$=15	$C_{\mathrm{NB}}(P)$=90
$\mathcal{M}_7^{(6)} = \begin{bmatrix} a \\ a \\ a \\ a \\ b \\ b \end{bmatrix}$	$\mathcal{M}_8^{(6)} = \begin{bmatrix} a & 0 & 0 \\ b & 0 & 0 \\ 0 & a & 0 \\ 0 & b & 0 \\ 0 & 0 & a \\ 0 & 0 & a \end{bmatrix}$	$\mathcal{M}_9^{(6)} = \begin{bmatrix} a & 0 \\ a & 0 \\ a & 0 \\ b & 0 \\ 0 & a \\ 0 & b \end{bmatrix}$
$C_{\mathrm{NB}}(P)$=15	$C_{\mathrm{NB}}(P)$=180	$C_{\mathrm{NB}}(P)$=120
$\mathcal{M}_{10}^{(6)} = \begin{bmatrix} a \\ a \\ a \\ b \\ b \\ b \end{bmatrix}$	$\mathcal{M}_{11}^{(6)} = \begin{bmatrix} a & 0 \\ a & 0 \\ a & 0 \\ 0 & b \\ 0 & b \\ 0 & b \end{bmatrix}$	$\mathcal{M}_{12}^{(6)} = \begin{bmatrix} a & 0 \\ a & 0 \\ a & 0 \\ b & 0 \\ 0 & b \\ 0 & b \end{bmatrix}$
$C_{\mathrm{NB}}(P)$=10	$C_{\mathrm{NB}}(P)$=10	$C_{\mathrm{NB}}(P)$=60
$\mathcal{M}_{13}^{(6)} = \begin{bmatrix} a & 0 \\ a & 0 \\ b & 0 \\ b & 0 \\ 0 & a \\ 0 & b \end{bmatrix}$	$\mathcal{M}_{14}^{(6)} = \begin{bmatrix} a & 0 & 0 \\ a & 0 & 0 \\ a & 0 & 0 \\ 0 & b & 0 \\ 0 & b & 0 \\ 0 & 0 & b \end{bmatrix}$	$\mathcal{M}_{15}^{(6)} = \begin{bmatrix} a & 0 & 0 \\ a & 0 & 0 \\ 0 & b & 0 \\ 0 & a & 0 \\ 0 & 0 & b \\ 0 & 0 & b \end{bmatrix}$
$C_{\mathrm{NB}}(P)$=90	$C_{\mathrm{NB}}(P)$=60	$C_{\mathrm{NB}}(P)$=90
$\mathcal{M}_{16}^{(6)} = \begin{bmatrix} a & 0 & 0 \\ a & 0 & 0 \\ b & 0 & 0 \\ b & 0 & 0 \\ 0 & b & 0 \\ 0 & 0 & a \end{bmatrix}$	$\mathcal{M}_{17}^{(6)} = \begin{bmatrix} a & 0 & 0 \\ b & 0 & 0 \\ 0 & a & 0 \\ 0 & b & 0 \\ 0 & 0 & a \\ 0 & 0 & b \end{bmatrix}$	$\mathcal{M}_{18}^{(6)} = \begin{bmatrix} a & 0 & 0 & 0 \\ a & 0 & 0 & 0 \\ 0 & b & 0 & 0 \\ 0 & b & 0 & 0 \\ 0 & 0 & a & 0 \\ 0 & 0 & 0 & b \end{bmatrix}$
$C_{\mathrm{NB}}(P)$=90	$C_{\mathrm{NB}}(P)$=60	$C_{\mathrm{NB}}(P)$=90

$$\mathcal{M}_{19}^{(6)} = \begin{bmatrix} a & 0 & 0 & 0 \\ b & 0 & 0 & 0 \\ 0 & a & 0 & 0 \\ 0 & b & 0 & 0 \\ 0 & 0 & a & 0 \\ 0 & 0 & 0 & b \end{bmatrix}$$

$C_{\mathrm{NB}}(P)$=180

$$\mathcal{M}_{20}^{(6)} = \begin{bmatrix} a & 0 & 0 \\ a & 0 & 0 \\ a & 0 & 0 \\ a & 0 & 0 \\ 0 & b & 0 \\ 0 & 0 & c \end{bmatrix}$$

$C_{\mathrm{NB}}(P)$=15

$$\mathcal{M}_{21}^{(6)} = \begin{bmatrix} a & 0 & 0 \\ a & 0 & 0 \\ a & 0 & 0 \\ b & 0 & 0 \\ 0 & a & 0 \\ 0 & 0 & c \end{bmatrix}$$

$C_{\mathrm{NB}}(P)$=120

$$\mathcal{M}_{22}^{(6)} = \begin{bmatrix} a & 0 & 0 \\ a & 0 & 0 \\ b & 0 & 0 \\ c & 0 & 0 \\ 0 & a & 0 \\ 0 & 0 & a \end{bmatrix}$$

$C_{\mathrm{NB}}(P)$=90

$$\mathcal{M}_{23}^{(6)} = \begin{bmatrix} a \\ a \\ a \\ a \\ b \\ c \end{bmatrix}$$

$C_{\mathrm{NB}}(P)$=15

$$\mathcal{M}_{24}^{(6)} = \begin{bmatrix} a & 0 \\ a & 0 \\ a & 0 \\ a & 0 \\ 0 & b \\ 0 & c \end{bmatrix}$$

$C_{\mathrm{NB}}(P)$=15

$$\mathcal{M}_{25}^{(6)} = \begin{bmatrix} a & 0 \\ a & 0 \\ a & 0 \\ b & 0 \\ 0 & a \\ 0 & c \end{bmatrix}$$

$C_{\mathrm{NB}}(P)$=120

$$\mathcal{M}_{26}^{(6)} = \begin{bmatrix} a & 0 \\ a & 0 \\ b & 0 \\ c & 0 \\ 0 & a \\ 0 & a \end{bmatrix}$$

$C_{\mathrm{NB}}(P)$=90

$$\mathcal{M}_{27}^{(6)} = \begin{bmatrix} a & 0 \\ a & 0 \\ b & 0 \\ 0 & a \\ 0 & a \\ 0 & c \end{bmatrix}$$

$C_{\mathrm{NB}}(P)$=90

$$\mathcal{M}_{28}^{(6)} = \begin{bmatrix} a & 0 & 0 \\ a & 0 & 0 \\ b & 0 & 0 \\ 0 & a & 0 \\ 0 & c & 0 \\ 0 & 0 & a \end{bmatrix}$$

$C_{\mathrm{NB}}(P)$=360

$$\mathcal{M}_{29}^{(6)} = \begin{bmatrix} a & 0 & 0 \\ a & 0 & 0 \\ b & 0 & 0 \\ 0 & a & 0 \\ 0 & a & 0 \\ 0 & 0 & c \end{bmatrix}$$

$C_{\mathrm{NB}}(P)$=180

$$\mathcal{M}_{30}^{(6)} = \begin{bmatrix} a & 0 & 0 \\ b & 0 & 0 \\ c & 0 & 0 \\ 0 & a & 0 \\ 0 & a & 0 \\ 0 & 0 & a \end{bmatrix}$$

$C_{\mathrm{NB}}(P)$=180

$$\mathcal{M}_{31}^{(6)} = \begin{bmatrix} a & 0 & 0 \\ a & 0 & 0 \\ 0 & a & 0 \\ 0 & a & 0 \\ 0 & 0 & b \\ 0 & 0 & c \end{bmatrix}$$

$C_{\mathrm{NB}}(P)$=45

$$\mathcal{M}_{32}^{(6)} = \begin{bmatrix} a & 0 & 0 \\ a & 0 & 0 \\ 0 & a & 0 \\ 0 & b & 0 \\ 0 & 0 & a \\ 0 & 0 & c \end{bmatrix}$$

$C_{\mathrm{NB}}(P)$=180

$$\mathcal{M}_{33}^{(6)} = \begin{bmatrix} a & 0 & 0 & 0 \\ a & 0 & 0 & 0 \\ 0 & a & 0 & 0 \\ 0 & a & 0 & 0 \\ 0 & 0 & b & 0 \\ 0 & 0 & 0 & c \end{bmatrix}$$

$C_{\mathrm{NB}}(P)$=180

$$\mathcal{M}_{34}^{(6)} = \begin{bmatrix} a & 0 & 0 & 0 \\ a & 0 & 0 & 0 \\ 0 & a & 0 & 0 \\ 0 & b & 0 & 0 \\ 0 & 0 & a & 0 \\ 0 & 0 & 0 & c \end{bmatrix}$$

$C_{\mathrm{NB}}(P)$=45

$$\mathcal{M}_{35}^{(6)} = \begin{bmatrix} a & 0 & 0 & 0 \\ a & 0 & 0 & 0 \\ 0 & b & 0 & 0 \\ 0 & c & 0 & 0 \\ 0 & 0 & a & 0 \\ 0 & 0 & 0 & a \end{bmatrix}$$

$C_{\mathrm{NB}}(P)$=180

$$\mathcal{M}_{36}^{(6)} = \begin{bmatrix} a & 0 & 0 & 0 \\ b & 0 & 0 & 0 \\ 0 & a & 0 & 0 \\ 0 & c & 0 & 0 \\ 0 & 0 & a & 0 \\ 0 & 0 & 0 & a \end{bmatrix}$$

$C_{\mathrm{NB}}(P)$=45

$$\mathcal{M}_{37}^{(6)} = \begin{bmatrix} a \\ a \\ a \\ b \\ b \\ c \end{bmatrix}$$

$C_{\mathrm{NB}}(P)$=360

$$\mathcal{M}_{38}^{(6)} = \begin{bmatrix} a & 0 \\ a & 0 \\ a & 0 \\ b & 0 \\ 0 & b \\ 0 & c \end{bmatrix}$$

$C_{\mathrm{NB}}(P)$=90

$$\mathcal{M}_{39}^{(6)} = \begin{bmatrix} a & 0 \\ a & 0 \\ b & 0 \\ b & 0 \\ 0 & a \\ 0 & c \end{bmatrix}$$

$C_{\mathrm{NB}}(P)$=180

$C_{\mathrm{NB}}(P)$=60

$C_{\mathrm{NB}}(P)$=120

$C_{\mathrm{NB}}(P)$=180

$$\mathcal{M}_{40}^{(6)} = \begin{bmatrix} a & 0 \\ a & 0 \\ b & 0 \\ c & 0 \\ 0 & a \\ 0 & b \end{bmatrix}$$

$C_{\mathrm{NB}}(P)$=360

$$\mathcal{M}_{41}^{(6)} = \begin{bmatrix} a & 0 \\ a & 0 \\ a & 0 \\ c & 0 \\ 0 & b \\ 0 & b \end{bmatrix}$$

$C_{\mathrm{NB}}(P)$=60

$$\mathcal{M}_{42}^{(6)} = \begin{bmatrix} a & 0 \\ b & 0 \\ b & 0 \\ c & 0 \\ 0 & a \\ 0 & a \end{bmatrix}$$

$C_{\mathrm{NB}}(P)$=180

$$\mathcal{M}_{43}^{(6)} = \begin{bmatrix} a & 0 \\ a & 0 \\ a & 0 \\ 0 & b \\ 0 & b \\ 0 & c \end{bmatrix}$$

$C_{\mathrm{NB}}(P)$=60

$$\mathcal{M}_{44}^{(6)} = \begin{bmatrix} a & 0 & 0 \\ a & 0 & 0 \\ b & 0 & 0 \\ b & 0 & 0 \\ 0 & a & 0 \\ 0 & 0 & c \end{bmatrix}$$

$C_{\mathrm{NB}}(P)$=180

$$\mathcal{M}_{45}^{(6)} = \begin{bmatrix} a & 0 & 0 \\ a & 0 & 0 \\ b & 0 & 0 \\ c & 0 & 0 \\ 0 & a & 0 \\ 0 & 0 & b \end{bmatrix}$$

$C_{\mathrm{NB}}(P)$=60

$$\mathcal{M}_{46}^{(6)} = \begin{bmatrix} a & 0 & 0 \\ b & 0 & 0 \\ b & 0 & 0 \\ c & 0 & 0 \\ 0 & a & 0 \\ 0 & 0 & a \end{bmatrix}$$

$C_{\mathrm{NB}}(P)$=180

$$\mathcal{M}_{47}^{(6)} = \begin{bmatrix} a & 0 & 0 \\ a & 0 & 0 \\ a & 0 & 0 \\ 0 & b & 0 \\ 0 & b & 0 \\ 0 & 0 & c \end{bmatrix}$$

$C_{\mathrm{NB}}(P)$=360

$$\mathcal{M}_{48}^{(6)} = \begin{bmatrix} a & 0 & 0 \\ a & 0 & 0 \\ a & 0 & 0 \\ 0 & b & 0 \\ 0 & c & 0 \\ 0 & 0 & b \end{bmatrix}$$

$C_{\mathrm{NB}}(P)$=180

$$\mathcal{M}_{49}^{(6)} = \begin{bmatrix} a & 0 & 0 \\ b & 0 & 0 \\ c & 0 & 0 \\ 0 & a & 0 \\ 0 & b & 0 \\ 0 & 0 & a \end{bmatrix}$$

$C_{\mathrm{NB}}(P)$=60

$$\mathcal{M}_{50}^{(6)} = \begin{bmatrix} b & 0 & 0 \\ b & 0 & 0 \\ c & 0 & 0 \\ 0 & a & 0 \\ 0 & a & 0 \\ 0 & 0 & a \end{bmatrix}$$

$C_{\mathrm{NB}}(P)$=120

$$\mathcal{M}_{51}^{(6)} = \begin{bmatrix} a & 0 & 0 \\ a & 0 & 0 \\ 0 & a & 0 \\ 0 & b & 0 \\ 0 & 0 & b \\ 0 & 0 & c \end{bmatrix}$$

$C_{\mathrm{NB}}(P)$=720

$$\mathcal{M}_{52}^{(6)} = \begin{bmatrix} a & 0 & 0 \\ a & 0 & 0 \\ 0 & b & 0 \\ 0 & b & 0 \\ 0 & 0 & a \\ 0 & 0 & c \end{bmatrix}$$

$C_{\mathrm{NB}}(P)$=180

$$\mathcal{M}_{53}^{(6)} = \begin{bmatrix} a & 0 & 0 \\ b & 0 & 0 \\ 0 & a & 0 \\ 0 & b & 0 \\ 0 & 0 & a \\ 0 & 0 & c \end{bmatrix}$$

$C_{\mathrm{NB}}(P)$=360

$$\mathcal{M}_{54}^{(6)} = \begin{bmatrix} a & 0 & 0 & 0 \\ a & 0 & 0 & 0 \\ 0 & b & 0 & 0 \\ 0 & b & 0 & 0 \\ 0 & 0 & a & 0 \\ 0 & 0 & 0 & c \end{bmatrix}$$

$C_{\mathrm{NB}}(P)$=180

$$\mathcal{M}_{55}^{(6)} = \begin{bmatrix} a & 0 & 0 & 0 \\ b & 0 & 0 & 0 \\ 0 & a & 0 & 0 \\ 0 & b & 0 & 0 \\ 0 & 0 & a & 0 \\ 0 & 0 & 0 & c \end{bmatrix}$$

$C_{\mathrm{NB}}(P)$=360

$$\mathcal{M}_{56}^{(6)} = \begin{bmatrix} a & 0 & 0 & 0 \\ a & 0 & 0 & 0 \\ 0 & b & 0 & 0 \\ 0 & c & 0 & 0 \\ 0 & 0 & a & 0 \\ 0 & 0 & 0 & b \end{bmatrix}$$

$C_{\mathrm{NB}}(P)$=180

$$\mathcal{M}_{57}^{(6)} = \begin{bmatrix} a & 0 & 0 & 0 \\ b & 0 & 0 & 0 \\ 0 & a & 0 & 0 \\ 0 & c & 0 & 0 \\ 0 & 0 & a & 0 \\ 0 & 0 & 0 & b \end{bmatrix}$$

$C_{\mathrm{NB}}(P)$=360

$$\mathcal{M}_{58}^{(6)} = \begin{bmatrix} a \\ a \\ b \\ b \\ c \\ c \end{bmatrix}$$

$C_{\mathrm{NB}}(P)$=15

$$\mathcal{M}_{59}^{(6)} = \begin{bmatrix} a & 0 \\ a & 0 \\ b & 0 \\ b & 0 \\ 0 & c \\ 0 & c \end{bmatrix}$$

$C_{\mathrm{NB}}(P)$=45

$$\mathcal{M}_{60}^{(6)} = \begin{bmatrix} a & 0 \\ a & 0 \\ b & 0 \\ c & 0 \\ 0 & b \\ 0 & c \end{bmatrix}$$

$C_{\mathrm{NB}}(P)$=180

$$\mathcal{M}_{61}^{(6)} = \begin{bmatrix} a & 0 & 0 \\ a & 0 & 0 \\ 0 & b & 0 \\ 0 & b & 0 \\ 0 & 0 & c \\ 0 & 0 & c \end{bmatrix}$$

$C_{\mathrm{NB}}(P)$=15

$$\mathcal{M}_{62}^{(6)} = \begin{bmatrix} a & 0 & 0 \\ b & 0 & 0 \\ 0 & a & 0 \\ 0 & b & 0 \\ 0 & 0 & c \\ 0 & 0 & c \end{bmatrix}$$

$C_{\mathrm{NB}}(P)$=90

$$\mathcal{M}_{63}^{(6)} = \begin{bmatrix} a & 0 & 0 \\ b & 0 & 0 \\ 0 & b & 0 \\ 0 & c & 0 \\ 0 & 0 & a \\ 0 & 0 & c \end{bmatrix}$$

$C_{\mathrm{NB}}(P)$=120

$$\mathcal{M}_{64}^{(6)} = \begin{bmatrix} a \\ a \\ a \\ b \\ c \\ d \end{bmatrix}$$

$C_{\mathrm{NB}}(P)$=10

$$\mathcal{M}_{65}^{(6)} = \begin{bmatrix} a & 0 \\ a & 0 \\ a & 0 \\ b & 0 \\ 0 & c \\ 0 & d \end{bmatrix}$$

$C_{\mathrm{NB}}(P)$=30

$$\mathcal{M}_{66}^{(6)} = \begin{bmatrix} a & 0 \\ a & 0 \\ b & 0 \\ c & 0 \\ 0 & a \\ 0 & d \end{bmatrix}$$

$C_{\mathrm{NB}}(P)$=90

$$\mathcal{M}_{67}^{(6)} = \begin{bmatrix} a & 0 \\ b & 0 \\ c & 0 \\ d & 0 \\ 0 & a \\ 0 & a \end{bmatrix}$$

$C_{\mathrm{NB}}(P)$=30

$$\mathcal{M}_{68}^{(6)} = \begin{bmatrix} a & 0 & 0 \\ a & 0 & 0 \\ a & 0 & 0 \\ b & 0 & 0 \\ 0 & c & 0 \\ 0 & 0 & d \end{bmatrix}$$

$C_{\mathrm{NB}}(P)$=60

$$\mathcal{M}_{69}^{(6)} = \begin{bmatrix} a & 0 & 0 \\ a & 0 & 0 \\ b & 0 & 0 \\ c & 0 & 0 \\ 0 & a & 0 \\ 0 & 0 & d \end{bmatrix}$$

$C_{\mathrm{NB}}(P)$=180

$$\mathcal{M}_{70}^{(6)} = \begin{bmatrix} a & 0 & 0 \\ b & 0 & 0 \\ c & 0 & 0 \\ d & 0 & 0 \\ 0 & a & 0 \\ 0 & 0 & a \end{bmatrix}$$

$C_{\mathrm{NB}}(P)$=60

$$\mathcal{M}_{71}^{(6)} = \begin{bmatrix} a & 0 \\ a & 0 \\ a & 0 \\ 0 & b \\ 0 & c \\ 0 & d \end{bmatrix}$$

$C_{\mathrm{NB}}(P)$=180

$$\mathcal{M}_{72}^{(6)} = \begin{bmatrix} a & 0 \\ a & 0 \\ b & 0 \\ 0 & a \\ 0 & c \\ 0 & d \end{bmatrix}$$

$C_{\mathrm{NB}}(P)$=60

$$\mathcal{M}_{73}^{(6)} = \begin{bmatrix} a & 0 & 0 \\ a & 0 & 0 \\ a & 0 & 0 \\ 0 & b & 0 \\ 0 & c & 0 \\ 0 & 0 & d \end{bmatrix}$$

$C_{\mathrm{NB}}(P)$=20

$$\mathcal{M}_{74}^{(6)} = \begin{bmatrix} a & 0 & 0 \\ a & 0 & 0 \\ b & 0 & 0 \\ 0 & a & 0 \\ 0 & c & 0 \\ 0 & 0 & d \end{bmatrix}$$

$C_{\mathrm{NB}}(P)$=180

$$\mathcal{M}_{75}^{(6)} = \begin{bmatrix} a & 0 & 0 \\ a & 0 & 0 \\ b & 0 & 0 \\ 0 & c & 0 \\ 0 & d & 0 \\ 0 & 0 & a \end{bmatrix}$$

$C_{\mathrm{NB}}(P)$=60

$$\mathcal{M}_{76}^{(6)} = \begin{bmatrix} a & 0 & 0 \\ b & 0 & 0 \\ c & 0 & 0 \\ 0 & a & 0 \\ 0 & a & 0 \\ 0 & 0 & d \end{bmatrix}$$

$C_{\mathrm{NB}}(P)$=360

$$\mathcal{M}_{77}^{(6)} = \begin{bmatrix} a & 0 & 0 \\ b & 0 & 0 \\ c & 0 & 0 \\ 0 & a & 0 \\ 0 & d & 0 \\ 0 & 0 & a \end{bmatrix}$$

$C_{\mathrm{NB}}(P)$=180

$$\mathcal{M}_{78}^{(6)} = \begin{bmatrix} b & 0 & 0 \\ c & 0 & 0 \\ d & 0 & 0 \\ 0 & a & 0 \\ 0 & a & 0 \\ 0 & 0 & a \end{bmatrix}$$

$C_{\mathrm{NB}}(P)$=180

$$\mathcal{M}_{79}^{(6)} = \begin{bmatrix} a & 0 & 0 & 0 \\ a & 0 & 0 & 0 \\ a & 0 & 0 & 0 \\ 0 & b & 0 & 0 \\ 0 & 0 & c & 0 \\ 0 & 0 & 0 & d \end{bmatrix}$$

$C_{\mathrm{NB}}(P)$=20

$$\mathcal{M}_{80}^{(6)} = \begin{bmatrix} a & 0 & 0 & 0 \\ a & 0 & 0 & 0 \\ b & 0 & 0 & 0 \\ 0 & a & 0 & 0 \\ 0 & 0 & c & 0 \\ 0 & 0 & 0 & d \end{bmatrix}$$

$C_{\mathrm{NB}}(P)$=180

$$\mathcal{M}_{81}^{(6)} = \begin{bmatrix} a & 0 & 0 & 0 \\ b & 0 & 0 & 0 \\ c & 0 & 0 & 0 \\ 0 & a & 0 & 0 \\ 0 & 0 & a & 0 \\ 0 & 0 & 0 & d \end{bmatrix}$$

$C_{\mathrm{NB}}(P)$=180

$$\mathcal{M}_{82}^{(6)} = \begin{bmatrix} b & 0 & 0 & 0 \\ c & 0 & 0 & 0 \\ d & 0 & 0 & 0 \\ 0 & a & 0 & 0 \\ 0 & 0 & a & 0 \\ 0 & 0 & 0 & a \end{bmatrix}$$

$C_{\mathrm{NB}}(P)$=20

$$\mathcal{M}_{83}^{(6)} = \begin{bmatrix} a & 0 & 0 \\ a & 0 & 0 \\ 0 & a & 0 \\ 0 & b & 0 \\ 0 & 0 & c \\ 0 & 0 & d \end{bmatrix}$$

$C_{\mathrm{NB}}(P)$=90

$$\mathcal{M}_{84}^{(6)} = \begin{bmatrix} a & 0 & 0 \\ b & 0 & 0 \\ 0 & a & 0 \\ 0 & c & 0 \\ 0 & 0 & a \\ 0 & 0 & d \end{bmatrix}$$

$C_{\mathrm{NB}}(P)$=60

$$\mathcal{M}_{85}^{(6)} = \begin{bmatrix} a & 0 & 0 & 0 \\ a & 0 & 0 & 0 \\ 0 & a & 0 & 0 \\ 0 & b & 0 & 0 \\ 0 & 0 & c & 0 \\ 0 & 0 & 0 & d \end{bmatrix}$$

$C_{\mathrm{NB}}(P)$=180

$$\mathcal{M}_{86}^{(6)} = \begin{bmatrix} a & 0 & 0 & 0 \\ a & 0 & 0 & 0 \\ 0 & b & 0 & 0 \\ 0 & c & 0 & 0 \\ 0 & 0 & a & 0 \\ 0 & 0 & 0 & d \end{bmatrix}$$

$C_{\mathrm{NB}}(P)$=180

$$\mathcal{M}_{87}^{(6)} = \begin{bmatrix} a & 0 & 0 & 0 \\ b & 0 & 0 & 0 \\ 0 & a & 0 & 0 \\ 0 & c & 0 & 0 \\ 0 & 0 & a & 0 \\ 0 & 0 & 0 & d \end{bmatrix}$$

$C_{\mathrm{NB}}(P)$=360

$$\mathcal{M}_{88}^{(6)} = \begin{bmatrix} b & 0 & 0 & 0 \\ c & 0 & 0 & 0 \\ 0 & a & 0 & 0 \\ 0 & d & 0 & 0 \\ 0 & 0 & a & 0 \\ 0 & 0 & 0 & a \end{bmatrix}$$

$C_{\mathrm{NB}}(P)$=180

$$\mathcal{M}_{89}^{(6)} = \begin{bmatrix} a \\ a \\ b \\ b \\ c \\ d \end{bmatrix}$$

$C_{\mathrm{NB}}(P)$=45

$$\mathcal{M}_{90}^{(6)} = \begin{bmatrix} a & 0 \\ a & 0 \\ b & 0 \\ b & 0 \\ 0 & c \\ 0 & d \end{bmatrix}$$

$C_{\mathrm{NB}}(P)$=45

$$\mathcal{M}_{91}^{(6)} = \begin{bmatrix} a & 0 \\ a & 0 \\ b & 0 \\ c & 0 \\ 0 & b \\ 0 & d \end{bmatrix}$$

$C_{\mathrm{NB}}(P)$=180

$$\mathcal{M}_{92}^{(6)} = \begin{bmatrix} a & 0 \\ a & 0 \\ c & 0 \\ d & 0 \\ 0 & b \\ 0 & b \end{bmatrix}$$

$C_{\mathrm{NB}}(P)$=45

$$\mathcal{M}_{93}^{(6)} = \begin{bmatrix} a & 0 \\ b & 0 \\ c & 0 \\ d & 0 \\ 0 & a \\ 0 & b \end{bmatrix}$$

$C_{\mathrm{NB}}(P)$=45

$$\mathcal{M}_{94}^{(6)} = \begin{bmatrix} a & 0 & 0 \\ a & 0 & 0 \\ b & 0 & 0 \\ b & 0 & 0 \\ 0 & c & 0 \\ 0 & 0 & d \end{bmatrix}$$

$C_{\mathrm{NB}}(P)$=360

$$\mathcal{M}_{95}^{(6)} = \begin{bmatrix} a & 0 & 0 \\ a & 0 & 0 \\ b & 0 & 0 \\ c & 0 & 0 \\ 0 & b & 0 \\ 0 & 0 & d \end{bmatrix}$$

$C_{\mathrm{NB}}(P)$=90

$$\mathcal{M}_{96}^{(6)} = \begin{bmatrix} a & 0 & 0 \\ a & 0 & 0 \\ d & 0 & 0 \\ c & 0 & 0 \\ 0 & b & 0 \\ 0 & 0 & b \end{bmatrix}$$

$C_{\mathrm{NB}}(P)$=180

$$\mathcal{M}_{97}^{(6)} = \begin{bmatrix} a & 0 & 0 \\ b & 0 & 0 \\ c & 0 & 0 \\ d & 0 & 0 \\ 0 & a & 0 \\ 0 & 0 & b \end{bmatrix}$$

$C_{\mathrm{NB}}(P)$=360

$$\mathcal{M}_{98}^{(6)} = \begin{bmatrix} a & 0 \\ a & 0 \\ c & 0 \\ 0 & b \\ 0 & b \\ 0 & d \end{bmatrix}$$

$C_{\mathrm{NB}}(P)$=90

$$\mathcal{M}_{99}^{(6)} = \begin{bmatrix} a & 0 \\ b & 0 \\ c & 0 \\ 0 & a \\ 0 & b \\ 0 & d \end{bmatrix}$$

$C_{\mathrm{NB}}(P)$=90

$$\mathcal{M}_{100}^{(6)} = \begin{bmatrix} a & 0 & 0 \\ a & 0 & 0 \\ c & 0 & 0 \\ 0 & b & 0 \\ 0 & b & 0 \\ 0 & 0 & d \end{bmatrix}$$

$C_{\mathrm{NB}}(P)$=180

$$\mathcal{M}_{101}^{(6)} = \begin{bmatrix} a & 0 & 0 \\ a & 0 & 0 \\ c & 0 & 0 \\ 0 & b & 0 \\ 0 & d & 0 \\ 0 & 0 & b \end{bmatrix}$$

$C_{\mathrm{NB}}(P)$=90

$$\mathcal{M}_{102}^{(6)} = \begin{bmatrix} a & 0 & 0 \\ b & 0 & 0 \\ c & 0 & 0 \\ 0 & a & 0 \\ 0 & b & 0 \\ 0 & 0 & d \end{bmatrix}$$

$C_{\mathrm{NB}}(P)$=180

$C_{\mathrm{NB}}(P)$=360

$C_{\mathrm{NB}}(P)$=360

$\mathcal{M}_{103}^{(6)} = \begin{bmatrix} a & 0 & 0 \\ b & 0 & 0 \\ c & 0 & 0 \\ 0 & a & 0 \\ 0 & d & 0 \\ 0 & 0 & b \end{bmatrix}$

$C_{\mathrm{NB}}(P)$=720

$\mathcal{M}_{104}^{(6)} = \begin{bmatrix} a & 0 & 0 \\ c & 0 & 0 \\ d & 0 & 0 \\ 0 & b & 0 \\ 0 & b & 0 \\ 0 & 0 & a \end{bmatrix}$

$C_{\mathrm{NB}}(P)$=180

$\mathcal{M}_{105}^{(6)} = \begin{bmatrix} a & 0 & 0 \\ c & 0 & 0 \\ d & 0 & 0 \\ 0 & a & 0 \\ 0 & b & 0 \\ 0 & 0 & b \end{bmatrix}$

$C_{\mathrm{NB}}(P)$=360

$\mathcal{M}_{106}^{(6)} = \begin{bmatrix} a & 0 & 0 \\ a & 0 & 0 \\ 0 & b & 0 \\ 0 & b & 0 \\ 0 & 0 & c \\ 0 & 0 & d \end{bmatrix}$

$C_{\mathrm{NB}}(P)$=45

$\mathcal{M}_{107}^{(6)} = \begin{bmatrix} a & 0 & 0 \\ a & 0 & 0 \\ 0 & b & 0 \\ 0 & c & 0 \\ 0 & 0 & b \\ 0 & 0 & d \end{bmatrix}$

$C_{\mathrm{NB}}(P)$=180

$\mathcal{M}_{108}^{(6)} = \begin{bmatrix} a & 0 & 0 \\ b & 0 & 0 \\ 0 & a & 0 \\ 0 & c & 0 \\ 0 & 0 & b \\ 0 & 0 & d \end{bmatrix}$

$C_{\mathrm{NB}}(P)$=360

$\mathcal{M}_{109}^{(6)} = \begin{bmatrix} a & 0 & 0 & 0 \\ a & 0 & 0 & 0 \\ 0 & b & 0 & 0 \\ 0 & b & 0 & 0 \\ 0 & 0 & c & 0 \\ 0 & 0 & 0 & d \end{bmatrix}$

$C_{\mathrm{NB}}(P)$=45

$\mathcal{M}_{110}^{(6)} = \begin{bmatrix} a & 0 & 0 & 0 \\ a & 0 & 0 & 0 \\ 0 & c & 0 & 0 \\ 0 & d & 0 & 0 \\ 0 & 0 & b & 0 \\ 0 & 0 & 0 & b \end{bmatrix}$

$C_{\mathrm{NB}}(P)$=180

$\mathcal{M}_{111}^{(6)} = \begin{bmatrix} a & 0 & 0 & 0 \\ a & 0 & 0 & 0 \\ 0 & c & 0 & 0 \\ 0 & d & 0 & 0 \\ 0 & 0 & b & 0 \\ 0 & 0 & 0 & d \end{bmatrix}$

$C_{\mathrm{NB}}(P)$=360

$\mathcal{M}_{112}^{(6)} = \begin{bmatrix} a & 0 & 0 & 0 \\ c & 0 & 0 & 0 \\ 0 & a & 0 & 0 \\ 0 & d & 0 & 0 \\ 0 & 0 & b & 0 \\ 0 & 0 & 0 & b \end{bmatrix}$

$C_{\mathrm{NB}}(P)$=45

$\mathcal{M}_{113}^{(6)} = \begin{bmatrix} a & 0 & 0 & 0 \\ b & 0 & 0 & 0 \\ 0 & a & 0 & 0 \\ 0 & c & 0 & 0 \\ 0 & 0 & b & 0 \\ 0 & 0 & 0 & d \end{bmatrix}$

$C_{\mathrm{NB}}(P)$=90

$\mathcal{M}_{114}^{(6)} = \begin{bmatrix} a & 0 & 0 & 0 \\ c & 0 & 0 & 0 \\ 0 & b & 0 & 0 \\ 0 & d & 0 & 0 \\ 0 & 0 & a & 0 \\ 0 & 0 & 0 & b \end{bmatrix}$

$C_{\mathrm{NB}}(P)$=360

$\mathcal{M}_{115}^{(6)} = \begin{bmatrix} a \\ a \\ b \\ c \\ d \\ e \end{bmatrix}$

$C_{\mathrm{NB}}(P)$=$\frac{15}{2}$

$\mathcal{M}_{116}^{(6)} = \begin{bmatrix} a & 0 \\ a & 0 \\ b & 0 \\ c & 0 \\ 0 & d \\ 0 & e \end{bmatrix}$

$C_{\mathrm{NB}}(P)$=720

$\mathcal{M}_{117}^{(6)} = \begin{bmatrix} a & 0 \\ b & 0 \\ c & 0 \\ d & 0 \\ 0 & a \\ 0 & e \end{bmatrix}$

$C_{\mathrm{NB}}(P)$=360

$\mathcal{M}_{118}^{(6)} = \begin{bmatrix} b & 0 \\ c & 0 \\ d & 0 \\ e & 0 \\ 0 & a \\ 0 & a \end{bmatrix}$

$C_{\mathrm{NB}}(P)$=$\frac{15}{2}$

$\mathcal{M}_{119}^{(6)} = \begin{bmatrix} a & 0 & 0 \\ a & 0 & 0 \\ b & 0 & 0 \\ c & 0 & 0 \\ 0 & d & 0 \\ 0 & 0 & e \end{bmatrix}$

$C_{\mathrm{NB}}(P)$=45

$\mathcal{M}_{120}^{(6)} = \begin{bmatrix} a & 0 & 0 \\ b & 0 & 0 \\ c & 0 & 0 \\ d & 0 & 0 \\ 0 & a & 0 \\ 0 & 0 & e \end{bmatrix}$

$C_{\mathrm{NB}}(P)$=60

$\mathcal{M}_{121}^{(6)} = \begin{bmatrix} b & 0 & 0 \\ c & 0 & 0 \\ d & 0 & 0 \\ e & 0 & 0 \\ 0 & a & 0 \\ 0 & 0 & a \end{bmatrix}$

$C_{\mathrm{NB}}(P)$=15

$\mathcal{M}_{122}^{(6)} = \begin{bmatrix} a & 0 \\ a & 0 \\ b & 0 \\ 0 & c \\ 0 & d \\ 0 & e \end{bmatrix}$

$C_{\mathrm{NB}}(P)$=60

$\mathcal{M}_{123}^{(6)} = \begin{bmatrix} a & 0 \\ b & 0 \\ c & 0 \\ 0 & a \\ 0 & d \\ 0 & e \end{bmatrix}$

$C_{\mathrm{NB}}(P)$=90

$$\mathcal{M}_{124}^{(6)} = \begin{bmatrix} a & 0 & 0 \\ a & 0 & 0 \\ b & 0 & 0 \\ 0 & c & 0 \\ 0 & d & 0 \\ 0 & 0 & e \end{bmatrix}$$

$C_{\mathrm{NB}}(P)$=180

$$\mathcal{M}_{125}^{(6)} = \begin{bmatrix} a & 0 & 0 \\ b & 0 & 0 \\ c & 0 & 0 \\ 0 & a & 0 \\ 0 & d & 0 \\ 0 & 0 & e \end{bmatrix}$$

$C_{\mathrm{NB}}(P)$=360

$$\mathcal{M}_{126}^{(6)} = \begin{bmatrix} a & 0 & 0 \\ b & 0 & 0 \\ c & 0 & 0 \\ 0 & d & 0 \\ 0 & e & 0 \\ 0 & 0 & a \end{bmatrix}$$

$C_{\mathrm{NB}}(P)$=180

$$\mathcal{M}_{127}^{(6)} = \begin{bmatrix} b & 0 & 0 \\ c & 0 & 0 \\ d & 0 & 0 \\ 0 & a & 0 \\ 0 & a & 0 \\ 0 & 0 & e \end{bmatrix}$$

$C_{\mathrm{NB}}(P)$=60

$$\mathcal{M}_{128}^{(6)} = \begin{bmatrix} b & 0 & 0 \\ c & 0 & 0 \\ d & 0 & 0 \\ 0 & a & 0 \\ 0 & e & 0 \\ 0 & 0 & a \end{bmatrix}$$

$C_{\mathrm{NB}}(P)$=120

$$\mathcal{M}_{129}^{(6)} = \begin{bmatrix} a & 0 & 0 & 0 \\ a & 0 & 0 & 0 \\ b & 0 & 0 & 0 \\ 0 & c & 0 & 0 \\ 0 & 0 & d & 0 \\ 0 & 0 & 0 & e \end{bmatrix}$$

$C_{\mathrm{NB}}(P)$=60

$$\mathcal{M}_{130}^{(6)} = \begin{bmatrix} a & 0 & 0 & 0 \\ b & 0 & 0 & 0 \\ c & 0 & 0 & 0 \\ 0 & a & 0 & 0 \\ 0 & 0 & d & 0 \\ 0 & 0 & 0 & e \end{bmatrix}$$

$C_{\mathrm{NB}}(P)$=120

$$\mathcal{M}_{131}^{(6)} = \begin{bmatrix} b & 0 & 0 & 0 \\ c & 0 & 0 & 0 \\ d & 0 & 0 & 0 \\ 0 & a & 0 & 0 \\ 0 & 0 & a & 0 \\ 0 & 0 & 0 & e \end{bmatrix}$$

$C_{\mathrm{NB}}(P)$=60

$$\mathcal{M}_{132}^{(6)} = \begin{bmatrix} a & 0 & 0 & 0 & 0 \\ a & 0 & 0 & 0 & 0 \\ 0 & a & 0 & 0 & 0 \\ 0 & 0 & b & 0 & 0 \\ 0 & 0 & 0 & c & 0 \\ 0 & 0 & 0 & 0 & d \end{bmatrix}$$

$C_{\mathrm{NB}}(P)$=180

$$\mathcal{M}_{133}^{(6)} = \begin{bmatrix} a & 0 & 0 & 0 & 0 \\ b & 0 & 0 & 0 & 0 \\ 0 & a & 0 & 0 & 0 \\ 0 & 0 & a & 0 & 0 \\ 0 & 0 & 0 & c & 0 \\ 0 & 0 & 0 & 0 & d \end{bmatrix}$$

$C_{\mathrm{NB}}(P)$=60

$$\mathcal{M}_{134}^{(6)} = \begin{bmatrix} b & 0 & 0 & 0 & 0 \\ c & 0 & 0 & 0 & 0 \\ 0 & a & 0 & 0 & 0 \\ 0 & 0 & a & 0 & 0 \\ 0 & 0 & 0 & a & 0 \\ 0 & 0 & 0 & 0 & d \end{bmatrix}$$

$C_{\mathrm{NB}}(P)$=60

$$\mathcal{M}_{135}^{(6)} = \begin{bmatrix} a & 0 & 0 \\ a & 0 & 0 \\ 0 & b & 0 \\ 0 & c & 0 \\ 0 & 0 & d \\ 0 & 0 & e \end{bmatrix}$$

$C_{\mathrm{NB}}(P)$=180

$$\mathcal{M}_{136}^{(6)} = \begin{bmatrix} a & 0 & 0 \\ b & 0 & 0 \\ 0 & a & 0 \\ 0 & c & 0 \\ 0 & 0 & d \\ 0 & 0 & e \end{bmatrix}$$

$C_{\mathrm{NB}}(P)$=60

$$\mathcal{M}_{137}^{(6)} = \begin{bmatrix} a & 0 & 0 & 0 \\ a & 0 & 0 & 0 \\ 0 & b & 0 & 0 \\ 0 & c & 0 & 0 \\ 0 & 0 & d & 0 \\ 0 & 0 & 0 & e \end{bmatrix}$$

$C_{\mathrm{NB}}(P)$=$\frac{45}{2}$

$$\mathcal{M}_{138}^{(6)} = \begin{bmatrix} a & 0 & 0 & 0 \\ b & 0 & 0 & 0 \\ 0 & a & 0 & 0 \\ 0 & c & 0 & 0 \\ 0 & 0 & d & 0 \\ 0 & 0 & 0 & e \end{bmatrix}$$

$C_{\mathrm{NB}}(P)$=90

$$\mathcal{M}_{139}^{(6)} = \begin{bmatrix} a & 0 & 0 & 0 \\ b & 0 & 0 & 0 \\ 0 & c & 0 & 0 \\ 0 & d & 0 & 0 \\ 0 & 0 & a & 0 \\ 0 & 0 & 0 & e \end{bmatrix}$$

$C_{\mathrm{NB}}(P)$=90

$$\mathcal{M}_{140}^{(6)} = \begin{bmatrix} b & 0 & 0 & 0 \\ c & 0 & 0 & 0 \\ 0 & d & 0 & 0 \\ 0 & e & 0 & 0 \\ 0 & 0 & a & 0 \\ 0 & 0 & 0 & a \end{bmatrix}$$

$C_{\mathrm{NB}}(P)$=180

$$\mathcal{M}_{141}^{(6)} = \begin{bmatrix} a \\ b \\ c \\ d \\ e \\ f \end{bmatrix}$$

$C_{\mathrm{NB}}(P)$=360

$$\mathcal{M}_{142}^{(6)} = \begin{bmatrix} a & 0 \\ b & 0 \\ c & 0 \\ d & 0 \\ 0 & e \\ 0 & f \end{bmatrix}$$

$C_{\mathrm{NB}}(P)$=45

$$\mathcal{M}_{143}^{(6)} = \begin{bmatrix} a & 0 & 0 \\ b & 0 & 0 \\ c & 0 & 0 \\ d & 0 & 0 \\ 0 & e & 0 \\ 0 & 0 & f \end{bmatrix}$$

$C_{\mathrm{NB}}(P)$=$\frac{1}{6}$

$$\mathcal{M}_{144}^{(6)} = \begin{bmatrix} a & 0 \\ b & 0 \\ c & 0 \\ 0 & d \\ 0 & e \\ 0 & f \end{bmatrix}$$

$C_{\mathrm{NB}}(P)$=$\frac{5}{2}$

$C_{\mathrm{NB}}(P)$=$\frac{15}{2}$

$C_{\mathrm{NB}}(P)$=5

$$\mathcal{M}^{(6)}_{145} = \begin{bmatrix} a & 0 & 0 \\ b & 0 & 0 \\ c & 0 & 0 \\ 0 & d & 0 \\ 0 & e & 0 \\ 0 & 0 & f \end{bmatrix}$$

$C_{\mathrm{NB}}(P)$=30

$$\mathcal{M}^{(6)}_{146} = \begin{bmatrix} a & 0 & 0 & 0 \\ b & 0 & 0 & 0 \\ c & 0 & 0 & 0 \\ 0 & d & 0 & 0 \\ 0 & 0 & e & 0 \\ 0 & 0 & 0 & f \end{bmatrix}$$

$C_{\mathrm{NB}}(P)$=20

$$\mathcal{M}^{(6)}_{147} = \begin{bmatrix} a & 0 & 0 \\ b & 0 & 0 \\ 0 & c & 0 \\ 0 & d & 0 \\ 0 & 0 & e \\ 0 & 0 & f \end{bmatrix}$$

$C_{\mathrm{NB}}(P)=\frac{5}{2}$

$$\mathcal{M}^{(6)}_{148} = \begin{bmatrix} a & 0 & 0 & 0 \\ b & 0 & 0 & 0 \\ 0 & c & 0 & 0 \\ 0 & d & 0 & 0 \\ 0 & 0 & e & 0 \\ 0 & 0 & 0 & f \end{bmatrix}$$

$C_{\mathrm{NB}}(P)=\frac{45}{2}$

$$\mathcal{M}^{(6)}_{149} = \begin{bmatrix} a & 0 & 0 & 0 & 0 & 0 \\ 0 & b & 0 & 0 & 0 & 0 \\ 0 & 0 & c & 0 & 0 & 0 \\ 0 & 0 & 0 & d & 0 & 0 \\ 0 & 0 & 0 & 0 & e & 0 \\ 0 & 0 & 0 & 0 & 0 & f \end{bmatrix}$$

$C_{\mathrm{NB}}(P)$=1

$$\mathcal{M}^{(6)}_{150} = \begin{bmatrix} b & 0 & 0 & 0 \\ b & 0 & 0 & 0 \\ 0 & a & 0 & 0 \\ 0 & c & 0 & 0 \\ 0 & 0 & a & 0 \\ 0 & 0 & 0 & a \end{bmatrix}$$

$C_{\mathrm{NB}}(P)$=180

$$\mathcal{M}^{(6)}_{151} = \begin{bmatrix} a & 0 & 0 & 0 \\ b & 0 & 0 & 0 \\ 0 & b & 0 & 0 \\ 0 & c & 0 & 0 \\ 0 & 0 & a & 0 \\ 0 & 0 & 0 & a \end{bmatrix}$$

$C_{\mathrm{NB}}(P)$=360

$$\mathcal{M}^{(6)}_{152} = \begin{bmatrix} a & 0 & 0 \\ a & 0 & 0 \\ c & 0 & 0 \\ 0 & b & 0 \\ 0 & b & 0 \\ 0 & 0 & a \end{bmatrix}$$

$C_{\mathrm{NB}}(P)$=180

$$\mathcal{M}^{(6)}_{153} = \begin{bmatrix} a & 0 & 0 \\ b & 0 & 0 \\ 0 & a & 0 \\ 0 & b & 0 \\ 0 & 0 & c \\ 0 & 0 & d \end{bmatrix}$$

$C_{\mathrm{NB}}(P)$=90

$$\mathcal{M}^{(6)}_{154} = \begin{bmatrix} a & 0 & 0 & 0 \\ b & 0 & 0 & 0 \\ 0 & a & 0 & 0 \\ 0 & b & 0 & 0 \\ 0 & 0 & c & 0 \\ 0 & 0 & 0 & d \end{bmatrix}$$

$C_{\mathrm{NB}}(P)$=90

$$\mathcal{M}^{(6)}_{155} = \begin{bmatrix} a & 0 & 0 & 0 \\ b & 0 & 0 & 0 \\ 0 & c & 0 & 0 \\ 0 & d & 0 & 0 \\ 0 & 0 & a & 0 \\ 0 & 0 & 0 & b \end{bmatrix}$$

$C_{\mathrm{NB}}(P)$=180

$$\mathcal{M}^{(6)}_{156} = \begin{bmatrix} a & 0 & 0 & 0 & 0 \\ b & 0 & 0 & 0 & 0 \\ 0 & c & 0 & 0 & 0 \\ 0 & 0 & d & 0 & 0 \\ 0 & 0 & 0 & e & 0 \\ 0 & 0 & 0 & 0 & f \end{bmatrix}$$

$C_{\mathrm{NB}}(P)$=90

$$\mathcal{M}^{(6)}_{157} = \begin{bmatrix} a & 0 & 0 & 0 & 0 \\ a & 0 & 0 & 0 & 0 \\ 0 & b & 0 & 0 & 0 \\ 0 & 0 & c & 0 & 0 \\ 0 & 0 & 0 & d & 0 \\ 0 & 0 & 0 & 0 & e \end{bmatrix}$$

$C_{\mathrm{NB}}(P)$=15

$$\mathcal{M}^{(6)}_{158} = \begin{bmatrix} a & 0 & 0 & 0 & 0 \\ b & 0 & 0 & 0 & 0 \\ 0 & a & 0 & 0 & 0 \\ 0 & 0 & c & 0 & 0 \\ 0 & 0 & 0 & d & 0 \\ 0 & 0 & 0 & 0 & e \end{bmatrix}$$

$C_{\mathrm{NB}}(P)$=120

$$\mathcal{M}^{(6)}_{159} = \begin{bmatrix} b & 0 & 0 & 0 & 0 \\ c & 0 & 0 & 0 & 0 \\ 0 & a & 0 & 0 & 0 \\ 0 & 0 & a & 0 & 0 \\ 0 & 0 & 0 & d & 0 \\ 0 & 0 & 0 & 0 & e \end{bmatrix}$$

$C_{\mathrm{NB}}(P)$=90

APPENDIX B

COLUMN SIGNATURES AND THEIR LOCAL GENERATING FUNCTIONS

With $q_3 = \mu_3^2$ and $q_4 = \mu_4 - 3$, define

$$D(t) = 1 - q_4 t, \qquad E(t) = 1 + q_3 t, \qquad x(t) = \frac{t}{D(t)^3}.$$

These quantities were also defined in Section 3.1.

Table B.1: Documented column-signature patterns and their local generating functions.

Column pattern	Local generating function $G(c)$
`aaabbb`	$\dfrac{q_3 t}{E(t)}$
`00aaaa`	$\dfrac{q_4 t}{D(t)}$
`x00aaa`	$\dfrac{\mu_3 t}{E(t)D(t)}$
`xx0000`	$\dfrac{t}{D(t)^2}\left(\dfrac{1}{D(t)} - \dfrac{2q_3 t}{E(t)}\right)$
`xxaaaa`	$\dfrac{q_4 t}{D(t)}$
`xxxaaa`	$\dfrac{\mu_3 t}{E(t)}\left(\dfrac{1+2q_4 t}{D(t)}\right)$
`xxxx00`	$\dfrac{t}{D(t)}\left(1 - \dfrac{4q_3 t}{E(t)} - \dfrac{12 q_4 q_3 t^2}{D(t)E(t)} + \dfrac{6q_4 t}{D(t)} + \dfrac{3q_4^2 t^2}{D(t)^2}\right)$
`xxxxxx`	$t + \dfrac{15q_4 t^2}{D(t)} - \dfrac{10 q_3 t^2}{E(t)} + \dfrac{45 q_4^2 t^3}{D(t)^2} - \dfrac{60 q_4 q_3 t^3}{E(t)D(t)} + \dfrac{15 q_4^3 t^4}{D(t)^3} - \dfrac{90 q_4^2 q_3 t^4}{E(t)D(t)^2}$

APPENDIX C

THE GILBERT–VARSHAMOV BOUND FOR BINARY LINEAR CODES

Proof of Theorem 12.2.16. Let $\delta = k/m$. Let r be any integer with $r \leq (1 - H(\delta))\, m$. We show by the probabilistic method that there is a binary linear code of block length m, dimension at least r, and minimum distance at least $k+1$. Indeed, pick a generator matrix $M \in \mathbb{F}_2^{m \times r}$ uniformly at random, i.e. each of the mr entries of M is chosen independently to be 0 or 1 with probability $1/2$. This M defines a linear code

$$C \;=\; \{Mx : x \in \mathbb{F}_2^r\}.$$

We want to ensure that no nonzero message x is mapped to a codeword Mx with Hamming weight $\mathrm{wt}(Mx) \leq k$, so that $\mathrm{wt}(Mx) \geq k+1$ for every $x \neq 0$.

Fix a nonzero message $x \in \mathbb{F}_2^r$. Since M was chosen at random, Mx is uniformly distributed in $\mathbb{F}_2^m$. Thus

$$\Pr_M\big[\mathrm{wt}(Mx) \leq k\big] \;=\; \frac{\big|\{y \in \mathbb{F}_2^m : \mathrm{wt}(y) \leq k\}\big|}{2^m}.$$

The number of $\mathbf{y} \in \mathbb{F}_2^m$ with $\mathrm{wt}(y) \leq k$ can be bounded above by

$$\sum_{i=0}^{k} \binom{m}{i} \;\leq\; 2^{mH(\delta)}.$$

Hence

$$\Pr_M\big[\mathrm{wt}(Mx) \leq k\big] \leq 2^{-m}\, 2^{mH(\delta)} = 2^{-m(1-H(\delta))}.$$

Next, apply the union bound over all nonzero messages $x \in \mathbb{F}_2^r \setminus \{\mathbf{0}\}$. There are $2^r - 1$

such messages, so

$$\Pr_{M}\Big[\exists\, x \neq \mathbf{0} \text{ with } \operatorname{wt}(Mx) \leq k\Big] \;\leq\; (2^r - 1)\, 2^{-m(1-H(\delta))}.$$

Since $r \;\leq\; (1 - H(\delta))\, m$, we have $2^r \leq 2^{m(1-H(\delta))}$. It follows that the above probability is at most

$$\big(2^r - 1\big)\, 2^{-m(1-H(\delta))} \;<\; 1.$$

Hence there is at least one choice of M (one generator matrix) under which every nonzero message maps to a codeword with weight greater than k. In other words, that choice of M generates a linear code of dimension r whose minimum (Hamming) distance is at least $k + 1$. Since r was any integer up to $\lfloor (1 - H(\delta))\, m \rfloor$, the theorem is proved. $\square$

REFERENCES

[AAK+07] Noga Alon, Alexandr Andoni, Tali Kaufman, Kevin Matulef, Ronitt Rubinfeld, and Ning Xie. Testing k-wise and almost k-wise independence. In *Proceedings of the 39th Annual Symposium on Theory of Computing (STOC)*, pages 496–505, 2007.

[ABN+92] Noga Alon, Jehoshua Bruck, Joseph Naor, Moni Naor, and Ron M. Roth. Construction of asymptotically good low-rate error-correcting codes through pseudo-random graphs. *IEEE Transactions on Information Theory*, 38(2):509–516, 1992.

[ACR97] Alexander E. Andreev, Andrea E. F. Clementi, and José D. P. Rolim. Efficient constructions of hitting sets for systems of linear functions. In *Proceedings of the 14th Annual Symposium on Theoretical Aspects of Computer Science (STACS)*, pages 387–398, 1997.

[AGHP92] Noga Alon, Oded Goldreich, Johan Håstad, and René Peralta. Simple constructions of almost k-wise independent random variables. *Random Structures Algorithms*, 3(3):289–304, 1992.

[AGM03] Noga Alon, Oded Goldreich, and Yishay Mansour. Almost k-wise independence versus k-wise independence. *Inform. Process. Lett.*, 88(3):107–110, 2003.

[AKK16] Andris Ambainis, Martins Kokainis, and Robin Kothari. Nearly Optimal Separations Between Communication (or Query) Complexity and Partitions. In *Proceedings of the 31st Conference on Computational Complexity (CCC)*, pages 4:1–4:14, 2016.

[Alo86] N. Alon. Explicit construction of exponential sized families of k-independent sets. *Discrete Math.*, 58(2):191–193, 1986.

[AN21] Sepehr Assadi and Vishvajeet N. Graph streaming lower bounds for parameter estimation and property testing via a streaming XOR lemma. In *Proceedings of the 53rd Annual ACM SIGACT Symposium on Theory of Computing (STOC)*, pages 612–625, 2021.

[BCG20] Mark Braverman, Gil Cohen, and Sumegha Garg. Pseudorandom pseudo-distributions with near-optimal error for read-once branching programs. *SIAM J. Comput.*, 49(5):STOC18–242–STOC18–299, 2020.

[BDVY13] Andrej Bogdanov, Zeev Dvir, Elad Verbin, and Amir Yehudayoff. Pseudorandomness for width-2 branching programs. *Theory Comput.*, 9:283–292, 2013.

[Bec23] Dominik Beck. On the fourth moment of a random determinant. *Random Matrices: Theory and Applications*, 2023.

[BHPP22] Andrej Bogdanov, William M. Hoza, Gautam Prakriya, and Edward Pyne. Hitting Sets for Regular Branching Programs. In *Proceedings of the 37th Computational Complexity Conference (CCC)*, pages 3:1–3:22, 2022.

[BLP23] Dominik Beck, Zelin Lv, and Aaron Potechin. The sixth moment of random determinants. *Journal of Integer Sequences*, 26(6), 2023. Article 23.6.3.

[BLP24] Dominik Beck, Zelin Lv, and Aaron Potechin. On the second moment of the determinant of random symmetric, wigner, and hermitian matrices. 2024. arXiv:2409.14620.

[BLP25] Dominik Beck, Zelin Lv, and Aaron Potechin. Analysing the moments of the determinant of a random matrix via analytic combinatorics of permutation tables. 2025. arXiv:2507.03651.

[BLP26] Dominik Beck, Zelin Lv, and Aaron Potechin. The sixth moment of random determinants for arbitrarily distributed random entries. 2026. arXiv:2607.26857.

[Bor18] Michael Borinsky. Generating asymptotics for factorially divergent sequences. *Electronic Journal of Combinatorics*, 25(4), 2018. Paper P4.1.

[BRRY14] Mark Braverman, Anup Rao, Ran Raz, and Amir Yehudayoff. Pseudorandom generators for regular branching programs. *SIAM J. Comput.*, 43(3):973–986, 2014.

[BS88] Bernd Becker and Hans-Ulrich Simon. How robust is the n-cube? *Inform. and Comput.*, 77(2):162–178, 1988.

[Bsh14] Nader H. Bshouty. Testers and their applications [extended abstract]. In *Proceedings of the 5th Conference on Innovations in Theoretical Computer Science (ITCS)*, pages 327–351. ACM, New York, 2014.

[Bsh16] Nader H. Bshouty. Derandomizing chernoff bound with union bound with an application to k-wise independent sets, 2016.

[BV10a] Andrej Bogdanov and Emanuele Viola. Pseudorandom bits for polynomials. *SIAM J. Comput.*, 39(6):2464–2486, April 2010.

[BV10b] Joshua Brody and Elad Verbin. The coin problem and pseudorandomness for branching programs. In *2010 IEEE 51st Annual Symposium on Foundations of Computer Science*, pages 30–39, 2010.

[CCD+26] Ben Chen, Gil Cohen, Dean Doron, Yuval Khaskelberg, and Amnon Ta-Shma. Improved error reduction for weighted PRGs. ECCC preprint TR26-064, revision 3, 2026.

[CH22] Kuan Cheng and William M. Hoza. Hitting sets give two-sided derandomization of small space. *Theory Comput.*, 18:Paper No. 21, 32, 2022.

[CHL+23] Lijie Chen, William M. Hoza, Xin Lyu, Avishay Tal, and Hongxun Wu. Weighted pseudorandom generators via inverse analysis of random walks and shortcutting. In *Proceedings of the 64th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 1224–1239, 2023.

[CK16] Ruiwen Chen and Valentine Kabanets. Correlation bounds and #SAT algorithms for small linear-size circuits. *Theoret. Comput. Sci.*, 654:2–10, 2016.

[CKMZ83] Ashok K. Chandra, Lawrence T. Kou, George Markowsky, and Shmuel Zaks. On sets of Boolean n-vectors with all k-projections surjective. *Acta Inform.*, 20(1):103–111, 1983.

[CL24] Eshan Chattopadhyay and Jyun-Jie Liao. Recursive Error Reduction for Regular Branching Programs. In *15th Innovations in Theoretical Computer Science Conference (ITCS)*, pages 29:1–29:20, 2024.

[CRSW13] L. Elisa Celis, Omer Reingold, Gil Segev, and Udi Wieder. Balls and bins: smaller hash families and faster evaluation. *SIAM J. Comput.*, 42(3):1030–1050, 2013.

[CT25] Ben Chen and Amnon Ta-Shma. Better weighted pseudorandom generators against low weight read-once branching programs. ECCC preprint TR25-067, 2025.

[CTV06] Kevin P. Costello, Terence Tao, and Van H. Vu. Random symmetric matrices are almost surely nonsingular. *Duke Mathematical Journal*, 135(2):395–413, 2006.

[CW79] J. Lawrence Carter and Mark N. Wegman. Universal classes of hash functions. *J. Comput. System Sci.*, 18(2):143–154, 1979.

[CW26] Kuan Cheng and Ruiyang Wu. Weighted pseudorandom generators for read-once branching programs via weighted pseudorandom reductions. In *Proceedings of the 37th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 3423–3458, 2026.

[De11] Anindya De. Pseudorandomness for permutation and regular branching programs. In *26th Annual Conference on Computational Complexity (CCC)*, pages 221–231, 2011.

[Dem89] Amir Dembo. On random determinants. *Quarterly of Applied Mathematics*, 47(2):185–195, 1989.

[DH25] Dean Doron and William M. Hoza. Implications of better PRGs for permutation branching programs. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2025)*, pages 28:1–28:20, 2025.

[FG15] Michael A. Forbes and Venkatesan Guruswami. Dimension Expanders via Rank Condensers. In *Proceedings of the 19th International Workshop on Randomization and Computation (RANDOM)*, pages 800–814, 2015. preprint: https://arxiv.org/abs/1411.7455.

[For51] Robert Fortet. Random determinants. *Journal of Research of the National Bureau of Standards*, 47:465–470, 1951.

[FS09] Philippe Flajolet and Robert Sedgewick. *Analytic Combinatorics*. Cambridge University Press, 2009.

[FT52] George E. Forsythe and John W. Tukey. The extent of n random unit vectors. *Bulletin of the American Mathematical Society*, 58:502, 1952.

[Gir80] V. L. Girko. The central limit theorem for random determinants. *Theory of Probability and Its Applications*, 24(4):729–740, 1980.

[GK09] Friedrich Götze and Holger Kösters. On the second-order correlation function of the characteristic polynomial of a hermitian wigner matrix. *Communications in Mathematical Physics*, 285:1183–1205, 2009.

[GKM18] Parikshit Gopalan, Daniel M. Kane, and Raghu Meka. Pseudorandomness via the discrete Fourier transform. *SIAM J. Comput.*, 47(6):2451–2487, 2018.

[GKST18] Alexander Golovnev, Alexander S. Kulikov, Alexander V. Smal, and Suguru Tamaki. Gate elimination: circuit size lower bounds and #SAT upper bounds. *Theoret. Comput. Sci.*, 719:46–63, 2018.

[GMR+12] Parikshit Gopalan, Raghu Meka, Omer Reingold, Luca Trevisan, and Salil Vadhan. Better pseudorandom generators from milder pseudorandom restrictions. In *Proceedings of the 53rd Annual Symposium on Foundations of Computer Science (FOCS)*, pages 120–129, 2012.

[GPW18] Mika Göös, Toniann Pitassi, and Thomas Watson. Deterministic communication vs. partition number. *SIAM J. Comput.*, 47(6):2435–2450, 2018.

[HH24] Pooya Hatami and William Hoza. Paradigms for unconditional pseudorandom generators. *Found. Trends Theor. Comput. Sci.*, 16(1-2):1–210, 2024.

[HHV+24] Itamar Harel, William M. Hoza, Gal Vardi, Itay Evron, Nathan Srebro, and Daniel Soudry. Provable tempered overfitting of minimal nets and typical nets, 2024.

[HL25a] William M. Hoza and Zelin Lv. Fooling near-maximal decision trees. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2025)*, volume 353 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 35:1–35:24, 2025.

[HL25b] William M. Hoza and Zelin Lv. On sums of inw pseudorandom generators. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2025)*, volume 353 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 67:1–67:24, 2025.

[Hoz21] William M. Hoza. Better Pseudodistributions and Derandomization for Space-Bounded Computation. In *Proceedings of the 25th International Conference on Randomization and Computation (RANDOM)*, pages 28:1–28:23, 2021.

[Hoz22] William M. Hoza. Recent progress on derandomizing space-bounded computation. *Bull. Eur. Assoc. Theor. Comput. Sci. EATCS*, (138):114–143, 2022.

[HPV21] William M. Hoza, Edward Pyne, and Salil Vadhan. Pseudorandom Generators for Unbounded-Width Permutation Branching Programs. In *12th Innovations in Theoretical Computer Science Conference (ITCS)*, pages 7:1–7:20, 2021.

[HPV24] William M. Hoza, Edward Pyne, and Salil Vadhan. Limitations of the Impagliazzo–Nisan–Wigderson pseudorandom generator against permutation branching programs. *Algorithmica*, 2024.

[Hub64] Peter J. Huber. Robust estimation of a location parameter. *Ann. Math. Statist.*, 35:73–101, 1964.

[HZ20] William M. Hoza and David Zuckerman. Simple optimal hitting sets for small-success **RL**. *SIAM J. Comput.*, 49(4):811–820, 2020.

[INW94] Russell Impagliazzo, Noam Nisan, and Avi Wigderson. Pseudorandomness for network algorithms. In *Proceedings of the 26th Annual Symposium on Theory of Computing (STOC)*, pages 356–364, 1994.

[KKS95] Jeff Kahn, János Komlós, and Endre Szemerédi. On the probability that a random ± 1-matrix is singular. *Journal of the American Mathematical Society*, 8(1):223–240, 1995.

[KM93] Eyal Kushilevitz and Yishay Mansour. Learning decision trees using the Fourier spectrum. *SIAM J. Comput.*, 22(6):1331–1348, 1993.

[KM97] Howard Karloff and Yishay Mansour. On construction of k-wise independent random variables. *Combinatorica*, 17(1):91–107, 1997.

[KNP11] Michal Koucký, Prajakta Nimbhorkar, and Pavel Pudlák. Pseudorandom generators for group products [extended abstract]. In *Proceedings of the 43rd ACM Symposium on Theory of Computing (STOC)*, pages 263–272, 2011.

[KRDS15] Robin Kothari, David Racicot-Desloges, and Miklos Santha. Separating decision tree complexity from subcube partition complexity. In *Proceedings of the 19th International Workshop on Randomization and Computation (RANDOM)*, pages 915–930, 2015.

[KS73] Daniel J. Kleitman and Joel Spencer. Families of k-independent sets. *Discrete Math.*, 6:255–262, 1973.

[Lov09] Shachar Lovett. Unconditional pseudorandom generators for low-degree polynomials. *Theory of Computing*, 5(3):69–82, 2009.

[LPV23] Chin Ho Lee, Edward Pyne, and Salil Vadhan. On the Power of Regular and Permutation Branching Programs. In *Proceedings of the 27th International Conference on Randomization and Computation (RANDOM)*, pages 44:1–44:22, 2023.

[LRTV09] Shachar Lovett, Omer Reingold, Luca Trevisan, and Salil Vadhan. Pseudorandom bit generators that fool modular sums. In *Proceedings of the 13th International Workshop on Randomization and Computation (RANDOM)*, pages 615–630, 2009.

[LT09] Shachar Lovett and Yoav Tzur. Explicit lower bound for fooling polynomials by the sum of small-bias generators. *Electronic Colloquium on Computational Complexity (ECCC)*, (TR09-088), 2009.

[LV17] Chin Ho Lee and Emanuele Viola. Some limitations of the sum of small-bias distributions. *Theory of Computing*, 13(16):1–23, 2017.

[Lv23] Zelin Lv. *On the Moments of Random Determinants*. University of Chicago, 2023. Master's thesis.

[MN98] Madan Lal Mehta and Jean-Marie Normand. Probability density of the determinant of a random hermitian matrix. *Journal of Physics A: Mathematical and General*, 31(23):5377–5391, 1998.

[MNT93] Yishay Mansour, Noam Nisan, and Prasoon Tiwari. The computational complexity of universal hashing. *Theoretical Computer Science*, 107(1):121–133, 1993.

[MRT19] Raghu Meka, Omer Reingold, and Avishay Tal. Pseudorandom generators for width-3 branching programs. In *Proceedings of the 51st Annual Symposium on Theory of Computing (STOC)*, pages 626–637, 2019.

[MZ09] Raghu Meka and David Zuckerman. Small-bias spaces for group products. In *Proceedings of the 13th International Workshop on Randomization and Computation (RANDOM)*, pages 658–672, 2009.

[MZ13] Raghu Meka and David Zuckerman. Pseudorandom generators for polynomial threshold functions. *SIAM J. Comput.*, 42(3):1275–1301, 2013.

[Nis92] Noam Nisan. Pseudorandom generators for space-bounded computation. *Combinatorica*, 12(4):449–461, 1992.

[NN93] Joseph Naor and Moni Naor. Small-bias probability spaces: efficient constructions and applications. *SIAM J. Comput.*, 22(4):838–856, 1993.

[NRR54] Harry Nyquist, S. O. Rice, and John Riordan. The distribution of random determinants. *Quarterly of Applied Mathematics*, 12(2):97–104, 1954.

[NSS95] Moni Naor, Leonard J. Schulman, and Aravind Srinivasan. Splitters and near-optimal derandomization. In *Proceedings of 36th Annual Conference on Foundations of Computer Science (FOCS)*, pages 182–191, 1995.

[NV14] Hoi H. Nguyen and Van H. Vu. Random matrices: Law of the determinant. *Annals of Probability*, 42(1):146–167, 2014.

[NZ96] Noam Nisan and David Zuckerman. Randomness is linear in space. *J. Comput. System Sci.*, 52(1):43–52, 1996.

[OZ18] Ryan O'Donnell and Yu Zhao. On Closeness to k-Wise Uniformity. In *Proceedings of the 22nd International Conference on Randomization and Computation (RANDOM)*, pages 54:1–54:19, 2018.

[Pré67] András Prékopa. On random determinants i. *Studia Scientiarum Mathematicarum Hungarica*, 2:125–132, 1967.

[PRZ23] Edward Pyne, Ran Raz, and Wei Zhan. Certified hardness vs. randomness for log-space. In *Proceedings of the 64th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 989–1007, 2023.

[Sav02] Petr Savický. On determinism versus unambiquous nondeterminism for decision trees. ECCC preprint TR02-009, 2002.

[SB88] Gadiel Seroussi and Nader H. Bshouty. Vector sets for exhaustive testing of logic circuits. *IEEE Trans. Inform. Theory*, 34(3):513–522, 1988.

[Shp09] Amir Shpilka. Constructions of low-degree and error-correcting ε-biased generators. *Computational Complexity*, 18(4):495, 2009.

[Sko22] Maciej Skorski. Tight Chernoff-Like Bounds Under Limited Independence. In *Proceedings of the 26th International Conference on Randomization and Computation (RANDOM)*, pages 15:1–15:14, 2022.

[SSS95] Jeanette P. Schmidt, Alan Siegel, and Aravind Srinivasan. Chernoff-Hoeffding bounds for applications with limited independence. *SIAM J. Discrete Math.*, 8(2):223–250, 1995.

[Ste12] Thomas Steinke. Pseudorandomness for permutation branching programs without the group theory. ECCC preprint TR12-083, 2012.

[SVW17] Thomas Steinke, Salil Vadhan, and Andrew Wan. Pseudorandomness and Fourier-growth bounds for width-3 branching programs. *Theory Comput.*, 13:Paper No. 12, 50, 2017.

[SZ95] Michael Saks and David Zuckerman. Unpublished manuscript. Unpublished, 1995.

[Tik20] Konstantin Tikhomirov. Singularity of random bernoulli matrices. *Annals of Mathematics*, 191(2):593–634, 2020.

[Tur55] Pál Turán. On a problem in the theory of determinants. *Acta Mathematica Academiae Scientiarum Hungaricae*, 5:417–423, 1955.

[TV06] Terence Tao and Van H. Vu. On random ± 1 matrices: Singularity and determinant. *Random Structures & Algorithms*, 28(1):1–23, 2006.

[TW83] Donald T. Tang and Lin S. Woo. Exhaustive test pattern generation with constant weight vectors. *IEEE Transactions on Computers*, C-32(12):1145–1150, 1983.

[Vad12] Salil P. Vadhan. Pseudorandomness. *Foundations and Trends in Theoretical Computer Science*, 7(1–3):1–336, 2012.

[Vio09] Emanuele Viola. The sum of d small-bias generators fools polynomials of degree d. *Comput. Complexity*, 18(2):209–217, 2009.

[Vu20] Van H. Vu. Recent progress in combinatorial random matrix theory. *Probability Surveys*, 17:49–110, 2020.

[WC81] Mark N. Wegman and J. Lawrence Carter. New hash functions and their use in authentication and set equality. *J. Comput. System Sci.*, 22(3):265–279, 1981. Special issue dedicated to Michael Machtey.

[Zhu68] Igor G. Zhurbenko. Moments of random determinants. *Theory of Probability and Its Applications*, 13(4):682–686, 1968.