Western Graduate&PostdoctoralStudies

Western University
Scholarship@Western

Electronic Thesis and Dissertation Repository

4-5-2019 9:30 AM

Algorithms for Bohemian Matrices

Steven E. Thornton, *The University of Western Ontario*

Supervisor: Corless, Robert M., *The University of Western Ontario*
Co-Supervisor: Moreno Maza, Marc, *The University of Western Ontario*
A thesis submitted in partial fulfillment of the requirements for the Doctor of Philosophy degree in Applied Mathematics
© Steven E. Thornton 2019

Follow this and additional works at: https://ir.lib.uwo.ca/etd

Part of the Other Applied Mathematics Commons

Recommended Citation
Thornton, Steven E., "Algorithms for Bohemian Matrices" (2019). *Electronic Thesis and Dissertation Repository*. 6069.
https://ir.lib.uwo.ca/etd/6069

This Dissertation/Thesis is brought to you for free and open access by Scholarship@Western. It has been accepted for inclusion in Electronic Thesis and Dissertation Repository by an authorized administrator of Scholarship@Western. For more information, please contact wlswadmin@uwo.ca.

Abstract

This thesis develops several algorithms for working with matrices whose entries are multivariate polynomials in a set of parameters. Such *parametric linear systems* often appear in biology and engineering applications where the parameters represent physical properties of the system. Some computations on parametric matrices, such as the rank and Jordan canonical form, are discontinuous in the parameter values. Understanding where these discontinuities occur provides a greater understanding of the underlying system.

Algorithms for computing a complete case discussion of the rank, *Zigzag form*, and the Jordan canonical form of parametric matrices are presented. These algorithms use the theory of regular chains to provide a unified framework allowing for algebraic or semi-algebraic constraints on the parameters. Corresponding implementations for each algorithm in the MAPLE computer algebra system are provided.

In some applications, all entries may be parameters whose values are limited to finite sets of integers. Such matrices appear in applications such as graph theory where matrix entries are limited to the sets $\{0, +1\}$, or $\{-1, 0, +1\}$. These types of parametric matrices can be explored using different techniques and exhibit many interesting properties.

A *family of Bohemian matrices* is a set of low to moderate dimension matrices where the entries are independently sampled from a finite set of integers of bounded height. Properties of Bohemian matrices are studied including the distributions of their eigenvalues, symmetries, and integer sequences arising from properties of the families. These sequences provide connections to other areas of mathematics and have been archived in the Characteristic Polynomial Database. A study of two families of structured matrices: upper Hessenberg and upper Hessenberg Toeplitz, and properties of their characteristic polynomials are presented.

Keywords: Parametric matrices, Jordan canonical form, Frobenius form, rational form, Zigzag form, matrix rank, regular chains, parametric linear systems, Bohemian matrices, random matrices, eigenvalues, upper Hessenberg, Toeplitz, rhapsodic matrices, Characteristic Polynomial Database.

Co-Authorship Statement

This integrated-article thesis is based on 5 papers. Chapter 2 has been submitted for publication, and Chapters 3, and 4 are based on the papers [1], and [2]. For these chapters, Marc Moreno Maza provided assistance with the theoretical understanding behind regular chains, and Rob Corless provided assistance with finding applications of the work. A version of Chapter 5 is being prepared for publication. Rob Corless provided feedback on the paper. A version of Chapter 6 has been submitted for publication. The initial basis for this chapter was developed by Rob Corless, Laurenao Gonzalez-Vega, Rafael Sendra, and Juana Sendra. Eunice Chan provided assistance with compiling the prior work related to the paper and she generalized work on similar matrices in Theorem 6.5.10. The work in Sections 6.10 and 6.11 was completed by Rob Corless.

Bibliography

[1] R. M. Corless, M. Moreno Maza, and S. E. Thornton. Zigzag form over families of parametric matrices. *ACM Communications in Computer Algebra*, 48(3/4):109–112, Feb 2015.

[2] R. M. Corless, M. Moreno Maza, and S. E. Thornton. Jordan canonical form with parameters from Frobenius form with parameters. In *Proceeding of The International Conference on Mathematical Aspects of Computer and Information Sciences 2017*, pages 179–194. Springer, 2017.

Acknowledgements

I am grateful for the support of many people who without, this thesis may have never reached completion.

First, without my Ph.D. supervisors, Rob Corless and Marc Moreno Maza, this thesis would not have been possible. I thank Rob for giving me the opportunity to pursue graduate school under his supervision. The knowledge Rob has shared with me over the past 6 years has been invaluable. I thank Marc for his generosity in sharing his expertise of regular chains and aiding in the development of several key components of thesis.

Next, I thank my parents, Linda and Scott Thornton, for their constant support throughout graduate school. Their enthusiasm has kept me going through the course of my thesis. I am thankful for my partner, Emily Cozens, for helping me stay grounded throughout graduate school.

I would like to thank my current employer, Rafael Nicolas Fermin Cota, for his flexibility in allowing me to complete my Ph.D. while concurrently working full time.

After several years of dedication to expanding my knowledge of mathematics, I am happy to be contributing this thesis to the mathematics community.

Contents

List of Figures

List of Tables

Abbreviations

CAS Computer algebra system

CPDB Characteristic polynomial database

GCD Greatest common divisor

JCF Jordan canonical form

OEIS Online Encyclopedia of Integer Sequences

Chapter 1

Introduction

1.1 Parametric Linear Systems

Linear systems are a universal tool in mathematics with their use spanning nearly all applications. Many applications contain parameters that may be unknown quantities, or approximate values found through experimentation. The values the parameters take may lead to a significant difference in the meaning of the underlying system. Understanding the influence of the parameter values on the underlying system is of great interest. Solving linear systems with parameters has been studied extensively with early work by Sit in [2]. Significantly less work has been done on computing other properties of these systems such as the distribution of rank as a function of the parameters, or the possible Jordan canonical forms (JCF).

Many computer algebra systems (CAS) struggle with these types of problems. Asking for the rank of a parametric matrix in Maple for example will return a generic solution assuming the parameter values are transcendental numbers. For example, for the matrix

$$A = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & \alpha \end{bmatrix}$$

where $\alpha \in \mathbb{C}$, Maple will compute the rank to be 3. If we specialize $\alpha = 9$, we find that the rank is 2. Even worse, in Maple, asking for the JCF of a parametric matrix whose characteristic polynomial cannot be factored such that all irreducible terms are of degree less than 5 will simply fail to provide a solution, see Figure 1.1 for example. These problems appear to be universal across CAS. Sometimes the implementations will warn the user that the answer may not be correct for all parameter values. In the Sage CAS for example, a user is warned when computing the JCF of a parametric matrix and the generic solution is provided. Solving parametric linear systems have been more successful in those CAS with many specialized packages developed for working with these systems.

```
> with(LinearAlgebra) :
> A := CompanionMatrix(x^5 + x^4 + x^3 + x^2 + x + α, x)
```

$$A := \begin{bmatrix} 0 & 0 & 0 & 0 & -\alpha \\ 1 & 0 & 0 & 0 & -1 \\ 0 & 1 & 0 & 0 & -1 \\ 0 & 0 & 1 & 0 & -1 \\ 0 & 0 & 0 & 1 & -1 \end{bmatrix}$$

```
> JordanForm(A)
```

$$JordanForm\left(\begin{bmatrix} 0 & 0 & 0 & 0 & -\alpha \\ 1 & 0 & 0 & 0 & -1 \\ 0 & 1 & 0 & 0 & -1 \\ 0 & 0 & 1 & 0 & -1 \\ 0 & 0 & 0 & 1 & -1 \end{bmatrix}\right)$$

Figure 1.1: The `JordanForm` function in the `LinearAlgebra` library in MAPLE fails to compute the Jordan form of a matrix with a single parameter. This example was run in MAPLE 2018.

In this thesis, methods for analyzing matrices with entries that are multivariate polynomials in a set of parameters are developed. Initially motivated by the failures of MAPLE when computing the JCF of a parametric matrix, several algorithms are developed including one for the JCF where the input matrix is in Frobenius form and contains polynomial entries. The complexity of these problems is high when parameters are present. As such, the algorithms are typically limited to small systems with few parameters. They may succeed on larger systems but this success is dependent on the linear system. A MAPLE package called `ParametricMatrixTools` has been developed to share the algorithms with the greater mathematics community. These algorithms have been developed using the theory of regular chains [1] and the `ParametricMatrixTools` package has been built on top of the `RegularChains` package.

Some applications contain matrices where all entries are parameters. Further, such parameter values may be restricted to belonging to small sets of integers. Such matrices appear in graph theory where the entries are restricted to the sets $\{0, +1\}$, or $\{-1, 0, +1\}$. Since the entries are restricted to small sets, different approaches may be used for the analysis of these systems. In some cases, there may be a small enough set of distinct matrices that all matrices can be exhaustively explored. Relationships within these families can further reduce the computation required for many properties. Exploring these families of parametric matrices turns out to be a very interesting problem on its own.

1.2 Bohemian Matrices

Low dimension square integer matrices are commonly used in introductory linear algebra curricula for teaching the fundamental concepts. These matrices are often used as examples for analyzing linear systems including solving the system, and computing its eigenvalues. Despite their simplicity, many questions remain about these low dimension matrices. For example, how many singular 6×6 matrices with entries from the set $\{-1, 0, +1\}$ exist? To date the answer is unknown. While the computation for a single matrix is simple, and is not outside of the scope of what an undergraduate student should be able to compute, the difficulty comes from the number of such matrices. For this example there are 3^{36} = 150,094,635,296,999,121 matrices, most of which have likely never appeared on a linear algebra exam. Even with modern computing power, questions like these still remain outside the scope of what can be computed on standard hardware. Since the number of matrices grows exponentially in the square of the dimension, computational hardware will never be able to make much progress on these problems.

The study of **Bohemian matrices** focuses on answering questions about distributions of low dimension integer matrices with entries of bounded height. A **Bohemian family** is a distribution of Bohemian matrices where the **population** is the set of integers the entries are sampled from. Inspiration for studying these types of problems originated when exploring density plots of the eigenvalues of such types of random matrices. Discrete structures appear in the eigenvalue densities that do not have obvious explanations, see Figure 1.2 for example.

The exploration of new Bohemian families typically begins with plotting the density of the eigenvalues in the complex plane. To ease this exploration phase, a Matlab framework was developed to assist with generating mathematically accurate plots. This framework has been made available at `https://github.com/BohemianMatrices/BHIME-Project`.

Specializing to Bohemian families where the matrices are structured (e.g. upper Hessenberg, Toeplitz, circulant, etc.) has shown to be more successful in developing an understanding of relationships within these families. With these special structures, existing work on such structured matrices can be extended by restricting the entries to belong to a small population of integers. Further, brute force exploration can help identify patterns within these families.

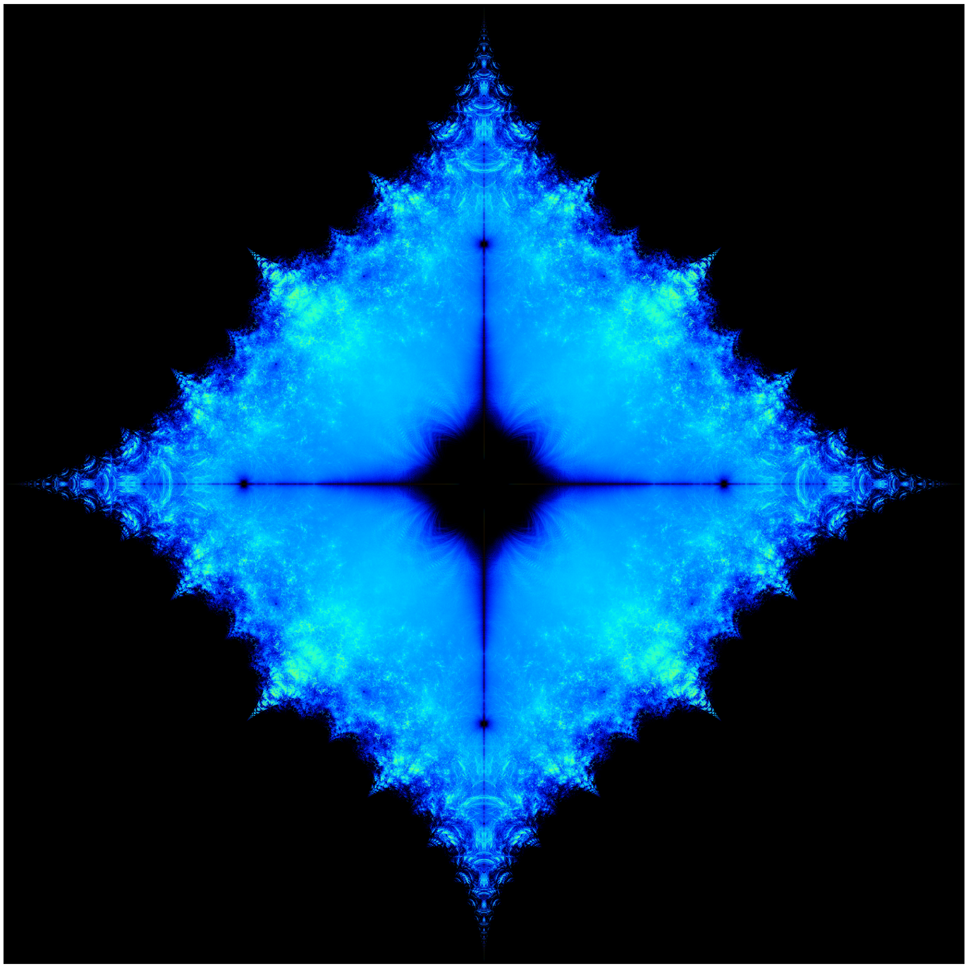


Figure 1.2: Density of the eigenvalues of a sample of 100 million tridiagonal matrices with entries sampled from the set $\{-1, +1\}$ with entries on the main diagonal fixed at 0. The figure is viewed over the complex range $-2 - 2i$ to $2 + 2i$.

Brute force computation over Bohemian families can also be used to find the distributions of characteristic polynomials. In many families, the size of the set of characteristic polynomials is substantially smaller than the set of matrices. Thus, for some questions, working with the set of characteristic polynomials can be easier than with the family of Bohemian matrices. For example, the distribution of determinants within a family can be read directly from the characteristic polynomials. Questions like this have inspired the development of the **Characteristic Polynomial Database** (CPDB). The CPDB provides distributions of the characteristic polynomials for families of Bohemian matrices. The database is publicly available at `http://www.bohemianmatrices.com/cpdb/` and currently contains 1,762,728,065 characteristic polynomials from 2,366,960,967,336 matrices.

1.3 Outline

This thesis begins with three chapters on algorithms for parametric matrices. In Chapters 2 through 4, algorithms for computing the rank, Zigzag form, Frobenius (rational) form and Jordan form of parametric matrices are discussed. Chapters 5 and 6 focus on Bohemian matrices. Chapter 5 provides a general discussion of Bohemian matrices followed by a detailed study of a specific family of Bohemian matrices in Chapter 6.

Chapter 2 presents an algorithm for computing the rank of a parametric matrix as a function of the parameters while avoiding explicitly solving the corresponding parametric linear system. As input, this algorithm takes a matrix with multivariate polynomial entries whose indeterminates are regarded as parameters and are subject to a system of polynomial equations and inequalities. The algorithm relies on the theory of regular chains. An implementation of the algorithm in the Maple computer algebra system is presented, which has been built on top of the `RegularChains` library. The effectiveness of the implementation is demonstrated by comparing it to a naïve implementation and by using it to find the rank of several examples from the literature.

In Chapter 3, an algorithm for computing the *Zigzag form* of a parametric matrix as a function of the parameters is presented. This work was motivated by a desire to compute the Frobenius (rational) canonical form of a parametric matrix. By first computing the Zigzag form, the Frobenius form can be obtained by GCD computations. The algorithm for the constant case has been taken from [3] and has $\mathcal{O}(n^3)$ complexity. This algorithm has been modified to provide a full case discussion for matrices with parameters.

Chapter 4 introduces an algorithm for computing the JCF of a parametric matrix that is in Frobenius form. The algorithm takes as input a matrix in Frobenius canonical form where the entries are multivariate polynomials in the parameters and computes a complete case discussion for the JCF. The JCF of a square matrix is a foundational tool in matrix analysis. If the matrix $\mathbf{A}$ is known exactly, symbolic computation of the JCF is possible though expensive. When the matrix contains parameters, exact computation requires either a potentially very expensive case discussion, significant expression swell, or both. For this reason, no current computer algebra system will compute a case discussion for the JCF of a matrix $\mathbf{A}(\alpha)$ where α is a (vector of) parameter(s). This problem is extremely difficult in general, even though the JCF is encountered early in most curricula. The algorithm presented is based on the theory of regular chains and an implementation built on the `RegularChains` library in Maple is discussed.

Chapter 5 addresses some of the interesting features of Bohemian families including symmetries, distribution of eigenvalues, and integer sequences for related properties. A

Matlab framework for visualizing distributions of eigenvalues is presented and used as an experimental tool for understanding discrete structures found in these distributions. While developing the framework, two families of Bohemian matrices were found where the Matlab eigenvalue solver fails to produce solutions in some instances. The techniques used for computing the properties and characteristic polynomials found in the Characteristic Polynomial Database are introduced.

Chapter 6 explores a special family of Bohemian matrices, specifically those with entries from the set $\{-1, 0, +1\}$. More, these matrices are specialized to be upper Hessenberg, with sub-diagonal entries ± 1. Many properties remain after these specializations, some of which were surprising. Two recursive formulae for the characteristic polynomials of upper Hessenberg matrices are given. Focusing on only those matrices whose characteristic polynomials have maximal height allows us to explicitly identify these polynomials and give a lower bound on their height. This bound is exponential in the order of the matrix. We count stable matrices, normal matrices, and neutral matrices, and tabulate the results of our experiments. We prove a theorem about the only possible kinds of normal matrices amongst a specific family of Bohemian upper Hessenberg matrices.

Bibliography

[1] P. Aubry, D. Lazard, and M. Moreno Maza. On the theories of triangular sets. *Journal of Symbolic Computation*, 28(1-2):105–124, 1999.

[2] W. Y. Sit. An algorithm for solving parametric linear systems. *Journal of Symbolic Computation*, 13(4):353–394, 1992.

[3] A. Storjohann. An $\mathcal{O}(n^3)$ algorithm for the Frobenius normal form. In *Proceedings of the 1998 International Symposium on Symbolic and Algebraic Computation*, pages 101–105. ACM, 1998.

Chapter 2

Comprehensive Rank Computation for Matrices Depending on Parameters

2.1 Introduction

Determining the rank of a matrix is a simple computation traditionally presented in introductory linear algebra courses. Unfortunately, the computation for parametric matrices is a tedious process which, to our knowledge, does not yet have a completely satisfactory solution. In this paper we present an algorithmic approach to extending the methods of rank computation to parametric matrices with polynomial entries. External equality and inequality constraints on the parameters may be inherited in the problem being solved and will be considered in the computations. Additionally, we present an implementation of our algorithm in the MAPLE computer algebra system (CAS).

For an $m \times n$ matrix $A(\alpha)$, where the parameters α are subject to a system of polynomial constraints S, our rank computation proceeds as follows. We compute the null space of $A(\alpha)$ by means of a triangular decomposition of the polynomial system S' obtained by adding to S the equations of $A(\alpha)X = 0$, where X is a column vector of unknowns $x_1, \ldots, x_n$. By means of set-theoretic operations on algebraic or semi-algebraic sets, we deduce a decomposition of the parameter space into cells $C_0, \ldots, C_n$ such that above C_r, for all $0 \leq r \leq n$ the rank of $A(\alpha)$ is equal to r. The proposed method is, in fact, stated for both algebraic and semi-algebraic constraints. This feature is achieved thanks to the theory of regular chains which provides us with a unified framework, reviewed in Section 2.2, for solving polynomial systems over both the complex and the real numbers.

In addition, the proposed method is tailored to the problem of parametric matrix rank computation. That is, we avoid the usage of general tools for solving parametric polynomial systems, such as *comprehensive Gröbner bases* [22], *comprehensive triangular decomposition* [8], or *dynamic evaluation* [4, 12]. In fact, we rely on the non-comprehensive triangular decomposition algorithms presented in [9] and [6] for the complex and real cases, respectively.

Our approach is presented in Section 2.4, following two lemmas established in Section 2.3. We implemented our algorithms in the MAPLE CAS. Section 4.6 reports on the successful application of our implementation to various examples taken from the literature. In addition, the experimental part of our work revealed the importance of a tailored method, that is, a method avoiding general tools for solving parametric polynomial systems. Indeed, a preliminary implementation, based on comprehensive triangular decomposition, was generating much more complex output and was substantially slower than the method presented in Section 2.4.

Works related to this paper include polynomial eigenvalue problems [18, 21] which are sub-problems of the question studied in this paper. In addition, control theory, where the rank of a real matrix can be used to determine whether a linear system is controllable, or observable, is an important area of applications for parametric matrix rank computation. Extensive work has been done on solving parametric linear systems [2, 11, 13, 17], with some of the earliest work done by William Sit [20].

2.2 Preliminaries

The algebraic material reviewed below supports the algorithm presented in Section 2.4. The notion of a *regular chain*, introduced independently in [16] and [23], is closely related to that of a triangular decomposition of a polynomial system. Broadly speaking, a *triangular decomposition* of a polynomial system S is a set of simpler (in a precise sense) polynomial systems $S_1, \ldots, S_e$ such that a point p is a solution of S if, and only if, p is a solution of (at least) one of the systems $S_1, \ldots, S_e$.

When the purpose is to describe all the solutions of S, whether their coordinates are real numbers or not, in which case S is said to be *algebraic*, those simpler systems are required to be regular chains. We refer to [1, 9] for a formal presentation on the concepts of a regular chain and a triangular decomposition of an algebraic system.

If the coefficients of S are real numbers and only the real solutions are required, (in which case S is said to be *semi-algebraic*), then those real solutions can be obtained by a triangular decomposition into so-called *regular semi-algebraic systems*, a notion introduced in [6]. In both cases, each of these simpler systems has a triangular shape and remarkable properties, which justifies the terminology.

Multivariate polynomials. Let $\mathbb{K}$ be a field. If $\mathbb{K}$ is an ordered field, then we assume that it is a real closed field like the field $\mathbb{R}$ of real numbers. Otherwise, we assume that $\mathbb{K}$ is algebraically closed, like the field $\mathbb{C}$ of complex numbers. Let $X_1 < \cdots < X_s$ be $s \geq 1$ ordered variables. We denote by $\mathbb{K}[X_1, \ldots, X_s]$ the ring of polynomials in the variables $X_1, \ldots, X_s$ with coefficients in $\mathbb{K}$. For a non-constant polynomial $p \in \mathbb{K}[X_1, \ldots, X_s]$, the greatest variable in p is called the *main variable* of p, denoted by $\mathrm{mvar}(p)$, and the leading coefficient of p w.r.t. $\mathrm{mvar}(p)$ is called the *initial* of p, denoted by $\mathrm{init}(p)$.

Regular chains. A set R of non-constant polynomials in $\mathbb{K}[X_1, \ldots, X_s]$ is called a *triangular set*, if for all $p,q \in R$ with $p \neq q$ we have $\mathrm{mvar}(p) \neq \mathrm{mvar}(q)$. A variable X_i is said to be *free* w.r.t. R if there exists no $p \in R$ such that $\mathrm{mvar}(p) = X_i$. For a nonempty triangular set R, we define the *saturated ideal* $\mathrm{sat}(R)$ of R to be the ideal $\langle R \rangle : h_R^{\infty}$, where h_R is the product of the initials of the polynomials in R. The saturated ideal of the empty triangular set is defined as the trivial ideal $\langle 0 \rangle$. From now on, R denotes a triangular set of $\mathbb{K}[X_1, \ldots, X_s]$. The ideal $\mathrm{sat}(R)$ has several properties, and in particular it is unmixed [3]. We denote its height, that is, the number of polynomials in R, by e, thus $\mathrm{sat}(R)$ has dimension $s - e$. Let $X_{i_1} < \cdots < X_{i_e}$ be the main variables of the polynomials in R. We denote by r_j the polynomial of R whose main variable is X_{i_j} and by h_j the initial of r_j. Thus h_R is the product $h_1 \cdots h_e$. We say that R is a *regular chain* whenever R is empty or, $\{r_1, \ldots, r_{e-1}\}$ is a regular chain and h_e is regular modulo the saturated ideal $\mathrm{sat}(\{r_1, \ldots, r_{e-1}\})$.

Constructible sets. Let $F \subset \mathbb{K}[X_1, \ldots, X_s]$ be a set of polynomials and $g \in \mathbb{K}[X_1, \ldots, X_s]$ be a polynomial. We denote by $V(F) \subseteq \mathbb{K}^s$ the *zero set* or *affine variety* of F, that is, the set of points in the affine space $\mathbb{K}^s$ at which every polynomial $f \in F$ vanishes. If F consists of a single polynomial f, we write $V(f)$ instead of $V(F)$. We call a *constructible set* any subset of $\mathbb{K}^s$ of the form $V(F) \setminus V(g)$. Let $R \subset \mathbb{K}[X_1, \ldots, X_s]$ be a regular chain and let $h \in \mathbb{K}[X_1, \ldots, X_s]$ be a polynomial. We say that the pair $[R, h]$ is a *regular system* whenever h is regular modulo $\mathrm{sat}(R)$ and $V(h_R) \subseteq V(h)$ holds. We write $Z(R,h)$ for $V(R) \setminus V(h)$. One should observe that for a regular system $[R, h]$ the zero set $Z(R,h)$ is necessarily not empty. Regular systems provide an encoding for constructible sets. More precisely, there exists a finite family $\mathcal{T}$ of regular systems $[R_1, h_1], \ldots, [R_e, h_e]$ of $\mathbb{K}[X_1, \ldots, X_s]$ such that

$$V(F) \setminus V(g) \; = \; Z(R_1, h_1) \, \cup \, \cdots \, \cup \, Z(R_e, h_e).$$

We call $\mathcal{T}$ a *triangular decomposition* of the constructible set $V(F) \setminus V(g)$.

In the sequel of this section, we assume that $\mathbb{K}$ is a real closed field.

Regular semi-algebraic systems. A *regular semi-algebraic system* of $\mathbb{K}[X_1, \ldots, X_s]$ is a triple $[T, Q, P]$ where $T \subset \mathbb{K}[X_1, \ldots, X_s]$ is a regular chain, Q is a quantifier-free formula involving only the free variables of T and P is a set of positive inequalities defined by polynomials of $\mathbb{K}[X_1, \ldots, X_s]$; moreover $[T, Q, P]$ must satisfy the following properties:

- (i) Q defines a non-empty open set in the space of the free variables of T;
- (ii) at any point α defined by Q, the product h_T of the initials of T does not vanish, the specialized regular chain T_α generates a radical ideal and, each specialized polynomial in P_α is invertible modulo the ideal generated by T_α;
- (iii) at any point α defined by Q, the specialized semi-algebraic system $[T_\alpha, P_\alpha]$ admits at least one real solution β, that is, every polynomial in T_α is zero at β, and every polynomial in P_α is positive at β.

We denote by $Z(T, Q, P)$ the set of the points in the affine space $\mathbb{K}^s$ simultaneously satisfying the quantifier-free formula Q, the equation $f = 0$ for each $f \in T$ and, each of the inequalities of P.

Semi-algebraic sets. We call a *semi-algebraic system* of $\mathbb{K}[X_1, \ldots, X_s]$ any polynomial system S of the form

$$f_1 = \cdots = f_a = 0, g \neq 0, p_1 > 0, \ldots, p_b > 0, q_1 \geq 0, \ldots, q_c \geq 0,$$

where $f_1, \ldots, f_a, g, p_1, \ldots, p_b, q_1, \ldots, q_c$ are polynomials of $\mathbb{K}[X_1, \ldots, X_s]$. The *solution set* S consists of all points in the affine space $\mathbb{K}^s$ satisfying simultaneously the above constraints. We call a *semi-algebraic set* any subset of $\mathbb{K}^s$ which is the solution set of a semi-algebraic system of $\mathbb{K}[X_1, \ldots, X_s]$. Regular semi-algebraic systems provide an encoding for semi-algebraic sets. More precisely, there exists a finite family $\mathcal{T}$ of regular semi-algebraic systems $[T_1, Q_1, P_1], \ldots, [T_e, Q_e, P_e]$ of $\mathbb{K}[X_1, \ldots, X_s]$ such that we have

$$S \;=\; Z(T_1, Q_1, P_1) \;\cup\; \cdots \;\cup\; Z(T_e, Q_e, P_e).$$

We call $\mathcal{T}$ a *triangular decomposition* of the semi-algebraic set S. Examples are provided in Appendix 2.6. An important property of any regular semi-algebraic system $[T, Q, P]$ is the fact that it is a parametrization of its zero set. Therefore, a triangular decomposition of a semi-algebraic system S decomposes the zero set of S into components, each of which is given by a parametric representation. This encoding of a semi-algebraic set is very useful to compute geometrical quantities such as dimension.

Encoding constructible sets (resp. semi-algebraic sets) with regular systems (resp. regular semi-algebraic systems) has another benefit. It leads to efficient algorithms for performing set-theoretic operations on constructible and semi-algebraic sets; see [8] and [7] respectively. These operations, as well as the above mentioned triangular decomposition algorithms, are part of the `RegularChains` library [5, 19] distributed with the Maple CAS. In Section 2.4, our algorithm refers to the operations `Triangularize`, `RealTriangularize` and `Difference` of the `RegularChains` library. The first two operations compute a triangular decomposition of a constructible set and a semi-algebraic set, respectively. The latter applies to a couple (A,B), of either constructible sets or semi-algebraic sets, and returns the set-theoretic difference $A \setminus B$.

2.3 Lemmas

We use the same notations as in Section 2.2. In addition, we consider $k \geq 1$ ordered variables $\alpha_1 < \cdots < \alpha_k$ that we shall view as parameters. Let $A(\alpha) = A(\alpha_1, \ldots, \alpha_k)$ be an $m \times n$ matrix with coefficients in $\mathbb{K}[\alpha_1, \ldots, \alpha_k]$.

If $\mathbb{K}$ is a real closed field, we assume that $\alpha_1, \ldots, \alpha_k$ are subject to a semi-algebraic system S defined by polynomials of $\mathbb{K}[\alpha_1, \ldots, \alpha_k]$. We denote by $\Sigma \subseteq \mathbb{K}^k$ the semi-algebraic set defined by S. If $\mathbb{K}$ is algebraically closed, we assume that $\alpha_1, \ldots, \alpha_k$ are subject to an algebraic system that we denote by S and which is defined by polynomials of $\mathbb{K}[\alpha_1, \ldots, \alpha_k]$. We denote by $\Sigma \subseteq \mathbb{K}^k$ the corresponding constructible set.

Our aim is to compute the rank of $A(\alpha)$ for all $\alpha \in \Sigma$. More precisely, we aim at decomposing Σ into cells $C_0, \ldots, C_n$ such that the rank of $A(\alpha)$ is r for all $\alpha \in C_r$, for $0 \leq r \leq n$.

Let X be an n-element column vector whose entries are ordered variables $x_1, \ldots, x_n$ satisfying $\alpha_1 < \cdots < \alpha_k < x_1 < \cdots < x_n$. Denote by Π the standard projection from $\mathbb{K}^{k+n}$ onto the space of the least k coordinates. We consider the polynomial system S' obtained by adding to S the equations of $A(\alpha)X = 0$. These are equations given by polynomials of $\mathbb{K}[\alpha_1 < \cdots < \alpha_k < x_1 < \cdots < x_n]$. Let $\mathcal{T}$ be a triangular decomposition of the zero set of S'. The following two lemmas state respectively in the complex and real cases a key property which allows us to deduce from $\mathcal{T}$ a case discussion for the computation of the null space of $A(\alpha)$. This will be used in Section 2.4 in order to obtain the desired parametric rank computation.

Lemma 2.3.1. *If $\mathbb{K}$ is algebraically closed, then $\mathcal{T}$ is a finite family of regular systems $[T_1, h_1], \ldots, [T_e, h_e]$ of $\mathbb{K}[\alpha_1, \ldots, \alpha_k, x_1, \ldots, x_n]$ such that the following properties hold:*

(*i*) *each polynomial in each regular chain* $T_1, \ldots, T_e$ *has degree zero or one w.r.t. each of the variables* $x_1, \ldots, x_n$*;*

(*ii*) *each polynomial* h_i *belongs to* $\mathbb{K}[\alpha_1, \ldots, \alpha_k]$*;*

(*iii*) *for each* $1 \leq i \leq e$*, the projection* $\Pi(Z(T_i, h_i))$ *is given by* $Z(T_i \cap \mathbb{K}[\alpha_1, \ldots, \alpha_k], h_i)$*, that is,*

$$\Pi^{-1}(\Pi(Z(T_i, h_i))) = Z(T_i \cap \mathbb{K}[\alpha_1, \ldots, \alpha_k], h_i),$$

thus, $\Pi(Z(T_i, h_i))$ *is obtained by "erasing" from* $[T_i, h_i]$ *those polynomials where at least one of the variables* $x_1, \ldots, x_n$ *appears.*

Proof. We first prove (i). Since variables are ordered as $\alpha_1 < \cdots < \alpha_k < x_1 < \cdots < x_n$ and since the input polynomials have degree zero or one w.r.t. each of the variables $x_1, \ldots, x_n$, the triangular decomposition algorithm of [9] (which relies on polynomial GCD and resultant computations) generates polynomials which all have degree zero or one w.r.t. each of the variables $x_1, \ldots, x_n$. This observation implies (i). Next, we prove (ii). Following again the triangular decomposition algorithm of [9], each of the polynomials $h_1, \ldots, h_e$ comes either from the input system S' or, is a factor of an initial or, a factor of a resultant computed by the triangular decomposition algorithm of [9]. It follows from (i) that each of $h_1, \ldots, h_e$ necessarily belongs to $\mathbb{K}[\alpha_1, \ldots, \alpha_k]$. Finally, we prove ($iii$). Let $[R_i, h_i]$ be any of the regular systems of $\mathcal{T}$. Let β be a point in the parameter space. Since h_i does not involve any of the variables $x_1, \ldots, x_n$ the inequation $h_i(\beta) \neq 0$ makes sense. Since $h_i(\beta) \neq 0$ implies that none of the initials of T_i vanishes at β, the conditions

$$h_i(\beta) \neq 0 \ \text{ and } \ f(\beta) = 0 \ (\forall f \in T_i \cap \mathbb{K}[\alpha_1, \ldots, \alpha_k]),$$

are sufficient for β to be extended to a zero of $Z(T_i, h_i)$. The conclusion follows. ♮

Lemma 2.3.2. *If* $\mathbb{K}$ *is a real closed field, then* $\mathcal{T}$ *is a finite family of regular semi-algebraic systems* $[T_1, Q_1, P_1], \ldots, [T_e, Q_e, P_e]$ *of* $\mathbb{K}[\alpha_1, \ldots, \alpha_k, x_1, \ldots, x_n]$ *such that the following properties hold:*

(1) *each polynomial in each regular chain* $T_1, \ldots, T_e$ *has degree zero or one w.r.t. each of the variables* $x_1, \ldots, x_n$*;*

(2) *each set of polynomial inequalities* P_i *is empty;*

(3) *for each* $i = 1 \cdots e$*, we have*

$$\Pi^{-1}(\Pi(Z(T_i, Q_i, P_i))) \ = \ Z(T_i \cap \mathbb{K}[\alpha_1, \ldots, \alpha_k], Q_i, \varnothing).$$

Proof. A first step is to compute a triangular decomposition $\mathcal{T}_{\overline{\mathbb{K}}}$ over the algebraic closure of $\mathbb{K}$ of the system S'' consisting only of the equations of S'. (See Line 1 of Algorithm 2 in [6].) Lemma 2.3.1 applies to S'' and $\mathcal{T}_{\overline{\mathbb{K}}}$. Hence $\mathcal{T}_{\overline{\mathbb{K}}}$ consists of regular systems $[T_1, h_1], \ldots, [T_e, h_e]$ satisfying properties (i), (ii) and (iii) of Lemma 2.3.1. A second step is to refine $\mathcal{T}_{\overline{\mathbb{K}}}$ (still over the algebraic closure of $\mathbb{K}$) by using the inequations and inequalities of S' as inequations. (See Lines 2 to 15 of Algorithm 2 in [6].) Since S' has no inequations or inequalities involving (at least one of) the variables $x_1 < \cdots < x_n$, we can still assume that, after this second step, we have a triangular decomposition consisting of regular systems $[T_1, h_1], \ldots, [T_e, h_e]$ satisfying properties (i), (ii) and (iii) of Lemma 2.3.1. A third and final step is, for each regular system $[T_i, h_i]$, to check whether or not it has real solutions and, if yes, to generate the quantifier free Q_i such that the regular semi-algebraic system $[T_i, Q_i, \varnothing]$ describes those real solutions (See Lines 16 to 19 of Algorithm 2 together with Algorithms 3, 5 and 6 in [6].) One should observe that each Q_i may contain inequalities. However, those inequalities involve the parameters $\alpha_1, \ldots, \alpha_k$ only. Claims (1) and (2) follow from the above observations. Finally, Claim (3) follows from Claims (1) and (2) and the properties of a regular semi-algebraic system. ♮

Lemmas 2.3.1 and 2.3.2 imply that the Π-projections of the zero sets of the regular systems (resp. regular semi-algebraic systems) of the triangular decomposition $\mathcal{T}$ decompose the constructible set (resp. semi-algebraic set) Σ into cells $B_0, \ldots, B_e$ above which the solutions of the parametric linear system $A(\alpha)X = 0$ is given by one of the regular systems in $\mathcal{T}$. However, this does not yet solve our parametric rank computation problem. Indeed, the solution set of $A(\alpha)X = 0$ above a cell B_i might be contained into the solution set of $A(\alpha)X = 0$ above another cell B_j, for some $0 \leq i < j \leq e$. In fact, dealing with redundant components is a well-known issue in all types of algorithms for decomposing polynomial systems. This difficulty is handled in Section 2.4 by a post-processing of the triangular decomposition $\mathcal{T}$.

2.4 Algorithm

Reusing the notations of Section 2.3, recall that $\mathcal{T}$ is a triangular decomposition of the zero set of the system S' obtained by adding to S the equations of $A(\alpha)X = 0$, where S is a polynomial system on the parameters of the $m \times n$ parametric matrix $A(\alpha)$. The following procedure computes a decomposition of the zero set $\Sigma \subseteq \mathbb{K}^k$ of S into cells $C_0, C_1, \ldots, C_n$ such that for all $0 \leq r \leq n$ and all $\alpha^* \in C_i$ the rank of the specialized matrix $A(\alpha^*)$ is r. We make use of the commands of the `RegularChains` library specified in Section 2.2. Assume first that $\mathbb{K}$ is algebraically closed.

Step 1: Let $\mathcal{T} :=$ `Triangularize`$(S', \mathbb{K}[\alpha_1 < \cdots < \alpha_k < x_1 < \cdots < x_n])$

Step 2: For $0 \leq r \leq n$, let C_r be the constructible set of $\mathbb{K}^k$ given by all regular systems $[T_j \cap \mathbb{K}[\alpha_1 < \cdots < \alpha_k], h_j]$ such that $[T_j, h_j] \in \mathcal{T}$ and the number of polynomials of T_j of positive degree in (at least) one of the variables $x_1 < \cdots < x_n$ is exactly r.

Step 3: For $r := n$ down to 1 do

$$C_r \ := \ \texttt{Difference}(C_r, C_{r-1} \cup \cdots \cup C_0)$$

Now, we state the algorithm for the case where $\mathbb{K}$ is real closed.

Step 1: Let $\mathcal{T} :=$ `RealTriangularize`$(S', \mathbb{K}[\alpha_1 < \cdots < \alpha_k < x_1 < \cdots < x_n])$

Step 2: For $0 \leq r \leq n$, let C_r be the semi-algebraic set of $\mathbb{K}^k$ given by all regular semi-algebraic systems $[T_j \cap \mathbb{K}[\alpha_1 < \cdots < \alpha_k], Q_j, \varnothing]$ such that $[T_j, Q_j, \varnothing] \in \mathcal{T}$ and the number of polynomials of T_j of positive degree in (at least) one of the variables $x_1 < \cdots < x_n$ is exactly r.

Step 3: For $r := n$ down to 1 do

$$C_r \ := \ \texttt{Difference}(C_r, C_{r-1} \cup \cdots \cup C_0)$$

Theorem 2.4.1. *Whether $\mathbb{K}$ is algebraically closed or real closed, the above procedure satisfies the claimed specification.*

Proof. Let $0 \leq r \leq n$ and let $\alpha^* \in C_i$. By virtue of Lemmas 2.3.1 and 2.3.2, the point α^* can be extended to a solution of a regular chain with n polynomials of positive degree in (at least) one of the variables $x_1 < \cdots < x_n$. Thus, the null space of $A(\alpha^*)$ has dimension at most $n - r$. Using the fact that the cells $C_0, C_1, \ldots, C_n$ are pairwise disjoint (this property is achieved by **Step 3**) it follows from the rank-nullity theorem that the rank of $A(\alpha^*)$ is exactly r. ♮

2.5 Implementation

Computing the rank of matrices depending on parameters is only one of many computations on matrices with parameters we are considering. We are developing a MAPLE packaged called `ParametricMatrixTools` for computations on matrices containing parameters [10]. The implementations in our package are based on the theory of regular chains and build on the `RegularChains` package in MAPLE. The source including examples of the main procedures of the package is available at `https://github.com/steventhornton/ParametricMatrixTools`. The `ComprehensiveRank` and `RealComprehensiveRank` routines implement the algorithms discussed in Section 2.4 and are applied on the examples that follow.

Our implementations include some heuristics that aim to improve the computation time. The first heuristic we use applies to non-square matrices. When computing the rank of an $m \times n$ matrix where $n > m$, the transpose is taken as this typically results in a speed improvement. This improvement is a consequence of the triangular decomposition computation in Step 1 of our algorithm. Computing a triangular decomposition of n equations equations which are linear in the largest m variables $x_1, \ldots, x_m$ for $n > m$ is less expensive than when $m > n$. This improvement is illustrated in Figure 2.1.

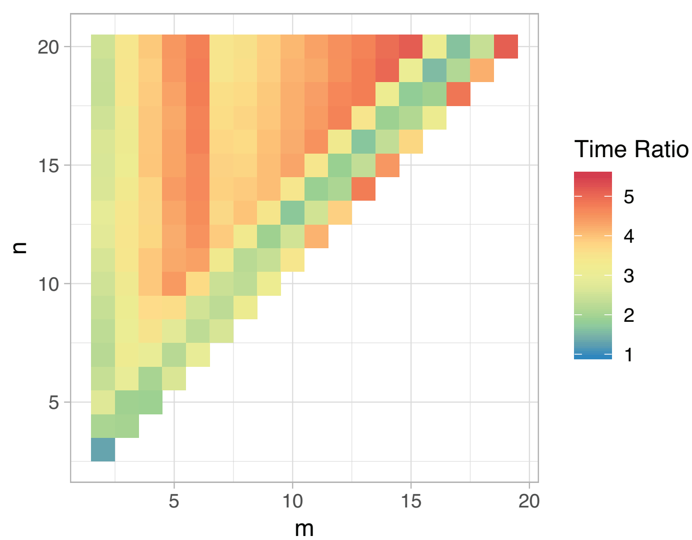


Figure 2.1: If A is an $m \times n$ matrix, the color represents the ratio of the time to compute the rank of A over the time to compute the rank of A^T. The ratio plotted is an average of the ratios for 100 randomly generated $m \times n$ matrices with integer entries between -10 and 10, and between 1 and 5 entries containing parameters with at most 5 unique parameters. For each matrix sampled, the minimum time from 10 iterations is taken.

The second heuristic we introduce uses the `SuggestVariableOrder` function from the `RegularChains` package to determine an order for the linear variables $x_1, \ldots, x_n$ that is expected to speed up the triangular decomposition (Step 1 in the algorithms from Section 2.4). The parameters are excluded from the resulting suggested variable ordering and they remain in the order given as input to the `ComprehensiveRank` or `RealComprehensiveRank` functions such that they are all less than the linear variables.

The MAPLE scripts used for all the examples and timing below are available on GitHub at `https://github.com/steventhornton/Comprehensive_Rank_Computation_for_Matrices_Depending_on_Parameters`.

2.5.1 Comparison With Other Implementations

Despite extensive literature on solving parametric linear systems and their corresponding implementations, we have been unable to find any algorithms or implementations that provide the full decomposition of the rank of a parametric linear system. To illustrate the effectiveness of our algorithm, we compare it with a naive implementation based on the computation of a comprehensive triangular decomposition of the linear system.

Our naive algorithm will compute a decomposition of the zero set $\Sigma \subseteq \mathbb{K}^k$ of S into cells $D_0, D_1, \ldots, D_n$ such that for all $0 \leq r \leq n$ and all $\alpha^* \in D_i$ the rank of the specialized matrix $A(\alpha^*)$ is r. This algorithm is included in the `ParametricMatrixTools` package and can be used by calling the `ComprehensiveRank` procedure with the `algorithm=ctd` option. As in Section 2.4, let S' be the polynomial system obtained by adding to S the equations of $A(\alpha)X = 0$. Our algorithm follows the notation of [8] for a comprehensive triangular decomposition.

Step 1: Let the comprehensive triangular decomposition of $S' \subset \mathbb{K}[\alpha_1, \ldots, \alpha_k, x_1, \ldots, x_k]$ be given by the pair $(\mathcal{T}_C, C \in \mathcal{C})$ for $\mathcal{C} = \Pi_\alpha(V(S'))$ where $\alpha = \alpha_1 < \cdots < \alpha_k$.

Step 2: For $0 \leq r \leq n$, let D_r be the union of all constructible sets $C \in \mathcal{C}$ such that no regular chain $T \in \mathcal{T}_C$ contains less than r polynomials in $x_1, \ldots, x_n$, and at least one $T \in \mathcal{T}_C$ contains exactly r polynomials in $x_1, \ldots, x_n$.

Both the implementation of the algorithm discussed here and our naive implementation were tested on a corpus of 540 parametric matrices generated by Ballarin and Kauers for their paper on solving parametric linear systems [2]. The corpus is available at `https://github.com/steventhornton/corpus-of-parametric-linear-systems`[1]. The 540 parametric matrices are all square matrices ranging in size from 4×4 to 6×6 with polynomial entries containing between 0 and 3 parameters, of total degree between 2 and 10, between 0 and 12 symbolic entries, and between 0 and 12 zero entries.

[1]The corpus was originally available at `http://www21.in.tum.de/~ballarin/data/c540/` but appears to have been removed as of early 2018.

Our experiment ran each of the 540 examples 25 times on each implementation for a maximum time of 10 minutes. We take the fastest time from the 25 runs as the execution time for each example. All timings were run with Maple 2017 on an AMD Ryzen Threadripper 1950X with 64Gb of RAM.

Of the 540 examples, 45 contained no parameters and are excluded from our comparison. Of the remaining 495 examples, there were 120 that neither implementation was able to complete in less than 10 minutes. Of the 375 examples that at least one of the implementations completed in under 10 minutes and contained parameters, our algorithm computed the rank faster than the naive algorithm in 372 cases. The other 3 examples were slower by only 1ms. Figure 2.2 compares our algorithm with the naive version across several properties of the example matrices.

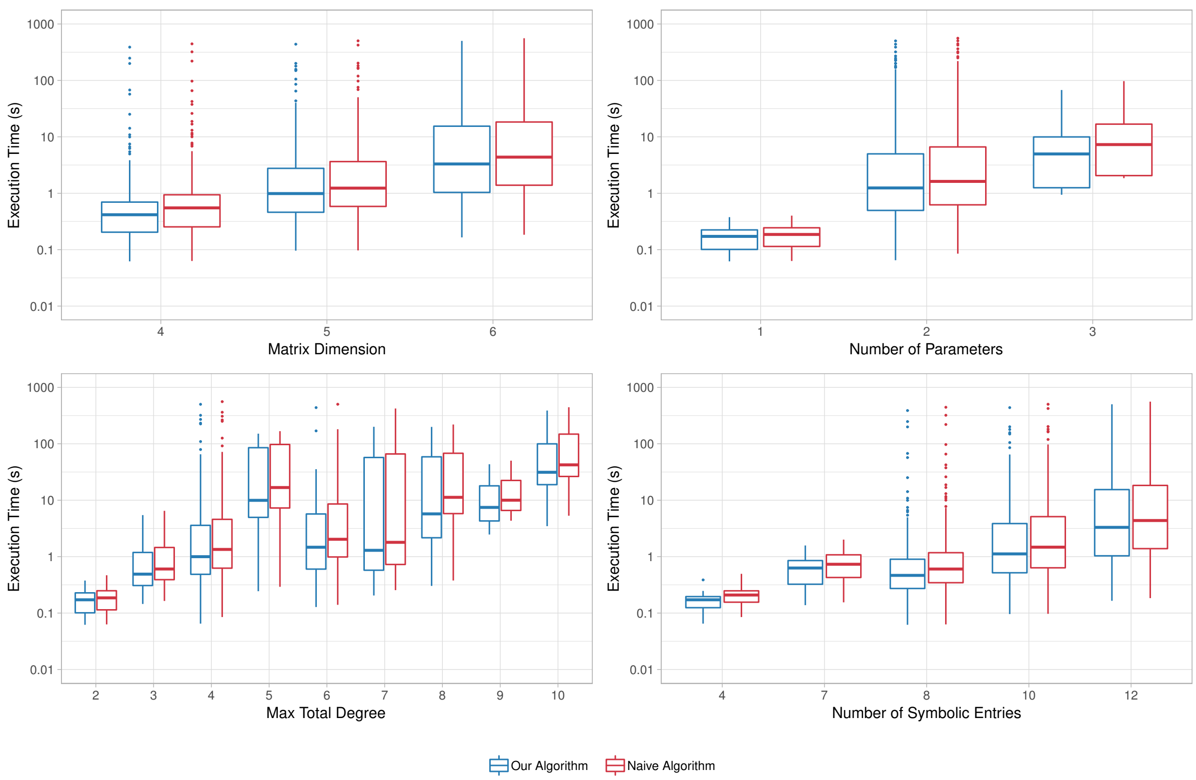


Figure 2.2: Bar chart of the execution times for computing the rank of 495 matrices for our algorithm compared to a naive version.

2.5.2 Example 1

Taking an example from [15] from control theory, we look for the conditions on the parameters such that the matrix is full rank. When it is full rank we know we have a controllable system.

$$E = \begin{pmatrix} 1 & 3 & 1 \\ 3 & 1 & 1 \\ 0 & 0 & 0 \end{pmatrix} A_1 = \begin{pmatrix} 1 & 1 & 3 \\ 1 & 3 & 1 \\ 0 & 0 & 0 \end{pmatrix}, A_2 = \begin{pmatrix} \lambda & 3\lambda & \lambda \\ 3\lambda + \mu & \lambda + \mu & \lambda + 3\mu \\ 0 & 0 & 0 \end{pmatrix}, B = \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}$$

$$C = \begin{pmatrix} -E & 0 & 0 & 0 & B & 0 & 0 & 0 & 0 & 0 \\ -A_1 & -E & 0 & 0 & 0 & B & 0 & 0 & 0 & 0 \\ A_2 & -A_1 & -E & 0 & 0 & 0 & B & 0 & 0 & 0 \\ 0 & A_2 & -A_1 & -E & 0 & 0 & 0 & B & 0 & 0 \\ 0 & 0 & A_2 & -A_1 & 0 & 0 & 0 & 0 & B & 0 \\ 0 & 0 & 0 & A_2 & 0 & 0 & 0 & 0 & 0 & B \end{pmatrix}$$

As stated in [15], C only has full rank if $\lambda \neq 0$. We verify this using our `ComprehensiveRank` routine and find C to be full rank when $\lambda \neq 0$ and $\mu \neq 1/2$. Figure 2.5 in Appendix 2.B gives the complete output of our implementation with all possible rank values.

2.5.3 Example 2

A second example from [24], we have a matrix depending on 6 complex parameters, z_{ij}, for $i = 1,2$, $j = 1,2,3$

$$X = \begin{bmatrix} -4z_{11} - 4z_{12} & -4z_{12} - 4z_{13} & 20z_{13} + 24z_{11} + 44z_{12} \\ -7z_{11} - 6z_{12} + z_{13} & -18z_{12} - 12z_{13} - 6z_{11} & 54z_{13} + 72z_{11} + 126z_{12} \\ -z_{21} + z_{23} & -12z_{22} - 6z_{21} - 6z_{23} & 24z_{23} + 60z_{22} + 36z_{21} \end{bmatrix}$$

Since $\det(X) \equiv 0$ we immediately know $\text{rank}(X) < 3$. The result computed using our algorithm gives 23 cases, but most importantly, no cases where the rank is 3. Sample cases include:

$$\text{rank}(X) = 1 \quad \text{if} \quad \begin{cases} 2z_{11} + 3z_{12} + z_{13} = 0 \\ 2z_{21} + 3z_{22} + z_{23} = 0 \\ z_{22} + z_{23} \neq 0 \end{cases}$$

$$\text{rank}(X) = 2 \quad \text{if} \quad \begin{cases} z_{12} + z_{13} = 0 \\ z_{22} + z_{23} = 0 \\ z_{11} - z_{13} \neq 0 \\ z_{21} - z_{23} \neq 0 \end{cases}$$

For the full list of cases see Figures 2.6 and 2.7 in Appendix 2.B.

2.5.4 Example 3

The final example we show is a modified version of the example in [14] where we introduce a new parameter c such that $c > 0$ and maintain the condition that $0.2 \leq a \leq 1.2$.

$$A = \begin{bmatrix} -1 & 1 & 1 & 1 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 1/2 & 0 & 1/2 & 0 & 1 & 0 \\ 0 & ca & 0 & a & 0 & 0 & 0 \\ 0 & 0 & -ca & 0 & -a & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

We find that a rank of 6 or 7 is possible. The resulting conditions on a and c to have a rank of 6 are

$$\begin{cases} c = 2 \\ \frac{1}{5} \leq a \leq \frac{6}{5}\,, \end{cases}$$

and the conditions for rank 7 are

$$\begin{cases} c > 0 \\ c \neq 2 \\ \frac{1}{5} \leq a \leq \frac{6}{5}\,. \end{cases}$$

The commands executed are displayed in Figure 2.8 in Appendix 2.B.

2.6 Conclusion

For an $m \times n$ parametric matrix $A(\alpha)$, where the parameters α are subject to polynomial constraints S, we have successfully developed and implemented a method for the computation of the rank of $A(\alpha)$. By taking advantage of the methods of the `RegularChains` library we are able to simplify the problem into computing disjoint sets of conditions where each set corresponds to a unique value of the rank. We have developed methods for both the case where we have algebraic and semi-algebraic constraints on the parameters.

Bibliography

[1] P. Aubry, D. Lazard, and M. Moreno Maza. On the theories of triangular sets. *Journal of Symbolic Computation*, 28(1-2):105–124, 1999.

[2] C. Ballarin and M. Kauers. Solving parametric linear systems: an experiment with constraint algebraic programming. *ACM Sigsam Bulletin*, 38(2):33–46, 2004.

[3] F. Boulier, F. Lemaire, and M. Moreno Maza. Well known theorems on triangular systems and the D5 principle. In *Proceedings of Transgressive Computing*, Granada, Spain, 2006.

[4] P. A. Broadbery, T. Gómez-Díaz, and S. M. Watt. On the implementation of dynamic evaluation. In *Proceedings of the 1995 International Symposium on Symbolic and Algebraic Computation*, pages 77–84, 1995.

[5] C. Chen, J. H. Davenport, F. Lemaire, M. Moreno Maza, N. Phisanbut, B. Xia, R. Xiao, and Y. Xie. Solving semi-algebraic systems with the `regularchains` library in Maple. In *Proceedings of Mathematical Aspects of Computer and Information Sciences*, pages 38–51, 2011.

[6] C. Chen, J. H. Davenport, J. P. May, M. Moreno Maza, B. Xia, and R. Xiao. Triangular decomposition of semi-algebraic systems. *Journal of Symbolic Computation*, 49:3–26, 2013.

[7] C. Chen, J. H. Davenport, M. Moreno Maza, C. Xia, and R. Xiao. Computing with semi-algebraic sets represented by triangular decomposition. In *Proceedings of the 2011 International Symposium on Symbolic and Algebraic Computation*, pages 75–82. ACM, 2011.

[8] C. Chen, O. Golubitsky, F. Lemaire, M. Moreno Maza, and W. Pan. Comprehensive triangular decomposition. In *Proceedings of Computer Algebra in Scientific Computing*, volume 4770 of *Lecture Notes in Computer Science*, pages 73–101, 2007.

[9] C. Chen and M. Moreno Maza. Algorithms for computing triangular decomposition of polynomial systems. *Journal of Symbolic Computation*, 47(6):610–642, 2012.

[10] R. M. Corless and S. E. Thornton. A package for parametric matrix computations. In *Proceedings of the International Congress on Mathematical Software*, pages 442–449. Springer, 2014.

[11] M. D. Darmian and A. Hashemimir. Parametric FGLM algorithm. *Journal of Symbolic Computation*, 82:38–56, 2017.

[12] J. Della Dora, C. Dicrescenzo, and D. Duval. About a new method for computing in algebraic number fields. In *European Conference on Computer Algebra*, pages 289–290, 1985.

[13] G. M. Diaz-Toca, L. Gonzalez-Vega, and H. Lombardi. Generalizing Cramer's rule: Solving uniformly linear systems of equations. *SIAM Journal on Matrix Analysis and Applications*, 27(3):621–637, 2005.

[14] S. G. Dietz, C. W. Scherer, and W. Huygen. Linear parameter-varying controller synthesis using matrix sum-of-squares relaxations. In *Brazilian Automation Conference*, 2006.

[15] M. I. García-Planas and J. Clotet. Analyzing the set of uncontrollable second order generalized linear systems. *International Journal of Applied Mathematics and Informatics*, 1(2):76–83, 2007.

[16] M. Kalkbrener. *Three contributions to elimination theory*. PhD thesis, Johannes Kepler University, Linz, 1991.

[17] D. Kapur. An approach for solving systems of parametric polynomial equations. *Principles and Practices of Constraint Programming*, pages 217–244, 1995.

[18] M. Karow, D. Kressner, and F. Tisseur. Structured eigenvalue condition numbers. *SIAM Journal on Matrix Analysis and Applications*, 28(4):1052–1068, 2006.

[19] F. Lemaire, M. Moreno Maza, and Y. Xie. The `regularchains` library. In I. S. Kotsireas, editor, *Maple Conference*, pages 355–368, 2005.

[20] W. Y. Sit. An algorithm for solving parametric linear systems. *Journal of Symbolic Computation*, 13(4):353–394, 1992.

[21] F. Tisseur and N. J. Higham. Structured pseudospectra for polynomial eigenvalue problems, with applications. *SIAM journal on matrix analysis and applications*, 23(1):187–208, 2001.

[22] V. Weispfenning. Comprehensive Gröbner bases. *Journal of Symbolic Computation*, 14(1):1–30, 1992.

[23] L. Yang and J. Zhang. Searching dependency between algebraic equations: an algorithm applied to automated reasoning. Technical Report IC/89/263, International Atomic Energy Agency, Miramare, Trieste, Italy, 1991.

[24] B. Zhou and G. Duan. An explicit solution to the matrix equation $AX - XF = BY$. *Linear Algebra and its Applications*, 402:345–366, 2005.

2.A Appendix A

In this section, we provide examples of triangular decompositions of polynomial systems. As a first illustration let us consider the following semi-algebraic system

$$\left\{ \begin{array}{rcl} (x-1)(y^2+t^2)+(x-2)(y^2-t) & = & 0 \\ (x-1)(x-2) & = & 0, \end{array} \right. \tag{2.1}$$

and solve it with the `RealTriangularize` command of the `RegularChains` library, leading to the computations in Figure 2.3.

```
> R := PolynomialRing([y, x, t]);
```

$$R := polynomial\_ring$$

```
> F := [(x - 1)·(x - 2), (x - 1)·(y^2 + t^2) + (x - 2)·(y^2 - t)];
```

$$F := \left[(x-1)\,(x-2),\ (x-1)\,\left(t^2+y^2\right)+(x-2)\,\left(y^2-t\right)\right]$$

```
> RealTriangularize(F, R, output = record);
```

$$\left\{ \begin{array}{c} y=0 \\ x-2=0 \\ t=0 \end{array} \right.,\ \left\{ \begin{array}{c} y^2-t=0 \\ x-1=0 \\ t>0 \end{array} \right.,\ \left\{ \begin{array}{c} y=0 \\ x-1=0 \\ t=0 \end{array} \right.$$

Figure 2.3: A triangular decomposition into semi-algebraic systems computed with the `RealTriangularize` command.

The above triangular decomposition consists of three regular semi-algebraic systems. Let us denote them respectively by $[T_1, Q_1, P_1]$, $[T_2, Q_2, P_2]$, $[T_3, Q_3, P_3]$. The first and the third ones consist simply of a regular chain, thus we have $P_1 = P_3 = \varnothing$ and $Q_1 = Q_3 =$ true. In fact each of $[T_1, Q_1, P_1]$, $[T_3, Q_3, P_3]$ simply encodes a point, that is, a zero-dimensional component. For the second one, we have $P_2 = \varnothing$ and $Q_2 = 0 < t$, thus $T_2 = \{y^2 - t, x - 1\}$. Therefore, $[T_2, Q_2, P_2]$, is a parametrization of a one-dimensional component.

```
> R := PolynomialRing([x, y, z]);
```

$$R := polynomial\_ring$$

```
> F := [5·x^2 + 2·z^2·x + 5·y^6 + 15·y^4 − 5·y^3 − 15·y^5 + 5·z^2];
```

$$F := [5y^6 - 15y^5 + 15y^4 + 2z^2x - 5y^3 + 5x^2 + 5z^2]$$

```
> RealTriangularize(F, R, output = record);
```

$$\left\{\begin{array}{l} 5x^2 + 2z^2x + 5y^6 + 15y^4 - 5y^3 - 15y^5 + 5z^2 = 0 \\ 25y^6 - 75y^5 + 75y^4 - z^4 - 25y^3 + 25z^2 < 0 \end{array}\right.,$$

$$\left\{\begin{array}{c} 5x + z^2 = 0 \\ 25y^6 - 75y^5 + 75y^4 - 25y^3 - z^4 + 25z^2 = 0 \\ 64z^4 - 1600z^2 + 25 > 0 \\ z \neq 0 \\ z - 5 \neq 0 \\ z + 5 \neq 0 \end{array}\right., \left\{\begin{array}{c} x = 0 \\ y - 1 = 0 \\ z = 0 \end{array}\right., \left\{\begin{array}{c} x = 0 \\ y = 0 \\ z = 0 \end{array}\right.,$$

$$\left\{\begin{array}{c} x + 5 = 0 \\ y - 1 = 0 \\ z - 5 = 0 \end{array}\right., \left\{\begin{array}{c} x + 5 = 0 \\ y = 0 \\ z - 5 = 0 \end{array}\right., \left\{\begin{array}{c} x + 5 = 0 \\ y - 1 = 0 \\ z + 5 = 0 \end{array}\right., \left\{\begin{array}{c} x + 5 = 0 \\ y = 0 \\ z + 5 = 0 \end{array}\right.,$$

$$\left\{\begin{array}{c} 5x + z^2 = 0 \\ 2y - 1 = 0 \\ 64z^4 - 1600z^2 + 25 = 0 \end{array}\right.$$

Figure 2.4: Output of the `RealTriangularize` command for the *EVE* surface.

Figure 2.4 contains a second and more advanced example, where the purpose of the MAPLE session is to obtain a description of the real points of the hypersurface *EVE* from the *Algebraic Surface Gallery*[2] and whose equation is $5x^2+2xz^2+5y^6+15y^4+5z^2-15y^5-5y^3 = 0$. The solutions of the above are all (x,y,z) where x,y,z are complex numbers satisfying this equation. The output of `RealTriangularize` consists of 9 regular semi-algebraic systems, for which the variables are ordered as $x > y > z$. The first regular semi-algebraic system represents a two-dimensional component. Indeed, it defines x as the solution of

[2]This is a collection of algebraic surfaces, well-known in the mathematical literature and available at `http://homepage.univie.ac.at/herwig.hauser/bildergalerie/gallery.html`

a parametric equation of degree 2, where y,z are regarded as parameters subject to an inequality (defined by the discriminant of the equation) which ensures the existence of two x-values for each valid (y,z)-value. The second regular semi-algebraic system represents a one-dimensional component: the two equations define (x,y) as functions of z, which is subject to various inequalities. Each of the other 7 regular semi-algebraic systems encodes a zero-dimensional component, that is, a finite set of points.

2.B Appendix B

In this section, we show the full solutions to the examples presented in Section 5. In the first example we computed that the matrix can have a rank of 15 through 18 depending upon the parameters. Figure 2.5 shows the conditions resulting in each rank. For the

```
> z3 := ZeroMatrix(3) : z1 := ZeroMatrix(3, 1) :
> E, B := Matrix([[1, 3, 1], [3, 1, 1, ], [0, 0, 0]]), Matrix([[0], [0], [1]]);
```

$$E, B := \begin{bmatrix} 1 & 3 & 1 \\ 3 & 1 & 1 \\ 0 & 0 & 0 \end{bmatrix}, \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}$$

```
> A1, A2 := Matrix([[1, 1, 3], [1, 3, 1], [0, 0, 0]]), Matrix([[λ, 3·λ, λ], [3·λ + μ, λ + μ, λ + 3·μ],
    [0, 0, 0]]);
```

$$A1, A2 := \begin{bmatrix} 1 & 1 & 3 \\ 1 & 3 & 1 \\ 0 & 0 & 0 \end{bmatrix}, \begin{bmatrix} \lambda & 3\lambda & \lambda \\ 3\lambda+\mu & \lambda+\mu & \lambda+3\mu \\ 0 & 0 & 0 \end{bmatrix}$$

```
> A := Matrix([[-E, z3, z3, z3, B, z1, z1, z1, z1, z1], [-A1, -E, z3, z3, z1, B, z1, z1,
     z1, z1], [A2, A1, -E, z3, z1, z1, B, z1, z1, z1], [z3, A2, -A1, -E, z1, z1, z1, B,
     z1, z1], [z3, z3, A2, -A1, z1, z1, z1, z1, B, z1], [z3, z3, z3, A2, z1, z1, z1, z1,
     z1, B]]) :
> R := PolynomialRing([μ, λ]) :
> rank := ComprehensiveRank(A, R) :
> seq(print(Display(rank[i], R)), i = 1 ..nops(rank));
```

$$\left[18, \begin{cases} 2\mu - 1 \neq 0 \\ \lambda \neq 0 \end{cases} \right]$$

$$\left[17, \begin{cases} 2\mu - 1 = 0 \\ \lambda \neq 0 \end{cases} \right]$$

$$\left[16, \begin{cases} \lambda = 0 \\ \mu + 1 \neq 0 \end{cases} \right]$$

$$\left[15, \begin{cases} \mu + 1 = 0 \\ \lambda = 0 \end{cases} \right]$$

Figure 2.5: The computed rank values and the corresponding conditions on the parameters for Example 1

second example we show all conditions on the parameters resulting in a rank of 0, 1 or 2. Figure 2.6 and 2.7 show all conditions for the respective ranks. The executed commands and the output for Example 3 are displayed in Figure 2.8.

```
> A := Matrix([[-4·z[1, 1] - 4·z[1, 2], -4·z[1, 2] - 4·z[1, 3],  20·z[1, 3] + 24·z[1, 1] + 44·z[1  , 2]], [
    -7·z[1, 1] - 6·z[1, 2] + z[1, 3], -18·z[1, 2] - 12·z[1, 3] - 6·z[1, 1], 54·z[1, 3] + 72·z[1, 1]
    + 126·z[1, 2]], [-z[2, 1] + z[2, 3], -12·z[2, 2] - 6·z[2, 1] - 6·z[2, 3], 24·z[2, 3] + 60·z[2, 2]
    + 36·z[2, 1]]]);
```

$$A := \begin{bmatrix} -4z_{1,1} - 4z_{1,2} & -4z_{1,2} - 4z_{1,3} & 20z_{1,3} + 24z_{1,1} + 44z_{1,2} \\ -7z_{1,1} - 6z_{1,2} + z_{1,3} & -18z_{1,2} - 12z_{1,3} - 6z_{1,1} & 54z_{1,3} + 72z_{1,1} + 126z_{1,2} \\ -z_{2,1} + z_{2,3} & -12z_{2,2} - 6z_{2,1} - 6z_{2,3} & 24z_{2,3} + 60z_{2,2} + 36z_{2,1} \end{bmatrix}$$

```
> Determinant(A);
```

$$0$$

```
> R := PolynomialRing([z[1, 1], z[1, 2], z[1, 3], z[2, 1], z[2, 2], z[2, 3]]):
> rank := ComprehensiveRank(A, R):
> seq(print(Display(rank[i], R)), i = 1..nops(rank));
```

$$\left[2, \begin{cases} z_{1,2} + z_{1,3} \neq 0 \\ z_{2,1} + 2z_{2,2} + z_{2,3} \neq 0 \\ 3z_{2,1} + 5z_{2,2} + 2z_{2,3} \neq 0 \\ 2z_{2,1} + 3z_{2,2} + z_{2,3} \neq 0 \\ 3z_{2,1} + 4z_{2,2} + z_{2,3} \neq 0 \\ z_{2,2} + z_{2,3} \neq 0 \end{cases}, \begin{cases} z_{2,2} + z_{2,3} = 0 \\ z_{1,2} + z_{1,3} \neq 0 \\ z_{2,1} - z_{2,3} \neq 0 \end{cases}, \begin{cases} 3z_{2,1} + 5z_{2,2} + 2z_{2,3} = 0 \\ z_{1,2} + z_{1,3} \neq 0 \\ z_{2,2} + z_{2,3} \neq 0 \end{cases}, \right.$$

$$\begin{cases} z_{2,1} + 2z_{2,2} + z_{2,3} = 0 \\ z_{1,2} + z_{1,3} \neq 0 \\ z_{2,2} + z_{2,3} \neq 0 \end{cases}, \begin{cases} 2z_{2,1} + 3z_{2,2} + z_{2,3} = 0 \\ 2z_{1,1} + 3z_{1,2} + z_{1,3} \neq 0 \\ z_{1,2} + z_{1,3} \neq 0 \\ z_{2,2} + z_{2,3} \neq 0 \end{cases}, \begin{cases} 3z_{2,1} + 4z_{2,2} + z_{2,3} = 0 \\ 3z_{1,1} + 4z_{1,2} + z_{1,3} \neq 0 \\ z_{1,2} + z_{1,3} \neq 0 \\ z_{2,2} + z_{2,3} \neq 0 \end{cases},$$

$$\begin{cases} z_{2,1} - z_{2,3} = 0 \\ z_{2,2} + z_{2,3} = 0 \\ 2z_{1,1} + 3z_{1,2} + z_{1,3} \neq 0 \\ 3z_{1,1} + 4z_{1,2} + z_{1,3} \neq 0 \\ z_{1,2} + z_{1,3} \neq 0 \end{cases}, \begin{cases} z_{1,2} + z_{1,3} = 0 \\ z_{1,1} - z_{1,3} \neq 0 \\ z_{2,1} + 2z_{2,2} + z_{2,3} \neq 0 \\ 3z_{2,1} + 5z_{2,2} + 2z_{2,3} \neq 0 \\ 2z_{2,1} + 3z_{2,2} + z_{2,3} \neq 0 \\ 3z_{2,1} + 4z_{2,2} + z_{2,3} \neq 0 \\ z_{2,2} + z_{2,3} \neq 0 \end{cases}, \begin{cases} z_{1,2} + z_{1,3} = 0 \\ z_{2,2} + z_{2,3} = 0 \\ z_{1,1} - z_{1,3} \neq 0 \\ z_{2,1} - z_{2,3} \neq 0 \end{cases},$$

$$\begin{cases} z_{1,2} + z_{1,3} = 0 \\ 3z_{2,1} + 5z_{2,2} + 2z_{2,3} = 0 \\ z_{1,1} - z_{1,3} \neq 0 \\ z_{2,2} + z_{2,3} \neq 0 \end{cases}, \begin{cases} z_{1,2} + z_{1,3} = 0 \\ z_{2,1} + 2z_{2,2} + z_{2,3} = 0 \\ z_{1,1} - z_{1,3} \neq 0 \\ z_{2,2} + z_{2,3} \neq 0 \end{cases}, \begin{cases} z_{1,2} + z_{1,3} = 0 \\ 2z_{2,1} + 3z_{2,2} + z_{2,3} = 0 \\ z_{1,1} - z_{1,3} \neq 0 \\ z_{2,2} + z_{2,3} \neq 0 \end{cases},$$

$$\left. \begin{cases} z_{1,2} + z_{1,3} = 0 \\ 3z_{2,1} + 4z_{2,2} + z_{2,3} = 0 \\ z_{1,1} - z_{1,3} \neq 0 \\ z_{2,2} + z_{2,3} \neq 0 \end{cases}, \begin{cases} z_{1,2} + z_{1,3} = 0 \\ z_{2,1} - z_{2,3} = 0 \\ z_{2,2} + z_{2,3} = 0 \\ z_{1,1} - z_{1,3} \neq 0 \end{cases} \right]$$

Figure 2.6: The computed rank values and the corresponding conditions on the parameters for Example 2 (part 1)

$$
\left[\left[1, \begin{cases} z_{1,1}-z_{1,3}=0 \\ z_{1,2}+z_{1,3}=0 \\ z_{2,2}+z_{2,3}\neq 0 \end{cases}, \begin{cases} z_{1,1}-z_{1,3}=0 \\ z_{1,2}+z_{1,3}=0 \\ z_{2,2}+z_{2,3}=0 \\ z_{2,1}-z_{2,3}\neq 0 \end{cases}, \begin{cases} 3z_{1,1}+4z_{1,2}+z_{1,3}=0 \\ 3z_{2,1}+4z_{2,2}+z_{2,3}=0 \\ z_{2,2}+z_{2,3}\neq 0 \end{cases}, \begin{cases} 3z_{1,1}+4z_{1,2}+z_{1,3}=0 \\ z_{2,1}-z_{2,3}=0 \\ z_{2,2}+z_{2,3}=0 \\ z_{1,2}+z_{1,3}\neq 0 \end{cases},\right.\right.
$$

$$
\begin{cases} 2z_{1,1}+3z_{1,2}+z_{1,3}=0 \\ 2z_{2,1}+3z_{2,2}+z_{2,3}=0 \\ z_{2,2}+z_{2,3}\neq 0 \end{cases}, \begin{cases} 2z_{1,1}+3z_{1,2}+z_{1,3}=0 \\ z_{2,1}-z_{2,3}=0 \\ z_{2,2}+z_{2,3}=0 \\ z_{1,2}+z_{1,3}\neq 0 \end{cases}, \begin{cases} z_{1,1}-z_{1,3}=0 \\ z_{1,2}+z_{1,3}=0 \\ z_{2,1}+2z_{2,2}+z_{2,3}=0 \\ z_{2,2}+z_{2,3}\neq 0 \end{cases},
$$

$$
\left.\begin{cases} z_{1,1}-z_{1,3}=0 \\ z_{1,2}+z_{1,3}=0 \\ 3z_{2,1}+5z_{2,2}+2z_{2,3}=0 \\ z_{2,2}+z_{2,3}\neq 0 \end{cases}\right]
$$

$$
\left.\left[0, \begin{cases} z_{1,1}-z_{1,3}=0 \\ z_{1,2}+z_{1,3}=0 \\ z_{2,1}-z_{2,3}=0 \\ z_{2,2}+z_{2,3}=0 \end{cases}\right]\right]
$$

Figure 2.7: The computed rank values and the corresponding conditions on the parameters for Example 2 (part 2)

```
> A := Matrix([[-1, 1, 1, 1, 1, 0, 1], [0, 0, 1, 0, 1, 0, 0], [0, 1/2, 0, 1/2, 0, 1, 0], [0, c·a, 0, a, 0, 0,
   0], [0, 0, -c·a, 0, -a, 0, 1], [0, 0, 1, 0, 0, 0, 0], [1, 1, 0, 0, 0, 0, 0]]);
```

$$
A := \begin{bmatrix} -1 & 1 & 1 & 1 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & \frac{1}{2} & 0 & \frac{1}{2} & 0 & 1 & 0 \\ 0 & c\,a & 0 & a & 0 & 0 & 0 \\ 0 & 0 & -c\,a & 0 & -a & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}
$$

```
> R := PolynomialRing([a, c]):
> rank := RealComprehensiveRank(A, [], [a - 1/5, 6/5 - a], [c], [], R):
> seq(print(Display(rank[i], R)), i = 1..nops(rank));
```

$$
\left[7, \left[\begin{cases} 5a-6=0 \\ c>0 \textbf{ and } c-2\neq 0 \end{cases}, \begin{cases} 5a-1=0 \\ c>0 \textbf{ and } c-2\neq 0 \end{cases}, 5a-1>0 \textbf{ and } 5a<6 \textbf{ and } c>0 \textbf{ and } c-2\neq 0\right]\right]
$$

$$
\left[6, \left[\begin{cases} 5a-6=0 \\ c-2=0 \end{cases}, \begin{cases} 5a-1=0 \\ c-2=0 \end{cases}, \begin{cases} c-2=0 \\ 5a-1>0 \textbf{ and } 5a<6 \end{cases}\right]\right]
$$

Figure 2.8: The computed rank values and the corresponding conditions on the parameters for Example 3

Chapter 3

Zigzag Form over Families of Parametric Matrices

3.1 Introduction

Currently, computations on parametric matrices are considered difficult and costly because canonical forms such as the Frobenius, Jordan and Weyr forms are discontinuous; this requires special cases for completeness, and exhaustive analysis produces combinatorially many cases. Some papers considering special cases with parameters include [1] and [4].

There are a large number of methods for computing the Frobenius form of a constant matrix such as in [2], [6], [7], [8], [9] and [10]. We instead modify the algorithm of Storjohann from [11] and [12] for computations on parametric matrices. This algorithm requires the computation of a so-called *Zigzag form* before the Frobenius form can be computed. The Zigzag form itself is not directly useful for applications but provides a matrix from which the Frobenius form can easily be obtained.

3.2 Background Material

Let $\mathbb{K}$ be an algebraically closed field or a real closed field. Let $\alpha_1 < \cdots < \alpha_m$ be $m \geq 1$ ordered variables. We denote by $\mathbb{K}[\alpha] = \mathbb{K}[\alpha_1, \ldots, \alpha_m]$ the ring of polynomials in the variables $\alpha = \alpha_1, \ldots, \alpha_m$ with coefficients in $\mathbb{K}$. We denote by $\mathbb{K}(\alpha)$ the quotient field of $\mathbb{K}[\alpha]$, that is, the field of multivariate rational functions in α with coefficients in $\mathbb{K}$.

Proposition 3.2.1 ([5, 3]). *For two constructible (resp. semi-algebraic) sets $S_1, S_2 \subset \mathbb{K}^m$, one can compute a triangular decomposition of their intersection $S_1 \cap S_2$, their union $S_1 \cup S_2$ and the set theoretical difference $S_1 \setminus S_2$.*

Remark 3.2.2. Let $S \subseteq \mathbb{K}^m$ be a constructible (resp. semi-algebraic) set and $f(\alpha) \in \mathbb{K}[\alpha]$. A useful tool for later computations will be finding the partition $(S_{\text{eq}}, S_{\text{neq}})$ of S defined by

$$S_{\text{eq}} \;=\; S \cap V(f(\alpha)) \quad \text{and} \quad S_{\text{neq}} \;=\; S \setminus V(f(\alpha)) = S \setminus S_{\text{eq}}$$

by means of the algorithms of [5, 3].

3.3 Zigzag Matrix

Parametric Polynomial. Let $f(x;\alpha)$ be a monic polynomial of degree r w.r.t. x. We write:

$$f(x;\alpha) = f_0(\alpha) + f_1(\alpha)x + \cdots + f_{r-1}(\alpha)x^{r-1} + x^r \tag{3.1}$$

with coefficients $f_0(\alpha), \ldots, f_{r-1}(\alpha) \in \mathbb{K}(\alpha)$. The α values are constrained to belong to a constructible (resp. semi-algebraic) set S such that the denominator of every coefficient $f_0(\alpha), \ldots, f_{r-1}(\alpha)$ is nonzero everywhere on S.

Zigzag Matrix. A parametric *Zigzag matrix* takes the form

$$\begin{bmatrix}
C_{c_1(x;\alpha)} & B_1 & & & & & & \\
 & C^T_{c_2(x;\alpha)} & & & & & & \\
 & B_2 & C_{c_3(x;\alpha)} & B_3 & & & & \\
 & & & C^T_{c_4(x;\alpha)} & & & & \\
 & & & & \ddots & & & \\
 & & & & & C^T_{c_{k-2}(x;\alpha)} & & \\
 & & & & & B_{k-2} & C_{c_{k-1}(x;\alpha)} & B_{k-1} \\
 & & & & & & & C^T_{c_k(x;\alpha)}
\end{bmatrix}$$

for k even.

Each polynomial $c_1(x;\alpha), \ldots, c_k(x;\alpha)$ takes the same form as Equation (3.1) and each $C_{c_i(x;\alpha)}$ is a companion matrix of $c_i(x;\alpha)$. The B_i blocks have all entries zero except those in the upper left corner which are either 0 or 1; each B_i block has its size determined by its neighbouring companion blocks.

If there is an odd number of diagonal blocks we allow $\deg c_k = 0$ while $\deg c_i \geq 1$ for $1 \leq i < k$. This allows the kth diagonal block to have dimension zero and hence the block B_{k-1} above $C^T_{c_k(x;\alpha)}$ will also have dimension zero.

3.4 Computation

Theorem 3.4.1. *For every matrix $A(\alpha) \in \mathbb{K}^{n\times n}[\alpha]$, there exists a partition $(S_1, \ldots, S_N)$ of input constructible (resp. semi-algebraic) set S such that for each S_i, there exists a matrix $Z_i(\alpha) \sim A(\alpha)$ in Zigzag form where the denominators of the coefficients of the entries of $Z_i(\alpha)$ are all non-zero everywhere on S_i.*

We follow the same algorithm presented in Section 2 of [11]. Stages 1 and 3 must be modified for finding pivots vanishing nowhere on the underlying constructible (resp. semi-algebraic) set.

Once computation has split into two branches, one where a pivot has been found and the set S has been replaced with S_{neq}, and another where the pivot has not yet been found and the search for a pivot continues with S replaced by S_{eq}, the computations proceed in parallel (or by stack execution sequentially).

3.5 Implementation

A sequential implementation has been written in Maple to compute the set of Zigzag forms similar to an input parametric matrix under algebraic or semi-algebraic constraints. The `RegularChains` library in Maple contains many useful procedures and sub-packages for performing polynomial computations with parameters. See `www.regularchains.org` for details. The `ConstructibleSetTools` and `SemiAlgebraicSetTools` sub-packages of `RegularChains` are useful for representing constructible sets and semi-algebraic sets respectively and, performing set operations on them, as mentioned in Proposition 3.2.1. The `GeneralConstruct` procedure from the `ConstructibleSetTools` sub-package obtains a triangular decomposition of an input system of polynomial equations and inequations. Analogously, the `RealTriangularize` procedure computes a triangular decomposition of a semi-algebraic set given by an input system of polynomial equations, inequations and inequalities. The intersection and set theoretical difference computations needed in Remark 3.2.2 are performed by the `Intersection` and `Difference` commands of the `ConstructibleSetTools` and `SemiAlgebraicSetTools` sub-packages.

Cost. As one can expect, we have verified experimentally that the main cost of the implemented algorithm comes from the computations of the intersections needed in Remark 3.2.2. Clearly, this costs grows exponentially with the number m of parameters $\alpha_1, \ldots, \alpha_m$.

Example 3.5.1. Consider the 3×3 matrix with a single parameter α over the complex numbers $\mathbb{C}$

$$A(\alpha) = \begin{bmatrix} -1 & -\alpha-1 & 0 \\ -1/2 & \alpha-2 & 1/2 \\ -2 & 3\alpha+1 & -1 \end{bmatrix}$$

with no input conditions on α. The Zigzag forms similar to this matrix are

$$Z_1(\alpha) = \begin{bmatrix} 0 & 0 & 4\alpha \\ 1 & 0 & 4(\alpha-1) \\ 0 & 1 & \alpha-4 \end{bmatrix} \; \alpha+3 \neq 0, \quad Z_2(\alpha) = \left[\begin{array}{cc|c} 0 & -4 & 1 \\ 1 & -4 & 0 \\ \hline 0 & 0 & -3 \end{array}\right] \; \alpha+3 = 0\,.$$

Clearly, $Z_1(\alpha)$ is already in Frobenius form whereas $Z_2(\alpha)$ requires additional work to obtain the Frobenius form. This example turns out to have a continuous Frobenius form in the parameter, hence the Frobenius form is $Z_1(\alpha)$ for all values of α.

3.6 Applications

An implementation of the algorithm for computing the Zigzag form is available as part of the `ParametricMatrixTools` package we are developing for the Maple computer algebra system. A repository containing the implementation as well as several examples is available at . We are currently extending the functionalities offered by this implementation dedicated to parametric matrices under algebraic or semi-algebraic constraints. Indeed, from the Zigzag form, the Frobenius form can easily be determined, leading to the computation of the Jordan form and the minimal polynomial.

Bibliography

[1] V. I. Arnol'd. On matrices depending on parameters. *Russian Mathematical Surveys*, 26(2):29–43, 1971.

[2] D. Augot and P. Camion. On the computation of minimal polynomials, cyclic vectors, and Frobenius forms. *Linear Algebra and its Applications*, 260:61–94, 1997.

[3] C. Chen, J. H. Davenport, J. P. May, M. Moreno Maza, B. Xia, and R. Xiao. Triangular decomposition of semi-algebraic systems. *Journal of Symbolic Computation*, 49:3–26, 2013.

[4] G. Chen. Computing the normal forms of matrices depending on parameters. In *Proceedings of the 1989 International Symposium on Symbolic and Algebraic Computation*, pages 244–249. ACM, 1989.

[5] V. G. Ganzha, E. W. Mayr, and E. V. Vorozhtsov, editors. *Computer Algebra in Scientific Computing*, volume 4770 of *Lecture Notes in Computer Science*. Springer, 2007.

[6] M. Giesbrecht. Fast algorithms for matrix normal forms. In *Symposium on Foundations of Computer Science*, pages 121–130. IEEE, 1992.

[7] M. Giesbrecht. Nearly optimal algorithms for canonical matrix forms. *SIAM Journal on Computing*, 24(5):948–969, 1995.

[8] E. Kaltofen, M. S. Krishnamoorthy, and B. D. Saunders. Parallel algorithms for matrix normal forms. *Linear Algebra and its Applications*, 136:189–208, 1990.

[9] K. R. Matthews. A rational canonical form algorithm. *Mathematica Bohemica*, 117(3):315–324, 1992.

[10] P. Ozello. *Calcul exact des formes de Jordan et de Frobenius d'une matrice*. PhD thesis, Université Joseph-Fourier-Grenoble I, 1987.

[11] A. Storjohann. An $\mathcal{O}(n^3)$ algorithm for the Frobenius normal form. In *Proceedings of the 1998 International Symposium on Symbolic and Algebraic Computation*, pages 101–105. ACM, 1998.

[12] A. Storjohann. *Algorithms for matrix canonical forms*. PhD thesis, ETH Zurich, 2000.

Chapter 4

Jordan Canonical Form with Parameters From Frobenius Form with Parameters

4.1 Introduction

The Jordan canonical form (JCF) of a matrix and its close cousin the Weyr canonical form are foundational tools in the analysis of eigenvalue problems and dynamical systems. For a summary of theory, see for instance Chapter 6 in The Handbook of Linear Algebra [20]; for the Weyr form, see [29].

The first use usually seen for the JCF is as a canonical form for matrix similarity: two matrices are similar if and only if they have identical (sets of, up to ordering) Jordan canonical forms [21]. Of course, there are other (often better) canonical forms for similarity such as the Frobenius (rational) canonical form, or the rational Jordan form [13, 22].

The JCF is well known to be discontinuous with respect to changes in the entries if the base field $\mathbb{K}$ has nonempty open sets. We typically take $\mathbb{K} = \mathbb{C}$, the field of complex numbers. Therefore, the JCF cannot be computed numerically with small forward error, even when using a numerically stable algorithm. A numerically stable algorithm gives the exact answer for a nearby input $A + E$ with $||E|| = \mathcal{O}(\mu)$ or $\mathcal{O}(\mu||A||)$ or even $|e_{ij}| < \mathcal{O}(\mu)|a_{ij}|$ where μ is unit roundoff.

This has forced the development of alternatives to the JCF, such as the Schur form, which is numerically stable and useful in the computation of matrix functions via the Parlett recurrences, for instance [19]. Consider the computation of the matrix exponential. First computing the JCF is one of the famous "Nineteen Dubious Ways to Compute the

Exponential of a Matrix" [19, 27]; computing the matrix exponential remains of serious interest today (or perhaps is even of increased interest) because of new methods for "geometric" numerical integration of large systems [11, 17, 24].

Analysis of small systems containing symbolic parameters is also of great interest, in mathematical biology especially (models of disease dynamics in populations and in individual hosts, evolutionary or ecological models) but also in many other dynamical systems applications such as fluid-structure interactions, robot kinematics, and electrical networks. The algorithmic situation for systems containing parameters is much less well-developed than is the corresponding situation for numerical systems. Although alternatives to the JCF exist for the analysis of these systems, the JCF has become a standard tool with implementations available in every major computer algebra system (CAS).

The current situation in Maple is that explicit computation of the JCF of a matrix containing parameters of dimension 5 or more may fail in some simple cases. For example, Maple simply does not provide a result for the JCF of the Frobenius companion matrix of $p(x) = x^5 + x^4 + x^3 + x^2 + x + a$. Similar failures occur for the `MatrixFunction` and `MatrixExponential` procedures. Wolfram Alpha gives the generic answers, but fails to give non-generic ones. Computing matrix functions may succeed in cases where computing the JCF does not because the JCF need not be used (an interpolation algorithm can be used instead); see for instance Definition 1.4 in [19].

Most computer algebra systems have adopted some variation of the definition of algebraic functions as implicit roots of their defining polynomials. In Maple, the syntax uses `RootOf`; together with an `alias` facility. This gives a useful way to encode the mathematical statement (for instance) "Let α be a root of the polynomial $x^5+\varepsilon x+1 = 0$".

```
> alias(alpha = RootOf(x^5+eps*x+1,x)):
```

This should, in theory, allow symbolic computation of the JCF of (small) matrices, even ones containing parameters. To date in practice it has not. "In theory there is no difference between theory and practice; in practice there is."[1]

In this paper, we offer some progress, although we note that combinatorial growth of the resulting expressions remains a difficulty. However, the tools we provide here are already useful for some example applications and go some way towards filling a scientific and engineering need. We aim to minimize unnecessary growth throughout the computation.

The tools we use here include provisos [8] and comprehensive square-free factorization

[1] `http://wikiquote.org/wiki/Jan_L._A._van_de_Snepscheut`

with the `RegularChains` package. This is in the tradition of Cauchy:

> Thus Cauchy insisted in the preface to his Cours d'Analyse [1821] that mathematical reasoning must not be based on arguments "drawn from the generality of algebra," arguments which "tend to attribute an indefinite scope to the algebraic formulas, whereas in reality the majority hold true only under certain conditions and for certain quantities of the variables involved." T.Hawkins [18, p. 122]

Consider, for example, the Jacobian matrix in [36]. If you compute the JCF of this matrix using Maple's built-in `JordanForm` command it will return a diagonal matrix where the eigenvalues are large nested radical expressions as a result of explicitly solving the characteristic polynomial. In contrast, our `ComprehensiveJordanForm` method gives a full case discussion. Two interesting cases where the JCF is non-trivial are shown in Figure 4.1. Further details of this example are given in Section 4.6.5.

$$> J := \begin{bmatrix} 0 & 2\rho & 0 \\ a & 2\beta & 2\nu \\ b & -2\nu & 2\beta \end{bmatrix} :$$

```
> R := PolynomialRing([a, b, ρ, β, ν]) :
> JCF := ComprehensiveJordanForm(J, R,'output'='lazard') :
> Display(JCF[1], R), Display(JCF[25], R);
```

$$\left[\begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix}, \begin{cases} a = 0 \\ \beta = 0 \\ \nu = 0 \\ b \neq 0 \\ \rho \neq 0 \end{cases} \right], \left[\begin{bmatrix} 2\beta & 0 & 0 \\ 0 & \beta & 1 \\ 0 & 0 & \beta \end{bmatrix}, \begin{cases} 2\rho a + \beta^2 = 0 \\ \nu = 0 \\ 2\rho \neq 0 \\ \beta \neq 0 \end{cases} \right]$$

Figure 4.1: Our implementation provides a full case discussion of the JCF of a matrix with 5 parameters. Two non-trivial cases are shown.

In Section 4.5, we present an algorithm for computing the JCF of a matrix in Frobenius form where the entries are multivariate polynomials whose indeterminates are regarded as parameters. Our approach uses comprehensive square-free factorization to provide a complete case discussion. Classical approaches for computing the JCF rely on elementary row and column operations that maintain a similarity relation at each step [3, 14, 30]. Because the entries of the matrices we are considering are multivariate polynomials, row and column operations lead to significant expression growth that can be difficult to control. Additionally, this would require us to work over matrices of multivariate rational functions in the parameters, again making it difficult to control expression growth. By instead computing fraction free square-free factorizations, we are able to maintain better control over expression growth. Because our implementation does not use elementary row and column operations, we do not compute the similarity transformation matrix Q such that $J = Q^{-1}AQ$ gives the Jordan form J of A. We leave this problem for future work.

We present an implementation of our algorithm in Section 4.6 and use it to solve several problems taken from the literature. These examples are not in Frobenius form and we do not discuss in detail how we obtain the Frobenius form. The Frobenius form implementation uses standard algorithms based on GCD computations of parametric polynomials to find the Smith form of $A - xI$ and the relation between this and the Frobenius form of A [22].

Section 4.4 presents a new approach for computing the JCF of a non-parametric matrix in Frobenius form over the splitting field of the characteristic polynomial. Our discussion is based on the theory of regular chains. We do not apply this splitting field approach in the parametric case because the square-free factorization approach we use gives the complete structure of the JCF. Constructing the splitting field would be vastly more expensive than the approach presented in Section 4.5.

4.2 Some Prior Work

As previously mentioned, the JCF of a matrix $A \in \mathbb{C}^{n\times n}$ as a function of the entries of A has discontinuities. These discontinuities are often precisely what is important in applications. This also means that even numerically stable algorithms can sometimes give results with $\mathcal{O}(1)$ forward error. This is often also stated by saying that "computing the JCF is an ill-posed problem" [3].

This has not prevented people from trying to compute the JCF numerically anyway (see [3] and the references therein), but in general such efforts cannot always be satisfactory: discontinuous is ill-posed, and without regularization such efforts are (sometimes) doomed. There have been at least three responses in the literature.

One is to find other ways to solve your problem, i.e. compute matrix functions such as A^n and $\exp(tA)$, without first computing the JCF, and the invention of the numerically stable Schur factoring and the Parlett recurrences for instance has allowed significant success [19].

The second response is to find a canonical form that explicitly preserves the continuity or smoothness of the matrix; the versal forms of [1] do this. Incidentally, the Frobenius form with parameters is an example of a versal form (Arnol'd calls this a Sylvester family), but there are others. The paper [7] uses Carleman linearization to do something similar.

The third response is to assume exact input and try to do exact or symbolic computation of the JCF. Early attempts, e.g. [14], had high complexity: $\mathcal{O}(n^8)$ [30] in the dimension n and with expression growth $\mathcal{O}(2^{n^2})$. A key step is the computation of the Frobenius form, and the current best complexity algorithm is $\mathcal{O}(n^3)$ field operations, and keeps expression swell to a minimum [32]. Boolean circuit complexity results can be found in [22].

Inclusion of symbolic parameters makes things much more complicated and expensive, of course. Early work by Guoting Chen, who used computation in series in a single parameter [6] does not seem to have been improved upon. Some modern computer algebra systems simply give up when asked to compute the JCF of a matrix bigger than 5×5 that contains a parameter as we showed in Section 4.1.

This present paper attempts to strengthen the direct approach and improve the capabilities of CAS to find the JCF of a matrix that contains parameters. Our reasoning is that to use either the versal form or the Frobenius form or Carleman linearization [7] requires educating the user, which while in principle is the best idea, in practice can lead to suboptimal results (such as the user refusing to be educated). So if the user wants the JCF, let's try to give it to them. This can have an advantage, in that the user will learn about the special cases, and those may be what was actually needed.

There has been a significant body of computational algebraic work relevant to this problem, in computing the Frobenius form, the Zigzag form, and the Smith form [32, 33] but relatively few works [1, 6] on matrices with parameters. The difficulty appears to be combinatorial growth in the number of possible different cases. In the context of solving parametric linear systems, not eigenvalues, a significant amount of work has been done [2, 4, 9, 10, 23, 31]. Parametric nonlinear systems are studied in [26, 28, 37] and the references therein.

4.3 Preliminaries

Sections 4.3.1 and 4.3.2 gather the basic concepts and results from polynomial algebra that are needed in this paper. Meanwhile, Sections 4.3.3 and 4.3.4 review the notions of the Frobenius canonical form and the Jordan canonical form.

4.3.1 Regular chain theory

Let $\mathbb{K}$ be a field and $\overline{\mathbb{K}}$ its algebraic closure. Let $X_1 < \cdots < X_s$ be $s \geq 1$ ordered variables. We denote by $\mathbb{K}[X]$ the ring of polynomials in the variables $X = X_1, \ldots, X_s$ with coefficients in $\mathbb{K}$. For $F \subset \mathbb{K}[X]$, we denote by $\langle F \rangle$ and $V(F)$, the ideal generated by F in $\mathbb{K}[X]$ and the algebraic set of $\overline{\mathbb{K}}^s$ consisting of the common roots of the polynomials of F. For a non-constant polynomial $p \in \mathbb{K}[X]$, the greatest variable of p is called the *main variable* of p and denoted $\mathrm{mvar}(p)$, and the leading coefficient of p w.r.t. $\mathrm{mvar}(p)$ is called the *initial* of p, denoted by $\mathrm{init}(p)$. The Zariski closure of $W \subseteq \overline{\mathbb{K}}^s$, denoted by $\overline{W}$, is the intersection of all algebraic sets $V \subseteq \overline{\mathbb{K}}^s$ such that $W \subseteq V$ holds.

A set $T \subset \mathbb{K}[X] \setminus \mathbb{K}$ is *triangular* if $\mathrm{mvar}(t) \neq \mathrm{mvar}(t')$ holds for all $t \neq t'$ in T. Let h_T be the product of the initials of the polynomials in T. We denote by $\mathrm{sat}(T)$ the *saturated ideal* of T; if T is empty, then $\mathrm{sat}(T)$ is defined as the trivial ideal $\langle 0 \rangle$, otherwise it is the ideal $\langle T \rangle : h_T^\infty$. The *quasi-component* $W(T)$ of T is defined as $V(T) \setminus V(h_T)$. The following property holds: $\overline{W(T)} = V(\mathrm{sat}(T))$.

A triangular set $T \subset \mathbb{K}[X]$ is a *regular chain* if either T is empty, or the set T' is a regular chain, and the initial of p is regular (that is, neither zero nor zero divisor) modulo $\mathrm{sat}(T')$, where p is the polynomial of T with largest main variable, and $T' := T \setminus \{p\}$. Let $T \subset \mathbb{K}[X]$ be a regular chain. If T contains s polynomials $t_1(X_1), t_2(X_1, X_2), \ldots, t_s(X_1, \ldots, X_s)$, then T generates a zero-dimensional ideal which is equal to the saturated ideal $\mathrm{sat}(T)$. If, in addition, the ideal $\mathrm{sat}(T)$ is prime (and, thus maximal in this case), then T is an encoding of the field extension $\mathbb{L} := \mathbb{K}[X]/\langle T \rangle$.

Let $H \subset \mathbb{K}[X]$. The pair $[T, H]$ is a *regular system* if each polynomial in H is regular modulo $\mathrm{sat}(T)$; the zero set of $[T, H]$, denoted by $Z(T, H)$, consists of all points of $\mathbb{K}^s$ satisfying $t = 0$ for all $t \in T$, $h \neq 0$ for all $h \in H \cup \{h_T\}$. A regular chain T, or a regular system $[T,H]$, is *square-free* if for all $t \in T$, the polynomial $\mathrm{der}(t)$ is regular w.r.t. $\mathrm{sat}(T)$, where $\mathrm{der}(t) = \frac{\partial t}{\partial v}$ and $v = \mathrm{mvar}(t)$.

The zero set S of an arbitrary system of polynomial equations and inequations is called a *constructible set* and can be decomposed as the union of the zero sets (resp. the Zariski closure of the zero sets) of finitely many square-free regular systems $[T_1,H_1], \ldots, [T_e, H_e]$. When this holds we have $S = Z(T_1, H_1) \cup \cdots \cup Z(T_e, H_e)$ (resp. $\overline{S} = \overline{Z(T_1, H_1)} \cup \cdots \cup \overline{Z(T_e, H_e)}$) and we say that $[T_1,H_1], \ldots, [T_e, H_e]$ is a *triangular decomposition* of S in the sense of Lazard and Wu (resp. in the sense of Kalkbrener).

We specify below a core routine thanks to which triangular decompositions can be computed. For more details about the theory of regular chains and its algorithmic aspects, we refer to [5].

Notation 1. The function $\mathsf{Squarefree\_RC}(p, T, H)$ computes a set of triples $((b_{i,1}, \ldots, b_{i,\ell_i}), T_i, H_i)$ with $1 \leq i \leq e$, such that $[T_1,H_1], \ldots, [T_e, H_e]$ are regular systems forming a triangular decomposition of $Z(T, H)$, and for all $1 \leq i \leq e$:

1. $b_{i,1}, \ldots, b_{i,\ell_i}$ are polynomials with the same main variable $v = \mathrm{mvar}(p)$ such that we have $p \equiv \prod_{j=1}^{\ell_i} b_{i,j}^j \mod \mathrm{sat}(T_i)$,

2. all discriminants $\mathrm{discr}(b_{i,j}, v)$ and all resultants $\mathrm{res}(b_{i,j}, b_{i,k}, v)$ are regular modulo $\mathrm{sat}(T_i)$, thus $\prod_{j=1}^{\ell_i} b_{i,j}^j$ is a square-free factorization of p modulo $\mathrm{sat}(T_i)$.

4.3.2 Regular chain representation of a splitting field

Let $p(x) \in \mathbb{K}[x]$ be a monic univariate polynomial. The *splitting field* of $p(x)$ over $\mathbb{K}$ is the smallest field extension of $\mathbb{K}$ over which $p(x)$ splits into linear factors,

$$p(x) = \prod_{i=1}^{\ell} (x - r_i)^{m_i} . \tag{4.1}$$

The set $\{r_1, \ldots, r_\ell\}$ generates $\mathbb{L}$ over $\mathbb{K}$. That is, $\mathbb{L} = \mathbb{K}(r_1, \ldots, r_\ell)$.

Assume that $p(x)$ is an irreducible, monic polynomial in $\mathbb{K}[x]$ of degree $n \geq 2$. To construct the splitting field $\mathbb{L}$ of $p(x)$ and compute the factorization of $p(x)$ into linear factors over $\mathbb{L}$, we proceed as follows.

1. Initialize $i := 1$, $y_i := x$, $\mathbb{L} := \mathbb{K}$, $T := \{\}$, $\mathcal{P} := \{\}$ and $\mathcal{F} := \{p\}$; the set $\mathcal{F}$ is assumed to maintain a list of univariate polynomials in y_i, irreducible over the current value of $\mathbb{L}$ and, of degree at least two,
2. While $\mathcal{F}$ is not empty do
   - (S1) pick a polynomial $f(y_i) \in \mathcal{F}$ over $\mathbb{L}$,
   - (S2) let α_i be a root of $f(y_i)$ (in the algebraic closure of $\mathbb{K}$),
   - (S3) replace $\mathbb{L}$ by $\mathbb{L}(\alpha_i)$, that is, by adjoining α_i to $\mathbb{L}$,
   - (S4) replace T by $T \cup \{t_i(y_1, \ldots, y_i)\}$, where the multivariate $t_i(y_1, \ldots, y_i)$ is obtained from $f(y_i)$ after replacing the algebraic numbers $\alpha_1, \ldots, \alpha_{i-1}$ with the variables $y_1, \ldots, y_{i-1}$,
   - (S5) replace $\mathcal{P}$ by $\mathcal{P} \cup \{x - y_i\}$,
   - (S6) factor $f(y_i)$ into irreducible factors over $\mathbb{L}$, then add the obtained factors of degree 1 (resp. greater than 1) to $\mathcal{P}$ (resp. $\mathcal{F}$); when adding a factor to $\mathcal{P}$, replace $\alpha_1, \ldots, \alpha_{i-1}$ with $y_1, \ldots, y_{i-1}$; when adding a factor to $\mathcal{F}$, replace y_i with y_{i+1}.
   - (S7) if $\mathcal{F}$ is not empty then $i := i + 1$.
3. Set $s := i$ and return $(s, T, \mathcal{P})$.

At the end of this procedure, the set T is a regular chain in the polynomial ring $\mathbb{K}[y_1, \ldots, y_s]$ generating a maximal ideal such that $\mathbb{K}[y_1, \ldots, y_s]/\langle T \rangle$ is isomorphic to the splitting field $\mathbb{K}(p)$ of $p(x)$. This procedure can be derived from S. Landau's paper [25]; note that the factorization at Step (S6) can be performed, for instance, by the algorithm of B. Trager [34]. Example: with $p(x) = x^3 - 2$, one can find $T = \{y_1^3 - 2, y_2^2 + y_1 y_2 + y_1^2\}$ and

$\mathcal{P} = \{x - y_1, x - y_2, x + y_2 + y_1\}$.

4.3.3 The Frobenius canonical form

Throughout the sequel of this section, we denote by A a square matrix of dimension n with entries in a field $\mathbb{K}$.

Let $p(x) = x^n + a_{n-1}x^{n-1} + \cdots + a_1 x + a_0$ be a monic polynomial in $\mathbb{K}[x]$. The *Frobenius companion matrix*[2] of $p(x)$ is a square $n \times n$ matrix of the form

$$C(p(x)) = \begin{bmatrix} 0 & 0 & \cdots & 0 & -a_0 \\ 1 & 0 & \cdots & 0 & -a_1 \\ 0 & 1 & \cdots & 0 & -a_2 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & 1 & -a_{n-1} \end{bmatrix}. \tag{4.2}$$

A matrix $F \in \mathbb{K}^{n \times n}$ is said to be in *Frobenius (rational) canonical form* if it is a block diagonal matrix where the blocks are companion matrices of monic polynomials $\psi_i(x) \in \mathbb{K}[x]$

$$F = \bigoplus_{i=1}^{m} C(\psi_i(x)) \tag{4.3}$$

such that $\psi_{i-1} \mid \psi_i$ for $i = 1, \ldots, m-1$. The polynomials ψ_i are called the *invariant factors* of F. Further details can be found in [12, 15, 22].

We recall a few properties.

1. Every companion matrix is in Frobenius canonical form.
2. For all $i = 1, \ldots, m$, the companion matrix $C(\psi_i)$ is non-derogatory[3].
3. There exists a non-singular matrix $Q \in \mathbb{K}^{n \times n}$ such that $F := Q^{-1}AQ$ is in Frobenius canonical form. The matrix F is called the *Frobenius canonical form* of A and the matrices A and F are said to be *similar*. We note that A and F have the same invariant factors.
4. The polynomial ψ_1 is the *minimal polynomial* of F and the product $\prod \psi_i$ is the *characteristic polynomial* of F.

[2]There are many other companion matrices, but in this paper a "companion matrix" is a Frobenius companion matrix.

[3]The characteristic polynomial and the minimal polynomial coincide up to a factor of ± 1.

4.3.4 The Jordan canonical form

An element $\lambda \in \overline{\mathbb{K}}$ is an *eigenvalue* of A if it satisfies $\det(A - \lambda I_n) = 0$ where I_n is the identity matrix of dimension n. The *algebraic multiplicity* of an eigenvalue λ is its multiplicity as a root of the characteristic polynomial of A, and its *geometric multiplicity* is the dimension of the null space of $A - \lambda I_n$.

Let $F = \operatorname{diag}(C(\psi_1), C(\psi_2), \ldots, C(\psi_m))$ be the Frobenius form of A where $C(\psi_i)$ is the companion matrix of the ith invariant factor ψ_i of A. We note that the geometric multiplicity of an eigenvalue λ of A is the number of invariant factors that λ is a solution for. Thus, the Frobenius form of A tells us both the algebraic and geometric multiplicities of all eigenvalues of A.

A matrix is called a *Jordan block* of dimension n if it is zero everywhere except for ones along its super-diagonal, and a single value λ along its main diagonal. A Jordan block has one eigenvalue λ with geometric multiplicity 1 and algebraic multiplicity n. We use the notation $\mathrm{JBM}_n(\lambda)$ to denote a Jordan block of dimension n with eigenvalue λ.

Let F be a matrix in Frobenius form as in Equation (4.3). The *Jordan canonical form* of F is given by

$$J = \bigoplus_{i=1}^{m} \mathrm{JCF}(C(\psi_i(x))) \tag{4.4}$$

where $\mathrm{JCF}(C(\psi(x)))$ is the Jordan form of a companion matrix of $\psi(x)$, see Chapter VI, §6 of [12] for a proof.

4.4 JCF Over a Splitting Field

4.4.1 Jordan form of a companion matrix

Let $\psi(x) \in \mathbb{K}[x]$ be a univariate monic polynomial of degree n. Let $\mathbb{L}$ be the splitting field of $\psi(x)$ over $\mathbb{K}$. Let $C = C(\psi(x))$ be the companion matrix of $\psi(x)$. Assume that the complete factorization into linear factors of $\psi(x)$ writes

$$\psi(x) = \prod_{i=1}^{\ell} (x - r_i)^{m_i} \tag{4.5}$$

where $r_i \in \mathbb{L}$ for $i = 1 \ldots \ell$ and $r_i \neq r_j$ for $i \neq j$. Then, the Jordan form of C is given by

$$J = \bigoplus_{i=1}^{\ell} \mathrm{JBM}_{m_i}(r_i) \tag{4.6}$$

where the entries of J are in $\mathbb{L}$. Thus, once the splitting field of $\psi(x)$ is computed, the Jordan canonical form of the companion matrix of $\psi(x)$ can be constructed.

Using the algorithm described in Section 4.3.2, the roots $r_1, \ldots, r_\ell$ of $\psi(x)$ are represented by the residue classes of multivariate polynomials $r_1(y_1, \ldots, y_s), \ldots, r_\ell(y_1, \ldots, y_s)$ modulo $\langle T \rangle$, since the regular chain $T = t_1(y_1), \ldots, t_s(y_1, \ldots, y_s)$ encodes the splitting field $\mathbb{K}(\psi)$ of $\psi(x)$ in the sense that this field is isomorphic to $\mathbb{K}[y_1, \ldots, y_i]/\langle T \rangle$. Therefore, the Jordan form of C is given by

$$\bigoplus_{i=1}^{\ell} \mathrm{JBM}_{m_i}(r_i(y_1, \ldots, y_s)) \tag{4.7}$$

together with the regular chain T.

4.4.2 Frobenius form to Jordan form

Let $F \in \mathbb{K}^{n \times n}$ be in Frobenius form, with $F = \mathrm{diag}(C(\psi_1), C(\psi_2), \ldots, C(\psi_m))$, where the polynomials ψ_i are the invariant factors of F. By Equation (4.4), the Jordan form of F is given by

$$J = \bigoplus_{i=1}^{m} \mathrm{JCF}(C(\psi_i)) \tag{4.8}$$

and a regular chain T defining the splitting field of ψ_1. This is, indeed, sufficient to compute all the entries of the JCF of F, since every subsequent polynomial ψ_i divides ψ_1.

4.4.3 Example

Let $\psi(x) = (x^3 + x^2 + x - 1)(x^2 + x + 1)^2$, where the coefficients are in $\mathbb{Q}$. Let C be the companion matrix of ψ. The JCF of C over the splitting field $\mathbb{L}$ of ψ over $\mathbb{Q}$ is

$$\begin{bmatrix} y_1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & y_3 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 - y_1 - y_3 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & y_2 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & y_2 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & -1 - y_2 & 1 \\ 0 & 0 & 0 & 0 & 0 & & -1 - y_2 \end{bmatrix}$$

where (y_1, y_2, y_3) are any point in the zero set $V(T)$ where T is

$$T = \{y_1^2 + (1 + y_3)y_1 + y_3^2 + y_3 + 1, y_2^2 + y_2 + 1, y_3^3 + y_3^2 + y_3 - 1\}.$$

4.5 JCF of a Matrix with Parameters

In this section we show how to compute a complete case discussion for the JCF of a matrix F in Frobenius form where the entries are polynomials in $\mathbb{K}[\alpha_1, \ldots, \alpha_s]$. Note that, as Arnol'd points out in [1], a parametric Frobenius form is continuous in its parameters, though its Jordan form may not be. Throughout this section, $T \subset \mathbb{K}[\alpha]$ will be a regular chain and $H \subset \mathbb{K}[\alpha]$ a set of polynomial inequations such that $[T,H]$ forms a regular system.

4.5.1 Square-free factorization of a parametric polynomial

Let $\alpha_1 < \cdots < \alpha_s$ be $s \geq 1$ ordered variables. Let $\mathbb{K}[\alpha] = \mathbb{K}[\alpha_1, \ldots, \alpha_s]$ be the ring of polynomials in the variables $\alpha = \alpha_1, \ldots, \alpha_s$. Let x be a variable. Let $\mathbb{K}[x]$ (resp. $\mathbb{K}[\alpha][x]$) be the ring of polynomials in x with coefficients in $\mathbb{K}$ (resp. $\mathbb{K}[\alpha]$). A polynomial $p(x; \alpha) \in \mathbb{K}[\alpha][x]$ is called a *univariate, parametric polynomial* in x and takes the form

$$p(x; \alpha) = a_n(\alpha)x^n + \cdots + a_1(\alpha)x + a_0(\alpha) \tag{4.9}$$

where the coefficients $a_i(\alpha)$ are polynomials in $\mathbb{K}[\alpha]$.

Let $p(x; \alpha) = \prod_{i=1}^{\ell} b_i(x; \alpha)^i$ be a square-free factorization of $p(x; \alpha)$, regarded as a univariate polynomial in $\mathbb{K}[\alpha][x]$. Then, the following properties must hold:

1. each polynomial $b_i(x; \alpha)$ is square-free as a polynomial in $\mathbb{K}[\alpha][x]$, and
2. the GCD of $b_i(x; \alpha)$ and $b_j(x; \alpha)$, as polynomials in $\mathbb{K}[\alpha][x]$, has degree zero in x, for all $1 \leq i < j \leq \ell$.

We note that each of the square-free factors $b_1, \ldots, b_\ell$ of $p(x; \alpha)$ is uniquely defined up to a multiplicative element of $\mathbb{K}[\alpha]$.

Definition 1. We say that the sequence of polynomials $b_1, \ldots, b_\ell$ *specializes well* at a point $\alpha^* = (\alpha_1^*, \ldots, \alpha_s^*) \in \overline{\mathbb{K}}^s$ whenever

1. the degree in x of the specialized polynomial $b_i(x; \alpha^*)$ is the same as the degree in x of b_i as a polynomial in $\mathbb{K}[\alpha][x]$, for all $1 \leq i \leq \ell$;
2. each specialized polynomial $b_i(x; \alpha^*)$ is square-free, as a polynomial in $\mathbb{K}[x]$, for all $1 \leq i \leq \ell$; and

3. the GCD of $b_i(x;\alpha^*)$ and $b_j(x;\alpha^*)$, as polynomials in $\mathbb{K}[x]$, has degree zero in x, for all $1 \le i < j \le \ell$.

From the theory of *border polynomials* [26, 28, 37] the following result holds.

Proposition 1. The set of points $\alpha \in \overline{\mathbb{K}}^s$ at which the sequence of polynomials $b_1, \ldots, b_\ell$ specializes well is the complement of the algebraic set given by

$$\left\{\bigcup_{i=1}^{i=e} V(\Delta_i)\right\} \ \cup\ \left\{\bigcup_{1\le i<j\le e} V(R_{i,j})\right\}, \tag{4.10}$$

where $\Delta_i := \mathrm{discr}(b_i(x;\alpha), x)$ denotes the discriminant of $b_i(x;\alpha)$ w.r.t. x and $R_{i,j} := \mathrm{res}(b_i(x;\alpha), b_j(x;\alpha), x)$ denotes the resultant of $b_i(x;\alpha)$ and $b_j(x;\alpha)$ w.r.t. x.

Definition 2. We call the *proviso* of the sequence of polynomials $b_1, \ldots, b_\ell$ the algebraic set (actually hypersurface) given by Equation (4.10) and denote it by $\mathsf{Proviso}(b_1, \ldots, b_\ell)$. We call the *square-free factorization with proviso* of $p(x;\alpha)$ the pair $(\prod_{i=1}^{\ell} b_i(x;\alpha)^i, \mathsf{Proviso}(b_1, \ldots, b_\ell))$.

We note that the zero set of the border polynomial of $p(x;\alpha)$ (in the sense [28, 37]) is usually defined whenever $p(x;\alpha)$ is square-free w.r.t. x, in which case it coincides with $\mathsf{Proviso}(b_1, \ldots, b_\ell)$.

We are now interested in obtaining a complete case discussion for the square-free factorization of $p(x;\alpha)$, that is, including the cases where $\alpha^* \in \mathsf{Proviso}(p(x;\alpha), x)$ holds. This can be achieved by using the function $\mathsf{Squarefree\_RC}(p, T, H)$ specified in Section 4.3.1.

4.5.2 JCF of a companion matrix with parameters

From now on, we assume that $\overline{\mathbb{K}}$ is the field $\mathbb{C}$ of complex numbers. Let $C \in \mathbb{K}[\alpha]^{n\times n}$ be a companion matrix with characteristic polynomial $\psi(x;\alpha) \in \mathbb{K}[\alpha][x]$. Let $\prod_{i=1}^{\ell} b_i(x;\alpha)^i$ be a square-free factorization of $\psi(x;\alpha)$. We observe that in the complement of $\mathsf{Proviso}(b_1, \ldots, b_\ell)$, the roots (in x) of $b_1, \ldots, b_\ell$, as functions of α, define continuous, disjoint graphs. Let us denote those functions by $\lambda_{i,1}, \ldots, \lambda_{i,n_i}$ corresponding to the polynomial b_i, for $1 \le i \le \ell$. Therefore, one can construct the JCF of C uniformly over the complement of $\mathsf{Proviso}(b_1, \ldots, b_\ell)$ as follows

$$\bigoplus_{i=1}^{\ell} \bigoplus_{j=1}^{n_i} \mathrm{JBM}_i(\lambda_{i,j})\,. \tag{4.11}$$

More generally, for a regular system $[T, H]$ let $((b_{i,1}, \ldots, b_{i,\ell_i}), T_i, H_i)$, with $1 \leq i \leq e$, be the output of $\mathsf{Squarefree\_RC}(\psi(x;\alpha), T, H)$ (in Notation 1). Then, for every $1 \leq i \leq e$, one can construct the JCF of C uniformly over $Z(T_i, H_i)$ (resp. $\overline{Z(T_i, H_i)}$) as the regular systems $[T_1,H_1], \ldots, [T_e, H_e]$ form a triangular decomposition of $Z(T, H)$ in the sense Lazard-Wu (resp. Kalkbrener).

4.5.3 Frobenius form to JCF with parameters

Let $F \in \mathbb{K}[\alpha]^{n\times n}$ be a matrix in Frobenius form with invariant factors $\psi_i(x;\alpha) \in \mathbb{K}[\alpha][x]$ for $1 \leq i \leq m$. Let $\prod_{i=1}^{\ell} b_i(x;\alpha)^i$ be a square-free factorization of the minimal polynomial, $\psi_1(x;\alpha)$. The JCF over the complement of $\mathsf{Proviso}(b_1, \ldots, b_\ell)$ is defined continuously for each companion matrix $C(\psi_i(x;\alpha))$, $1 \leq i \leq m$. This is a consequence of the property of invariant factors that each subsequent $\psi_i(x;\alpha)$ divides $\psi_1(x;\alpha)$.

In the sense of Lazard-Wu (resp. Kalkbrener), the construction of the JCF of $C(\psi_1(x;\alpha))$ defines a decomposition into the zero sets (resp. the Zariski closure of the zero sets) of finitely many square-free regular systems $[T_1,H_1], \ldots, [T_e, H_e]$. Over each regular system, the JCF of each companion matrix $C(\psi_i(x;\alpha))$ for $1 \leq i \leq m$ is defined continuously.

4.6 Experimentation

4.6.1 Maple implementation

We are actively developing a package called `ParametricMatrixTools` in MAPLE that implements algorithms for computations on matrices with parameters. The source for this package, including numerous examples, is available at `https://github.com/steventhornton/ParametricMatrixTools` and is compatible with the version of the `RegularChains` library included in MAPLE 2016 and later. The `ComprehensiveJordanForm` method implements the algorithm discussed in Section 4.5. Our implementation is based on the theory of regular chains and its MAPLE implementation, the `RegularChains` package. Further details can be found at `http://regularchains.org`.

For each of the examples that follow, we have first computed a full case discussion for the Frobenius form using the `ComprehensiveFrobeniusForm` routine in our package. The details of the Frobenius form implementation have been omitted and we are actively working to improve our current implementation.

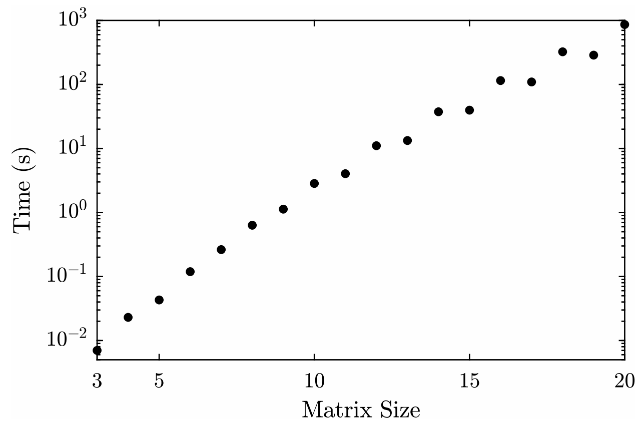


Figure 4.2: Time to compute the JCF of each Frobenius form in the full case discussion of the Frobenius form of the matrix in section 4.6.2. For all n, the Frobenius form splits into two cases: $\rho = 0$ and $\rho \neq 0$. The JCF is computed over each of these branches. Note the exponential growth. Timing was done on a 2016, 3.3GHz quad-core Intel Core i7 iMac with 16GB of RAM using MAPLE 2016.2.

4.6.2 Kac-Murdock-Szegö matrices

The inverse matrix $K_n^{-1}(\rho)$ from [35] is

$$\frac{1}{1-\rho^2}\begin{bmatrix} 1 & -\rho & 0 & \cdots & 0 & 0 & 0 \\ -\rho & 1+\rho^2 & -\rho & \cdots & 0 & 0 & 0 \\ 0 & -\rho & 1+\rho^2 & \cdots & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & 1+\rho^2 & -\rho & 0 \\ 0 & 0 & 0 & \cdots & -\rho & 1+\rho^2 & -\rho \\ 0 & 0 & 0 & \cdots & 0 & -\rho & 1 \end{bmatrix}.$$

The cost to compute a full case discussion of the JCF of $(1-\rho^2)K_n^{-1}(\rho)$ grows exponentially with n. See Figure 4.2.

4.6.3 The Belousov-Zhabotinskii reaction

The report [16] contains a very readable account of the famous B-Z reaction and its history. This is a chemical oscillator. In non-dimensional form with $\varepsilon = \delta = 1$ we have

$$\begin{aligned} \dot{x} &= qy - xy + x(1-x) \\ \dot{y} &= -qy - xy + fz \\ \dot{z} &= x - z\,. \end{aligned}$$

The equilibria include $x = z$ being a positive root of the quadratic

$$x(x-1+f)+q(x-1-f)=0\,. \tag{4.12}$$

The Jacobian at the equilibrium is

$$A=\begin{bmatrix}1-x-y & q-x & 0\\ -y & -(q+x) & f\\ 1 & 0 & -1\end{bmatrix} \tag{4.13}$$

and the Jordan form of A splits into many cases. One non-trivial example is

$$J=\begin{bmatrix}\alpha & 0 & 0\\ 0 & \beta & 1\\ 0 & 0 & \beta\end{bmatrix} \tag{4.14}$$

where

$$\alpha = \frac{1}{9994}(-81q^5+804q^4-3882q^3+12209q^2-6288q-59636)$$
$$\beta = \frac{1}{2}(-\alpha+3q-10)$$

under the following constraints on the indeterminates of A:

$$x=z=-2y$$
$$f=-1$$
$$(q^5-13q^4+86q^3-359q^2+911q-742)z-4q^2-8=0$$
$$q^6-15q^5+112q^4-531q^3+1633q^2-2564q+1492=0\,.$$

There are real values of q satisfying this equation, and hence this case is real.

4.6.4 Nuclear magnetic resonance

In [19], Section 2.2, we find a concise description of an application of the matrix exponential to solve the so-called Solomon equations

$$\dot{M}=-RM,\quad M(0)=I\quad\text{by}\quad M(t)=e^{-Rt}\,. \tag{4.15}$$

Here R is a symmetric, diagonally dominant matrix called the relaxation matrix, and M is the matrix of intensities. Suppose R is in fact tridiagonal, with ones on the sub- and super-diagonals, and diagonal parameters $|r_i| > 1$. Using MAPLE's built-in `MatrixExponential` gets answers (e.g. when the dimension n is 3) but we are not convinced that the generic answer returned is correct, always. So we try computing the JCF. Doing so, we find that indeed there are special cases that the generic code missed.

For example, when R is of dimension 3, the JCF of R is

$$\begin{bmatrix} (r_1+r_2+r_3)/3 & 1 & \\ 0 & (r_1+r_2+r_3)/3 & 1 \\ 0 & 0 & (r_1+r_2+r_3)/3 \end{bmatrix} \tag{4.16}$$

when

$$r_1^2 + r_2^2 + r_3^2 - r_1 r_2 - r_1 r_3 - r_2 r_3 + 6 = 0 \tag{4.17}$$

$$((r_1 - r_3)^2 - 1)((r_1 - r_3)^2 + 8) = 0\,. \tag{4.18}$$

When $\mathrm{discr}(\det(R - \lambda I)) \neq 0$ the JCF is simply $\mathrm{diag}(\lambda_1, \lambda_2, \lambda_3)$ for the distinct eigenvalues $\lambda_1, \lambda_2, \lambda_3$. And for the remaining parameter values, the JCF consists of a Jordan block of dimension 2 with eigenvalue λ_1, and a Jordan block of dimension 1 with eigenvalue λ_2 for $\lambda_1 \neq \lambda_2$. The only case corresponding to real values of r_1, r_2, r_3 is the trivial diagonal case. In the cases where the JCF is not a diagonal matrix, the result computed by the `MatrixExponential` function in MAPLE contains discontinuities.

4.6.5 Bifurcation studies

The mathematical methods used in bifurcation studies are highly sophisticated, both symbolically and numerically. Tools used include normal forms and the action of symmetry groups. Consider the matrix

$$J = \begin{bmatrix} 0 & 2\rho & 0 \\ a & 2\beta & 2v \\ b & -2v & 2\beta \end{bmatrix} \tag{4.19}$$

which is the Jacobian matrix of a dynamical system at equilibrium. The analysis of this system in [36] is quite complete, yet the evolution of trajectories near the equilibria, governed by

$$\xi' = J\xi, \quad \xi(0) = I \tag{4.20}$$

or $\xi = \exp(tJ)$, is of interest. When the JCF of J is nontrivial, one can anticipate phenomena such as greater sensitivity to modelling error, for instance. Our implementation is able to find a complete case discussion of the JCF, starting from the complete case discussion of the Frobenius form, in approximately 2 seconds. We find cases corresponding to each of the 5 possible Jordan structures for a 3×3 matrix with a total of 46 cases. Of the 46 cases, 14 are defined by polynomials of total degree greater than 4. The worst case contains a polynomial of degree 12 in the parameters with 19 terms.

One non-trivial case we were able to automatically identify is where the JCF of J is given by

$$\begin{bmatrix} 2\beta & 0 & 0 \\ 0 & \beta & 1 \\ 0 & 0 & \beta \end{bmatrix} \tag{4.21}$$

when $2\rho a + \beta^2 = 0$, $v = 0$, and a, ρ and β are non-zero.

4.7 Concluding Remarks

At the movie theatre watching a horror show, one often hears audience members warning the movie characters "don't go in there!". One could imagine a similar warning about the JCF. As Arnol'd says, "when investigating a family of matrices smoothly depending on parameters, then although each individual matrix can be reduced to Jordan normal form, it is unwise to do so, since in such an operation the smoothness (and also the continuity) relative to the parameters is lost."

Our approach here has been to ignore the warning, and go ahead anyway. We try to trap all the monsters. It is a complex problem and we claim only partial success (small monsters only!). But we believe this approach is already useful for some purposes. In particular, the discontinuities are sometimes the very quantities of interest and we can display them explicitly.

Bibliography

[1] V. I. Arnol'd. On matrices depending on parameters. *Russian Mathematical Surveys*, 26(2):29–43, 1971.

[2] C. Ballarin and M. Kauers. Solving parametric linear systems: an experiment with constraint algebraic programming. *ACM Sigsam Bulletin*, 38(2):33–46, 2004.

[3] T. Beelen and P. Van Dooren. Computational aspects of the Jordan canonical form. *Oxford Science Publishing, Oxford University Press*, pages 57—-72, 1990.

[4] P. A. Broadbery, T. Gómez-Díaz, and S. M. Watt. On the implementation of dynamic evaluation. In *Proceedings of the 1995 International Symposium on Symbolic and Algebraic Computation*, pages 77–84. ACM, 1995.

[5] C. Chen and M. Moreno Maza. Algorithms for computing triangular decomposition of polynomial systems. *Journal of Symbolic Computation*, 47(6):610–642, 2012.

[6] G. Chen. Computing the normal forms of matrices depending on parameters. In *Proceedings of the 1989 International Symposium on Symbolic and Algebraic Computation*, pages 244–249. ACM, 1989.

[7] G. Chen and J. Della Dora. Rational normal form for dynamical systems by Carleman linearization. In *Proceedings of the 1999 International Symposium on Symbolic and Algebraic Computation*, pages 165–172. ACM, 1999.

[8] R. M. Corless and D. J. Jeffrey. Well...it isn't quite that simple. *ACM Sigsam Bulletin*, 26(3):2–6, 1992.

[9] R. M. Corless and S. E. Thornton. A package for parametric matrix computations. In *Proceedings of the International Congress on Mathematical Software*, pages 442–449. Springer, 2014.

[10] G. M. Diaz-Toca, L. Gonzalez-Vega, and H. Lombardi. Generalizing Cramer's rule: Solving uniformly linear systems of equations. *SIAM Journal on Matrix Analysis and Applications*, 27(3):621–637, 2005.

[11] J. Frank, W. Huang, and B. Leimkuhler. Geometric integrators for classical spin systems. *Journal of Computational Physics*, 133(1):160–172, 1997.

[12] F. R. Gantmacher. *The theory of matrices. Vol. 1.* Chelsea Publishing Company, 1960.

[13] M. Giesbrecht. Nearly optimal algorithms for canonical matrix forms. *SIAM Journal on Computing*, 24(5):948–969, 1995.

[14] I. Gil. Computation of the Jordan canonical form of a square matrix (using the Axiom programming language). In *Proceedings of the 1992 International Symposium on Symbolic and Algebraic Computation*, pages 138–145. ACM, 1992.

[15] I. Gohberg, P. Lancaster, and L. Rodman. *Matrix polynomials*. SIAM, 2009.

[16] C. R. Gray. Analysis of the Belousov-Zhabotinskii reaction. *Rose-Hulman Undergraduate Mathematics Journal*, 3(1), 2002.

[17] E. Hairer, C. Lubich, and G. Wanner. *Geometric numerical integration: structure-preserving algorithms for ordinary differential equations*, volume 31. Springer, 2006.

[18] T. Hawkins. Weierstrass and the theory of matrices. *Archive for History of Exact Sciences*, 17(2):119–163, 1977.

[19] N. J. Higham. *Functions of matrices: theory and computation*. SIAM, 2008.

[20] L. Hogben. *Handbook of linear algebra*. CRC Press, 2016.

[21] R. A. Horn and C. R. Johnson. *Matrix analysis*. CUP, 2012.

[22] E. Kaltofen, M. Krishnamoorthy, and B. D. Saunders. Fast parallel algorithms for similarity of matrices. In *Proceedings of the fifth ACM Symposium on Symbolic and Algebraic Computation*, pages 65–70. ACM, 1986.

[23] D. Kapur. An approach for solving systems of parametric polynomial equations. *Principles and Practices of Constraint Programming*, pages 217–244, 1995.

[24] P. Kunkel and V. Mehrmann. *Differential-algebraic equations: analysis and numerical solution*, volume 2. European Mathematical Society, 2006.

[25] S. Landau. Factoring polynomials over algebraic number fields. *SIAM Journal on Computing*, 14(1):184–195, 1985.

[26] D. Lazard and F. Rouillier. Solving parametric polynomial systems. *Journal of Symbolic Computation*, 42(6):636–667, 2007.

[27] C. Moler and C. Van Loan. Nineteen dubious ways to compute the exponential of a matrix, twenty-five years later. *SIAM review*, 45(1):3–49, 2003.

[28] M. Moreno Maza, B. Xia, and R. Xiao. On solving parametric polynomial systems. *Mathematics in Computer Science*, 6(4):457–473, 2012.

[29] K. O'Meara, J. Clark, and C. Vinsonhaler. *Advanced Topics in Linear Algebra: Weaving Matrix Problems Through the Weyr Form*. Oxford University Press, 2011.

[30] P. Ozello. *Calcul exact des formes de Jordan et de Frobenius d'une matrice*. PhD thesis, Université Joseph-Fourier-Grenoble I, 1987.

[31] W. Y. Sit. An algorithm for solving parametric linear systems. *Journal of Symbolic Computation*, 13(4):353–394, 1992.

[32] A. Storjohann. An $\mathcal{O}(n^3)$ algorithm for the Frobenius normal form. In *Proceedings of the 1998 International Symposium on Symbolic and Algebraic Computation*, pages 101–105. ACM, 1998.

[33] A. Storjohann. *Algorithms for matrix canonical forms*. PhD thesis, Swiss Federal Institute of Technology Zurich, 2013.

[34] B. M. Trager. Algebraic factoring and rational function integration. In R. D. Jenks, editor, *SYMSAC 1976, Proceedings of the third ACM Symposium on Symbolic and Algebraic Manipulation, Yorktown Heights, New York, USA, August 10-12, 1976*, pages 219–226. ACM, 1976.

[35] W. F. Trench. Properties of some generalizations of Kac-Murdock-Szegö matrices. *Structured Matrices in Mathematics, Computer Science II Control, Signal and Image Processing (AMS Contemporary Mathematics Series)*, 281, 2001.

[36] S. A. Van Gils, M. Krupa, and W. F. Langford. Hopf bifurcation with non-semisimple 1:1 resonance. *Nonlinearity*, 3(3):825, 1990.

[37] L. Yang, X. Hou, and B. Xia. A complete algorithm for automated discovering of a class of inequality-type theorems. *Science in China Series F: Information Sciences*, 44(1):33–49, 2001.

Chapter 5

Bohemian Matrices and Their Eigenvalues

5.1 Introduction

A *family of Bohemian matrices* is a set of structured matrices where the entries are from a finite set of integers. Studying such matrices leads to many unanswered questions. Through extensive experimental work, we have discovered many properties of families of Bohemian matrices and their eigenvalues that lack obvious explanations. Our focus is fixed on matrices of low dimension, typically no more than 20×20. In some cases we consider structured matrices of higher dimension, although typically with constraints such that the number of free entries grows linearly in the dimension.

By plotting the distributions of the eigenvalues of all matrices in a Bohemian family over the complex plane, many interesting discrete structures appear. For example, in Figure 5.1, distinct "holes" appear in the distribution. Other families exhibit fractal-like structures and diffraction patterns. By studying these families in greater detail we are able understand why some structures appear. Many examples of the discrete structures that appear in the distributions of eigenvalues can be found at `http://www.bohemianmatrices.com/gallery/`.

Our experimental work is only possible thanks to advances in the processing power of common personal computers. This has allowed us to explore families containing upwards of 1 trillion matrices on a laptop. Through brute-force computation we are able to answer questions such as "how many 6×6 matrices with entries from the set $\{-1, +1\}$ are nilpotent?" The answer is 3,781,503. Further, these computations have helped us make connections between properties of matrices that we may not have made otherwise.

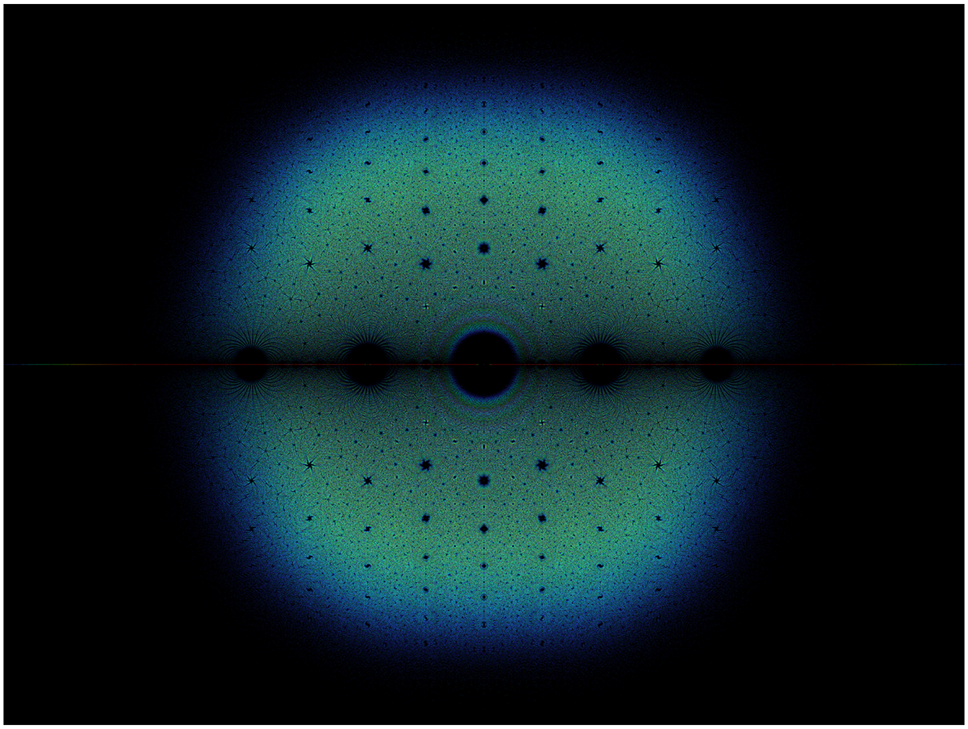


Figure 5.1: Density plot over the complex plane of the eigenvalues of all 5×5 matrices with entries from the set $\{-1, 0, +1\}$. The plot is viewed on $-4.13 - 3.1i$ to $4.13 + 3.1i$.

For example, the number of $n \times n$ matrices with entries from the set $\{-1, +1\}$ that are nilpotent is also the "number of acyclic digraphs (or DAGs) with n labelled nodes", as was discovered through the Online Encyclopedia of Integer Sequences (OEIS) [22] (sequence A003024).

The idea of visualizing the eigenvalues of random samples of matrices is not new. L. N. Trefethen [24] used this idea to visualize the pseudospectra of several test matrices. Related to the eigenvalues of matrices, many authors have studied the zeros of polynomials whose coefficients belong to discrete sets of integers. Early work by Odlyzko and Poonen [20] studied the zeros of polynomials with coefficients in $\{0, 1\}$. More recently, the distributions of the roots of Littlewood polynomials [18] (polynomials with coefficients ± 1 in the monomial basis) have been studied [1, 2, 3, 11, 21]. In Figure 5.2, the distribution of the roots of all degree 25 Littlewood polynomials are visualized. This is, of course, equivalent to the eigenvalues of a Bohemian family of (Frobenius) companion matrices with entries from the set $\{-1, +1\}$.

Our interest in studying these families started when we first generated a plot similar to that of Figure 5.1. This work was first presented as a poster [13] at the East Coast Computer Algebra Day (ECCAD) 2015. The poster mainly focused on the visualization of distributions of eigenvalues. Section 5.4 of this paper builds on many of those ideas.

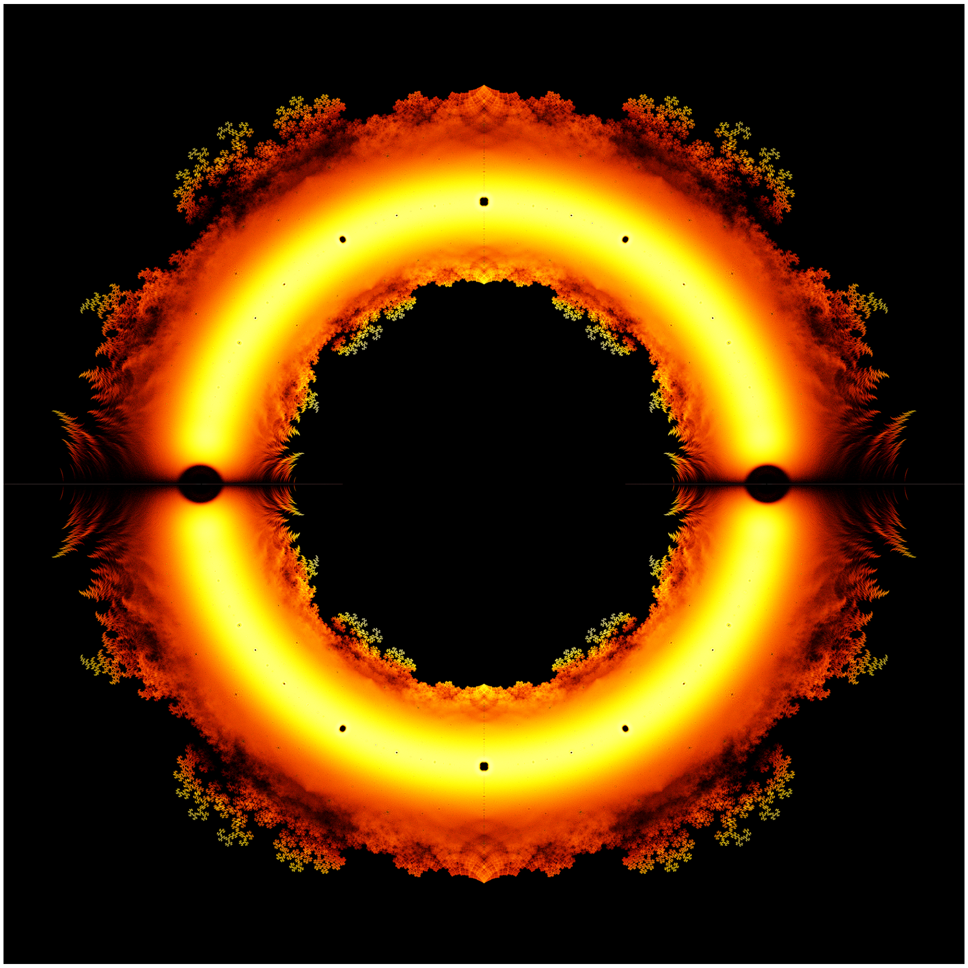


Figure 5.2: The roots of all degree 25 polynomials with ± 1 coefficients.

Some of our original interests in Bohemian families are presented in [12]. Many specialized cases have been previously studied in [4, 5, 6, 7, 8, 9].

We begin by introducing the notation used throughout this paper. Next, we explore symmetries in families of Bohemian matrices in Section 5.3. These symmetries have been helpful for reducing the number of matrices required to compute some properties of Bohemian families. Next, we discuss our MATLAB framework for visualizing the distributions of the eigenvalues of Bohemian matrices. Since eigenvalues are computed numerically, we can visualize the numerical error in computation of eigenvalues using our MATLAB framework. In Section 5.5, we show the numeric error in eigenvalue computation. We also discuss some example families that were discovered while generating plots of eigenvalue distributions where the MATLAB `eig` function failed to compute the eigenvalues. Finally, in Section 5.6, the Characteristic Polynomial Database (CPDB) is introduced. The methods used for computing distributions of characteristic polynomials are discussed.

Many properties of Bohemian families are included in the CPDB. The computational methods used for these properties are discussed. Finally, we end with 21 conjectures related to integer sequences arising from the properties of Bohemian families.

5.2 Terminology

Definition 5.2.1. A *family of Bohemian matrices* is a set of matrices of dimension n where the free entries are from the discrete set of bounded height P called the *population* of the Bohemian family. We denote a family of Bohemian matrices by $\mathcal{M}$.

Definition 5.2.2. A *Bohemian matrix* is a matrix belonging to a family of Bohemian matrices.

Definition 5.2.3. *Bohemian eigenvalues* are the eigenvalues of a family of Bohemian matrices.

Definition 5.2.4. An *eigenvalue exclusion zone* is a distinct region in the complex plane containing at most a single eigenvalue.

Definition 5.2.5. The *characteristic height* of a matrix is the height of its characteristic polynomial (orignially defined in [9].)

5.3 Symmetry in Bohemian Families

Symmetries within Bohemian families can help reduce the amount of computation required for analysis. When computing the distribution of characteristic polynomials of a Bohemian family, we noticed a large reduction in the number of characteristic polynomials compared to the family size. For example, in Table 5.1 we see that for the family of 5×5 matrices with population $\{-1, 0, +1\}$, each distinct characteristic polynomial corresponds to nearly 500,000 matrices on average. The set of distinct eigenvalues for this family shows a further reduction although not nearly as substantial. By exploring some symmetries in this family we are able to reduce the family size to a smaller set that approaches the dimension of the set of characteristic polynomials. Here we discuss several symmetries in Bohemian families.

5.3.1 Complex Conjugate Eigenvalue Symmetry

Proposition 5.3.1. *Let $P \subset \mathbb{Z}$ be the population of the Bohemian family $\mathcal{M}$. For every matrix $\mathbf{A} \in \mathcal{M}$, if λ is an eigenvalue of $\mathbf{A}$, $\overline{\lambda}$ is also an eigenvalue of $\mathbf{A}$.*

n	# of Matrices	# of Characteristic Polynomials	# of Distinct Eigenvalues
1	3	3	3
2	81	16	21
3	19,683	209	375
4	43,046,721	8,739	24,823
5	847,288,609,443	1,839,102	7,963,249

Table 5.1: For the Bohemian family of $n \times n$ matrices with population $\{-1, 0, +1\}$, the table reports the number of matrices (3^{n^2}), the number of distinct characteristic polynomials, and the number of distinct eigenvalues.

Remark 5.3.2. This symmetry presents itself as a reflection over the real axis when plotting the eigenvalues of the family, see Figure 5.1 for example.

5.3.2 Negation Symmetry

Proposition 5.3.3. *Consider a Bohemian family $\mathcal{M}$ of $n \times n$ matrices with population $P \subset \mathbb{Z}$ where the population is a symmetric set*[1]*. Let Λ be the set of eigenvalues for all matrices in $\mathcal{M}$. For every eigenvalue $\lambda \in \Lambda$, $-\lambda$ is also in Λ.*

Proof. Let $\mathbf{A}$ be a matrix in $\mathcal{M}$. $-\mathbf{A}$ must also be in $\mathcal{M}$ since P is a symmetric set. Let $p(x) = \det(x\mathbf{I} - \mathbf{A})$ be the characteristic polynomial of $\mathbf{A}$, and $q(x) = \det(x\mathbf{I} + \mathbf{A})$ be the characteristic polynomial of $-\mathbf{A}$. Let λ be an eigenvalue of $\mathbf{A}$. Then

$$\begin{aligned} p(\lambda) &= \det(\lambda\mathbf{I} - \mathbf{A}) & (5.1)\\ &= (-1)^n \det(-\lambda\mathbf{I} + \mathbf{A}) & (5.2)\\ &= (-1)^n q(-\lambda)\,. & (5.3) \end{aligned}$$

Hence, when $p(\lambda) = 0$, $q(-\lambda) = 0$ and $-\lambda$ is an eigenvalue of $-\mathbf{A}$. ♮

Remark 5.3.4. This symmetry presents itself as a reflection over the imaginary axis when plotting the eigenvalues of the family, see Figure 5.1 for example.

5.3.3 Rhapsodic Matrices

Definition 5.3.5. A matrix $\mathbf{A}$ in a Bohemian family $\mathcal{M}$ is said to be a *strictly rhapsodic* matrix if there exists a matrix $\mathbf{B} \in \mathcal{M}$ such that $\mathbf{A}^{-1} = \mathbf{B}$.

[1] A symmetric set S is a set such that if $s \in S$, $-s \in S$.

Definition 5.3.6. A Bohemian family $\mathcal{M}$ is said to be a *strictly rhapsodic* family if all matrices $\mathbf{A} \in \mathcal{M}$ are strictly rhapsodic.

Definition 5.3.7. Two matrices $\mathbf{A}$ and $\mathbf{B}$ are said to be *similar* (denoted $\mathbf{A} \sim \mathbf{B}$) if there exists an invertible matrix $\mathbf{Q}$ such that $\mathbf{B} = \mathbf{Q}^{-1}\mathbf{A}\mathbf{Q}$.

Remark 5.3.8. Similar matrices have the same Jordan canonical form and hence the same set of eigenvalues. They share many other properties including their rank, characteristic polynomial, and minmal polynomial.

Definition 5.3.9. A matrix $\mathbf{A}$ in a Bohemian family $\mathcal{M}$ is said to be a *non-strictly rhapsodic* matrix if there exists a matrix $\mathbf{B} \in \mathcal{M}$ such that $\mathbf{A}^{-1} \sim \mathbf{B}$.

Definition 5.3.10. A Bohemian family $\mathcal{M}$ is said to be a *non-strictly rhapsodic* family if all matrices $\mathbf{A} \in \mathcal{M}$ are non-strictly rhapsodic.

n	# of Matrices	# of Strictly Rhapsodic	# of Non-strictly Rhapsodic
1	3	2	2
2	81	40	40
3	19,683	4,656	6,528
4	43,046,721	2,808,192	9,175,104

Table 5.2: For the Bohemian family of $n \times n$ matrices with population $\{-1, 0, +1\}$, the table reports the number of matrices (3^{n^2}), the number of strictly rhapsodic matrices, and the number of non-strictly rhapsodic matrices.

Proposition 5.3.11. *If $p(x)$ is the characteristic polynomial of an invertible matrix $\mathbf{A}$, the characteristic polynomial of $\mathbf{A}^{-1}$ is $(-1)^n \frac{\mathrm{rev}(p(x))}{\det \mathbf{A}}$.*

Proposition 5.3.12. *Let $\mathcal{M}$ be a strictly rhapsodic family and let Λ be the set of eigenvalues of all matrices in $\mathcal{M}$. For every eigenvalue $\lambda \in \Lambda$, λ^{-1} is also in Λ.*

Proof. Let $p(x)$ be the characteristic polynomial of a matrix $\mathbf{A} \in \mathcal{M}$. Since all matrices in $\mathcal{M}$ are strictly rhapsodic, $\mathbf{A}^{-1}$ is also in $\mathcal{M}$. By Proposition 5.3.11, the characteristic polynomial of $\mathbf{A}^{-1}$ is $q(x) = \frac{\mathrm{rev}(p(x))}{\det \mathbf{A}}$. Let λ be a root of $p(x)$. Then

$$q(\lambda^{-1}) = \frac{\mathrm{rev}(p(\lambda^{-1}))}{\det \mathbf{A}} \tag{5.4}$$

$$= \frac{\lambda^n p(\lambda)}{\det \mathbf{A}} \tag{5.5}$$

$$= 0\,. \tag{5.6}$$

♮

Remark 5.3.13. This symmetry presents itself as a reflection over the unit circle when plotting the eigenvalues of the strictly rhapsodic family, see Figure 5.2 for example.

5.3.4 Permutations

Definition 5.3.14. Let $\mathcal{M}$ be a Bohemian family. A *permutation normal subset* of $\mathcal{M}$ is any set $\mathcal{M}_P \subseteq \mathcal{M}$ such that for every matrix $\mathbf{A} \in \mathcal{M}$, there exists a matrix $\mathbf{B} \in \mathcal{M}_P$ such that $\mathbf{A} = \mathbf{PBP}^{-1}$ for some permutation matrix $\mathbf{P}$. Additionally, for every matrix $\mathbf{A} \in \mathcal{M}_P$, there is no matrix $\mathbf{B} \in \mathcal{M}_P$ $(\mathbf{A} \neq \mathbf{B})$ such that $\mathbf{A} = \mathbf{PBP}^{-1}$ for any permutation matrix $\mathbf{P}$.

Proposition 5.3.15. *Let $\mathcal{M}$ be a Bohemian family and let $\mathcal{M}_P$ be a permutation normal subset of $\mathcal{M}$. The set of characteristic polynomials of $\mathcal{M}$ is the same as the set of characteristic polynomials of $\mathcal{M}_P$.*

Proof. For every matrix $\mathbf{A} \in \mathcal{M}_P$, there exists a permutation matrix $\mathbf{P}$ such that there is a matrix $\mathbf{B} \in \mathcal{M}$ where $\mathbf{B} = \mathbf{PAP}^{-1}$. Since $\mathbf{B}$ is similar to $\mathbf{A}$, they must have the same characteristic polynomial. Similarly, for every matrix $\mathbf{B} \in \mathcal{M}$, there exists a permutation matrix $\mathbf{P}$ such that there is a matrix $\mathbf{A} \in \mathcal{M}_P$ where $\mathbf{B} = \mathbf{PAP}^{-1}$. Since $\mathbf{B}$ is similar to $\mathbf{A}$, they must have the same characteristic polynomials. ♮

Proposition 5.3.16. *The set of eigenvalues of $\mathcal{M}$ is the same as the set of eigenvalues of $\mathcal{M}_P$.*

Proof. Since $\mathcal{M}$ and $\mathcal{M}_P$ have the same set of characteristic polynomials by Proposition 5.3.15, they must also have the same set of eigenvalues. ♮

Proposition 5.3.17. *For a Bohemian family $\mathcal{M}$ of $n \times n$ matrices with a permutation normal subset $\mathcal{M}_P$, $\#\mathcal{M}_P \geq \#\mathcal{M}/n!$.*

Proof. Let $\mathcal{M}$ be a Bohemian family such that for every matrix $\mathbf{A} \in \mathcal{M}$, and for each of the $n!$ permutation matrices, $\mathbf{A} \neq \mathbf{PAP}^{-1}$ except when $\mathbf{P} = \mathbf{I}$. A permutation normal subset for this family contains exactly $\#\mathcal{M}/n!$ matrices. ♮

Remark 5.3.18. Most families do not reach this bound. Let $\mathcal{M}$ be a Bohemian family that contains the identity matrix. For each of the $n!$ permutation matrices $\mathbf{P}$, $\mathbf{PIP}^{-1} = \mathbf{I}$. Hence, for this family the number of matrices in a permutation normal subset must be greater than $\#\mathcal{M}/n!$.

Computing a permutation normal subset of a Bohemian family is very expensive because it requires evaluating all $n!$ permutations of all matrices in the family. In Table 5.3 the size of permutation normal subsets are shown for an example family. The size of the permutation normal subsets do not quite reach the $\#\mathcal{M}/n!$ bound. In the 5×5 case $\#\mathcal{M}/\#\mathcal{M}_P = 847{,}288{,}609{,}443/7{,}071{,}729{,}867 \approx 119.8$ which is close to the $5! = 120$ bound. The method used for computing the size of the permutation normal subsets is discussed further in Section 5.6. Conjecture 21 in Section 5.6 states that the size of a permutation normal subset for this family is given by sequence A004105 on the OEIS for the "number of point-self-dual nets with $2n$ nodes [and the] number of directed 2-multigraphs with loops on n nodes".

n	$\#\mathcal{M}$	$\#\mathcal{M}_P$	$\log_{10}\left(1 - \frac{\#\mathcal{M}}{n!\#\mathcal{M}_P}\right)$	$\#\mathcal{M}_O$
1	3	3	$-\infty$	3
2	81	45	-1	54
3	19,683	3,411	-1.417	7,290
4	43,046,721	1,809,459	-2.058	7,971,615
5	847,288,609,443	7,071,729,867	-2.808	73,222,472,421

Table 5.3: Permutation symmetries for the Bohemian family $\mathcal{M}$ of $n \times n$ matrices with population $\{-1, 0, +1\}$. The $\#\mathcal{M}$ column gives the number of matrices in the family, $\#\mathcal{M}_P$ gives the number of matrices in a permutation normal subset of $\mathcal{M}$, and $\#\mathcal{M}_O$ gives the number of matrices in the subset of $\mathcal{M}$ with entries ordered along the diagonal. The $\log_{10}\left(1 - \frac{\#\mathcal{M}}{n!\#\mathcal{M}_P}\right)$ column shows the convergence of the permutation normal subset to the bound given in Proposition 5.3.17.

A less computationally expensive alternative to the permutation normal subset of a Bohemian family is to find a subset where the matrix entries along the diagonal are ordered according to some ordering $\prec$ over the population. This subset captures some of the permutations that a permutation normal subset finds but misses permutations within blocks of equal values along the diagonal. This subset can be constructed by directly sampling the matrices in the subset. The size of these subsets remains an order of magnitude larger than the permutation normal subsets as is shown in the last column of Table 5.3 for an example family.

Proposition 5.3.19. *Let $\mathcal{M}_O$ be the subset of a Bohemian family $\mathcal{M}$ of $n \times n$ matrices such that the diagonal entries of the matrices in $\mathcal{M}_O$ are ordered according to an ordering $\prec$. Let the population of $\mathcal{M}$ contain m elements. The number of matrices in $\mathcal{M}_O$ is*

$$\binom{n+m-1}{m} m^{n^2-n} \,. \tag{5.7}$$

Proof. The number of ordered combinations of the entries along the diagonal is the same as the number of multisets of length n from m entries in the population. Thus, there are $\binom{n+m-1}{m}$ possible orderings for the diagonal entries. The number of off-diagonal entries of the matrix is $n^2 - n$. The number of combinations of the m values in P for the $n^2 - n$ off diagonal entries is m^{n^2-n}. ♮

As an example, consider the family of 4×4 matrices with population $\{-1, 0, +1\}$ and order $-1 < 0 < 1$ over the population. The matrix $\mathbf{A}_1$ in Equation (5.8) is ordered but matrix $\mathbf{A}_2$ is not. Here $\mathbf{A}_1 = \mathbf{P}\mathbf{A}_2\mathbf{P}^{-1}$ for the permutation matrix $\mathbf{P}$.

$$\mathbf{A}_1 = \begin{bmatrix} -1 & 0 & 1 & 1 \\ 0 & 0 & -1 & -1 \\ 1 & -1 & 1 & 1 \\ 1 & 1 & 0 & 1 \end{bmatrix} \quad \mathbf{A}_2 = \begin{bmatrix} 1 & 1 & 1 & 0 \\ -1 & 0 & 0 & -1 \\ 1 & 0 & -1 & 1 \\ 1 & -1 & 1 & 1 \end{bmatrix} \quad \mathbf{P} = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \end{bmatrix} \tag{5.8}$$

5.3.5 Similar Matrices

Definition 5.3.20. Let $\mathcal{M}$ be a Bohemian family. A *similarity normal subset* of $\mathcal{M}$ is a set $\mathcal{M}_S \subseteq \mathcal{M}$ such that for every matrix $\mathbf{A} \in \mathcal{M}$, there exists a matrix $\mathbf{B} \in \mathcal{M}_S$ such that $\mathbf{A} \sim \mathbf{B}$. Additionally, for every matrix $\mathbf{A} \in \mathcal{M}_S$, there is no matrix $\mathbf{B} \in \mathcal{M}_S$ ($\mathbf{A} \neq \mathbf{B}$) such that $\mathbf{A} \sim \mathbf{B}$.

Proposition 5.3.21. *The set of Jordan canonical forms of a Bohemian family is isomorphic to a similarity normal subset of the family.*

Proof. Let $\mathcal{M}$ be a Bohemian family, let $\mathcal{M}_S$ be a similarity normal subset of $\mathcal{M}$ and let $\mathcal{J}$ be the set of Jordan canonical forms of all matrices in $\mathcal{M}$. For each matrix $\mathbf{J} \in \mathcal{J}$, there exists a matrix $\mathbf{S}$ such that $\mathbf{S}\mathbf{J}\mathbf{S}^{-1}$ is a matrix in $\mathcal{M}$. Since $\mathcal{J}$ is the set of Jordan forms, which are unique up to the ordering of the Jordan blocks, given a matrix $\mathbf{J}_1 \in \mathcal{J}$, there is no matrix $\mathbf{J}_2 \in \mathcal{J}$ ($\mathbf{J}_1 \neq \mathbf{J}_2$) such that $\mathbf{J}_1 \sim \mathbf{J}_2$. ♮

Since the set of Jordan canonical forms of a Bohemian family is isomorphic to a similarity normal subset, this can be a useful way to compute a similarity normal subset. Once we have the set of Jordan canonical forms, we can test if a new matrix (not necessarily from the same family) is similar to a matrix in the Bohemian family by computing its Jordan form and testing if it belongs to the set of Jordan forms. In Table 5.4 we give the size of the set of Jordan forms, or equivalently the size of any similarity normal subset of the example family. Details on the method used to count the number of distinct Jordan forms is discussed in detail in Section 5.6.

n	# of Matrices	# of Characteristic Polynomials	# of Distinct JCFs
1	3	3	3
2	81	16	19
3	19,683	209	225
4	43,046,721	8,739	8,971
5	847,288,609,443	1,839,102	

Table 5.4: Number of distinct characteristic polynomials and Jordan canonical forms (JCFs) for the family of $n \times n$ matrices with population $\{-1, 0, +1\}$. The number of distinct Jordan forms for the 5×5 family is currently unknown.

5.4 Visualizing Distributions of Bohemian Eigenvalues

When exploring a new family of Bohemian matrices, the first thing we typically look at is the distribution of the eigenvalues. These distributions are one of the main motivations for our further exploration into Bohemian families and are helpful for understanding some properties. Often these distributions display interesting discrete structures that we are unable to fully explain. Our exploration into Bohemian families has been inspired by these questions. Some of the questions we have considered follow.

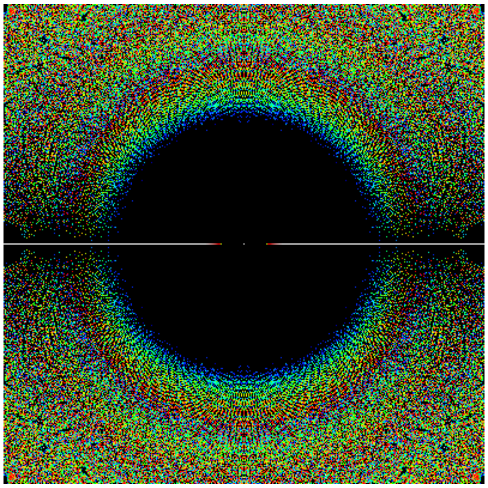
1. Many families have an eigenvalue exclusion zone centred at 0, see Figure 5.3a for example. What is size of this gap along the real line? What is the size of the gap in the imaginary plane?

2. What are the centres and radii of the eigenvalue exclusion zones? See Figure 5.3b for an example.

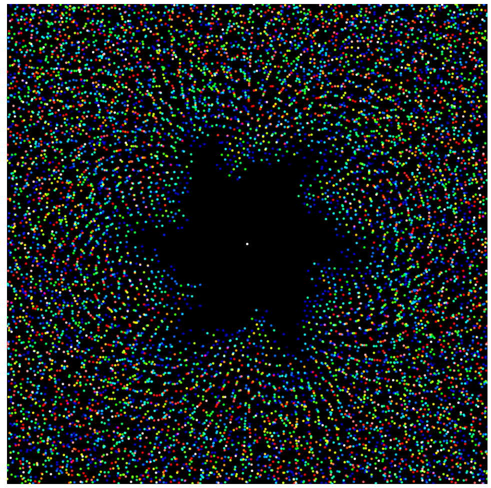
3. Some families (companion matrices of Littlewood polynomials, upper Hessenberg Toeplitz matrices, for example) show fractal like structures near the edges of the eigenvalue inclusion regions (see Figure 5.3c). Are these patterns truly fractals as $n \to \infty$?

4. Diffraction-like patterns appear in the densities of eigenvalues, see Figure 5.3d for an example. What is the cause of these patterns?

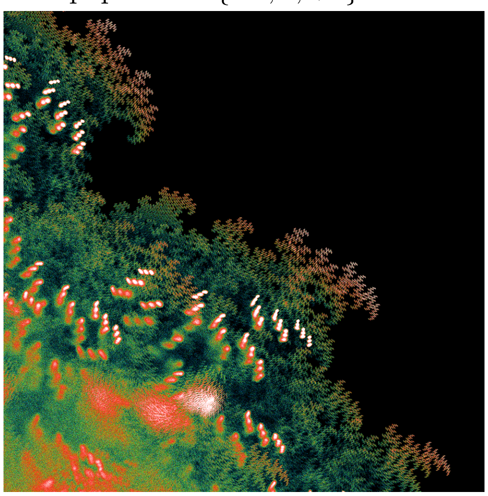
5. What is the radius of the spectrum? For many families this grows at a rate much slower than $\mathcal{O}(\sqrt{n})$. See Figure 5.4 for example.



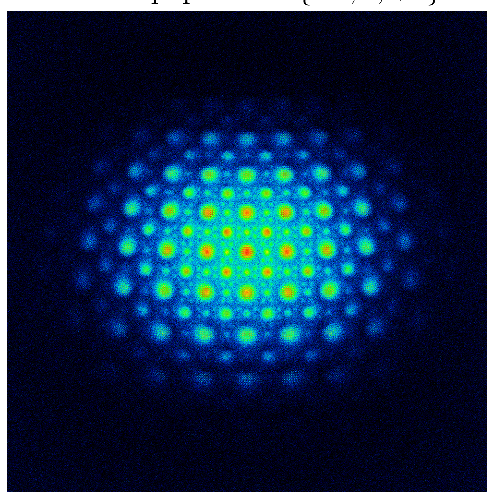
(a) Eigenvalue exclusion zone centred at 0 for the family of 5×5 matrices with population $\{-1, 0, +1\}$.



(b) Eigenvalue exclusion zone centred at $\frac{1}{2} + \frac{\sqrt{3}}{2}i$ for the family of 5×5 matrices with population $\{-1, 0, +1\}$.



(c) Fractal-like pattern appearing at the edge of the eigenvalue inclusion zone for the family of 25×25 upper Hessenberg Toeplitz matrices with main diagonal entries fixed at 0, subdiagonal entries fixed at 1, and population $\{-1, 0, +1\}$.



(d) Diffraction pattern appearing in the eigenvalues from the Bohemian family of 5×5 matrices with population $\{-20, -1, 0, +1, +20\}$.

Figure 5.3: Examples of structures appearing in the plots of Bohemian eigenvalues.

5.4.1 Plotting Eigenvalues in Matlab and Python

Visualizing the distributions of eigenvalues for a Bohemian family can be done in only a few lines of Matlab or Python code. Listings 5.1 and 5.2 are short scripts that will sample 1 million matrices from the family of 5×5 matrices with population $\{-1, 0, +1\}$, compute their eigenvalues and plot the density of eigenvalues over the complex plane in

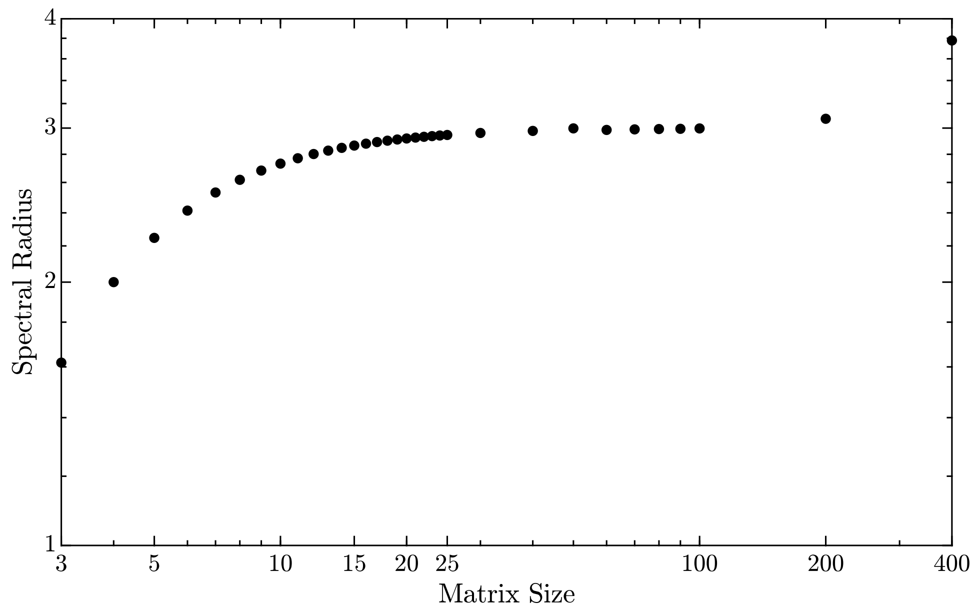


Figure 5.4: Radius of the spectrum for the Bohemian family of $n \times n$ upper Hessenberg matrices with a Toeplitz structure, entries on the main diagonal fixed at 0, and population $\{-1, +1\}$. Radius values for dimensions 3 to 25 are exact. For dimensions larger than 25 the radius has been approximated from a sample of 100 million matrices at each dimension. All computations were performed in double precision.

Matlab and Python respectively.

Although plots of the distributions of eigenvalues can be generated using only a few lines of code, these scripts lack the flexibility required to generate plots in general. To address this shortcoming, we have developed a Matlab framework for easily generating plots of the eigenvalues of families of Bohemian matrices. The framework, including numerous examples, is available on GitHub at `https://github.com/BohemianMatrices/BHIME-Project`. A Python version is under active development and is available at `https://github.com/BohemianMatrices/bohemian`. All of the plots of eigenvalues in this paper have been generating using the Matlab framework.

5.4.2 Overview of the `BHIME-Project` Framework

The `BHIME-Project` Matlab framework provides a simple interface to efficiently generating plots of Bohemian eigenvalues. A basic example for the Bohemian family of 5×5 matrices with population $\{-1, 0, +1\}$ is presented in Listing 5.3. An extensively commented version of this example is available in `Example1.m` on the GitHub repository. Figure 5.5 shows the resulting image.

```
L = zeros(5, 1e6);
for i=1:1e6
    A = randi([-1, 0, 1], 5, 5);
    L(:, i) = eig(A);
end
L = reshape(L, [5e6, 1]);
d = hist3([imag(L), real(L)], [1000, 1000]);
imagesc(log(d+1));
```

Listing 5.1: Example MATLAB script for generating a density plot over the complex plane of the eigenvalues for a sample of 1 million random matrices from the Bohemian family of 5×5 matrices with population $\{-1, 0, +1\}$.

```
import numpy as np
import matplotlib.pyplot as plt
A = np.random.choice([-1, 0, 1], size=(10**6, 5, 5))
L = np.linalg.eigvals(A).flatten()
H, x, y = np.histogram2d(L.real, L.imag, bins=1000)
plt.imshow(np.log(H.T+1), extent=[x[0], x[-1], y[0], y[-1]])
```

Listing 5.2: Example Python script for generating a density plot over the complex plane of the eigenvalues for a sample of 1 million random matrices from the Bohemian family of 5×5 matrices with population $\{-1, 0, +1\}$.

```
workingDir = '~/Real5x5_d3/';
g = @() randomMatrix([-1, 0, 1], 5);
generateRandomSample(g, workingDir);
pFilename = processData(workingDir);
processImage(pFilename, workingDir);
```

Listing 5.3: Simple example of using the `BHIME-Project` framework for plotting the eigenvalues for a sample of 5×5 matrices with population $\{-1, 0, +1\}$. The image produced from this example is given in Figure 5.5.

The framework breaks the plotting of eigenvalues into three main steps:

1. Compute the eigenvalues for a random sample of matrices,
2. Compute a two-dimensional histogram over the complex plane of eigenvalue densities, and

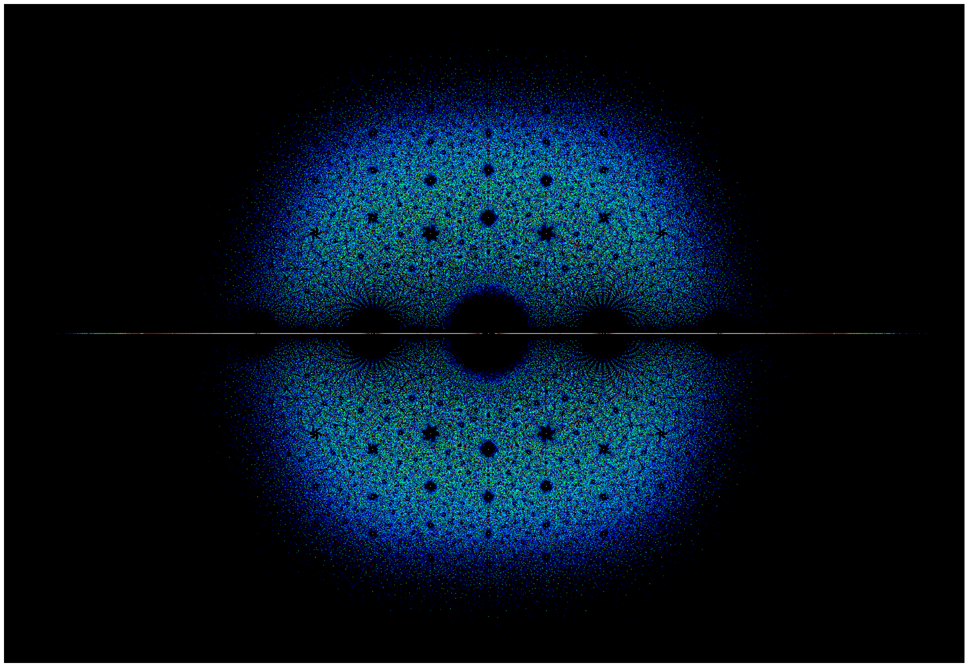


Figure 5.5: Density plot in the complex plane of the eigenvalues of a random sample of matrices from the Bohemian family of 5×5 matrices with population $\{-1, 0, +1\}$.

3. Generate the final image.

Each of these three steps corresponds to a single function within the framework. Each of these functions takes a directory (`workingDir`) as input and uses this as the base directory to save data files and images. The directory can be specified as either an absolute path, or a path relative to the current directory in Matlab. If the directory does not exist the framework will create it. It is highly recommended that the script file used to generate the images is stored in the working directory. Inside the working directory the following three folders are generated corresponding to the three steps of the image creation:

- `Data`
- `ProcessedData`
- `Images`

The `Data` directory is created by the `generateRandomSample` function and is where the computed eigenvalues are stored in Matlab `.mat` files. The `ProcessedData` directory is created by the `processData` function and stores the eigenvalue density matrix (two dimensional histogram over the complex plane) of the eigenvalues as a matrix in a Matlab `.mat` file. The `Images` directory is generated by the `processImage` function and will contain all output images in `png` format. Each of these directories will additionally contain a `README.txt` file that is automatically generated and appended to each time one of the three functions is called. The `README.txt` files contain general information that can be useful for reproducing the final image.

5.4.3 Computing Eigenvalues

Two approaches are available for computing eigenvalues. Matrices can either be randomly sampled from a family, or all matrices can be evaluated sequentially given a mapping from the positive integers to matrices in the family. The second approach is useful computing the eigenvalues of all matrices in a family when the size of the family is small. Random sampling is highly effective even for small samples ($\ll 1\%$). The distinct patterns and attributes of the distribution of the eigenvalues remain visible, see Figure 5.6 for example. Notice that fine diffraction pattern requires more sampling. Similarly, so do other subtle features.

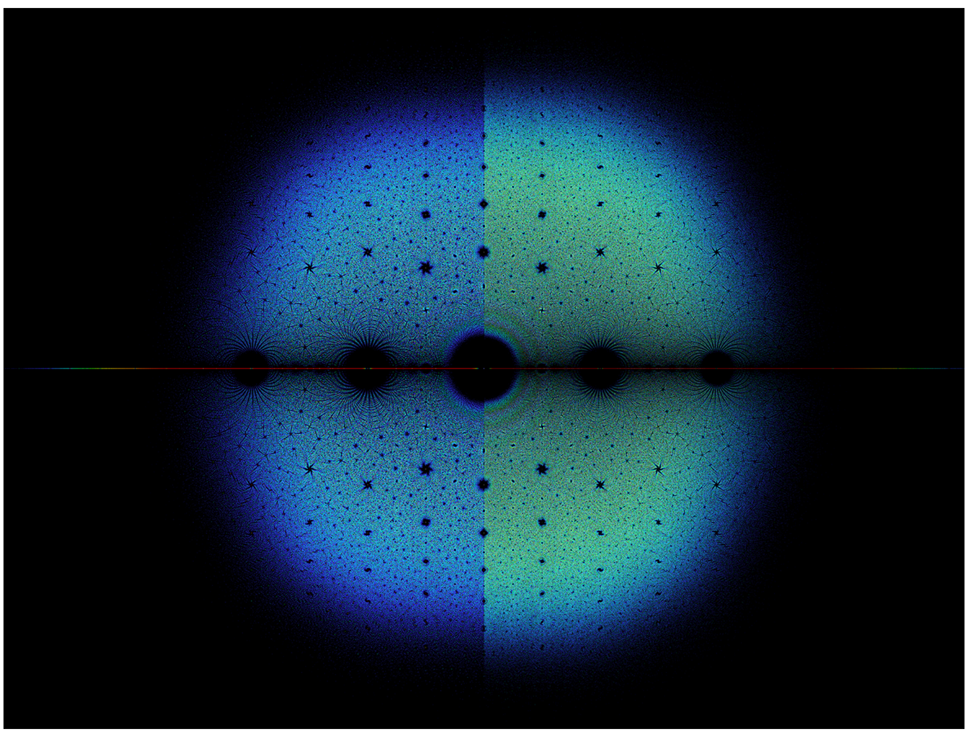


Figure 5.6: Eigenvalues from the family of 5×5 Bohemian matrices with population $\{-1, 0, +1\}$. The left half contains the eigenvalues from a sample of 100 million matrices and the right half contains the eigenvalues from all 3^{25} = 847,288,609,443 matrices in the family.

Randomly Sampling Matrices

The `generateRandomSample` function is the only function required for randomly sampling eigenvalues from Bohemian families. Its first input argument is a function that will return a single matrix at random from the Bohemian family each time it is called. Several template functions for common Bohemian families are available in the framework in the `src/matrixGenerators` directory. The following code will compute the eigenvalues of 10×10 Toeplitz matrices with population $\{-1, 1\}$ using the provided `randomToeplitzMatrix` function.

```
g = randomToeplitzMatrix([-1, 1], 10);
generateRandomSample(g, workingDir);
```

By default the `generateRandomSample` function will sample $\lfloor 1\,000\,000/n \rfloor$ matrices from a Bohemian family containing matrices of dimension n. An optional third argument can be provided to specify additional parameters for the random sampling. This argument should be a Matlab `struct` object where the field is the name of the option, and the value is the corresponding option value. Two options are available for controlling the number of matrices sampled. Eigenvalue data can be spread over multiple `.mat` files using the `numDataFiles` option (1 by default). This is useful when more eigenvalues than can fit in memory are being computed. The limit on the number of eigenvalues that can be computed in a family is thus limited by storage, not memory. The `matricesPerFile` option controls the number of matrices sampled for each `.mat` file. For example, the following code would generate 10 data files each containing the eigenvalues of 1 million matrices.

```
opts = struct('numDataFiles', 10, ...
              'matricesPerFile', 1000000);
g = randomToeplitzMatrix([-1, 1], 10);
generateRandomSample(g, workingDir, opts);
```

The `generateRandomSample` function saves the eigenvalue data in the `Data` subdirectory. By default, the data files are named `BHIME_i.mat` where `i` is the index of the file (1 through 10 in the above example). The file prefix can be controlled using the `filenamePrefix` option (`BHIME` by default). Each file contains a matrix of dimension `matricesPerFile` $\times\, n$ containing eigenvalues. Eigenvalues are always computed in double precision but stored in single precision by default to reduce file size. For most families the additional precision available with storing eigenvalues in double precision will not have any noticeable effect on the resulting images. The `dataPrecision` option can be set to `'double'` if double precision is required.

Eigenvalue computation is done in parallel using the `parfor` construct of the Parallel Computing Toolbox[2] in Matlab. Eigenvalues are computed in batches of size $\lfloor \texttt{matricesPerFile}/(10 \cdot \texttt{nCores}) \rfloor$. For example, if we are computing 1 million matrices per file, on a system with 16 cores, each parallel job will compute the eigenvalues of $\lfloor 1000000/(10 \cdot 16) \rfloor = 6250$ matrices.

Each time the `generateRandomSample` function is called it will create additional data files. So if it is called twice with the `numDataFiles` option set to 10, 20 files indexed 1 through 20 will be saved to the `Data` directory.

Eigenvalues of All Matrices

The `generateAllMatrices` function can be used to compute the eigenvalues of all matrices in a family. The first argument is an injective function that maps an integer between 1 and the number of matrices in the family, to a matrix in the family. An example function for mapping positive integers to matrices is given by the `matrixAtIndex` function in the `matrixGenerators` directory of the repository. The third argument to `generateAllMatrices` is an integer specifying the number of matrices in the family. The function will then iterate through all matrices and compute their eigenvalues.

As in the random sampling case, this function allows a fourth argument that specifies additional parameters. The `matricesPerFile` option can again be set to split the eigenvalues over multiple files. Since all matrices are sampled, the number of files generated is determined by the number of matrices in the family. For example, for the family of 4×4 matrices with population $\{-1, 0, +1\}$, which contains $3^{16} = 43{,}046{,}721$ matrices, if `matricesPerFile` is set to 10^6, 44 files files containing eigenvalues will be saved. The first 43 files will contain the eigenvalues of 1 million matrices and the final file will contain the eigenvalues of the last 46,721 matrices.

This approach is limited by the family size. For large families, the time to sample and compute eigenvalues, or the amount of storage required to store all eigenvalues, may prevent this from being an appropriate choice. For example, consider the family of 5×5 matrices with population $\{-1, 0, +1\}$. This family contains $3^{25} = 847{,}288{,}609{,}443$ matrices and 4,236,443,047,215 eigenvalues. Storing the eigenvalues (uncompressed) in

[2] `https://www.mathworks.com/products/parallel-computing.html`

single precision (32-bits per eigenvalue) would require more than 15TB or storage. Further, the time to compute all eigenvalues on a desktop computer[3] has been projected to take nearly 4 months. Due to the exponential growth of families of matrices, computing all eigenvalues using this approach can become intractable very quickly. If we instead looked at 4×4 matrices with population $\{-1, 0, +1\}$, the eigenvalue data requires only 656MB of storage and all eigenvalues can be computed in about 9 minutes (using the same computer as before.)

5.4.4 Plotting Eigenvalues

Once the eigenvalues have been computed, the next step is to plot their density in the complex plane. This is broken into two steps: first a two-dimensional histogram is computed over the complex plane and stored as a matrix, then the matrix is converted to an image. These two steps are performed using the `processData` and `processImage` functions respectively.

Computing Eigenvalue Density

The final output plot is a two-dimensional histogram of eigenvalue densities over the complex plane. Let a *density matrix* be the two-dimensional histogram over the complex plane. Each entry of the density matrix represents one pixel in the output image. To compute the density matrix we must first define the height of the image (number of rows of the density matrix) and the boundaries of the image. The width of the density matrix is computed from the height and boundaries as

$$\texttt{width} := \texttt{height} \cdot (\texttt{right} - \texttt{left})/(\texttt{top} - \texttt{bottom})$$

where `left`, `right`, `top`, and `bottom` are the left, right, top and bottom values for the boundaries respectively. For example, if we want to generate a plot over the complex area $[-2 - 3i, 4 + 5i]$ with height 1000, the width would be

$$1000 \cdot (4 - (-2))/(5 - (-3)) = 750\,. \tag{5.9}$$

The density matrix is then computed by counting the number of eigenvalues that fall inside each bin. When eigenvalues fall on the boundaries of a bin, they are included in the bin to their left (if on a left/right boundary), or the bin above (if on a top/bottom

[3]Computations run on a 2015 iMac with a 3.3 GHz Intel Core i7 processor and 16GB of RAM.

boundary).

The `processData` function computes the density matrix and saves it to a MATLAB `.mat` file. This file is saved inside the `processedData` subdirectory of the `workingDir`. Each time the `processData` function is called a new file will be generated.

By default, `processedData` will generate an image with a height of 1001 pixels and boundaries determined such that all eigenvalues in the eigenvalue data files are included in the density matrix.

Several options are available for controlling the output image. The height of the image in pixels, and the boundary of the image can be set using the `height` and `margin` options respectively. For example, if we wanted to view a plot on the rectangle $-3 \leq \mathrm{Im}(z) \leq 3$, $-4 \leq \mathrm{Re}(z) \leq 4$ with a height of 1000 pixels, the `processData` function can be called as follows.

```
margin = struct('left', -4, ...
                'right',  4, ...
               'bottom', -3, ...
                  'top',  3);
opts = struct('height', 1000, ...
              'margin', margin);
pFilename = processData(workingDir, opts);
```

Symmetry in the eigenvalues across the real and imaginary axes can be used to effectively double or quadruple the number of eigenvalues plotted. The `symmetryRe` and `symmetryIm` options are available for adding symmetry across the real and imaginary axes respectively. If $a + bi$ is an eigenvalue, the `symmetryRe` option adds $a - bi$ to the set of eigenvalues, when `symmetryIm` used, $-a + bi$ is added to the set of eigenvalues. When both are used, $a + bi$, $-a + bi$, and $-a - bi$ are all appended to the set of eigenvalues. These options should be used with care as adding the `symmetryRe` option for a family that does not have negation symmetry will produce an incorrect plot. For families of matrices with real entries, the `symmetryIm` option will not have any effect due to the complex conjugate eigenvalue symmetry. This option should only be used when exploring Bohemian families over the complex numbers when it is known that both $\mathbf{A}$ and $\overline{\mathbf{A}}$ are in the family for all matrices $\mathbf{A}$.

The density of purely real eigenvalues is typically much higher than the density off the real axis. For example, when plotting the eigenvalues of the family of 5×5 matrices with population $\{-1, 0, +1\}$ on a 2001×2001 pixel grid over the complex range $-4 - 4i$ to $4 + 4i$, the average density of bins that contain the real axis (excluding empty bins) is

1.023×10^9 while the average density of bins not on the real axis (excluding empty bins) is only 1.768×10^6. This can make selecting a colormap difficult and including the real eigenvalues often has minimal affect on the plot (since most of the structure in the plots is in the complex plane.) The `ignoreReal` option allows real eigenvalues to be excluded from the plot. This option works in conjunction with the `ignoreRealTol` option which specifies the tolerance such that an eigenvalue is considered real. If z is an eigenvalue, and $|\text{Re}(z)| <$ `ignoreRealTol`, then it is considered purely real and is excluded. By default, the `ignoreRealTol` is 10^{-8}.

Producing the Final Plot

Once the matrix of eigenvalue densities has been computed, the only remaining step is to convert this matrix of integer counts into an image. The `processImage` function converts the density matrix to the final output image. It begins by creating an `Images` subdirectory within the `workingDir`. Images are saved to this directory with the name `Image-i.png` where `i` is a positive integer selected such that no image is ever overwritten. The `processImage` function can be called as follows where the variable `pFilename` is the output of the `processData` function.

```
T = [  0,   0,   0;
      85,   0,   0;
     170,   0,   0;
     255,   0,   0;
     255,  85,   0;
     255, 170,   0;
     255, 255,   0;
     255, 255, 127;
     255, 255, 255;
     255, 255, 255]/255;
x = [0, 0.01, 0.04, 0.9, 0.13, 0.18, 0.23, 0.28, 0.4, 1.0];

processImage(pFilename, workingDir, T, x);
```

Several options are provided to allow maximal control over the final plot. The eigenvalue counts are always plotted on a log scale as the densities typically have a large range with some bins containing only a single eigenvalue while others (typically along the real axis or at 0) have density many orders of magnitude larger. This is illustrated in Figure 5.7

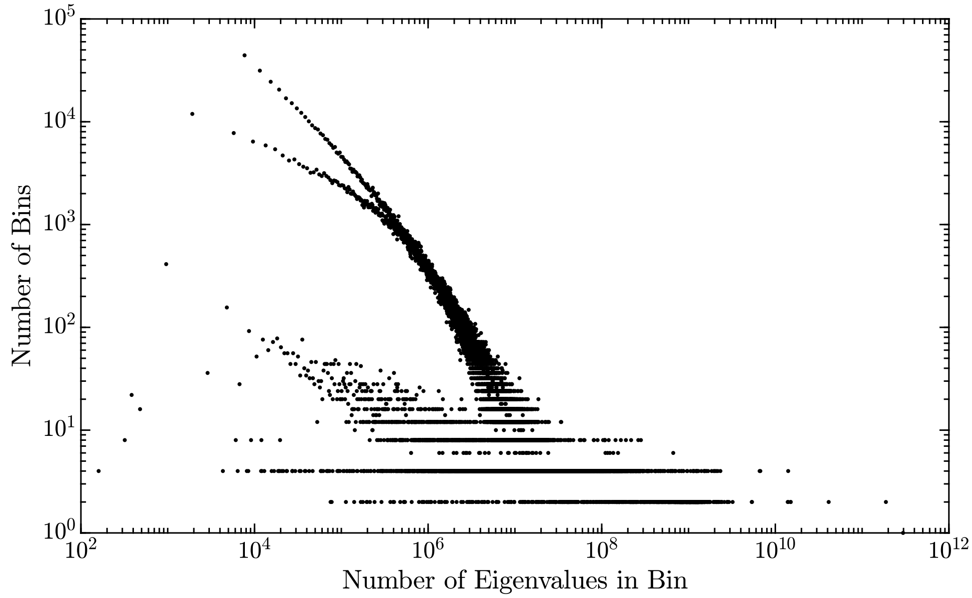


Figure 5.7: A histogram of the densities in the bins (pixels) for the family of 5×5 matrices with population $\{-1, 0, +1\}$ over the 2001×2001 pixel grid ranging from $-4-4i$ to $4+4i$. The bin with the highest density contains 296,330,735,533 eigenvalues and is the bin that contains 0. The lowest density bins only contain 160 eigenvalues and occurs 4 times in the density matrix. Bins containing 3,840 eigenvalues are the most common (excluding bins with no eigenvalues) and occur 78,440 times in the density matrix.

The colormap that is applied to the log densities (`T` and `x` in the above listing) is provided in two parts. First an $n \times 3$ matrix of RGB values where a row $[0, 0, 0]$ is black and $[1, 1, 1]$ is white defines the colors to use. A strictly increasing length n vector starting at 0 and ending at 1 is also required defining the relative locations of the colors. A default colormap and weighting are set if the `T` and `x` arguments to `processImage` are omitted.

5.4.5 Eigenvalue Computation Timing

Computing the eigenvalues of a dimension n matrix is an $\mathcal{O}(n^3)$ operation. Our framework aims to make the computation as efficient as possible by optimizing the use of parallel resources and avoiding copying data unnecessarily. On a 2015 iMac with a 3.3 GHz Intel Core i7 processor and 16GB of RAM, we can compute and plot the eigenvalues of 1 million 5×5 matrices with population $\{-1, 0, +1\}$ on a 2001×2001 pixel grid in under 25 seconds using only a single core. In Figure 5.8, the time to compute and plot the eigenvalues for a range of matrix dimensions is shown.

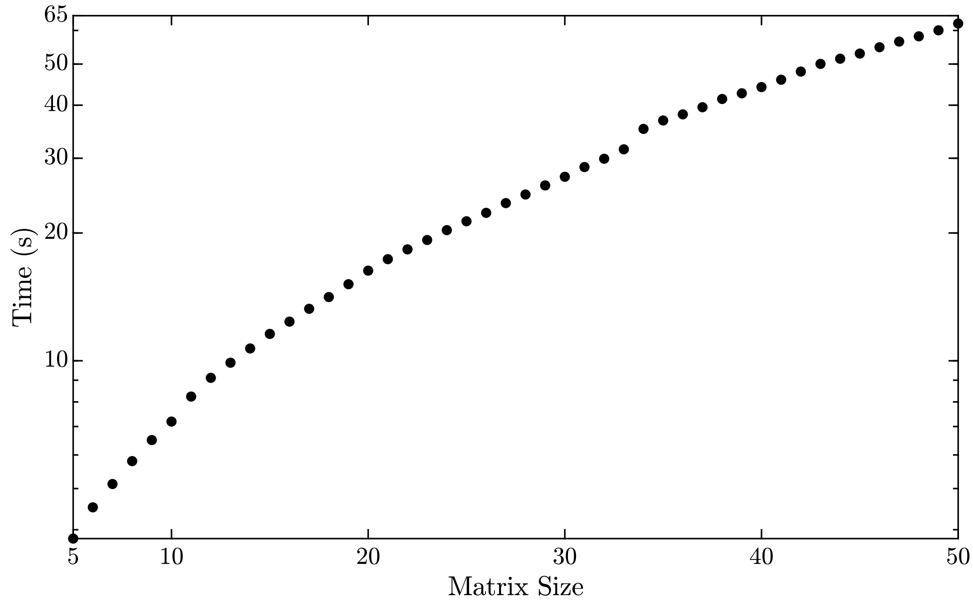


Figure 5.8: Time to compute and plot the eigenvalues of 1 million matrices for a range of matrix dimensions. Computations were done using 16 cores on an AMD Ryzen Threadripper 1950X 16 core/32 thread 3.7GHz with 64GB of RAM.

5.4.6 Language Comparison

The top high-level languages used in numerical analysis (MATLAB, Python, and Julia) all use LAPACK for computing eigenvalues, but how they interface the LAPACK routines affects the time to compute eigenvalues. While this may be minor for computing the eigenvalues of a single matrix, and negligible when computing the eigenvalues of large dimension matrices, this is magnified when computing the eigenvalues of many low dimension matrices. Table 5.5 shows a comparison of the time to compute the eigenvalues of 1 million 5×5 matrices with population $\{-1, 0, +1\}$.

The `numpy.linalg.eig` function in Python is the only eigenvalue solver in these languages that allows batched computation of eigenvalues. The Batched BLAS [15] API provides optimized implementations of the routines in BLAS for running the same computation on multiple matrices. Currently no implementation appears to be available for an eigenvalue solvers that uses the Batched BLAS API when computing eigenvalues for batches of matrices. The batched functionality of the `numpy.linalg.eig` function does however provide significant speedup by reducing the overhead to the LAPACK eigenvalue routines. The "Python (NumPy) Batched" row reports the time to compute the eigenvalues using the batched functionality of the `numpy.linalg.eig` function. The function allows input of an $m \times n \times n$ array (m = 1,000,000 in Table 5.5) where the eigenvalues of each slice are computed. This minimizes the overhead from calling the function 1,000,000 times as in the "Python (NumPy) Sequential" row. Sampling is also faster for the batched version as only a single $1{,}000{,}000 \times 5 \times 5$ array must be sampled rather than 1,000,000 5×5 matrices.

Language	Sample & Eigenvalues	Sample	Eigenvalues
MATLAB	13.113	4.650	6.397
Python (NumPy) Sequential	43.344	9.631	25.121
Python (NumPy) Batched	5.406	0.344	5.077
Julia	5.888	0.345	5.046

Table 5.5: Comparison of the time (in seconds) to sample and compute the eigenvalues of 1 million 5×5 matrices with population $\{-1, 0, +1\}$ in MATLAB, Python (NumPy), and Julia. The "Sample and Eigenvalues" column gives the time taken to sample and compute eigenvalues, the "Sample" column is the time to sample 1 million matrices, and the "Eigenvalues" column gives the time to compute the eigenvalues of a matrix 1 million times. The "Eigenvalues" column is based on computing the eigenvalues of a matrix 1 million times and is repeated for a sample of 100 matrices. The average time is given. The "Python (NumPy) Sequential" row gives the time to repeatedly sample 5×5 matrices and then compute their eigenvalues whereas the "Python (NumPy) Batched" row gives the time to sample a single array of dimension $1{,}000{,}000 \times 5 \times 5$ and use the batched functionality of the `numpy.linalg.eig` function to compute the eigenvalues of all matrices with only one function call. All scripts were run on a single thread on a computer with an AMD Ryzen Threadripper 1950X 16 core/32 thread 3.7GHz processor and 64GB of RAM using MATLAB R2018a, Python 3.6.2 (NumPy 1.13.1), and Julia 1.0.2.

5.5 A Test Class for Eigenvalue Solvers

Through our experimental work we estimate to have computed the eigenvalues of well over 1 trillion low dimension matrices. We have encountered a few matrices where the `eig` command in MATLAB fails to compute the eigenvalues. The error message MATLAB gives is "Algorithm did not succeed" and no eigenvalues are returned. The errors appear to be system dependent as a matrix where `eig` fails on one computer may be successful on another. Nevertheless we have been able to identify matrices that fail on all computers we have tried. These failures are rare appearing about once in every 8.6 million matrices that we sample based on a sample of 1.5 billion matrices. All of the matrices we have found that fail in MATLAB are successful in other languages (R, Python, and Julia). Further, computing the eigenvalues of $\mathbf{A}^T$, and $\mathbf{A} + \varepsilon\mathbf{I}$ both succeed for $|\varepsilon| > 2^{-52}$. We suspect the error is in the Intel MKL library and not the LAPACK function. To date we have not discovered any matrices where the eigenvalue solvers in Python (NumPy), Julia, or R fail, although we have spent significantly less time exploring eigenvalues with those languages.

The smallest real matrices we have encountered are of dimension 15. They are generated from a family of Bohemian matrices called *border* matrices. A *border* matrix is one where entries in the first and last row, and the first and last column are randomly populated from the population and all other entries are fixed at 0. As of Matlab 2018a, this family of matrices no longer causes an error in the `eig` function.

An example matrix that fails on an iMac (Matlab 2017b, Intel MKL version 11.3.1, and LAPACK version 3.5.0), but is successful on other computers is the 15×15 matrix

$$\mathbf{A}_{15} = \begin{bmatrix} 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & -1 & 1 & 0 & 1 & -1 & 1 & 1 & 1 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 \\ 0 & 1 & -1 & -1 & -1 & 0 & 0 & 1 & -1 & -1 & 1 & 1 & 1 & 0 & -1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}. \tag{5.10}$$

This matrices does not appear as a border matrix due to the balancing step used by the `eig` function. The matrix in Equation (5.10) is the result of calling the `balance` function on a border matrix (which the `eig` function does by default). Turning off the balancing step and computing the eigenvalues of the original border matrix succeeds. This matrix has characteristic polynomial $x^{12}(x^3 - x - 2)$ with an eigenvalue at 0 with algebraic multiplicity 12 and geometric multiplicity 11.

We have also found that the family of 12×12 tridiagonal matrices with population $\{-1, 1, i, -i, 20, -20, 20i, -20i\}$ fails for all version of Matlab up to and including 2018b. As in the previous case, these matrices appear to be system dependent. One example that fails on an iMac in Matlab 2018b is the matrix

$$\begin{bmatrix} 20 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ -20i & i & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 20 & -i & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & -i & -1 & -i & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 20i & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & -1 & 20i & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 20 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & -i & 20 & 20i & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 20 & 1 & 20 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 20i & -20 & -20i & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & -i & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -i & -i \end{bmatrix}. \qquad (5.11)$$

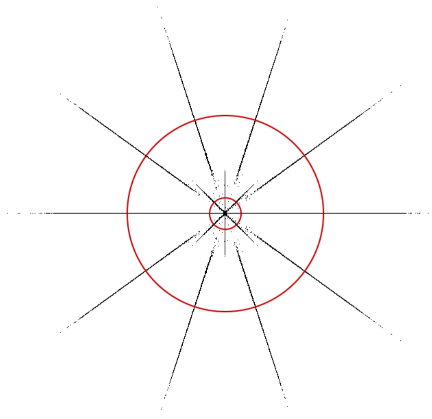
5.5.1 Numerical Error for Multiple Eigenvalues

Visualizing the eigenvalues of a family of Bohemian matrices can be useful for visually observing the numerical error surrounding high-multiplicity eigenvalues. Eigenvalues centred at 0 are commonly of high multiplicity. Figure 5.9 shows a closeup around 0 for two families of matrices. In both families a distinctive star shape appears. The outermost star shape contains $2n$ rays where n is the dimension of the matrix. The eigenvalues that form this part of the star are from matrices with an eigenvalue at 0 of multiplicity n. The next smaller star has $2(n-1)$ rays and is formed by matrices with an eigenvalue at 0 of multiplicity $n-1$. The numeric error in the eigenvalues from matrices with eigenvalues at 0 of multiplicity m is $\varepsilon^{1/m}$. This appears as the radius of the $2m$-pointed stars in the figures. The symmetry is indicative of rounding errors[4].

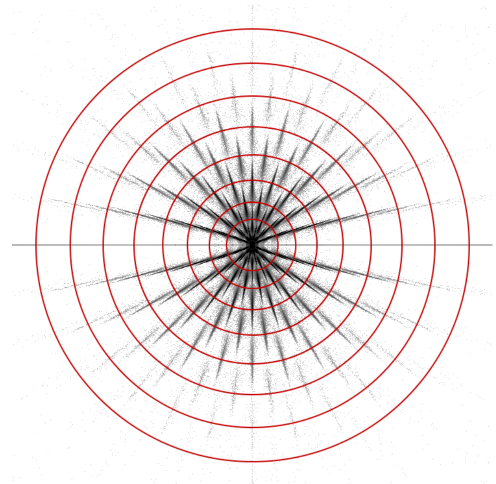
5.6 Characteristic Polynomial Database

Because the populations we consider contain only integers, we are able to compute the characteristic polynomials of matrices in these families exactly over the integers. This proves to be useful for the exploration of many properties of the families and helps answer questions such as how many matrices in a family are singular, or how many distinct

[4] An experienced numerical analyst picked out this numerical artifact at a glance of the eigenvalue picture. Knowing machine epsilon $\varepsilon = 2^{-52}$ he was able to deduce the multiplicity of the zero eigenvalues from the number of rays in the star and the size of $|z^{1/m}|$



(a) Numerical error in the multiple eigenvalues at 0 for the family of 5×5 matrices with population $\{-1, 0, +1\}$. A random sample of 100 million matrices from this family is shown and the image is viewed on the complex range $-0.0018 - 0.0018i$ to $0.0018 + 0.0018i$. The red circles show the expected error when there is eigenvalues of multiplicity 5 (outer red circle, $\varepsilon^{1/5}$) or 4 (inner red circle, $\varepsilon^{1/4}$) at 0.



(b) Numerical error in the multiple eigenvalues at 0 for the family of 20×20 antitridiagonal matrices with population $\{-1, 0, +1\}$. A sample of 100 million matrices from this family is shown and the image is viewed on the complex range $-0.1 - 0.1i$ to $0.1 + 0.1i$. The red circles show the expected error, $\varepsilon^{1/m}$, for eigenvalues of multiplicity m at 0 for m from 15 (outermost circle) to 8 (innermost circle).

Figure 5.9: Numeric error in multiple eigenvalues at 0 for two families of matrices. Red circles have been added to show the expected error in an eigenvalue of multiplicity m at 0 of $\varepsilon^{1/m}$ where ε is machine epsilon.

eigenvalues does a family have. By working with the characteristic polynomials rather than the matrices themselves, we get a compact representation of the family. For many of the families we have explored, the sets of characteristic polynomials are substantially smaller than the sets of matrices (see Table 5.1). While the symmetries discussed in Section 5.3 are helpful in reducing the size of the families for some analysis, it turns out that brute-force computation of all characteristic polynomials in a family can be done much faster. Symmetries can however be helpful in reducing the number of matrices we must compute the characteristic polynomials of by avoiding redundant computation (such as computing the characteristic polynomial of both $\mathbf{A}$ and $-\mathbf{A}$). We do however lose some information about the original matrices by working with their characteristic polynomials such as the geometric multiplicity of their eigenvalues, the eigenvectors, and the eigenvalue condition numbers.

5.6.1 Exhaustive Characteristic Polynomial Computations

Computing the entire set of characteristic polynomials for families of Bohemian matrices can be done when the number of characteristic polynomials is not too large. For example, for the family of 5×5 matrices with population $\{-1, 0, +1\}$, there are 1,839,102 characteristic polynomials. Storing each of these polynomials requires storing 5 64-bit integers (for this family 16-bit integers would be sufficient but in general they are not). Thus, the set of all characteristic polynomials would require about 70MB of memory. If a family has more characteristic polynomials than we are able to fit in memory, managing the set of characteristic polynomials becomes more difficult. Since the number of characteristic polynomials is unknown a priori, approximating the number from lower dimension matrices is useful to ensure all polynomials can fit in memory. Another limitation is the number of matrices in a family. For the 5×5 example, we must compute the characteristic polynomials of 847,288,609,443 matrices which is close to the limit of what can be completed on a personal computer. We were able to compute all characteristic polynomials for this family in less than a week.

When computing sets of characteristic polynomials, the number of occurrences of each polynomial are recorded to provide a complete description of the set of characteristic polynomials. This allows us to compute properties such as counting the number of singular matrices by simply summing the frequencies of all polynomials with constant term equal to zero.

To compute the sets of characteristic polynomials and their frequencies we took advantage of the efficiency of compiled C++ code. Since all matrices are over the integers, we are able to represent the characteristic polynomials of these matrices exactly by using a vector of integers. Dimension n matrices were stored as vectors of length n^2. For structured matrices, only those entries that are sampled from the population are stored in the vector of entries. For example, for upper triangular matrices, we would only use a vector of length $(n^2 + n)/2$. The matrices are sampled using a mapping from the non-negative integers to a vector of entries of each matrix. For example, for the family of $n \times n$ matrices with population $\{-1, 0, +1\}$, the mapping first writes the non-negative integers in base 3, pads the number with zeros until we have a length n^2 array of numbers, and then subtracts 1 from each number. For example, the integer 12345 would be written as 102101112 in base 3. If $n = 4$, we would pad this with enough zeros to give a length 16 array of numbers giving 0000000102101112. Next we would subtract 1 from each number giving the entries of the matrix as a vector: $[-1, -1, -1, -1, -1, -1, -1, 0, -1, 1, 0, -1, 0, 0, 0, 1]$.

Computing the coefficients of the characteristic polynomials was done by generating a program for computing the coefficients as a function of the (free) entries of the matrix using MAPLE. While this is undoubtedly not the most efficient method for finding the characteristic polynomials the flexibility for new families of matrices, and the small dimensions we focus on were sufficient for us to allow the overhead. To store the characteristic polynomials we use the map class in C++ where the keys are vectors of length n containing the coefficients (excluding the leading coefficient) of the characteristic polynomials, and the values are the frequencies of each characteristic polynomial. This has $\mathcal{O}(N)$ look-up time given that the map contains N values. For a Bohemian family that contains M matrices and has N distinct characteristic polynomials, the cost to compute the set of characteristic polynomials is $\mathcal{O}(MN)$.

This technique has proved to be very successful even for families with more than 1 billion characteristic polynomials. To date the largest family we have computed the characteristic polynomials for is the family of 8×8 upper-Hessenberg matrices with population $\{-1, +1\}$. This family contains $2^{9\cdot(8)/2} =$ 68,719,476,736 matrices and 1,279,227,671 characteristic polynomials.

We have made available sets of characteristic polynomials along with the frequencies of the polynomials on the Characteristic Polynomial Database [23] (CPDB) for several families. Currently, the CPDB contains 1,762,728,065 characteristic polynomials from 2,366,960,967,336 matrices and is available on the Bohemian matrices website at `http://www.bohemianmatrices.com/cpdb/`. The code used for computing the sets of characteristic polynomials is available on GitHub at `https://github.com/BohemianMatrices/characteristic-polynomial-database`.

5.6.2 Properties

The distributions of characteristic polynomials have been used to compute several properties of their Bohemian families. This has proven effective in the discovery of several interesting properties and sequences for various families. Tables of the properties we have computed are also available on the CPDB. Here we discuss the techniques we used for computing these properties.

Counting Eigenvalues

Since the set of roots of all characteristic polynomials is the same as the set of all eigenvalues for a Bohemian family, we can count the number of distinct eigenvalues in a Bohemian family from the set of characteristic polynomials. This computation can be done exactly since all characteristic polynomials have integer coefficients. The algorithm we use to count the distinct roots computes a GCD-free basis of the set of characteristic polynomials. That is, we compute a set of polynomials with the same roots as the set of characteristic polynomials such that any two non-equal polynomials in the set have GCD 1. By making all polynomials square-free we have a set of polynomials where the sum of their degrees gives the number of distinct roots.

Algorithm 1: Count the number of distinct roots of a set of polynomials.

Input : The set T of all characteristic polynomials of a Bohemian family.

$T_{\text{irred}} \leftarrow \{\}$.
for $t \in T$ **do**
  Let S be the set of irreducible factors of t over the integers.
  $T_{\text{irred}} \leftarrow T_{\text{irred}} \cup S$

The number of distinct roots in T is $\sum_{t \in T_{\text{irred}}} \deg(t)$.

We tested our algorithm using MAPLE on the set of 1,839,102 characteristic polynomials from the Bohemian family of 5×5 matrices with population $\{-1, 0, +1\}$. On an AMD Ryzen Threadripper 1950X 16 core/32 thread 3.7GHz processor it takes 1211 seconds to count the eigenvalues.

To count the number of distinct real eigenvalues we sum the number of real roots of each polynomial in T_{irred}. The number of real roots of each polynomial is computed using Sturm's theorem (`sturm` function in MAPLE). This took an additional 633 seconds starting from T_{irred}. Counting the real roots using Descartes' rule of signs (`realroot` function in MAPLE) takes 844 seconds.

Minimal Polynomials

The sets of minimal polynomials for Bohemian families were computed by exhaustively computing the minimal polynomial of each matrix in the family using MAPLE. The sets of minimal polynomials for all families have also been provided on the CPDB. The set of minimal polynomials also tells us the number of non-derogatory[5] matrices since a matrix of dimension n with minimal polynomial of degree n must be non-derogatory.

[5]A matrix is non-derogatory if its characteristic polynomial and minimal polynomial are equal up to a factor of ± 1.

Jordan Canonical Forms

The Jordan canonical form (JCF) provides information on several properties of a matrix including its characteristic polynomial, minimal polynomial, and both the algebraic and geometric multiplicities of the eigenvalues. Computing the JCF numerically is unstable because it is discontinuous with respect to changes in its entries. Symbolically computing the JCF can be done although requires an extension to the set of algebraic numbers (Bohemian matrices are over the field of integers). This can lead to a JCF where the eigenvalues can only be represented exactly as the solutions to algebraic equations, specifically when the dimension of the matrix is larger than 4. To avoid a field extension, the rational Jordan form [16, 17] can be used. The Frobenius (rational) form is another alternative that is a unique canonical form over the base field (integers). Since a matrix in Frobenius form has a unique JCF [17], computing the Frobenius form is sufficient for counting the number of JCFs in a family of Bohemian matrices. Since the set of JCFs is isomorphic to similarity normal subset of a Bohemian family (see Proposition 5.3.21), and each matrix in Frobenius form has a unique JCF, the set of Frobenius forms must also be isomorphic to a similarity normal subset of a Bohemian family.

To count the number of distinct JCFs (or equivalently the number of distinct Frobenius forms, or the size of a similarity normal subset), we compute the Frobenius form of every matrix in a family symbolically and count the number of distinct Frobenius forms. In Table 5.4, we report the number of distinct JCFs for an example family.

Permutation Normal Subset

Computing a permutation normal subset requires evaluating all $n!$ permutations of all matrices in the family. Algorithm 2 counts the size of the permutation normal subset without storing the subset. This algorithm can be modified to return a permutation normal subset by tracking the integer hashes of the matrices in the subset and mapping these back to matrices from the family.

Rhapsodic Matrices

To count the number of strictly rhapsodic matrices in a family we exhaustively compute the inverse of every invertible matrix in the family and count the number of inverses that belong to the same family. We use Maple for this computation. To count the number of non-strictly rhapsodic matrices, we first find the sets of Frobenius forms for all matrices in the family (two matrices are similar if and only if they have the same Frobenius form). We

Algorithm 2: Compute the size of a permutation normal subset.

```
Let decode be an injective function that maps integers to unique matrices in the
  family with domain [1, m] where m is the number of matrices in the family.
Let encode be the inverse of decode.

count ← 0
for i from 1 to m do
    permutation_normal ← True
    A ← decode(i)
    j ← 1
    while permutation_normal is True and j ≤ n! do
        B ← jth permutation of A
        permutation_normal ← encode(B) ≥ i
        j ← j + 1
    if permutation_normal is True then
        count ← count+1

count gives the size of a permutation normal subset of the family.
```

then compute the inverse of every invertible matrix in the family, compute its Frobenius form, and check if its Frobenius form is in the set of Frobenius forms of the family.

Properties of Determinants

Given the set of characteristic polynomials for a Bohemian family we also get the set of determinants by isolating the constant coefficients. This allows us to easily compute several properties of the matrices that are directly related to determinants. The properties we have computed include:

- the maximum absolute determinant,
- the number of unimodular matrices (determinant ± 1),
- the number of singular matrices (determinant 0),
- the number of distinct determinants, and
- the smallest positive integer that is not a determinant.

5.6.3 Integer Sequences

The properties we compute are integer sequences over the dimension of the Bohemian family. For example, how many $n \times n$ matrices with population $\{-1, 0, +1\}$ are singular.

We have found several of these sequences already exist on the OEIS [22]. For the previous example we refer to OEIS sequence A057981. Some of the sequences we compute appear to align with sequences already present on the OEIS but in many cases we lack proof of these matches. To help find proofs to these, we have compiled a list of conjectures linking sequences related to Bohemian families to existing sequences on the OEIS. These conjectures are listed below. Conjectures 1, 3, 4, 5, 7 and 10 already have proofs and Conjecture 2 has been disproved. These conjectures have been included for completeness.

Conjectures

1. The number of nilpotent $n \times n$ matrices with entries from the set $\{0, +1\}$ is given by the sequence A003024. Proof in [14], reference provided by Jianxiang Chen.

2. The maximal characteristic height of $n \times n$ matrices with entries from the set $\{0, +1\}$ is given by the sequence A082914. Disproved by Jianxiang Chen.

3. The number of nilpotent $n \times n$ matrices with entries from the set $\{0, +1\}$ and diagonal entries fixed at 0 is given by the sequence A003024. Proof in [14], reference provided by Jianxiang Chen.

4. The maximal absolute determinant of $n \times n$ matrices with entries from the set $\{-1, 0, +1\}$ is given by the sequence A003433. Proof in [14], reference provided by Jianxiang Chen.

5. The number of nilpotent $n \times n$ matrices with entries from the set $\{-1, 0, +1\}$ and diagonal entries fixed at 0 is given by the sequence A085506. Proof in [19], reference provided by Jianxiang Chen.

6. The number of nilpotent $n \times n$ matrices with entries from the set $\{0, +1, +2\}$ is given by the sequence A188457.

7. The maximum absolute determinant of an $n \times n$ upper-Hessenberg matrix with entries from the set $\{0, +1\}$ and subdiagonal entries fixed at 1 is given by the Fibonacci sequence A000045. Proof in [10], reference provided by Nick Higham.

8. The maximum absolute determinant of an $n \times n$ upper-Hessenberg matrix with entries from the set $\{0, +1, +2\}$ and subdiagonal entries fixed at 1 is given by sequence A052542.

9. The number of distinct determinants of an $n \times n$ upper-Hessenberg matrix with entries from the set $\{0, 1\}$, subdiagonal entries fixed at 1, and diagonal entries fixed at 0 is given by sequence A212264.

10. All upper-Hessenberg matrices with subdiagonal entries fixed at 1 are non-derogatory. Proof in [8], Proposition 5.3.

11. The maximum characteristic height of an $n \times n$ upper-Hessenberg matrix with entries from the set $\{0, +1, +2\}$, subdiagonal entries fixed at 1, and diagonal entries fixed at 0 is given by sequence A058764.

12. The number of distinct determinants of an $n \times n$ upper-Hessenberg matrix with entries from the set $\{-1, 0\}$, subdiagonal entries fixed at 1, and diagonal entries fixed at 0 is given by sequence A001611.

13. The maximum absolute determinant of an $n \times n$ upper-Hessenberg matrix with entries from the set $\{-1, 0\}$, subdiagonal entries fixed at 1, and diagonal entries fixed at 0 is given by the Fibonacci sequence A000045.

14. The number of distinct determinants of an $n \times n$ upper-Hessenberg matrix with entries from the set $\{-1, 0, +1\}$, subdiagonal entries fixed at 1, and diagonal entries fixed at 0 is given by sequence A001588.

15. The maximum absolute determinant of an $n \times n$ upper-Hessenberg matrix with entries from the set $\{-1, 0, +1\}$, subdiagonal entries fixed at 1, and diagonal entries fixed at 0 is given by the Fibonacci sequence A000045.

16. The number of distinct determinants of an $n \times n$ upper-Hessenberg matrix with entries from the set $\{-1, +1\}$, subdiagonal entries fixed at 1, and diagonal entries fixed at 0 is given by sequence A001611.

17. The maximum absolute determinant of an $n \times n$ upper-Hessenberg matrix with entries from the set $\{-1, +1\}$, subdiagonal entries fixed at 1, and diagonal entries fixed at 0 is given by the Fibonacci sequence A000045.

18. The number of distinct determinants of an $n \times n$ upper-Hessenberg matrix with entries from the set $\{-1, 0\}$ and subdiagonal entries fixed at 1 is given by sequence A000051.

19. The number of distinct determinants of an $n \times n$ upper-Hessenberg matrix with entries from the set $\{-1, 0, +1\}$ and subdiagonal entries fixed at 1 is given by sequence A000051.

20. The number of distinct determinants of an $n \times n$ upper-Hessenberg matrix with entries from the set $\{-1, +1\}$ and subdiagonal entries fixed at 1 is given by sequence A000051.

21. The size of a permutation normal subset of the family of $n \times n$ matrices with population $\{-1, 0, +1\}$ is given by sequence A004105.

5.7 Conclusion

Studying Bohemian matrices has become a fascinating journey of unanswered questions first inspired by the strange discrete structures appearing in plots of Bohemian eigenvalues. Here we provided a general overview of Bohemian matrices and the questions we are interested in. The symmetries discussed in Section 5.3 have proven useful when working with Bohemian matrices. The `BHIME-project` Matlab framework for plotting Bohemian eigenvalues, which has been used to generate thousands of unique and interesting images, has been discussed. In Section 5.5, two families of Bohemian matrices where the eigenvalue routine in Matlab fails to provide a solution were discussed. Finally, we introduced the Characteristic Polynomial Database and the tools we use to compute properties of Bohemian families.

Many questions relating to Bohemian families remain unanswered. Our future work is focused on a few main problems. First, we are developing a Python package for generating plots of Bohemian eigenvalues and hope this package will make Bohemian eigenvalues accessible to a wider audience. Next, the exploration of the distributions of eigenvalue condition numbers are of interest. Are the eigenvalues within some families or for certain structures inherently better conditioned than in other families? Inverse eigenvalue problems are also of interest. That is, given an eigenvalue and a Bohemian family, identify a matrix in the family with the given eigenvalue. Finally, we hope that the list of conjectures available through the Characteristic Polynomial Database inspires others to explore these types of problems.

Bibliography

[1] J. Baez. The beauty of roots. *Available at: https://johncarlosbaez.wordpress.com/2011/12/11/the-beauty-of-roots/*, 2011.

[2] P. Borwein and L. Jörgenson. Visible structures in number theory. *The American Mathematical Monthly*, 108(10):897–910, 2001.

[3] P. Borwein and C. Pinner. Polynomials with $\{0, +1, -1\}$ coefficients and a root close to a given point. *Canadian Journal of Mathematics*, 49(5):887–915, 1997.

[4] E. Y. S. Chan. A comparison of solution methods for Mandelbrot-like polynomials. *Electronic Thesis and Dissertation Repository*, 2016. `https://ir.lib.uwo.ca/etd/4028`.

[5] E. Y. S. Chan and R. M. Corless. A new kind of companion matrix. *Electronic Journal of Linear Algebra*, 32:335–342, 2017.

[6] E. Y. S. Chan and R. M. Corless. Minimal height companion matrices for Euclid polynomials. *Mathematics in Computer Science*, Jul 2018.

[7] E. Y. S. Chan, R. M. Corless, L. Gonzalez-Vega, J. R. Sendra, and J. Sendra. Algebraic linearizations of matrix polynomials. *Linear Algebra and its Applications*, 563:373–399, 2019.

[8] E. Y. S. Chan, R. M. Corless, L. Gonzalez-Vega, J. R. Sendra, J. Sendra, and S. E. Thornton. Bohemian upper Hessenberg matrices. *arXiv preprint arXiv:1809.10653*, 2018.

[9] E. Y. S. Chan, R. M. Corless, L. Gonzalez-Vega, J. R. Sendra, J. Sendra, and S. E. Thornton. Bohemian upper Hessenberg Toeplitz matrices. *arXiv preprint arXiv:1809.10664*, 2018.

[10] L. Ching. The maximum determinant of an $n \times n$ lower Hessenberg $(0, 1)$ matrix. *Linear algebra and its applications*, 183:147–153, 1993.

[11] D. Christensen. Plots of roots of polynomials with integer coefficients. `http://jdc.math.uwo.ca/roots/`. Accessed: 2016-06-25.

[12] R. M. Corless and S. E. Thornton. The Bohemian eigenvalue project. *ACM Communications in Computer Algebra*, 50(4):158–160, 2016.

[13] R. M. Corless and S. E. Thornton. Visualizing eigenvalues of random matrices. *ACM Communications in Computer Algebra*, 50(1):35–39, apr 2016.

[14] D. Cvetkovic, M. Doob, and H. Sachs. Spectra of graphs-theory and applications 3rd edn. 1995.

[15] J. Dongarra, S. Hammarling, N. J. Higham, S. D. Relton, P. Valero-Lara, and M. Zounon. The design and performance of batched BLAS on modern high- performance computing systems. *Procedia Computer Science*, 108:495–504, 2017.

[16] M. Giesbrecht. Nearly optimal algorithms for canonical matrix forms. *SIAM Journal on Computing*, 24(5):948–969, 1995.

[17] E. Kaltofen, M. Krishnamoorthy, and B. D. Saunders. Fast parallel algorithms for similarity of matrices. In *Proceedings of the fifth ACM symposium on Symbolic and Algebraic Computation*, pages 65–70. ACM, 1986.

[18] J. E. Littlewood. On polynomials $\sum^n \pm z^m$, $\sum^n e^{\alpha_m i} z^m$, $z = e^{\theta_i}$. *Journal of the London Mathematical Society*, 41:367–376, 1966.

[19] B. D. McKay, F. E. Oggier, G. F. Royle, N. J. A. Sloane, I. M. Wanless, and H. S. Wilf. Acyclic digraphs and eigenvalues of (0, 1)-matrices. *Journal of Integer Sequences*, 7(2):3, 2004.

[20] A. Odlyzko and B. Poonen. Zeros of polynomials with 0, 1 coefficients. *Enseign. Math*, 39:317–348, 1993.

[21] R. Reyna and S. Damelin. On the structure of the Littlewood polynomials and their zero sets. *arXiv preprint arXiv:1504.08058*, 2015.

[22] N. J. A. Sloane. The on-line encyclopedia of integer sequences. Published electronically at `https://oeis.org` (Jan. 11, 2019).

[23] S. E. Thornton. The characteristic polynomial database. Available at `http://bohemianmatrices.com/cpdb` (Sept. 7, 2018).

[24] L. N. Trefethen. Pseudospectra of matrices. *Numerical analysis*, 91:234–266, 1991.

Chapter 6

Bohemian Upper Hessenberg and Toeplitz Matrices

6.1 Introduction

A matrix family is called **Bohemian** if its entries come from a fixed finite discrete (and hence bounded) set, usually integers. The name is a mnemonic for **Bo**unded **He**ight **M**atrix of **I**ntegers. Such populations arise in many applications (e.g. compressed sensing) and the properties of matrices selected "at random" from such families are of practical and mathematical interest. For example, Tao and Vu have shown that random matrices (more specifically real symmetric random matrices in which the upper-triangular entries $\xi_{i,j}$, $i < j$ and diagonal entries $\xi_{i,i}$ are independent) have simple spectrum [24]. An overview of some of our original interest in Bohemian matrices can be found in [16].

Bohemian families have been studied for a long time, although not under that name. For instance, Olga Taussky-Todd's paper "Matrices of Rational Integers" [25] begins by saying

> "This subject is very vast and very old. It includes all of the arithmetic theory of quadratic forms, as well as many of other classical subjects, such as Latin squares and matrices with elements $+1$ or -1 which enter into Euler's, Sylvester's or Hadamard's famous conjectures."

The paper [20] by C. W. Gear is another instance. What is new here is the idea that these families are themselves interesting objects of study, and susceptible to brute-force computational experiments as well as to asymptotic analysis. These experiments have generated many conjectures, some of which we resolve in this paper. Others remain unsolved, and are listed on the Characteristic Polynomial Database [26]. Many of the conjectures have a number-theoretic or combinatorial flavour.

Typical computational puzzles arise on asking simple-looking questions such as "how many 6×6 matrices with the population[1] $\{-1, 0, +1\}$ are singular." The answer is not known as we write this, although we can give a probabilistic estimate (0.205 after 20,000,000 sample determinants[2]): brute computation seems futile because there are $3^{36} \doteq 1.7 \times 10^{17}$ such matrices. We do know the answers up to size five by five: The number of n by n singular matrices with population $\{-1, 0, +1\}$ is, for $n = 1, 2, 3, 4$, and 5, just 1, 33, 7,875, 15,099,201, and 237,634,987,683. This represents fractions of their numbers (3^{n^2}) of 0.333, 0.407, 0.400, 0.351, and 0.280, respectively.

Yet such matrix families are both useful and interesting. For instance, one may use discrete optimization over a family to look for improved growth factor bounds [21]. Matrices with the population $\{-1, 0, +1\}$ have minimal height[3] over all integer matrices; finding a matrix in this family which has a given polynomial $p(\lambda) \in \mathbb{Z}[\lambda]$ as characteristic polynomial identifies a so-called "minimal height companion matrix", which may confer numerical benefits.

Recently the study of eigenvalues of structured Bohemian matrices (e.g. tridiagonal, complex symmetric) has been undertaken and several puzzling features are seen resulting from extensive experimental computations. For instance, some of the images at `http://www.bohemianmatrices.com/gallery` show common features including "holes".

Different matrix structures produce remarkably different pictures. One structure useful in eigenvalue computation is the upper Hessenberg matrix, which means a matrix $\mathbf{H}$ such that $h_{i,j} = 0$ if $i > j + 1$. These arise naturally in eigenvalue computation because the QR iteration is cheaper for matrices in Hessenberg form. Results on the determinants of Hessenberg matrices can be found in [22].

Remark 6.1.1. *on computing eigenvalues by first computing characteristic polynomials.* Numerical analysts are familiar with the superior numerical stability of computing eigenvalues iteratively, usually by the QR algorithm or some variant, rather than first computing characteristic polynomials and then finding roots. As is well-known, such an algorithm is *numerically unstable* because polynomials are usually badly-conditioned while eigenvalues are usually well-conditioned[4]. Somewhat surprisingly, for several families

[1] The population of a Bohemian family is the set of permissible entries.

[2] 4103732 singular matrices out of twenty million sampled.

[3] $\text{height}(A) := ||\text{vec}(A)||_\infty$ is the largest absolute value of any entry in A.

[4] This has been well-known to the point of folklore since the work of Wilkinson. The well-conditioning of eigenvalues has only recently been quantified in some cases, but for instance the results of [4] do confirm the folklore.

of Bohemian matrices, characteristic polynomials become valuable again: first because the matrix dimensions are typically small or at most moderate, the ill-conditioning does not matter much, and second because for some families (not all!) the number of distinct characteristic polynomials is vastly smaller than the number of matrices in the family. For instance, for the general five by five matrices with population $\{-1, 0, +1\}$, there are nearly one trillion such matrices, but fewer than two million characteristic polynomials. This compression is significant.

For other families of matrices, such as upper Hessenberg Toeplitz matrices, there is no compression at all because each matrix has a distinct characteristic polynomial. Circulant matrices fall between, having fewer characteristic polynomials but not vastly fewer. The lesson is that for some questions (though not others), prior computation of characteristic polynomials is valuable.

We begin our study in this paper by considering determinants of Bohemian upper Hessenberg matrices. We prove two recursive formulae for the characteristic polynomials of upper Hessenberg matrices[5]. For another recursive formula we refer to [18]. During the course of our computations, we encountered "maximal polynomial height" characteristic polynomials when the matrices were not only upper Hessenberg, but Toeplitz ($h_{i,j}$ constant along diagonals $j - i = k$). Further restrictions to this class allowed identification of key results including explicit formulae for the characteristic polynomials of maximal height. In what follows, we lay out definitions and prove several facts of interest about characteristic polynomials and their respective height for these families.

In Figure 6.1 we see all eigenvalues of 6×6 upper Hessenberg matrices with sub-diagonals fixed at 1 and upper triangular entries from the population $P = \{-1, 0, +1\}$. We denote this set of matrices $\mathcal{H}^{6\times 6}_{\{0\}}(P)$. There are $3^{21} = 10{,}460{,}353{,}203$ such matrices. We see a wide octagonal shape. The width of the figure reflects that some matrices might have diagonals -1, while some have diagonals 0, and others have diagonals 1. Of course mixed diagonals are also possible, but this should only tend to push things towards the centre.

In Figure 6.2, we see all the eigenvalues of all 14×14 upper Hessenberg Toeplitz matrices with sub-diagonals fixed at 1 and upper triangular entries from the population $P = \{-1, 0, +1\}$. We denote this set of matrices $\mathcal{M}^{14\times 14}_{\{0\}}(P)$. There are $3^{14} = 4{,}782{,}969$ such matrices. We now see a wide irregular hexagonal shape. More, the density of eigenvalues (here, a darker colour indicates higher density of eigenvalues) is quite irregular, with high-density flecks dispersed throughout. In some ways the picture is reminiscent

[5]We do not claim originality; recursion relations for upper Hessenberg determinants are known.

of seeds in a cotton ball, if the cotton ball has been flattened. The conjugate symmetry and $z \to -z$ symmetry are evident; to save space, we could have plotted only the first quadrant, but for completeness have included all four. This helps to show that there is a slightly lower density of eigenvalues near (not on) the real line. The density of eigenvalues actually *on* the real line is quite high, although this is not evident from the picture.

The one thing that is easily explained about Figure 6.2 is the wide flat top (and bottom). To do this, consider eigenvalues of upper Hessenberg Toeplitz matrices with *zero diagonal*. Figure 6.4 is a picture of the set of eigenvalues of all 14×14 upper Hessenberg Toeplitz matrices, sub-diagonal 1, diagonal 0, and upper triangular entries from the population $P = \{-1, 0, +1\}$. There are $3^{13} = 1{,}594{,}323$ such matrices. Here, we also see a hexagonal shape, but this time, it is not as wide. The matrices B giving rise to Figure 6.2 are exactly the matrices $B = A$, $B = A + I$ and $B = A - I$ where the matrices A give rise to Figure 6.4; thus the eigenvalues of each A occur three times, once with zero shift, once with -1 shift, and once with 1 shift. That is, Figure 6.2 is simply three copies of Figure 6.4 placed side by side, giving the appearance of a flat (or mostly flat) top and bottom.

In Figure 6.3 we show the set of eigenvalues of upper Hessenberg matrices, sub-diagonal 1, diagonal 0, and upper triangular entries from the population $P = \{-1, 0, +1\}$. We denote this set by $\mathcal{Z}^{6\times 6}_{\{0\}}(P)$. There are substantially fewer matrices here than in the $\mathcal{H}^{6\times 6}_{\{0\}}(P)$ family, only $3^{15} = 14{,}348{,}907$ to be exact. Roughly speaking, Figure 6.1 is partially explained by saying that, along with other eigenvalues, it contains three copies of Figure 6.3 placed with centres at -1, at 0, and at $+1$.

In Figure 6.4 we see more clearly that the high-density “flecks” occur moderately near to the edge of the eigenvalue inclusion region. We have no explanation for this. We also see that the eigenvalues fit into a rough diamond shape; one wonders if the eigenvalues $\lambda = x + iy$ fit into a region of shape $|x| + |y| \le O(\sqrt{n})$. Again, we have no explanation for this (or even much data; we do not know if this guess is even correct experimentally).

In this paper we seek to explain some of the features of these pictures, and to learn some things about these families of Bohemian matrices. We provide supplementary material through a git repository available at `https://github.com/BohemianMatrices/Bohemian_Upper_Hessenberg_Toeplitz_Matrices`. This repository provides all code and data used to generate the results, figures, and tables in this paper.

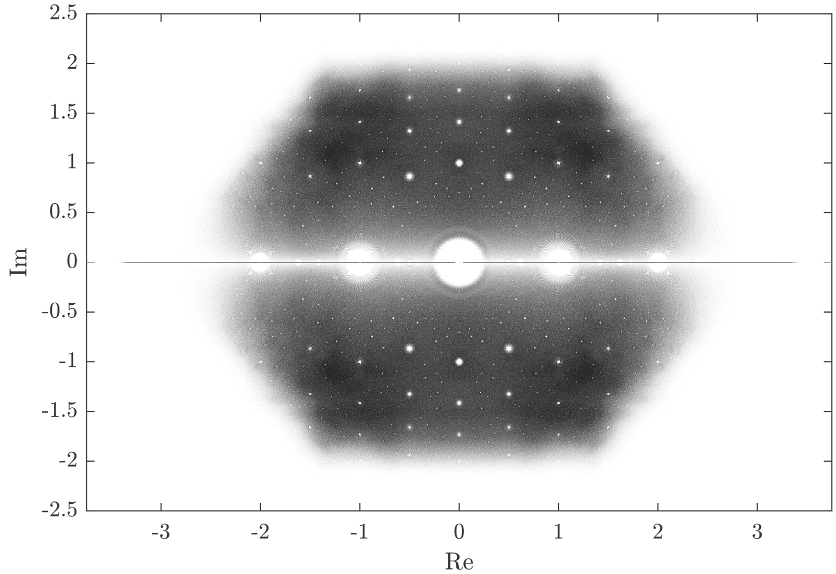


Figure 6.1: The set of eigenvalues of all 10,460,353,203 six by six upper Hessenberg matrices $\mathbf{H}$ with entries $\mathbf{H}_{i,j} \in \{-1, 0, +1\}$ for $1 \le i \le j \le 6$, and $\mathbf{H}_{i+1,i} = 1$ for $1 \le i < 6$. A more detailed image can be found at `assets.bohemianmatrices.com/gallery/UH_6x6.png`

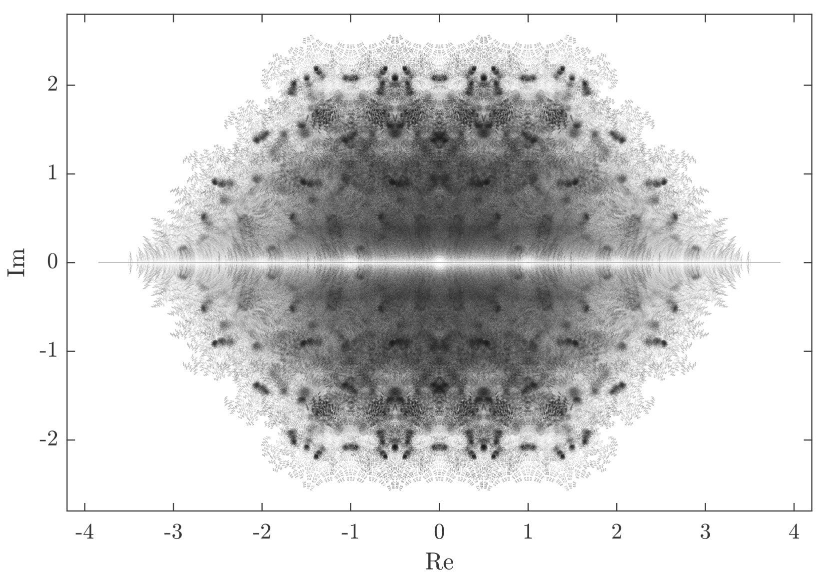


Figure 6.2: The set of eigenvalues of all 14×14 upper Hessenberg Toeplitz matrices with sub-diagonal entries equal to 1, and all other entries from the set $\{-1, 0, +1\}$. A more detailed image can be found at `assets.bohemianmatrices.com/gallery/UHT_14x14.png`

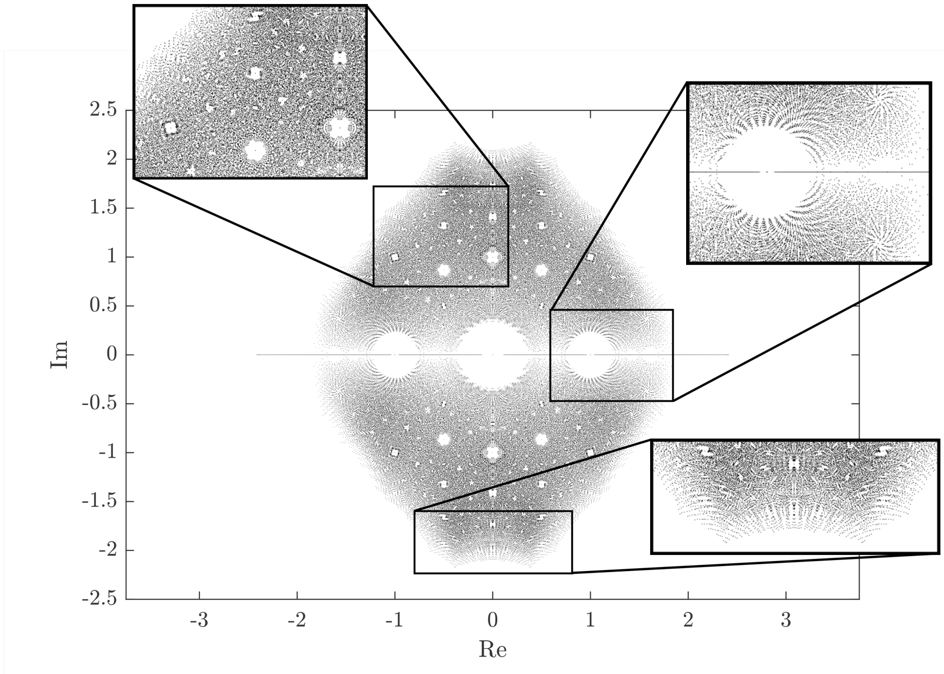


Figure 6.3: The set of eigenvalues of all 14,348,907 matrices in $\mathcal{Z}^{6\times 6}_{\{0\}}(\{-1, 0, +1\})$; that is, six by six upper Hessenberg matrices $\mathbf{H}$ with entries $\mathbf{H}_{i,j} \in \{-1, 0, +1\}$ for $1 \le i < j \le 6$, diagonal entries fixed as zero, and $\mathbf{H}_{i+1,i} = 1$ for $1 \le i < 6$. A more detailed image can be found at `assets.bohemianmatrices.com/gallery/UH_0_Diag_6x6.png`

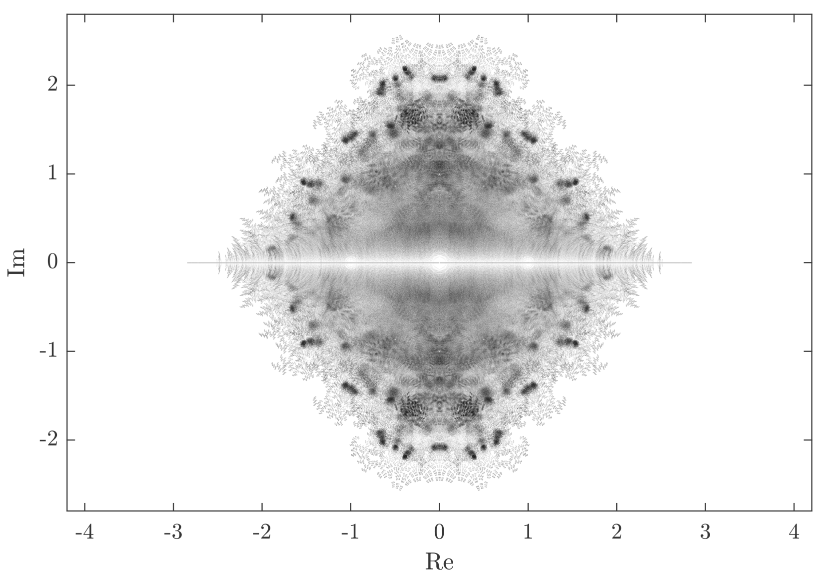


Figure 6.4: The set of eigenvalues of all 14×14 upper Hessenberg Toeplitz matrices sub-diagonal entries equal to 1, diagonal entries equal to 0, and all other entries from the set $\{-1, 0, +1\}$. A more detailed image can be found at `assets.bohemianmatrices.com/gallery/UHT_0_Diag_14x14.png`

6.2 Prior Work

Visible features of graphs of roots and eigenvalues from structured families of polynomials and matrices have been previously studied. One well-known polynomial whose roots produce interesting pictures is the Littlewood polynomial,

$$p(x) = \sum_{i=0}^{n} a_i x^i \,, \tag{6.1}$$

where $a_i \in \{-1, +1\}$. These polynomials have been studied in [2], [6], [7], and [8]. The image of their roots raises many questions, ranging from whether the set is (ultimately, as $n \to \infty$) a fractal and what the boundary of the set is, to questions about the holes in the image and its connection to various properties, such as degree and coefficients of the polynomial. Answers to some of these questions, particularly the ones involving the holes, have been shown to have some significance in number theory [3]. Roots of other polynomials have also been visualized; for more, see Christensen's[6] and Jörgenson's[7] web pages.

Corless used a generalization of the Littlewood polynomial (to Lagrange bases). In his paper [13], he gave a new kind of companion matrix for polynomials expressed in a Lagrange basis. He used generalized Littlewood polynomials as test problems for his algorithm.

"The Bohemian Eigenvalue Project" was first presented as a poster [17] at the East Coast Computer Algebra Day (ECCAD) 2015. The poster focused on preliminary results and many of the questions raised when visualizing the distributions of Bohemian eigenvalues over the complex plane. In particular, the poster focused on "eigenvalue exclusion zones" (i.e. distinct regions within the domain of the eigenvalues where no eigenvalues exist), computational methods for visualizing eigenvalues, and some results on eigenvalue conditioning over distributions of random matrices.

In Chan's Master's thesis [9], she extended Piers W. Lawrence's construction of the companion matrix for the Mandelbrot polynomials [15, 14] to other families of polynomials, mainly the Fibonacci-Mandelbrot polynomials and the Narayana-Mandelbrot polynomials. What is relevant here about this construction is that these matrices are upper Hessenberg and contain entries from a constrained set of numbers: $\{-1, 0\}$, and therefore fall under the category of being Bohemian upper Hessenberg. Both the Fibonacci-Mandelbrot matrices and Narayana-Mandelbrot matrices are also Bohemian upper Hessenberg, but the set that

[6] `https://jdc.math.uwo.ca/roots/`
[7] `http://www.cecm.sfu.ca/~loki/Projects/Roots/`

the entries draw from is $\{-1, 0, +1\}$. At the time of submission for Chan's Master's thesis, the largest number of eigenvalues successfully computed (using a machine with 32 GB of memory) were 32,767, 17,710, and 18,559 for the Mandelbrot, Fibonacci-Mandelbrot, and Narayana-Mandelbrot matrices, respectively. This makes the 16^{th} Mandelbrot matrix the "largest" Bohemian matrix that we have solved at the time we write this paper.

These new constructions led Chan and Corless to a new kind of companion matrix for polynomials of the form $c(z) = za(z)b(z) + c_0$. A first step towards this was first proved using the Schur complement in [10]. Knuth then suggested that Chan and Corless look at the Euclid polynomials [11], based on the Euclid numbers. It was the success of this construction that led to the realization that this construction is general, and gives a genuinely new kind of companion matrix. Similar to the previous three families of matrices, the Euclid matrices are also upper Hessenberg and Bohemian, as the entries are comprised from the set $\{-1, 0, +1\}$. In addition, an interesting property of these companion matrices is that their inverses are also Bohemian with the same population, a property which we call "the matrix family having *rhapsody* [12]."

As an extension of this generalization, Chan et al. [12] showed how to construct linearizations of matrix polynomials, particularly of the form $z\mathbf{a}(z)\mathbf{d}_0+\mathbf{c}_0$, $\mathbf{a}(z)\mathbf{b}(z)$, $\mathbf{a}(z)+\mathbf{b}(z)$ (when $\deg(\mathbf{b}(z)) < \deg(\mathbf{a}(z))$, and $z\mathbf{a}(z)\mathbf{d}_0\mathbf{b}(z) + \mathbf{c}_0$, using a similar construction.

6.3 Notation

In what follows, we present some results on upper Hessenberg Bohemian matrices of the form

$$\mathbf{H}_n = \begin{bmatrix} h_{1,1} & h_{1,2} & h_{1,3} & \cdots & h_{1,n} \\ s & h_{2,2} & h_{2,3} & \cdots & h_{2,n} \\ 0 & s & h_{3,3} & \cdots & h_{3,n} \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ 0 & \cdots & 0 & s & h_{n,n} \end{bmatrix} \tag{6.2}$$

with $s = \exp(i\theta_k)$, usually $s \in \{-1, +1\}$ (we do not allow zero sub-diagonal entries, because that reduces the problem to smaller ones) and $h_{i,j} \in \{-1, 0, +1\}$ for $1 \le i \le j \le n$. We denote the characteristic polynomial $Q_n(z) \equiv \det(z\mathbf{I} - \mathbf{H}_n)$.

Definition 6.3.1. The set of all $n \times n$ Bohemian upper Hessenberg matrices with upper triangle population P and sub-diagonal population from a discrete set of roots of unity, say $s \in \{e^{i\theta_k}\}$ where $\{\theta_k\}$ is some finite set of angles, is called $\mathcal{H}^{n\times n}_{\{\theta_k\}}(P)$. In particular,

$\mathcal{H}^{n\times n}_{\{0\}}(P)$ is the set of all $n \times n$ Bohemian upper Hessenberg matrices with upper triangle entries from P and sub-diagonal entries equal to 1 and $\mathcal{H}^{n\times n}_{\{\pi\}}(P)$ is when the sub-diagonals entries are -1.

It will often be true that the average value of a population will be zero. In that case, matrices with trace zero will be common. It is a useful oversimplification to look in that case at matrices whose diagonal is exactly zero.

Definition 6.3.2. For a population P such that $0 \in P$, let $\mathcal{Z}^{n\times n}_{\{\theta_k\}}(P)$ be the subset of $\mathcal{H}^{n\times n}_{\{\theta_k\}}(P)$ where the main diagonal entries are fixed at 0.

6.4 Results of Experiments

The methods used for computing the characteristic polynomials and counting the number of eigenvalues presented in Tables 6.1–6.10 will be discussed in detail in a forthcoming paper. Many of the smaller-dimension computations were done directly in MAPLE 2017; for instance, computation of the characteristic polynomials of all two million or so matrices in $\mathcal{H}^{5\times 5}_{\{0\}}(\{0, +1\})$ took about six hours on a Surface Pro. The greater number of higher-dimension matrices, or matrices with larger populations, required special techniques and larger & faster machines. Eigenvalue computations were also done in MATLAB and in Python. The computed characteristic polynomials are available through the Characteristic Polynomial Database [26].

n	#matrices	#cpolys	#neutral polys	#neutral matrices
2	27	16	2	4
3	729	166	3	24
4	59,049	3,317	7	332
5	14,348,907	133,255	11	9,909
6	10,460,353,203	10,872,459	25	696,083

Table 6.1: Some properties of matrices in $\mathcal{H}^{n\times n}_{\{0\}}(\{-1, 0, +1\})$. The #matrices column reports the number of distinct matrices at each dimension. The #cpolys column reports the number of distinct characteristic polynomials at each dimension. The #neutral polys reports the number of characteristic polynomials where all roots have zero real part. The #neutral matrices column reports the number of matrices where all eigenvalues have zero real part.

Other questions than those answered in these tables can be asked of this data. For instance, one might be interested in the proportion of singular matrices. By asking which

n	#matrices	#cpolys	#neutral polys	#neutral matrices
2	3	3	2	2
3	27	15	3	6
4	729	140	7	66
5	59,049	2,297	11	1,069
6	14,348,907	67,628	25	45,375
7	10,460,353,203	3,606,225	45	4,105,977

Table 6.2: Some properties of matrices in $\mathcal{Z}_{\{0\}}^{n\times n}(\{-1,0,+1\})$. The #matrices column reports the number of distinct matrices at each dimension. The #cpolys column reports the number of distinct characteristic polynomials at each dimension. The #neutral polys reports the number of characteristic polynomials where all roots have zero real part. The #neutral matrices column reports the number of matrices where all eigenvalues have zero real part.

n	multiplicity 1	2	3	4	5	6
2	5	1				
3	35	0	1			
4	431	5	0	1		
5	9,497	9	3	0	1	
6	363,143	51	5	1	0	1

Table 6.3: Number of distinct eigenvalues of various multiplicities of matrices in $\mathcal{Z}_{\{0\}}^{n\times n}(\{-1,0,+1\})$. Most eigenvalues are simple. It turns out that every multiple eigenvalue also occurs as a simple eigenvalue for some other matrix. The only n-multiple eigenvalue of the class of n by n matrices is, of course, $\lambda = 0$.

characteristic polynomials have zero constant coefficient, and counting the number of matrices that have that characteristic polynomial, one can answer such questions. In the case of six by six upper Hessenberg matrices with population $\{-1,+1\}$, there are 383,680 singular matrices, or about 18.3%. Recall that for "random" six by six matrices, where the entries are chosen perhaps uniformly over some real interval, the probability of singularity is *zero* because such matrices come from a set of measure zero. Yet in applications, the probability of singular matrices is often *nonzero* because of structure. By looking at Bohemian matrices, we get some idea of the influence of structure for finite dimensions n.

n	#matrices	#cpolys	#distinct real λ	#neutrals polys	#neutral matrices
2	8	6	6	1	1
3	64	28	25	1	1
4	1,024	197	219	1	1
5	32,768	2,235	3,264	1	1
6	2,097,152	39,768	75,045	1	1
7	268,435,456	1,140,848	2,694,199	1	1

Table 6.4: Some properties of matrices in $\mathcal{H}^{n\times n}_{\{0\}}(\{0,+1\})$. The #matrices column reports the number of distinct matrices at each dimension. The #cpolys column reports the number of distinct characteristic polynomials at each dimension. The #distinct real λ column reports the number of distinct real eigenvalues in $\mathcal{H}^{n\times n}_{\{0\}}(\{0,+1\})$. The #neutral polys reports the number of characteristic polynomials where all roots have zero real part (here only z^n). We conjecture that this is always so (and that there is only one matrix for that neutral polynomial). The #neutral matrices column reports the number of matrices where all eigenvalues have zero real part.

n	multiplicity 1	2	3	4	5	6
2	6	2				
3	43	2	2			
4	413	6	2	2		
5	6,920	6	3	2	2	
6	166,005	45	6	2	2	2

Table 6.5: Number of distinct eigenvalues of various multiplicities matrices in $\mathcal{H}^{n\times n}_{\{0\}}(\{0,+1\})$. Note that in this class of matrices, diagonal entries of the matrix need not be zero.

6.5 Upper Hessenberg Matrices

We can make sense of some of those experiments by theoretical results and proofs. We begin with a recurrence relation for the characteristic polynomial $Q_n(z) = \det(z\mathbf{I} - \mathbf{H}_n)$ for $\mathbf{H}_n \in \mathcal{H}^{n\times n}_{\{\theta_k\}}(P)$ where $s = \exp(i\theta_k)$. Later we will specialize the population P to contain only zero and numbers of unit magnitude, usually $\{-1,0,+1\}$.

Theorem 6.5.1.

$$Q_n(z) = zQ_{n-1}(z) - \sum_{k=1}^{n} s^{k-1} h_{n-k+1,n} Q_{n-k}(z) \tag{6.3}$$

with the convention that $Q_0(z) = 1$ *(*$\mathbf{H}_0 = [\,]$*, the empty matrix).*

n	#matrices	#cpolys	#stables	#neutral polys	#neutral matrices	#distinct real λ
2	8	6	1	1	2	5
3	64	32	3	0	0	29
4	1,024	289	14	1	6	233
5	32,768	4,958	93	0	0	7,363
6	2,097,152	162,059	992	2	430	299,477
7	268,435,456	10,318,948		0	0	

Table 6.6: Some properties of matrices from $\mathcal{H}^{n\times n}_{\{0\}}(\{-1,+1\})$. The column #stables reports the number of characteristic polynomials with all roots in the left half plane; the corresponding number of *matrices* is 1, 4, 28, 424, and 11,613. Other columns are as in previous tables. Blank table entries represent unknowns.

n	multiplicity 1	2	3	4	5	6
2	9	1				
3	65	0	0			
4	689	5	0	0		
5	20,565	3	0	0	0	
6	887,539	59	9	1	1	1

Table 6.7: Number of distinct eigenvalues of various multiplicities of matrices from $\mathcal{H}^{n\times n}_{\{0\}}(\{-1,+1\})$. The diagonal entries are not zero.

Proof. We begin by proving the following equality:

$$\det\left[\begin{array}{cccc|c} & & & & -h_{1,n} \\ & z\mathbf{I}-\mathbf{H}_{i-1} & & & \vdots \\ & & & & -h_{i-1,n} \\ \hline 0 & \cdots & 0 & -s & -h_{i,n}\end{array}\right] = -\sum_{k=1}^{i} s^{k-1}h_{i-k+1,n}Q_{i-k}(z) \tag{6.4}$$

for $1 \le i \le n$.

When $i=1$ the left side of equation (6.4) reduces to $\det\left[-h_{1,n}\right] = -h_{1,n}$, and the right side reduces to $-\sum_{k=1}^{1} s^{k-1}h_{1-k+1,n}Q_{1-k}(z) = -h_{1,n}$.

Assume inductively that

$$\det\left[\begin{array}{cccc|c} & & & & -h_{1,n} \\ & z\mathbf{I}-\mathbf{H}_{j-1} & & & \vdots \\ & & & & -h_{j-1,n} \\ \hline 0 & \cdots & 0 & -s & -h_{j,n}\end{array}\right] = -\sum_{k=1}^{j} s^{k-1}h_{j-k+1,n}Q_{j-k}(z) \tag{6.5}$$

for $i = j - 1$. Then

$$\det \left[\begin{array}{cccc|c} & & & & -h_{1,n} \\ & z\mathbf{I} - \mathbf{H}_j & & & \vdots \\ & & & & -h_{j,n} \\ \hline 0 & \cdots & 0 & -s & -h_{j+1,n} \end{array}\right] = -h_{j+1,n} \det(z\mathbf{I} - \mathbf{H}_j) + s \det \left[\begin{array}{cccc|c} & & & & -h_{1,n} \\ & z\mathbf{I} - \mathbf{H}_{j-1} & & & \vdots \\ & & & & -h_{j-1,n} \\ \hline 0 & \cdots & 0 & -s & -h_{j,n} \end{array}\right]$$

$$= -h_{j+1,n} Q_j(z) + s \left(- \sum_{k=1}^{j} s^{k-1} h_{j-k+1,n} Q_{j-k}(z) \right) \tag{6.6}$$

$$= -h_{j+1,n} Q_j(z) - \sum_{k=1}^{j} s^k h_{j-k+1,n} Q_{j-k}(z) \tag{6.7}$$

$$= - \sum_{k=0}^{j} s^k h_{j-k+1,n} Q_{j-k}(z) \tag{6.8}$$

$$= - \sum_{k=1}^{j+1} s^{k-1} h_{(j+1)-k+1,n} Q_{(j+1)-k}(z)\ . \tag{6.9}$$

Next we prove the theorem. Performing Laplace expansion on the last row of $z\mathbf{I} - \mathbf{H}_n$ we get

$$Q_n(z) = \det \left[\begin{array}{cccc|c} & & & & -h_{1,n} \\ & z\mathbf{I} - \mathbf{H}_{n-1} & & & \vdots \\ & & & & -h_{n-1,n} \\ \hline 0 & \cdots & 0 & -s & z - h_{n,n} \end{array}\right] \tag{6.10}$$

$$= (z - h_{n,n}) \det(z\mathbf{I} - \mathbf{H}_{n-1}) + s \det \left[\begin{array}{cccc|c} & & & & -h_{1,n} \\ & z\mathbf{I} - \mathbf{H}_{n-2} & & & \vdots \\ & & & & -h_{n-2,n} \\ \hline 0 & \cdots & 0 & -s & -h_{n-1,n} \end{array}\right] \tag{6.11}$$

$$= zQ_{n-1}(z) - h_{n,n} Q_{n-1}(z) + s \left(- \sum_{k=1}^{n-1} s^{k-1} h_{n-1-k+1,n} Q_{n-1-k}(z) \right) \tag{6.12}$$

$$= zQ_{n-1}(z) - h_{n,n} Q_{n-1}(z) - \sum_{k=1}^{n-1} s^k h_{n-k,n} Q_{n-1-k}(z) \tag{6.13}$$

$$= zQ_{n-1}(z) - \sum_{k=0}^{n-1} s^k h_{n-k,n} Q_{n-1-k}(z) \tag{6.14}$$

$$= zQ_{n-1}(z) - \sum_{k=1}^{n} s^{k-1} h_{n-k+1,n} Q_{n-k}(z)\,. \tag{6.15}$$

♮

Theorem 6.5.2. *Expanding $Q_n(z)$ as*

$$Q_n(z) = q_{n,n} z^n + q_{n,n-1} z^{n-1} + \cdots + q_{n,0}, \tag{6.16}$$

we can express the coefficients recursively by

$$q_{n,n} = 1, \tag{6.17a}$$

$$q_{n,j} = q_{n-1,j-1} - \sum_{k=1}^{n-j} s^{k-1} h_{n-k+1,n} q_{n-k,j} \quad \textit{for} \quad 1 \le j \le n-1, \tag{6.17b}$$

$$q_{n,0} = -\sum_{k=1}^{n} s^{k-1} h_{n-k+1,n} q_{n-k,0} \quad \textit{for} \quad n > 0, \quad \textit{and} \tag{6.17c}$$

$$q_{0,0} = 1\,. \tag{6.17d}$$

Proof. By Theorem 6.5.1

$$Q_n(z) = zQ_{n-1}(z) - \sum_{k=1}^{n} s^{k-1} h_{n-k+1,n} Q_{n-k}(z)\,. \tag{6.18}$$

The first term can be written

$$zQ_{n-1}(z) = z\left[z^{n-1} + q_{n-1,n-2} z^{n-2} + \cdots + q_{n-1,0}\right] \tag{6.19}$$

$$= z\left[z^{n-1} + \sum_{j=0}^{n-2} q_{n-1,j} z^j\right] \tag{6.20}$$

$$= z^n + \sum_{j=0}^{n-2} q_{n-1,j} z^{j+1} \tag{6.21}$$

$$= z^n + \sum_{j=1}^{n-1} q_{n-1,j-1} z^j \tag{6.22}$$

and the second term

$$s^{k-1} h_{n-k+1,n} Q_{n-k}(z) = s^{k-1} h_{n-k+1,n}\left[q_{n-k,n-k} z^{n-k} + q_{n-k,n-k-1} z^{n-k-1} + \cdots + q_{n-k,0}\right]$$

$$= s^{k-1} h_{n-k+1,n} \sum_{j=0}^{n-k} q_{n-k,j} z^j \,. \tag{6.23}$$

Therefore,

$$\begin{aligned} Q_n(z) &= z^n + \sum_{j=1}^{n-1} q_{n-1,j-1} z^j - \sum_{k=1}^{n} s^{k-1} h_{n-k+1,n} \sum_{j=0}^{n-k} q_{n-k,j} z^j \\ &= z^n + \sum_{j=1}^{n-1} q_{n-1,j-1} z^j - \sum_{j=0}^{n-1} \left(\sum_{k=1}^{n-j} s^{k-1} h_{n-k+1,n} q_{n-k,j} \right) z^j \\ &= z^n + \sum_{j=1}^{n-1} \left(q_{n-1,j-1} - \sum_{k=1}^{n-j} s^{k-1} h_{n-k+1,n} q_{n-k,j} \right) z^j - \sum_{k=1}^{n} s^{k-1} h_{n-k+1,n} q_{n-k,0} \,. \end{aligned}$$

♮

Proposition 6.5.3. *All matrices in $\mathcal{H}^{n\times n}_{\{\theta_k\}}(P)$ are non-derogatory*[8].

Proof. Let $\mathbf{H} \in \mathcal{H}^{n\times n}_{\{\theta_k\}}(P)$. Because $\mathbf{H}$ is upper Hessenberg

$$\mathbf{H}^k_{i,j} = \begin{cases} f_{i,j,k} & \text{for} \quad i < j+k \\ s^k & \text{for} \quad i = j+k \\ 0 & \text{for} \quad i > j+k \end{cases} \tag{6.24}$$

for $0 \le k \le n-1$ where $f_{i,j,k}$ are some functions of the entries of $\mathbf{H}$. Let

$$\mathbf{A} = r(\mathbf{H}) = \sum_{k=0}^{n-1} c_k \mathbf{H}^k = \mathbf{0} \,. \tag{6.25}$$

We find $\mathbf{A}_{n,1} = s^{n-1} c_{n-1} = 0$ and therefore $c_{n-1} = 0$. Continuing recursively for k from $n-2$ to 1 we find $\mathbf{A}_{k+j,j} = s^k c_k = 0$ for $1 \le j \le n-k$ and therefore $c_k = 0$ (since $c_j = 0$ for $j > k$) for $1 \le k \le n-1$. We have $\mathbf{A} = c_0 \mathbf{H}^0 = \mathbf{0}$ and hence $c_0 = 0$. Thus, no non-zero polynomial of degree less than n exists that satisfies $r(\mathbf{H}) = \mathbf{0}$. Therefore, the minimal degree non-zero polynomial that satisfies $r(\mathbf{H}) = \mathbf{0}$ is the characteristic polynomial of $\mathbf{H}$. ♮

Definition 6.5.4. The *characteristic height* of a matrix is the height of its characteristic polynomial.

[8] A non-derogatory matrix is a matrix for which its characteristic polynomial and minimal polynomial coincide (up to a factor of ±1)

Remark 6.5.5. The height of a polynomial is in fact a norm (the infinity norm of the vector of coefficients).

Proposition 6.5.6. *For any matrix* $\mathbf{A}$*,* $-\mathbf{A}$ *has the same characteristic height as* $\mathbf{A}$*.*

Proposition 6.5.7. *The maximal characteristic height of* $\mathbf{H}_n \in \mathcal{H}^{n\times n}_{\{0,\pi\}}(\{-1,0,+1\})$ *occurs when* $s^{k-1}h_{i,i+k-1} = -1$ *for* $1 \le i \le n-k+1$ *and* $1 \le k \le n$*.*

Proof. Since $s \in \{-1,+1\}$ and $h_{i,j} \in \{-1,0,+1\}$, $s^{k-1}h_{i,i+k-1} \in \{-1,0,+1\}$ and hence $\max|s^{k-1}h_{i,i+k-1}| = 1$. Let $s^{k-1}h_{i,i+k-1} = -1$. By Theorem 6.5.2

$$q_{n,0} = -\sum_{k=1}^{n} s^{k-1}h_{n-k+1,n}q_{n-k,0} \tag{6.26}$$

$$= \sum_{k=1}^{n} q_{n-k,0} \tag{6.27}$$

and

$$q_{n,j} = q_{n-1,j-1} - \sum_{k=1}^{n-j} s^{k-1}h_{n-k+1,n}q_{n-k,j} \tag{6.28}$$

$$= q_{n-1,j-1} + \sum_{k=1}^{n-j} q_{n-k,j}\ . \tag{6.29}$$

Since $q_{0,0} = 1$, and equations (6.27) and (6.29) are independent of s and $h_{i,j}$, all $q_{n,j}$ must be positive and the maximum characteristic height is attained. ♮

Remark 6.5.8. When $s = 1$ ($\theta = 0$) and $h_{i,j} = -1$ for all $1 \le i \le j \le n$, $\mathbf{H}_n$ attains maximal characteristic height. By Proposition 6.5.6, $s = -1$ ($\theta = \pi$) and $h_{i,j} = 1$ will also attain maximal characteristic height. Both of these cases correspond to upper Hessenberg matrices with a Toeplitz structure as we explore in further detail in Sections 6.6, and 6.8.

Definition 6.5.9. P is invariant under multiplication by a fixed unit $e^{i\theta}$ if $e^{i\theta}P = P$; that is, each entry of P, say p, is such that $e^{i\theta}p$ is also in P. For instance, $\{-1,0,+1\}$ is invariant under multiplication by -1. Note that invariance with respect to $e^{i\theta}$ implies invariance with respect to $e^{-i\theta}$.

Theorem 6.5.10. *Suppose* $\mathbf{H}_n \in \mathcal{H}^{n\times n}_{\{\theta_k\}}(P)$ *and* P *is invariant under multiplication by each* $e^{i\theta_k}$ *and by* $-e^{i\theta_k}$*. Then* $\mathbf{H}_n$ *is similar to a matrix in* $\mathcal{H}^{n\times n}_{\{\pi\}}(P)$*, and similar to a matrix in* $\mathcal{H}^{n\times n}_{\{0\}}(P)$*.*

Proof. We use induction. The case $n = 1$ is vacuously upper Hessenberg, though it is

$$\begin{bmatrix} e^{i\theta_k} \end{bmatrix} \begin{bmatrix} h_{11} \end{bmatrix} \begin{bmatrix} e^{-i\theta_k} \end{bmatrix} = \begin{bmatrix} h_{11} \end{bmatrix} \in \mathcal{H}^{1\times 1}_{\{\theta_k\}}(P) \,.$$

For $n > 1$, partition the matrix as

$$\begin{bmatrix} h_{11} & h_{12} \cdots & h_{1n} \\ s & & \\ & \mathbf{H}_{n-1} & \\ & & \end{bmatrix}$$

where $s = e^{i\theta_k}$ for some θ_k. Then conjugate by

$$\begin{bmatrix} 1 & & \\ & -e^{i\theta_k} & \\ & & \mathbf{I}_{n-2} \end{bmatrix} \begin{bmatrix} h_{11} & h_{12} \cdots & h_{1n} \\ s & & \\ & \mathbf{H}_{n-1} & \\ & & \end{bmatrix} \begin{bmatrix} 1 & & \\ & -e^{i\theta_k} & \\ & & \mathbf{I}_{n-2} \end{bmatrix}^{-1}$$

$$= \begin{bmatrix} h_{11} & -e^{-i\theta_k} h_{12} & \cdots \\ -1 & & \\ & \tilde{\mathbf{H}}_{n-1} & \end{bmatrix} .$$

Clearly $\tilde{\mathbf{H}}_{n-1}$ is in $\mathcal{H}^{n-1\times n-1}_{\{\theta_k\}}(P)$. By induction the proof is complete. ♮

Remark 6.5.11. For clarity, consider the case $n = 2$:

$$\mathbf{H} = \begin{bmatrix} a & b \\ s & c \end{bmatrix} , \tag{6.30}$$

where $a, b, c \in P$ and $s = e^{i\theta_k}$. Then, the following similarity transforms reduce the problem to one in $\mathcal{H}^{2\times 2}_{\{0\}}(P)$ and one in $\mathcal{H}^{2\times 2}_{\{\pi\}}(P)$.

$$\begin{bmatrix} 1 & 0 \\ 0 & e^{-i\theta_k} \end{bmatrix} \mathbf{H} \begin{bmatrix} 1 & 0 \\ 0 & e^{i\theta_k} \end{bmatrix} = \begin{bmatrix} a & be^{i\theta_k} \\ 1 & c \end{bmatrix} \tag{6.31}$$

$$\begin{bmatrix} 1 & 0 \\ 0 & -e^{-i\theta_k} \end{bmatrix} \mathbf{H} \begin{bmatrix} 1 & 0 \\ 0 & -e^{i\theta_k} \end{bmatrix} = \begin{bmatrix} a & -be^{i\theta_k} \\ -1 & c \end{bmatrix} . \tag{6.32}$$

6.6 Upper Hessenberg Toeplitz Matrices

Proposition 6.5.7 gives matrices in $\mathcal{H}^{n\times n}_{\{0,\pi\}}(\{-1,0,+1\})$ with maximal characteristic height[9]. We noticed that they are Toeplitz matrices. This motivates our interest in upper Hessenberg Toeplitz matrices.

Consider upper Hessenberg matrices with a Toeplitz structure of the form

$$\mathbf{M}_n = \begin{bmatrix} t_1 & t_2 & t_3 & \cdots & t_n \\ s & t_1 & t_2 & \cdots & t_{n-1} \\ 0 & s & t_1 & \cdots & t_{n-2} \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ 0 & \cdots & 0 & s & t_1 \end{bmatrix} \tag{6.33}$$

with $s = \exp(i\theta_k)$, (we again do not allow zero sub-diagonal entries).

Definition 6.6.1. The set of all $n \times n$ Bohemian upper Hessenberg Toeplitz matrices with upper triangle population P and sub-diagonal population from a discrete set of roots of unity, say $s \in \{e^{i\theta_k}\}$ where $\{\theta_k\}$ is some finite set of angles, is called $\mathcal{M}^{n\times n}_{\{\theta_k\}}(P)$.

We will restrict our analysis in this section to those matrices with population $\{-1,0,+1\}$ and sub-diagonals fixed at 1. We will denote this set by $\mathcal{M}^{n\times n} = \mathcal{M}^{n\times n}_{\{0\}}(\{-1,0,+1\})$. We denote the characteristic polynomial $P_n(z) \equiv \det(z\mathbf{I} - \mathbf{M}_n)$ for $\mathbf{M}_n \in \mathcal{M}^{n\times n}$.

Proposition 6.6.2. *The characteristic polynomial recurrence from Theorem 6.5.1 can be written for upper Hessenberg Toeplitz matrices in $\mathcal{M}^{n\times n}_{\{0\}}(P)$ as*

$$P_n(z) = zP_{n-1}(z) - \sum_{k=1}^{n} t_k P_{n-k}(z) \tag{6.34}$$

with the convention that $P_0(z) = 1$ ($\mathbf{M}_0 = [\,]$, the empty matrix).

Proof. For a matrix $\mathbf{M}_n \in \mathcal{M}^{n\times n}_{\{0\}}(P)$, the entries at the ith row and the $i+k-1$-th column for $1 \le i \le n-k+1$ (i.e. the $k-1$-th diagonal) are all equal to t_k. In equation (6.3), we can replace $h_{n-k+1,n}$ with t_k ($i = n-k+1$) recovering equation (6.34). ♮

Proposition 6.6.3. *The characteristic polynomial recurrence from Theorem 6.5.2 can be written for upper Hessenberg Toeplitz matrices in $\mathcal{M}^{n\times n}_{\{0\}}(P)$ as*

$$p_{n,n} = 1, \tag{6.35a}$$

[9] We did not report the numbers of such matrices and polynomials that we found in our "results" section.

$$p_{n,j} = p_{n-1,j-1} - \sum_{k=1}^{n-j} t_k p_{n-k,j} \quad \textit{for} \quad 1 \le j \le n-1, \tag{6.35b}$$

$$p_{n,0} = -\sum_{k=1}^{n} t_k p_{n-k,0}, \textit{ and} \tag{6.35c}$$

$$p_{0,0} = 1\,. \tag{6.35d}$$

Proof. Performing the same replacement as above (a notational change), we recover equation (6.35). ♮

Proposition 6.6.4. *$p_{n,i}$ is independent of t_j for $j > n - i$.*

Proof. First, assume $p_{n,\ell}$ is a function of $t_1, \ldots, t_{n-\ell}$ for $\ell = i$ and all n. By Proposition 6.6.3

$$p_{n,\ell} = p_{n-1,\ell-1} - \sum_{k=1}^{n-\ell} t_k p_{n-k,\ell}\,. \tag{6.36}$$

Isolating the $p_{n-1,\ell-1}$ term, we have

$$p_{n-1,\ell-1} = p_{n,\ell} + \sum_{k=1}^{n-\ell} t_k p_{n-k,\ell} \tag{6.37}$$

The first term, $p_{n,\ell}$, is a function of $t_1, \ldots, t_{n-\ell}$. Each term $t_k p_{n-k,\ell}$ in the sum is a function of $t_1, \ldots, t_{n-k-\ell}, t_k$. Taking $k = n - \ell$, we have the sum is a function of $t_1, \ldots, t_{n-\ell}$. Hence, $p_{n-1,\ell-1}$ is a function of $t_1, \ldots, t_{n-1-(\ell-1)} = t_{n-\ell}$.

When $i = 0$, by Proposition 6.6.3 we have

$$p_{n,0} = -\sum_{k=1}^{n} t_k p_{n-k,0} \tag{6.38}$$

which is a function of $t_1, \ldots, t_n$. ♮

Theorem 6.6.5. *The set of characteristic polynomials for all matrices $\mathbf{M}_n \in \mathcal{M}^{n\times n}$ has cardinality 3^n.*

Proof. Let

$$\mathbf{A}_n = \begin{bmatrix} a_1 & a_2 & a_3 & \cdots & a_n \\ 1 & a_1 & a_2 & \cdots & a_{n-1} \\ 0 & 1 & a_1 & \cdots & a_{n-2} \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ 0 & \cdots & 0 & 1 & a_1 \end{bmatrix} \tag{6.39}$$

with $a_k \in \{-1, 0, +1\}$ for $1 \le k \le n$. Let $R_n(z; a_1, \ldots, a_n)$ be the characteristic polynomial of $\mathbf{A}_n$. Assume $P_\ell = R_\ell$ for $\ell < n$. By Proposition 6.6.2, for $\mathbf{A}_n$ and $\mathbf{M}_n$ to have the same characteristic polynomial we find

$$zP_{n-1} - \sum_{k=1}^{n} t_k P_{n-k} = zR_{n-1} - \sum_{k=1}^{n} a_k R_{n-k}\,. \tag{6.40}$$

Since $P_\ell = R_\ell$ for all $\ell < n$, and the $\sum_{k=1}^{n} t_k P_{n-k}$ and $\sum_{k=1}^{n} t_k R_{n-k}$ terms are polynomials of degree $n-1$ in z, we find $P_n = R_n$ only when $t_k = a_k$ for all $1 \le k \le n$ (the zP_{n-1} and zR_{n-1} terms are the only terms of degree n in z). Hence, for each combination of t_k, no other upper Hessenberg Toeplitz matrix with $t_k \in \{-1, 0, +1\}$ and sub-diagonal 1 has the same characteristic polynomial. ♮

Proposition 6.6.6. *The characteristic height of $\mathbf{M}_n \in \mathcal{M}^{n\times n}$ is maximal when $t_k = -1$ for $1 \le k \le n$.*

Proof. Following from Proposition 6.5.7, the entries in the ith row and $i+k-1$-th column for $1 \le i \le n-k+1$ correspond to t_k, after substituting $s = 1$ we find $t_k = -1$ gives the maximal characteristic height. ♮

Proposition 6.6.7. *Let $F \subset \mathbb{R}$ be a closed and bounded set with $a = \min(F)$, $b = \max(F)$ and $\#F \ge 2$. Let $\mathbf{M}_n \in \mathcal{M}_{\{0\}}^{n\times n}(F)$. If $|a| \ge |b|$, $\mathbf{M}_n$ is of maximal characteristic height when $t_k = a$ for all $1 \le k \le n$. If $|b| \ge |a|$, $\mathbf{M}_n$ is of maximal characteristic height for $t_k = a$ for k even, and $t_k = b$ for k odd.*

Proof. First, consider the case when $|a| \ge |b|$. Since $a < b$ we find $a < 0$. Let $\bar{t}_k = -t_k$. Writing Proposition 6.5.6 in terms of $\bar{t}_k$ gives

$$p_{n,n} = 1, \tag{6.41a}$$

$$p_{n,j} = p_{n-1,j-1} + \sum_{k=1}^{n-j} \bar{t}_k p_{n-k,j} \quad \text{for} \quad 1 \le j \le n-1, \tag{6.41b}$$

$$p_{n,0} = \sum_{k=1}^{n} \bar{t}_k p_{n-k,0}, \text{ and} \tag{6.41c}$$

$$p_{0,0} = 1\,. \tag{6.41d}$$

If all $\bar{t}_k$ are positive then $p_{n,j}$ must be positive for all n and j. Hence, the maximal characteristic height is attained when $\bar{t}_k$ is maximal, or equivalently t_k is minimal and negative. Thus $t_k = \min(F) = a$ gives maximal characteristic height.

Next, consider when $|b| \geq |a|$. Since $a < b$ we find $b > 0$. By Proposition 6.5.6 we know that the characteristic height of $\mathbf{M}_n$ is equal to the characteristic height of $-\mathbf{M}_n$. Rewriting Proposition 6.6.3 for $-\mathbf{M}_n$ by substituting $p_{n,j}$ with $(-1)^{n-j}p_{n,j}$ we find the recurrence for the characteristic polynomial of $-\mathbf{M}_n$:

$$p_{n,n} = 1, \tag{6.42a}$$

$$p_{n,j} = p_{n-1,j-1} + \sum_{k=1}^{n-j}(-1)^{k-1}t_k p_{n-k,j} \quad \text{for} \quad 1 \leq j \leq n-1, \tag{6.42b}$$

$$p_{n,0} = \sum_{k=1}^{n}(-1)^{k-1}t_k p_{n-k,0}, \text{ and} \tag{6.42c}$$

$$p_{0,0} = 1\,. \tag{6.42d}$$

Separating out the even and odd values of k in the sums we can write the recurrence as

$$p_{n,n} = 1, \tag{6.43a}$$

$$p_{n,j} = p_{n-1,j-1} + \sum_{k \text{ odd}}^{n-j} t_k p_{n-k,j} - \sum_{k \text{ even}}^{n-j} t_k p_{n-k,j} \quad \text{for} \quad 1 \leq j \leq n-1, \tag{6.43b}$$

$$p_{n,0} = \sum_{k \text{ odd}}^{n} t_k p_{n-k,0} - \sum_{k \text{ even}}^{n} t_k p_{n-k,0}, \text{ and} \tag{6.43c}$$

$$p_{0,0} = 1\,. \tag{6.43d}$$

The odd sums are maximal for $t_k = \max(F) = b$ and the even sums are maximal for $t_k = \min(F) = a$. Hence, the maximal characteristic height is attained for $t_k = b$ when k is odd, and $t_k = a$ when k is even.

When $|a| = |b|$, equations (6.41) and (6.43) are equivalent and the maximal height is attained both when $t_k = b$ for all k, and $t_k = b$ for k odd and $t_k = a$ for k even. ♮

Proposition 6.6.8. $\mathbf{M}_n \in \mathcal{M}^{n\times n}$ *also attains maximal characteristic height when* $t_k = (-1)^{k-1}$ *for* $1 \leq k \leq n$*.*

Proof. By Proposition 6.6.7, we have $F = \{-1, 0, +1\}$ with $a = -1$, and $b = +1$. Thus $\mathbf{M}_n$ is also of maximal characteristic height for $t_k = b = +1$ for odd values of k, and $t_k = a = -1$ for even values of k. ♮

6.7 Maximal Characteristic Height Upper Hessenberg Toeplitz Matrices

In this section we restrict our analysis to those matrices in $\mathcal{M}^{n\times n}$ of maximal characteristic height. We denote this subset by $\overline{\mathcal{M}}^{n\times n}$. Let τ_n be the characteristic height of $\overline{\mathcal{M}}^{n\times n}$ (the height is the same for all matrices in $\overline{\mathcal{M}}^{n\times n}$) and let μ_n be the degree of the term of the characteristic polynomial of $\overline{\mathbf{M}}_n \in \overline{\mathcal{M}}^{n\times n}$ whose coefficient gives the height. In Proposition 6.7.5 we prove that μ_n the same for all matrices in $\overline{\mathcal{M}}^{n\times n}$. τ_n and μ_n and the number of matrices with maximal characteristic height for dimensions 2 to 10 are given in Table 6.8.

n	τ_n	μ_n	# max char height
2	2	1	6
3	5	1	6
4	12	1	6
5	27	1	6
6	66	2	18
7	168	2	18
8	416	2	18
9	1,008	2	18
10	2,528	3	54

Table 6.8: Maximum height, τ_n, degree of term of characteristic polynomial corresponding to maximum height, μ_n, and the number of matrices in $\overline{\mathcal{M}}^{n\times n}$ for dimensions 2 to 10.

Proposition 6.7.1. *The characteristic height, τ_n grows at least exponentially in n.*

Proof. When $t_k = -1$ for $1 \le k \le n$, the characteristic height is maximal by Proposition 6.6.6. Equation (6.35c) from Proposition 6.6.3 reduces to

$$p_{n,0} = \sum_{k=1}^{n} p_{n-k,0} = 2^{n-1} \tag{6.44}$$

for $n \ge 1$ with $p_{0,0} = 1$ by equation (6.35d). Thus, the maximal characteristic height must grow at least exponentially in n. ♮

Conjecture 6.7.2. *The maximum characteristic height, τ_n, approaches $C(1+\varphi)^n$ as $n \to \infty$ for some constant C where φ is the golden ratio.*

Remark 6.7.3. This limit is illustrated in Figure 6.5, motivating this conjecture.

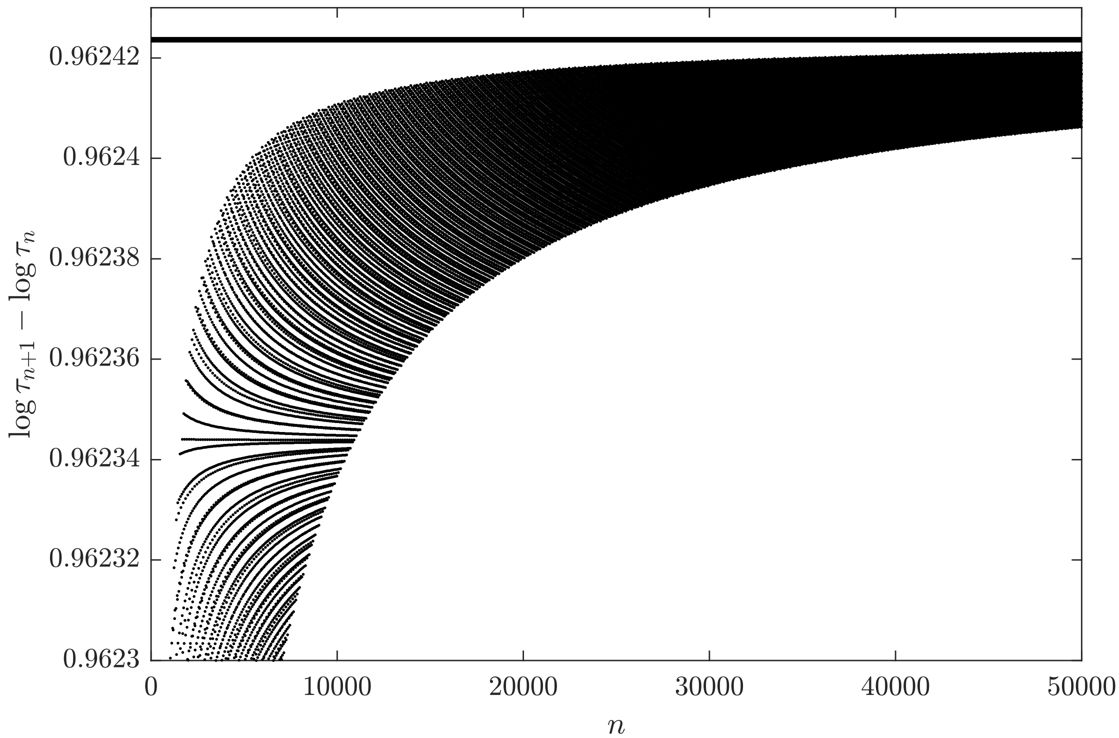


Figure 6.5: The points are $\log\tau_{n+1} - \log\tau_n$ for n from 0 to 50,000 where τ_n is the maximal characteristic height of $\mathcal{M}^{n\times n}$ (i.e. when $t_k = -1$, for example). The solid line is $\log(1+\varphi)$ where φ is the golden ratio.

Proposition 6.7.4. *The characteristic height, τ_n is independent of t_j for $j > n - \mu_n$.*

Proof. Let P_n be the characteristic polynomial of $\overline{\mathbf{M}}_n \in \overline{\mathcal{M}}^{n\times n}$. By Proposition 6.6.4, p_{n,μ_n} is independent of t_j for $j > n - \mu_n$. Thus, t_j for $j > n - \mu_n$ only affects $p_{n,k}$ for $k < \mu_n$. Since $\overline{\mathbf{M}}_n$ is of maximal height, $|p_{n,k}| \le |p_{n,\mu_n}|$ for $k < \mu_n$ for all $t_j \in \{-1, 0, +1\}$ with $j > n - \mu_n$. ♮

Proposition 6.7.5. *For fixed n, μ_n is the same for all $\overline{\mathbf{M}}_n \in \overline{\mathcal{M}}^{n\times n}$.*

Proof. The characteristic polynomial of $\overline{\mathbf{M}}_n$ when $t_k = -1$ has the same coefficients as the characteristic polynomial of $\overline{\mathbf{M}}_n$ for $t_k = (-1)^{k-1}$ up to a sign change. By Proposition 6.7.4, changing any of the entries t_j of $\overline{\mathbf{M}}_n$ for $j > n - \mu_n$ does not affect the value of μ_n. Therefore μ_n is fixed. ♮

Theorem 6.7.6. *$\overline{\mathcal{M}}^{n\times n}$ contains $2 \cdot 3^{\mu_n}$ matrices.*

Proof. By Proposition 6.6.6, $t_k = -1$ for $1 \le k \le n$ gives maximal characteristic height. By Proposition 6.7.4, any combination of $t_j \in \{-1, 0, +1\}$ for $j > n - \mu_n$ will not affect the characteristic height. Thus, the 3^{μ_n} matrices with $t_k = -1$ for $1 \le k \le n - \mu_n$, and $t_k \in \{-1, 0, +1\}$ for $n - \mu_n + 1 \le k \le n$ all have maximal characteristic height. Similarly, by Proposition 6.6.8, $t_k = (-1)^{k-1}$ for $1 \le k \le n$ gives maximal characteristic height. Again, by Proposition 6.7.4, 3^{μ_n} matrices with $t_k = (-1)^{k-1}$ for $1 \le k \le n - \mu_n$, and $t_k \in \{-1, 0, +1\}$ for $n - \mu_n + 1 \le k \le n$ all have maximal characteristic height. ♮

Remark 6.7.7. We have found that μ_n remains constant for 3 or 4 subsequent values of n followed by an increment by 1. We have verified this pattern experimentally up to degree 50,000. Figure 6.6 shows the pattern for matrix dimension up to 100.

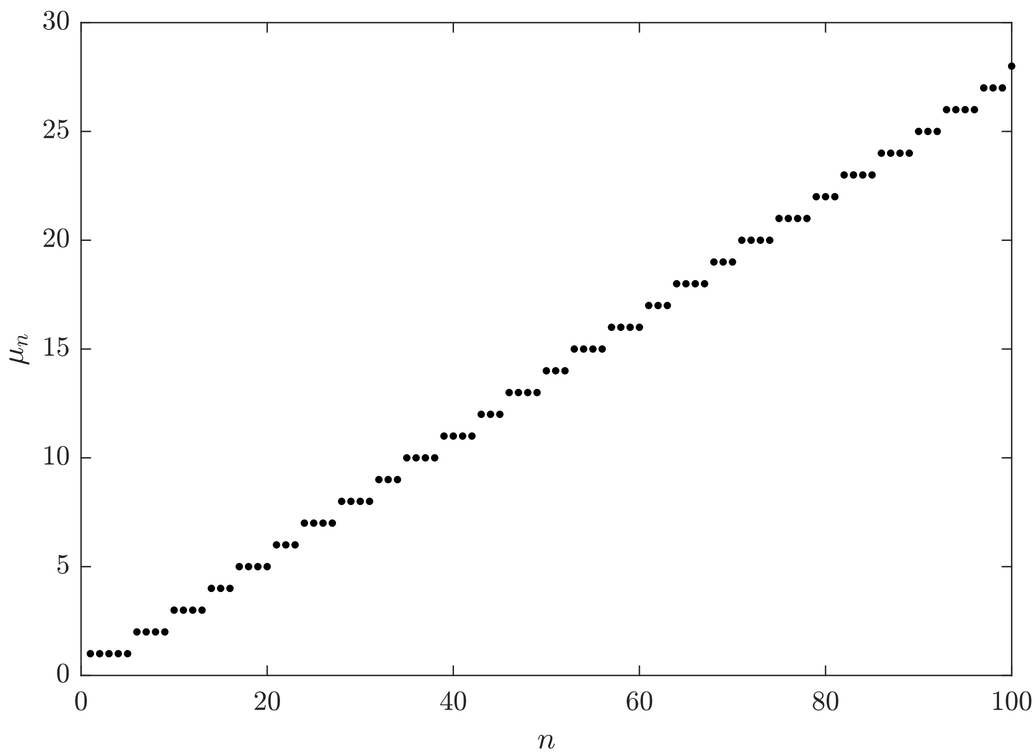


Figure 6.6: Degree of the term corresponding to the height of the characteristic polynomial of an $n \times n$ upper Hessenberg Toeplitz matrix of maximal characteristic height.

Remark 6.7.8. The sequence $\mu_{n+1} - \mu_n$ is nearly equivalent to the sequence for the generalized Fibonacci word $f^{[3]}$

$$a(n) = \left\lfloor \frac{n+2}{\varphi+2} \right\rfloor - \left\lfloor \frac{n+1}{\varphi+2} \right\rfloor \tag{6.45}$$

(A221150 on the OEIS). We have found that up to at least degree 50,000, $\mu_{n+1} - \mu_n = a(n + 326)$ except when $n \in \{0, 2, 24{,}148, 24{,}149\}$.

Remark 6.7.9. The sequence μ_n is nearly equivalent to the sequence

$$\left\lfloor \frac{n+327}{\varphi+2} \right\rfloor - 90 \tag{6.46}$$

for $n > 2$. The two sequences are equal for all values up to $n = 50{,}000$ except when $n = 24{,}149$.

The sequences presented in the previous remarks are examples of *high-precision fraud* [5] requiring evaluation up to dimension 25,000 and nearly 25,000 digits of precision to identity.

6.8 Maximal Height Characteristic Polynomials

In this section we restrict our analysis to the matrix $\widetilde{\mathbf{M}}_n \in \overline{\mathcal{M}}^{n\times n}$ with $t_k = -1$ for all k. By Proposition 6.6.6, $\widetilde{\mathbf{M}}_n$ is of maximal characteristic height.

Proposition 6.8.1. *The characteristic polynomial of* $\widetilde{\mathbf{M}}_n$ *is of the form*

$$P_n = z^n + p_{n,n-1}z^{n-1} + \cdots + p_{n,0} \tag{6.47}$$

where $p_{n,j}$ *is positive for all* n *and* j.

Proof. When $t_k = -1$ for $1 \le k \le n$, Proposition 6.6.3 reduces to

$$p_{n,n} = 1, \tag{6.48a}$$

$$p_{n,j} = p_{n-1,j-1} + \sum_{k=1}^{n-j} p_{n-k,j} \quad \text{for} \quad 1 \le j \le n-1, \tag{6.48b}$$

$$p_{n,0} = \sum_{k=1}^{n} p_{n-k,0}, \text{ and} \tag{6.48c}$$

$$p_{0,0} = 1\,. \tag{6.48d}$$

Since $p_{0,0}$ is positive, and all coefficients in the above equations are positive, $p_{n,j}$ must be positive for all n and j. ♮

Proposition 6.8.2. *The generating function of the sequence* $(p_{i,i}, p_{i+1,i}, \ldots)$ *for all* $i \ge 0$ *is*

$$G_i(x) = \left(\frac{1-x}{1-2x}\right)^{i+1}. \tag{6.49}$$

Proof. First we will prove the $i = 0$ case. Let

$$G_0(x) = \sum_{\ell=0}^{\infty} p_{\ell,0} x^{\ell} \, . \tag{6.50}$$

Then,

$$(1-2x)G_0(x) = p_{0,0} + \sum_{\ell=1}^{\infty} (p_{\ell,0} - 2p_{\ell-1,0}) x^{\ell} \, . \tag{6.51}$$

From equation (6.48c),

$$\begin{aligned}
(1-2x)G_0(x) &= p_{0,0} + (p_{1,0} - 2p_{0,0})x + \sum_{\ell=2}^{\infty} (p_{\ell,0} - 2p_{\ell-1,0}) x^{\ell} && (6.52)\\
&= p_{0,0} + (p_{1,0} - 2p_{0,0})x + \sum_{\ell=2}^{\infty} \left(\sum_{k=1}^{\ell} p_{\ell-k,0} - 2 \sum_{k=1}^{\ell-1} p_{\ell-1-k,0} \right) x^{\ell} && (6.53)\\
&= p_{0,0} + (p_{1,0} - 2p_{0,0})x + \sum_{\ell=2}^{\infty} \left(\sum_{k=1}^{\ell} p_{\ell-k,0} - 2 \sum_{k=2}^{\ell} p_{\ell-k,0} \right) x^{\ell} && (6.54)\\
&= p_{0,0} + (p_{1,0} - 2p_{0,0})x + \sum_{\ell=2}^{\infty} \left(p_{\ell-1,0} - \sum_{k=2}^{\ell} p_{\ell-k,0} \right) x^{\ell} \, . && (6.55)
\end{aligned}$$

Since $p_{0,0} = p_{1,0} = 1$,

$$\begin{aligned}
(1-2x)G_0(x) &= 1 - x + \sum_{\ell=2}^{\infty} \left(p_{\ell-1,0} - \sum_{k=1}^{\ell-1} p_{\ell-1-k,0} \right) x^{\ell} && (6.56)\\
&= 1 - x \, . && (6.57)
\end{aligned}$$

Therefore

$$G_0(x) = \frac{1-x}{1-2x} \, . \tag{6.58}$$

Next we prove the general case for $i > 0$. Assume inductively that

$$G_i(x) = \left(\frac{1-x}{1-2x} \right)^{i+1} = \sum_{\ell=0}^{\infty} p_{i+\ell,i} x^{\ell} \, . \tag{6.59}$$

$$\sum_{\ell=0}^{\infty} p_{i+\ell+1,i+1} x^{\ell} = \left(\frac{1-2x}{1-2x} \right) \sum_{\ell=0}^{\infty} p_{i+\ell+1,i+1} x^{\ell}$$

$$= \left(\frac{1}{1-2x}\right)\left[\sum_{\ell=0}^{\infty} p_{i+\ell+1,i+1}x^\ell - 2x\sum_{\ell=0}^{\infty} p_{i+\ell+1,i+1}x^\ell\right]$$
$$= \left(\frac{1}{1-2x}\right)\left[p_{i+1,i+1} + \sum_{\ell=1}^{\infty}(p_{i+\ell+1,i+1} - 2p_{i+\ell,i+1})x^\ell\right]$$

Because $p_{i+1,i+1} = 1 = p_{i,i}$

$$= \left(\frac{1}{1-2x}\right)\left[p_{i,i} + \sum_{\ell=1}^{\infty}(p_{i+\ell+1,i+1} - 2p_{i+\ell,i+1})x^\ell\right]$$
$$= \left(\frac{1}{1-2x}\right)\left[p_{i,i} + \sum_{\ell=1}^{\infty}\left(p_{i+\ell+1,i+1} - p_{i+\ell,i+1} - p_{i+\ell,i+1}\right)x^\ell\right]$$
$$= \left(\frac{1}{1-2x}\right)\left[p_{i,i} + \sum_{\ell=1}^{\infty}\left(p_{i+\ell+1,i+1} - p_{i+\ell,i+1} - \sum_{k=0}^{\ell-1} p_{i+\ell-k,i+1} + \sum_{k=1}^{\ell-1} p_{i+\ell-k,i+1}\right)x^\ell\right]$$
$$= \left(\frac{1}{1-2x}\right)\left[p_{i,i} + \sum_{\ell=1}^{\infty}\left(p_{i+\ell+1,i+1} - p_{i+\ell,i+1} - \sum_{k=0}^{\ell-1} p_{i+\ell-k,i+1} + \sum_{k=1}^{\ell-1} p_{i+\ell-k,i+1}\right)x^\ell\right]$$
$$= \left(\frac{1}{1-2x}\right)\left[p_{i,i} + \sum_{\ell=1}^{\infty}\left(\left(p_{i+\ell+1,i+1} - \sum_{k=1}^{\ell} p_{i+\ell+1-k,i+1}\right) - \left(p_{i+\ell,i+1} - \sum_{k=1}^{\ell-1} p_{i+\ell-k,i+1}\right)\right)\right.$$

Rewriting equation (6.48b) as

$$p_{n,j} = p_{n+1,j+1} - \sum_{k=1}^{n-j} p_{n+1-k,j+1}, \tag{6.60}$$

we find

$$\sum_{\ell=0}^{\infty} p_{i+\ell+1,i+1}x^\ell = \left(\frac{1}{1-2x}\right)\left[p_{i,i} + \sum_{\ell=1}^{\infty}(p_{i+\ell,i} - p_{i+\ell-1,i})x^\ell\right]$$
$$= \left(\frac{1}{1-2x}\right)\left[\sum_{\ell=0}^{\infty} p_{i+\ell,i}x^\ell - \sum_{\ell=1}^{\infty} p_{i+\ell-1,i}x^\ell\right]$$
$$= \left(\frac{1}{1-2x}\right)\left[\sum_{\ell=0}^{\infty} p_{i+\ell,i}x^\ell - \sum_{\ell=0}^{\infty} p_{i+\ell,i}x^{\ell+1}\right]$$
$$= \left(\frac{1-x}{1-2x}\right)\sum_{\ell=0}^{\infty} p_{i+\ell,i}x^\ell$$
$$= \left(\frac{1-x}{1-2x}\right)^{i+2}$$

♮

Proposition 6.8.3. *The coefficients $p_{n,k}$ are given by the OEIS sequence A105306 for the "number of directed column-convex polynomials of area n, having the top of the right-most column at height k." We have $p_{n,k} = T_{n+1,k+1}$ where*

$$T_{n,k} = \begin{cases} \sum_{j=0}^{n-k-1} \binom{k+j}{k-1} \binom{n-k-1}{j} & \text{if } k < n \\ 1 & \text{if } k = n \end{cases} \tag{6.61}$$

MAPLE *"simplifies" this to*

$$T_{n,k} = \begin{cases} kF\left(\left. \begin{matrix} k+1, k+1-n \\ 2 \end{matrix} \right| -1 \right) & \text{if } n \neq k \\ 1 & \text{if } n = k \end{cases} \tag{6.62}$$

where $F(\cdot)$ is the hypergeometric function defined as

$$F\left(\left. \begin{matrix} a, b \\ c \end{matrix} \right| z \right) = \sum_{n=0}^{\infty} \frac{a^{\bar{n}} b^{\bar{n}}}{c^{\bar{n}}} \frac{z^n}{n!} \tag{6.63}$$

where $q^{\bar{n}}$ is $q \cdot (q+1) \cdots (q+n-1)$.

Proof. We will show that

$$p_{i+n,i} = T_{n+i+1,i+1} = \begin{cases} \sum_{j=0}^{n-1} \binom{i+j+1}{i} \binom{n-1}{j} & \text{if } n > 0 \\ 1 & \text{if } n = 0\,. \end{cases} \tag{6.64}$$

By Proposition 6.8.2

$$p_{i+n,i} = \frac{1}{n!} \frac{\mathrm{d}^n}{\mathrm{d}x^n} G_i(x) \Big|_{x=0} \tag{6.65}$$

where

$$G_i(x) = \left(\frac{1-x}{1-2x} \right)^{i+1} = f_i(g(x)) \tag{6.66}$$

with

$$f_i(x) = x^{i+1}, \text{ and} \tag{6.67}$$

$$g(x) = \frac{1-x}{1-2x} = \frac{1}{1-2x} - \frac{x}{1-2x}\,. \tag{6.68}$$

Differentiating $f_i(x)$ and $g(x)$ with respect to x,

$$\frac{\mathrm{d}^n}{\mathrm{d}x^n} f_i(x) = \begin{cases} (i+1)(i)\cdots(i-n+2)x^{i+1-n} & \text{for} \quad n \le i+1 \\ 0 & \text{for} \quad n > i+1 \end{cases} \tag{6.69}$$

$$= \binom{i+1}{n} n! x^{i+1-n} \tag{6.70}$$

and

$$\frac{\mathrm{d}^n}{\mathrm{d}x^n} g(x) = \frac{\mathrm{d}^n}{\mathrm{d}x^n}\frac{1}{1-2x} + \frac{\mathrm{d}^n}{\mathrm{d}x^n}\frac{x}{1-2x} \tag{6.71}$$

$$= \frac{2^n n!}{(1-2x)^{n+1}} + \frac{2^{n-1}n!}{(1-2x)^n} + \frac{2^n n!\, x}{(1-2x)^{n+1}} \tag{6.72}$$

$$= \frac{2^n n!(1-x)}{(1-2x)^{n+1}} - \frac{2^{n-1}n!}{(1-2x)^n} \tag{6.73}$$

with

$$\left.\frac{\mathrm{d}^n}{\mathrm{d}x^n} g(x)\right|_{x=0} = \begin{cases} n!\, 2^{n-1} & \text{for} \quad n > 0 \\ 1 & \text{for} \quad n = 0\,. \end{cases} \tag{6.74}$$

When $n = 0$,

$$p_{i+n,i} = p_{i,i} = G_i(0) = 1\,. \tag{6.75}$$

For $n > 0$, by Faà di Bruno's formula we have

$$\frac{\mathrm{d}^n}{\mathrm{d}x^n} G_i(x) = \frac{\mathrm{d}^n}{\mathrm{d}x^n} f_i(g(x)) \tag{6.76}$$

$$= \sum_{k=1}^{n} f_i^{(k)}\big(g(x)\big) B_{n,k}(g'(x), g''(x), \ldots, g^{(n-k+1)}(x)) \tag{6.77}$$

and therefore

$$\left.\frac{\mathrm{d}^n}{\mathrm{d}x^n} G_i(x)\right|_{x=0} = \sum_{k=1}^{n} f_i^{(k)}\big(g(0)\big) B_{n,k}(g'(0), g''(0), \ldots, g^{(n-k+1)}(0)) \tag{6.78}$$

$$= \sum_{k=1}^{n} f_i^{(k)}(1) B_{n,k}(1, 4, 24, \ldots, (n-k+1)!2^{n-k})\,. \tag{6.79}$$

By Theorem 6 of [1],

$$B_{n,k}(1, 4, 24, \ldots, (n-k+1)!2^{n-k}) = B_{n,k}(q_0(1), q_1(2), \ldots, q_{n-k}(n-k+1)) \tag{6.80}$$

$$= \binom{n-1}{k-1} \frac{n!}{k!} 2^{n-k} \tag{6.81}$$

because the function

$$q_n(x) = \frac{x!}{(x-n)!} 2^n \tag{6.82}$$

satisfies

$$q_n(x+y) = \sum_{k=0}^{n} \binom{n}{k} q_k(y) q_{n-k}(x) \,. \tag{6.83}$$

Returning to the proof,

$$p_{i+n,i} = \frac{1}{n!} \frac{\mathrm{d}^n}{\mathrm{d}x^n} G_i(x)\Big|_{x=0} \tag{6.84}$$

$$= \frac{1}{n!} \sum_{k=1}^{n} \binom{i+1}{k} \binom{n-1}{k-1} k! \frac{n!}{k!} 2^{n-k} \tag{6.85}$$

$$= \sum_{k=1}^{n} \binom{i+1}{k} \binom{n-1}{k-1} 2^{n-k} \tag{6.86}$$

$$= \sum_{k=0}^{n-1} \binom{i+1}{k+1} \binom{n-1}{k} 2^{n-k-1} \tag{6.87}$$

$$= \sum_{k=0}^{n-1} \binom{i+1}{k+1} \binom{n-1}{k} \sum_{j=0}^{n-k-1} \binom{n-k-1}{j} \tag{6.88}$$

$$= \sum_{k=0}^{n-1} \sum_{j=0}^{n-k-1} \binom{n-1}{k} \binom{i+1}{k+1} \binom{n-k-1}{j} \tag{6.89}$$

$$= \sum_{j=0}^{n-1} \sum_{k=0}^{n-j-1} \binom{n-1}{k} \binom{i+1}{k+1} \binom{n-k-1}{j} \tag{6.90}$$

$$= \sum_{j=0}^{n-1} \sum_{k=0}^{j} \binom{n-1}{k} \binom{i+1}{k+1} \binom{n-k-1}{n-j-1} \tag{6.91}$$

$$= \sum_{j=0}^{n-1} \sum_{k=0}^{j} \binom{n-1}{n-j-1} \binom{j}{k} \binom{i+1}{k+1} \tag{6.92}$$

$$= \sum_{j=0}^{n-1} \binom{n-1}{j} \sum_{k=0}^{j} \binom{j}{k} \binom{i+1}{k+1} \tag{6.93}$$

$$= \sum_{j=0}^{n-1} \binom{n-1}{j} \binom{i+j+1}{j+1} \tag{6.94}$$

$$= \sum_{j=0}^{n-1} \binom{n-1}{j} \binom{i+j+1}{i} \tag{6.95}$$

$\natural$

Proposition 6.8.4. *The characteristic polynomial of* $\widetilde{\mathbf{M}}_n$ *is*

$$P_n(z) = \sum_{\ell=0}^{\lfloor n/2 \rfloor} \binom{n}{2\ell}\left(\frac{z}{2}+1\right)^{n-2\ell}\left(1+\frac{z^2}{4}\right)^{\ell} + \frac{z}{2}\sum_{\ell=0}^{\lfloor \frac{n-1}{2} \rfloor}\binom{n}{2\ell+1}\left(\frac{z}{2}+1\right)^{n-2\ell-1}\left(1+\frac{z^2}{4}\right)^{\ell}.$$

This proposition can be proved in several ways. We choose below to think of $z \in \mathbb{C} \setminus \{\pm 2i\}$, for a reason that will become clear. Since the end result is a polynomial in z, proving the formula for $z \neq \pm 2i$ will recover the exceptional cases by continuity.

Another equally valid approach would be to think of z as being transcendental and noting that the characteristic polynomial of $\widetilde{\mathbf{M}}_n$ has integer coefficients.

Proof. From Proposition 6.6.2

$$P_n(z) = zP_{n-1}(z) - \sum_{k=1}^{n} t_k P_{n-k}(z) \tag{6.96}$$

$$= zP_{n-1}(z) - \sum_{k=0}^{n-1} t_{n-k} P_k(z)\,. \tag{6.97}$$

If $t_k = -1$ for $1 \le k \le n$,

$$P_n(z) = zP_{n-1}(z) + \sum_{k=0}^{n-1} P_k(z)\,. \tag{6.98}$$

Let $T_j(z) = \sum_{k=0}^{j} P_k(z)$. $T_n(z) = T_{n-1}(z) + P_n(z)$, so

$$P_n(z) = zP_{n-1}(z) + T_{n-1}(z) \tag{6.99}$$

$$T_n(z) = zP_{n-1}(z) + 2T_{n-1}(z) \tag{6.100}$$

or

$$\begin{bmatrix} P_n(z) \\ T_n(z) \end{bmatrix} = \begin{bmatrix} z & 1 \\ z & 2 \end{bmatrix}^n \begin{bmatrix} P_0(z) \\ T_0(z) \end{bmatrix} \tag{6.101}$$

$$= \begin{bmatrix} z & 1 \\ z & 2 \end{bmatrix}^n \begin{bmatrix} 1 \\ 1 \end{bmatrix} \tag{6.102}$$

since $P_0(z) = 1$ and $T_0(z) = \sum_{j=0}^{0} P_0(z) = 1$. The eigenvalues of this matrix are

$$\lambda_+ = 1 + \frac{z}{2} + \Delta \tag{6.103}$$
$$\lambda_- = 1 + \frac{z}{2} - \Delta \tag{6.104}$$
$$\Delta = \sqrt{1 + z^2/4}\,. \tag{6.105}$$

If $z = \pm 2i$ the eigenvalues are multiple and our approach would have to be modified. We ignore this and recover the true result at the end. The eigenvectors are

$$\mathbf{V} = \begin{bmatrix} 1 & 1 \\ 1 - \frac{z}{2} + \Delta & 1 - \frac{z}{2} - \Delta \end{bmatrix} \tag{6.106}$$

and

$$\mathbf{V}^{-1} = \frac{-1}{2\Delta} \begin{bmatrix} 1 - \frac{z}{2} - \Delta & -1 \\ -1 + \frac{z}{2} - \Delta & 1 \end{bmatrix} \tag{6.107}$$

hence

$$\mathbf{V}^{-1} \begin{bmatrix} 1 \\ 1 \end{bmatrix} = \frac{-1}{2\Delta} \begin{bmatrix} \frac{-z}{2} - \Delta \\ \frac{z}{2} - \Delta \end{bmatrix} = \begin{bmatrix} \frac{1}{2} + \frac{z}{4\Delta} \\ \frac{1}{2} - \frac{z}{4\Delta} \end{bmatrix}. \tag{6.108}$$

Therefore

$$\begin{bmatrix} P_n(z) \\ T_n(z) \end{bmatrix} = \begin{bmatrix} 1 & 1 \\ 1 - \frac{z}{2} + \Delta & 1 - \frac{z}{2} - \Delta \end{bmatrix} \begin{bmatrix} \lambda_+^n \left(\frac{1}{2} + \frac{z}{4\Delta}\right) \\ \lambda_-^n \left(\frac{1}{2} - \frac{z}{4\Delta}\right) \end{bmatrix} \tag{6.109}$$

and in particular

$$P_n(z) = \lambda_+^n \left(\frac{1}{2} + \frac{z}{4\Delta}\right) + \lambda_-^n \left(\frac{1}{2} - \frac{z}{4\Delta}\right). \tag{6.110}$$

Now

$$\lambda_+^n = \left(\frac{z}{2} + 1 + \Delta\right)^n \tag{6.111}$$
$$= \sum_{k=0}^{n} \binom{n}{k} \left(\frac{z}{2} + 1\right) \Delta^k \tag{6.112}$$

and

$$\lambda_-^n = \left(\frac{z}{2} + 1 - \Delta\right)^n \tag{6.113}$$
$$= \sum_{k=0}^{n} \binom{n}{k} \left(\frac{z}{2} + 1\right) (-\Delta)^k\,. \tag{6.114}$$

$$\therefore P_n(z) = \sum_{k=0}^{n} \binom{n}{k} \left(\frac{z}{2}+1\right)^{n-k} \left(\frac{1}{2}\Delta^k + \frac{1}{2}(-\Delta)^k\right) + \frac{z}{4\Delta} \sum_{k=0}^{n} \binom{n}{k} \left(\frac{z}{2}+1\right)^{n-k} \left(\Delta^k - (-\Delta)^k\right) . \quad (6.115)$$

Every odd term drops out of the first, and every even out of the second.

$$\therefore P_n(z) = \sum_{\substack{k=0 \\ k \text{ even}}}^{n} \binom{n}{k} \left(\frac{z}{2}+1\right)^{n-k} \Delta^k + \frac{z}{4\Delta} \sum_{\substack{k=0 \\ k \text{ odd}}}^{n} \binom{n}{k} \left(\frac{z}{2}+1\right)^{k} \cdot 2\Delta^k$$

$$= \sum_{\ell=0}^{\lfloor n/2 \rfloor} \binom{n}{2\ell} \left(\frac{z}{2}+1\right)^{n-2\ell} \left(1+\frac{z^2}{4}\right)^{\ell} + \frac{z}{2} \sum_{\ell=0}^{\lfloor \frac{n-1}{2} \rfloor} \binom{n}{2\ell+1} \left(\frac{z}{2}+1\right)^{n-2\ell-1} \left(1+\frac{z^2}{4}\right)^{\ell} .$$

At this point the difficulty with $\Delta = 0$ has been resolved by continuity. We see that $P_n(z)$ is a polynomial of degree n. ♮

6.9 A Connection with Compositions

Consider the case with symbolic entries t_i, and sub-diagonals -1 for convenience with minus signs in the formulae. For instance, the 5 by 5 example upper Hessenberg Toeplitz matrix is

$$\mathbf{M}_5 = \begin{bmatrix} t_1 & t_2 & t_3 & t_4 & t_5 \\ -1 & t_1 & t_2 & t_3 & t_4 \\ 0 & -1 & t_1 & t_2 & t_3 \\ 0 & 0 & -1 & t_1 & t_2 \\ 0 & 0 & 0 & -1 & t_1 \end{bmatrix} . \quad (6.116)$$

In this section we consider what happens when we take determinants $P_n(z) = \det(z\mathbf{I}-\mathbf{M}_n)$. Examining $P_0(0)$, $P_1(0)$, $P_2(0)$, $P_3(0)$, and $P_4(0)$, and in particular $P_k(0)$ (i.e. $\det(-\mathbf{M}_k)$) we see that

$$P_0(0) = 1 \text{ by convention} \quad (6.117)$$
$$P_1(0) = t_1 \quad (6.118)$$
$$P_2(0) = t_1^2 + t_2 \quad (6.119)$$
$$P_3(0) = t_1^3 + 2t_1t_2 + t_3 \quad (6.120)$$
$$P_4(0) = t_1^4 + 3t_1^2t_2 + 2t_1t_3 + t_2^2 + t_4 . \quad (6.121)$$

One may interpret these (looking at the subscripts) as *compositions*: $2 = 1+1 = 2$; $3 = 1+1+1 = 1+2 = 2+1 = 3$; $4 = 1+1+1+1 = 2+1+1 = 1+2+1 = 1+1+2 = 1+3 = 3+1 = 2+2 = 4$. The number of compositions of n is 2^{n-1}, which we get if all $t_j = 1$.

From the Wikipedia entry on composition (combinatorics), "a composition of an integer n is a way of writing n as the sum of a sequence of strictly positive integers."

One may interpret the recurrence relation

$$p_{n,0} = \sum_{k=1}^{n} t_k p_{n-k,0} \tag{6.122}$$

from Proposition 6.6.3 as saying that to generate a composition of n, you get the composition of $n-k$ and then add the number "k" to them; adding these together gives all compositions. For example, when $n = 5$ we have $p_{0,0} = 1$, $p_{1,0} = t_1$, $p_{2,0} = t_1^2 + t_2$, $p_{3,0} = t_1^3 + 2t_1t_2 + t_3$, and $p_{4,0} = t_1^4 + 3t_1^2t_2 + 2t_1t_3 + t_2^2 + t_4$. Then

$$\begin{aligned} p_{5,0} &= t_1p_{4,0} + t_2p_{3,0} + t_3p_{2,0} + t_4p_{1,0} + t_5p_{0,0} \\ &= t_1^5 + 3t_1^3t_2 + 2t_1^2t_3 + t_1t_2^2 + t_1t_4 + t_2t_1^3 + 2t_1t_2^2 + t_2t_3 + t_1^2t_3 + t_2t_3 + t_4t_1 + t_5 \\ &= t_1^5 + 4t_1^3t_2 + 3t_1^2t_3 + 3t_1t_2^2 + 2t_1t_4 + 2t_2t_3 + t_5\,. \end{aligned}$$

Remark 6.9.1. This determinant also contains the whole characteristic polynomial. Simply replace t, with $t_1 - z$ and we get $\det(\mathbf{M}_n - z\mathbf{I}) = (-1)^n P_n$. This suggests that "compositions with all parts bigger than 1" can be used to generate all compositions. This fact is well-known. The combinatorial analysis of this recurrence formula is not quite trivial.

6.10 Zero Diagonal Upper Hessenberg Matrices

Theorem 6.10.1. *Let $\mathbf{A}_n \in \mathcal{Z}_{\{0\}}^{n\times n}(P)$ for $P = \{0, w_1, \ldots, w_m\}$ for some fixed positive integer m and each $|w_j| = 1$. If $\mathbf{A}_n$ is normal, i.e. $\mathbf{A}_n^*\mathbf{A}_n = \mathbf{A}_n\mathbf{A}_n^*$, then for $n \geq 3$, $\mathbf{A}_n$ is symmetric, w_j-skew symmetric for some fixed $1 \leq j \leq m$ or w_j-skew circulant. These $2m$ matrices (m symmetric/w_j-skew symmetric, and m w_j-skew circulant matrices) are the only normal matrices in $\mathcal{Z}_{\{0\}}^{n\times n}(P)$. (For $n = 1$, this is only $[0]$; for $n = 2$, the symmetric and circulant cases coalesce, so that there are only m such matrices.)*

Proof. To prove this theorem, we establish a sequence of lemmas. First, we partition $\mathbf{A}_n$.

Put

$$\mathbf{A}_n = \begin{bmatrix} 0 & \mathbf{T}^* \\ \mathbf{e} & \mathbf{A}_{n-1} \end{bmatrix} \tag{6.123}$$

where

$$\mathbf{e}^* = \begin{bmatrix} 1 & 0 & \cdots & 0 \end{bmatrix} \tag{6.124}$$

and

$$\mathbf{T}^* = \begin{bmatrix} t_{12} & t_{13} & \cdots & t_{1n} \end{bmatrix} = \begin{bmatrix} t_{21}^* & t_{31}^* \cdots t_{n1}^* \end{bmatrix}^* . \tag{6.125}$$

Then the conditions of normality are

$$\mathbf{A}_n\mathbf{A}_n^* = \begin{bmatrix} \mathbf{T}^*\mathbf{T} & \mathbf{T}^*\mathbf{A}_{n-1}^* \\ \mathbf{A}_{n-1} & \mathbf{e}\mathbf{e}^* + \mathbf{A}_{n-1}\mathbf{A}_{n-1}^* \end{bmatrix} \tag{6.126}$$

must equal

$$\mathbf{A}_n^*\mathbf{A}_n = \begin{bmatrix} 1 & \mathbf{e}^*\mathbf{A}_{n-1} \\ \mathbf{A}_{n-1}^*\mathbf{e} & \mathbf{T}\mathbf{T}^* + \mathbf{A}_{n-1}^*\mathbf{A}_{n-1} \end{bmatrix} . \tag{6.127}$$

♮

Lemma 6.10.2. *The first row of $\mathbf{A}_n$ contains exactly one nonzero element, say τ in position j $(2 \le j \le n)$.*

Proof.

$$\mathbf{T}^*\mathbf{T} = \sum_{j=2}^{n} |t_{ij}|^2 = 1 \tag{6.128}$$

from the upper left corner. Since each nonzero element of P has magnitude 1, exactly one entry must be nonzero. ♮

Lemma 6.10.3. *If $\mathbf{A}_{n-1}$ is normal then $\mathbf{T} = \tau\mathbf{e}$ and $\mathbf{A}_n$ is τ-skew symmetric.*

Proof. If $\mathbf{A}_{n-1}$ is normal, then $\mathbf{T}\mathbf{T}^* + \mathbf{A}_{n-1}^*\mathbf{A}_{n-1}$ being equal to $\mathbf{e}\mathbf{e}^* + \mathbf{A}_{n-1}\mathbf{A}_{n-1}^*$ implies $\mathbf{T}\mathbf{T}^* = \mathbf{e}\mathbf{e}^*$ so that $\mathbf{T}^* = \begin{bmatrix} \tau^* & 0 & \cdots & 0 \end{bmatrix}$ for some τ with $|\tau| = 1$. Then

$$\mathbf{T}^*\mathbf{A}_{n-1}^* = \mathbf{e}^*\mathbf{A}_{n-1} \Rightarrow \tau^* \begin{bmatrix} 1 & 0 & \cdots & 0 \end{bmatrix} \mathbf{A}_{n-1}^* = \mathbf{e}^*\mathbf{A}_{n-1}^* \tag{6.129}$$

and this says τ^* times the first row of $\mathbf{A}_{n-1}^*$ is the first row of $\mathbf{A}_{n-1}$.

But the first row of $\mathbf{A}_{n-1}^*$ is $\begin{bmatrix} 0 & 1 & 0 & \cdots & 0 \end{bmatrix}$ because $\mathbf{A}_{n-1}$ is upper Hessenberg with zero diagonal. Thus the first row of $\mathbf{A}_{n-1}$ is $\begin{bmatrix}0 & \tau^* & 0 & \cdots & 0\end{bmatrix}$. Thus

$$\mathbf{A}_n = \left[\begin{array}{c|c|c} 0 & \tau^* & \\ \hline 1 & 0 & \tau^* \\ \hline & 1 & \\ & & \mathbf{A}_{n-2} \\ & & \end{array}\right] \quad \text{(remember } n \geq 3) \tag{6.130}$$

and

$$\mathbf{A}_{n-1} = \begin{bmatrix} 0 & \tau^* & \\ 1 & & \\ & \mathbf{A}_{n-2} & \end{bmatrix} \tag{6.131}$$

is normal. Because $\mathbf{A}_{n-1}$ is normal, and

$$\mathbf{A}_{n-1}^* = \begin{bmatrix} 0 & 1 & & \\ \tau & 0 & 1 & \\ & \tau & & \\ & & \mathbf{A}_{n-2}^* & \\ & & & \end{bmatrix} \tag{6.132}$$

we have $\mathbf{A}_{n-1}^*\mathbf{A}_{n-1} = \mathbf{A}_{n-1}\mathbf{A}_{n-1}^*$ or

$$\begin{bmatrix} 0 & 1 & & \\ \tau & 0 & 1 & \\ & 1 & & \\ & & \mathbf{A}_{n-2} & \\ & & & \end{bmatrix} \begin{bmatrix} 0 & \tau^* & & \\ 1 & 0 & \tau^* & \\ & 1 & & \\ & & \mathbf{A}_{n-2} & \\ & & & \end{bmatrix}$$

$$= \begin{bmatrix} 1 & 0 & \tau^* \\ 0 & 2 & \mathbf{e}_{n-2}^*\mathbf{A}_{n-2} \\ \tau & & \\ & \tau\mathbf{A}_{n-2}^+\mathbf{e}_{n-2} & \mathbf{e}\mathbf{e}^* + \mathbf{A}_{n-2}^*\mathbf{A}_{n-2} \\ & & \end{bmatrix}$$

must equal

$$\begin{bmatrix} 0 & \tau^* & & \\ 1 & 0 & \tau^* & \\ & 1 & & \\ & & \mathbf{A}_{n-2} & \\ & & & \end{bmatrix} \begin{bmatrix} 0 & 1 & & \\ \tau & 0 & 1 & \\ & 1 & & \\ & & \mathbf{A}_{n-2} & \\ & & & \end{bmatrix}$$

$$= \begin{bmatrix} 1 & 0 & \tau^* \\ 0 & 2 & \mathbf{e}_{n-2}^* \mathbf{A}_{n-2} \\ \tau & \tau \mathbf{A}_{n-2}^+ \mathbf{e}_{n-2} & \mathbf{e}\mathbf{e}^* + \mathbf{A}_{n-2}^* \mathbf{A}_{n-2} \end{bmatrix} .$$

The lower left block gives $\mathbf{e}\mathbf{e}^* + \mathbf{A}_{n-2}\mathbf{A}_{n-2}^* = \mathbf{e}\mathbf{e}^* + \mathbf{A}_{n-2}^*\mathbf{A}_{n-2}$ so $\mathbf{A}_{n-2}$ must also be normal.

At this point, we see the outline of an induction:

$$\mathbf{A}_n = \left[\begin{array}{c|c} 0 & \tau^* \\ \hline 1 & \mathbf{A}_{n-1} \end{array}\right] \tag{6.133}$$

being normal with $\mathbf{A}_{n-1}$ being normal implies that

$$\mathbf{A}_{n-1} = \left[\begin{array}{c|c} 0 & \tau^* \\ \hline 1 & \mathbf{A}_{n-2} \end{array}\right] \tag{6.134}$$

where $\mathbf{A}_{n-2}$ is normal. Explicit computation of the $n = 3$ case shows the induction terminates. ♮

We now consider the harder case where

$$\mathbf{A}_n = \begin{bmatrix} 0 & \mathbf{T}^* \\ \mathbf{e}_{n-1} & \mathbf{A}_{n-1} \end{bmatrix} \tag{6.135}$$

but where $\mathbf{A}_{n-1}$ is not itself normal. From Lemma 6.10.2 we know that $\mathbf{T}^*$ has only one nonzero element; call it τ^* as before. Then

$$\mathbf{T}\mathbf{T}^* = \begin{bmatrix} 0 & & & & & & \\ & \ddots & & & & & \\ & & 0 & & & & \\ & & & 1 & & & \\ & & & & 0 & & \\ & & & & & \ddots & \\ & & & & & & 0 \end{bmatrix} \tag{6.136}$$

while

$$\mathbf{e}\mathbf{e}^* = \begin{bmatrix} 1 & & & \\ & 0 & & \\ & & \ddots & \\ & & & 0 \end{bmatrix}, \tag{6.137}$$

and we may assume that the 1 in $\mathbf{T}\mathbf{T}^*$ does not occur in the first row and column (else we are in the previous case, and $\mathbf{A}_{n-1}$ will be normal). Here

$$\mathbf{A}_{n-1}\mathbf{A}_{n-1}^* - \mathbf{A}_{n-1}^*\mathbf{A}_{n-1} = \mathbf{T}\mathbf{T}^* - \mathbf{e}\mathbf{e}^* = \begin{bmatrix} -1 & & & & & & & \\ & 0 & & & & & & \\ & & \ddots & & & & & \\ & & & 0 & & & & \\ & & & & 1 & & & \\ & & & & & 0 & & \\ & & & & & & \ddots & \\ & & & & & & & 0 \end{bmatrix} \tag{6.138}$$

is the departure of $\mathbf{A}_{n-1}$ from normality. We will establish that in fact

$$\mathbf{T}^* = \begin{bmatrix} 0 & 0 & 0 & \cdots & 0 & \tau^* \end{bmatrix} \tag{6.139}$$

and that

$$\mathbf{A}_{n-1} = \begin{bmatrix} 0 & & & & \\ 1 & 0 & & & \\ & 1 & 0 & & \\ & & \ddots & \ddots & \\ & & & 1 & 0 \end{bmatrix}; \tag{6.140}$$

that is, the nonzero element can only occur in the last place. Notice that the upper left corner of 6.138 is, if the top row of $\mathbf{A}_{n-1}$ is $\begin{bmatrix} 0 & a_{1,2} & a_{1,3} & \cdots a_{1,n-1} \end{bmatrix}$,

$$\sum_{j=2}^{n-1} |a_{1,j}|^2 - 1\,. \tag{6.141}$$

Therefore, all $a_{1,j} = 0$ and the first row of $\mathbf{A}_{n-1}$ must be zero: i.e.

$$\mathbf{A}_{n-1} = \begin{bmatrix} 0 & 0 & 0 & \cdots & 0 \\ 1 & 0 & a_{3,3} & \cdots & a_{2,n-1} \\ & 1 & 0 & \ddots & \vdots \\ & & \ddots & \ddots & a_{n-2,n-1} \\ & & & 1 & 0 \end{bmatrix} \tag{6.142}$$

Then,

$$\mathbf{A}_{n-1}\mathbf{T} = \mathbf{A}^*_{n-1}\mathbf{e} = \begin{bmatrix} 0 & 1 & & \\ 0 & 0 & \ddots & \\ \vdots & & \ddots & 1 \\ 0 & \cdots & \cdots & 0 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix} . \tag{6.143}$$

If

$$\mathbf{T} = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ \tau \\ 0 \\ \vdots \\ 0 \end{bmatrix} , \tag{6.144}$$

then

$$\mathbf{A}_{n-1}\mathbf{T} = \begin{bmatrix} 0 \\ \tau a_{2,j} \\ \vdots \\ \tau a_{j-1,j} \\ 0 \\ \tau \\ 0 \\ \vdots \\ 0 \end{bmatrix} , \tag{6.145}$$

which is impossible unless $j = n$ (when the τ term is not present). Therefore,

$$\mathbf{A}_{n-1} = \begin{bmatrix} 0 & 0 & \cdots & 0 & 0 \\ 1 & x & \cdots & x & 0 \\ & 1 & \ddots & \vdots & \vdots \\ & & \ddots & x & 0 \\ & & & 1 & 0 \end{bmatrix} = \begin{bmatrix} 0 & 0 \\ \mathbf{U} & 0 \end{bmatrix}, \tag{6.146}$$

and

$$\mathbf{A}_{n-1}\mathbf{A}_{n-1}^{*} - \mathbf{A}_{n-1}^{*}\mathbf{A}_{n-1} = \begin{bmatrix} -1 & & & & \\ & 0 & & & \\ & & \ddots & & \\ & & & 0 & \\ & & & & 1 \end{bmatrix}. \tag{6.147}$$

Since

$$\mathbf{A}_{n-1}^{*} = \begin{bmatrix} 0 & \mathbf{U}^{*} \\ 0 & 0 \end{bmatrix} \tag{6.148}$$

and

$$\mathbf{A}_{n-1}\mathbf{A}_{n-1}^{*} = \begin{bmatrix} 0 & 0 \\ 0 & \mathbf{U}\mathbf{U}^{*} \end{bmatrix} \tag{6.149}$$

and

$$\mathbf{A}_{n-1}^{*}\mathbf{A}_{n-1} = \begin{bmatrix} \mathbf{U}^{*}\mathbf{U} & 0 \\ 0 & 0 \end{bmatrix}, \tag{6.150}$$

$$\begin{bmatrix} 0 & 0 \\ 0 & \mathbf{U}\mathbf{U}^{*} \end{bmatrix} - \begin{bmatrix} \mathbf{U}^{*}\mathbf{U} & 0 \\ 0 & 0 \end{bmatrix} \tag{6.151}$$

must be diagonal. Therefore, the first row of $\mathbf{U}\mathbf{U}^{*}$ must be zero except for the first element.

Remark 6.10.4. For $n = 4$, and $P = \{0, i, -i\}$ ($m = 2$) the following 4 matrices are normal:

w_j	w_j-skew symmetric	w_j-skew circulant
i	$\begin{bmatrix} 0 & i & 0 & 0 \\ 1 & 0 & i & 0 \\ & 1 & 0 & i \\ & & 1 & 0 \end{bmatrix}$	$\begin{bmatrix} 0 & 0 & 0 & i \\ 1 & 0 & 0 & 0 \\ & 1 & 0 & 0 \\ & & 1 & 0 \end{bmatrix}$
$-i$	$\begin{bmatrix} 0 & -i & 0 & 0 \\ 1 & 0 & -i & 0 \\ & 1 & 0 & -i \\ & & 1 & 0 \end{bmatrix}$	$\begin{bmatrix} 0 & 0 & 0 & -i \\ 1 & 0 & 0 & 0 \\ & 1 & 0 & 0 \\ & & 1 & 0 \end{bmatrix}$

6.11 Stable Matrices

6.11.1 Type I Stable Matrices

A *Type I stable matrix* $\mathbf{A}$ is a matrix with all of its eigenvalues strictly in the left half plane: if λ is an eigenvalue of A then $\Re(\lambda) < 0$. This nomenclature comes from differential equations, in that all solutions of the linear system of ODEs $dy/dt = \mathbf{A}y$ will ultimately decay as $t \to \infty$ if $\mathbf{A}$ is a type I stable matrix.

If the matrix $\mathbf{A}$ is not *normal*, then *pseudospectra* can play a role, in that even though all solutions y must ultimately decay, they might first grow large. See [19] for details.

By Theorem 6.10.1, only $2m$ of the zero diagonal upper Hessenberg matrices with population $P = \{-1, 0, +1\}$ are normal, where here $m = 2$. Similarly, when the population is $P = \{0, +1\}$ then $m = 1$ and only two matrices of every dimension are normal (the symmetric matrix with 1s on its upper diagonal, and the circulant matrix with a 1 in the last column of the first row).

Theorem 6.11.1. *No $\mathbf{A}_n \in \mathcal{Z}^{n\times n}_{\{\theta_k\}}(P)$ is Type I stable, for any population P.*

Proof. Suppose $\mathbf{A}_n \in \mathcal{Z}^{n\times n}_{\{\theta_k\}}(P)$ has eigenvalues $\{\lambda_k\}_{k=1}^{n}$. Then

$$\sum_{k=1}^{n} \lambda_k = \mathrm{Trace}(\mathbf{A}_n) = 0\,. \tag{6.152}$$

Therefore, $\sum_{k=1}^{n} \mathrm{Re}(\lambda_k) = 0$. This is n times the average, and so the average is zero. Since the maximum $\mathrm{Re}(\lambda_k)$ must be larger than the average, this proves the theorem. ♮

The proof of this theorem did not depend on the structure or population. Thus if we consider $\mathcal{H}_{\{0\}}^{n\times n}(P)$ instead of $\mathcal{Z}_{\{0\}}^{n\times n}(P)$, then we may simplify our search for stable matrices by restricting the computation to those with negative trace. This is in fact the first inequality of the Hurwitz criteria[10], which leads to an effective and efficient method to count stable matrices: start from the database of characteristic polynomials [26], decide using the Hurwitz criteria if all roots are in the left half-plane, and if so add its matrices to the count.

n	$\mathcal{H}_{\{0\}}^{n\times n}(\{-1,0,+1\})$	$\mathcal{H}_{\{0\}}^{n\times n}(\{-1,+1\})$
2	4	1
3	44	4
4	1,386	28
5	130,735	424
6	35,217,156	11,613
7		617,619

Table 6.9: The numbers of Type I stable matrices for various populations and dimensions.

Remark 6.11.2. For stable matrices in $\mathcal{H}_{\{0\}}^{n\times n}(\{-1,0,+1\})$ the maximum real part of any eigenvalue is, for $n=2$, just -0.5 while for $n=3$ it is $-1.226\cdot 10^{-1}$. For $n=4$ it is $-1.591\cdot 10^{-2}$. For $n=5$ it is $-5.176\cdot 10^{-4}$. For $n=6$ it is $-2.42\cdot 10^{-5}$. The maximum real part of the eigenvalues seems to be approaching the real axis at least exponentially in n, for this population. It would be nice to have a good asymptotic estimate.

The sequence of maximum real parts of eigenvalues for $\mathcal{H}_{\{0\}}^{n\times n}(\{-1,+1\})$ gives at $n=2$ $\Re(\lambda)=-1$, -0.5, $-2.168\cdot 10^{-2}$, $-2.66\cdot 10^{-3}$, $-1.70\cdot 10^{-4}$, and $-2.62\cdot 10^{-6}$ for $n=7$.

6.11.2 Type II Stable matrices

A *Type II Stable Matrix* $\mathbf{A}$ has all its eigenvalues inside the unit circle. This class of matrices arises naturally on studying the simple linear recurrence relation $y_{n+1}=\mathbf{A}y_n$. Fairly obviously, all solutions of this difference equation will ultimately decay to 0 as $n\to\infty$ if and only if all eigenvalues of $\mathbf{A}$ are inside the unit circle (again, pseudospectra can play a role in the transient behaviour, sometimes significantly).

Theorem 6.11.3. *If* $\mathbf{A}$ *is a Bohemian matrix with integer population* P*, then it is Type II stable if and only if it is nilpotent, in which case all its eigenvalues are* 0.

[10]The Maple command `PolynomialTools[Hurwitz]` implements a well-known test to decide if $p\in\mathbb{C}[z]$ has all its roots strictly in the left half plane. Because that routine considers the complex case, and tests for pathological cases, it is too inefficient to use in this context. We unrolled the loops, essentially converting the code to specific tests of the principal minors of the Hurwitz matrix.

Proof. Suppose to the contrary that some eigenvalues are not zero.

The determinant of **A** must necessarily be an integer. If the integer is not zero, it is at least 1 in magnitude. The product of the eigenvalues is thus at least 1 in magnitude; hence there must be at least one eigenvalue that is at least 1 in magnitude.

If the matrix **A** has zero determinant but not all eigenvalues zero, then after factoring out z^m for the multiplicity of the zero eigenvalue, the product of the other eigenvalues becomes the constant coefficient (what was the coefficient of z^m in the original). This coefficient again must be an integer, and again at least one eigenvalue must be at least 1 in magnitude.

This proves the theorem, by contradiction. ♮

Remark 6.11.4. We did not, in fact, use that the matrix came from a Bohemian family; only that its entries were integers.

Searching for nilpotent matrices in various classes of Bohemian matrices turns up several puzzles. We give some preliminary results here in Table 6.10, but leave this mostly to future work. For instance, it seems clear from our experiments that the only nilpotent matrix in $\mathcal{H}_{\{0\}}^{n\times n}(\{0,+1\})$ is the (transpose of the) complete Jordan block of n zero eigenvalues; contrariwise the irregular behaviour for $\mathcal{H}_{\{0\}}^{n\times n}(\{-1,+1\})$ is very puzzling.

n	$\mathcal{Z}_{\{0\}}^{n\times n}(\{-1,0,+1\})$	$\mathcal{H}_{\{0\}}^{n\times n}(\{0,+1\})$	$\mathcal{H}_{\{0\}}^{n\times n}(\{-1,+1\})$
2	1	1	2
3	3	1	0
4	21	1	0
5	271	1	0
6	9,075	1	324

Table 6.10: The numbers of nilpotent matrices for various populations and dimensions

Considering *general* Bohemian matrices with population $\{-1,0,+1\}$, so that there are 3^{n^2} such matrices, we find that there are 1, 9, 481, 148,817, and 243,782,721 nilpotent matrices at dimensions 1 through 5 inclusive. We can fit this experimentally with the formula $\exp(0.5+0.38n+0.23n^2)$, or something like 1.26^{n^2}, which vanishes very quickly compared to 3^{n^2}. This formula predicts that for $n=6$ the probability of finding a nilpotent matrix is about 2.75×10^{-14}. It would be gratifying to have a better understanding of the number of nilpotent matrices in a family.

6.12 Concluding Remarks

The class of upper Hessenberg Bohemian matrices gives a useful way to study Bohemian matrices in general. This is an instance of Polya's adage "find a useful specialization." [23, p. 190] Because these classes are simpler than the general case, we were able to establish several theorems. Note that the three families $\mathcal{H}_{\{0\}}^{n\times n}(\{0,+1\})$, $\mathcal{H}_{\{0\}}^{n\times n}(\{-1,+1\})$, and $\mathcal{Z}_{\{0\}}^{n\times n}(\{-1,0,+1\})$ are all subfamilies of $\mathcal{H}_{\{0\}}^{n\times n}(\{-1,0,+1\})$.

In this paper we have introduced two new formulae for computing the characteristic polynomials of upper Hessenberg matrices. Our first formula, given in Theorem 6.5.1, also computes the characteristic polynomials recursively. Our second formula, given in Theorem 6.5.2, computes the coefficients recursively.

We extended the formulae for the characteristic polynomials to upper Hessenberg Toeplitz matrices in Proposition 6.6.6 and Proposition 6.6.8. In Proposition 6.7.1 we show that the maximal characteristic height of upper Hessenberg matrices in $\mathcal{H}_{\{0\}}^{n\times n}(\{-1,0,+1\})$ is at least 2^{n-1}. In Theorem 6.7.6 we show that the number of upper Hessenberg Toeplitz matrices of maximal height in $\mathcal{M}_{\{0\}}^{n\times n}(\{-1,0,+1\})$ is $3\cdot 2^{\mu_n}$ where μ_n is the degree of the coefficient of the characteristic polynomial whose coefficient, in absolute value, is the height.

We also explored some properties of zero diagonal Bohemian upper Hessenberg matrices. In Theorem 6.10.1, we show that the subset of these matrices that are normal are always symmetric, w_j-skew symmetric for some fixed $1\leq j\leq m$, or w_j-skew circulant. In Theorem 6.11.1, we showed that no $\mathbf{H}\in\mathcal{Z}_{\{\theta_k\}}^{n\times n}(P)$ is stable.

Many puzzles remain. Perhaps the most striking is the angular appearance of the set $\boldsymbol{\Lambda}(\mathcal{H}_{\{0\}}^{n\times n}(P))$ of eigenvalues of $\mathcal{H}_{\{0\}}^{n\times n}(P)$, such as in Figures 6.1 and 6.3. General matrices have eigenvalues asymptotic to a (scaled) disc [24]; our computations suggest that as $n\to\infty$, $\boldsymbol{\Lambda}(\mathcal{H}_{\{0\}}^{n\times n}(P))/n^{1/2}$ tends to an irregular hexagonal shape, rather than a disk. More, the density does not appear to be approaching uniformity. Further, the boundary is irregular, with shapes suggestive of what is popularly known as the "dragon curve" (in reverse—these delineate where the eigenvalues are absent, near the edge). We have no explanation for this.

Bibliography

[1] M. Abbas and S. Bouroubi. On new identities for Bell's polynomials. *Discrete Mathematics*, 293(1-3):5–10, 2005.

[2] J. Baez. The beauty of roots. *Available at: https://johncarlosbaez.wordpress.com/2011/12/11/the-beauty-of-roots/*, 2011.

[3] F. Beaucoup, P. Borwein, D. W. Boyd, and C. Pinner. Multiple roots of $[-1, 1]$ power series. *Journal of the London Mathematical Society*, 57(1):135–147, 1998.

[4] C. Beltrán and D. Armentano. The polynomial eigenvalue problem is well conditioned for random inputs. *arXiv preprint arXiv:1706.06025*, 2017.

[5] J. M. Borwein and P. B. Borwein. Strange series and high precision fraud. *The American Mathematical Monthly*, 99(7):622–640, 1992.

[6] P. Borwein. *Computational excursions in analysis and number theory*. Springer Science & Business Media, 2012.

[7] P. Borwein and L. Jörgenson. Visible structures in number theory. *The American Mathematical Monthly*, 108(10):897–910, 2001.

[8] P. Borwein and C. Pinner. Polynomials with $\{0, +1, -1\}$ coefficients and a root close to a given point. *Canadian Journal of Mathematics*, 49(5):887–915, 1997.

[9] E. Y. S. Chan. A comparison of solution methods for Mandelbrot-like polynomials. *Electronic Thesis and Dissertation Repository*, 2016. `https://ir.lib.uwo.ca/etd/4028`.

[10] E. Y. S. Chan and R. M. Corless. A new kind of companion matrix. *Electronic Journal of Linear Algebra*, 32:335–342, 2017.

[11] E. Y. S. Chan and R. M. Corless. Minimal height companion matrices for Euclid polynomials. *Mathematics in Computer Science*, Jul 2018.

[12] E. Y. S. Chan, R. M. Corless, L. Gonzalez-Vega, J. R. Sendra, and J. Sendra. Algebraic linearizations of matrix polynomials. *Linear Algebra and its Applications*, 563:373–399, 2019.

[13] R. M. Corless. Generalized companion matrices in the Lagrange basis. In *Proceedings of Encuentro de Algebra Computacional y Aplicaciones*, pages 317–322. Santander, Spain: Universidad de Cantabria, 2004.

[14] R. M. Corless and P. W. Lawrence. Mandelbrot polynomials and matrices. In preparation.

[15] R. M. Corless and P. W. Lawrence. The largest roots of the Mandelbrot polynomials. In *Computational and Analytical Mathematics*, pages 305–324. Springer, 2013.

[16] R. M. Corless and S. E. Thornton. The Bohemian eigenvalue project. *ACM Communications in Computer Algebra*, 50(4):158–160, 2016.

[17] R. M. Corless and S. E. Thornton. Visualizing eigenvalues of random matrices. *ACM Communications in Computer Algebra*, 50(1):35–39, apr 2016.

[18] M. Elouafi and A. D. A. Hadj. A recursion formula for the characteristic polynomial of Hessenberg matrices. *Applied Mathematics and Computation*, 208(1):177–179, 2009.

[19] M. Embree. Pseudospectra. In L. Hogben, editor, *Handbook of Linear Algebra*, chapter 23. Chapman and Hall/CRC, 2013.

[20] C. W. Gear. A simple set of test matrices for eigenvalue programs. *Mathematics of Computation*, 23(105):119–125, 1969.

[21] N. J. Higham. Bohemian matrices in numerical linear algebra. Available at `http://www.maths.manchester.ac.uk/~higham/conferences/bohemian/higham_bohemian18.pdf`(June 20, 2018).

[22] K. Kaygısız and A. Sahin. Determinant and permanent of Hessenberg matrix and Fibonacci type numbers. *Gen*, 9(2):32–41, 2012.

[23] G. Polya. *How to solve it: A new aspect of mathematical method*. Princeton University Press, 2014.

[24] T. Tao and V. Vu. Random matrices have simple spectrum. *Combinatorica*, 37(3):539–553, 2017.

[25] O. Taussky. Matrices of rational integers. *Bulletin of the American Mathematical Society*, 66(5):327–345, 1960.

[26] S. E. Thornton. The characteristic polynomial database. Available at `http://bohemianmatrices.com/cpdb` (Sept. 7, 2018).

Chapter 7

Concluding Remarks

7.1 Parametric Matrices

Motivated by the goal of computing the Jordan canonical form of matrices depending on parameters, several algorithms have been developed that work in conjunction to achieve this goal. Focusing on matrices where the entries are multivariate polynomials whose indeterminates are regarded as parameters, we are able to use the theory of regular chains to develop algorithms for parametric matrices. Further, the theory of regular chains provides a uniform framework for working over systems with both algebraic and semi-algebraic constraints. The `RegularChains`[1] library in the MAPLE computer algebra system has allowed for implementations of the presented algorithms. A MAPLE package called `ParametricMatrixTools` contains implementations of these algorithms and is available at `https://github.com/steventhornton/ParametricMatrixTools`. These algorithms have proven effective in problems from control theory and biology where parameters represent underlying physical properties of the systems.

In Chapter 2, an algorithm for computing the rank of a parametric matrix as a function of its parameters was presented. The algorithm avoids the use of general tools for solving parametric linear systems. Experiments comparing this algorithm to a naïve one illustrate its effectiveness. By using the theory of regular chains, the algorithm was presented for both the case of real-value parameters and complex valued. The algorithm was applied to several problems from the literature and provided meaningful insight into the systems being studied.

Chapter 3 introduced the Zigzag form and its use for computing the Frobenius form. An algorithm for computing the Zigzag form was adapted from [1] to work with matrices

[1]See `http://regularchains.org/` for details.

where the entries are multivariate polynomials in the parameters.

Chapter 4 presented an algorithm for computing the Jordan canonical form of a matrix in Frobenius form. The input matrix is in Frobenius form where the entries are multivariate polynomials in the parameters. A complete case discussion for the Jordan form was computed and allows for input algebraic or semi-algebraic constraints on the parameters.

Work on parametric matrices still remains including many improvements to the algorithms presented. The algorithm based on the Zigzag form for computing the Frobenius form of a matrix can lead to large expression growth and unnecessary splitting. Including heuristics in the implementation to help select pivots that delay splitting and control expression growth may allow the algorithm to be applied to larger problems. Alternatively, computing the Smith form of $\mathbf{A} - x\mathbf{I}$ may also provide an improvement because entries can be treated as univariate polynomials in x with coefficients that are multivariate polynomials in the parameters. For both the computation of the Frobenius form and the computation of the Jordan form, research remains to be done to generate algorithms that maintain the similarity transformation matrices.

7.2 Bohemian Matrices

Bohemian matrices have proven to be fascinating objects to study. The `http://www.bohemianmatrices.com` website has been created to archive research related to Bohemian matrices.

Density plots of Bohemian eigenvalues have been a main source of motivation for continued study of many Bohemian families. A MATLAB framework has been developed to make the generation of plots of Bohemian eigenvalues faster and is available at `https://github.com/BohemianMatrices/BHIME-Project`. An extensive gallery of density plots of Bohemian eigenvalues is available at `http://www.bohemianmatrices.com/gallery/`.

The Characteristic Polynomial Database was created to provide a centralized database of the distributions of characteristic polynomials for several interesting Bohemian families. These distributions have led to many new connections between the study of Bohemian matrices and other areas of mathematics. The database is available at `http://www.bohemianmatrices.com/cpdb/` and includes a list of 21 conjectures related to Bohemian matrices and their properties.

In Chapter 5, an extensive introduction to Bohemian matrices and the problems of interest related to them was presented. The MATLAB framework developed for generating plots of Bohemian matrices was presented. Two families of Bohemian matrices were discovered where the `eig` function in MATLAB fails to provide solutions in some instances. The Characteristic Polynomial Database and the motivation for its development was introduced. Properties of interest related to Bohemian families were discussed. Finally, a list of 21 conjectures connecting sequences of properties of Bohemian families with known sequences on the OEIS were given.

Chapter 6 focused on two specialized families of Bohemian matrices: upper Hessenberg, and upper Hessenberg matrices with a Toeplitz structure. These matrices were further specialized to have population $\{-1, 0, +1\}$. Two recursive formulae for the characteristic polynomials of upper Hessenberg matrices were given. By specializing to matrices whose characteristic polynomials are of maximal height we were able to give a bound on their height.

Many questions remain unanswered about Bohemian matrices and new families remain to be explored. The Characteristic Polynomial Database will continue to expand to include many new families including structured matrices such as symmetric, circulant, and tridiagonal. As new families are explored new questions will arise. The list of properties computed on these families will continue to grow and new algorithms will be required as family sizes grow. New conjectures will appear and the list of conjectures will continue to expand.

The visualization of Bohemian eigenvalues has provided a new class of mathematical objects that can be appreciated regardless of someone's mathematical background. A Python package for visualization is under development with the hope of making Bohemian eigenvalues accessible to a wider audience.

The work on Bohemian matrices presented in this thesis serves as a basis from which many new and interesting questions can be tackled.

Bibliography

[1] A. Storjohann. An $\mathcal{O}(n^3)$ algorithm for the Frobenius normal form. In *Proceedings of the 1998 International Symposium on Symbolic and Algebraic Computation*, pages 101–105. ACM, 1998.

Steven E. Thornton

Education	**The University of Western Ontario** Ph.D. Candidate in Applied Mathematics	**2014 – Present**
	The University of Western Ontario M.Sc. Candidate in Applied Mathematics (direct entry to Ph.D.)	**2013 – 2014**
	The University of Western Ontario B.Sc. Honors Applied Mathematics, minor in Physics	**2009 – 2013**
Academic Awards	Ontario Graduate Scholarship	**2016 – 2017**

Publications

Corless, R. M., Moreno Maza, M., & **Thornton, S. E.** (2017, November). "Jordan Canonical Form with Parameters from Frobenius Form with Parameters." In Proceedings of *The International Conference on Mathematical Aspects of Computer and Information Sciences* (pp. 179-194). Springer, Cham.

Corless, R. M., & **Thornton, S. E.** (2014, August). "A package for parametric matrix computations." In In Proceedings of *The International Congress on Mathematical Software* (pp. 442-449). Springer, Berlin, Heidelberg.

Corless, R. M., Moreno Maza, M., & **Thornton, S. E.** (2015). "Zigzag form over families of parametric matrices." *ACM Communications in Computer Algebra*, 48(3/4), 109-112.

Submitted Publications

Thornton, S. E. (2019). "Bohemian Matrices and Their Eigenvalues."

Corless, R. M., Moreno Maza, M., & **Thornton, S. E.** (2018). "Comprehensive Rank Computation for Matrices Depending on Parameters."

Chan, E. Y. S., Corless, R. M., Gonzalez-Vega, L., Sendra, J. R., Sendra, J. & **Thornton, S. E.** (2018). "Bohemian Upper Hessenberg and Toeplitz Matrices."

Conference Presentations

"Computing Bohemian Eigenvalues," presented at Bohemian Matrices and Applications 2018, University of Manchester, Manchester, England.

"Jordan Canonical Form with Parameters from Frobenius Form with Parameters," presented at the International Conference on Mathematical Aspects of Computer and Information Sciences (MACIS) 2017, Vienna, Austria.

"The Bohemian Eigenvalue Project," presented at The Canadian Applied and Industrial Mathematics Society (CAIMS) 2017, Halifax, Canada.

"A Maple Package for Parametric Matrix Computations," presented at the Southern Ontario Numerical Analysis Day (SONAD) 2015, Toronto, Canada.

"A Maple Package for Parametric Matrix Computations," presented at the International Congress on Mathematical Software (ICMS) 2014, Seoul, Korea.

"Zigzag Form over Families of Parametric Matrices," presented at Encuentro de Algebra Computacional y Aplicaciones (EACA) 2015, Barcelona, Spain.

Posters

"The Bohemian Eigenvalue Project," presented at the Fallona Interdisciplinary Research Showcase, Western University, London, Canada, 2017.

"The Weyr Canonical Form," presented at the Ontario Research Centre for Computer Algebra (ORCCA) joint lab meeting, Western University, London, Canada, 2016.

"The Bohemian Eigenvalue Project," presented at the International Symposium on Symbolic and Algebraic Computation (ISSAC), Wilfrid Laurier University, Waterloo, Canada, 2016.

"Visualizing Eigenvalues of Structured Random Matrices," presented at the East Coast Computer Algebra Day (ECCAD), Fields Institute, Toronto, Canada, 2015.

"Rank Computation of Parametric Matrices," presented at the East Coast Computer Algebra Day (ECCAD), Fields Institute, Toronto, Canada, 2015.

"A Maple Package for Parametric matrix Computations," presented at the Western University Research Showcase, Western University, London, Canada, 2014.

"Zigzag Form Over Families of Parametric Matrices," presented at the International Symposium on Symbolic and Algebraic Computation (ISSAC), Kobe University, Kobe, Japan, 2014.

Teaching Experience

The University of Western Ontario **2013 – 2017**
Teaching Assisitant, Department of Applied Mathematics
Numerical Analysis, Introduction to Experimental Mathematics, Calculus, Linear Algebra with Numberical Analysis for Engineering, Advanced Applied Mathematics for Mechanical Engineering, Advanced Applied Mathematics for Electrical Engineering.

The University of Western Ontario **2016**
Introduction to Python for Data Science, Instructor
Ivey Business School

Teaching Awards

Faculty of Science Graduate Student Teaching Assistant Award, 2016.