# Learning via Automaton Minimization and Matrix Factorization


Ines Marušić
Merton College
University of Oxford

A thesis submitted for the degree of
*Doctor of Philosophy*

Trinity 2016

To my parents

# Acknowledgements

I am most grateful to my supervisors, James Worrell, Stefan Kiefer, and Michael Benedikt, for their support and guidance.

I am thankful to my collaborators, Dmitry Chistikov and Mahsa Shirmohammadi, for the many interesting research conversations.

I am grateful to my DPhil viva examiners, Paul Goldberg and Colin de la Higuera, for their helpful comments on this thesis.

I would also like to thank Marta Kwiatkowska for her mentorship throughout my DPhil.

I am thankful to my college advisor, Luke Ong, for his support. I am also grateful to the Graduate Studies Administrator, Julie Sheppard, for her helpful advice.

This work was supported by the Engineering and Physical Sciences Research Council (EPSRC). I also gratefully acknowledge the financial support by Merton College and Google.

My time at Oxford has been greatly enriched by my friends at Merton College and OxWoCS.

Lastly, my gratitude goes to my parents and my sister, Martina, for their unconditional love and support.

# Abstract

This thesis focuses on weighted-automaton learning and minimization, matrix factorization, and the relationships between these areas.

The first part of the thesis studies minimization and learning of weighted automata with weights in a field, focusing on tree automata as representations of distributions over trees. We give a minimization algorithm that runs in polynomial time assuming unit-cost arithmetic, and show that a polynomial bound in the Turing model would require a breakthrough in the complexity of polynomial identity testing. We also improve the complexity of minimizing weighted word automata. Secondly, we look at learning minimal weighted automata in both active and passive learning frameworks, analysing both the computational and query complexity. For active learning, we give a new algorithm for learning weighted tree automata in Angluin's exact learning model that efficiently processes DAG counterexamples, and show a complexity lower bound in terms of polynomial identity testing. For passive learning, we characterise the complexity of finding a minimal weighted automaton that is consistent with a set of observations.

The second part of the thesis studies nonnegative matrix factorization (NMF), a powerful dimension-reduction and feature-extraction technique with numerous applications in machine learning and other scientific disciplines. Despite being an important technique in practice, there are a number of open questions about the theory of NMF—especially about its complexity and the existence of good heuristics. We answer a major open problem posed in 1993 by Cohen and Rothblum: the nonnegative ranks over the rationals and over the reals differ. We first tackle a variant of the problem, restricted NMF, which has applications in topic modelling and a geometric interpretation as the nested polytope problem. Lastly we show that NMF is polynomial-time reducible to automaton minimization, and then use our results on NMF to answer old open problems on minimizing labelled Markov chains and probabilistic automata.

# Contents

# List of Figures

# Chapter 1

# Introduction

Trees are a natural model of hierarchical data and arise rapidly in diverse application domains, including natural text and speech processing, computer vision, bioinformatics, web information extraction, and social network analysis. Many of these application domains require representing probability distributions and more general real-valued functions over large sets of trees. A powerful algebraic model for such functions are *multiplicity tree automata*, i.e., weighted tree automata over a field.

Multiplicity tree automata are nondeterministic finite-state machines that run on trees, assigning a weight from a field $\mathbb{F}$ to each tree. A multiplicity tree automaton thus computes a function from a set of trees into the field $\mathbb{F}$. A special case of multiplicity tree automata are *multiplicity word automata*, which can be viewed as multiplicity tree automata on unary trees.

Two of the most fundamental problems concerning multiplicity automata are *learning* and *minimization*. The general problem of *learning* a concept class is, given examples of an unknown target concept from that class, to infer a hypothesis concept that fits the given examples and accurately predicts future examples. A key component of learning is the *inference* problem: given a sample of weight-labelled trees (resp., words), derive a multiplicity automaton over trees (resp., words) that is

consistent with or closely fits that sample. Here *consistency* means that the weight assigned by the automaton to each tree (resp., word) in the sample is equal to its label. The problem of *minimizing* multiplicity automata is to compute, for a given multiplicity automaton, a smallest automaton computing the same function. In the latter case, we say that the two automata are *equivalent*.

When learning automata, we favour simple and compact hypotheses. Specifically, given two multiplicity automata that are both consistent with a given sample, for most theoretical and practical purposes we would prefer to learn the smaller one; this principle is referred to as *Occam's razor*. Indeed, many existing automaton-learning algorithms [e.g., Angluin, 1987, Beimel et al., 2000] learn a *minimal* automaton consistent with the given sample.

In this thesis we focus on the *efficient exact learning* framework, where the aim is to identify a target concept *exactly* (i.e., without any approximation) from a finite set of examples in time polynomial in the size of the representation of the concept and the maximal size of the representation of an example. Here, unlike the PAC-learning framework, the examples are not assumed to be drawn according to some distribution. A concept class is said to be *efficiently exactly learnable* if there is an algorithm for efficient exact learning of any target concept from that class.

We consider two different efficient exact learning frameworks for learning automata: a *passive* and an *active* efficient exact learning framework. In a *passive learning* framework, the learning algorithm *passively* receives a finite set of observations, i.e., a finite set of weight-labelled trees or words, and returns a weighted automaton that is consistent with those observations. In an *active learning* framework, the learning algorithm does not just passively receive data but also *actively* participates in the selection of examples by asking queries. In this thesis, we give results on both passive and active efficient exact learning of multiplicity automata.

Within active learning, a central framework is the *exact learning model* of Angluin

[1987], also called the *Minimally Adequate Teacher (MAT)* model in the literature. In Angluin's exact learning model there is an oracle that can answer membership and equivalence queries posed by the learner, i.e., the learning algorithm. In asking a *membership query*, the learner asks for the weight of a particular input tree or word. In asking an *equivalence query*, the learner proposes a candidate automaton, to which the oracle answers YES if the candidate is equivalent to the target concept, and otherwise answers NO and provides a *counterexample*: a tree that has a different label under the target concept and the candidate automaton. Thus when analysing the performance of an exact learning algorithm, one needs to consider not just its computational complexity, but also its *query complexity*, i.e., the number of queries required to learn the target concept.

We study the query and computational complexity of learning multiplicity tree automata in the exact learning model. A central computational question from the point of view of the oracle is testing the equivalence of two multiplicity tree automata: the target concept and the candidate. Our first contribution is to characterise the complexity of the equivalence problem for multiplicity tree automata, showing that it is logspace equivalent to polynomial identity testing.

We then give a new exact learning algorithm for multiplicity tree automata whose key feature is that counterexamples to equivalence queries are represented as directed acyclic graphs (DAGs). Upon receiving a counterexample tree succinctly represented as a DAG, our learning algorithm is able to process it efficiently—taking full advantage of the succinctness of the representation—and use it to update the candidate automaton. The query complexity of our algorithm is quadratic in the target automaton size and linear in the size of a largest counterexample—improving the best previously-known algorithm [Habrard and Oncina, 2006] by an exponential factor. We complement this query-complexity upper bound by deriving lower bounds on the number of queries needed to learn multiplicity tree automata over both fixed

and arbitrary fields. In the latter case, the bound is linear in the size of the target automaton—almost matching our upper bound.

The second topic of this thesis is minimization of multiplicity automata, a fundamental problem closely related to both learning and equivalence testing. Minimization of multiplicity automata has numerous applications including image compression [Albert and Kari, 2009] and reducing the space complexity of speech recognition tasks [Mohri et al., 1996, Eisner, 2003]. Our focus is on the problem of minimization with respect to the number of states. That is, given a multiplicity automaton, we wish to compute an equivalent automaton with fewest states; such an automaton is called *minimal*. Minimal multiplicity automata are known to be unique up to change of basis [Bozapalidis and Alexandrakis, 1989].

We give a minimization algorithm for multiplicity tree automata over the field of rational numbers that runs in polynomial time assuming unit-cost arithmetic, and also show that a polynomial bound in the standard Turing model would require a breakthrough in the complexity of polynomial identity testing by proving that the latter problem is logspace equivalent to the decision version of minimization. The developed techniques also improve the state of the art in multiplicity word automata: we give an NC algorithm for minimizing multiplicity word automata, improving upon a previous randomized NC result.

We also consider the problem of learning a minimal multiplicity automaton in a passive efficient exact learning framework, where the aim is to find a minimal automaton consistent with a given set of observations. Specifically, we look at the following *minimal consistency problem*: Does there exist a multiplicity automaton with a given number of states that is consistent with a given finite sample of weight-labelled words or trees? We show that, over both words and trees, this decision problem is interreducible with the problem of deciding the truth of existential first-order sentences over the field of rationals—whose decidability is a longstanding open

problem.

The second part of this thesis studies matrix factorization and its relations to automaton learning and minimization. In this regard, we note that our learning and minimization algorithms mentioned above all rely on a matricial representation, called the *Hankel matrix*, of the multiplicity automaton. The Hankel matrix representing a multiplicity automaton is a bi-infinite matrix with finite rank that stores all values of the automaton in a redundant yet convenient way.

The notion of a Hankel matrix plays an important role in learning and system identification. This representation has been used in tandem with matrix completion and factorization, namely the singular value decomposition (SVD), to develop efficient and provably correct algorithms for learning weighted word automata from labelled data [Hsu et al., 2012, Balle and Mohri, 2012]. The basis of these *spectral learning* algorithms is the "Hankel trick", which consists of learning a low-rank approximation of the Hankel matrix, and then factoring it.

Our minimization algorithms for multiplicity word and tree automata rely on producing small spanning sets for the forward and backward space of the Hankel matrix of the input automaton. This yields a *forward-backward* factorization of the Hankel matrix into a product of its *forward* matrix $W$ and *backward* matrix $H$. Each of the columns of $W$ corresponds to computations of the automaton with a specific final state. The forward-backward factorization yields a parts-based representation of the distribution given by the automaton: the weight of a word or tree is a linear combination, with coefficients given in the corresponding column of $H$, of the weights of all accepting computations. In particular, the rank of the Hankel matrix is at most the smallest number of states of any multiplicity automaton it represents. In fact, equality is known to hold. This fundamental result was shown by Carlyle and Paz [1971] and Fliess [1974] for multiplicity word automata and by Bozapalidis and Louscou-Bozapalidou [1983] for multiplicity tree automata.

When the multiplicity automaton is probabilistic or has nonnegative weights, its Hankel matrix is nonnegative (i.e., has nonnegative entries) and so are its forward matrix $W$ and backward matrix $H$. Therefore, in this case the forward-backward factorization $W \cdot H$ is a *nonnegative matrix factorization* of the Hankel matrix.

Nonnegative matrix factorization (NMF) is the process of representing a given nonnegative matrix $M$ as a product $W \cdot H$ where $W$ and $H$ are nonnegative (low-rank) matrices. For an NMF $M = W \cdot H$, the number of columns in $W$ is called the *inner dimension*. The smallest inner dimension of any NMF of $M$ is called the *nonnegative rank* of $M$. The nonnegative rank of $M$, a notion introduced by Gregory and Pullman [1983], can equivalently be defined as the smallest number of nonnegative rank-one matrices into which $M$ can be decomposed additively.

The forward-backward factorization induced by a probabilistic automaton with $n$ states thus corresponds to an NMF with inner dimension $n$ of the Hankel matrix. Therefore, the nonnegative rank of the Hankel matrix is at most $n$. We note that any equivalent probabilistic automaton with $n$ states also induces a forward-backward factorization of inner dimension $n$.

NMF is a powerful dimension-reduction technique and has been applied to datasets in many scientific disciplines. Nonnegative matrices commonly occur as data matrices representing text and image datasets. In machine learning, NMF was popularised by the seminal work of Lee and Seung [1999] as a tool for finding features in facial-image databases. Since then, NMF has found a broad range of applications, including document clustering, topic modelling, computer vision, recommender systems, bioinformatics, and acoustic signal processing [Bucak and Günsel, 2007, Berry et al., 2009, Cichocki et al., 2009, Tjioe et al., 2010, Yokota et al., 2015, Zhang et al., 2006].

From a computational perspective, the nonnegative rank of a matrix is a nontrivial quantity to compute. The usual rank of a matrix $M$ is greater than or equal to $r$ if and only if $M$ has an $r \times r$ submatrix of rank $r$. However, the same property does

not hold for nonnegative rank. This follows from a construction by Moitra [2016] of a family of matrices, indexed by $r, n \in \mathbb{N}$, that have size $3rn \times 3rn$ and nonnegative rank at least $4r$, but no $(n-1) \times 3rn$ submatrix of nonnegative rank greater than $3r$.

Despite being an important technique in practice, there are a number of open questions about the complexity of NMF and the existence of efficient and provably-correct heuristics. A central computational problem is the so-called *NMF problem*, which asks whether a nonnegative matrix admits a nonnegative factorization of inner dimension at most a given natural number. The NMF problem was shown to be NP-hard by Vavasis [2009]. It is not known whether the NMF problem lies in NP.

A closely-related question, posed by Cohen and Rothblum [1993], is that of finding an optimal nonnegative matrix factorization over the rational numbers: Given a nonnegative rational matrix $M$ as input, can one always find an optimal nonnegative factorization of $M$ into rational matrices? This question, henceforth referred to as the *Cohen–Rothblum problem*, is important both for determining the complexity of the NMF problem and for usability of the output in machine learning applications. Indeed, from the point of view of computational complexity, a natural route to establishing membership of the NMF problem in NP would be to show that there always exists an optimal nonnegative factorization over the rational numbers of polynomial bit-length in the input.

Rationality of solutions to optimization problems is a central topic in computer science, in part due to its connection to the computational complexity of finding optimal solutions. For various computational problems associated with partially observable Markov decision processes [Vlassis et al., 2012], Nash equilibria [Etessami and Yannakakis, 2010], concurrent stochastic games [Etessami and Yannakakis, 2008], and Shapley's stochastic games [Shapley, 1953], it has been shown that irrational numbers may be needed to express optimal solutions. In conjunction with this, complexity-theoretic lower bounds have been derived, e.g., SQRT-SUM-hardness. In computa-

tional geometry, Abrahamsen et al. [2017] show that for the *art gallery problem*, where the aim is to place a minimum number of guards inside a simple polygon with integer coordinates such that the guards together can see the whole polygon, an optimal solution may require guard positions with irrational coordinates.

We first study a variant of the Cohen–Rothblum problem for *restricted NMF (RNMF)*, a notion introduced by Gillis and Glineur [2012], which is any NMF $M = W \cdot H$ where the columns of $W$ span the same vector space as the columns of $M$. The *restricted NMF (RNMF) problem* is defined as the NMF problem except that the factorization is required to be restricted. Restricted NMF has applications in topic modelling and a natural geometric interpretation as the nested polytope problem.

We show that the RNMF problem for rational nonnegative matrices $M$ of rank 3 or less can be solved in polynomial time.[1] In fact, we show that there is always a *rational* restricted NMF of $M$ with minimal inner dimension, and that it can be computed in polynomial time in the Turing model of computation. This improves a result in [Gillis and Glineur, 2012] where the RNMF problem for matrices of rank 3 or less is shown to be solvable in polynomial time assuming a RAM model with unit-cost arithmetic. Complementing our rationality result for rank-3 matrices, we exhibit a rank-4 matrix that has an RNMF with inner dimension 5 but *no rational* RNMF with inner dimension 5. We thus answer the RNMF variant of the Cohen–Rothblum problem negatively.

We use our insights from the restricted NMF variant as a stepping stone towards solving the Cohen–Rothblum problem in its full generality. Specifically, we exhibit a rational matrix $M$ that has different nonnegative ranks over $\mathbb{R}$ and over $\mathbb{Q}$. In other words, any NMF $M = W \cdot H$ of minimal inner dimension has irrational entries in $W$ and $H$. Our counterexample is almost optimal inasmuch as $M$ has rank 4, whereas for matrices of rank at most 2 it was known already to Cohen and Rothblum [1993]

[1] We remark that a rank-3 nonnegative matrix can have an arbitrarily large nonnegative rank.

that nonnegative ranks coincide over $\mathbb{R}$ and $\mathbb{Q}$.

Nonnegative matrix factorization has deep connections with minimization of probabilistic automata and labelled Markov chains. As a final contribution of this thesis, we apply our results on rationality of NMF and restricted NMF to answer longstanding open problems concerning minimization of probabilistic automata and labelled Markov chains (LMCs). We consider two minimization problems for LMCs: the problem of finding a minimal covering LMC and the problem of finding a minimal equivalent LMC. We reduce NMF to each of these minimization problems.

In his seminal 1971 textbook, Paz [1971] asked the following question about the nature of minimization for LMCs: When searching for a minimal covering LMC, can one look only at LMCs that have the same rank as the input LMC? We use restricted NMF to answer Paz's question negatively, thus falsifying a positive answer claimed in 1974 by Bancilhon [1974]. Instrumental to our counterexample is the observation that restricted nonnegative rank and nonnegative rank can be different. We moreover apply our answer to the Cohen–Rothblum problem, namely that nonnegative matrix factorization requires irrationality, to show that state minimization of labelled Markov chains can require the introduction of irrational transition probabilities.

In the following Section 1.1 we discuss the main contributions of the thesis. In Section 1.2 we outline the thesis structure and summarise the content of each chapter.

## 1.1 Contributions

In this section, we discuss the main contributions of this thesis.

We first study the problem of learning multiplicity tree automata in the exact learning model of Angluin [1987]. We are interested in questions of succinctness and computational efficiency, both from the point of view of the oracle and the learning algorithm. We first characterise the complexity of the equivalence problem for mul-

tiplicity tree automata, a central computational problem from the point of view of the oracle, showing that it is logspace equivalent to polynomial identity testing. We then derive lower bounds on the number of queries needed to learn multiplicity tree automata over both fixed and arbitrary fields. In the latter case, the bound is linear in the size of the target automaton.

Furthermore, we give a new learning algorithm for multiplicity tree automata in which counterexamples to equivalence queries are represented as DAGs. The query complexity of this algorithm is quadratic in the target automaton size and linear in the size of a largest counterexample. In particular, if the oracle always returns DAG counterexamples of minimal size then the query complexity is quadratic in the target automaton size—almost matching the lower bound, and improving the best previously-known algorithm [Habrard and Oncina, 2006] by an exponential factor.

Closely related to learning and equivalence is the problem of minimizing the number of states in a multiplicity tree automaton over the field $\mathbb{Q}$ of rational numbers. Here we give a minimization algorithm that runs in polynomial time assuming unit-cost arithmetic, and show that a polynomial bound in the standard Turing model would require a breakthrough in the complexity of polynomial identity testing by proving that the latter problem is logspace equivalent to the decision version of minimization. We also give an NC algorithm for minimizing the number of states in a multiplicity word automaton over $\mathbb{Q}$, improving on a randomized NC procedure of Kiefer et al. [2013]. Finally, in the context of learning a minimal multiplicity automaton in a passive learning framework, we consider the following minimal consistency problem: Does there exist a multiplicity automaton with a given number of states that is consistent with a given finite sample of weight-labelled words or trees? We show that this decision problem is interreducible with the problem of determining the truth of existential first-order sentences over $\mathbb{Q}$, whose decidability is a longstanding open problem.

The second part of the thesis focuses on nonnegative matrix factorization and its relations to learning and minimization for probabilistic automata. Nonnegative matrix factorization (NMF) is the problem of decomposing a given nonnegative $n \times m$ matrix $M$ into a product of a nonnegative $n \times d$ matrix $W$ and a nonnegative $d \times m$ matrix $H$. There are a number of open questions about the theory of NMF. One important longstanding open problem, due to Cohen and Rothblum [1993], asks whether the nonnegative ranks of a rational matrix over the reals and over the rationals coincide.

We first consider a variant of this problem for restricted NMF, which requires that the column spaces of $M$ and $W$ coincide. We show that a rational matrix $M$ of rank at most 3 always has a restricted NMF of minimal inner dimension whose factors $W$ and $H$ are also rational. Furthermore, we exhibit a rank-4 matrix for which $W$ and $H$ require irrational entries.

Using the restricted NMF variant as a stepping stone, we proceed to give a negative answer to the Cohen–Rothblum problem in its full generality, by exhibiting a rank-4 matrix $M$ that has different nonnegative ranks over $\mathbb{R}$ and over $\mathbb{Q}$.

As an application of our results on nonnegative matrix factorization, we answer some longstanding open problems concerning minimization of labelled Markov chains. We consider the problem of finding a minimal covering LMC and the problem of finding a minimal equivalent LMC, and show that NMF is reducible to each of these problems.

We apply our results on restricted NMF to give a negative answer to the following question asked by Paz in his seminal 1971 textbook: When searching for a minimal covering labelled Markov chain, can one look only at labelled Markov chains that have the same rank as the input? We also falsify a positive answer claimed in 1974.

Secondly, we consider the following analogue of the Cohen–Rothblum problem: Given an LMC with rational transition probabilities, is there always a minimal equiv-

alent LMC also with rational transition probabilities? We apply our solution to the Cohen–Rothblum problem to give a negative answer to this question.

## 1.2 Thesis Structure

This thesis consists of two core parts: Part I, which focuses on automaton learning and minimization, and Part II, which focuses on matrix factorization. We conclude the thesis with Part III, which summarises our results and discusses future work.

In the following we summarise each chapter of this thesis and elaborate on the material contained therein. This thesis contains material that has been previously published, and we list below under each chapter any publications it is based on. The authors in all publications are listed in alphabetical order.

### Chapter 2: Overview of Multiplicity Automata

We introduce the basic definitions and concepts relevant to Part I of the thesis, which focuses on automaton learning and minimization. We first give some basic notation and terminology that is used throughout the thesis. We then introduce multiplicity automata over words and trees, and prove some basic technical results concerning these automata. In particular, we define the Hankel matrix representation of multiplicity automata over words and trees.

The material presented in this chapter is based on the following journal paper:

> Ines Marušić and James Worrell. Complexity of Equivalence and Learning for Multiplicity Tree Automata. *Journal of Machine Learning Research (JMLR)*, Volume 16, Dec. 2015.

## Chapter 3: Exact Learning of Multiplicity Tree Automata

We consider the query and computational complexity of learning multiplicity tree automata in the exact learning model. We first introduce the notion of a multiplicity tree automaton running on DAGs. We show that the equivalence problem for multiplicity tree automata is logspace equivalent to polynomial identity testing. We then give a new learning algorithm for multiplicity tree automata in which counterexamples to equivalence queries are represented as DAGs. Assuming the oracle always returns DAG counterexamples of minimal size, our algorithm improves the best previously-known algorithm [Habrard and Oncina, 2006] by an exponential factor. Lastly, we derive lower bounds on the number of queries needed to learn multiplicity tree automata over both fixed and arbitrary fields.

The material presented in this chapter is based on the following conference paper and the subsequent journal paper:

> Ines Marušić and James Worrell. Complexity of Equivalence and Learning for Multiplicity Tree Automata. In *Proceedings of the International Symposium on Mathematical Foundations of Computer Science (MFCS)*, 2014.

> Ines Marušić and James Worrell. Complexity of Equivalence and Learning for Multiplicity Tree Automata. *Journal of Machine Learning Research (JMLR)*, Volume 16, Dec. 2015.

## Chapter 4: Minimization of Multiplicity Tree Automata

We consider the problem of minimizing the number of states in a multiplicity tree automaton over the field $\mathbb{Q}$. We give a minimization algorithm that runs in polynomial time assuming unit-cost arithmetic, and show that the decision version of this

problem is logspace equivalent to polynomial identity testing. We also give an NC algorithm for minimizing multiplicity word automata. Lastly, we consider the minimal consistency problem: does there exist a multiplicity automaton with a given number of states that is consistent with a given finite sample of weight-labelled words or trees? We show that, over both words and trees, this decision problem is interreducible with the problem of deciding the truth of existential first-order sentences over $\mathbb{Q}$.

The material presented in this chapter is based on the following conference paper and the subsequent journal paper:

> Stefan Kiefer, Ines Marušić, and James Worrell. Minimisation of Multiplicity Tree Automata. In *Proceedings of the International Conference on Foundations of Software Science and Computation Structures (FoSSaCS)*, 2015.

> Stefan Kiefer, Ines Marušić, and James Worrell. Minimisation of Multiplicity Tree Automata. To appear in the *Logical Methods in Computer Science (LMCS)* journal.

## Chapter 5: Overview of Nonnegative Matrix Factorization

We introduce the basic definitions and concepts relevant for Part II of the thesis, namely the nonnegative matrix factorization, the restricted nonnegative matrix factorization, and relevant geometric definitions including the nested polytope problem. We also provide a high-level introduction to our subsequent results in Chapters 6 and 7, and give an overview of the related literature.

The work presented in this chapter is based on the following papers:

> Dmitry Chistikov, Stefan Kiefer, Ines Marušić, Mahsa Shirmohammadi, and James Worrell. On Restricted Nonnegative Matrix Factorization. In

*Proceedings of the International Colloquium on Automata, Languages and Programming (ICALP)*, 2016.

Dmitry Chistikov, Stefan Kiefer, Ines Marušić, Mahsa Shirmohammadi, and James Worrell. Nonnegative Matrix Factorization Requires Irrationality. To appear in *SIAM Journal on Applied Algebra and Geometry (SIAGA)*.

## Chapter 6: Restricted Nonnegative Matrix Factorization Requires Irrationality

We investigate whether a rational matrix $M$ always has a restricted NMF $M = W \cdot H$ of minimal inner dimension whose factors $W$ and $H$ are also rational. We show that this holds for matrices $M$ of rank at most 3 and we exhibit a rank-4 matrix for which $W$ and $H$ require irrational entries.

The work presented in this chapter is based on the following publication:

Dmitry Chistikov, Stefan Kiefer, Ines Marušić, Mahsa Shirmohammadi, and James Worrell. On Restricted Nonnegative Matrix Factorization. In *Proceedings of the International Colloquium on Automata, Languages and Programming (ICALP)*, 2016.

## Chapter 7: Nonnegative Matrix Factorization Requires Irrationality

We answer a longstanding open question about the complexity of nonnegative matrix factorization, posed by Cohen and Rothblum [1993], which asks whether an optimal nonnegative factorization into rational matrices can always be found if the input matrix is rational. We answer this question negatively, by exhibiting a rank-4 matrix that has different nonnegative ranks over $\mathbb{R}$ and over $\mathbb{Q}$.

The work presented in this chapter is based on the following papers:

Dmitry Chistikov, Stefan Kiefer, Ines Marušić, Mahsa Shirmohammadi, and James Worrell. On Rationality of Nonnegative Matrix Factorization. In *Proceedings of the ACM-SIAM Symposium on Discrete Algorithms (SODA)*, 2017.

Dmitry Chistikov, Stefan Kiefer, Ines Marušić, Mahsa Shirmohammadi, and James Worrell. Nonnegative Matrix Factorization Requires Irrationality. To appear in *SIAM Journal on Applied Algebra and Geometry (SIAGA)*.

## Chapter 8: Minimization of Labelled Markov Chains and Nonnegative Matrix Factorization

We apply our results from Chapters 6 and 7 to minimization of labelled Markov chains. First we look at the problem of finding a minimal covering labelled Markov chain, focusing on the following open question posed by Paz [1971]: When searching for a minimal covering labelled Markov chain, can one look only at those labelled Markov chains that have the same rank as the input? We answer this question negatively, and also falsify a positive answer claimed in 1974. Secondly, we show that state minimization of labelled Markov chains can require the introduction of irrational transition probabilities.

The material presented in this section is based on the following conference papers:

Dmitry Chistikov, Stefan Kiefer, Ines Marušić, Mahsa Shirmohammadi, and James Worrell. On Restricted Nonnegative Matrix Factorization. In *Proceedings of the International Colloquium on Automata, Languages and Programming (ICALP)*, 2016.

Dmitry Chistikov, Stefan Kiefer, Ines Marušić, Mahsa Shirmohammadi, and James Worrell. On Rationality of Nonnegative Matrix Factorization. In *Proceedings of the ACM-SIAM Symposium on Discrete Algorithms (SODA)*, 2017.

## Chapter 9: Conclusion

We summarise the results of the thesis.

## Chapter 10: Future Work

We highlight some open problems and possible directions for future research.

# Part I

# Learning via Automaton Minimization

# Chapter 2

# Overview of Multiplicity Automata

**Abstract**

*We present the necessary background for this first part of the thesis, which focuses on automaton learning and minimization. This chapter is based on the material in [Marušić and Worrell, 2015].*

---

In Section 2.1 we introduce some basic notation and terminology that we will use in this thesis. We introduce multiplicity automata over words and trees in Sections 2.2 and 2.3, respectively. These automata compute functions mapping words and trees, respectively, to field elements. We also introduce a redundant yet convenient matricial representation of these functions, called the Hankel matrix, which reveals a lot of useful properties of the corresponding automaton representations.

## 2.1 Basic Notation and Terminology

Let $\mathbb{N}$ and $\mathbb{N}_0$ denote the set of all positive and nonnegative integers, respectively. For any $n \in \mathbb{N}$, we write $[n]$ for the set $\{1, 2, \ldots, n\}$ and write $I_n$ for the identity matrix

of order $n$. We write $\mathbf{1}_n$ (resp., $\mathbf{0}_n$) for the $n$-dimensional column vector with all ones (resp., zeros), where we omit the subscript if it is understood from the context. For every $i \in [n]$, we write $e_i$ for the $i^{\text{th}}$ $n$-dimensional coordinate row vector.

Given an ordered field $\mathbb{F}$, we denote by $\mathbb{F}_+$ the set of all its nonnegative elements. We write $\mathbb{F}^n = \mathbb{F}^{n\times 1}$ for the set of all $n$-dimensional column vectors with entries in the field $\mathbb{F}$. Given a vector $v \in \mathbb{F}^n$, we write $v_i$ for its $i^{\text{th}}$ entry. A vector of real numbers $v$ is called *pseudo-stochastic* if its entries sum up to one. A pseudo-stochastic vector $v$ is called *stochastic* if its entries are also nonnegative.

For any matrix $M$, we write $M_{i,:}$ for its $i^{\text{th}}$ row, $M_{:,j}$ for its $j^{\text{th}}$ column, and $M_{i,j}$ for its $(i,j)^{\text{th}}$ entry. Given nonempty subsets $I$ and $J$ of the row and column indices of $M$, respectively, we write $M_{I,J}$ for the submatrix $(M_{i,j})_{i\in I, j\in J}$ of $M$. For singletons, we write simply $M_{i,J} := M_{\{i\},J}$ and $M_{I,j} := M_{I,\{j\}}$. We also consider matrices whose rows and columns are indexed by tuples of natural numbers ordered lexicographically. The *column space* (resp., *row space*) of $M$, written $\mathrm{Col}(M)$ (resp., $\mathrm{Row}(M)$), is the vector space spanned by the columns (resp., rows) of $M$.

A matrix is called *nonnegative* (resp., *zero* or *rational*) if so are all its entries. A matrix is *column-stochastic* (resp., *row-stochastic*) if its columns (resp., rows) are stochastic. Column-stochastic matrices will henceforth simply be called stochastic matrices.

Let $V$ be a set. For any subset $S \subseteq V$, the *characteristic function* of $S$ (relative to $V$) is the function $\chi_S : V \to \{0,1\}$ such that $\chi_S(x) = 1$ if $x \in S$, and $\chi_S(x) = 0$ otherwise.

We will use $\Sigma$ to denote a finite alphabet and $\varepsilon$ to denote the empty word. The set of all words over $\Sigma$ is denoted by $\Sigma^*$, and the length of a word $w \in \Sigma^*$ is denoted by $|w|$. For any $n \in \mathbb{N}_0$ we write $\Sigma^n := \{w \in \Sigma^* : |w| = n\}$, $\Sigma^{\leq n} := \bigcup_{l=0}^{n} \Sigma^l$, and $\Sigma^{<n} := \Sigma^{\leq n} \setminus \Sigma^n$. Given two words $x, y \in \Sigma^*$, we denote by $xy$ the concatenation of $x$ and $y$. Given two sets $X, Y \subseteq \Sigma^*$, we define $XY := \{xy : x \in X, y \in Y\}$.

## 2.2 Multiplicity Word Automata

Multiplicity word automata were introduced by Schützenberger [1961], and have been used for modelling probabilistic word automata and ambiguity in nondeterministic finite automata. They are essentially nondeterministic finite automata whose edges are assigned weights from some field $\mathbb{F}$. A multiplicity word automaton assigns a weight from $\mathbb{F}$ to every input string $x$, which is the sum of the weights of all paths in the automaton consistent with $x$, where the weight of a path is the product of the weights on all its edges.

Formally, let $\Sigma$ be a finite alphabet and let $\mathbb{F}$ be a field. A *word series* over $\Sigma$ with coefficients in $\mathbb{F}$ is a mapping $f : \Sigma^* \to \mathbb{F}$. The set of all word series over $\Sigma$ with coefficients in $\mathbb{F}$ is denoted by $\mathbb{F}\langle\langle\Sigma^*\rangle\rangle$. The *Hankel matrix* of a word series $f \in \mathbb{F}\langle\langle\Sigma^*\rangle\rangle$ is a bi-infinite matrix $H : \Sigma^* \times \Sigma^* \to \mathbb{F}$ such that $H_{x,y} = f(xy)$ for all $x, y \in \Sigma^*$.

An $\mathbb{F}$-*multiplicity word automaton* ($\mathbb{F}$-*MWA*) is a 5-tuple $\mathcal{A} = (n, \Sigma, \mu, \alpha, \gamma)$ which consists of the *dimension* $n \in \mathbb{N}_0$ representing the number of states, a finite alphabet $\Sigma$, a function $\mu : \Sigma \to \mathbb{F}^{n\times n}$ assigning a *transition matrix* $\mu(\sigma)$ to each $\sigma \in \Sigma$, the *initial weight vector* $\alpha \in \mathbb{F}^{1\times n}$, and the *final weight vector* $\gamma \in \mathbb{F}^{n\times 1}$. We speak of an MWA if the field $\mathbb{F}$ is clear from the context or irrelevant. Figure 2.1 gives an example of a $\mathbb{Q}$-multiplicity word automaton.

We extend the function $\mu$ from $\Sigma$ to $\Sigma^*$ by defining $\mu(\varepsilon) := I_n$, and $\mu(\sigma_1 \cdots \sigma_k) := \mu(\sigma_1) \cdot \ldots \cdot \mu(\sigma_k)$ for any $\sigma_1, \ldots, \sigma_k \in \Sigma$. It is easy to see that for any $x, y \in \Sigma^*$,

$$\mu(xy) = \mu(x) \cdot \mu(y). \tag{2.1}$$

Automaton $\mathcal{A}$ *recognises* the word series $\|\mathcal{A}\| : \Sigma^* \to \mathbb{F}$ where $\|\mathcal{A}\|(w) = \alpha \cdot \mu(w) \cdot \gamma$ for every $w \in \Sigma^*$.

Two MWAs $\mathcal{A}_1$, $\mathcal{A}_2$ are said to be *equivalent* if $\|\mathcal{A}_1\| = \|\mathcal{A}_2\|$. An MWA is said to

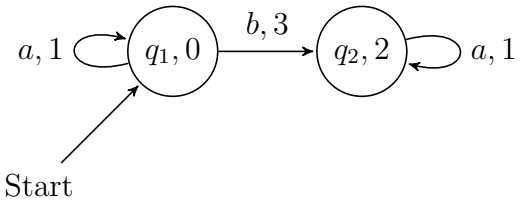



Figure 2.1: Graph representation of a $\mathbb{Q}$-MWA $\mathcal{A} = (2, \Sigma, \mu, e_1, \gamma)$ where $\Sigma = \{a, b\}$, $\mu(a) = \left(\begin{smallmatrix} 1 & 0 \\ 0 & 1 \end{smallmatrix}\right)$, $\mu(b) = \left(\begin{smallmatrix} 0 & 3 \\ 0 & 0 \end{smallmatrix}\right)$, and $\gamma = \left(\begin{smallmatrix} 0 & 2 \end{smallmatrix}\right)^\top$. The two states of the automaton $\mathcal{A}$ are called $q_1$ and $q_2$, and their initial weights are 1 and 0, respectively. Their final weights (marked inside the states) are 0 and 2, respectively. Each transition arrow corresponds to a nonzero entry of one of the transition matrices $\mu(a)$ and $\mu(b)$, representing a state-to-state transition with nonzero weight, with the number on the arrow corresponding to the value of that entry.

be *minimal* if no equivalent automaton has strictly smaller dimension. A word series is called *recognisable* if it is recognised by some MWA.

The following result of Carlyle and Paz [1971] and Fliess [1974] is a fundamental theorem in the theory of word series. It relates the number of states in a minimal automaton recognising a given word series to the rank of its Hankel matrix.

**Theorem 1** (Carlyle and Paz, 1971, Fliess, 1974)**.** *Let $\Sigma$ be a finite alphabet, $\mathbb{F}$ be a field, and $f \in \mathbb{F}\langle\langle\Sigma^*\rangle\rangle$. Let $H$ be the Hankel matrix of $f$. Then, $f$ is recognisable if and only if $H$ has finite rank over $\mathbb{F}$. In case $f$ is recognisable, the dimension of a minimal automaton recognising $f$ is* $\operatorname{rank}(H)$ *over $\mathbb{F}$.*

**Remark 2.** *The proofs of Theorem 1 by Carlyle and Paz [1971] and Fliess [1974] show that if $X, Y \subseteq \Sigma^*$ are such that* $\operatorname{rank}(H_{X,Y}) = \operatorname{rank}(H)$*, then $f$ is uniquely determined by $H_{X,Y}$ and $H_{X\Sigma,Y}$.*

## 2.3 Multiplicity Tree Automata

Multiplicity tree automata were introduced by Berstel and Reutenauer [1982] under the terminology of linear representations of tree series. They augment classical finite tree automata by having transitions labelled by numerical quantities—formally, a

value in some field $\mathbb{F}$. A multiplicity tree automaton can be evaluated on a tree to give a value in the field $\mathbb{F}$, using a definition that generalises the evaluation of an ordinary nondeterministic tree automaton: while an ordinary tree automaton performs a run on a tree node $v$ by making runs on the children of $v$ to get an accept or a reject value and returns true on $v$ if and only if all these sub-runs accept, a multiplicity automaton performs a run on the children on $v$ to obtain a value and then multiplies those values with the weight of the transition into $v$ from its children. Similarly, while a traditional automaton evaluates to true if and only if some run accepts, a multiplicity automaton performs all runs and adds the corresponding values.

Multiplicity tree automata augment not only classical tree automata but also multiplicity word automata, which can be viewed as multiplicity tree automata on unary trees.

In the rest of this section, we introduce tree series and their Hankel matrices in §2.3.1 and Kronecker product in §2.3.2 before formally defining multiplicity tree automata in §2.3.3. We define minimal multiplicity tree automata in §2.3.4. Lastly, we introduce product and difference of two multiplicity tree automata in §2.3.5.

### 2.3.1 Tree Series and their Hankel Matrices

A *ranked alphabet* is a tuple $(\Sigma, \mathrm{rk})$ where $\Sigma$ is a nonempty finite set of symbols and $\mathrm{rk} : \Sigma \to \mathbb{N}_0$ is a function. Ranked alphabet $(\Sigma, \mathrm{rk})$ is often written $\Sigma$ for short. For every $k \in \mathbb{N}_0$, we define the set of all *k-ary* symbols $\Sigma_k := \mathrm{rk}^{-1}(\{k\})$. If $\sigma \in \Sigma_k$ then we say that $\sigma$ has *rank* (or *arity*) $k$. We say that $\Sigma$ has *rank* $m$ if $m = \max\{\mathrm{rk}(\sigma) : \sigma \in \Sigma\}$.

The set of $\Sigma$-*trees* (*trees* for short), written as $T_\Sigma$, is the smallest set $T$ satisfying the following two conditions: (i) $\Sigma_0 \subseteq T$; and (ii) if $k \geq 1$, $\sigma \in \Sigma_k$, $t_1, \ldots, t_k \in T$ then $\sigma(t_1, \ldots, t_k) \in T$. Given a $\Sigma$-tree $t$, a *subtree* of $t$ is a $\Sigma$-tree consisting of a node in $t$ and all of its descendants in $t$. The set of all subtrees of $t$ is denoted by $Sub(t)$.

Let $\Sigma$ be a ranked alphabet and $\mathbb{F}$ be a field. A *tree series* over $\Sigma$ with coefficients in $\mathbb{F}$ is a function $f : T_\Sigma \to \mathbb{F}$. For every $t \in T_\Sigma$, we call $f(t)$ the *coefficient* of $t$ in $f$. The set of all tree series over $\Sigma$ with coefficients in $\mathbb{F}$ is denoted by $\mathbb{F}\langle\langle T_\Sigma \rangle\rangle$.

We define the tree series $\mathit{height}, \mathit{size}, \#_\sigma \in \mathbb{Q}\langle\langle T_\Sigma \rangle\rangle$ where $\sigma \in \Sigma$, as follows: (i) if $t \in \Sigma_0$ then $\mathit{height}(t) = 0$, $\mathit{size}(t) = 1$, $\#_\sigma(t) = \chi_{\{t=\sigma\}}$; and (ii) if $t = a(t_1, \ldots, t_k)$ where $k \geq 1$, $a \in \Sigma_k$, $t_1, \ldots, t_k \in T_\Sigma$ then $\mathit{height}(t) = 1 + \max_{i\in[k]} \mathit{height}(t_i)$, $\mathit{size}(t) = 1 + \sum_{i\in[k]} \mathit{size}(t_i)$, $\#_\sigma(t) = \chi_{\{a=\sigma\}} + \sum_{i\in[k]} \#_\sigma(t_i)$, respectively. For every $n \in \mathbb{N}_0$, we define the sets $T_\Sigma^{<n} := \{t \in T_\Sigma : \mathit{height}(t) < n\}$, $T_\Sigma^n := \{t \in T_\Sigma : \mathit{height}(t) = n\}$, and $T_\Sigma^{\leq n} := T_\Sigma^{<n} \cup T_\Sigma^n$.

Let $\Box$ be a nullary symbol not contained in $\Sigma$. The set $C_\Sigma$ of $\Sigma$-*contexts* (*contexts* for short) is the set of all $(\{\Box\} \cup \Sigma)$-trees in which $\Box$ occurs exactly once. The *concatenation* of $c \in C_\Sigma$ and $t \in T_\Sigma \dot\cup C_\Sigma$, written as $c[t]$, is the tree obtained by substituting $t$ for $\Box$ in $c$. Intuitively, the $\Box$-labelled leaf of $c$ acts as a variable in that substituting a $\Sigma$-tree (respectively, $\Sigma$-context) $t$ for that variable yields a new $\Sigma$-tree ($\Sigma$-context) $c[t]$.

Let $n \in \mathbb{N}_0$. We denote by $C_\Sigma^n$ the set of all contexts $c \in C_\Sigma$ where the distance between the root and the $\Box$-labelled node of $c$ is equal to $n$. Moreover, we write $C_\Sigma^{\leq n} := \bigcup_{l=0}^n C_\Sigma^l$ and $C_\Sigma^{<n} := C_\Sigma^{\leq n} \setminus C_\Sigma^n$. A *subtree* of $c \in C_\Sigma$ is a $\Sigma$-tree consisting of a node in $c$ and all of its descendants. Given a set $S \subseteq T_\Sigma$, we denote by $C_{\Sigma,S}^n$ the set of all contexts $c \in C_\Sigma^n$ where every subtree of $c$ is an element of $S$. Moreover, we write $C_{\Sigma,S}^{\leq n} := \bigcup_{l=0}^n C_{\Sigma,S}^l$ and $C_{\Sigma,S}^{<n} := C_{\Sigma,S}^{\leq n} \setminus C_{\Sigma,S}^n$.

A *suffix* of a $\Sigma$-tree $t$ is a $\Sigma$-context $c$ such that $t = c[t']$ for some $\Sigma$-tree $t'$. The *Hankel matrix* of a tree series $f \in \mathbb{F}\langle\langle T_\Sigma \rangle\rangle$ is a bi-infinite matrix $H : T_\Sigma \times C_\Sigma \to \mathbb{F}$ such that $H_{t,c} = f(c[t])$ for every $t \in T_\Sigma$ and $c \in C_\Sigma$.

### 2.3.2 Kronecker Product

Let $A$ be an $m_1 \times n_1$ matrix and $B$ an $m_2 \times n_2$ matrix. The *Kronecker product* of $A$ by $B$, written as $A \otimes B$, is an $m_1 m_2 \times n_1 n_2$ matrix where

$$(A \otimes B)_{(i_1,i_2),(j_1,j_2)} = A_{i_1,j_1} \cdot B_{i_2,j_2}$$

for every $i_1 \in [m_1]$, $i_2 \in [m_2]$, $j_1 \in [n_1]$, $j_2 \in [n_2]$. Recall here that $(A \otimes B)_{(i_1,i_2),(j_1,j_2)} := (A \otimes B)_{(i_1-1)m_2+i_2,(j_1-1)n_2+j_2}$.

The Kronecker product is bilinear, associative, and has the following *mixed-product property*: For any matrices $A$, $B$, $C$, $D$ such that products $A \cdot C$ and $B \cdot D$ are defined, it holds that $(A \otimes B) \cdot (C \otimes D) = (A \cdot C) \otimes (B \cdot D)$.

For every $k \in \mathbb{N}_0$ we define the *k-fold Kronecker power* of a matrix $A$, written as $A^{\otimes k}$, inductively by $A^{\otimes 0} = I_1$ and $A^{\otimes k} = A^{\otimes(k-1)} \otimes A$ for $k \geq 1$.

**Some Properties of Kronecker Product** Let $k \in \mathbb{N}_0$. For any square matrices $A$ and $B$, we have

$$(A \otimes B)^k = A^k \otimes B^k. \tag{2.2}$$

*Proof of Equation* (2.2). We use induction on $k$. The result trivially holds when $k = 0$. For the induction step, assume that $(A \otimes B)^k = A^k \otimes B^k$ for some $k \in \mathbb{N}_0$. Using the mixed-product property we get that

$$\begin{aligned}(A \otimes B)^{k+1} = (A \otimes B)^k \cdot (A \otimes B) &= (A^k \otimes B^k) \cdot (A \otimes B) \\ &= (A^k \cdot A) \otimes (B^k \cdot B) = A^{k+1} \otimes B^{k+1},\end{aligned}$$

which completes the proof by induction. □

Let $A_1, \ldots, A_k$ be matrices. We will often denote $\bigotimes_{l=1}^{k} A_l := A_1 \otimes \cdots \otimes A_k$.

Suppose that for every $l \in [k]$, matrix $A_l$ has $n_l$ rows. Then,

$$\left(\bigotimes_{l=1}^{k} A_l\right)_{(i_1,\ldots,i_k),:} = \bigotimes_{l=1}^{k} (A_l)_{i_l,:} \tag{2.3}$$

for every $(i_1, \ldots, i_k) \in [n_1] \times \cdots \times [n_k]$. Here we recall that row index $(i_1, \ldots, i_k)$ denotes the $(\sum_{l=1}^{k-1}(i_l - 1) \cdot (\prod_{p=l+1}^{k} n_p) + i_k)^{\text{th}}$ row of $\bigotimes_{l=1}^{k} A_l$.

*Proof of Equation* (2.3). We use induction on $k$. The result trivially holds if $k = 1$. Assume that (2.3) holds for some $k \in \mathbb{N}$. Then by definition, we have that

$$\left(\bigotimes_{l=1}^{k+1} A_l\right)_{(i_1,\ldots,i_{k+1}),:} = \left(\bigotimes_{l=1}^{k} A_l\right)_{(i_1,\ldots,i_k),:} \otimes (A_{k+1})_{i_{k+1},:} = \bigotimes_{l=1}^{k} (A_l)_{i_l,:}.$$

This completes the proof by induction. □

Let $k \in \mathbb{N}_0$, and let $A_1, \ldots, A_k$ and $B_1, \ldots, B_k$ be matrices such that the product $A_l \cdot B_l$ is defined for every $l \in [k]$. Then,

$$\bigotimes_{l=1}^{k} A_l \cdot \bigotimes_{l=1}^{k} B_l = \bigotimes_{l=1}^{k} (A_l \cdot B_l)\,. \tag{2.4}$$

*Proof of Equation* (2.4). We use induction on $k$. The case $k = 0$ trivially holds. Assume that (2.4) holds for some $k \in \mathbb{N}_0$. Then, by associativity and the mixed-product property of Kronecker product we have

$$\begin{aligned}\bigotimes_{l=1}^{k+1} A_l \cdot \bigotimes_{l=1}^{k+1} B_l &= \left(\bigotimes_{l=1}^{k} A_l \cdot \bigotimes_{l=1}^{k} B_l\right) \otimes (A_{k+1} \cdot B_{k+1}) \\ &= \bigotimes_{l=1}^{k} (A_l \cdot B_l) \otimes (A_{k+1} \cdot B_{k+1}) = \bigotimes_{l=1}^{k+1} (A_l \cdot B_l)\,.\end{aligned}$$

This completes the proof by induction. □

### 2.3.3 Multiplicity Tree Automata

In this subsection, we formally define multiplicity tree automata. We also give an example of a multiplicity tree automaton and describe its computation.

Let $\mathbb{F}$ be a field. An $\mathbb{F}$-*multiplicity tree automaton* ($\mathbb{F}$-*MTA*) is a quadruple $\mathcal{A} = (n, \Sigma, \mu, \gamma)$ which consists of the *dimension* $n \in \mathbb{N}_0$ representing the number of states, a ranked alphabet $\Sigma$, a family of *transition matrices* $\mu = \{\mu(\sigma) : \sigma \in \Sigma\}$, where $\mu(\sigma) \in \mathbb{F}^{n^{\mathrm{rk}(\sigma)} \times n}$, and the *final weight vector* $\gamma \in \mathbb{F}^{n \times 1}$. We speak of an MTA if the field $\mathbb{F}$ is clear from the context or irrelevant. The *size* of the automaton $\mathcal{A}$, written as $|\mathcal{A}|$, is defined as

$$|\mathcal{A}| := \sum_{\sigma \in \Sigma} n^{\mathrm{rk}(\sigma)+1} + n.$$

That is, the size of $\mathcal{A}$ is the total number of entries in all transition matrices and the final weight vector.[1]

**Example 1.** *Let* $\Sigma = \{0, 1, +, \times, -\}$ *be a ranked alphabet where* $0, 1$ *are nullary symbols and* $+, \times, -$ *are binary symbols. We define an* $\mathbb{F}$*-MTA* $\mathcal{A} = (2, \Sigma, \mu, \gamma)$ *as follows. Automaton* $\mathcal{A}$ *has two states,* $q_1$ *and* $q_2$*, and final weight vector* $\gamma = [0 \;\; 1]^\top$*. This means that states* $q_1$ *and* $q_2$ *have final weights* $\gamma_1 = 0$ *and* $\gamma_2 = 1$*, respectively. Given a symbol* $\sigma \in \Sigma$ *of rank* $k$*, the transition matrix* $\mu(\sigma)$ *has dimension* $2^k \times 2$ *and stores the weights of transitions from each* $k$*-tuple of origin states to each destination state on reading symbol* $\sigma$*. Specifically, the transition matrices of* $\mathcal{A}$ *are* $\mu(0) = [1 \;\; 0]$*,*

[1] We measure size assuming explicit rather than sparse representations of the transition matrices and final weight vector because minimal automata are only unique up to change of basis (see Theorem 5).

$\mu(1) = [1 \;\; 1]$,

$$\mu(+) = \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 0 & 1 \\ 0 & 0 \end{bmatrix}, \; \mu(-) = \begin{bmatrix} 1 & 0 \\ 0 & -1 \\ 0 & 1 \\ 0 & 0 \end{bmatrix}, \; \textit{and} \;\; \mu(\times) = \begin{bmatrix} 1 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 1 \end{bmatrix}.$$

*Entry* $\mu(1)_2 = 1$ *means that there is a transition* $1 \xrightarrow{1} q_2$ *with weight* 1 *into state* $q_2$ *on reading symbol* 1. *Similarly, entry* $\mu(+)_{(2,1),2} = 1$ *means*[2] *that there is a transition* $+(q_2, q_1) \xrightarrow{1} q_2$ *with weight* 1 *from pair of states* $(q_2, q_1)$ *into state* $q_2$ *on reading symbol* $+$.

We extend $\mu$ from $\Sigma$ to $T_\Sigma$ by defining

$$\mu(\sigma(t_1, \ldots, t_k)) := (\mu(t_1) \otimes \cdots \otimes \mu(t_k)) \cdot \mu(\sigma)$$

for every $\sigma \in \Sigma_k$ and $t_1, \ldots, t_k \in T_\Sigma$. The tree series $\|\mathcal{A}\| \in \mathbb{F}\langle\langle T_\Sigma \rangle\rangle$ *recognised* by $\mathcal{A}$ is defined by $\|\mathcal{A}\|(t) = \mu(t) \cdot \gamma$ for every $t \in T_\Sigma$. Note that a 0-dimensional multiplicity tree automaton necessarily recognises a zero tree series.

We further extend $\mu$ from $T_\Sigma$ to $C_\Sigma$ by treating $\Box$ as a unary symbol and defining $\mu(\Box) := I_n$. This allows to define $\mu(c) \in \mathbb{F}^{n \times n}$ for every $c = \sigma(t_1, \ldots, t_k) \in C_\Sigma$ inductively by writing $\mu(c) := (\mu(t_1) \otimes \cdots \otimes \mu(t_k)) \cdot \mu(\sigma)$. For every $t \in T_\Sigma \cup C_\Sigma$ and $c \in C_\Sigma$, it holds that

$$\mu(c[t]) = \mu(t) \cdot \mu(c). \tag{2.5}$$

*Proof of Equation* (2.5). We use induction on $\mathit{height}(c)$. The base case $c = \Box$ trivially holds since $\mu(\Box) = I_n$. Now, let $h \in \mathbb{N}_0$ and assume that the result holds whenever $\mathit{height}(c) \leqslant h$. Take any $c \in C_\Sigma$ such that $\mathit{height}(c) = h + 1$. Without loss of generality, we can assume that $c = \sigma(c_0, t_1, \ldots, t_{k-1})$ for some $k \geqslant 1$, $\sigma \in \Sigma_k$, $c_0 \in C_\Sigma$,

[2] We recall that the index $(2, 1)$ corresponds to the row $3 = 1 \cdot 2 + 1$ of matrix $\mu(+)$.

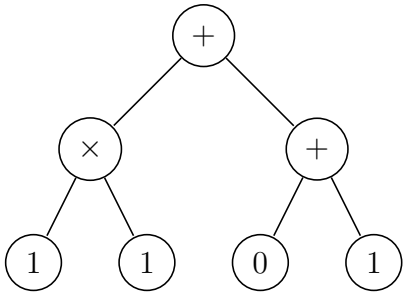
and $t_1, \ldots, t_{k-1} \in T_\Sigma$. For any $t \in T_\Sigma \cup C_\Sigma$, by the induction hypothesis and the mixed-product property of Kronecker product we have that

$$\begin{aligned}\mu(c[t]) &= (\mu(c_0[t]) \otimes \mu(t_1) \otimes \cdots \otimes \mu(t_{k-1})) \cdot \mu(\sigma) \\ &= ((\mu(t) \cdot \mu(c_0)) \otimes \mu(t_1) \otimes \cdots \otimes \mu(t_{k-1})) \cdot \mu(\sigma) \\ &= \mu(t) \cdot (\mu(c_0) \otimes \mu(t_1) \otimes \cdots \otimes \mu(t_{k-1})) \cdot \mu(\sigma) = \mu(t) \cdot \mu(c).\end{aligned}$$

This completes the induction step. □

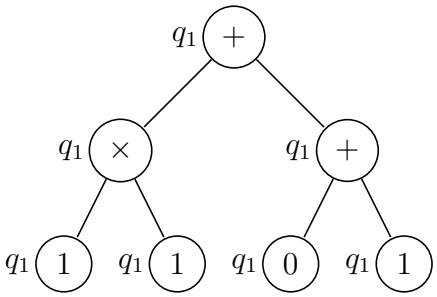
**Remark 3.** *MWAs can be seen as a special case of MTAs: An MWA $(n, \Sigma, \mu, \alpha, \gamma)$ "is" the MTA $(n, \Sigma \dot\cup \{\sigma_0\}, \mu, \gamma)$ where the symbols in $\Sigma$ are unary, symbol $\sigma_0$ is nullary, and $\mu(\sigma_0) = \alpha$. That is, we view $(\Sigma \dot\cup \{\sigma_0\})$-trees as words over $\Sigma$ by omitting the leaf symbol $\sigma_0$. Hence if a result holds for MTAs, it also holds for MWAs.*

**Example 2.** *Let us consider the computation of* $\mathbb{F}$-MTA $\mathcal{A} = (2, \Sigma, \mu, \gamma)$ *from Example 1 on the following $\Sigma$-tree $t$:*

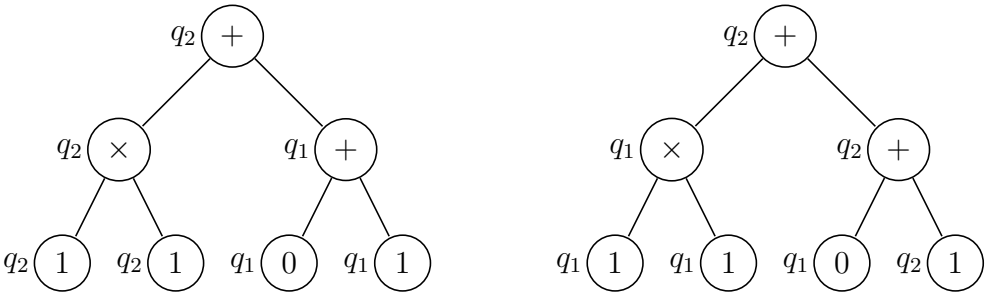



*The transition matrices define bottom-up runs of $\mathcal{A}$ on $t$. Intuitively, a run on $t$ corresponds to multiple copies of automaton $\mathcal{A}$ walking along $t$ from leaves to the root. Every such run has a weight in $\mathbb{F}$ which is defined as the product of the weights of all transitions taken.*

*On tree $t$, automaton $\mathcal{A}$ has one nonzero-weight run ending in state $q_1$, as follows:*




*Moreover, automaton $\mathcal{A}$ has two nonzero-weight runs ending in state $q_2$, as follows:*




*Each of the above three runs has weight* 1*. Therefore, the total weight of all runs of automaton $\mathcal{A}$ on tree $t$ in which the root is labelled $q_1$ is* 1*, and the total weight of all runs in which the root is labelled $q_2$ is* 2*. Indeed, algebraically, by definition of $\mu$ we have that*

$$
\begin{aligned}
\mu(t) &= (\mu(\times(1,1)) \otimes \mu(+(0,1))) \cdot \mu(+) \\
&= (((\mu(1) \otimes \mu(1)) \cdot \mu(\times)) \otimes ((\mu(0) \otimes \mu(1)) \cdot \mu(+))) \cdot \mu(+) \\
&= \left( \left( \begin{bmatrix} 1 & 1 & 1 & 1 \end{bmatrix} \cdot \begin{bmatrix} 1 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 1 \end{bmatrix} \right) \otimes \left( \begin{bmatrix} 1 & 1 & 0 & 0 \end{bmatrix} \cdot \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 0 & 1 \\ 0 & 0 \end{bmatrix} \right) \right) \cdot \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 0 & 1 \\ 0 & 0 \end{bmatrix} \\
&= \left( \begin{bmatrix} 1 & 1 \end{bmatrix} \otimes \begin{bmatrix} 1 & 1 \end{bmatrix} \right) \cdot \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 0 & 1 \\ 0 & 0 \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & 1 \end{bmatrix} \cdot \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 0 & 1 \\ 0 & 0 \end{bmatrix} = \begin{bmatrix} 1 & 2 \end{bmatrix}.
\end{aligned}
$$

*Finally, the weight $\|\mathcal{A}\|(t)$ of tree $t$ is the sum of the weights of all runs on $t$, where the weight of each run is multiplied by the final weight of its root label. Algebraically, we have*

$$\|\mathcal{A}\|(t) = \mu(t) \cdot \gamma = \begin{bmatrix} 1 & 2 \end{bmatrix} \cdot \begin{bmatrix} 0 & 1 \end{bmatrix}^{\top} = 2.$$

*Automaton $\mathcal{A}$ can be interpreted as evaluating the parse tree of an arithmetic computation. Formally, the weight $\|\mathcal{A}\|(t)$ is equal to the output of $t$, viewed as an arithmetic circuit. We will show later in Remark 12 that the same holds for any $\Sigma$-tree.*

We note that, given a general $\mathbb{Q}$-MTA $\mathcal{A} = (n, \Sigma, \mu, \gamma)$, the weight $\|\mathcal{A}\|(t)$ of a given tree $t \in T_\Sigma$ can be computed in polynomial time (in the Turing model) using the straightforward bottom-up algorithm following the definition of the function $\mu$.

### 2.3.4 Minimal Multiplicity Tree Automata

Two MTAs $\mathcal{A}_1$, $\mathcal{A}_2$ are said to be *equivalent* if $\|\mathcal{A}_1\| = \|\mathcal{A}_2\|$. An MTA is said to be *minimal* if no equivalent automaton has strictly smaller dimension.

A tree series $f$ is called *recognisable* if it is recognised by some MTA; such an automaton is called an *MTA-representation* of $f$. The set of all recognisable tree series in $\mathbb{F}\langle\langle T_\Sigma \rangle\rangle$ is denoted by $\mathrm{Rec}(\Sigma, \mathbb{F})$.

The following result was first shown by Bozapalidis and Louscou-Bozapalidou [1983]; an essentially equivalent result was later shown by Habrard and Oncina [2006].

**Theorem 4** (Bozapalidis and Louscou-Bozapalidou, 1983)**.** *Let $\Sigma$ be a ranked alphabet, $\mathbb{F}$ be a field, and $f \in \mathbb{F}\langle\langle T_\Sigma \rangle\rangle$. Let $H$ be the Hankel matrix of $f$. Then, $f \in Rec(\Sigma, \mathbb{F})$ if and only if $H$ has finite rank over $\mathbb{F}$. In case $f \in Rec(\Sigma, \mathbb{F})$, the dimension of a minimal MTA-representation of $f$ is* $\mathrm{rank}(H)$ *over $\mathbb{F}$.*

It follows from Theorem 4 that an $\mathbb{F}$-MTA $\mathcal{A}$ of dimension $n$ is minimal if and only if the Hankel matrix of $\|\mathcal{A}\|$ has rank $n$ over $\mathbb{F}$.

The following result by Bozapalidis and Alexandrakis [1989, Proposition 4] states that for any recognisable tree series, its minimal MTA-representation is unique up to change of basis.

**Theorem 5** (Bozapalidis and Alexandrakis, 1989)**.** *Let $\Sigma$ be a ranked alphabet, $\mathbb{F}$ be a field, and $f \in Rec(\Sigma, \mathbb{F})$. Let $r$ be the rank (over $\mathbb{F}$) of the Hankel matrix of $f$, and let $\mathcal{A}_1 = (r, \Sigma, \mu_1, \gamma_1)$ be an MTA-representation of $f$. Then, an $\mathbb{F}$-MTA $\mathcal{A}_2 = (r, \Sigma, \mu_2, \gamma_2)$ recognises $f$ if and only if there exists an invertible matrix $U \in \mathbb{F}^{r \times r}$ such that $\gamma_2 = U \cdot \gamma_1$ and $\mu_2(\sigma) = U^{\otimes \mathrm{rk}(\sigma)} \cdot \mu_1(\sigma) \cdot U^{-1}$ for every $\sigma \in \Sigma$.*

### 2.3.5 Product and Difference Automaton

In this subsection, we prove some closure properties for MTAs. First, we give two definitions: the product and the difference of two $\mathbb{F}$-MTAs. These concepts were introduced by Berstel and Reutenauer [1982] for linear representations of tree series, and by Borchardt [2004] for weighted tree automata with weights in a semiring.

Let $\mathcal{A}_1 = (n_1, \Sigma, \mu_1, \gamma_1)$ and $\mathcal{A}_2 = (n_2, \Sigma, \mu_2, \gamma_2)$ be two $\mathbb{F}$-multiplicity tree automata. The *difference* of $\mathcal{A}_1$ and $\mathcal{A}_2$, written as $\mathcal{A}_1 - \mathcal{A}_2$, is the $\mathbb{F}$-MTA $(n, \Sigma, \mu, \gamma)$ where:

- $n = n_1 + n_2$;
- For every $\sigma \in \Sigma$ and any $i \in [(n_1 + n_2)^{\mathrm{rk}(\sigma)}]$, $j \in [n_1 + n_2]$,

$$\mu(\sigma)_{i,j} = \begin{cases} \mu_1(\sigma)_{i,j} & \text{if } i \leq n_1^{\mathrm{rk}(\sigma)} \text{and } j \leq n_1 \\ \mu_2(\sigma)_{i,j} & \text{if } i > (n_1 + n_2)^{\mathrm{rk}(\sigma)} - n_2^{\mathrm{rk}(\sigma)} \text{and } j > n_1 \\ 0 & \text{otherwise;} \end{cases}$$

- $\gamma = \begin{bmatrix} \gamma_1 \\ -\gamma_2 \end{bmatrix}$.

The *product* of $\mathcal{A}_1$ by $\mathcal{A}_2$, written as $\mathcal{A}_1 \times \mathcal{A}_2$, is the $\mathbb{F}$-MTA $(n, \Sigma, \mu, \gamma)$ where:

- $n = n_1 \cdot n_2$;
- For every $\sigma \in \Sigma_k$, $\mu(\sigma) = P_k \cdot (\mu_1(\sigma) \otimes \mu_2(\sigma))$ where $P_k$ is a permutation matrix of order $(n_1 \cdot n_2)^k$ uniquely defined (see Remark 6 below) by

$$(u_1 \otimes \cdots \otimes u_k) \otimes (v_1 \otimes \cdots \otimes v_k) = ((u_1 \otimes v_1) \otimes \cdots \otimes (u_k \otimes v_k)) \cdot P_k \qquad (2.6)$$

  for all $u_1, \ldots, u_k \in \mathbb{F}^{1 \times n_1}$ and $v_1, \ldots, v_k \in \mathbb{F}^{1 \times n_2}$;
- $\gamma = \gamma_1 \otimes \gamma_2$.

**Remark 6.** *We argue that for every $k \in \mathbb{N}_0$ such that $k$ is the rank of a symbol in $\Sigma$, matrix $P_k$ is well-defined by Equation* (2.6)*. In order to do this, it suffices to show that $P_k$ is well-defined on a set of basis vectors of $\mathbb{F}^{1 \times n_1}$ and $\mathbb{F}^{1 \times n_2}$ and then extend linearly. To that end, let $(e^1_i)_{i \in [n_1]}$ and $(e^2_j)_{j \in [n_2]}$ be bases of $\mathbb{F}^{1 \times n_1}$ and $\mathbb{F}^{1 \times n_2}$, respectively. Let us define sets of vectors*

$$E_1 := \{(e^1_{i_1} \otimes \cdots \otimes e^1_{i_k}) \otimes (e^2_{j_1} \otimes \cdots \otimes e^2_{j_k}) : i_1, \ldots, i_k \in [n_1], j_1, \ldots, j_k \in [n_2]\}$$

*and*

$$E_2 := \{(e^1_{i_1} \otimes e^2_{j_1}) \otimes \cdots \otimes (e^1_{i_k} \otimes e^2_{j_k}) : i_1, \ldots, i_k \in [n_1], j_1, \ldots, j_k \in [n_2]\}.$$

*Then, $E_1$ and $E_2$ are two bases of the vector space $\mathbb{F}^{1 \times n_1 n_2}$. Therefore, $P_k$ is well-defined as an invertible matrix mapping basis $E_1$ to basis $E_2$.*

We now give the closure properties for MTAs in Proposition 7. Here we use the notion of the *Hadamard product* of two tree series $f_1, f_2 \in \mathbb{F}\langle\langle T_\Sigma \rangle\rangle$, which is defined as the tree series $f_1 \cdot f_2 \in \mathbb{F}\langle\langle T_\Sigma \rangle\rangle$ where for every $t \in T_\Sigma$, $(f_1 \cdot f_2)(t) = f_1(t) \cdot f_2(t)$.

**Proposition 7.** *Let $\mathcal{A}_1 = (n_1, \Sigma, \mu_1, \gamma_1)$ and $\mathcal{A}_2 = (n_2, \Sigma, \mu_2, \gamma_2)$ be two $\mathbb{F}$-MTAs. For their difference $\mathcal{A}_1 - \mathcal{A}_2$, it holds that $\|\mathcal{A}_1 - \mathcal{A}_2\| = \|\mathcal{A}_1\| - \|\mathcal{A}_2\|$. For their product $\mathcal{A}_1 \times \mathcal{A}_2 = (n, \Sigma, \mu, \gamma)$, the following properties hold:*

(i) *for every $t \in T_\Sigma$, $\mu(t) = \mu_1(t) \otimes \mu_2(t)$;*

(ii) *for every $c \in C_\Sigma$, $\mu(c) = \mu_1(c) \otimes \mu_2(c)$;*

(iii) $\|\mathcal{A}_1 \times \mathcal{A}_2\| = \|\mathcal{A}_1\| \cdot \|\mathcal{A}_2\|$.

*When $\mathbb{F} = \mathbb{Q}$, both automata $\mathcal{A}_1 - \mathcal{A}_2$ and $\mathcal{A}_1 \times \mathcal{A}_2$ can be computed from $\mathcal{A}_1$ and $\mathcal{A}_2$ in logarithmic space.*

*Proof.* The fact that $\|\mathcal{A}_1 - \mathcal{A}_2\| = \|\mathcal{A}_1\| - \|\mathcal{A}_2\|$ is shown by Berstel and Reutenauer [1982, Proposition 3.1] using the terminology of linear representations of tree series rather than multiplicity tree automata. Results (i) and (iii) for the product automaton are shown by Berstel and Reutenauer [1982, Proposition 5.1]; see also [Borchardt, 2004]. In the following we prove the remainder of the proposition.

We prove result (ii) using induction on the distance between the root and the $\Box$-labelled node of $c$. The base case is $c = \Box$. Here by definition we have that

$$\mu(c) = \mu(\Box) = I_{n_1 \cdot n_2} = I_{n_1} \otimes I_{n_2} = \mu_1(\Box) \otimes \mu_2(\Box) = \mu_1(c) \otimes \mu_2(c).$$

For the induction step, let $h \in \mathbb{N}_0$ and assume that (ii) holds for every context $c \in C_\Sigma^h$. Take any $c \in C_\Sigma^{h+1}$. Without loss of generality we can assume that $c = \sigma(c_1, t_2, \ldots, t_k)$ for some $k \geq 1$, $\sigma \in \Sigma_k$, $c_1 \in C_\Sigma^h$, and $t_2, \ldots, t_k \in T_\Sigma$. By the induction hypothesis, result (i), Equation (2.6), and the mixed-product property of Kronecker product, we now have

$$\mu(c) = \left( \mu(c_1) \otimes \bigotimes_{j=2}^{k} \mu(t_j) \right) \cdot \mu(\sigma)$$

$$
\begin{aligned}
&= \left( (\mu_1(c_1) \otimes \mu_2(c_1)) \otimes \bigotimes_{j=2}^{k} (\mu_1(t_j) \otimes \mu_2(t_j)) \right) \cdot P_k \cdot (\mu_1(\sigma) \otimes \mu_2(\sigma)) \\
&= \left( \left( \mu_1(c_1) \otimes \bigotimes_{j=2}^{k} \mu_1(t_j) \right) \otimes \left( \mu_2(c_1) \otimes \bigotimes_{j=2}^{k} \mu_2(t_j) \right) \right) \cdot (\mu_1(\sigma) \otimes \mu_2(\sigma)) \\
&= \left( \left( \mu_1(c_1) \otimes \bigotimes_{j=2}^{k} \mu_1(t_j) \right) \cdot \mu_1(\sigma) \right) \otimes \left( \left( \mu_2(c_1) \otimes \bigotimes_{j=2}^{k} \mu_2(t_j) \right) \cdot \mu_2(\sigma) \right) \\
&= \mu_1(c) \otimes \mu_2(c).
\end{aligned}
$$

This completes the proof of result (ii) by induction.

Now let $\mathbb{F} = \mathbb{Q}$. The $\mathbb{Q}$-MTA $\mathcal{A}_1 \times \mathcal{A}_2$ can be computed by a deterministic Turing machine which scans the transition matrices and the final weight vectors of $\mathcal{A}_1$ and $\mathcal{A}_2$, and then writes down the entries of the transition matrices and the final weight vector of their product $\mathcal{A}_1 \times \mathcal{A}_2$ onto the output tape. This computation requires maintaining only a constant number of pointers, which takes logarithmic space in the representation of automata $\mathcal{A}_1$ and $\mathcal{A}_2$. Hence, the Turing machine computing the automaton $\mathcal{A}_1 \times \mathcal{A}_2$ uses logarithmic space in the work tape. Analogously, the $\mathbb{Q}$-MTA $\mathcal{A}_1 - \mathcal{A}_2$ can be computed from $\mathcal{A}_1$ and $\mathcal{A}_2$ in logarithmic space. □

# Chapter 3

# Exact Learning of Multiplicity Tree Automata

**Abstract**

*In this chapter, we consider the query and computational complexity of learning multiplicity tree automata in the exact learning model. In this model, there is an oracle, called the Teacher, that can answer membership and equivalence queries posed by the Learner. Motivated by this feature, we first characterise the complexity of the equivalence problem for multiplicity tree automata, showing that it is logspace equivalent to polynomial identity testing. The latter is a very well studied problem, with a variety of randomized polynomial-time algorithms. The existence of a deterministic polynomial-time procedure for polynomial identity testing is a longstanding and prominent open problem.*

*We then move to query complexity, deriving lower bounds on the number of queries needed to learn multiplicity tree automata over both fixed and arbitrary fields. In the latter case, the bound is linear in the size of the target automaton. The best known upper bound on the query complexity*

*over arbitrary fields derives from an algorithm of Habrard and Oncina [2006], in which the number of queries is proportional to the size of the target automaton and the size of a largest counterexample, represented as a tree, that is returned by the Teacher. However, a smallest counterexample tree may already be exponential in the size of the target automaton. Thus the above algorithm has query complexity exponentially larger than our lower bound, and does not run in time polynomial in the size of the target automaton.*

*We give a new exact learning algorithm for multiplicity tree automata in which counterexamples to equivalence queries are represented as DAGs. As explained in this chapter, DAG counterexamples can be exponentially more succinct than tree counterexamples, and a smallest DAG counterexample is always linear in the size of the target automaton.*

*The query complexity of our learning algorithm is quadratic in the target automaton size and linear in the size of a largest counterexample. In particular, if the Teacher always returns DAG counterexamples of minimal size then the query complexity is quadratic in the target automaton size—almost matching the lower bound, and improving the best previously-known algorithm by an exponential factor.*

*This chapter is based on the material originally published as an extended abstract in [Marušić and Worrell, 2014] and a full version in [Marušić and Worrell, 2015].*

---

## 3.1 Introduction

Trees are a basic object in computer science and a natural model of hierarchical data, such as syntactic structures in natural language processing and XML data on the web. Trees arise across a broad range of applications, including natural text and speech processing, computer vision, bioinformatics, web information extraction, and social network analysis. Many of these applications require representing probability distributions over trees and more general functions from trees into the real numbers. A broad class of such functions can be defined by multiplicity tree automata, a powerful algebraic model (introduced in Section 2.3.3) which strictly generalises probabilistic tree automata, classical finite tree automata, and multiplicity word automata.

Multiplicity tree automata define many natural structural properties of trees and can be used to model probabilistic processes running on trees. Together with multiplicity word automata, they have been applied to a wide variety of machine learning problems, including speech recognition, image processing, character recognition, and grammatical inference [see Balle and Mohri, 2012].

The task of learning automata from examples and queries has been extensively studied since the 1960s. Two notable results in this domain show the impossibility of efficiently learning deterministic finite automata from positive and negative examples alone. First, Gold [1978] showed that the problem of exactly identifying the smallest deterministic finite automaton consistent with a set of accepted and rejected words is NP-hard. Later, Kearns and Valiant [1994] showed that the concept class of regular languages is not efficiently PAC learnable using any polynomially-evaluable hypothesis class under standard cryptographic assumptions.

A significant positive result on learning regular languages was achieved by Angluin [1987], who considered a Learner that did not just passively receive data but that was also able to ask queries. Specifically, Angluin considered *membership queries*, in which the Learner asks an oracle whether a given word belongs to the target

language, and *equivalence queries*, in which the Learner asks an oracle whether a hypothesis is correct, obtaining a counterexample if it is not. Subsequent research has sought to establish the learnability of many other hypothesis classes in the same setting, including classes of Boolean formulae, decision trees, context-free languages, and polynomials; see the book of Kearns and Vazirani [1994, Chapter 8] for more details and references.

In this chapter, we study the problem of learning multiplicity tree automata in the *exact learning model* of Angluin [1988], outlined above. Formally, in this model a Learner actively collects information about the target function from a Teacher through *membership queries*, which ask for the value of the function on a specific input, and *equivalence queries*, which suggest a hypothesis to which the Teacher provides a counterexample if one exists. A class of functions $\mathscr{C}$ is *exactly learnable* if there exists an exact learning algorithm such that for any function $f \in \mathscr{C}$, the Learner identifies $f$ using polynomially many membership and equivalence queries in the size of a shortest representation of $f$ and the size of a largest counterexample returned by the Teacher during the execution of the algorithm. The exact learning model is an important theoretical model of the learning process. It is well known that learnability in the exact learning model also implies learnability in the PAC model with membership queries [Mohri et al., 2012, Theorem 13.3].

We are interested in questions of succinctness and computational efficiency, both from the point of view of the Teacher and the Learner. From the point of view of the Teacher, one of the main questions is checking *equivalence* of multiplicity tree automata, i.e., whether two multiplicity tree automata define the same function on trees. Seidl [1990] proved that equivalence of multiplicity tree automata is decidable in polynomial time assuming unit-cost arithmetic, and in randomized polynomial time in the usual bit-cost model. No finer analysis of the complexity of this problem exists to date. In contrast, the complexity of equivalence for classical nondeterministic

word and tree automata has been completely characterised: PSPACE-complete over words [Aho et al., 1974] and EXPTIME-complete over trees [Seidl, 1990].

Our first contribution (Section 3.4) is to show that the equivalence problem for multiplicity tree automata is logspace equivalent to polynomial identity testing, i.e., the problem of deciding whether a polynomial given as an arithmetic circuit is zero. The latter problem is very well studied, with a variety of randomized polynomial-time algorithms [DeMillo and Lipton, 1978, Schwartz, 1980, Zippel, 1979], but, as yet, no deterministic polynomial-time procedure [see Arora and Barak, 2009].

We note that, even though multiplicity tree automata augment classical nondeterministic tree automata, their semantics are different. Intuitively, equivalence for multiplicity tree automata is a stronger notion because one needs to consider the weights of all runs on a given tree. Therefore, it is not surprising that the equivalence problems for multiplicity tree automata and nondeterministic tree (and word) automata have different complexities.

Our second contribution (Section 3.6) is to give lower bounds on the number of queries needed to learn multiplicity tree automata in the exact learning model, both for the case of an arbitrary and a fixed underlying field. The bound in the former case is linear in the automaton size. In the latter case, the bound is linear in the automaton size for alphabets of a fixed maximal rank. To the best of our knowledge, these are the first lower bounds on the query complexity of exactly learning multiplicity tree automata.

Habrard and Oncina [2006] give an algorithm for learning multiplicity tree automata in the exact learning model. Consider a target multiplicity tree automaton whose minimal representation $\mathcal{A}$ has $n$ states. The algorithm of Habrard and Oncina [2006] makes at most $n$ equivalence queries and number of membership queries proportional to $|\mathcal{A}| \cdot s$, where $|\mathcal{A}|$ is the size of $\mathcal{A}$ and $s$ is the size of a largest counterexample returned by the Teacher. Since this algorithm assumes that the Teacher

returns counterexamples represented explicitly as trees, $s$ can be exponential in $|\mathcal{A}|$, even for a Teacher that returns counterexamples of minimal size (see Example 3). This observation reveals an exponential gap between the query complexity of the algorithm of Habrard and Oncina [2006] and our above-mentioned lower bound, which is only linear in $|\mathcal{A}|$. Another consequence is that the worst-case time complexity of this algorithm is exponential in the size of the target automaton.

Given two inequivalent multiplicity tree automata with $n$ states in total, the algorithm of Seidl [1990] produces a subtree-closed set of trees of cardinality at most $n$ that contains a tree on which the automata differ. It follows that the counterexample contained in this set has at most $n$ subtrees, and hence can be represented as a DAG with at most $n$ vertices (see §3.4.2). Thus in the context of exact learning it is natural to consider a Teacher that can return succinctly-represented counterexamples, i.e., trees represented as DAGs.

DAGs have been used as succinct representations of trees in a number of domains, including classification problems [Sperduti and Starita, 1997] and query evaluation for XML [Buneman et al., 2003, Frick et al., 2003]. Tree automata that run on DAG representations of finite trees were first introduced by Charatonik [1999] as extensions of ordinary tree automata, and were further studied by Anantharaman et al. [2005]. The automata considered by Charatonik [1999] and Anantharaman et al. [2005] run on fully-compressed DAGs. Fila and Anantharaman [2006] extend this definition by introducing tree automata that run on DAGs that may be partially compressed. In this chapter, we employ the latter framework in the context of learning multiplicity automata.

In Section 3.5, we present a new exact learning algorithm for multiplicity tree automata that achieves the same bound on the number of equivalence queries as the algorithm of Habrard and Oncina [2006], while using number of membership queries quadratic in the target automaton size and linear in the largest counterexample size,

even when counterexamples are given as DAGs. Assuming that the Teacher provides minimal DAG representations of counterexamples, our algorithm therefore makes quadratically many queries in the target automaton size. This is exponentially fewer queries than the best previously-known algorithm [Habrard and Oncina, 2006] and quadratic in the above-mentioned lower bound. Furthermore, our algorithm performs a quadratic number of arithmetic operations in the size of the target automaton, and can be implemented in randomized polynomial time in the Turing model.

Like the algorithm of Habrard and Oncina [2006], our algorithm constructs a matricial representation of the target automaton: its Hankel matrix (introduced in §2.3.1). However on receiving a counterexample tree $z$, the former algorithm adds a new column to the Hankel matrix for every suffix of $z$, while our algorithm adds (at most) one new row for each subtree of $z$. Crucially the number of suffixes may be exponential in the size of a DAG representation of $z$, whereas the number of subtrees is only linear in the size of a DAG representation.

## 3.2 Related Work

One of the earliest results about the exact learning model was the proof of Angluin [1987] that deterministic finite automata are learnable. This result was generalised by Drewes and Högberg [2007] to show exact learnability of deterministic finite (bottom-up) tree automata, generalising also a result of Sakakibara [1990] on the exact learnability of context-free grammars from their structural descriptions[1].

The learning algorithm of Drewes and Högberg [2007] was generalised by Maletti [2007] to show that deterministic weighted tree automata over a (commutative) semifield are exactly learnable, generalising also an earlier result of Drewes and Vogler [2007] which was restricted to the class of deterministic *all-accepting* (i.e., every final weight is nonzero) weighted tree automata. More recently, a unifying framework

[1] *Structural descriptions* of a context-free grammar are unlabelled derivation trees of the grammar.

for exact learning of deterministic weighted tree automata over a semifield has been proposed by Drewes et al. [2011]. Specifically, the latter paper introduces the notion of *abstract observation tables*, an abstract data type for learning deterministic weighted tree automata in the exact learning model, and show that every correct implementation of abstract observation tables yields a correct learning algorithm.

Exact learnability of nondeterministic weighted automata over a field (here called multiplicity automata) has also been extensively studied. Beimel et al. [2000] show that multiplicity word automata can be learned efficiently, and apply this to learn various classes of DNF formulae and polynomials. These results were generalised by Klivans and Shpilka [2006] to show exact learnability of restricted algebraic branching programs and noncommutative set-multilinear arithmetic formulae. Bisht et al. [2006] give an almost tight (up to a *log* factor) lower bound on the number of queries made by any exact learning algorithm for the class of multiplicity word automata.

An exact learning algorithm for a class of nondeterministic tree automata, namely *residual finite* tree automata, is given by Kasprzik [2013]. The latter paper identifies the size of counterexamples as a hidden exponential factor in the complexity of the learning algorithm, observing in particular that a smallest counterexample can have exponential size in the number of states of the target automaton. Such a phenomenon does not prevent the class of tree automata from being exactly learnable since in the exact learning model the complexity measure takes into account the size of a largest counterexample. However, this does raise the question of developing a learning algorithm whose complexity would be polynomial in the size of succinctly-represented counterexamples, which is one of the motivations for the present work.

Denis and Habrard [2007] consider the problem of learning probability distributions over trees that are recognised by a multiplicity tree automaton from samples drawn independently according to the target distribution. They give an inference algorithm that exactly identifies such recognisable probability distributions in the limit

with probability one (with respect to the randomly-drawn examples). Here the target distribution is specified by a multiplicity tree automaton that recognises it. Most closely related to the topic of this chapter is the work of Habrard and Oncina [2006], who give an algorithm for learning multiplicity tree automata in the exact learning model, as discussed above.

A variety of spectral methods have been employed for learning multiplicity word and tree automata [Bailly et al., 2009, Balle and Mohri, 2012, Denis et al., 2014, Gybels et al., 2014]. This line of research originates in earlier work of Hsu et al. [2012] that gives a spectral learning algorithm (based on singular value decomposition) for hidden Markov models. Particularly close to the present chapter is the work of Bailly et al. [2010], which learns probability distributions over trees that are recognised by some multiplicity tree automaton. Their approach lies within a passive learning framework in which one is given a sample of trees independently drawn according to a target distribution, and the aim is to infer a multiplicity tree automaton that approximates the target. As in our approach, the notion of a Hankel matrix plays a central role in the algorithm of Bailly et al. [2010]. There the Hankel matrix is called an *observation matrix*, and it encodes an empirical distribution on trees obtained by sampling from the target distribution. Bailly et al. [2010] apply principal component analysis in order to identify a low-dimensional approximation of the vector space spanned by the residuals of the target probability distribution. From this approximation they build an automaton whose associated tree series approximates the target distribution. They moreover obtain bounds on the estimation error of the output tree series with respect to the target distribution in terms of the sample size and the desired confidence.

In contrast to the above-described approach of Bailly et al. [2010], in our work the target dimension (i.e., number of states) is not part of the input since our aim is to learn a *minimal* multiplicity tree automaton that exactly represents the target

tree series. Moreover, in this chapter the entries of the Hankel matrix are determined by active queries rather than passive observations, and the learning process continues until we know a sufficient number of entries to be able to exactly construct a representation of the target.

## 3.3 Preliminaries

In this section, we present some background material for this chapter.

In Section 3.3.1 we introduce directed acyclic graphs (DAGs) as (succinct) representations of trees. In Section 3.3.2, we introduce the notion of a multiplicity tree automaton running on DAGs. To the best of our knowledge, this notion has not been studied before.

In Section 3.3.3 we introduce arithmetic circuits. Lastly, in Section 3.3.4 we introduce the exact learning model.

### 3.3.1 DAG Representations of Trees

Given a set $V$, we denote by $V^*$ the set of all finite ordered tuples of elements from $V$.

A *directed multigraph* consists of a set of nodes $V$ and a multiset of directed edges $E \subseteq V \times V$. We say that a directed multigraph is *acyclic* if the underlying directed graph has no cycles; we say it is *ordered* if a linear order on the successors of each node is assumed. A directed multigraph is *rooted* if there is a distinguished *root* node $v$ such that all other nodes are reachable from $v$.

Let $\Sigma$ be a ranked alphabet. A *directed acyclic graph (DAG) representation of a $\Sigma$-tree* ($\Sigma$-*DAG* or *DAG* for short) is a rooted acyclic ordered directed multigraph whose nodes are labelled with symbols from $\Sigma$ such that the outdegree of each node is equal to the rank of the symbol it is labelled with. Formally a $\Sigma$-DAG consists of a set of nodes $V$, for each node $v \in V$ a list of successors $succ(v) \in V^*$, and a node

labelling $\lambda : V \to \Sigma$ where for each node $v \in V$ it holds that $\lambda(v) \in \Sigma_{|succ(v)|}$. Note that $\Sigma$-trees are a subclass of $\Sigma$-DAGs.

Let $G$ be a $\Sigma$-DAG. The *size* of $G$, denoted by $size(G)$, is the number of nodes in $G$. The *height* of $G$, denoted by $height(G)$, is the length of a longest directed path in $G$. For any node $v$ in $G$, the *sub-DAG* of $G$ *rooted at* $v$, denoted by $G|_v$, is the $\Sigma$-DAG consisting of the node $v$ and all of its descendants in $G$. Clearly, if $v$ is the root of $G$ then $G|_v = G$. The set $\{G|_v : v \text{ is a node in } G\}$ of all the sub-DAGs of $G$ is denoted by $Sub(G)$.

For any $\Sigma$-DAG $G$, we define its *unfolding* into a $\Sigma$-tree, denoted by $unfold(G)$, inductively as follows: If the root of $G$ is labelled with a symbol $\sigma$ and has the list of successors $v_1, \ldots, v_k$, then

$$unfold(G) = \sigma(unfold(G|_{v_1}), \ldots, unfold(G|_{v_k})).$$

The next proposition follows easily from the definition.

**Proposition 8.** *If $G$ is a $\Sigma$-DAG, then $Sub(unfold(G)) = unfold(Sub(G))$.*

Because a context has exactly one occurrence of symbol $\Box$, any *DAG representation of a $\Sigma$-context* is a $(\{\Box\} \cup \Sigma)$-DAG that has a unique path from the root to the (unique) $\Box$-labelled node. The *concatenation* of a DAG $K$, representing a $\Sigma$-context, and a $\Sigma$-DAG $G$ is the $\Sigma$-DAG, denoted by $K[G]$, obtained by substituting the root of $G$ for $\Box$ in $K$.

**Proposition 9.** *Let $K$ be a DAG representation of a $\Sigma$-context, and let $G$ be a $\Sigma$-DAG. Then, $unfold(K[G]) = unfold(K)(unfold(G))$.*

*Proof.* The proof is by induction on $height(K)$. For the base case, let $height(K) = 0$. Then, we have that $K = \Box$ and therefore

$$unfold(\Box[G]) = unfold(G) = unfold(\Box)(unfold(G))$$

for any $\Sigma$-DAG $G$.

For the induction step, let $h \in \mathbb{N}_0$ and assume that the result holds if $\mathit{height}(K) \leq h$. Let $K$ be a DAG representation of a $\Sigma$-context such that $\mathit{height}(K) = h + 1$. Let the root of $K$ have label $\sigma$ and list of successors $v_1, \ldots, v_k$. By definition, there is a unique path in $K$ going from the root to the $\Box$-labelled node. Without loss of generality, we can assume that the $\Box$-labelled node is a successor of $v_1$. Take an arbitrary $\Sigma$-DAG $G$. Since $\mathit{height}(K|_{v_1}) \leq h$, we have by the induction hypothesis that

$$\begin{aligned}
\mathit{unfold}(K[G]) &= \sigma(\mathit{unfold}(K|_{v_1}[G]), \mathit{unfold}(K|_{v_2}), \ldots, \mathit{unfold}(K|_{v_k})) \\
&= \sigma(\mathit{unfold}(K|_{v_1})(\mathit{unfold}(G)), \mathit{unfold}(K|_{v_2}), \ldots, \mathit{unfold}(K|_{v_k})) \\
&= \sigma(\mathit{unfold}(K|_{v_1}), \mathit{unfold}(K|_{v_2}), \ldots, \mathit{unfold}(K|_{v_k}))(\mathit{unfold}(G)) \\
&= \mathit{unfold}(K)(\mathit{unfold}(G)).
\end{aligned}$$

This completes the proof by induction. □

### 3.3.2 Multiplicity Tree Automata on DAGs

Let $\mathbb{F}$ be a field, and $\mathcal{A} = (n, \Sigma, \mu, \gamma)$ be an $\mathbb{F}$-multiplicity tree automaton. The computation of automaton $\mathcal{A}$ on a $\Sigma$-DAG $G = (V, E)$ is defined as follows: A *run* of $\mathcal{A}$ on $G$ is a mapping $\rho : \mathit{Sub}(G) \to \mathbb{F}^n$ such that for every node $v \in V$, if $v$ is labelled with $\sigma$ and has the list of successors $\mathit{succ}(v) = v_1, \ldots, v_k$ then

$$\rho(G|_v) = (\rho(G|_{v_1}) \otimes \cdots \otimes \rho(G|_{v_k})) \cdot \mu(\sigma).$$

Automaton $\mathcal{A}$ assigns to $G$ a *weight* $\|\mathcal{A}\|(G) \in \mathbb{F}$ where $\|\mathcal{A}\|(G) = \rho(G) \cdot \gamma$.

In the following proposition, we show that the weight assigned by a multiplicity tree automaton to a DAG is equal to the weight assigned to its tree unfolding.

**Proposition 10.** *Let $\mathbb{F}$ be a field, and $\mathcal{A} = (n, \Sigma, \mu, \gamma)$ be an $\mathbb{F}$-multiplicity tree automaton. For any $\Sigma$-DAG $G$, it holds that $\rho(G) = \mu(\mathit{unfold}(G))$ and $\|\mathcal{A}\|(G) = \|\mathcal{A}\|(\mathit{unfold}(G))$.*

*Proof.* Let $V$ be the set of nodes of $G$. First we show that for every $v \in V$,

$$\rho(G|_v) = \mu(\mathit{unfold}(G|_v)). \tag{3.1}$$

The proof is by induction on $\mathit{height}(G|_v)$. For the base case, let $\mathit{height}(G|_v) = 0$. This implies that $G|_v = \sigma \in \Sigma_0$. Therefore, by definition we have that

$$\rho(G|_v) = \mu(\sigma) = \mu(\mathit{unfold}(\sigma)) = \mu(\mathit{unfold}(G|_v)).$$

For the induction step, let $h \in \mathbb{N}_0$ and assume that Equation (3.1) holds for every $v \in V$ such that $\mathit{height}(G|_v) \leq h$. Take any $v \in V$ such that $\mathit{height}(G|_v) = h + 1$. Let the root of $G|_v$ be labelled with a symbol $\sigma$ and have list of successors $\mathit{succ}(v) = v_1, \ldots, v_k$. Then for every $j \in [k]$, we have that $\mathit{height}(G|_{v_j}) \leq h$ and thus $\rho(G|_{v_j}) = \mu(\mathit{unfold}(G|_{v_j}))$ holds by the induction hypothesis. This implies that

$$\begin{aligned}
\rho(G|_v) &= (\rho(G|_{v_1}) \otimes \cdots \otimes \rho(G|_{v_k})) \cdot \mu(\sigma) \\
&= (\mu(\mathit{unfold}(G|_{v_1})) \otimes \cdots \otimes \mu(\mathit{unfold}(G|_{v_k}))) \cdot \mu(\sigma) \\
&= \mu(\sigma(\mathit{unfold}(G|_{v_1}), \ldots, \mathit{unfold}(G|_{v_k}))) \\
&= \mu(\mathit{unfold}(G|_v)),
\end{aligned}$$

which completes the proof of Equation (3.1) for all $v \in V$ by induction.

Taking $v$ to be the root of $G$, we have by Equation (3.1) that $\rho(G) = \mu(\mathit{unfold}(G))$. Therefore, $\|\mathcal{A}\|(G) = \rho(G) \cdot \gamma = \mu(\mathit{unfold}(G)) \cdot \gamma = \|\mathcal{A}\|(\mathit{unfold}(G))$. □

In the following we give an example of two inequivalent multiplicity tree automata

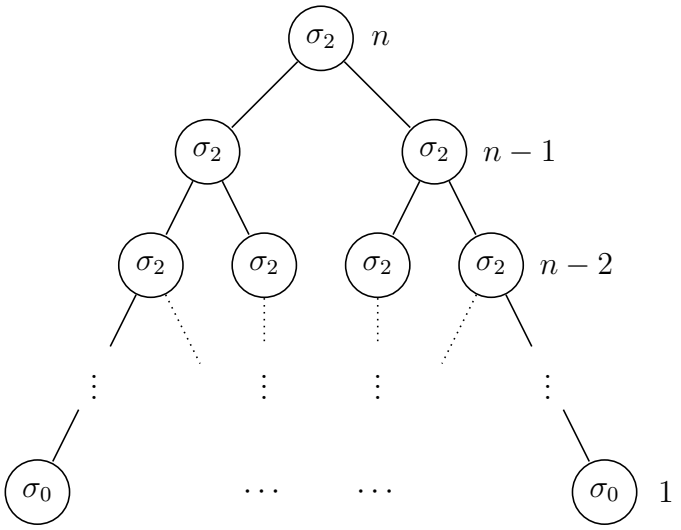



Figure 3.1: Tree $t_n$

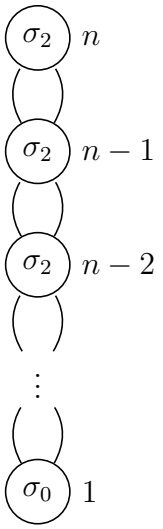



Figure 3.2: DAG $G_n$

such that the smallest tree that is counterexample to their equivalence has size exponential in their total number of states. Moreover, we show that this tree has an exponentially more succinct DAG representation.

**Example 3.** *Let* $\Sigma = \{\sigma_0, \sigma_2\}$ *be a ranked alphabet such that* $\mathrm{rk}(\sigma_0) = 0$ *and* $\mathrm{rk}(\sigma_2) = 2$. *Take any* $n \in \mathbb{N}$. *Let* $t_n$, *depicted in Figure 3.1, be the perfect binary* $\Sigma$*-tree of height* $n-1$. *Note that* $\mathit{size}(t_n) = O(2^n)$.

*Define an* $\mathbb{F}$*-MTA* $\mathcal{A} = (n, \Sigma, \mu, e_1^\top)$ *such that* $\mu(\sigma_0) = e_n \in \mathbb{F}^{1\times n}$ *and* $\mu(\sigma_2) \in \mathbb{F}^{n^2\times n}$ *where* $\mu(\sigma_2)_{(i+1,i+1),i} = 1$ *for every* $i \in [n-1]$, *and all other entries of* $\mu(\sigma_2)$ *are zero. It can be checked, using the definition of the automaton* $\mathcal{A}$, *that* $\|\mathcal{A}\|(t_n) = 1$ *and* $\|\mathcal{A}\|(t) = 0$ *for every* $t \in T_\Sigma \setminus \{t_n\}$.

*Let* $\mathcal{B}$ *be the* 0*-dimensional* $\mathbb{F}$*-MTA over* $\Sigma$ *(so that* $\|\mathcal{B}\| \equiv 0$*). Suppose we were to check whether automata* $\mathcal{A}$ *and* $\mathcal{B}$ *are equivalent. Then the only counterexample to their equivalence, namely the tree* $t_n$, *has size* $O(2^n)$. *Note, however, that* $t_n$ *has an exponentially more succinct DAG representation* $G_n$, *given in Figure 3.2.*

### 3.3.3 Arithmetic Circuits

An *arithmetic circuit* is a finite acyclic vertex-labelled directed multigraph whose vertices, called *gates*, have indegree 0 or 2. Vertices of indegree 0 are called *input gates* and are labelled with a nonnegative integer or a variable from the set $\{x_i : i \in \mathbb{N}\}$. Vertices of indegree 2 are called *internal gates* and are labelled with an arithmetic operation $+$, $\times$, or $-$. We assume that there is a unique gate with outdegree 0 called the *output gate*. An arithmetic circuit is called *variable-free* if all input gates are labelled with a nonnegative integer.

Given two gates $u$ and $v$ of an arithmetic circuit $C$, we call $u$ a *child* of $v$ if $(u, v)$ is a directed edge in $C$. The *size* of $C$ is the number of gates in $C$. The *height* of a gate $v$ in $C$, written as $height(v)$, is the length of a longest directed path from an input gate to $v$. The *height* of $C$ is the maximal height of a gate in $C$. That is, the height of $C$ is the height of the output gate of $C$.

An arithmetic circuit $C$ computes a polynomial over the integers as follows: An input gate of $C$ labelled with $\alpha \in \mathbb{N}_0 \cup \{x_i : i \in \mathbb{N}\}$ computes the polynomial $\alpha$. An internal gate of $C$ labelled with $* \in \{+, \times, -\}$ computes the polynomial $p_1 * p_2$ where $p_1$ and $p_2$ are the polynomials computed by its children. For any gate $v$ in $C$, we write $f_v$ for the polynomial computed by $v$. The *output* of $C$, written as $f_C$, is the polynomial computed by the output gate of $C$. The *arithmetic circuit identity testing (*ACIT*)* problem asks whether the output of a given arithmetic circuit is equal to the zero polynomial.

**Remark 11.** *Without loss of generality we can assume that every integer-labelled input gate of an arithmetic circuit is labelled with* 0 *or* 1*. Indeed, any other integer label given in binary can be encoded as an arithmetic circuit whose input gates are labelled with* 0 *or* 1*.*

**Remark 12.** *Any variable-free arithmetic circuit $C$ can be seen as a $\Sigma$-DAG with*

$\Sigma = \{0, 1, +, \times, -\}$ *where* $0, 1$ *are nullary symbols and* $+, \times, -$ *are binary symbols. Consequently, any multiplicity tree automaton over the ranked alphabet* $\Sigma$ *can be evaluated on* $C$*.*

*For instance, let* $\mathcal{A} = (2, \Sigma, \mu, \gamma)$ *be the MTA from Example 1. Then, for any gate* $v$ *in* $C$ *it holds that* $\mu(C|_v) = (1 \quad f_v)$*, where* $C|_v$ *is the sub-DAG of* $C$ *rooted at* $v$ *and* $f_v$ *is the number computed at gate* $v$*. (This can be easily proved using induction on height*$(C|_v)$*.) In particular, when* $v$ *is the output gate of* $C$ *we get that*

$$\|\mathcal{A}\|(C) = \mu(C) \cdot \gamma = \begin{pmatrix} 1 & f_C \end{pmatrix} \cdot \begin{pmatrix} 0 & 1 \end{pmatrix}^\top = f_C.$$

*That is, the weight of* $\mathcal{A}$ *on* $C$ *(viewed as a DAG) is equal to the output of* $C$ *(viewed as an arithmetic circuit). In other words, automaton* $\mathcal{A}$ *evaluates the circuit* $C$*.*

### 3.3.4 Exact Learning Model

In this subsection, we define the *exact learning model* of Angluin [1988]: Let $f$ be a *target function*. A *Learner* (*learning algorithm*) may, in each step, propose a hypothesis function $h$ by making an *equivalence query* to a *Teacher*. If $h$ is equivalent to $f$, then the Teacher returns YES and the Learner succeeds and halts. Otherwise, the Teacher returns NO with a *counterexample*, which is an assignment $x$ such that $h(x) \neq f(x)$. Moreover, the Learner may query the Teacher for the value of the function $f$ on a particular assignment $x$ by making a *membership query* on $x$. The Teacher returns the value $f(x)$ to such a query.

We say that a class of functions $\mathscr{C}$ is *exactly learnable* if there is a Learner that for any target function $f \in \mathscr{C}$, outputs a hypothesis $h \in \mathscr{C}$ such that $h(x) = f(x)$ for all assignments $x$, and does so in time polynomial in the size of a shortest representation of $f$ and the size of a largest counterexample returned by the Teacher. We moreover say that the class $\mathscr{C}$ is *exactly learnable in (randomized) polynomial time* if

the learning algorithm can be implemented to run in (randomized) polynomial time in the Turing model.

## 3.4 Equivalence Queries

In the exact learning model, one of the principal algorithmic questions from the point of view of the Teacher is the computational complexity of equivalence testing. In this section, we characterise the computational complexity of equivalence testing for multiplicity tree automata, showing that this problem is logspace equivalent to polynomial identity testing. The latter is a well-studied problem for which there are numerous randomized polynomial-time algorithms, with the existence of a deterministic polynomial-time algorithm being a longstanding open problem. Moreover in this section, we explain why it is natural to expect the Teacher to return succinct DAG counterexamples in the case of inequivalence.

### 3.4.1 Computational Complexity of MTA Equivalence

A key algorithmic component of the exact learning framework is checking the equivalence of the hypothesis and the target function: a task for the Teacher rather than the Learner. The existence of efficient algorithms to perform such equivalence checks is important for several applications of the exact learning framework [see, e.g., Feng et al., 2011]. With this in mind, in this subsection we characterise the computational complexity of the equivalence problem for $\mathbb{Q}$-multiplicity tree automata. Here we specialise the weight field to be $\mathbb{Q}$ since we want to work within the classical Turing model of computation. Parts of this subsection also exploit the fact that $\mathbb{Q}$ is an ordered field. Our main result is:

**Theorem 13.** *The equivalence problem for $\mathbb{Q}$-multiplicity tree automata is logspace interreducible with* ACIT.

A version of this result appeared in the author's MSc thesis. A related result, characterising equivalence of probabilistic visibly pushdown automata on words in terms of polynomial identity testing, was shown by Kiefer et al. [2013].

On several occasions in this section, we will implicitly make use of the fact that a composition of two logspace reductions is again a logspace reduction [Arora and Barak, 2009, Lemma 4.17].

### From MTA Equivalence to ACIT

First, we present a logspace reduction from the equivalence problem for $\mathbb{Q}$-MTAs to ACIT. We start with the following lemma.

**Lemma 14.** *Given an integer $n \in \mathbb{N}$ and a $\mathbb{Q}$-multiplicity tree automaton $\mathcal{A}$ over a ranked alphabet $\Sigma$, one can compute, in logarithmic space in $|\mathcal{A}|$ and $n$, a variable-free arithmetic circuit that has output $\sum_{t \in T_\Sigma^{\leq n}} \|\mathcal{A}\|(t)$.*

*Proof.* Let $\mathcal{A} = (r, \Sigma, \mu, \gamma)$, and let $m$ be the rank of $\Sigma$. By definition, it holds that

$$\sum_{t \in T_\Sigma^{\leq n}} \|\mathcal{A}\|(t) = \left( \sum_{t \in T_\Sigma^{\leq n}} \mu(t) \right) \cdot \gamma. \tag{3.2}$$

We have $\sum_{t \in T_\Sigma^{\leq 1}} \mu(t) = \sum_{\sigma \in \Sigma_0} \mu(\sigma)$. Furthermore for every $i \in \mathbb{N}$, it holds that

$$T_\Sigma^{\leq i+1} = \{\sigma(t_1, \ldots, t_k) : k \in \{0, \ldots, m\}, \sigma \in \Sigma_k, t_1, \ldots, t_k \in T_\Sigma^{\leq i}\}$$

and thus by bilinearity of Kronecker product,

$$\begin{aligned}
\sum_{t \in T_\Sigma^{\leq i+1}} \mu(t) &= \sum_{k=0}^{m} \sum_{\sigma \in \Sigma_k} \sum_{t_1 \in T_\Sigma^{\leq i}} \cdots \sum_{t_k \in T_\Sigma^{\leq i}} (\mu(t_1) \otimes \cdots \otimes \mu(t_k)) \cdot \mu(\sigma) \\
&= \sum_{k=0}^{m} \sum_{\sigma \in \Sigma_k} \left( \left( \sum_{t_1 \in T_\Sigma^{\leq i}} \mu(t_1) \right) \otimes \cdots \otimes \left( \sum_{t_k \in T_\Sigma^{\leq i}} \mu(t_k) \right) \right) \cdot \mu(\sigma)
\end{aligned}$$

$$= \sum_{k=0}^{m} \left( \sum_{t \in T_\Sigma^{\leq i}} \mu(t) \right)^{\otimes k} \sum_{\sigma \in \Sigma_k} \mu(\sigma). \tag{3.3}$$

Next, we define a variable-free arithmetic circuit $\Phi$ that has output $\sum_{t \in T_\Sigma^{\leq n}} \|\mathcal{A}\|(t)$. First, let us denote $G(i) := \sum_{t \in T_\Sigma^{\leq i}} \mu(t)$ for every $i \in \mathbb{N}$. Then by Equation (3.3),

$$G(i+1) = \sum_{k=0}^{m} G(i)^{\otimes k} \cdot S(k)$$

where $S(k) := \sum_{\sigma \in \Sigma_k} \mu(\sigma)$ for every $k \in \{0, \ldots, m\}$. In coordinate notation, for every $j \in [r]$ we have by Equation (2.3) that

$$G(i+1)_j = \sum_{k=0}^{m} \sum_{(l_1,\ldots,l_k) \in [r]^k} \prod_{a=1}^{k} G(i)_{l_a} \cdot S(k)_{(l_1,\ldots,l_k),j}. \tag{3.4}$$

We present $\Phi$ as a straight-line program, with built-in constants

$$\{\mu^{\sigma}_{(l_1,\ldots,l_k),j}, \gamma_j : k \in \{0, \ldots, m\}, \sigma \in \Sigma_k, (l_1, \ldots, l_k) \in [r]^k, j \in [r]\}$$

representing the entries of the transition matrices and the final weight vector of $\mathcal{A}$, internal variables $\{s^k_{(l_1,\ldots,l_k),j} : k \in \{0, \ldots, m\}, (l_1, \ldots, l_k) \in [r]^k, j \in [r]\}$ and $\{g_{i,j} : i \in [n], j \in [r]\}$ evaluating the entries of matrices $S(k)$ and vectors $G(i)$ respectively, and the final internal variable $f$ computing the value of $\Phi$.

Formally, the straight-line program $\Phi$ is given in Table 3.1. Here the statements are given in indexed-sum and indexed-product notation, which can easily be expanded in terms of the corresponding binary operations. It follows from Equations (3.2) and (3.4) that the output of $\Phi$ is $G(n) \cdot \gamma = \sum_{t \in T_\Sigma^{\leq n}} \|\mathcal{A}\|(t)$.

The input gates of $\Phi$ are labelled with rational numbers. However, by separately encoding numerators and denominators, we can in logarithmic space reduce $\Phi$ to an arithmetic circuit where all input gates are labelled with integers.

1. For $j \in [r]$ do $g_{1,j} \leftarrow \sum_{\sigma \in \Sigma_0} \mu_j^\sigma$

2. For $k \in \{0, \ldots, m\}$, $(l_1, \ldots, l_k) \in [r]^k$, $j \in [r]$ do $s^k_{(l_1,\ldots,l_k),j} \leftarrow \sum_{\sigma \in \Sigma_k} \mu^\sigma_{(l_1,\ldots,l_k),j}$

3. For $i = 1$ to $n - 1$ do

   3.1. For $k \in \{0, \ldots, m\}$, $(l_1, \ldots, l_k) \in [r]^k$, $j \in [r]$ do

   $$h^{i,k}_{(l_1,\ldots,l_k),j} \leftarrow \prod_{a=1}^{k} g_{i,l_a} \cdot s^k_{(l_1,\ldots,l_k),j}$$

   3.2. For $j \in [r]$ do

   $$g_{i+1,j} \leftarrow \sum_{k=0}^{m} \sum_{(l_1,\ldots,l_k) \in [r]^k} h^{i,k}_{(l_1,\ldots,l_k),j}$$

4. For $j \in [r]$ do $f_j \leftarrow g_{n,j} \cdot \gamma_j$

5. $f \leftarrow \sum_{j \in [r]} f_j$.

Table 3.1: Straight-line program $\Phi$

Recalling that a composition of two logspace reductions is again a logspace reduction, we conclude that the entire computation takes logarithmic space in $|\mathcal{A}|$ and $n$. □

Before presenting the reduction in Proposition 16, we recall the following characterisation [Seidl, 1990, Theorem 4.2] of equivalence of two multiplicity tree automata over an arbitrary field.

**Proposition 15** (Seidl, 1990)**.** *Suppose $\mathcal{A}$ and $\mathcal{B}$ are multiplicity tree automata of dimension $n_1$ and $n_2$, respectively, and over a ranked alphabet $\Sigma$. Then, $\mathcal{A}$ and $\mathcal{B}$ are equivalent if and only if $\|\mathcal{A}\|(t) = \|\mathcal{B}\|(t)$ for every $t \in T_\Sigma^{<n_1+n_2}$.*

We now turn to the reduction:

**Proposition 16.** *The equivalence problem for $\mathbb{Q}$-multiplicity tree automata is logspace reducible to* ACIT*.*

*Proof.* Let $\mathcal{A}$ and $\mathcal{B}$ be $\mathbb{Q}$-multiplicity tree automata over a ranked alphabet $\Sigma$, and let $n$ be the sum of their dimensions. Proposition 7 (iii) implies that

$$\begin{aligned}\sum_{t \in T_\Sigma^{\leq n}} (\|\mathcal{A}\|(t) - \|\mathcal{B}\|(t))^2 &= \sum_{t \in T_\Sigma^{\leq n}} \left(\|\mathcal{A}\|(t)^2 + \|\mathcal{B}\|(t)^2 - 2\|\mathcal{A}\|(t)\|\mathcal{B}\|(t)\right) \\ &= \sum_{t \in T_\Sigma^{\leq n}} \left(\|\mathcal{A} \times \mathcal{A}\|(t) + \|\mathcal{B} \times \mathcal{B}\|(t) - 2\|\mathcal{A} \times \mathcal{B}\|(t)\right).\end{aligned}$$

Thus by Proposition 15, automata $\mathcal{A}$ and $\mathcal{B}$ are equivalent if and only if

$$\sum_{t \in T_\Sigma^{\leq n}} \|\mathcal{A} \times \mathcal{A}\|(t) + \sum_{t \in T_\Sigma^{\leq n}} \|\mathcal{B} \times \mathcal{B}\|(t) - 2 \sum_{t \in T_\Sigma^{\leq n}} \|\mathcal{A} \times \mathcal{B}\|(t) = 0. \tag{3.5}$$

By Proposition 7, automata $\mathcal{A} \times \mathcal{A}$, $\mathcal{B} \times \mathcal{B}$, and $\mathcal{A} \times \mathcal{B}$ can be computed in logarithmic space. Thus by Lemma 14 one can compute, in logarithmic space in $|\mathcal{A}|$ and $|\mathcal{B}|$, variable-free arithmetic circuits that have outputs $\sum_{t \in T_\Sigma^{\leq n}} \|\mathcal{A} \times \mathcal{A}\|(t)$, $\sum_{t \in T_\Sigma^{\leq n}} \|\mathcal{B} \times \mathcal{B}\|(t)$, and $\sum_{t \in T_\Sigma^{\leq n}} \|\mathcal{A} \times \mathcal{B}\|(t)$, respectively. Using Equation (3.5), we can now easily construct a variable-free arithmetic circuit that has output 0 if and only if $\mathcal{A}$ and $\mathcal{B}$ are equivalent. □

### From ACIT to MTA Equivalence

We now present a converse reduction: from ACIT to the equivalence problem for $\mathbb{Q}$-MTAs. Allender et al. [2009, Proposition 2.2] give a logspace reduction of the general ACIT problem to the special case of ACIT for variable-free circuits. The latter can be reformulated as the problem of deciding whether two variable-free arithmetic circuits with only + and ×-internal gates compute the same number. As mentioned in Remark 11, we can assume that every input-gate label is either 0 or 1. We now turn to the reduction:

**Proposition 17.** ACIT *is logspace reducible to the equivalence problem for* $\mathbb{Q}$*-MTAs.*

*Proof.* Let $C_1$ and $C_2$ be two variable-free arithmetic circuits whose internal gates are labelled with $+$ or $\times$. By padding with extra gates, without loss of generality we can assume that in each circuit the children of a height-$i$ gate both have height $i-1$, $+$-gates have even height, $\times$-gates have odd height, and the output gate has an even height $h$.

In the following we define two $\mathbb{Q}$-MTAs, $\mathcal{A}_1$ and $\mathcal{A}_2$, that are equivalent if and only if circuits $C_1$ and $C_2$ have the same output. Automata $\mathcal{A}_1$ and $\mathcal{A}_2$ are both defined over a ranked alphabet $\Sigma = \{\sigma_0, \sigma_1, \sigma_2\}$ where $\sigma_0$ is a nullary, $\sigma_1$ is a unary, and $\sigma_2$ is a binary symbol. Intuitively, automata $\mathcal{A}_1$ and $\mathcal{A}_2$ both recognise the common 'tree unfolding' of circuits $C_1$ and $C_2$.

We now derive $\mathcal{A}_1$ from $C_1$; $\mathcal{A}_2$ is analogously derived from $C_2$. Let $\{v_1, \ldots, v_r\}$ be the set of gates of $C_1$ where $v_r$ is the output gate. Automaton $\mathcal{A}_1$ has a state $q_i$ for every gate $v_i$ of $C_1$. Formally, $\mathcal{A}_1 = (r, \Sigma, \mu, e_r^\top)$ where for every $i \in [r]$:

- If $v_i$ is an input gate with label 1 then $\mu(\sigma_0)_i = 1$, otherwise $\mu(\sigma_0)_i = 0$.
- If $v_i$ is a $+$-gate with children $v_{j_1}$ and $v_{j_2}$ then $\mu(\sigma_1)_{j_1,i} = \mu(\sigma_1)_{j_2,i} = 1$ if $j_1 \neq j_2$, $\mu(\sigma_1)_{j_1,i} = 2$ if $j_1 = j_2$, and $\mu(\sigma_1)_{l,i} = 0$ for every $l \notin \{j_1, j_2\}$. If $v_i$ is an input gate or a $\times$-gate then $\mu(\sigma_1)_{:,i} = \mathbf{0}_r$.
- If $v_i$ is a $\times$-gate with children $v_{j_1}$ and $v_{j_2}$, then $\mu(\sigma_2)_{(j_1,j_2),i} = 1$ and $\mu(\sigma_2)_{(l_1,l_2),i} = 0$ for every $(l_1, l_2) \neq (j_1, j_2)$. If $v_i$ is an input gate or a $+$-gate, then $\mu(\sigma_2)_{:,i} = \mathbf{0}_{r^2}$.

We define a sequence of trees $(t_n)_{n\in\mathbb{N}_0} \subseteq T_\Sigma$ by $t_0 = \sigma_0$, $t_{n+1} = \sigma_1(t_n)$ for $n$ odd, and $t_{n+1} = \sigma_2(t_n, t_n)$ for $n$ even. In the following we show that $\|\mathcal{A}_1\|(t_h) = f_{C_1}$. For every gate $v$ of $C_1$, by assumption it holds that all paths from $v$ to the output gate have equal length. We now prove that for every $i \in [r]$,

$$\mu(t_{h_i})_i = f_{v_i} \tag{3.6}$$

where $h_i := \mathit{height}(v_i)$. The proof uses induction on $h_i \in \{0, \ldots, h\}$. For the base case, let $h_i = 0$. Then, $v_i$ is an input gate and thus by definition of automaton $\mathcal{A}_1$ we have

$$\mu(t_{h_i})_i = \mu(t_0)_i = \mu(\sigma_0)_i = f_{v_i}.$$

For the induction step, let $n \in [h]$ and assume that Equation (3.6) holds for every gate $v_i$ of height less than $n$. Take an arbitrary gate $v_i$ of $C_1$ such that $h_i = n$. Let gates $v_{j_1}$ and $v_{j_2}$ be the children of $v_i$. Then $h_{j_1} = h_{j_2} = h_i - 1 = n - 1$ by assumption. The induction hypothesis now implies that $\mu(t_{h_i-1})_{j_1} = f_{v_{j_1}}$ and $\mu(t_{h_i-1})_{j_2} = f_{v_{j_2}}$. Depending on the label of $v_i$, there are two possible cases as follows:

(i) If $v_i$ is a $+$-gate, then $h_i$ is even and thus by definition of $\mathcal{A}_1$ we have

$$\begin{aligned}\mu(t_{h_i})_i = \mu(\sigma_1(t_{h_i-1}))_i &= \mu(t_{h_i-1}) \cdot \mu(\sigma_1)_{:,i} \\ &= \mu(t_{h_i-1})_{j_1} + \mu(t_{h_i-1})_{j_2} = f_{v_{j_1}} + f_{v_{j_2}} = f_{v_i}.\end{aligned}$$

(ii) If $v_i$ is a $\times$-gate, then $h_i$ is odd and thus by definition of $\mathcal{A}_1$ and Equation (2.3) we have

$$\begin{aligned}\mu(t_{h_i})_i = \mu(\sigma_2(t_{h_i-1}, t_{h_i-1}))_i &= \mu(t_{h_i-1})^{\otimes 2} \cdot \mu(\sigma_2)_{:,i} \\ &= \mu(t_{h_i-1})_{j_1} \cdot \mu(t_{h_i-1})_{j_2} = f_{v_{j_1}} \cdot f_{v_{j_2}} = f_{v_i}.\end{aligned}$$

This completes the proof of Equation (3.6) by induction. Now for the output gate $v_r$ of $C_1$, we get from Equation (3.6) that $\mu(t_h)_r = f_{v_r}$ since $h_r = h$. Therefore,

$$\|\mathcal{A}_1\|(t_h) = \mu(t_h) \cdot e_r^\top = \mu(t_h)_r = f_{v_r} = f_{C_1}.$$

Analogously, it holds that $\|\mathcal{A}_2\|(t_h) = f_{C_2}$. It is moreover clear by construction

that $\|\mathcal{A}_1\|(t) = 0$ and $\|\mathcal{A}_2\|(t) = 0$ for every $t \in T_\Sigma \setminus \{t_h\}$. Therefore, automata $\mathcal{A}_1$ and $\mathcal{A}_2$ are equivalent if and only if arithmetic circuits $C_1$ and $C_2$ have the same output. □

Propositions 16 and 17 together imply Theorem 13. On a positive note, it should be remarked that there are numerous efficient randomized algorithms for ACIT. Indeed, it was already known that there is a randomized polynomial-time algorithm for equivalence of multiplicity tree automata [Seidl, 1990]. On the other hand, we have shown that obtaining a deterministic polynomial-time algorithm for multiplicity tree automaton equivalence would imply also a deterministic polynomial-time algorithm for ACIT.

### 3.4.2 DAG Counterexamples

In the exact learning model, when answering an equivalence query the Teacher not only checks equivalence but also provides a counterexample in case of inequivalence. As mentioned above, there is a randomized polynomial-time algorithm for checking MTA equivalence [Seidl, 1990]. In this subsection, we explain why a Teacher using this algorithm would naturally give succinct DAG counterexamples.

Although the paper of Seidl [1990] does not mention counterexamples, they can be easily extracted from the algorithm presented therein. Indeed the correctness proof of that algorithm shows, *inter alia*, that for any two inequivalent multiplicity tree automata $\mathcal{A}_1 = (n_1, \Sigma, \mu_1, \gamma_1)$ and $\mathcal{A}_2 = (n_2, \Sigma, \mu_2, \gamma_2)$, there exists a tree $t$ such that $\|\mathcal{A}_1\|(t) \neq \|\mathcal{A}_2\|(t)$ and $t$ can be represented by a DAG with at most $n_1 + n_2$ vertices. To see this, we now briefly describe the main idea behind the procedure: Given MTAs $\mathcal{A}_1$ and $\mathcal{A}_2$ as above, a prefix-closed set of trees $S \subseteq T_\Sigma$ is maintained such that $\{[\mu_1(t) \;\; \mu_2(t)] : t \in S\}$ is a linearly independent set of vectors. Note that since this set of vectors lies in $\mathbb{F}^{n_1+n_2}$, it necessarily holds that $|S| \leq n_1 + n_2$. The

algorithm terminates when

$$\operatorname{span}\left\{\begin{bmatrix}\mu_1(t) & \mu_2(t)\end{bmatrix} : t \in S\right\} = \operatorname{span}\left\{\begin{bmatrix}\mu_1(t) & \mu_2(t)\end{bmatrix} : t \in T_\Sigma\right\}$$

and reports that $\mathcal{A}_1$ and $\mathcal{A}_2$ are inequivalent just in case a tree $t \in S$ is found such that

$$\begin{bmatrix}\mu_1(t) & \mu_2(t)\end{bmatrix} \cdot \begin{bmatrix}\gamma_1 \\ -\gamma_2\end{bmatrix} \neq 0,$$

i.e., $\|\mathcal{A}_1\|(t) \neq \|\mathcal{A}_2\|(t)$. Such a tree $t$, if one exists, has at most $n_1 + n_2$ subtrees and thus has a DAG representation of size at most $n_1 + n_2$. As we have seen in Example 3, the number of vertices of tree $t$ may be exponential in $n_1 + n_2$; thus it is natural that a Teacher that resolves equivalence queries using the algorithm of Seidl [1990] would return counterexamples represented succinctly as DAGs.

## 3.5 Learning Algorithm

In this section, we present our exact learning algorithm for multiplicity tree automata. The algorithm is polynomial in the size of a minimal automaton equivalent to the target and the size of a largest counterexample given as a DAG. As seen in Example 3, DAG counterexamples can be exponentially more succinct than tree counterexamples. Therefore, achieving a polynomial bound in the context of DAG representations is a more exacting criterion.

Over an arbitrary field $\mathbb{F}$, the algorithm can be seen as running on a Blum-Shub-Smale machine that can write and read field elements to and from its memory at unit cost and that can also perform arithmetic operations and equality tests on field elements at unit cost [see Arora and Barak, 2009]. Over the field of rational numbers $\mathbb{Q}$, the algorithm can be implemented in randomized polynomial time by representing

rationals as arithmetic circuits and using a coRP algorithm for equality testing of such circuits [see Allender et al., 2009].

This section is organised as follows: In §3.5.1 we present the learning algorithm, called LMTA. In §3.5.2 we prove correctness on trees, and then argue in §3.5.3 that the algorithm can be faithfully implemented using a DAG representation of trees. Finally, in §3.5.4 we give a complexity analysis of the algorithm assuming the DAG representation.

### 3.5.1 The Algorithm

In this subsection we present the learning algorithm, called LMTA. Let $f \in \mathrm{Rec}(\Sigma, \mathbb{F})$ be the target function. We recall from §2.3.4 that $\mathrm{Rec}(\Sigma, \mathbb{F})$ denotes the set of all recognisable tree series over $\Sigma$ with coefficients in $\mathbb{F}$. Algorithm LMTA learns an MTA-representation of $f$ using its Hankel matrix $H$, which has finite rank over $\mathbb{F}$ by Theorem 4.

The algorithm iteratively constructs a full row-rank submatrix of the Hankel matrix $H$. At each stage, the algorithm maintains the following data:

- An integer $n \in \mathbb{N}$.
- A set of $n$ 'rows' $X = \{t_1, \ldots, t_n\} \subseteq T_\Sigma$.
- A finite set of 'columns' $Y \subseteq C_\Sigma$ such that $\Box \in Y$.
- A submatrix $H_{X,Y}$ of $H$ that has full row rank.

These data determine a *hypothesis automaton* $\mathcal{A}$ of dimension $n$, whose states correspond to the rows of $H_{X,Y}$, with the $i^{\text{th}}$ row corresponding to the state reached after reading tree $t_i$. The Learner makes an equivalence query on the hypothesis $\mathcal{A}$. In case the Teacher answers NO, the Learner receives a counterexample $z$. The Learner then parses $z$ bottom-up to find a minimal subtree of $z$ that is also a counterexample,

and uses this subtree to augment the row set $X$ and the column set $Y$ in a way that increases the rank of the submatrix $H_{X,Y}$.

Formally, the algorithm LMTA is given in Table 3.2. Here for any $k$-ary symbol $\sigma \in \Sigma$ we write $\sigma(X, \ldots, X) := \{\sigma(t_{i_1}, \ldots, t_{i_k}) : (i_1, \ldots, i_k) \in [n]^k\}$.

**Algorithm** LMTA

**Target:** $f \in \mathrm{Rec}(\Sigma, \mathbb{F})$, where $\Sigma$ has rank $m$ and $\mathbb{F}$ is a field

1. Make an equivalence query on the 0-dimensional $\mathbb{F}$-MTA over $\Sigma$.
   If the answer is YES then **output** the 0-dimensional $\mathbb{F}$-MTA over $\Sigma$ and halt.
   Otherwise the answer is NO and $z$ is a counterexample. Initialise:
   $n \leftarrow 1$, $t_n \leftarrow z$, $X \leftarrow \{t_n\}$, $Y \leftarrow \{\Box\}$.

2. 2.1. For every $k \in \{0, \ldots, m\}$, $\sigma \in \Sigma_k$, and $(i_1, \ldots, i_k) \in [n]^k$:
   If $H_{\sigma(t_{i_1}, \ldots, t_{i_k}),Y}$ is not a linear combination of $H_{t_1,Y}, \ldots, H_{t_n,Y}$ then
   $n \leftarrow n + 1$, $t_n \leftarrow \sigma(t_{i_1}, \ldots, t_{i_k})$, $X \leftarrow X \cup \{t_n\}$.

   2.2. Define an $\mathbb{F}$-MTA $\mathcal{A} = (n, \Sigma, \mu, \gamma)$ as follows:

   - $\gamma = H_{X,\Box}$.
   - For every $k \in \{0, \ldots, m\}$ and $\sigma \in \Sigma_k$:
     Define matrix $\mu(\sigma) \in \mathbb{F}^{n^k \times n}$ by the equation

$$\mu(\sigma) \cdot H_{X,Y} = H_{\sigma(X,\ldots,X),Y}. \tag{3.7}$$

3. 3.1. Make an equivalence query on $\mathcal{A}$.
   If the answer is YES then **output** $\mathcal{A}$ and halt.
   Otherwise the answer is NO and $z$ is a counterexample. Searching bottom-up, find a subtree $\sigma(\tau_1, \ldots, \tau_k)$ of $z$ that satisfies the following two conditions:

   (i) For every $j \in [k]$, $H_{\tau_j,Y} = \mu(\tau_j) \cdot H_{X,Y}$.
   (ii) For some $c \in Y$, $H_{\sigma(\tau_1,\ldots,\tau_k),c} \neq \mu(\sigma(\tau_1, \ldots, \tau_k)) \cdot H_{X,c}$.

   3.2. For every $j \in [k]$ and $(i_1, \ldots, i_{j-1}) \in [n]^{j-1}$:
   $Y \leftarrow Y \cup \{c[\sigma(t_{i_1}, \ldots, t_{i_{j-1}}, \Box, \tau_{j+1}, \ldots, \tau_k)]\}$.

   3.3. For every $j \in [k]$:
   If $H_{\tau_j,Y}$ is not a linear combination of $H_{t_1,Y}, \ldots, H_{t_n,Y}$ then
   $n \leftarrow n + 1$, $t_n \leftarrow \tau_j$, $X \leftarrow X \cup \{t_n\}$.

   3.4. Go to 2.

Table 3.2: Algorithm LMTA

Algorithm LMTA follows a classical scheme: it generalises the procedure of Beimel

et al. [2000] by working with a more general notion of a Hankel matrix: one representing a tree series rather than a word series. Moreover, LMTA differs from the procedure of Habrard and Oncina [2006] in the way counterexamples are treated and the hypothesis automaton updated; we provide more details on this point at the end of this section.

### 3.5.2 Correctness Proof

In this subsection, we prove the correctness of the exact learning algorithm LMTA. Specifically, we show that, given a target $f \in \mathrm{Rec}(\Sigma, \mathbb{F})$, algorithm LMTA outputs a minimal MTA-representation of $f$ after at most $\mathrm{rank}(H)$ iterations of the main loop.

The correctness proof naturally breaks down into several lemmas. First, we show that matrix $H_{X,Y}$ has full row rank.

**Lemma 18.** *Linear independence of the set of vectors $\{H_{t_1,Y}, \ldots, H_{t_n,Y}\}$ is an invariant of the loop consisting of Step 2 and Step 3.*

*Proof.* We argue inductively on the number of iterations of the loop. The base case $n = 1$ clearly holds since $f(z) \neq 0$.

For the induction step, suppose that the set $\{H_{t_1,Y}, \ldots, H_{t_n,Y}\}$ is linearly independent at the start of an iteration of the loop. If a tree $t \in T_\Sigma$ is added to $X$ during Step 2.1, then $H_{t,Y}$ is not a linear combination of $H_{t_1,Y}, \ldots, H_{t_n,Y}$, and therefore $\{H_{t_1,Y}, \ldots, H_{t_n,Y}, H_{t,Y}\}$ is a linearly independent set of vectors. Hence, the set $\{H_{t_1,Y}, \ldots, H_{t_n,Y}\}$ is linearly independent at the start of Step 3.

Unless the algorithm halts in Step 3.1, it proceeds to Step 3.2 where the set of columns $Y$ is increased, which clearly preserves linear independence of vectors $H_{t_1,Y}, \ldots, H_{t_n,Y}$. If a tree $\tau_j$ is added to $X$ in Step 3.3, then $H_{\tau_j,Y}$ is not a linear combination of $H_{t_1,Y}, \ldots, H_{t_n,Y}$ which implies that the vectors $H_{t_1,Y}, \ldots, H_{t_n,Y}, H_{\tau_j,Y}$ are linearly independent. Hence, the set $\{H_{t_1,Y}, \ldots, H_{t_n,Y}\}$ is linearly independent at the start of the next iteration of the loop. This completes the induction step. □

Secondly, we show that Step 2.2 of LMTA can always be performed.

**Lemma 19.** *Whenever Step 2.2 starts, for every $k \in \{0, \ldots, m\}$ and $\sigma \in \Sigma_k$ there exists a unique matrix $\mu(\sigma) \in \mathbb{F}^{n^k \times n}$ satisfying Equation* (3.7).

*Proof.* Take any $(i_1, \ldots, i_k) \in [n]^k$. Step 2.1 ensures that $H_{\sigma(t_{i_1},\ldots,t_{i_k}),Y}$ can be represented as a linear combination of vectors $H_{t_1,Y}, \ldots, H_{t_n,Y}$. This representation is unique since $H_{t_1,Y}, \ldots, H_{t_n,Y}$ are linearly independent vectors by Lemma 18. Row $\mu(\sigma)_{(i_1,\ldots,i_k)} \in \mathbb{F}^{1\times n}$ is, therefore, uniquely defined by the equation $\mu(\sigma)_{(i_1,\ldots,i_k),:} \cdot H_{X,Y} = H_{\sigma(t_{i_1},\ldots,t_{i_k}),Y}$. □

Thirdly, we show that Step 3.1 of LMTA can always be performed.

**Lemma 20.** *Suppose that upon making an equivalence query on $\mathcal{A}$ in Step 3.1, the Learner receives the answer NO and a counterexample $z$. Then, there exists a subtree $\sigma(\tau_1, \ldots, \tau_k)$ of $z$ that satisfies the following two conditions:*

(i) *For every $j \in [k]$, $H_{\tau_j,Y} = \mu(\tau_j) \cdot H_{X,Y}$.*

(ii) *For some $c \in Y$, $H_{\sigma(\tau_1,\ldots,\tau_k),c} \neq \mu(\sigma(\tau_1, \ldots, \tau_k)) \cdot H_{X,c}$.*

*Proof.* Towards a contradiction, assume that there exists no subtree $\sigma(\tau_1, \ldots, \tau_k)$ of $z$ that satisfies conditions (i) and (ii). We claim that then for every subtree $\tau$ of $z$, it holds that

$$H_{\tau,Y} = \mu(\tau) \cdot H_{X,Y}. \tag{3.8}$$

In the following we prove this claim using induction on $height(\tau)$. The base case $\tau \in \Sigma_0$ follows immediately from Equation (3.7). For the induction step, let $0 \leq h < height(z)$ and assume that Equation (3.8) holds for every subtree $\tau \in T_\Sigma^{\leq h}$ of $z$. Take an arbitrary subtree $\tau \in T_\Sigma^{h+1}$ of $z$. Then $\tau = \sigma(\tau_1, \ldots, \tau_k)$ for some $k \in [m]$, $\sigma \in \Sigma_k$, and $\tau_1, \ldots, \tau_k \in T_\Sigma^{\leq h}$, where $\tau_1, \ldots, \tau_k$ are subtrees of $z$. The induction hypothesis

implies that $H_{\tau_j,Y} = \mu(\tau_j) \cdot H_{X,Y}$ holds for every $j \in [k]$. Hence, subtree $\tau$ satisfies condition (i). By assumption, no subtree of $z$ satisfies both conditions (i) and (ii). Thus $\tau$ does not satisfy condition (ii), i.e., it holds that $H_{\tau,Y} = \mu(\tau) \cdot H_{X,Y}$. This completes the proof by induction.

Equation (3.8) for $\tau = z$ gives $H_{z,Y} = \mu(z) \cdot H_{X,Y}$. Since $\square \in Y$, this in particular implies that

$$f(z) = H_{z,\square} = \mu(z) \cdot H_{X,\square} = \mu(z) \cdot \gamma = \|\mathcal{A}\|(z),$$

which yields a contradiction since $z$ is a counterexample for the hypothesis $\mathcal{A}$. $\square$

Finally, we show that the row set $X$ is augmented with at least one element in each iteration of the main loop.

**Lemma 21.** *Every complete iteration of the Step 2 - 3 loop strictly increases the cardinality of the row set $X$.*

*Proof.* It suffices to show that in Step 3.3 at least one of the trees $\tau_1, \ldots, \tau_k$ is added to $X$. By Lemma 18, at the start of Step 3.2 vectors $H_{t_1,Y}, \ldots, H_{t_n,Y}$ are linearly independent. Thus by condition (i) of Step 3.1, for every $j \in [k]$ it holds that

$$H_{\tau_j,Y} = \mu(\tau_j) \cdot H_{X,Y} \tag{3.9}$$

and, moreover, Equation (3.9) is the unique representation of vector $H_{\tau_j,Y}$ as a linear combination of vectors $H_{t_1,Y}, \ldots, H_{t_n,Y}$. Clearly, vectors $H_{t_1,Y}, \ldots, H_{t_n,Y}$ remain linearly independent when Step 3.2 ends.

Towards a contradiction, assume that in Step 3.3 none of the trees $\tau_1, \ldots, \tau_k$ is added to $X$. This means that for every $j \in [k]$, vector $H_{\tau_j,Y}$ can be represented as a linear combination of $H_{t_1,Y}, \ldots, H_{t_n,Y}$. The latter representation is unique, since vectors $H_{t_1,Y}, \ldots, H_{t_n,Y}$ are linearly independent, and is given by Equation (3.9). By

condition (ii) of Step 3.1 and Equations (3.7) and (2.3), we now have that

$$\begin{aligned} H_{\sigma(\tau_1,\ldots,\tau_k),c} &\neq \mu(\sigma(\tau_1,\ldots,\tau_k)) \cdot H_{X,c} \\ &= (\mu(\tau_1) \otimes \cdots \otimes \mu(\tau_k)) \cdot \mu(\sigma) \cdot H_{X,c} \\ &= (\mu(\tau_1) \otimes \ldots \otimes \mu(\tau_k)) \cdot H_{\sigma(X,\ldots,X),c} \\ &= \sum_{(i_1,\ldots,i_k)\in[n]^k} \left( \prod_{j=1}^{k} \mu(\tau_j)_{i_j} \right) \cdot H_{\sigma(t_{i_1},\ldots,t_{i_k}),c}. \end{aligned} \tag{3.10}$$

By Step 3.2, it holds that $c[\sigma(t_{i_1},\ldots,t_{i_{j-1}},\Box,\tau_{j+1},\ldots,\tau_k)] \in Y$ for every $j \in [k]$ and every $(i_1,\ldots,i_{j-1}) \in [n]^{j-1}$. Thus by Equation (3.9) for $j = k$, we have

$$\begin{aligned} &\sum_{(i_1,\ldots,i_k)\in[n]^k} \left( \prod_{j=1}^{k} \mu(\tau_j)_{i_j} \right) \cdot H_{\sigma(t_{i_1},\ldots,t_{i_k}),c} \\ &= \sum_{(i_1,\ldots,i_{k-1})\in[n]^{k-1}} \left( \prod_{j=1}^{k-1} \mu(\tau_j)_{i_j} \right) \cdot \sum_{i\in[n]} \mu(\tau_k)_i \cdot H_{t_i,c[\sigma(t_{i_1},\ldots,t_{i_{k-1}},\Box)]} \\ &= \sum_{(i_1,\ldots,i_{k-1})\in[n]^{k-1}} \left( \prod_{j=1}^{k-1} \mu(\tau_j)_{i_j} \right) \cdot \mu(\tau_k) \cdot H_{X,c[\sigma(t_{i_1},\ldots,t_{i_{k-1}},\Box)]} \\ &= \sum_{(i_1,\ldots,i_{k-1})\in[n]^{k-1}} \left( \prod_{j=1}^{k-1} \mu(\tau_j)_{i_j} \right) \cdot H_{\tau_k,c[\sigma(t_{i_1},\ldots,t_{i_{k-1}},\Box)]}. \end{aligned} \tag{3.11}$$

Proceeding inductively as above and using Equation (3.9) for every $j \in \{k-1,\ldots,1\}$, we get that the expression of (3.11) is equal to $H_{\tau_1,c[\sigma(\Box,\tau_2,\ldots,\tau_k)]}$. However, this contradicts Equation (3.10). The result follows. □

Putting together Lemmas 18 - 21, we conclude the following:

**Proposition 22.** *Let $\Sigma$ be a ranked alphabet and $\mathbb{F}$ be a field. Let $f \in Rec(\Sigma,\mathbb{F})$, let $H$ be the Hankel matrix of $f$, and let $r$ be the rank (over $\mathbb{F}$) of $H$. On target $f$, algorithm* LMTA *outputs a minimal MTA-representation of $f$ after at most $r$ iterations of the loop consisting of Step 2 and Step 3.*

*Proof.* Lemmas 19 and 20 show that every step of algorithm LMTA can be performed.

Theorem 4 implies that $r$ is finite. From Lemma 18 we know that, whenever Step 2 starts, matrix $H_{X,Y}$ has full row rank and thus $n = |X| \leq r$. Lemma 21 implies that $n$ increases by at least one in each iteration of the Step 2 - 3 loop. Therefore, the number of iterations of the loop is at most $r$.

The proof follows by observing that LMTA halts only upon receiving the answer YES to an equivalence query. □

### 3.5.3 Succinct Representations

In this subsection, we explain how algorithm LMTA can be correctly implemented using a DAG representation of trees. In particular, we assume that membership queries are made on $\Sigma$-DAGs, that the counterexamples are given as $\Sigma$-DAGs, the elements of $X$ are $\Sigma$-DAGs, and the elements of $Y$ are DAG representations of $\Sigma$-contexts, i.e., $(\{\Box\} \cup \Sigma)$-DAGs.

As shown in §3.3.2, multiplicity tree automata can run directly on DAGs and, moreover, they assign equal weight to a DAG and to its tree unfolding. Crucially also, as explained in the proof of Theorem 23, Step 3.1 can be run directly on a DAG representation of the counterexample, without unfolding. Specifically, Step 3.1 involves multiple executions of the hypothesis automaton on trees. By Proposition 10, we can faithfully carry out these executions on DAG representations of trees. Step 3.1 also involves considering all the subtrees of a given counterexample. However, by Proposition 8, this is equivalent to looking at all the sub-DAGs of a DAG representation of the counterexample.

At various points in the algorithm, we take $c \in Y$, $t \in X$ and compute their concatenation $c[t]$ in order to determine the corresponding entry $H_{t,c}$ of the Hankel matrix by making a membership query. Proposition 9 implies that this can be done faithfully using DAG representations of $\Sigma$-trees and $\Sigma$-contexts.

### 3.5.4 Complexity Analysis

In this subsection, we give a query and computational complexity analysis of our algorithm and compare it to the best previously-known exact learning algorithm for multiplicity tree automata [Habrard and Oncina, 2006] showing in particular an exponential improvement on the query complexity and the running time in the worst case.

**Theorem 23.** *Let $f \in Rec(\Sigma, \mathbb{F})$ where $\Sigma$ has rank $m$ and $\mathbb{F}$ is a field. Let $\mathcal{A}$ be a minimal MTA-representation of $f$, and let $r$ be the dimension of $\mathcal{A}$. Then, $f$ is learnable by the algorithm* LMTA*, making $r+1$ equivalence queries, $|\mathcal{A}|^2 + |\mathcal{A}| \cdot s$ membership queries, and $O(|\mathcal{A}|^2 + |\mathcal{A}| \cdot r \cdot s)$ arithmetic operations, where $s$ denotes the size of a largest counterexample $z$, represented as a DAG, that is obtained during the execution of the algorithm.*

*Proof.* Let $H$ be the Hankel matrix of $f$. Note that, by Theorem 4, the rank of $H$ is equal to $r$. Proposition 22 implies that on target $f$, algorithm LMTA outputs a minimal MTA-representation of $f$ after at most $r$ iterations of the Step 2 - 3 loop, thereby making at most $r+1$ equivalence queries.

From Lemma 18 we know that matrix $H_{X,Y}$ has full row rank, which implies that $|X| \leq r$. As for the cardinality of the column set $Y$, at the end of Step 1 we have $|Y| = 1$. Furthermore, in each iteration of Step 3.2 the number of columns added to $Y$ is at most

$$\sum_{j=1}^{k} n^{j-1} \leq \sum_{j=1}^{k} r^{j-1} = \frac{r^k - 1}{r - 1} \leq \frac{r^m - 1}{r - 1},$$

where $k$ and $n$ are as defined in Step 3.2. Since the number of iterations of Step 3.2 is at most $r-1$, we have $|Y| \leq r^m$.

The number of membership queries made in Step 2 over the whole algorithm is

$$\left(\sum_{\sigma\in\Sigma}|\sigma(X,\ldots,X)|+|X|\right)\cdot|Y|$$

because the Learner needs to ask for the values of the entries of matrices $H_{X,Y}$ and $H_{\sigma(X,\ldots,X),Y}$ for every $\sigma\in\Sigma$.

To analyse the number of membership queries made in Step 3, we now detail the procedure by which an appropriate sub-DAG of the counterexample $z$ is found in Step 3.1. By Lemma 20, there exists a sub-DAG $\tau$ of $z$ such that $H_{\tau,Y}\neq\mu(\tau)\cdot H_{X,Y}$. Thus given a counterexample $z$ in Step 3.1, the procedure for finding a required sub-DAG of $z$ is as follows: Check if $H_{\tau,Y}=\mu(\tau)\cdot H_{X,Y}$ for every sub-DAG $\tau$ of $z$ in a nondecreasing order of height; stop when a sub-DAG $\tau$ is found such that $H_{\tau,Y}\neq\mu(\tau)\cdot H_{X,Y}$.

In each iteration of Step 3, the Learner makes $size(z)\cdot|Y|\leq s\cdot|Y|$ membership queries because, for every sub-DAG $\tau$ of $z$, the Learner needs to ask for the values of the entries of vector $H_{\tau,Y}$. All together, the number of membership queries made during the execution of the algorithm is at most

$$\begin{aligned}&\left(\sum_{\sigma\in\Sigma}|\sigma(X,\ldots,X)|+|X|\right)\cdot|Y|+(r-1)\cdot s\cdot|Y|\\&\leq\left(\sum_{\sigma\in\Sigma}r^{\mathrm{rk}(\sigma)}+r\right)\cdot r^m+(r-1)\cdot s\cdot r^m\leq|\mathcal{A}|^2+|\mathcal{A}|\cdot s.\end{aligned}$$

As for the arithmetic complexity, in Step 2.1 one can determine if a vector $H_{\sigma(t_{i_1},\ldots,t_{i_k}),Y}$ is a linear combination of $H_{t_1,Y},\ldots,H_{t_n,Y}$ via Gaussian elimination using $O(n^2\cdot|Y|)$ arithmetic operations [see Cohen, 1993, Section 2.3]. Analogously, in Step 3.3 one can determine if $H_{\tau_j,Y}$ is a linear combination of $H_{t_1,Y},\ldots,H_{t_n,Y}$ via Gaussian elimination using $O(n^2\cdot|Y|)$ arithmetic operations. Since $|X|\leq r$ and $|Y|\leq r^m$, all together Step 2.1 and Step 3.3 require at most $O(|\mathcal{A}|^2)$ arithmetic

operations.

Lemma 19 implies that in each iteration of Step 2.2, for every $\sigma \in \Sigma$ there exists a unique matrix $\mu(\sigma) \in \mathbb{F}^{n^{\mathrm{rk}(\sigma)} \times n}$ that satisfies Equation (3.7). To perform an iteration of Step 2.2, we first put matrix $H_{X,Y}$ in echelon form and then, for each $\sigma \in \Sigma$, solve Equation (3.7) for $\mu(\sigma)$ by back substitution. It follows from standard complexity bounds on the conversion of matrices to echelon form [Cohen, 1993, Section 2.3] that the total operation count for Step 2.2 can be bounded above by $O(|\mathcal{A}|^2)$.

Finally, let us consider the arithmetic complexity of Step 3.1. In every iteration, for each sub-DAG $\tau$ of the counterexample $z$ the Learner needs to compute the vector $\mu(\tau)$ and the product $\mu(\tau) \cdot H_{X,Y}$. Note that $\mu(\tau)$ can be computed bottom-up from the sub-DAGs of $\tau$. Since $z$ has at most $s$ sub-DAGs, Step 3.1 requires at most $O(|\mathcal{A}| \cdot r \cdot s)$ arithmetic operations. All together, the algorithm requires at most $O(|\mathcal{A}|^2 + |\mathcal{A}| \cdot r \cdot s)$ arithmetic operations. □

Algorithm LMTA can be used to show that over $\mathbb{Q}$, multiplicity tree automata are exactly learnable in randomized polynomial time. The key idea is to represent numbers as arithmetic circuits. In executing LMTA, the Learner need only perform arithmetic operations on circuits (addition, subtraction, multiplication, and division), which can be done in constant time, and equality testing, which can be done in coRP [see Arora and Barak, 2009]. These suffice for all the operations detailed in the proof of Theorem 23; in particular they suffice for Gaussian elimination, which can be used to implement the linear-independence checks in LMTA.

The complexity of algorithm LMTA should be compared to the complexity of the algorithm of Habrard and Oncina [2006], which learns multiplicity tree automata by making $r + 1$ equivalence queries, $|\mathcal{A}| \cdot s$ membership queries, and a number of arithmetic operations polynomial in $|\mathcal{A}|$ and $s$, where $s$ is the size of a largest counterexample given as a tree. Note that the algorithm of Habrard and Oncina [2006] cannot be straightforwardly adapted to work directly with DAG representations of

trees since when given a counterexample $z$, every suffix of $z$ is added to the set of columns. However, the tree unfolding of a DAG can have exponentially many different suffixes in the size of the DAG. For example, the DAG in Figure 3.2 has size $n$, and its tree unfolding, shown in Figure 3.1, has $O(2^n)$ different suffixes.

## 3.6 Lower Bounds on Query Complexity

In this section, we study lower bounds on the query complexity of learning multiplicity tree automata in the exact learning model. Our results generalise the corresponding lower bounds for learning multiplicity word automata by Bisht et al. [2006], and make no assumption about the computational model of the learning algorithm.

First, we give a lower bound on the total number of queries required by an exact learning algorithm that works over any field, which is the situation of our algorithm in Section 3.5. Note that when we say that an algorithm works over any field, we mean that it just uses field arithmetic, equality testing, and the ability to store and communicate field elements to the Teacher, and its correctness depends only on these operations satisfying the field axioms.

**Theorem 24.** *Any exact learning algorithm that learns the class of multiplicity tree automata of dimension at most $r$, over a ranked alphabet $(\Sigma, \mathrm{rk})$ and any field, must make at least $\sum_{\sigma\in\Sigma} r^{\mathrm{rk}(\sigma)+1} - r^2$ queries.*

*Proof.* Take an arbitrary exact learning algorithm $\mathsf{L}$ that learns the class of multiplicity tree automata of dimension at most $r$, over a ranked alphabet $(\Sigma, \mathrm{rk})$ and over any field.

Let $\mathbb{F}$ be any field. Let $\mathbb{K} := \mathbb{F}(\{z^\sigma_{i,j} : \sigma \in \Sigma, i \in [r^{\mathrm{rk}(\sigma)}], j \in [r]\})$ be an extension field of $\mathbb{F}$, where the set $\{z^\sigma_{i,j} : \sigma \in \Sigma, i \in [r^{\mathrm{rk}(\sigma)}], j \in [r]\}$ is algebraically independent over $\mathbb{F}$. We define a 'generic' $\mathbb{K}$-multiplicity tree automaton $\mathcal{A} := (r, \Sigma, \mu, \gamma)$ where $\gamma = e_1^\top \in \mathbb{F}^{r\times 1}$ and $\mu(\sigma) = [z^\sigma_{i,j}]_{i,j} \in \mathbb{K}^{r^{rk(\sigma)}\times r}$ for every $\sigma \in \Sigma$. We define a tree series

$f := \|\mathcal{A}\|$. Observe that every $r$-dimensional $\mathbb{F}$-MTA over $\Sigma$ can be obtained from $\mathcal{A}$ by substituting values from the field $\mathbb{F}$ for the variables $z^{\sigma}_{i,j}$. Thus if the Hankel matrix of $f$ had rank less than $r$, then every $r$-dimensional $\mathbb{F}$-MTA over $\Sigma$ would have Hankel matrix of rank less than $r$. Therefore, the Hankel matrix of $f$ has rank $r$.

We run algorithm $\mathsf{L}$ on the target function $f$. By assumption, the output of $\mathsf{L}$ is an MTA $\mathcal{A}' = (r, \Sigma, \mu', \gamma')$ such that $\|\mathcal{A}'\| \equiv f$. Let $n$ be the number of queries made by $\mathsf{L}$ on target $f$. Let $t_1, \ldots, t_n \in T_\Sigma$ be the trees on which $\mathsf{L}$ either made a membership query, or which were received as the counterexample to an equivalence query. Then for every $l \in [n]$, there exists a multivariate polynomial $p_l \in \mathbb{F}[(z^{\sigma}_{i,j})_{i,j,\sigma}]$ such that $f(t_l) = p_l$.

Note that both $\mathcal{A}$ and $\mathcal{A}'$ are minimal MTA-representations of $f$. Thus by Theorem 5, there exists an invertible matrix $U \in \mathbb{K}^{r\times r}$ such that $\gamma = U \cdot \gamma'$ and $\mu(\sigma) = U^{\otimes \mathrm{rk}(\sigma)} \cdot \mu'(\sigma) \cdot U^{-1}$ for every $\sigma \in \Sigma$. This implies that the entries of matrices $\mu(\sigma)$, $\sigma \in \Sigma$, lie in an extension of $\mathbb{F}$ generated by the entries of $U$ and $\{p_l : l \in [n]\}$, i.e., by at most $r^2 + n$ elements. But since the entries of matrices $\mu(\sigma)$, $\sigma \in \Sigma$, form an algebraically independent set over $\mathbb{F}$, the total number $\sum_{\sigma\in\Sigma} r^{\mathrm{rk}(\sigma)+1}$ of such entries is at most $r^2 + n$. Therefore, the number of queries $n$ is at least $\sum_{\sigma\in\Sigma} r^{\mathrm{rk}(\sigma)+1} - r^2$. $\square$

One may wonder whether a learning algorithm could do better over a specific field $\mathbb{F}$ by exploiting particular features of that field such as having zero characteristic, being ordered, or being algebraically closed. In this setting, we have the following lower bound.

**Theorem 25.** *Let $\mathbb{F}$ be a fixed but arbitrary field. Any exact learning algorithm that learns the class of $\mathbb{F}$-multiplicity tree automata of dimension at most $r$, over a ranked alphabet $(\Sigma, \mathrm{rk})$ that has rank $m$ and contains at least one unary symbol, must make*

*number of queries at least*

$$\frac{1}{2^{m+1}} \cdot \left( \sum_{\sigma \in \Sigma} r^{\mathrm{rk}(\sigma)+1} - r^2 - r \right).$$

*Proof.* Without loss of generality, we can assume that $r$ is even and can, therefore, define a natural number $n := r/2$. Let $\mathsf{L}$ be an exact learning algorithm for the class of $\mathbb{F}$-multiplicity tree automata of dimension at most $r$, over a ranked alphabet $(\Sigma, \mathrm{rk})$ of rank $m$ such that $\mathrm{rk}^{-1}(\{1\}) \neq \emptyset$. We will identify a class of functions $\mathcal{C}$ such that $\mathsf{L}$ has to make at least $\sum_{\sigma \in \Sigma} n^{\mathrm{rk}(\sigma)+1} - n^2 - n$ queries to distinguish between the members of $\mathcal{C}$.

Let $\sigma_0, \sigma_1 \in \Sigma$ be a nullary and a unary symbol, respectively. Let $P \in \mathbb{F}^{n \times n}$ be the permutation matrix corresponding to the cycle $(1, 2, \ldots, n)$. Define $\mathcal{S}$ to be the set of all $\mathbb{F}$-multiplicity tree automata $(2n, \Sigma, \mu, \gamma)$ where:

- $\mu(\sigma_0) = [1 \;\; 0] \otimes e_1$ and $\mu(\sigma_1) = I_2 \otimes P$;
- For each $k$-ary symbol $\sigma \in \Sigma \setminus \{\sigma_0, \sigma_1\}$, there exists $B(\sigma) \in \mathbb{F}^{n^k \times n}$ such that

$$\mu(\sigma) = \begin{bmatrix} 1 & 1 \end{bmatrix} \otimes \left( \begin{bmatrix} I_n \\ -I_n \end{bmatrix}^{\otimes k} \cdot B(\sigma) \right);$$

- $\gamma = [1 \;\; 0]^\top \otimes e_1^\top$.

We define a set of recognisable tree series $\mathcal{C} := \{\|\mathcal{A}\| : \mathcal{A} \in \mathcal{S}\}$.

In Lemma 26 we state some properties of the functions in $\mathcal{C}$. Specifically, we show that the coefficient of a tree $t \in T_\Sigma$ in any series $f \in \mathcal{C}$ fundamentally depends on whether $t$ has zero, one, or at least two nodes whose label is not $\sigma_0$ or $\sigma_1$. Here for every $i \in \mathbb{N}_0$ and $t \in T_\Sigma$, we use $\sigma_1^i(t)$ to denote the tree $\underbrace{\sigma_1(\sigma_1(\ldots \sigma_1(}_{i} t) \ldots))$.

**Lemma 26.** *The following properties hold for every $f \in \mathcal{C}$ and $t \in T_\Sigma$:*

(i) *If* $t = \sigma_1^j(\sigma_0)$ *where* $j \in \{0, 1, \ldots, n-1\}$, *then* $f(t) = 1$ *if* $j = 0$ *and* $f(t) = 0$ *otherwise.*

(ii) *If* $t = \sigma_1^j(\sigma(\sigma_1^{i_1}(\sigma_0), \ldots, \sigma_1^{i_k}(\sigma_0)))$ *where* $k \in \{0, 1, \ldots, m\}$, $\sigma \in \Sigma_k \setminus \{\sigma_0, \sigma_1\}$, *and* $j, i_1, \ldots, i_k \in \{0, 1, \ldots, n-1\}$, *then* $f(t) = B(\sigma)_{(1+i_1,\ldots,1+i_k),(1+n-j) \bmod n}$.

(iii) *If* $\sum_{\sigma \in \Sigma \setminus \{\sigma_0, \sigma_1\}} \#_\sigma(t) \geq 2$, *then* $f(t) = 0$.

*Proof.* Let $\mathcal{A} = (2n, \Sigma, \mu, \gamma) \in \mathcal{S}$ be such that $\|\mathcal{A}\| \equiv f$. First, we prove property (i). Using Equation (2.2) and the mixed-product property of Kronecker product, we get that

$$\mu(\sigma_1^j(\sigma_0)) = \mu(\sigma_0) \cdot \mu(\sigma_1)^j = \left(\begin{bmatrix} 1 & 0 \end{bmatrix} \otimes e_1\right) \cdot \left(I_2 \otimes P^j\right) = \begin{bmatrix} 1 & 0 \end{bmatrix} \otimes e_1 P^j \quad (3.12)$$

and therefore

$$\begin{aligned} f(\sigma_1^j(\sigma_0)) = \mu(\sigma_1^j(\sigma_0)) \cdot \gamma &= \left(\begin{bmatrix} 1 & 0 \end{bmatrix} \otimes e_1 P^j\right) \cdot \left(\begin{bmatrix} 1 & 0 \end{bmatrix}^\top \otimes e_1^\top\right) \\ &= \left(\begin{bmatrix} 1 & 0 \end{bmatrix} \cdot \begin{bmatrix} 1 & 0 \end{bmatrix}^\top\right) \otimes \left(e_1 P^j \cdot e_1^\top\right) = e_{j+1} \cdot e_1^\top. \quad (3.13) \end{aligned}$$

If $j = 0$ then the expression of (3.13) is equal to 1, otherwise the expression of (3.13) is equal to 0. This completes the proof of property (i).

Next, we prove property (ii). By the mixed-product property of Kronecker product and Equations (2.2), (2.4), and (3.12), we have

$$\begin{aligned} &\mu(\sigma_1^j(\sigma(\sigma_1^{i_1}(\sigma_0), \ldots, \sigma_1^{i_k}(\sigma_0)))) \\ &= \left(\bigotimes_{l=1}^{k} \mu(\sigma_1^{i_l}(\sigma_0))\right) \cdot \mu(\sigma) \cdot \mu(\sigma_1)^j \\ &= \left(\begin{bmatrix} 1 \end{bmatrix} \otimes \bigotimes_{l=1}^{k} \mu(\sigma_1^{i_l}(\sigma_0))\right) \cdot \left(\begin{bmatrix} 1 & 1 \end{bmatrix} \otimes \left(\begin{bmatrix} I_n \\ -I_n \end{bmatrix}^{\otimes k} \cdot B(\sigma)\right)\right) \cdot (I_2 \otimes P)^j \end{aligned}$$

$$
\begin{aligned}
&= \left( \left( \begin{bmatrix} 1 \end{bmatrix} \cdot \begin{bmatrix} 1 & 1 \end{bmatrix} \right) \otimes \left( \bigotimes_{l=1}^{k} \mu(\sigma_1^{i_l}(\sigma_0)) \cdot \begin{bmatrix} I_n \\ -I_n \end{bmatrix}^{\otimes k} \cdot B(\sigma) \right) \right) \cdot (I_2 \otimes P^j) \\
&= \left( \begin{bmatrix} 1 & 1 \end{bmatrix} \otimes \left( \bigotimes_{l=1}^{k} \left( \left( \begin{bmatrix} 1 & 0 \end{bmatrix} \otimes e_1 P^{i_l} \right) \cdot \begin{bmatrix} I_n \\ -I_n \end{bmatrix} \right) \cdot B(\sigma) \right) \right) \cdot (I_2 \otimes P^j) \\
&= \left( \begin{bmatrix} 1 & 1 \end{bmatrix} \otimes \left( \bigotimes_{l=1}^{k} e_1 P^{i_l} \cdot B(\sigma) \right) \right) \cdot (I_2 \otimes P^j) \\
&= \left( \begin{bmatrix} 1 & 1 \end{bmatrix} \cdot I_2 \right) \otimes \left( \bigotimes_{l=1}^{k} e_{1+i_l} \cdot B(\sigma) \cdot P^j \right) \\
&= \begin{bmatrix} 1 & 1 \end{bmatrix} \otimes \left( B(\sigma)_{(1+i_1,\ldots,1+i_k),:} \cdot P^j \right) \qquad (3.14)
\end{aligned}
$$

and therefore, using the fact that $P^n = I_n$, we get that

$$
\begin{aligned}
f(\sigma_1^j(\sigma(\sigma_1^{i_1}(\sigma_0), \ldots, \sigma_1^{i_k}(\sigma_0)))) &= \left( \begin{bmatrix} 1 & 1 \end{bmatrix} \otimes \left( B(\sigma)_{(1+i_1,\ldots,1+i_k),:} \cdot P^j \right) \right) \cdot \gamma \\
&= \left( \begin{bmatrix} 1 & 1 \end{bmatrix} \cdot \begin{bmatrix} 1 & 0 \end{bmatrix}^\top \right) \otimes \left( B(\sigma)_{(1+i_1,\ldots,1+i_k),:} \cdot P^j \cdot e_1^\top \right) \\
&= B(\sigma)_{(1+i_1,\ldots,1+i_k),:} \cdot \left( e_1 P^{n-j} \right)^\top \\
&= B(\sigma)_{(1+i_1,\ldots,1+i_k),(1+n-j) \bmod n}.
\end{aligned}
$$

Lastly, we prove property (iii). If $\sum_{\sigma \in \Sigma \setminus \{\sigma_0, \sigma_1\}} \#_\sigma(t) \geq 2$, i.e., $t$ has at least two nodes whose label is not $\sigma_0$ or $\sigma_1$, then there exists a subtree $\sigma'(t_1, \ldots, t_k)$ of $t$ where $k \geq 1$, $\sigma' \in \Sigma_k \setminus \{\sigma_1\}$, and $\sum_{\sigma \in \Sigma \setminus \{\sigma_0, \sigma_1\}} \#_\sigma(t_i) = 1$ for some $i \in [k]$. It follows from Equation (3.14) that $\mu(t_i) = [1 \;\; 1] \otimes \alpha$ for some $\alpha \in \mathbb{F}^{1 \times n}$. By the mixed-product property of Kronecker product and Equation (2.4), we have

$$
\mu(\sigma'(t_1, \ldots, t_k)) = \left( \bigotimes_{j=1}^{k} \mu(t_j) \right) \cdot \left( \begin{bmatrix} 1 & 1 \end{bmatrix} \otimes \left( \begin{bmatrix} I_n \\ -I_n \end{bmatrix}^{\otimes k} \cdot B(\sigma') \right) \right)
$$

$$= \begin{bmatrix} 1 & 1 \end{bmatrix} \otimes \left( \bigotimes_{j=1}^{k} \mu(t_j) \cdot \begin{bmatrix} I_n \\ -I_n \end{bmatrix}^{\otimes k} \cdot B(\sigma') \right)$$
$$= \begin{bmatrix} 1 & 1 \end{bmatrix} \otimes \left( \bigotimes_{j=1}^{k} \left( \mu(t_j) \cdot \begin{bmatrix} I_n \\ -I_n \end{bmatrix} \right) \cdot B(\sigma') \right) = \mathbf{0}_{2n}^{\top}$$

where the last equality holds because

$$\mu(t_i) \cdot \begin{bmatrix} I_n \\ -I_n \end{bmatrix} = \begin{bmatrix} \alpha & \alpha \end{bmatrix} \cdot \begin{bmatrix} I_n \\ -I_n \end{bmatrix} = \mathbf{0}_n^{\top}.$$

Since $\sigma'(t_1, \ldots, t_k)$ is a subtree of $t$, we now have $\mu(t) = \mathbf{0}_{2n}^{\top}$ and thus $f(t) = 0$. □

**Remark 27.** *As $P^n = I_n$, we have that $\mu(\sigma_1)^n = I_{2n}$. Thus for every $f \in \mathcal{C}$, $k \in \{0, 1, \ldots, m\}$, $\sigma \in \Sigma_k \setminus \{\sigma_0, \sigma_1\}$, and $j, i_1, \ldots, i_k \in \mathbb{N}_0$, it holds that $f(\sigma_1^j(\sigma_0)) = f(\sigma_1^{j \bmod n}(\sigma_0))$ and*

$$f(\sigma_1^j(\sigma(\sigma_1^{i_1}(\sigma_0), \ldots, \sigma_1^{i_k}(\sigma_0)))) = f(\sigma_1^{j \bmod n}(\sigma(\sigma_1^{i_1 \bmod n}(\sigma_0), \ldots, \sigma_1^{i_k \bmod n}(\sigma_0)))).$$

Returning to the proof of Theorem 25, let us run the learning algorithm $\mathsf{L}$ on a target $f \in \mathcal{C}$. Lemma 26 (i) and Remark 27 imply that when $\mathsf{L}$ makes a membership query on a tree $t = \sigma_1^j(\sigma_0)$ where $j \in \mathbb{N}_0$, the Teacher returns 1 if $j \bmod n = 0$ and returns 0 otherwise. Furthermore, by Lemma 26 (iii), when $\mathsf{L}$ makes a membership query on $t \in T_\Sigma$ such that $\sum_{\sigma \in \Sigma \setminus \{\sigma_0, \sigma_1\}} \#_\sigma(t) \geq 2$, the Teacher returns 0. In these cases, $\mathsf{L}$ does not gain any new information about $f$ since every function in $\mathcal{C}$ is consistent with the values returned by the Teacher.

When $\mathsf{L}$ makes a membership query on a tree $t = \sigma_1^j(\sigma(\sigma_1^{i_1}(\sigma_0), \ldots, \sigma_1^{i_k}(\sigma_0)))$, where $k \in \{0, 1, \ldots, m\}$, $\sigma \in \Sigma_k \setminus \{\sigma_0, \sigma_1\}$, and $j, i_1, \ldots, i_k \in \mathbb{N}_0$, the Teacher returns an arbitrary number from the field $\mathbb{F}$ if the value $f(t)$ is not already known from an earlier query. It follows from Lemma 26 (ii) and Remark 27 that $\mathsf{L}$ thereby learns the

entry

$$B(\sigma)_{(1+(i_1 \bmod n),\ldots,1+(i_k \bmod n)),(1+n-j) \bmod n}.$$

When $\mathsf{L}$ makes an equivalence query on a hypothesis $h \in \mathcal{C}$ where $h \neq f$, the Teacher finds some entry $B(\sigma)_{(i_1,\ldots,i_k),j}$ that $\mathsf{L}$ does not already know from previous queries and returns the tree $\sigma_1^{1+n-j}(\sigma(\sigma_1^{i_1-1}(\sigma_0),\ldots,\sigma_1^{i_k-1}(\sigma_0)))$ as counterexample.

With each query, the Learner $\mathsf{L}$ learns at most one entry of a matrix $B(\sigma)$ where $\sigma \in \Sigma \setminus \{\sigma_0, \sigma_1\}$. The number of queries made by $\mathsf{L}$ on target $f$ is, therefore, at least the total number of entries of matrices $B(\sigma)$ for all $\sigma \in \Sigma \setminus \{\sigma_0, \sigma_1\}$. The latter number is equal to

$$\sum_{\sigma \in \Sigma \setminus \{\sigma_0,\sigma_1\}} n^{\mathrm{rk}(\sigma)+1} \geq \frac{1}{2^{m+1}} \cdot \sum_{\sigma \in \Sigma \setminus \{\sigma_0,\sigma_1\}} r^{\mathrm{rk}(\sigma)+1} = \frac{1}{2^{m+1}} \cdot \left( \sum_{\sigma \in \Sigma} r^{\mathrm{rk}(\sigma)+1} - r^2 - r \right).$$

This completes the proof. □

The lower bounds of Theorem 24 and Theorem 25 are both linear in the target automaton size. Note that when the alphabet rank is fixed, the lower bound for learning over a fixed field (Theorem 25) is the same, up to a constant factor, as for learning over an arbitrary field (Theorem 24).

Assuming a Teacher that represents counterexamples as succinctly as possible (see §3.4.2 for details), the upper bound of algorithm LMTA from Theorem 23 is quadratic in the target automaton size and, therefore, also quadratic in the lower bound of Theorem 24.

## 3.7 Conclusion

In this chapter, we characterised the query and computational complexity of learning multiplicity tree automata in the exact learning model. We gave the first-known lower

bound on the number of queries needed by any exact learning algorithm to learn a target recognisable tree series. This bound is linear in the size of a smallest multiplicity tree automaton recognising the series. We also gave a new learning algorithm whose query complexity is quadratic in the size of a smallest automaton recognising the target tree series and linear in the size of a largest DAG counterexample provided by the Teacher. With regard to computational complexity, we showed that the problem of deciding equivalence of multiplicity tree automata is logspace equivalent to polynomial identity testing.

The algebraic theory of recognisable word series, notably the connection to finite-rank Hankel matrices, generalises naturally to recognisable tree series and underlies many of the approaches to learning tree automata, including the present chapter (see §3.2 for more details). In the case of trees, however, the issue of succinctness of automaton and counterexample representations comes to the fore. As we have noted in Example 3, the smallest counterexample to the equivalence of two tree automata may be exponential in their total size. Therefore, in order to obtain even a polynomial query complexity, our learning algorithm works with a succinct representation of trees in terms of DAGs. The assumption of a Teacher that provides succinct DAG counterexamples is reasonable in light of the fact that the algorithm of Seidl [1990] for deciding equivalence of multiplicity tree automata can easily be modified to produce DAG counterexamples of minimal size in case of inequivalence.

The issue of succinctness of automaton representations seems to be more subtle and has not been addressed here. Indeed, in this chapter we have used the standard definition of automaton size, in which an automaton with $n$ states and maximum alphabet rank $m$ necessarily has size at least $n^{m+1}$. Adopting a sparse encoding of the transition matrices may result in an exponentially more succinct automaton representation. For instance, tree automata recognising parse trees for a context-free grammar can in general have large maximum alphabet rank $m$ but few transitions.

However, it seems a difficult problem to efficiently learn an automaton of minimal size under a sparse representation of transition matrices. In this regard, note that two different MTAs recognising the same tree series, both with a minimal number of states, can have considerably different sizes under a sparse representation since minimal MTAs are only unique up to change of basis.

One route to obtaining succinct automaton representations in the case of alphabets of unbounded rank is to use the encoding of unranked alphabets into binary alphabets presented by Comon et al. [2007] and Bailly et al. [2010]. Such an encoding would potentially allow to use our learning algorithm to learn recognisable tree series over an arbitrary alphabet $\Sigma$ (including even unranked alphabets) while maintaining hypothesis automaton and Hankel matrix over a binary alphabet. Note though that if the algorithm were required to present its hypotheses to the Teacher as automata over the original alphabet $\Sigma$, then it would need to translate automata over the binary encoding to corresponding automata over $\Sigma$—potentially leading to an exponential blow-up.

With regard to applications of tree-automaton learning algorithms to other problems, we recall that Beimel et al. [2000] apply their exact learning algorithm for multiplicity word automata to show exact learnability of certain classes of polynomials over both finite and infinite fields. Beimel et al. [2000] also prove the learnability of disjoint DNF formulae (i.e., DNF formulae in which each assignment satisfies at most one term) and, more generally, disjoint unions of geometric boxes over finite domains.

The learning framework considered in this chapter concerns multiplicity tree automata, which are strictly more expressive than multiplicity word automata. Moreover, our result on the computational complexity of equivalence testing for multiplicity tree automata shows that, through equivalence queries, the Learner essentially has an oracle for polynomial identity testing. Thus a natural direction for future work is to

seek to apply our algorithm to derive new results on exact learning of other concept classes, such as propositional formulae and polynomials (both in the commutative and noncommutative cases). Particularly relevant to this direction is the work of Klivans and Shpilka [2006] on exact learning of algebraic branching programs and arithmetic circuits and formulae, which relies on rank bounds for Hankel matrices of polynomials in noncommuting variables, obtained by considering a generalised notion of partial derivative. Here one could try to determine whether the extra expressiveness of tree series can be used to show learnability of more general classes of formulae and circuits than have hitherto been handled using learnability of word series.

Sakakibara [1990] showed that context-free grammars (CFGs) can be learned efficiently from their structural descriptions in the exact learning model, using structural membership queries and structural equivalence queries. Specifically, Sakakibara notes that the set of structural descriptions of a context-free grammar constitutes a rational tree language, and thereby reduces the problem of learning a context-free grammar from its structural descriptions to the problem of learning a tree automaton. Given the important role of weighted and probabilistic CFGs across a range of applications including linguistics, a natural next step would be to apply our algorithm to learn weighted CFGs. The idea is to reduce the problem of learning a weighted context-free grammar using structural membership queries and structural equivalence queries to the problem of learning a multiplicity tree automaton in the exact learning model. The basis for applying our algorithm in this setting is the fact that the tree series that maps unlabelled derivation trees to their total weights under a given weighted context-free grammar is recognisable.

# Chapter 4

# Minimization of Multiplicity Tree Automata

**Abstract**

*In this chapter, we consider the problem of minimizing the number of states in a multiplicity tree automaton over the field of rational numbers. We give a minimization algorithm that runs in polynomial time assuming unit-cost arithmetic. We also show that a polynomial bound in the standard Turing model would require a breakthrough in the complexity of polynomial identity testing by proving that the latter problem is logspace equivalent to the decision version of minimization. The developed techniques also improve the state of the art in multiplicity word automata: we give an* NC *algorithm for minimizing multiplicity word automata.*

*Finally, we consider the minimal consistency problem: does there exist a multiplicity automaton with a given number of states that is consistent with a given finite sample of weight-labelled words or trees? We show that, over both words and trees, this decision problem is interreducible with the problem of deciding the truth of existential first-order sentences over the*

*field of rationals—whose decidability is a longstanding open problem.*

*This chapter is based on the material originally published as an extended abstract in [Kiefer et al., 2015] and a full version in [Kiefer et al.].*

---

## 4.1 Introduction

Minimization is a fundamental problem in automata theory, closely related to both learning and equivalence testing. In this chapter we analyse the complexity of minimization for multiplicity automata, i.e., weighted automata over a field, introduced in Chapter 2. Minimization of multiplicity and weighted automata has numerous applications including image compression [Albert and Kari, 2009] and reducing the space complexity of speech recognition tasks [Mohri et al., 1996, Eisner, 2003].

We take a comprehensive view, looking at multiplicity automata over both words and trees and considering both function and decision problems. We also look at the closely-related problem of obtaining a minimal automaton consistent with a given finite set of observations. We characterise the complexity of these problems in terms of arithmetic and Boolean circuit classes. In particular, we give relationships to longstanding open problems in arithmetic complexity theory.

The minimization problem for multiplicity word automata has long been known to be solvable in polynomial time (in the Turing model) [Schützenberger, 1961, Tzeng, 1992]. In this chapter, we give a new procedure for computing minimal word automata and thereby place minimization in NC, improving also on a randomized NC procedure of Kiefer et al. [2013]. We recall that $\mathrm{NL} \subseteq \mathrm{NC} \subseteq \mathrm{P}$, where NC comprises those languages having L-uniform Boolean circuits of polylogarithmic depth and polynomial size, or, equivalently, those problems solvable in polylogarithmic time on parallel random-access machines with polynomially many processors.

By comparison, it is known that minimizing deterministic word automata is NL-complete [Cho and Huynh, 1992], while minimizing nondeterministic word automata is PSPACE-complete [Jiang and Ravikumar, 1993]. The latter result shows, in particular, that the bounds obtained in this chapter over $\mathbb{Q}$ do not apply to weighted automata over an arbitrary semi-ring, because nondeterministic automata can be viewed as weighted automata over the Boolean semi-ring.

Over trees, we give what is (to the best of our knowledge) the first complexity analysis of the problem of minimizing multiplicity automata. We present an algorithm that minimizes a given multiplicity tree automaton $\mathcal{A}$ in time $O\left(|\mathcal{A}|^2 \cdot r\right)$, where $|\mathcal{A}|$ is the size of $\mathcal{A}$ and $r$ is the maximum alphabet rank, assuming unit-cost arithmetic. This procedure can be viewed as a concrete version of the construction of a syntactic algebra of a recognisable tree series by Bozapalidis [1991]. We thus place the problem within PSPACE in the conventional Turing model, since a polynomial-time decidable problem in the unit-cost model lies in PSPACE [see, e.g., Allender et al., 2009].

We are moreover able to precisely characterise the complexity of the decision version of the minimization problem, showing that it is logspace equivalent to the arithmetic circuit identity testing (ACIT) problem, commonly also called the polynomial identity testing problem. Obtaining this complexity bound requires departing from the framework of Bozapalidis [1991]. In Chapter 3 we reduced equivalence testing of multiplicity tree automata to ACIT; the advance here is to reduce the more general problem of minimization also to ACIT.

Lastly, we consider the problem of computing a minimal multiplicity automaton consistent with a finite set of input-output behaviours. This is a natural learning problem whose complexity for deterministic finite automata was studied by Gold [1978], who showed that the problem of exactly identifying the smallest deterministic finite automaton consistent with a set of accepted and rejected words is NP-hard. For multiplicity word automata over the field $\mathbb{Q}$, we show that the decision version of this

problem, which we call the *minimal consistency problem*, is logspace equivalent to the problem of deciding the truth of existential first-order sentences over the structure $(\mathbb{Q}, +, \cdot, 0, 1)$, a longstanding open problem [see Poonen, 2003]. We observe that, by contrast, the minimal consistency problem for multiplicity word automata over the field $\mathbb{R}$ is in PSPACE, and likewise for multiplicity tree automata over $\mathbb{R}$ that have a fixed alphabet rank.

## 4.2 Related Work

Based on a generalisation of the Myhill-Nerode theorem to trees, one obtains a procedure for minimizing deterministic tree automata that runs in time quadratic in the size of the input automaton [Brainerd, 1968, Carrasco et al., 2007]. There have also been several works on minimizing deterministic tree automata with weights in a semi-field (i.e., a semi-ring with multiplicative inverses). In particular, Maletti [2009] gives a polynomial-time algorithm in this setting, assuming unit cost for arithmetic in the semi-field.

In the nondeterministic case, Carme et al. [2003] define the subclass of *residual finite* nondeterministic tree automata. They show that this class expresses the class of regular tree languages and admits a polynomial-space minimization procedure.

Recently, Balle et al. [2015] showed that, for any multiplicity word automaton computing an absolutely convergent series, there is a *canonical* minimal automaton whose forward-backward (rank) factorization of the Hankel matrix $H$ coincides with the rank factorization $H = (UD^{1/2})(VD^{1/2})^\top$ induced by the singular value decomposition $H = UDV^\top$. Moreover, they give an approximate minimization algorithm based on truncating the canonical automaton. This approximate-minimization approach was generalised to tree automata by Rabusseau et al. [2016], who showed that any multiplicity tree automaton computing a function that satisfies a strong

convergence property has a canonical minimal automaton whose forward-backward factorization coincides with the rank factorization induced by the SVD of $H$.

## 4.3 Row and Column Spaces

Given a field $\mathbb{F}$ and a set $S \subseteq \mathbb{F}^n$, we use $\langle S \rangle$ to denote the vector subspace of $\mathbb{F}^n$ that is spanned by $S$, where we often omit the braces when denoting $S$. We think of vectors in $\mathbb{F}^n$ as column vectors.

For the remainder of this section, let $\mathbb{F}$ be either the field of rationals $\mathbb{Q}$ or the field of reals $\mathbb{R}$. Let $A$ be an $m \times n$ matrix with entries in $\mathbb{F}$. The *row space* of $A$, written as $\mathrm{Row}(A)$, is the subspace of $\mathbb{F}^{1\times n}$ spanned by the rows of $A$. The *column space* of $A$, written as $\mathrm{Col}(A)$, is the subspace of $\mathbb{F}^{m\times 1}$ spanned by the columns of $A$. That is, $\mathrm{Row}(A) = \langle v^\top \cdot A : v \in \mathbb{F}^m \rangle$ and $\mathrm{Col}(A) = \langle A \cdot v : v \in \mathbb{F}^n \rangle$.

The following Lemmas 28-30 contain some basic results about row and column spaces that we will use in this chapter.

**Lemma 28.** *Let $A_1, A_2$ be matrices such that $\mathrm{Row}(A_1) \subseteq \mathrm{Row}(A_2)$. For any matrix $B$ such that $A_1 \cdot B$ (and thus also $A_2 \cdot B$) is defined, we have that*

$$\mathrm{Row}(A_1 \cdot B) \subseteq \mathrm{Row}(A_2 \cdot B).$$

*Proof.* Suppose $A_1 \in \mathbb{F}^{m_1\times n}$ and $A_2 \in \mathbb{F}^{m_2\times n}$. For every vector $v_1 \in \mathbb{F}^{m_1}$, it holds that $v_1^\top \cdot A_1 \in \mathrm{Row}(A_1) \subseteq \mathrm{Row}(A_2)$. Hence, there exists a vector $v_2 \in \mathbb{F}^{m_2}$ such that $v_1^\top \cdot A_1 = v_2^\top \cdot A_2$. Thus

$$\mathrm{Row}(A_1 \cdot B) = \langle v_1^\top \cdot A_1 \cdot B : v_1 \in \mathbb{F}^{m_1} \rangle \subseteq \langle v_2^\top \cdot A_2 \cdot B : v_2 \in \mathbb{F}^{m_2} \rangle = \mathrm{Row}(A_2 \cdot B),$$

which completes the proof. □

**Lemma 29.** *For any matrix $A \in \mathbb{F}^{m\times n}$, it holds that $\mathrm{Row}(A^\top A) = \mathrm{Row}(A)$.*

*Proof.* For any $v \in \mathbb{F}^n$ such that $(A^\top A)v = \mathbf{0}_n$ we have

$$(Av)^\top Av \;=\; v^\top A^\top Av \;=\; v^\top \mathbf{0}_n \;=\; 0\,,$$

and hence $Av = \mathbf{0}_m$. Conversely, for any $v \in \mathbb{F}^n$ with $Av = \mathbf{0}_m$ we have $(A^\top A)v = \mathbf{0}_n$. Therefore, matrices $A$ and $A^\top A$ have the same null space and hence the same row space. □

**Lemma 30.** *Let $A_1$, $A_2$, $B_1$, $B_2$ be matrices of dimension $n_1 \times m$, $n_2 \times m$, $m \times n_3$, $m \times n_4$, respectively. If $\mathrm{Row}(A_1) = \mathrm{Row}(A_2)$ and $\mathrm{Col}(B_1) = \mathrm{Col}(B_2)$, then*

$$\mathrm{rank}(A_1 \cdot B_1) = \mathrm{rank}(A_2 \cdot B_2).$$

*Proof.* By definition of rank as the dimension of row or column space, we have

$$\begin{aligned}
\mathrm{rank}(A_1 \cdot B_1) &= \dim \langle v^\top \cdot A_1 \cdot B_1 : v \in \mathbb{F}^{n_1} \rangle \\
&= \dim \langle v^\top \cdot A_2 \cdot B_1 : v \in \mathbb{F}^{n_2} \rangle \qquad (\text{using } \mathrm{Row}(A_1) = \mathrm{Row}(A_2)) \\
&= \dim \langle A_2 \cdot B_1 \cdot v : v \in \mathbb{F}^{n_3} \rangle \\
&= \dim \langle A_2 \cdot B_2 \cdot v : v \in \mathbb{F}^{n_4} \rangle \qquad (\text{using } \mathrm{Col}(B_1) = \mathrm{Col}(B_2)) \\
&= \mathrm{rank}(A_2 \cdot B_2).
\end{aligned}$$

This completes the proof. □

## 4.4 Fundamentals of Minimization

In this section, we prepare the ground for minimization algorithms. Let us fix a field $\mathbb{F}$ for the rest of this section and assume that all automata are over $\mathbb{F}$. We also fix an MTA $\mathcal{A} = (n, \Sigma, \mu, \gamma)$ for the rest of the section. We will construct from $\mathcal{A}$ another MTA $\tilde{\mathcal{A}}$ which we show to be equivalent to $\mathcal{A}$ and minimal. A crucial ingredient for

this construction are special vector spaces induced by $\mathcal{A}$, called the forward space and the backward space.

### 4.4.1 Forward and Backward Space

The *forward space* $\mathcal{F}$ of $\mathcal{A}$ is the (row) vector space $\mathcal{F} := \langle \mu(t) : t \in T_\Sigma \rangle$ over $\mathbb{F}$. The *backward space* $\mathcal{B}$ of $\mathcal{A}$ is the (column) vector space $\mathcal{B} := \langle \mu(c) \cdot \gamma : c \in C_\Sigma \rangle$ over $\mathbb{F}$. The following Propositions 31 and 32 provide fundamental characterisations of $\mathcal{F}$ and $\mathcal{B}$, respectively.

**Proposition 31.** *The forward space $\mathcal{F}$ has the following properties:*

(a) *The forward space $\mathcal{F}$ is the smallest vector space $V$ over $\mathbb{F}$ such that for all $k \in \mathbb{N}_0$, $v_1, \ldots, v_k \in V$, and $\sigma \in \Sigma_k$ it holds that $(v_1 \otimes \cdots \otimes v_k) \cdot \mu(\sigma) \in V$.*

(b) *The set of row vectors $\{\mu(t) : t \in T_\Sigma^{\leq n}\}$ spans $\mathcal{F}$.*

*Proof.* We start by proving result (a). Here we first show that $\mathcal{F}$ has the closure property stated in (a). To this end, let us take any $k \in \mathbb{N}_0$, $v_1, \ldots, v_k \in \mathcal{F}$, and $\sigma \in \Sigma_k$. By definition of $\mathcal{F}$, for every $i \in [k]$ we can express vector $v_i \in \mathcal{F}$ as

$$v_i = \sum_{j_i=1}^{m_i} \alpha_{j_i}^i \mu(t_{j_i}^i)$$

for some integer $m_i \in \mathbb{N}$, scalars $\alpha_1^i, \ldots, \alpha_{m_i}^i \in \mathbb{F}$, and trees $t_1^i, \ldots, t_{m_i}^i \in T_\Sigma$. From here, using bilinearity of Kronecker product we get that

$$\begin{aligned}
(v_1 \otimes \cdots \otimes v_k) \cdot \mu(\sigma) &= \left( \left( \sum_{j_1=1}^{m_1} \alpha_{j_1}^1 \mu(t_{j_1}^1) \right) \otimes \cdots \otimes \left( \sum_{j_k=1}^{m_k} \alpha_{j_k}^k \mu(t_{j_k}^k) \right) \right) \cdot \mu(\sigma) \\
&= \sum_{j_1=1}^{m_1} \cdots \sum_{j_k=1}^{m_k} \alpha_{j_1}^1 \cdots \alpha_{j_k}^k \left( \mu(t_{j_1}^1) \otimes \cdots \otimes \mu(t_{j_k}^k) \right) \cdot \mu(\sigma) \\
&= \sum_{j_1=1}^{m_1} \cdots \sum_{j_k=1}^{m_k} \alpha_{j_1}^1 \cdots \alpha_{j_k}^k \cdot \mu(\sigma(t_{j_1}^1, \ldots, t_{j_k}^k)).
\end{aligned}$$

Since $\mathcal{F}$ is a vector space, the above equation implies that $(v_1 \otimes \cdots \otimes v_k) \cdot \mu(\sigma) \in \mathcal{F}$.

Let $V$ be any vector space over $\mathbb{F}$ such that for all $k \in \mathbb{N}_0$, $v_1, \ldots, v_k \in V$, and $\sigma \in \Sigma_k$ it holds that $(v_1 \otimes \cdots \otimes v_k) \cdot \mu(\sigma) \in V$. We claim that $\mathcal{F} \subseteq V$. To prove this, it suffices to show that $\mu(t) \in V$ for every $t \in T_\Sigma$. Here we give a proof by induction on $\mathit{height}(t)$. The base case $t \in \Sigma_0$ is trivial. For the induction step, let $h \in \mathbb{N}_0$ and assume that $\mu(t) \in V$ for all $t \in T_\Sigma^{\leq h}$. Take any $t \in T_\Sigma^{h+1}$. Then, $t = \sigma(t_1, \ldots, t_k)$ for some $k \geq 1$, $\sigma \in \Sigma_k$, and $t_1, \ldots, t_k \in T_\Sigma^{\leq h}$. The induction hypothesis now implies that $\mu(t_1), \ldots, \mu(t_k) \in V$. By the choice of $V$, we therefore have that $\mu(t) = (\mu(t_1) \otimes \cdots \otimes \mu(t_k)) \cdot \mu(\sigma) \in V$. This completes the proof by induction.

The proof of result (b) follows from [Seidl, 1990, Main Lemma 4.1]. □

**Proposition 32.** *Let $S \subseteq T_\Sigma$ be a set of trees such that $\{\mu(t) : t \in S\}$ spans $\mathcal{F}$. The backward space $\mathcal{B}$ has the following properties:*

(a) *The backward space $\mathcal{B}$ is the smallest vector space $V$ over $\mathbb{F}$ such that:*

(P1) $\gamma \in V$.

(P2) *For every $v \in V$ and $c \in C^1_{\Sigma,S}$ it holds that $\mu(c) \cdot v \in V$.*

(b) *The set of column vectors $\{\mu(c) \cdot \gamma : c \in C^{<n}_{\Sigma,S}\}$ spans $\mathcal{B}$.*

*Proof.* First, we prove result (a). We have that $\gamma = \mu(\Box) \cdot \gamma \in \mathcal{B}$, hence $\mathcal{B}$ satisfies property (P1). To see that $\mathcal{B}$ satisfies property (P2), let us take any $v \in \mathcal{B}$ and $c \in C^1_{\Sigma,S}$. By definition of $\mathcal{B}$, vector $v$ can be expressed as

$$v = \sum_{i=1}^{m} \alpha_i \cdot \mu(c_i) \cdot \gamma$$

for some integer $m \in \mathbb{N}$, scalars $\alpha_1, \ldots, \alpha_m \in \mathbb{F}$, and contexts $c_1, \ldots, c_m \in C_\Sigma$. Thus by bilinearity of matrix multiplication and Equation (2.5) we have

$$\mu(c) \cdot v = \mu(c) \cdot \left( \sum_{i=1}^{m} \alpha_i \cdot \mu(c_i) \cdot \gamma \right) = \sum_{i=1}^{m} \alpha_i \cdot (\mu(c) \cdot \mu(c_i) \cdot \gamma) = \sum_{i=1}^{m} \alpha_i \cdot \mu(c_i[c]) \cdot \gamma,$$

which implies that $\mu(c) \cdot v \in \mathcal{B}$ since $\mathcal{B}$ is a vector space. Therefore, $\mathcal{B}$ satisfies properties (P1) and (P2).

Let now $V$ be any vector space over $\mathbb{F}$ satisfying properties (P1) and (P2). In order to show that $\mathcal{B} \subseteq V$, it suffices to show that $\mu(c) \cdot \gamma \in V$ for every $c \in C_\Sigma$. We prove the latter result using induction on the distance between the root and the $\Box$-labelled node of $c$. For the induction basis, let the distance be 0, i.e., $c = \Box$. Then we have $\mu(c) \cdot \gamma = \gamma \in V$ by property (P1). For the induction step, let $h \in \mathbb{N}_0$ and assume that $\mu(c) \cdot \gamma \in V$ for all $c \in C_\Sigma^{\leq h}$. Take any $c \in C_\Sigma^{h+1}$. Let $c' \in C_\Sigma^1$ and $c'' \in C_\Sigma^h$ be such that $c = c''[c']$. Without loss of generality we can assume that $c' = \sigma(\Box, \tau_2, \ldots, \tau_k)$ where $k \geq 1$, $\sigma \in \Sigma_k$, and $\tau_2, \ldots, \tau_k \in T_\Sigma$. Since $\mathcal{F} = \langle \mu(t) : t \in S \rangle$, for every $i \in \{2, \ldots, k\}$ there is an integer $m_i \in \mathbb{N}$, scalars $\alpha_1^i, \ldots, \alpha_{m_i}^i \in \mathbb{F}$, and trees $t_1^i, \ldots, t_{m_i}^i \in S$ such that

$$\mu(\tau_i) = \sum_{j_i=1}^{m_i} \alpha_{j_i}^i \mu(t_{j_i}^i).$$

From here, using bilinearity of Kronecker product, it follows that

$$\begin{aligned}
\mu(c) \cdot \gamma &= \mu(c') \cdot \mu(c'') \cdot \gamma \\
&= (I_n \otimes \mu(\tau_2) \otimes \cdots \otimes \mu(\tau_k)) \mu(\sigma) \cdot \mu(c'') \cdot \gamma \\
&= \left( I_n \otimes \left( \sum_{j_2=1}^{m_2} \alpha_{j_2}^2 \mu(t_{j_2}^2) \right) \otimes \cdots \otimes \left( \sum_{j_k=1}^{m_k} \alpha_{j_k}^k \mu(t_{j_k}^k) \right) \right) \mu(\sigma) \cdot \mu(c'') \cdot \gamma \\
&= \sum_{j_2=1}^{m_2} \cdots \sum_{j_k=1}^{m_k} \alpha_{j_2}^2 \cdots \alpha_{j_k}^k \cdot \left( I_n \otimes \mu(t_{j_2}^2) \otimes \cdots \otimes \mu(t_{j_k}^k) \right) \mu(\sigma) \cdot \mu(c'') \cdot \gamma \\
&= \sum_{j_2=1}^{m_2} \cdots \sum_{j_k=1}^{m_k} \alpha_{j_2}^2 \cdots \alpha_{j_k}^k \cdot \mu(\sigma(\Box, t_{j_2}^2, \ldots, t_{j_k}^k)) \cdot \mu(c'') \cdot \gamma \,,
\end{aligned}$$

where we note that $\sigma(\Box, t_{j_2}^2, \ldots, t_{j_k}^k) \in C_{\Sigma,S}^1$ for every $j_2 \in [m_2], \ldots, j_k \in [m_k]$. Moreover, we have $\mu(c'') \cdot \gamma \in V$ by the induction hypothesis. Thus by property (P2) we have $\mu(\sigma(\Box, t_{j_2}^2, \ldots, t_{j_k}^k)) \cdot \mu(c'') \cdot \gamma \in V$ for every $j_2 \in [m_2], \ldots, j_k \in [m_k]$. Since $V$ is

a vector space, we conclude that $\mu(c) \cdot \gamma \in V$. This completes the proof of result (a) by induction.

We denote by $C_{\Sigma,S}$ the set of all $c \in C_\Sigma$ where every subtree of $c$ is an element of $S$. It follows easily from part (a) that $\langle \mu(c) \cdot \gamma : c \in C_{\Sigma,S} \rangle = \mathcal{B}$ since $\langle \mu(c) \cdot \gamma : c \in C_{\Sigma,S} \rangle$ satisfies properties (P1) and (P2). Thus in order to prove result (b), it suffices to show that the set $\{\mu(c) \cdot \gamma : c \in C^{\leq n}_{\Sigma,S}\}$ spans $\langle \mu(c) \cdot \gamma : c \in C_{\Sigma,S} \rangle$. We show this using an argument that was similarly given, e.g., by Paz [1971]. If $\gamma$ is the zero vector $\mathbf{0}_n$, the statement is trivial. Let us now assume that $\gamma \neq \mathbf{0}_n$. For every $i \in \mathbb{N}$, we define the vector space $\mathcal{B}^i := \langle \mu(c) \cdot \gamma : c \in C^{\leq i}_{\Sigma,S} \rangle$ over $\mathbb{F}$. Since $\mathcal{B}^i$ is a subspace of $\mathcal{B}^{i+1}$ for every $i \in \mathbb{N}$, we have

$$1 \;\leq\; \dim(\mathcal{B}^1) \;\leq\; \dim(\mathcal{B}^2) \;\leq\; \cdots \;\leq\; \dim(\mathcal{B}^{n+1}) \;\leq\; n \,, \tag{4.1}$$

where the first inequality holds because $\gamma \neq \mathbf{0}_n$, and the last inequality holds because $\mathcal{B}^i \subseteq \mathbb{F}^n$ for all $i \in \mathbb{N}$. Not all inequalities in the inequality chain (4.1) can be strict, so we must have $\mathcal{B}^{i_0} = \mathcal{B}^{i_0+1}$ for some $i_0 \in [n]$. We claim that $\mathcal{B}^i = \mathcal{B}^{i+1}$ for all $i \geq i_0$. We give a proof by induction on $i$. The base case $i = i_0$ holds by definition of $i_0$. For the induction step, let $i \geq i_0$ and assume that $\mathcal{B}^i = \mathcal{B}^{i+1}$. Note that, by definition, for all $j \in \mathbb{N}$ we have $\mathcal{B}^{j+1} = \langle \gamma, \mu(c) \cdot \mathcal{B}^j : c \in C^1_{\Sigma,S} \rangle$. Using this result for $j \in \{i, i+1\}$, we obtain:

$$\mathcal{B}^{i+1} \;=\; \langle \gamma, \mu(c) \cdot \mathcal{B}^i : c \in C^1_{\Sigma,S} \rangle \;=\; \langle \gamma, \mu(c) \cdot \mathcal{B}^{i+1} : c \in C^1_{\Sigma,S} \rangle \;=\; \mathcal{B}^{i+2}$$

where the middle equation holds by the induction hypothesis. This completes the proof by induction, and we thus conclude that $\mathcal{B}^i = \mathcal{B}^{i+1}$ for all $i \geq i_0$. Since $n \geq i_0$, it follows that $\mathcal{B}^n = \bigcup_{i \geq n} \mathcal{B}^i$. Since $(\mathcal{B}^i)_{i \in \mathbb{N}}$ is an increasing sequence of vector spaces, we have $\mathcal{B}^n = \bigcup_{i \in \mathbb{N}} \mathcal{B}^i = \langle \mu(c) \cdot \gamma : c \in C_{\Sigma,S} \rangle$ as required. □

### 4.4.2 A Minimal Automaton

We recall that $\mathcal{F}$ is the forward space and $\mathcal{B}$ is the backward space of the automaton $\mathcal{A}$. Let $F$ and $B$ be matrices whose rows and columns, respectively, span $\mathcal{F}$ and $\mathcal{B}$. That is, $\mathrm{Row}(F) = \mathcal{F}$ and $\mathrm{Col}(B) = \mathcal{B}$. We discuss later (Section 4.5.1) how to efficiently compute $F$ and $B$. The following lemma states that $\mathrm{rank}(F \cdot B)$ is the dimension of a minimal automaton equivalent to $\mathcal{A}$.

**Lemma 33.** *A minimal automaton equivalent to $\mathcal{A}$ has $m := \mathrm{rank}(F \cdot B)$ states.*

*Proof.* Let $H$ be the Hankel matrix of $\|\mathcal{A}\|$. Define the matrix $\overline{F} \in \mathbb{F}^{T_\Sigma \times [n]}$ where $\overline{F}_{t,:} = \mu(t)$ for every $t \in T_\Sigma$. Define the matrix $\overline{B} \in \mathbb{F}^{[n] \times C_\Sigma}$ where $\overline{B}_{:,c} = \mu(c) \cdot \gamma$ for every $c \in C_\Sigma$. For every $t \in T_\Sigma$ and $c \in C_\Sigma$ we have by the definitions that

$$H_{t,c} = \|\mathcal{A}\|(c[t]) = \mu(c[t]) \cdot \gamma = \mu(t) \cdot \mu(c) \cdot \gamma = \overline{F}_{t,:} \cdot \overline{B}_{:,c} \,,$$

hence $H = \overline{F} \cdot \overline{B}$. Note that

$$\mathrm{Row}(\overline{F}) = \mathcal{F} = \mathrm{Row}(F) \qquad \text{and} \qquad \mathrm{Col}(\overline{B}) = \mathcal{B} = \mathrm{Col}(B)\,. \tag{4.2}$$

We now have:

$$\mathrm{rank}(H) = \mathrm{rank}(\overline{F} \cdot \overline{B}) = \mathrm{rank}(F \cdot B) = m,$$

where the second equality holds by (4.2) and Lemma 30. Theorem 4 thus implies that a minimal automaton equivalent to $\mathcal{A}$ has $\mathrm{rank}(H) = m$ states. □

Since $m = \mathrm{rank}(F \cdot B)$, there exist $m$ rows of $F \cdot B$ that span $\mathrm{Row}(F \cdot B)$. The corresponding $m$ rows of $F$ form a matrix $\tilde{F} \in \mathbb{F}^{m \times n}$ with $\mathrm{Row}(\tilde{F} \cdot B) = \mathrm{Row}(F \cdot B)$. Define a multiplicity tree automaton $\tilde{\mathcal{A}} = (m, \Sigma, \tilde{\mu}, \tilde{\gamma})$ with $\tilde{\gamma} = \tilde{F} \cdot \gamma$ and

$$\tilde{\mu}(\sigma) \cdot \tilde{F} \cdot B = \tilde{F}^{\otimes k} \cdot \mu(\sigma) \cdot B \qquad \text{for every } \sigma \in \Sigma_k. \tag{4.3}$$

We show that $\tilde{\mathcal{A}}$ minimizes $\mathcal{A}$:

**Proposition 34.** *The MTA $\tilde{\mathcal{A}}$ is well-defined and is a minimal automaton equivalent to $\mathcal{A}$.*

Before giving a full proof of Proposition 34 later in this subsection, we now prove this result for multiplicity *word* automata, stated as Proposition 35 below, which will be used in Section 4.5.2. The main arguments for the tree case are similar, but slightly more involved.

Let $\mathcal{A} = (n, \Sigma, \mu, \alpha, \gamma)$ be an MWA. The forward and backward space can then be written as $\mathcal{F} = \langle \alpha \cdot \mu(w) : w \in \Sigma^* \rangle$ and $\mathcal{B} = \langle \mu(w) \cdot \gamma : w \in \Sigma^* \rangle$, respectively. The MWA $\tilde{\mathcal{A}}$ can be written as $\tilde{\mathcal{A}} = (m, \Sigma, \tilde{\mu}, \tilde{\alpha}, \tilde{\gamma})$ with $\tilde{\gamma} = \tilde{F} \cdot \gamma$,

$$\tilde{\alpha} \cdot \tilde{F} \cdot B = \alpha \cdot B, \qquad \text{and} \tag{4.4}$$

$$\tilde{\mu}(\sigma) \cdot \tilde{F} \cdot B = \tilde{F} \cdot \mu(\sigma) \cdot B \qquad \text{for every } \sigma \in \Sigma. \tag{4.5}$$

**Proposition 35.** *The MWA $\tilde{\mathcal{A}}$ is well-defined and is a minimal automaton equivalent to $\mathcal{A}$.*

First, we show that $\tilde{\mathcal{A}}$ is a well-defined multiplicity word automaton:

**Lemma 36.** *There exists a unique vector $\tilde{\alpha}$ satisfying Equation* (4.4)*. For every $\sigma \in \Sigma$, there exists a unique matrix $\tilde{\mu}(\sigma)$ satisfying Equation* (4.5)*.*

*Proof.* Since the rows of $\tilde{F} \cdot B$ form a basis of $\mathrm{Row}(F \cdot B)$, it suffices to prove that $\alpha \cdot B \in \mathrm{Row}(F \cdot B)$ and $\mathrm{Row}(\tilde{F} \cdot \mu(\sigma) \cdot B) \subseteq \mathrm{Row}(F \cdot B)$ for every $\sigma \in \Sigma$. By Lemma 28, it further suffices to prove that $\alpha \in \mathrm{Row}(F)$ and $\mathrm{Row}(\tilde{F} \cdot \mu(\sigma)) \subseteq \mathrm{Row}(F)$ for every $\sigma \in \Sigma$.

We have $\alpha = \alpha \cdot \mu(\varepsilon) \in \mathcal{F} = \mathrm{Row}(F)$. Let $i \in [m]$. Since $\tilde{F}_{i,:} \in \mathrm{Row}(F) = \mathcal{F}$, it follows from Proposition 31 (a) that $(\tilde{F} \cdot \mu(\sigma))_{i,:} = \tilde{F}_{i,:} \cdot \mu(\sigma) \in \mathcal{F}$ for all $\sigma \in \Sigma$. □

We complete the proof of Proposition 35 by showing that MWA $\tilde{\mathcal{A}}$ minimizes $\mathcal{A}$:

**Lemma 37.** *The automaton $\tilde{\mathcal{A}}$ is a minimal MWA equivalent to $\mathcal{A}$.*

*Proof.* We claim that for every $w \in \Sigma^*$,

$$\tilde{\alpha} \cdot \tilde{\mu}(w) \cdot \tilde{F} \cdot B = \alpha \cdot \mu(w) \cdot B. \tag{4.6}$$

Our proof is by induction on $|w|$. For the base case $w = \varepsilon$, we have

$$\tilde{\alpha} \cdot \tilde{\mu}(\varepsilon) \cdot \tilde{F} \cdot B = \tilde{\alpha} \cdot \tilde{F} \cdot B \overset{\text{Eq. (4.4)}}{=} \alpha \cdot B = \alpha \cdot \mu(\varepsilon) \cdot B.$$

For the induction step, let $l \in \mathbb{N}_0$ and assume that (4.6) holds for every $w \in \Sigma^l$. Take any $w \in \Sigma^l$ and $\sigma \in \Sigma$. For every $b \in \mathcal{B} = \mathrm{Col}(B)$ we have by Proposition 32 (a) that $\mu(\sigma) \cdot b \in \mathcal{B}$, and thus by Equation (2.1) and the induction hypothesis for $w \in \Sigma^l$ it follows

$$\begin{aligned} \tilde{\alpha} \cdot \tilde{\mu}(w\sigma) \cdot \tilde{F} \cdot b = \tilde{\alpha} \cdot \tilde{\mu}(w) \cdot \tilde{\mu}(\sigma) \cdot \tilde{F} \cdot b &\overset{\text{Eq. (4.5)}}{=} \tilde{\alpha} \cdot \tilde{\mu}(w) \cdot \tilde{F} \cdot \mu(\sigma) \cdot b \\ &= \alpha \cdot \mu(w) \cdot \mu(\sigma) \cdot b = \alpha \cdot \mu(w\sigma) \cdot b \end{aligned}$$

which completes the proof by induction.

Now for any $w \in \Sigma^*$, since $\gamma \in \mathcal{B}$ we have

$$\|\tilde{\mathcal{A}}\|(w) = \tilde{\alpha} \cdot \tilde{\mu}(w) \cdot \tilde{\gamma} = \tilde{\alpha} \cdot \tilde{\mu}(w) \cdot \tilde{F} \cdot \gamma \overset{\text{Eq. (4.6)}}{=} \alpha \cdot \mu(w) \cdot \gamma = \|\mathcal{A}\|(w).$$

Hence, MWAs $\tilde{\mathcal{A}}$ and $\mathcal{A}$ are equivalent. Minimality of $\tilde{\mathcal{A}}$ follows from Lemma 33. □

We are now ready to prove Proposition 34 in its full generality. The proof is split in two lemmas, Lemmas 38 and 39, which together imply Proposition 34. First, we show that $\tilde{\mathcal{A}}$ is a well-defined multiplicity tree automaton:

**Lemma 38.** *For every $\sigma \in \Sigma_k$, there exists a unique matrix $\tilde{\mu}(\sigma)$ satisfying Equation* (4.3).

*Proof.* Since the rows of $\tilde{F} \cdot B$ form a basis of $\mathrm{Row}(F \cdot B)$, it suffices to prove that

$$\mathrm{Row}(\tilde{F}^{\otimes k} \cdot \mu(\sigma) \cdot B) \subseteq \mathrm{Row}(F \cdot B).$$

By Lemma 28, to do this it suffices to prove that $\mathrm{Row}(\tilde{F}^{\otimes k} \cdot \mu(\sigma)) \subseteq \mathrm{Row}(F)$. Let us therefore take an arbitrary row $(\tilde{F}^{\otimes k} \cdot \mu(\sigma))_{(i_1,\ldots,i_k),:}$ of $\tilde{F}^{\otimes k} \cdot \mu(\sigma)$, where $(i_1, \ldots, i_k) \in [m]^k$. We have

$$(\tilde{F}^{\otimes k} \cdot \mu(\sigma))_{(i_1,\ldots,i_k),:} = (\tilde{F}^{\otimes k})_{(i_1,\ldots,i_k),:} \cdot \mu(\sigma) \overset{\text{Eq. (2.3)}}{=} (\tilde{F}_{i_1,:} \otimes \cdots \otimes \tilde{F}_{i_k,:}) \cdot \mu(\sigma).$$

Since $\tilde{F}_{i_1,:}, \ldots, \tilde{F}_{i_k,:} \in \mathrm{Row}(\tilde{F}) \subseteq \mathrm{Row}(F) = \mathcal{F}$, by Proposition 31 (a) we have that $(\tilde{F}_{i_1,:} \otimes \cdots \otimes \tilde{F}_{i_k,:}) \cdot \mu(\sigma) \in \mathcal{F}$. Therefore, $(\tilde{F}^{\otimes k} \cdot \mu(\sigma))_{(i_1,\ldots,i_k),:} \in \mathcal{F} = \mathrm{Row}(F)$. □

Next, we show that MTA $\tilde{\mathcal{A}}$ minimizes $\mathcal{A}$:

**Lemma 39.** *The automaton $\tilde{\mathcal{A}}$ is a minimal MTA equivalent to $\mathcal{A}$.*

*Proof.* First we show that for every $t \in T_\Sigma$,

$$\tilde{\mu}(t) \cdot \tilde{F} \cdot B = \mu(t) \cdot B. \tag{4.7}$$

Our proof is by induction on $\mathit{height}(t)$. The base case $t = \sigma \in \Sigma_0$ follows immediately from Equation (4.3). For the induction step, let $h \in \mathbb{N}_0$ and assume that (4.7) holds for every $t \in T_\Sigma^{\leq h}$. Take any tree $t \in T_\Sigma^{h+1}$. Then $t = \sigma(t_1, \ldots, t_k)$ for some $k \geq 1$, $\sigma \in \Sigma_k$, and $t_1, \ldots, t_k \in T_\Sigma^{\leq h}$. Using bilinearity of Kronecker product we get that

$$\begin{aligned}
\tilde{\mu}(t) \cdot \tilde{F} \cdot B &= (\tilde{\mu}(t_1) \otimes \cdots \otimes \tilde{\mu}(t_k)) \cdot \tilde{\mu}(\sigma) \cdot \tilde{F} \cdot B \\
&= (\tilde{\mu}(t_1) \otimes \cdots \otimes \tilde{\mu}(t_k)) \cdot \tilde{F}^{\otimes k} \cdot \mu(\sigma) \cdot B && \text{by Eq. (4.3)} \\
&= ((\tilde{\mu}(t_1)\tilde{F}) \otimes \cdots \otimes (\tilde{\mu}(t_k)\tilde{F})) \cdot \mu(\sigma) \cdot B && \text{by Eq. (2.4)} \\
&= (\tilde{\mu}(t_1)\tilde{F}) \cdot (I_n \otimes (\tilde{\mu}(t_2)\tilde{F}) \otimes \cdots \otimes (\tilde{\mu}(t_k)\tilde{F})) \cdot \mu(\sigma) \cdot B.
\end{aligned}$$

Since $\mathrm{Row}(\tilde{F}) \subseteq \mathcal{F}$, for every $i \in \{2, \ldots, k\}$ it holds that $\tilde{\mu}(t_i)\tilde{F} \in \mathcal{F}$. Since $I_n = \mu(\square) \in \mathcal{F}$, we now have that $(I_n \otimes (\tilde{\mu}(t_2)\tilde{F}) \otimes \cdots \otimes (\tilde{\mu}(t_k)\tilde{F})) \cdot \mu(\sigma) \cdot B \in \mathcal{B}$ by Proposition 32 (a). Thus by the induction hypothesis for $t_1 \in T_{\Sigma}^{\leq h}$, we have

$$\begin{aligned}\tilde{\mu}(t) \cdot \tilde{F} \cdot B &= \mu(t_1) \cdot (I_n \otimes (\tilde{\mu}(t_2)\tilde{F}) \otimes \cdots \otimes (\tilde{\mu}(t_k)\tilde{F})) \cdot \mu(\sigma) \cdot B \\ &= (\mu(t_1) \otimes (\tilde{\mu}(t_2)\tilde{F}) \otimes \cdots \otimes (\tilde{\mu}(t_k)\tilde{F})) \cdot \mu(\sigma) \cdot B.\end{aligned}$$

From here we argue inductively as follows: Assume that for some $l \in [k-1]$,

$$\tilde{\mu}(t) \cdot \tilde{F} \cdot B = (\mu(t_1) \otimes \cdots \otimes \mu(t_l) \otimes (\tilde{\mu}(t_{l+1})\tilde{F}) \otimes \cdots \otimes (\tilde{\mu}(t_k)\tilde{F})) \cdot \mu(\sigma) \cdot B.$$

Then by bilinearity of Kronecker product, we get that $\tilde{\mu}(t) \cdot \tilde{F} \cdot B$ is equal to

$$(\tilde{\mu}(t_{l+1})\tilde{F}) \cdot (\mu(t_1) \otimes \cdots \otimes \mu(t_l) \otimes I_n \otimes (\tilde{\mu}(t_{l+2})\tilde{F}) \otimes \cdots \otimes (\tilde{\mu}(t_k)\tilde{F})) \cdot \mu(\sigma) \cdot B.$$

Here $(\mu(t_1) \otimes \cdots \otimes \mu(t_l) \otimes I_n \otimes (\tilde{\mu}(t_{l+2})\tilde{F}) \otimes \cdots \otimes (\tilde{\mu}(t_k)\tilde{F})) \cdot \mu(\sigma) \cdot B \in \mathcal{B}$ by the same reasoning as above. The induction hypothesis for $t_{l+1} \in T_{\Sigma}^{\leq h}$ now implies

$$\begin{aligned}&\tilde{\mu}(t) \cdot \tilde{F} \cdot B \\ &= \mu(t_{l+1}) \cdot (\mu(t_1) \otimes \cdots \otimes \mu(t_l) \otimes I_n \otimes (\tilde{\mu}(t_{l+2})\tilde{F}) \otimes \cdots \otimes (\tilde{\mu}(t_k)\tilde{F})) \cdot \mu(\sigma) \cdot B \\ &= (\mu(t_1) \otimes \cdots \otimes \mu(t_l) \otimes \mu(t_{l+1}) \otimes (\tilde{\mu}(t_{l+2})\tilde{F}) \otimes \cdots \otimes (\tilde{\mu}(t_k)\tilde{F})) \cdot \mu(\sigma) \cdot B.\end{aligned}$$

Continuing our inductive argument, for $l = k-1$ we get that

$$\tilde{\mu}(t) \cdot \tilde{F} \cdot B = (\mu(t_1) \otimes \cdots \otimes \mu(t_k)) \cdot \mu(\sigma) \cdot B = \mu(t) \cdot B.$$

This completes the proof of (4.7) by induction.

Now since $\gamma \in \mathcal{B}$, for every $t \in T_\Sigma$ we have

$$\|\tilde{\mathcal{A}}\|(t) = \tilde{\mu}(t) \cdot \tilde{\gamma} = \tilde{\mu}(t) \cdot \tilde{F} \cdot \gamma \overset{\text{Eq. (4.7)}}{=} \mu(t) \cdot \gamma = \|\mathcal{A}\|(t).$$

Hence, MTAs $\tilde{\mathcal{A}}$ and $\mathcal{A}$ are equivalent. Minimality follows from Lemma 33. □

As stated in Theorem 5, all equivalent minimal MTAs are equal up to a change of basis. Thus the MTA $\tilde{\mathcal{A}}$ is "canonical" in the sense that any minimal MTA equivalent to $\mathcal{A}$ can be obtained from $\tilde{\mathcal{A}}$ via a linear transformation: any $m$-dimensional MTA $\tilde{\mathcal{A}}' = (m, \Sigma, \tilde{\mu}', \tilde{\gamma}')$ is equivalent to $\mathcal{A}$ if and only if there exists an invertible matrix $U \in \mathbb{F}^{m \times m}$ such that $\tilde{\gamma}' = U \cdot \tilde{\gamma}$ and $\tilde{\mu}'(\sigma) = U^{\otimes \mathrm{rk}(\sigma)} \cdot \tilde{\mu}(\sigma) \cdot U^{-1}$ for every $\sigma \in \Sigma$.

### 4.4.3 Spanning Sets for the Forward and Backward Spaces

The minimal automaton $\tilde{\mathcal{A}}$ from Section 4.4.2 is defined in terms of matrices $F$ and $B$ whose rows and columns span the forward space $\mathcal{F}$ and the backward space $\mathcal{B}$, respectively. In fact, the central algorithmic challenge for minimization lies in the efficient computation of such matrices. In this section we prove a key result, Proposition 40 below, suggesting a way to compute $F$ and $B$, which we exploit in Sections 4.5.2 and 4.6.

Propositions 31 and 32 and their proofs already suggest an efficient algorithm for iteratively computing bases of $\mathcal{F}$ and $\mathcal{B}$. We make this algorithm more explicit and analyse its unit-cost complexity in Section 4.5.1. The drawback of the resulting algorithm will be the use of "if-conditionals": the algorithm branches according to whether certain sets of vectors are linearly independent. Such conditionals are ill-suited for efficient *parallel* algorithms and also for many-one reductions. Thus it cannot be used for an NC-algorithm in Section 4.5.2 nor for a reduction to ACIT in Section 4.6.

The following proposition exhibits polynomial-size sets of spanning vectors for $\mathcal{F}$

and $\mathcal{B}$, which, as we will see later, can be computed efficiently without branching. The proposition is based on the *product* automaton $\mathcal{A} \times \mathcal{A}$, introduced in Section 2.3.3. It defines a sequence $(f(l))_{l \in \mathbb{N}}$ of row vectors and a sequence $(b(l))_{l \in \mathbb{N}}$ of square matrices. Part (a) states that the vector $f(n)$ and the matrix $b(n)$ determine matrices $F$ and $B$, whose rows and columns span $\mathcal{F}$ and $\mathcal{B}$, respectively. Part (b) gives a recursive characterisation of the sequences $(f(l))_{l \in \mathbb{N}}$ and $(b(l))_{l \in \mathbb{N}}$. This allows for an efficient computation of $f(n)$ and $b(n)$.

**Proposition 40.** *Let $\Sigma$ have rank $r$. Let MTA $\mathcal{A} \times \mathcal{A} = (n^2, \Sigma, \mu', \gamma^{\otimes 2})$ be the product of $\mathcal{A}$ by $\mathcal{A}$. For every $l \in \mathbb{N}$, define*

$$f(l) := \sum_{t \in T_\Sigma^{<l}} \mu'(t) \in \mathbb{F}^{1 \times n^2} \quad \text{and} \quad b(l) := \sum_{c \in C^{<l}_{\Sigma, T_\Sigma^{\leq n}}} \mu'(c) \in \mathbb{F}^{n^2 \times n^2}.$$

(a) *Let $F \in \mathbb{F}^{n \times n}$ be the matrix with $F_{i,j} = f(n) \cdot (e_i \otimes e_j)^\top$ for all $i, j \in [n]$. Let $B \in \mathbb{F}^{n \times n}$ be the matrix with $B_{i,j} = (e_i \otimes e_j) \cdot b(n) \cdot \gamma^{\otimes 2}$ for all $i, j \in [n]$. Then, $\mathrm{Row}(F) = \mathcal{F}$ and $\mathrm{Col}(B) = \mathcal{B}$.*

(b) *We have $f(1) = \sum_{\sigma \in \Sigma_0} \mu'(\sigma)$ and $b(1) = I_{n^2}$. For all $l \in \mathbb{N}$, it holds that*

$$f(l+1) = \sum_{k=0}^{r} f(l)^{\otimes k} \sum_{\sigma \in \Sigma_k} \mu'(\sigma), \text{ and}$$
$$b(l+1) = I_{n^2} + \sum_{k=1}^{r} \sum_{j=1}^{k} \left( f(n)^{\otimes (j-1)} \otimes b(l) \otimes f(n)^{\otimes (k-j)} \right) \sum_{\sigma \in \Sigma_k} \mu'(\sigma).$$

*Proof.* First, we prove that $\mathrm{Row}(F) = \mathcal{F}$ in part (a). Let $\widehat{F} \in \mathbb{F}^{T_\Sigma^{\leq n} \times [n]}$ be a matrix such that $\widehat{F}_{t,:} = \mu(t)$ for every $t \in T_\Sigma^{\leq n}$. From Proposition 31 (b) it follows that $\mathrm{Row}(\widehat{F}) = \mathcal{F}$. By Lemma 29 we now have $\mathrm{Row}(\widehat{F}^\top \widehat{F}) = \mathrm{Row}(\widehat{F}) = \mathcal{F}$. Thus in order to prove that $\mathrm{Row}(F) = \mathcal{F}$, it suffices to show that $\widehat{F}^\top \widehat{F} = F$. Indeed, using

the mixed-product property of Kronecker product, we have for all $i, j \in [n]$:

$$\begin{aligned}(\widehat{F}^\top \widehat{F})_{i,j} = (\widehat{F}^\top)_{i,:} \cdot (\widehat{F})_{:,j} &= \sum_{t \in T_\Sigma^{\leq n}} \mu(t)_i \cdot \mu(t)_j \\ &= \sum_{t \in T_\Sigma^{\leq n}} (\mu(t) \cdot e_i^\top) \otimes (\mu(t) \cdot e_j^\top) \\ &= \left( \sum_{t \in T_\Sigma^{\leq n}} (\mu(t) \otimes \mu(t)) \right) \cdot (e_i \otimes e_j)^\top \\ &\overset{\text{Prop. 7}}{=} \left( \sum_{t \in T_\Sigma^{\leq n}} \mu'(t) \right) \cdot (e_i \otimes e_j)^\top = f(n) \cdot (e_i \otimes e_j)^\top.\end{aligned}$$

Next, we complete the proof of part (a) by proving that $\mathrm{Col}(B) = \mathcal{B}$. To avoid notational clutter, in the following we write

$$C := C^{\leq n}_{\Sigma, T_\Sigma^{\leq n}}.$$

Define a matrix $\widehat{B} \in \mathbb{F}^{[n] \times C}$ such that $\widehat{B}_{:,c} = \mu(c) \cdot \gamma$ for all $c \in C$ . From Proposition 32 (b) it follows that $\mathrm{Col}(\widehat{B}) = \mathcal{B}$. By Lemma 29 we now have $\mathrm{Col}(\widehat{B}\widehat{B}^\top) = \mathrm{Col}(\widehat{B}) = \mathcal{B}$. Therefore in order to prove that $\mathrm{Col}(B) = \mathcal{B}$, it suffices to show that $\widehat{B}\widehat{B}^\top = B$. Indeed, using the mixed-product property of Kronecker product, we have for all $i, j \in [n]$:

$$\begin{aligned}(\widehat{B} \cdot \widehat{B}^\top)_{i,j} &= (\widehat{B})_{i,:} \cdot (\widehat{B}^\top)_{:,j} \\ &= \sum_{c \in C} (\mu(c)_{i,:} \cdot \gamma) \cdot (\mu(c)_{j,:} \cdot \gamma) \\ &= \sum_{c \in C} (e_i \cdot \mu(c) \cdot \gamma) \otimes (e_j \cdot \mu(c) \cdot \gamma) \\ &= \sum_{c \in C} (e_i \otimes e_j) \cdot (\mu(c) \otimes \mu(c)) \cdot (\gamma \otimes \gamma) \\ &= (e_i \otimes e_j) \cdot \left( \sum_{c \in C} (\mu(c) \otimes \mu(c)) \right) \cdot (\gamma \otimes \gamma)\end{aligned}$$

$$
\begin{aligned}
&= (e_i \otimes e_j) \cdot \left( \sum_{c \in C} \mu'(c) \right) \cdot \gamma^{\otimes 2} && \text{by Proposition 7 (ii)} \\
&= (e_i \otimes e_j) \cdot b(n) \cdot \gamma^{\otimes 2} && \text{definition of } b(n) \\
&= B_{i,j}.
\end{aligned}
$$

We turn to the proof of part (b). Here we do not use the fact that we are dealing with a product automaton. We first prove the statement on $f(l)$. The equality $f(1) = \sum_{\sigma \in \Sigma_0} \mu'(\sigma)$ follows directly from the definition. For all $l \in \mathbb{N}$,

$$
T_\Sigma^{<l+1} = \{ \sigma(t_1, \ldots, t_k) : 0 \leq k \leq r,\ \sigma \in \Sigma_k,\ t_1, \ldots, t_k \in T_\Sigma^{<l} \} \,.
$$

Thus, by bilinearity of Kronecker product, it holds that

$$
\begin{aligned}
f(l+1) &= \sum_{t \in T_\Sigma^{<l+1}} \mu'(t) \\
&= \sum_{k=0}^{r} \sum_{\sigma \in \Sigma_k} \sum_{t_1 \in T_\Sigma^{<l}} \cdots \sum_{t_k \in T_\Sigma^{<l}} (\mu'(t_1) \otimes \cdots \otimes \mu'(t_k)) \cdot \mu'(\sigma) \\
&= \sum_{k=0}^{r} \left( \left( \sum_{t_1 \in T_\Sigma^{<l}} \mu'(t_1) \right) \otimes \cdots \otimes \left( \sum_{t_k \in T_\Sigma^{<l}} \mu'(t_k) \right) \right) \cdot \sum_{\sigma \in \Sigma_k} \mu'(\sigma) \\
&= \sum_{k=0}^{r} \left( \sum_{t \in T_\Sigma^{<l}} \mu'(t) \right)^{\otimes k} \sum_{\sigma \in \Sigma_k} \mu'(\sigma) \\
&= \sum_{k=0}^{r} f(l)^{\otimes k} \sum_{\sigma \in \Sigma_k} \mu'(\sigma) \,.
\end{aligned}
$$

Finally, we prove the statement on $b(l)$. The equality $b(1) = I_{n^2}$ follows from the definition. To avoid notational clutter we write $T := T_\Sigma^{\leq n}$ in the following. Recall that $f(n) = \sum_{t \in T} \mu'(t)$. We have for all $l \in \mathbb{N}$:

$$
C_{\Sigma,T}^{\leq l+1} = \{\square\} \cup \big\{ \sigma(t_1, \ldots, t_{j-1}, c_j, t_{j+1}, \ldots, t_k) : k \in [r],\ j \in [k],\ \sigma \in \Sigma_k,\ c_j \in C_{\Sigma,T}^{\leq l},
$$

$$t_1, \ldots, t_{j-1}, t_{j+1}, \ldots, t_k \in T\} \ .$$

Thus, using bilinearity of Kronecker product, we get that

$$\begin{aligned}
b(l+1) &= \sum_{c \in C^{\leq l+1}_{\Sigma,T}} \mu'(c) \\
&= \mu'(\square) + \sum_{k=1}^{r} \sum_{j=1}^{k} \sum_{\sigma \in \Sigma_k} \sum_{t_1,\ldots,t_{j-1} \in T} \sum_{c_j \in C^{\leq l}_{\Sigma,T}} \sum_{t_{j+1},\ldots,t_k \in T} (\mu'(t_1) \otimes \cdots \otimes \mu'(c_j) \\
&\qquad \otimes \cdots \otimes \mu'(t_k)) \cdot \mu'(\sigma) \\
&= I_{n^2} + \sum_{k=1}^{r} \sum_{j=1}^{k} \left( \left( \sum_{t_1 \in T} \mu'(t_1) \right) \otimes \cdots \otimes \left( \sum_{c_j \in C^{\leq l}_{\Sigma,T}} \mu'(c_j) \right) \right. \\
&\qquad \left. \otimes \cdots \otimes \left( \sum_{t_k \in T} \mu'(t_k) \right) \right) \cdot \sum_{\sigma \in \Sigma_k} \mu'(\sigma) \\
&= I_{n^2} + \sum_{k=1}^{r} \sum_{j=1}^{k} \left( f(n)^{\otimes(j-1)} \otimes b(l) \otimes f(n)^{\otimes(k-j)} \right) \sum_{\sigma \in \Sigma_k} \mu'(\sigma) \, .
\end{aligned}$$

This completes the proof. □

Loosely speaking, Proposition 40 says that the sum over a small subset of the forward space of the product automaton encodes a spanning set of the whole forward space of the original automaton, and similarly for the backward space.

## 4.5 Minimization Algorithms

In this section we devise algorithms for minimizing a given multiplicity automaton: Section 4.5.1 considers general MTAs, while Section 4.5.2 considers MWAs. For the sake of a complexity analysis in standard models, we fix the field $\mathbb{F} = \mathbb{Q}$.

At a high level, our minimization algorithms rely on producing small spanning sets for the forward and backward space of the input automaton. This yields a *forward-backward factorization* of the Hankel matrix $H$, corresponding to a parts-

based representation of the function computed by the automaton. Each multiplicity automaton corresponding to $H$ yields another forward-backward factorization of $H$. When the multiplicity automaton is minimal, this is a rank factorization of $H$.

### 4.5.1 Minimization of Multiplicity Tree Automata

We describe an implementation of the algorithm implicit in Section 4.4.2, and analyse the number of operations. We consider a multiplicity tree automaton $\mathcal{A} = (n, \Sigma, \mu, \gamma)$. We denote by $r$ the rank of $\Sigma$. The algorithm has three steps, as follows:

#### Step 1 "Forward"

The first step is to compute a matrix $F$ such that $\mathrm{Row}(F) = \mathcal{F}$. Seidl [1990] outlines a fixed-point algorithm for this, computing the smallest vector space that satisfies certain closure properties, and proves that the algorithm takes polynomial time assuming unit-cost arithmetic. Based on Proposition 31 (a) we now give in Table 4.1 an explicit version of Seidl's algorithm.

**Input:** $\mathbb{Q}$-multiplicity tree automaton $(n, \Sigma, \mu, \gamma)$
**Output:** matrix $F$ whose rows form a basis of the forward space $\mathcal{F}$

```
i := 0, j := 0
while i ≤ j do
    forall σ ∈ Σ do
        forall (l_1, ..., l_rk(σ)) ∈ [i]^rk(σ) \ [i − 1]^rk(σ) do
            v := (F_{l_1,:} ⊗ ··· ⊗ F_{l_rk(σ),:}) · μ(σ)
            if v ∉ ⟨F_{1,:}, ..., F_{j,:}⟩
                j := j + 1
                F_{j,:} := v
    i := i + 1
return matrix F ∈ Q^{j×n}
```

Table 4.1: Algorithm for computing a matrix $F$

Our algorithm satisfies the following properties:

**Lemma 41.** *The algorithm in Table 4.1 returns a matrix $F \in \mathbb{Q}^{\overrightarrow{n} \times n}$ whose rows form a basis of the forward space $\mathcal{F}$. Each row of $F$ equals $\mu(t)$ for some tree $t \in T_\Sigma^{<n}$. The algorithm executes $O\left(\sum_{k=0}^{r} |\Sigma_k| \cdot n^{2k+1}\right)$ operations.*

*Proof.* The fact that the rows of $F$ span $\mathcal{F}$ follows from Proposition 31 (a). Moreover, it is clear from the algorithm that the rows of $F$ are linearly independent.

A straightforward induction shows that for each row index $j \geq 1$, the row $F_{j,:}$ equals $\mu(t)$ for some tree $t \in T_\Sigma^{<j}$. Matrix $F \in \mathbb{Q}^{\overrightarrow{n} \times n}$ has full row rank, and therefore $\overrightarrow{n} \leq n$. Hence, each row of $F$ equals $\mu(t)$ for some tree $t \in T_\Sigma^{<n}$.

It remains to analyse the number of operations. Let us consider an iteration of the innermost "for" loop. The computation of $F_{l_1,:} \otimes \cdots \otimes F_{l_{\mathrm{rk}(\sigma)},:}$ requires $O(n^{\mathrm{rk}(\sigma)})$ operations (by iteratively computing partial products). The vector

$$v = (F_{l_1,:} \otimes \cdots \otimes F_{l_{\mathrm{rk}(\sigma)},:}) \cdot \mu(\sigma)$$

is the product of a $1 \times n^{\mathrm{rk}(\sigma)}$ vector with an $n^{\mathrm{rk}(\sigma)} \times n$ matrix. Thus, computing $v$ takes $O(n^{\mathrm{rk}(\sigma)+1})$ operations. For the purpose of checking membership of $v$ in the vector space $\mathcal{F}' := \langle F_{1,:}, \ldots, F_{j,:} \rangle$ it is useful to maintain a matrix $F'$, which is upper triangular (up to a permutation of its columns) and whose rows form a basis of $\mathcal{F}'$. To check whether $v \in \mathcal{F}'$ we compute a vector $v'$ as the result of performing a Gaussian elimination of $v$ against $F'$, which requires $O(j \cdot n)$ operations. If this membership test fails, we extend the matrix $F'$ at the bottom by row $v'$. This preserves the upper-triangular shape of $F'$. Thus, an iteration of the innermost "for" loop takes $O(n^{\mathrm{rk}(\sigma)+1})$ operations. For every $\sigma \in \Sigma$, this "for" loop is executed $O(n^{\mathrm{rk}(\sigma)})$ times. Therefore, the algorithm executes $O\left(\sum_{k=0}^{r} |\Sigma_k| \cdot n^{2k+1}\right)$ operations. □

**Step 2 "Backward"**

The next step suggested in §4.4.2 is to compute a matrix $B$ such that $\mathrm{Col}(B) = \mathcal{B}$. By Lemma 41, each row of the matrix $F$ computed by the algorithm in Table 4.1 equals $\mu(t)$ for some tree $t \in T_\Sigma^{<n}$. Let $S$ denote the set of those trees. Since $\mathrm{Row}(F) = \mathcal{F}$, set $\{\mu(t) : t \in S\}$ spans $\mathcal{F}$. Thus by Proposition 32 (a), $\mathcal{B}$ is the smallest vector space $V \subseteq \mathbb{Q}^n$ such that $\gamma \in V$ and $M \cdot v \in V$ for all $M \in \mathcal{M} := \{\mu(c) : c \in C^1_{\Sigma,S}\}$ and $v \in V$. Tzeng [1992] shows, for an arbitrary column vector $\gamma \in \mathbb{Q}^n$ and an arbitrary finite set of matrices $\mathcal{M} \subseteq \mathbb{Q}^{n\times n}$, how to compute a basis of $V$ in time $O(|\mathcal{M}| \cdot n^4)$. This can be improved to $O(|\mathcal{M}| \cdot n^3)$ [see, e.g., Cortes et al., 2006]. This leads to the following lemma:

**Lemma 42.** *Given the matrix $F \in \mathbb{Q}^{\overrightarrow{n}\times n}$ which is the output of the algorithm in Table 4.1, a matrix $B$ whose columns form a basis of the backward space $\mathcal{B}$ can be computed with $O\left(\sum_{k=1}^{r} |\Sigma_k| \cdot (kn^{2k} + kn^{k+2})\right)$ operations.*

*Proof.* Consider the computation of an arbitrary $M \in \mathcal{M} := \{\mu(c) : c \in C^1_{\Sigma,S}\}$. We have:

$$M = G \cdot \mu(\sigma)\ , \quad \text{where} \tag{4.8}$$

$$G = F_{l_1,:} \otimes \cdots \otimes F_{l_{i-1},:} \otimes I_n \otimes F_{l_{i+1},:} \otimes \cdots \otimes F_{l_{\mathrm{rk}(\sigma)},:} \in \mathbb{Q}^{n\times n^{\mathrm{rk}(\sigma)}} \tag{4.9}$$

is such that $\sigma \in \Sigma \setminus \Sigma_0$, $i \in [\mathrm{rk}(\sigma)]$, $l_1, \ldots, l_{i-1}, l_{i+1}, \ldots, l_{\mathrm{rk}(\sigma)} \in [\overrightarrow{n}]$.

Exploiting the sparsity pattern in matrix $G$ as in (4.9), the computation of the nonzero entries of $G$ takes $O(n^{\mathrm{rk}(\sigma)})$ operations. Exploiting sparsity again, the computation of matrix $M$ as in (4.8) then takes $O(n^{\mathrm{rk}(\sigma)+1})$ operations. Since $\overrightarrow{n} \leq n$, it follows from (4.8) and (4.9) that

$$|\mathcal{M}| \in O\left(\sum_{k=1}^{r} |\Sigma_k| \cdot k \cdot n^{k-1}\right)\ .$$

Thus, the number of operations required to compute $\mathcal{M}$ is $O\left(\sum_{k=1}^{r} |\Sigma_k| \cdot k \cdot n^{2k}\right)$. Given $\mathcal{M}$, computing a basis of $\mathcal{B}$ takes

$$O(|\mathcal{M}| \cdot n^3) = O\left(\sum_{k=1}^{r} |\Sigma_k| \cdot k \cdot n^{k-1} \cdot n^3\right)$$

operations, using, e.g., the above-mentioned method of Cortes et al. [2006]. Thus, the total operation count for computing a matrix $B$ is $O\left(\sum_{k=1}^{r} |\Sigma_k| \cdot (kn^{2k} + kn^{k+2})\right)$. □

**Step 3 "Solve"**

The final step suggested in Section 4.4.2 has two substeps. The first substep is to compute a matrix $\tilde{F} \in \mathbb{Q}^{m \times n}$ with $m = \operatorname{rank}(F \cdot B)$ and $\operatorname{Row}(\tilde{F} \cdot B) = \operatorname{Row}(F \cdot B)$. Such a matrix $\tilde{F}$ can be computed from $F$ by going through the rows of $F$ one by one and including only those rows that are linearly independent of the previous rows when multiplied by $B$. This can be done in time $O(n^3)$, e.g., by transforming the matrix $F \cdot B$ into a triangular form using Gaussian elimination.

The second substep is to compute the minimal MTA $\tilde{\mathcal{A}} = (m, \Sigma, \tilde{\mu}, \tilde{\gamma})$. The vector $\tilde{\gamma} = \tilde{F} \cdot \gamma$ is easy to compute. Solving Equation (4.3) for each $\tilde{\mu}(\sigma)$ can be done via Gaussian elimination in time $O(n^3)$; however, the bottleneck is the computation of $\tilde{F}^{\otimes k} \cdot \mu(\sigma)$ for every $\sigma \in \Sigma_k$, which takes

$$O\left(\sum_{k=0}^{r} |\Sigma_k| \cdot n^k \cdot n^k \cdot n\right) = O\left(\sum_{k=0}^{r} |\Sigma_k| \cdot n^{2k+1}\right)$$

operations. Putting together the results of this subsection, we get:

**Theorem 43.** *There is an algorithm that transforms a given $\mathbb{Q}$-MTA $\mathcal{A} = (n, \Sigma, \mu, \gamma)$ into an equivalent minimal $\mathbb{Q}$-MTA. Assuming unit-cost arithmetic, the algorithm takes time*

$$O\left(\sum_{k=0}^{r} |\Sigma_k| \cdot (n^{2k+1} + kn^{2k} + kn^{k+2})\right),$$

*which is* $O(|\mathcal{A}|^2 \cdot r)$. □

### 4.5.2 Minimization of Multiplicity Word Automata in NC

In this subsection, we consider the problem of minimizing a given $\mathbb{Q}$-multiplicity word automaton. We prove the following result:

**Theorem 44.** *There is an* NC *algorithm that transforms a given $\mathbb{Q}$-MWA into an equivalent minimal $\mathbb{Q}$-MWA. In particular, given a $\mathbb{Q}$-MWA and a number $d \in \mathbb{N}_0$, one can decide in* NC *whether there exists an equivalent $\mathbb{Q}$-MWA of dimension at most $d$.*

Theorem 44 improves on two results of Kiefer et al. [2013]. First, [Kiefer et al., 2013, Theorem 4.2] states that deciding whether a $\mathbb{Q}$-MWA is minimal is in NC. Second, [Kiefer et al., 2013, Theorem 4.5] states the same thing as our Theorem 44, but with NC replaced with *randomized* NC.

*Proof of Theorem 44.* The algorithm relies on Propositions 35 and 40. Let the given $\mathbb{Q}$-MWA be $\mathcal{A} = (n, \Sigma, \mu, \alpha, \gamma)$. In the notation of Proposition 40, we have for all $l \in \mathbb{N}$ that

$$b(l+1) = I_{n^2} + b(l) \cdot \sum_{\sigma \in \Sigma} \mu'(\sigma).$$

From here one can easily show, using an induction on $l$, that for all $l \in \mathbb{N}$:

$$b(l) = \sum_{k=0}^{l-1} \left( \sum_{\sigma \in \Sigma} \mu'(\sigma) \right)^k.$$

It follows for the matrix $B \in \mathbb{Q}^{n \times n}$ from Proposition 40 that for all $i, j \in [n]$:

$$B_{i,j} = (e_i \otimes e_j) \cdot b(n) \cdot \gamma^{\otimes 2} = (e_i \otimes e_j) \cdot \left( \sum_{k=0}^{n-1} \Big( \sum_{\sigma \in \Sigma} \mu'(\sigma) \Big)^k \right) \cdot \gamma^{\otimes 2}.$$

Note that, since $\mathcal{A}$ is an MWA, we have $f(l) = b(l)$ for all $l \in \mathbb{N}$. We now have for

the matrix $F \in \mathbb{Q}^{n \times n}$ from Proposition 40 and all $i, j \in [n]$:

$$F_{i,j} = \alpha^{\otimes 2} \cdot \left( \sum_{k=0}^{n-1} \Big( \sum_{\sigma \in \Sigma} \mu'(\sigma) \Big)^k \right) \cdot (e_i \otimes e_j)^\top.$$

Matrices $F$ and $B$ can be computed in NC since sums and matrix powers can be computed in NC [Cook, 1985]. Next we show how to compute in NC the matrix $\tilde{F}$, which is needed to compute the minimal $\mathbb{Q}$-MWA $\tilde{\mathcal{A}}$ from Section 4.4.2. Our NC algorithm includes the $i^{\text{th}}$ row of $F$ (i.e., $F_{i,:}$) in $\tilde{F}$ if and only if

$$\operatorname{rank}(F_{[i],[n]} \cdot B) > \operatorname{rank}(F_{[i-1],[n]} \cdot B).$$

This can be done in NC since the rank of a matrix can be determined in NC [Ibarra et al., 1980]. It remains to compute $\tilde{\gamma} := \tilde{F}\gamma$ and solve Equations (4.4) and (4.5) for $\tilde{\alpha}$ and $\tilde{\mu}(\sigma)$, respectively. Both are easily done in NC. □

## 4.6 Decision Problem

In this section we characterise the complexity of the following decision problem: Given a $\mathbb{Q}$-MTA and a number $d \in \mathbb{N}_0$, the *minimization* problem asks whether there is an equivalent $\mathbb{Q}$-MTA of dimension at most $d$. We show:

**Theorem 45.** *Minimization is logspace interreducible with* ACIT.

We consider the lower and the upper bound separately.

### Lower Bound

Given a $\mathbb{Q}$-MTA $\mathcal{A}$, the *zeroness* problem asks whether $\|\mathcal{A}\|(t) = 0$ for all trees $t$. Observe that $\|\mathcal{A}\|(t) = 0$ for all trees $t$ if and only if there exists an equivalent automaton of dimension 0. Therefore, zeroness is a special case of minimization.

We observe that there is a logspace reduction from ACIT to zeroness, which implies ACIT-hardness of minimization. Indeed, Theorem 13 states that the *equivalence* problem for $\mathbb{Q}$-MTAs is logspace equivalent to ACIT. By Proposition 7, one can reduce this problem to zeroness in logarithmic space.

## Upper Bound

We prove:

**Proposition 46.** *There is a logspace reduction from minimization to* ACIT.

*Proof.* Let $\mathcal{A} = (n, \Sigma, \mu, \gamma)$ be the given $\mathbb{Q}$-MTA, and let $d \in \mathbb{N}_0$ be the given number. In our reduction to ACIT, we allow input gates with rational labels as well as division gates. Rational numbers and division gates can be eliminated in a standard way by constructing separate gates for the numerators and denominators of the rational numbers computed by the original gates.

We know from Lemma 33 that the dimension of a minimal MTA equivalent to $\mathcal{A}$ is $m := \mathrm{rank}(F \cdot B)$ where $F$ and $B$ are matrices such that $\mathrm{Row}(F) = \mathcal{F}$ and $\mathrm{Col}(B) = \mathcal{B}$. Therefore, we have $m \leq d$ if and only if $\mathrm{rank}(F \cdot B) \leq d$. The recursive characterisation of $F$ and $B$ from Proposition 40 allows us to compute in logarithmic space an arithmetic circuit for $F \cdot B$. Thus, the result follows from Lemma 47 below. □

The following lemma follows easily from the well-known NC procedure for computing matrix rank by Csanky [1976].

**Lemma 47.** *Let $M \in \mathbb{Q}^{m \times n}$ and $d \in \mathbb{N}_0$. The problem of deciding whether* $\mathrm{rank}(M) \leq d$ *is logspace reducible to* ACIT.

*Proof.* It follows from the rank-nullity theorem of linear algebra that $\mathrm{rank}(M) \leq d$ if and only if $\dim(\ker(M)) \geq n - d$. Since $\ker(M) = \ker(M^\top M)$, this is equivalent to

$\dim(\ker(M^\top M)) \geq n - d$. Matrix $M^\top M$ is Hermitian, therefore $\dim(\ker(M^\top M)) \geq n - d$ if and only if the $n - d$ lowest-order coefficients of the characteristic polynomial of $M^\top M$ are all zero [Ibarra et al., 1980]. But these coefficients are representable by arithmetic circuits with inputs from $M$ [see Csanky, 1976]. □

We emphasise that our reduction to ACIT is a many-one reduction, thanks to Proposition 40: our reduction computes only a single instance of ACIT; there are no if-conditionals.

## 4.7 Minimal Consistent Multiplicity Automaton

In this section we consider a passive learning framework for multiplicity automata, where one is given a finite set of observations and the aim is to find an automaton that fits those observations.

Formally, let $\mathbb{F}$ be an arbitrary field. A natural computational problem is to compute an $\mathbb{F}$-MWA $\mathcal{A}$ of minimal dimension that is consistent with a given finite set of $\mathbb{F}$-weighted words $S = \{(w_1, r_1), \ldots, (w_m, r_m)\}$, where $w_i \in \Sigma^*$ and $r_i \in \mathbb{F}$ for every $i \in [m]$. Here *consistency* means that $\|\mathcal{A}\|(w_i) = r_i$ for every $i \in [m]$.

The main result of this section concerns the computability of the above consistency problem for the field of rational numbers. More specifically, we consider a decision version of this problem, which we call the *minimal consistency problem*, which asks whether there exists a $\mathbb{Q}$-MWA consistent with a set of input-output behaviours $S \subseteq \Sigma^* \times \mathbb{Q}$ and that has dimension at most some nonnegative integer bound $n$.

We show that the minimal consistency problem is logspace equivalent to the problem of deciding the truth of existential first-order sentences over the structure $(\mathbb{Q}, +, \cdot, 0, 1)$. This problem is known [see Koenigsmann, 2014] to be equivalent to Hilbert's Tenth Problem over $\mathbb{Q}$, and its decidability is a longstanding open problem [Poonen, 2003]. This should be compared with the result that the problem of

finding the smallest deterministic finite automaton consistent with a set of accepted or rejected words is NP-complete [Gold, 1978].

The reduction of the minimal consistency problem to the decision problem for existential first-order sentences over the structure $(\mathbb{Q}, +, \cdot, 0, 1)$ is immediate. The idea is to represent a $\mathbb{Q}$-MWA $\mathcal{A} = (n, \Sigma, \mu, \alpha, \gamma)$ "symbolically" by introducing separate variables for each entry of the initial weight vector $\alpha$, final weight vector $\gamma$, and each transition matrix $\mu(\sigma)$, $\sigma \in \Sigma$. Then, consistency of automaton $\mathcal{A}$ with a given finite sample $S \subseteq \Sigma^* \times \mathbb{Q}$ can directly be written as an existential sentence.

We note in passing that the minimal consistency problem for multiplicity word and tree automata over the field $\mathbb{R}$ is in like manner reducible to the problem of deciding the truth of existential first-order sentences over the structure $(\mathbb{R}, +, \cdot, 0, 1)$, which is well known to be decidable in PSPACE [Canny, 1988].[1]

Conversely, we reduce the decision problem for existential first-order sentences over the structure $(\mathbb{Q}, +, \cdot, 0, 1)$ to the minimal consistency problem for $\mathbb{Q}$-MWAs. In fact, it suffices to consider sentences in the restricted form

$$\exists x_1 \cdots \exists x_n \bigwedge_{i=1}^{m} f_i(x_1, \ldots, x_n) = 0\,, \tag{4.10}$$

where $f_i(x_1, \ldots, x_n) = \sum_{j=1}^{l_i} c_{i,j} x_1^{k_{i,j,1}} \cdots x_n^{k_{i,j,n}}$ is a polynomial with rational coefficients. We can make this simplification without loss of generality since a disjunction of atomic formulas $f = 0 \vee g = 0$, where $f$ and $g$ are polynomials, can be rewritten to

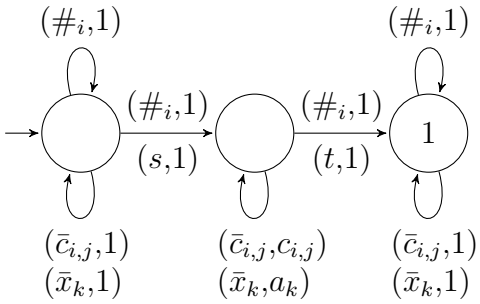
$$\exists x\, (x^2 - x = 0 \wedge x \cdot f = 0 \wedge (1 - x) \cdot g = 0)\,.$$

Moreover, the negation of an atomic formula $f \neq 0$ is equivalent to $\exists x\, (x \cdot f = 1)$.

[1] To consider this problem within the conventional Turing model, we assume that the set $S$ of input-output behaviours is still a subset of $\Sigma^* \times \mathbb{Q}$. Of course, the dimension of the smallest MWA consistent with a given finite set of behaviours $S$ depends on the weight field of the output automaton.

| | $st$ | $t$ | $\varepsilon$ |
|---|---|---|---|
| $\varepsilon$ | 1 | 0 | 0 |
| $s$ | 0 | 1 | 0 |
| $st$ | 0 | 0 | 1 |
| $\#_i$ | 1 | 1 | 0 |
| $s\#_i$ | 0 | 0 | 1 |
| $st\#_i$ | 0 | 0 | 1 |
| $\bar{c}_{i,j}$ | 1 | 0 | 0 |
| $s\bar{c}_{i,j}$ | 0 | $c_{i,j}$ | 0 |
| $st\bar{c}_{i,j}$ | 0 | 0 | 1 |
| $\bar{x}_k$ | 1 | 0 | 0 |
| $s\bar{x}_k$ | 0 | $a_k$ | 0 |
| $st\bar{x}_k$ | 0 | 0 | 1 |
| $t$ | 0 | 0 | 0 |
| $stt$ | 0 | 0 | 0 |
| $ss$ | 0 | 0 | 0 |
| $sts$ | 0 | 0 | 0 |

(a)




(b)

Figure 4.1: The left figure (a) shows a Hankel-matrix fragment $\tilde{H}$, where $i \in [m]$, $j \in [l_i]$, $k \in [n]$. The right figure (b) shows a graph representation of the automaton $\mathcal{A}$.

Define an alphabet

$$\Sigma := \{s, t\} \cup \{\#_i, \bar{c}_{i,j}, \bar{x}_k : i \in [m], j \in [l_i], k \in [n]\},$$

including symbols $\bar{c}_{i,j}$ and $\bar{x}_k$ for each coefficient $c_{i,j}$ and variable $x_k$, respectively. Over alphabet $\Sigma$ we consider the 3-dimensional $\mathbb{Q}$-MWA $\mathcal{A}$, depicted in Figure 4.1 (b). The transitions in this automaton are annotated by label-weight pairs in $\Sigma \times \mathbb{Q}$. Recall that the weights $c_{i,j}$ are coefficients of the polynomial $f_i$. For each $k \in [n]$, the weight $a_k$ is a fixed but arbitrary element of $\mathbb{Q}$.

Define $X, Y \subseteq \Sigma^*$ by $X = \{\varepsilon, s, st\}$ and $Y = \{st, t, \varepsilon\}$. Consider the fragment $\tilde{H} := H_{X \cup X\Sigma, Y}$, shown in Figure 4.1 (a), of the Hankel matrix $H$ of $\|\mathcal{A}\|$. We know from Theorem 1 that $\text{rank}(H) \leq 3$. Since $\text{rank}(H_{X,Y}) = 3$, we have $\text{rank}(H_{X,Y}) = \text{rank}(H) = 3$. Now, from Remark 2 it follows that any 3-dimensional $\mathbb{Q}$-MWA $\mathcal{A}'$

that is consistent with $H_{X,Y}$ and $H_{X\Sigma,Y}$ (i.e., consistent with $\tilde{H}$) is equivalent to $\mathcal{A}$.

Now for every $i \in [m]$, we encode polynomial $f_i$ by the word

$$w_i := \#_i \bar{c}_{i,1} \bar{x}_1^{k_{i,1,1}} \cdots \bar{x}_n^{k_{i,1,n}} \#_i \cdots \#_i \bar{c}_{i,l_i} \bar{x}_1^{k_{i,l_i,1}} \cdots \bar{x}_n^{k_{i,l_i,n}} \#_i$$

over alphabet $\Sigma$. Note that $w_i$ comprises $l_i$ 'blocks' of symbols, corresponding to the $l_i$ monomials in $f_i$, with each block enclosed by two $\#_i$ symbols. From the definition of $w_i$ it follows that $\|\mathcal{A}\|(w_i) = f_i(a_1, \ldots, a_n)$; the details are given below in the proof of Proposition 48.

We define a set of weighted words $S \subseteq \Sigma^* \times \mathbb{Q}$ as $S := S_1 \cup S_2$, where $S_1$ is the set of all pairs $(uv, \tilde{H}_{u,v})$ with $u \in X \cup X\Sigma$, $v \in Y$, and $uv \notin \{s\bar{x}_k t : k \in [n]\}$, and $S_2 := \{(w_i, 0) : i \in [m]\}$. That is, $S_1$ specifies all entries in the matrix $\tilde{H}$ except those that are in row $s\bar{x}_k$ and column $t$.

Any 3-dimensional $\mathbb{Q}$-MWA $\mathcal{A}'$ consistent with $S_1$ is equivalent to an automaton of the form $\mathcal{A}$ for some $a_1, \ldots, a_n \in \mathbb{Q}$. If $\mathcal{A}'$ is moreover consistent with $S_2$, then $f_i(a_1, \ldots, a_n) = 0$ for every $i \in [m]$. From this observation we have the following proposition:

**Proposition 48.** *The sample $S$ is consistent with a 3-dimensional $\mathbb{Q}$-MWA if and only if the sentence* (4.10) *is true in* $(\mathbb{Q}, +, \cdot, 0, 1)$.

*Proof.* We have already noted that any 3-dimensional $\mathbb{Q}$-MWA consistent with $S$ must be equivalent to an automaton of the form $\mathcal{A}$ in Figure 4.1 (b) for some $a_1, \ldots, a_n \in \mathbb{Q}$. However, such an automaton is consistent with $S$ if and only if it assigns weight 0 to each word $w_i$, $i \in [m]$. Now, we claim that this is the case if and only if $(a_1, \ldots, a_n)$ is a root of $f_i$ for every $i \in [m]$, where $a_k$ is the weight of the $\bar{x}_k$-labelled self-loop in the middle state, for every $k \in [n]$.

For every $i \in [m]$, the word $w_i$ has $l_i$ different accepting runs in $\mathcal{A}$, one for each monomial in $f_i$. The $j^{\text{th}}$ such run, in which the block $\bar{c}_{i,j} \bar{x}_1^{k_{i,j,1}} \cdots \bar{x}_n^{k_{i,j,n}}$ is

read in the middle state, has weight $c_{i,j}a_1^{k_{i,j,1}} \cdots a_n^{k_{i,j,n}}$, i.e., the value of monomial $c_{i,j}x_1^{k_{i,j,1}} \cdots x_n^{k_{i,j,n}}$ evaluated at $(a_1, \ldots, a_n)$. Thus $\|\mathcal{A}\|(w_i) = f_i(a_1, \ldots, a_n)$. □

From Proposition 48 we derive the main result of this section:

**Theorem 49.** *The minimal consistency problem for $\mathbb{Q}$-MWAs is logspace equivalent to the decision problem for existential first-order sentences over $(\mathbb{Q}, +, \cdot, 0, 1)$.* □

Theorem 49 also holds for $\mathbb{Q}$-MTAs of a fixed alphabet rank, because the minimal consistency problem can be reduced to the decision problem for existential first-order sentences over $(\mathbb{Q}, +, \cdot, 0, 1)$ in similar manner to the case of words. Here, fixing the alphabet rank keeps the reduction in polynomial time.

## 4.8 Conclusion

In this chapter, we have looked at the computational complexity of computing minimal multiplicity word and tree automata from several angles. Specifically, we have analysed the complexity of *computing* a minimal automaton equivalent to a given input automaton $\mathcal{A}$. We have considered also the corresponding *decision problem*, which asks whether there exists an automaton equivalent to $\mathcal{A}$ with a given number of states. Finally, we have considered the *minimal consistency problem*, in which the input is a finite set of word-weight pairs rather than a complete automaton.

One of the key technical contributions of this chapter is Proposition 40, which, based on the product of a given automaton by itself, provides small spanning sets for the forward space $\mathcal{F}$ and the backward space $\mathcal{B}$. This technology led us to an NC algorithm for minimizing multiplicity *word* automata, thus improving the best previous algorithms (polynomial time and randomized NC).

Our complexity bounds have drawn connections between automaton minimization and longstanding open questions in arithmetic complexity, including the complexity

of polynomial identity testing and the decidability of Hilbert's Tenth Problem over the rationals, which is equivalent to the problem of deciding the truth of existential sentences over the structure $(\mathbb{Q}, +, \cdot, 0, 1)$.

Our algorithmic results exclusively concern automata over the fields of rational or real numbers, in which weights are allowed to be negative. The minimization problems considered here all have natural analogues for the class of *probabilistic automata* over words and trees, in which the transition weights are probabilities. Minimization of probabilistic word automata was shown to be NP-hard by Kiefer and Wachter [2014]. A natural question is whether this minimization problem lies in NP, and whether the corresponding problem for tree automata is even harder. Related to this is the following question: Given a probabilistic automaton with rational transition weights, need there always be a minimal equivalent probabilistic automaton also with rational transition weights? We answer this question negatively in Chapter 8.

We have observed that the minimal consistency problem for multiplicity word automata over the reals is in PSPACE, since it is directly reducible to the problem of deciding the truth of existential first-order sentences over the structure $(\mathbb{R}, +, \cdot, 0, 1)$. For tree automata this reduction is exponential in the alphabet rank, and we leave as an open question the complexity of the minimal consistency problem for tree automata over the reals.

In all cases, we have considered minimizing automata with respect to the number of states. Another natural question is minimization with respect to the number of transitions. This is particularly pertinent to the case of tree automata, where the number of transitions is potentially exponential in the number of states.

# Part II

# Matrix Factorizaton

# Chapter 5

# Overview of Nonnegative Matrix Factorization

**Abstract**

*We present an overview of key concepts and background literature for the second part of the thesis, which focuses on nonnegative matrix factorization. This chapter is based on the material in [Chistikov et al., 2016] and [Chistikov et al.].*

---

## 5.1 Introduction

Nonnegative matrix factorization (NMF) is the task of factoring a matrix of nonnegative real numbers $M$ (henceforth a *nonnegative* matrix) as a product $M = W \cdot H$ such that the matrices $W$ and $H$ are also nonnegative. Note that the existence of an NMF $M = W \cdot H$ means that the columns of $M$ are in the cone generated by the columns of $W$. As well as being a natural problem in its own right [Thomas, 1974, Cohen and Rothblum, 1993], NMF has found many applications across various

domains, including machine learning, combinatorics, and communication complexity; see, e.g., [Venkatasubramanian, 2013, Gillis, 2014, Moitra, 2016, Yannakakis, 1991] and the references therein.

In applications, matrix $M$ can typically be seen as a matrix of data points: each column of $M$ corresponds to a data point and each row to a feature. Then, computing a nonnegative factorization $M = W \cdot H$ corresponds to expressing the data points (columns of $M$) as convex combinations of latent factors (columns of $W$), i.e., as linear combinations of latent factors with nonnegative coefficients (columns of $H$). In topic modelling [Arora et al., 2012b], NMF corresponds to reconstructing documents in a given corpus as distributions on a small number of topics, where each topic is a distribution on words. It has long been known [Cohen and Rothblum, 1993] that NMF can be interpreted geometrically as finding a set of vectors (columns of $W$) inside a unit simplex whose convex hull contains a given set of points (columns of $M$).

For an NMF $M = W \cdot H$, the number of columns in $W$ is called the *inner dimension*. The smallest inner dimension of any NMF of $M$ is called the *nonnegative rank* of $M$, written $\text{rank}_+(M)$; a notion introduced by Gregory and Pullman [1983]. We recall that the usual rank of $M$ is defined as the smallest integer $k$ such that $M$ can be factored as a product $M = L \cdot R$ of two (not necessarily nonnegative) matrices $L, R$ where $L$ has $k$ columns. Therefore, the nonnegative rank of $M$ is at least the rank of $M$.

From a computational perspective, perhaps the most basic problem concerning NMF is whether a given nonnegative matrix of rational numbers $M$ admits an NMF with inner dimension at most a given number $k$, i.e, whether $\text{rank}_+(M) \leq k$. This is known as the *NMF problem*.

Vavasis [2009] showed that the problem of deciding whether the rank of a nonnegative matrix is equal to its nonnegative rank is NP-hard. This result implies that generalisations of this problem, such as the NMF problem, the problem of computing

the factors $W, H$ (in both exact and approximate versions), and nonnegative rank determination, are also NP-hard. It is not known whether any of these problems are in NP.

In practical applications, various heuristics and local-search algorithms are used to compute an approximate nonnegative factorization, but little is known in terms of their theoretical guarantees. The NMF problem is tractable under the *separability* assumption of Donoho and Stodden [2003]: an NMF $M = W \cdot H$ is called *separable* if every column of $W$ is also a column of $M$. Arora et al. [2012a] showed that it is decidable in polynomial time whether a given matrix admits a separable NMF with a given inner dimension, and gave a polynomial-time algorithm for computing such a separable NMF. Further progress was made more recently, with several efficient algorithms for computing nonnegative factorization of near-separable matrices [Kumar et al., 2013, Gillis and Vavasis, 2015]. Still, the exact complexity of the NMF problem is open.

Currently the best complexity bound for the NMF problem is membership in PSPACE, which is obtained by translation into the existential theory of real-closed fields [Arora et al., 2012a]. Such a translation shows that one can always choose the entries of $W$ and $H$ to be algebraic numbers. Beyond this generic upper bound, the problem has been attacked from many different angles. Here we highlight the results of Moitra [2016], who found semi-algebraic descriptions of the sets of matrices of nonnegative rank at most $r$ in which the number of variables is $O(r^2)$, and Arora et al. [2012a], who identified several variants of the problem that are efficiently solvable.

Vavasis [2009, Section 5] notes that the difficulty in proving membership in NP lies in the fact that a certificate for a positive answer to the NMF problem seems to require the sought factors: a pair of nonnegative matrices $W$, $H$ such that $M = W \cdot H$. Vavasis further notes that this question is related to an older open problem about the nonnegative rank, due to Cohen and Rothblum [1993]:

> "PROBLEM. Show that the nonnegative ranks of a rational matrix over the reals and over the rationals coincide, or provide an example where the two ranks are different."

We refer to this problem as the *Cohen–Rothblum problem*. Here the *nonnegative rank* of a rational matrix $M$ *over the rationals* is the smallest inner dimension of an NMF $M = W \cdot H$ with matrices $W, H$ that have only *rational* entries, whereas the *nonnegative rank over the reals* is the usual nonnegative rank.

We first consider a variant of the Cohen–Rothblum problem for restricted NMF, a strict generalisation of separable NMF that was introduced by Gillis and Glineur [2012]. Formally, an NMF $M = W \cdot H$ is *restricted* if the columns of $W$ span the same vector space as the columns of $M$. The smallest inner dimension of any restricted NMF of $M$ is called the *restricted nonnegative rank* of $M$, written $\operatorname{rrank}_+(M)$. Clearly, the restricted nonnegative rank of $M$ is at least the nonnegative rank of $M$. When $\operatorname{rank}(M) \leq 2$, the rank, nonnegative rank, and restricted nonnegative rank of $M$ coincide.

In Chapter 6, we show that restricted NMF has a natural geometric interpretation as the nested polytope problem (NPP), defined in Section 5.5.1. Using this geometric interpretation, we show that any rational nonnegative matrix $M$ of rank 3 or less always has a *rational* restricted NMF with minimal inner dimension, that can moreover be computed in polynomial time in the Turing model of computation. This improves a result of Gillis and Glineur [2012] where this problem is shown to be in polynomial time assuming a RAM model with unit-cost arithmetic. Furthermore in Chapter 6, we exhibit a rank-4 matrix that has a restricted NMF with inner dimension 5 but *no rational* restricted NMF with inner dimension 5. We thus answer the restricted NMF variant of the Cohen–Rothblum problem negatively.

Then in Chapter 7, we solve the Cohen–Rothblum problem in its full generality by providing an example of a rational matrix $M$ that has different nonnegative ranks

over $\mathbb{R}$ and over $\mathbb{Q}$. In other words, any NMF $M = W \cdot H$ with minimal inner dimension has irrational entries in $W$ and $H$. Our counterexample is almost optimal inasmuch as $M$ has rank 4, whereas for matrices of rank at most 2 nonnegative ranks coincide over $\mathbb{R}$ and $\mathbb{Q}$ [Cohen and Rothblum, 1993].

While our negative resolution of the Cohen–Rothblum problem does not exclude NP membership of the NMF problem, it does rule out a hypothetical "simple" argument for membership in NP, wherein a certificate is an NMF with rational entries of small bit size.

## 5.2 Related Work

Uniqueness properties of NMF have been studied by many authors in the past few decades [see, e.g., Thomas, 1974, Laurberg et al., 2008, Gillis, 2012].

In the last few years, there has been progress towards resolving the Cohen–Rothblum problem. It was already known to Cohen and Rothblum [1993] that nonnegative ranks over $\mathbb{R}$ and $\mathbb{Q}$ coincide for matrices of rank at most 2. (Note that the usual ranks over $\mathbb{R}$ and $\mathbb{Q}$ coincide for all rational matrices.) Kubjas et al. [2015, Corollary 4.6] extended this result to matrices of nonnegative rank (over $\mathbb{R}$) at most 3. On the other hand, Shitov [2015] proved that the nonnegative rank of a matrix can indeed depend on the underlying field: he exhibited a nonnegative matrix with irrational entries whose nonnegative rank over a subfield of $\mathbb{R}$ is different from its nonnegative rank over $\mathbb{R}$.

An alternative solution to the Cohen–Rothblum problem was obtained, concurrently[1] and independently of our result, by Shitov [2016a] using a different construction. Shitov [2016a] provides an example of a $21 \times 21$ matrix that has rank 17, nonnegative rank 19, and nonnegative rank over $\mathbb{Q}$ at least 20. In contrast, our coun-

[1]Our paper [Chistikov et al.] was submitted to arXiv on 22 May 2016. The paper [Shitov, 2016a] was submitted to arXiv on 23 May 2016.

terexample matrix has rank 4, nonnegative rank 5, and nonnegative rank over $\mathbb{Q}$ equal to 6.

## 5.3 Nonnegative Matrix Factorization

We use the basic notation and terminology introduced in Section 2.1.

Let $\mathbb{F}$ be an ordered field, such as the reals $\mathbb{R}$ or the rationals $\mathbb{Q}$. Given a nonnegative matrix $M \in \mathbb{F}_+^{n\times m}$, a *nonnegative matrix factorization (NMF) over* $\mathbb{F}$ of $M$ is any representation of the form $M = W \cdot H$ where $W \in \mathbb{F}_+^{n\times d}$ and $H \in \mathbb{F}_+^{d\times m}$ are nonnegative matrices. Clearly, such a factorization entails that $\mathrm{Col}(M) \subseteq \mathrm{Col}(W)$. We refer to $d$ as the *inner dimension* of the NMF, and hence refer to NMF $M = W \cdot H$ as being *$d$-dimensional*. The *nonnegative rank over* $\mathbb{F}$ of $M$ is the smallest number $d \in \mathbb{N}_0$ such that there exists a $d$-dimensional NMF over $\mathbb{F}$ of $M$. The nonnegative rank over $\mathbb{R}$ will henceforth simply be called nonnegative rank, and will be denoted by $\mathrm{rank}_+(M)$. Note that for a matrix $M \in \mathbb{Q}_+^{n\times m}$, its nonnegative rank over $\mathbb{Q}$ is clearly at least $\mathrm{rank}_+(M)$.

An equivalent characterisation [Cohen and Rothblum, 1993] of the nonnegative rank of $M$ over $\mathbb{F}$ is as the smallest number $d$ such that $M$ is equal to the sum of $d$ rank-1 matrices in $\mathbb{F}_+^{n\times m}$. Here recall that a nonzero $n \times m$ matrix has rank 1 just in case it can be written as product $uv^\top$ of an $n$-dimensional column vector $u$ and an $m$-dimension row vector $v^\top$. Writing an NMF of inner dimension $d$ in the form

$$M = W \cdot H = \sum_{k=1}^{d} W_{:,k} \cdot H_{k,:},$$

it follows that such a factorization is equivalent to a decomposition of $M$ as the sum of $d$ nonnegative rank-1 matrices with entries in $\mathbb{F}$. Cohen and Rothblum [1993, Theorem 4.1] showed that if $\mathrm{rank}(M) \leq 2$ then $\mathrm{rank}(M) = \mathrm{rank}_+(M)$.

The nonnegative rank satisfies many of the properties of the general rank with

respect to the usual matrix operations, including being upper bounded by the number of rows and columns and being closed under transposition.

**Lemma 50** (Cohen and Rothblum, 1993)**.** *For any nonnegative matrix $M \in \mathbb{R}_+^{n\times m}$, it holds that:*

(i) $\operatorname{rank}(M) \leq \operatorname{rank}_+(M) \leq \min\{n, m\}$*;*

(ii) $\operatorname{rank}_+(M) = \operatorname{rank}_+(M^\top)$*.*

Given a nonzero matrix $M \in \mathbb{F}_+^{n\times m}$, by removing the zero columns of $M$ and dividing each remaining column by the sum of its elements, we obtain a stochastic matrix with equal nonnegative rank. Similarly, if $M = W \cdot H$ then after removing the zero columns in $W$ and multiplying with a suitable diagonal matrix $D$, we get $M = W \cdot H = WD \cdot D^{-1}H$ where $WD$ is stochastic. If $M$ is stochastic then we have

$$\mathbf{1}^\top = \mathbf{1}^\top M = \mathbf{1}^\top WD \cdot D^{-1}H = \mathbf{1}^\top D^{-1}H,$$

hence $D^{-1}H$ is stochastic as well. Thus when computing the nonnegative rank of a stochastic matrix $M$, one can, without loss of generality, only consider NMFs $M = W \cdot H$ in which $W$ and $H$ are stochastic matrices [see also Cohen and Rothblum, 1993, Theorem 3.2]. In such a case, we will call the factorization $M = W \cdot H$ *stochastic*. An NMF $M = W \cdot H$ is likewise called *rational* if both matrices $W$ and $H$ are rational.

The *NMF problem* is: Given a matrix $M \in \mathbb{Q}_+^{n\times m}$ and $k \in \mathbb{N}$, is $\operatorname{rank}_+(M) \leq k$? As mentioned in Section 5.1, the NMF problem is NP-hard, even for $k = \operatorname{rank}(M)$ [see Vavasis, 2009]. On the other hand, it is polynomial-time reducible to the existential theory of the reals, hence by Canny [1988] and Renegar [1992] it is in PSPACE.

## 5.4 Restricted Nonnegative Matrix Factorization

For all matrices $M \in \mathbb{F}_+^{n\times m}$, an NMF $M = W \cdot H$ is called *restricted NMF (RNMF)* if $\operatorname{rank}(M) = \operatorname{rank}(W)$ [Gillis and Glineur, 2012]. As we know $\operatorname{Col}(M) \subseteq \operatorname{Col}(W)$ holds for all NMF instances, the condition $\operatorname{rank}(M) = \operatorname{rank}(W)$ is then equivalent to $\operatorname{Col}(M) = \operatorname{Col}(W)$. The *restricted nonnegative rank over* $\mathbb{F}$ of $M$ is the smallest number $d \in \mathbb{N}_0$ such that there exists a $d$-dimensional restricted nonnegative factorization of $M$ over $\mathbb{F}$. Unless indicated otherwise, henceforth we will assume $\mathbb{F} = \mathbb{R}$ when speaking of the restricted nonnegative rank of $M$, and denote it by $\operatorname{rrank}_+(M)$.

We have the following basic properties of restricted nonnegative rank:

**Lemma 51** (Gillis and Glineur, 2012)**.** *Let $M \in \mathbb{R}_+^{n\times m}$. Then*

$$\operatorname{rank}(M) \leq \operatorname{rank}_+(M) \leq \operatorname{rrank}_+(M) \leq m.$$

*Moreover, if* $\operatorname{rank}(M) = \operatorname{rank}_+(M)$ *then* $\operatorname{rank}(M) = \operatorname{rrank}_+(M)$.

As mentioned in Section 5.3, if $\operatorname{rank}(M) \leq 2$ then $\operatorname{rank}(M) = \operatorname{rank}_+(M)$, and therefore $\operatorname{rank}(M) = \operatorname{rank}_+(M) = \operatorname{rrank}_+(M)$ by Lemma 51.

The restricted nonnegative rank does not share all of the properties of the rank and the nonnegative rank. Indeed, in contrast to Lemma 50, in general $\operatorname{rrank}_+(M) \not\leq n$ and $\operatorname{rrank}_+(M) \neq \operatorname{rrank}_+(M^\top)$. For example, Gillis and Glineur [2012, Lemma 1] show that the following matrix:

$$M = \begin{pmatrix} 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 \end{pmatrix} \in \mathbb{R}_+^{6\times 8} \tag{5.1}$$

satisfies $\text{rank}(M) = 4$, $\text{rank}_+(M) = 6$, and $\text{rrank}_+(M) = 8$.[2] Moreover, by Lemma 51 we have $\text{rrank}_+(M^\top) \leq 6$ and therefore $\text{rrank}_+(M^\top) \neq \text{rrank}_+(M)$.

The *RNMF problem* is: Given a matrix $M \in \mathbb{Q}_+^{n \times m}$ and $k \in \mathbb{N}$, is $\text{rrank}_+(M) \leq k$? With the above-mentioned NP-hardness result of Vavasis [2009], namely that the NMF problem is NP-hard even for $k = \text{rank}(M)$, it follows that the RNMF problem is also NP-hard and in PSPACE.

For a matrix $M \in \mathbb{Q}_+^{n \times m}$, its restricted nonnegative rank over $\mathbb{Q}$ is clearly at least $\text{rrank}_+(M)$. As with nonnegative rank, a natural question is whether the restricted nonnegative ranks of $M$ over $\mathbb{R}$ and over $\mathbb{Q}$ are equal. By [Cohen and Rothblum, 1993, Theorem 4.1] and Lemma 51, this is true when $\text{rank}(M) \leq 2$. In Chapter 6 we show that this remains true when $\text{rank}(M) \leq 3$, but in general not when $\text{rank}(M) \geq 4$.

## 5.5 Geometry

Given $l \in \mathbb{N}$, a *linear combination* of a set $\{v_1, \ldots, v_m\} \subset \mathbb{R}^l$ is a point $\lambda_1 v_1 + \cdots + \lambda_m v_m$ where $\lambda_1, \ldots, \lambda_m \in \mathbb{R}$. The *(linear) span* of $\{v_1, \ldots, v_m\}$, written as $\text{span}\{v_1, \ldots, v_m\}$, is the set of all linear combinations of $\{v_1, \ldots, v_m\}$. We say that the set $\{v_1, \ldots, v_m\}$ is *linearly independent* if whenever $\lambda_1 v_1 + \cdots + \lambda_m v_m = \mathbf{0}$ then $\lambda_1 = \cdots = \lambda_m = 0$. A subset $V$ of $\mathbb{R}^l$ is a *(linear) subspace* if it is closed under addition and scalar multiplication. The dimension of a subspace $V \subseteq \mathbb{R}^l$, written $\dim(V)$, is the size of a largest linearly independent set of vectors contained in $V$.

An *affine combination* of a set $\{v_1, \ldots, v_m\} \subset \mathbb{R}^l$ is a point $\lambda_1 v_1 + \cdots + \lambda_m v_m$ where $\lambda_1, \ldots, \lambda_m \in \mathbb{R}$ are such that $\lambda_1 + \cdots + \lambda_m = 1$. The *affine span* of $\{v_1, \ldots, v_m\}$ is the set of all affine combinations of $\{v_1, \ldots, v_m\}$. The set $\{v_1, \ldots, v_m\} \subset \mathbb{R}^l$ is called *affinely independent* if whenever $\lambda_1 v_1 + \cdots + \lambda_m v_m = \mathbf{0}$ and $\lambda_1 + \cdots + \lambda_m = 0$ then $\lambda_1 = \cdots = \lambda_m = 0$. Equivalently, a set $\{v_1, \ldots, v_m\} \subset \mathbb{R}^l$ is affinely independent if

[2] We later describe in Example 4 how the result $\text{rrank}_+(M) = 8$ follows from the geometric interpretation of the restricted nonnegative rank.

the set

$$\left\{ \begin{pmatrix} v_1 \\ 1 \end{pmatrix}, \ldots, \begin{pmatrix} v_m \\ 1 \end{pmatrix} \right\} \subset \mathbb{R}^{l+1}$$

is linearly independent [see, e.g., Lauritzen, 2009, Lemma 2.6.2]. A subset $S$ of $\mathbb{R}^l$ is an *affine subspace* if $S = x + V = \{ x + v \mid v \in V \}$ for some vector $x \in \mathbb{R}^l$ and linear subspace $V \subseteq \mathbb{R}^l$. In this case the *dimension* of $S$, written $\dim(S)$, is defined to be the dimension of $V$.

A nonempty set $\mathcal{C} \subseteq \mathbb{R}^l$ is a *cone* if $\lambda v + \mu w \in \mathcal{C}$ for every $v, w \in \mathcal{C}$ and $\lambda, \mu \geq 0$. A cone $\mathcal{C}$ is *polyhedral* if $\mathcal{C} = \{ x \in \mathbb{R}^l \mid Ax \geq \mathbf{0} \}$ for some matrix $A \in \mathbb{R}^{n \times l}$. By Minkowski-Weyl's Theorem (for cones), a cone $\mathcal{C}$ is polyhedral if and only if there is a finite set $\{v_1, \ldots, v_m\} \subset \mathbb{R}^l$ such that $\mathcal{C} = \{ \lambda_1 v_1 + \cdots + \lambda_m v_m \mid \lambda_1, \ldots, \lambda_m \geq 0 \}$; in this case we say that $\mathcal{C}$ is *generated by* the vectors $v_1, \ldots, v_m$.

A *convex combination* of a set $\{v_1, \ldots, v_m\} \subset \mathbb{R}^l$ is a point $\lambda_1 v_1 + \cdots + \lambda_m v_m$ where $(\lambda_1, \ldots, \lambda_m)$ is a stochastic vector. The *convex hull* of $\{v_1, \ldots, v_m\}$, written as $\operatorname{conv}\{v_1, \ldots, v_m\}$, is the set of all convex combinations of $\{v_1, \ldots, v_m\}$. We call $\operatorname{conv}\{v_1, \ldots, v_m\}$ a *polytope spanned by* $v_1, \ldots, v_m$. Equivalently, a *polytope* is a bounded set $\{ x \in \mathbb{R}^l \mid Ax + b \geq \mathbf{0} \}$ with $A \in \mathbb{R}^{n \times l}$ and $b \in \mathbb{R}^n$. When $l = 2$, a polytope is called a *polygon*. The *dimension* of a polytope $\mathcal{P}$ is the maximum number of affinely independent points in $\mathcal{P}$ minus 1. A polytope $\mathcal{P} \subseteq \mathbb{R}^l$ is *full-dimensional* if it has dimension $l$, i.e, if it is not contained in any affine subspace except $\mathbb{R}^l$. Intuitively, a polytope is full-dimensional if it has volume.

**Lemma 52.** *If a polytope $\mathcal{P} = \{ x \in \mathbb{R}^l \mid Ax + b \geq \mathbf{0} \}$ is full-dimensional, then the matrix $(A \;\; b)$ has rank equal to its number of columns $l + 1$.*

*Proof.* Towards a contradiction, suppose that $\operatorname{rank}((A \;\; b)) < l + 1$. Then either $\operatorname{rank}(A) < l$ or $b \in \operatorname{Col}(A)$.

If $\operatorname{rank}(A) < l$, then $\ker(A) \neq \{\mathbf{0}\}$, i.e., there exists a vector $x \in \mathbb{R}^l \setminus \{\mathbf{0}\}$ such

that $Ax = \mathbf{0}$. Now for any $v \in \mathcal{P}$ and $\lambda \in \mathbb{R}$, we have

$$A(\lambda x + v) + b = Av + b \geq \mathbf{0}$$

and therefore $\lambda x + v \in \mathcal{P}$. This implies that $\mathcal{P}$ is unbounded, which is a contradiction.

If $b \in \text{Col}(A)$, then there exists $y \in \mathbb{R}^l$ such that $Ay = b$. Consider the polyhedral cone $\mathcal{C} = \{\, x \in \mathbb{R}^l \mid Ax \geq \mathbf{0} \,\}$. For any $v \in \mathcal{P}$, we have $A(v + y) = Av + b \geq \mathbf{0}$ and thus $v + y \in \mathcal{C}$. For any $x \in \mathcal{C}$, we have $A(x - y) + b = Ax \geq \mathbf{0}$ and thus $x - y \in \mathcal{P}$. Therefore, $\mathcal{C} = \mathcal{P} + y = \{\, v + y \mid v \in \mathcal{P} \,\}$. Since $\mathcal{P}$ is full-dimensional, we have $\mathcal{C} \neq \{\mathbf{0}\}$ and therefore $\mathcal{C}$ is unbounded. But then $\mathcal{P}$ is also unbounded, which yields a contradiction. The result follows. □

### 5.5.1 Nested Polytope Problem

Given $l, n \in \mathbb{N}$, let $A \in \mathbb{Q}^{n \times l}$ and $b \in \mathbb{Q}^n$ be such that $\mathcal{P} = \{\, x \in \mathbb{R}^l \mid Ax + b \geq \mathbf{0} \,\}$ is a full-dimensional polytope. Let $\mathcal{R} = \text{conv}\{r_1, \ldots, r_m\} \subseteq \mathcal{P}$ be a full-dimensional polytope described by spanning points $r_1, \ldots, r_m$. The *nested polytope problem (NPP)* asks, given $A$, $b$, $r_1, \ldots, r_m$, and a number $k \in \mathbb{N}$, whether there exist $k$ points in $\mathbb{R}^l$ that span a polytope $\mathcal{Q}$ with $\mathcal{R} \subseteq \mathcal{Q} \subseteq \mathcal{P}$. Such a polytope $\mathcal{Q}$ is said to be *nested between $\mathcal{R}$ and $\mathcal{P}$*.

### 5.5.2 Nested Polygon Problem

When $l = 2$, the nested polytope problem becomes the *nested polygon problem* in the plane. Given two polygons $\mathcal{R} \subseteq \mathcal{P} \subseteq \mathbb{R}^2$, a polygon $\mathcal{Q} \subseteq \mathbb{R}^2$ nested between $\mathcal{R}$ and $\mathcal{P}$ (i.e., such that $\mathcal{R} \subseteq \mathcal{Q} \subseteq \mathcal{P}$) is called *minimal* if it has the minimum number of vertices among all polygons nested between $\mathcal{R}$ and $\mathcal{P}$. Figure 5.1 illustrates a nested polygon. Next, we recall from [Aggarwal et al., 1989] a standardised form for minimal nested polygons, which will play an important role in the subsequent development in

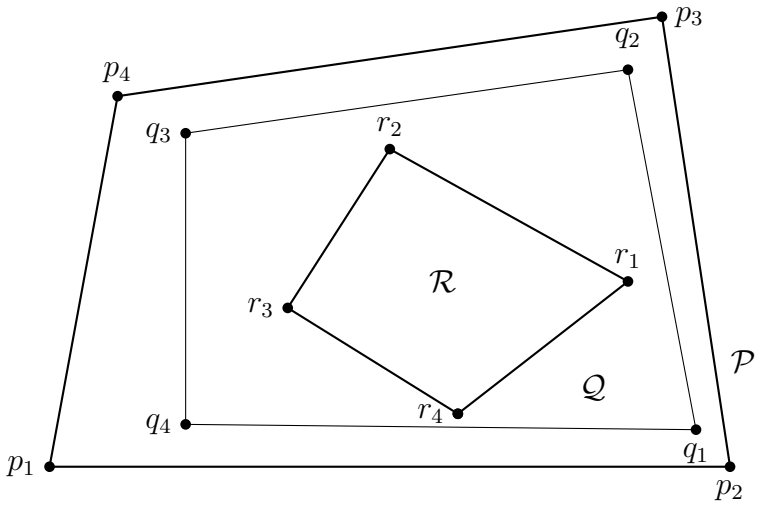



Figure 5.1: Polygon $\mathcal{Q}$ is nested between polygons $\mathcal{R}$ and $\mathcal{P}$.

Chapters 6 and 7.

Fix two polygons $\mathcal{R}$ and $\mathcal{P}$, with $\mathcal{R} \subseteq \mathcal{P}$. A *supporting line segment* is a directed line segment that has both endpoints on the boundary of the outer polygon $\mathcal{P}$ and touches the inner polygon $\mathcal{R}$ on its left. A nested polygon with vertices $q_1, \ldots, q_k$, listed in anti-clockwise order, is said to be *supporting* if the directed line segments $q_1q_2, q_2q_3, \ldots, q_{k-1}q_k$ are all supporting. Such a polygon is uniquely determined by the vertex $q_1$ [see Aggarwal et al., 1989, Section 2] and is henceforth denoted by $\mathcal{S}_{q_1}$. Figure 5.2 illustrates a supporting polygon.

It is shown in [Aggarwal et al., 1989] that there is guaranteed to exist a nested polygon that is both minimal and supporting. More specifically, we have:

**Lemma 53** (Aggarwal et al., 1989, Lemma 4)**.** *Consider a minimal nested polygon with vertices $q_1, \ldots, q_k$, listed in anti-clockwise order, where $q_1$ lies on the boundary of $\mathcal{P}$. The supporting polygon $\mathcal{S}_{q_1}$ is also minimal.*

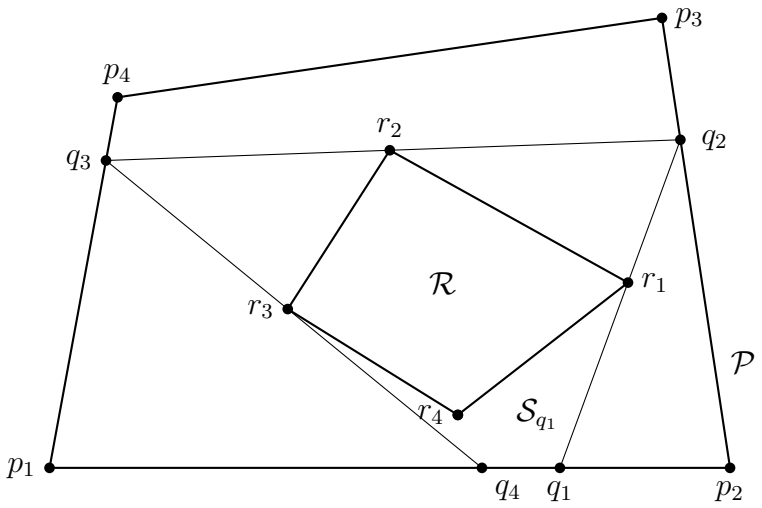



Figure 5.2: Polygon $\mathcal{S}_{q_1}$ is nested between polygons $\mathcal{R}$ and $\mathcal{P}$ of Figure 5.1 and is supporting since $q_1q_2, q_2q_3, q_3q_4$ are supporting line segments.

# Chapter 6

# Restricted Nonnegative Matrix Factorization Requires Irrationality

**Abstract**

*Nonnegative matrix factorization (NMF) is the problem of decomposing a given nonnegative $n \times m$ matrix $M$ into a product of a nonnegative $n \times d$ matrix $W$ and a nonnegative $d \times m$ matrix $H$. Restricted NMF requires in addition that the column spaces of $M$ and $W$ coincide. Finding the minimal inner dimension $d$ is known to be* NP*-hard, both for NMF and restricted NMF. We investigate whether a rational matrix $M$ always has a restricted NMF of minimal inner dimension whose factors $W$ and $H$ are also rational. We show that this holds for matrices $M$ of rank at most 3 and we exhibit a rank-4 matrix for which $W$ and $H$ require irrational entries. This answers the restricted NMF variant of the Cohen–Rothblum problem negatively. This chapter is an extension of the material in [Chistikov et al., 2016].*

---

## 6.1 Introduction

Nonnegative matrix factorization (NMF) is the task of factoring a nonnegative matrix $M$ as a product $M = W \cdot H$ such that matrices $W$ and $H$ are also nonnegative. We recall from Chapter 5 that the smallest inner dimension of any such factorization is called the *nonnegative rank* of $M$, written $\text{rank}_+(M)$. The *NMF problem* asks whether a given nonnegative matrix of rational numbers $M$ admits an NMF with inner dimension at most a given number $k$.

In a classical rank factorization $M = W \cdot H$, it holds that $\text{Col}(M) = \text{Col}(W)$. However, this need not be the case in a nonnegative factorization. Indeed, for a general NMF $M = W \cdot H$ of minimal inner dimension, we may have that $\text{Col}(M) \subsetneq \text{Col}(W)$. This motivates the study of nonnegative matrix factorizations $M = W \cdot H$ that have the property that $\text{Col}(M) = \text{Col}(W)$; such an NMF is called *restricted (RNMF)*. (Note that for any NMF, the column space of $M$ is a subspace of the column space of $W$.) The smallest inner dimension of any restricted NMF of $M$ is called the *restricted nonnegative rank* of $M$, written $\text{rrank}_+(M)$. Restricted NMFs were introduced and studied by Gillis and Glineur [2012].

An even more restricted class of nonnegative matrix factorizations, called *separable NMF*, has been widely studied and shown to have good algorithmic properties as well as a natural interpretation in terms of topic modelling. The notion of separability was introduced by Donoho and Stodden [2003]. Formally, an NMF $M = W \cdot H$ is *separable* if every column of $W$ is also a column of $M$, i.e., if for every row of $H$ there is a column of $H$ whose only nonzero entry lies in that row. In the context of topic modelling problems where $M$ is a document-word matrix and $W$, $H$ are document-topic and topic-word matrices, respectively, the separability assumption means that every topic has an *anchor word*, occurring in no other topic. If a matrix $M$ has such a factorization then there is a rational one that can moreover be found in polynomial time [Arora et al., 2012a].

In this chapter we focus on the *restricted* NMF (RNMF) problem, which is defined as the NMF problem except that the factorization is required to be restricted, i.e., the column spaces of $M$ and $W$ are required to coincide. The RNMF problem has a natural geometric interpretation as the *nested polytope problem (NPP)*: the problem of finding a minimum-vertex polytope nested between two given convex polytopes. In more detail, for a rank-$r$ matrix $M$, finding an RNMF with inner dimension $d$ is known to correspond exactly to finding a nested polytope with $d$ vertices in an $(r-1)$-dimensional NPP. This geometric connection is further described in Section 6.2.

As mentioned in Chapter 5, Cohen and Rothblum [1993] posed the question of whether, given a nonnegative matrix of rational numbers $M$, there always exists an NMF $M = W \cdot H$ of inner dimension equal to $\text{rank}_+(M)$ such that both $W$ and $H$ are also matrices of rational numbers. In this chapter, we answer the restricted NMF variant of the Cohen–Rothblum problem negatively.

We first show, in Section 6.3, that for rational matrices $M$ of rank 3 or less, there is always a rational RNMF of $M$ with inner dimension $\text{rrank}_+(M)$, and that it can be computed in polynomial time in the Turing model of computation. In particular, the RNMF problem for matrices of rank 3 or less can be solved in polynomial time. This improves a result in [Gillis and Glineur, 2012] where the RNMF problem for matrices of rank 3 or less is shown to be solvable in polynomial time assuming a RAM model with unit-cost arithmetic. Both our algorithm and the one of Gillis and Glineur [2012] exploit the connection to the 2-dimensional NPP, allowing us to take advantage of a geometric algorithm by Aggarwal et al. [1989]. We need to adapt the latter algorithm to ensure that the occurring numbers are rational and can be computed in polynomial time in the Turing model of computation.

In Section 6.4, we exhibit a rank-4 rational matrix that has an RNMF with inner dimension 5 but *no rational* RNMF with inner dimension 5. We construct this matrix via a particular instance of the 3-dimensional NPP, again taking advantage of the

geometric interpretation of RNMF.

## 6.2 Geometric Interpretation

The restricted NMF (RNMF) problem has a geometric interpretation as the nested polytope problem (NPP), introduced in Section 5.5.1.

The following proposition appears as Theorem 1 in [Gillis and Glineur, 2012].

**Proposition 54.** *The RNMF problem and the NPP are interreducible in polynomial time.*

More specifically, the reductions are as follows.

1. Given a nonnegative matrix $M \in \mathbb{Q}_+^{n\times m}$ of rank $r$, one can compute in polynomial time a matrix $A \in \mathbb{Q}^{n\times(r-1)}$ and vectors $b \in \mathbb{Q}^n$ and $r_1, \ldots, r_m \in \mathbb{Q}^{r-1}$ such that polytopes $\mathcal{P} = \{\, x \in \mathbb{R}^{r-1} \mid Ax + b \geq \mathbf{0} \,\}$ and $\mathcal{R} = \operatorname{conv}\{r_1, \ldots, r_m\}$ are full-dimensional and satisfy $\mathcal{R} \subseteq \mathcal{P}$, and such that for any $d \in \mathbb{N}$:

   (a) any $d$-dimensional RNMF (rational or irrational) of $M$ determines $d$ points that span a polytope $\mathcal{Q}$ with $\mathcal{R} \subseteq \mathcal{Q} \subseteq \mathcal{P}$, and

   (b) any $d$ points (rational or irrational) that span a polytope $\mathcal{Q}$ with $\mathcal{R} \subseteq \mathcal{Q} \subseteq \mathcal{P}$ determine a $d$-dimensional RNMF of $M$.

2. Let $A \in \mathbb{Q}^{n\times(r-1)}$ and $b \in \mathbb{Q}^n$ be such that $\mathcal{P} = \{\, x \in \mathbb{R}^{r-1} \mid Ax + b \geq \mathbf{0} \,\}$ is a full-dimensional polytope. Let $\mathcal{R} \subseteq \mathcal{P}$ be a full-dimensional polytope spanned by $r_1, \ldots, r_m \in \mathbb{Q}^{r-1}$. Then, matrix $M \in \mathbb{Q}_+^{n\times m}$ with $M_{:,i} = Ar_i + b$ for $i \in [m]$ satisfies (a) and (b).

Importantly, the correspondences (a) and (b) preserve rationality. In the following we detail the reduction from point 2 above, thereby filling in a small gap in the proof of Gillis and Glineur [2012].

*Proof of point 2 from Proposition 54.* Let $A \in \mathbb{Q}^{n\times(r-1)}$ and $b \in \mathbb{Q}^n$ be such that $\mathcal{P} = \{\, x \in \mathbb{R}^{r-1} \mid Ax + b \geq \mathbf{0} \,\}$ is a full-dimensional polytope. Hence, by Lemma 52, matrix $(A \;\; b) \in \mathbb{Q}^{n\times r}$ has full column rank $r$. Let $\mathcal{R} = \operatorname{conv}\{r_1, \ldots, r_m\} \subseteq \mathcal{P}$ be a full-dimensional polytope. Define a matrix $M \in \mathbb{Q}_+^{n\times m}$ with $M_{:,j} = Ar_j + b$ for $j \in [m]$. That is,

$$M = (A \;\; b) \begin{pmatrix} r_1 & r_2 & \cdots & r_m \\ 1 & 1 & \cdots & 1 \end{pmatrix}. \tag{6.1}$$

This implies that $\operatorname{Col}(M) \subseteq \operatorname{Col}((A \;\; b))$. Since polytope $\mathcal{R} = \operatorname{conv}\{r_1, \ldots, r_m\} \subset \mathbb{R}^{r-1}$ is full-dimensional, there exist $r$ affinely independent points in $\mathcal{R}$. This means that the (column) rank of the matrix

$$\begin{pmatrix} r_1 & r_2 & \cdots & r_m \\ 1 & 1 & \cdots & 1 \end{pmatrix}$$

is $r$, hence this matrix has full row rank and therefore also has a right inverse. It now follows from (6.1) that $\operatorname{Col}((A \;\; b)) \subseteq \operatorname{Col}(M)$ and therefore

$$\operatorname{Col}((A \;\; b)) = \operatorname{Col}(M). \tag{6.2}$$

We now show the correspondences (a) and (b) from 1:

To prove (a), consider an RNMF $M = W \cdot H$ with inner dimension $d$. We can, without loss of generality, assume that $W$ has no zero columns. Since the NMF $M = W \cdot H$ is restricted, by (6.2) we have

$$\operatorname{Col}(W) = \operatorname{Col}(M) = \operatorname{Col}((A \;\; b)).$$

Therefore, there exists a matrix $C \in \mathbb{R}^{r\times d}$ such that $W = (A \;\; b) \cdot C$. For all $i \in [d]$,

we define $\hat{c}_i \in \mathbb{R}^{r-1}$ so that

$$C_{:,i} = \begin{pmatrix} \hat{c}_i \\ C_{r,i} \end{pmatrix}.$$

Since matrix $W$ is nonnegative, we have $W_{:,i} = A\hat{c}_i + bC_{r,i} \geq \mathbf{0}$. Observe that there is no $y \in \mathbb{R}^{r-1} \setminus \{\mathbf{0}\}$ with $Ay \geq \mathbf{0}$: indeed, if $Ay \geq \mathbf{0}$ for some nonzero $y$ then for any $x \in \mathcal{P}$ and any $t > 0$ we would have $A(x + ty) + b \geq Ax + b \geq \mathbf{0}$, implying that $\mathcal{P}$ is unbounded, which is false. We use this observation to show that $C_{r,i} > 0$. Towards a contradiction, suppose $C_{r,i} \leq 0$. Then, one of the following two cases holds:

- If $C_{r,i} = 0$, then $A\hat{c}_i \geq \mathbf{0}$. Moreover $\hat{c}_i \neq \mathbf{0}$ since $W$ has no zero columns. This contradicts the observation above.

- If $C_{r,i} < 0$, then by dividing the inequality $A\hat{c}_i + bC_{r,i} \geq \mathbf{0}$ by $-C_{r,i}$ we obtain

$$A\left(-\frac{\hat{c}_i}{C_{r,i}}\right) - b \geq \mathbf{0}. \tag{6.3}$$

  Let $x \in \mathcal{P} \setminus \{\frac{\hat{c}_i}{C_{r,i}}\}$. Then $Ax + b \geq \mathbf{0}$. By adding the inequality (6.3), we obtain $A\left(x - \frac{\hat{c}_i}{C_{r,i}}\right) \geq \mathbf{0}$. Since $x \neq \frac{\hat{c}_i}{C_{r,i}}$, this also contradicts the observation above.

We conclude that $C_{r,i} > 0$ for all $i \in [d]$. Define a diagonal matrix $D \in \mathbb{R}^{d \times d}$ such that $D_{i,i} = C_{r,i} > 0$, and define a matrix $H' := D \cdot H$. Then we have:

$$\begin{aligned}
(A \;\; b)\begin{pmatrix} r_1 & r_2 & \cdots & r_m \\ 1 & 1 & \cdots & 1 \end{pmatrix} = M = W \cdot H &= (A \;\; b)\, C \cdot H \\
&= (A \;\; b)\, CD^{-1} \cdot DH \\
&= (A \;\; b)\begin{pmatrix} \frac{\hat{c}_1}{C_{r,1}} & \frac{\hat{c}_2}{C_{r,2}} & \cdots & \frac{\hat{c}_d}{C_{r,d}} \\ 1 & 1 & \cdots & 1 \end{pmatrix} H'.
\end{aligned}$$

Since the columns of $(A \;\; b)$ are linearly independent, it follows that:

$$\begin{pmatrix} r_1 & r_2 & \cdots & r_m \\ 1 & 1 & \cdots & 1 \end{pmatrix} = \begin{pmatrix} \frac{\hat{c}_1}{C_{r,1}} & \frac{\hat{c}_2}{C_{r,2}} & \cdots & \frac{\hat{c}_d}{C_{r,d}} \\ 1 & 1 & \cdots & 1 \end{pmatrix} H'.$$

Considering the last rows, we see that each column of $H'$ sums up to 1. Since $H'$ is moreover nonnegative, $H'$ is stochastic. For every $i \in [d]$, define $q_i := \frac{\hat{c}_i}{C_{r,i}}$. Then for all $j \in [m]$ we have $r_j \in \operatorname{conv}\{q_1, \ldots, q_d\}$. Hence, defining the polytope $\mathcal{Q} := \operatorname{conv}\{q_1, \ldots, q_d\}$ we have $\mathcal{R} \subseteq \mathcal{Q}$. Furthermore, for all $i \in [d]$ we have $A\hat{c}_i + bC_{r,i} \geq \mathbf{0}$, hence $Aq_i + b \geq \mathbf{0}$ since $C_{r,i} > 0$. It follows that $\mathcal{Q} \subseteq \mathcal{P}$. Thus, $\mathcal{Q}$ is nested between $\mathcal{R}$ and $\mathcal{P}$, which completes the proof of correspondence (a).

To prove (b), consider $d$ points $q_1, \ldots, q_d \in \mathbb{R}^{r-1}$ with $\mathcal{R} \subseteq \operatorname{conv}\{q_1, \ldots, q_d\} \subseteq \mathcal{P}$. Define a matrix $W \in \mathbb{Q}^{n \times d}$ where $W_{:,i} = Aq_i + b$ for all $i \in [d]$. Matrix $W$ is nonnegative since $q_i \in \mathcal{P}$ for all $i \in [d]$. By definition, $\operatorname{Col}(W) \subseteq \operatorname{Col}((A \;\; b))$. Define a stochastic matrix $H \in \mathbb{Q}_+^{d \times m}$ so that $r_j = \sum_{i=1}^{d} H_{i,j} q_i$ for all $j \in [m]$. Such entries $H_{i,j}$ exist as $r_j \in \operatorname{conv}\{q_1, \ldots, q_d\}$. We have for all $j \in [m]$:

$$\begin{aligned}
M_{:,j} &= Ar_j + b && \text{by definition of } M \\
&= A\left(\sum_{i=1}^{d} H_{i,j} q_i\right) + b && \text{by definition of } H \\
&= \sum_{i=1}^{d} H_{i,j} \cdot Aq_i + \sum_{i=1}^{d} H_{i,j} \cdot b && \text{as } H \text{ is stochastic} \\
&= \sum_{i=1}^{d} H_{i,j} \cdot (\underbrace{Aq_i + b}_{W_{:,i}}) \\
&= (W \cdot H)_{:,j}\,.
\end{aligned}$$

Hence, $M = W \cdot H$ is a $d$-dimensional NMF. This NMF is restricted since

$$\operatorname{Col}(W) \subseteq \operatorname{Col}((A \;\; b)) \overset{\text{Eq. (6.2)}}{=} \operatorname{Col}(M).$$

This completes the proof of correspondence (b). □

**Example 4** (Gillis and Glineur, 2012, Example 1)**.** *Consider an NPP instance where the inner and outer polytope are the standard 3D cube, i.e.,*

$$\mathcal{R} = \mathcal{P} = \{\, x \in \mathbb{R}^3 \mid x_i \in [0,1],\ i \in [3] \,\}.$$

*The corresponding RNMF problem consists of the matrix from* (5.1)*:*

$$M = \begin{pmatrix} 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 \end{pmatrix} \in \mathbb{R}_+^{6\times 8}.$$

*Clearly, the only polytope nested between $\mathcal{R}$ and $\mathcal{P}$ is $\mathcal{Q} = \mathcal{R} = \mathcal{P}$, which has 8 vertices. It now follows from Proposition 54 that* $\operatorname{rrank}_+(M) = 8$.

## 6.3 Restricted NMF of Rank-3 Matrices

In this section we consider rational matrices of rank at most 3. We show that for such matrices, the restricted nonnegative ranks over $\mathbb{R}$ and $\mathbb{Q}$ are equal and we give a polynomial-time algorithm that computes a minimal-dimension RNMF over $\mathbb{Q}$. Throughout this section, when we say polynomial time, we mean polynomial time in the Turing machine model of computation.

**Theorem 55.** *Given a matrix $M \in \mathbb{Q}_+^{n\times m}$ where* $\operatorname{rank}(M) \leq 3$*, there is a rational RNMF of $M$ with inner dimension* $\operatorname{rrank}_+(M)$ *and it can be computed in polynomial time.*

Using reduction 1 of Proposition 54, we can reduce in polynomial time the RNMF problem for rank-3 matrices to the 2-dimensional NPP, i.e., the nested polygon problem, introduced in Section 5.5.2. As noted in Section 6.2, the correspondence between restricted nonnegative factorizations and nested polygons preserves rationality. Thus to prove Theorem 55 it suffices to prove the following result:

**Theorem 56.** *Given polygons $\mathcal{R} \subseteq \mathcal{P} \subseteq \mathbb{R}^2$ with rational vertices, there exists a minimum-vertex polygon $\mathcal{Q}$ nested between $\mathcal{R}$ and $\mathcal{P}$ that also has rational vertices. Moreover there is an algorithm that, given $\mathcal{R}$ and $\mathcal{P}$, computes such a polygon in polynomial time.*

In fact, Aggarwal et al. [1989] give an algorithm for the nested polygon problem and prove that it runs in polynomial time in the RAM model with unit-cost arithmetic. However, they freely use trigonometric functions and do not address the rationality of the output of the algorithm nor its complexity in the Turing model. To prove Theorem 56 we show that, by adopting a suitable representation of the vertices of a nested polygon, the algorithm in [Aggarwal et al., 1989] can be adapted so that it runs in polynomial time under our assumption of the Turing model of computation. We furthermore use this representation to prove that the minimum-vertex nested polygon identified by the resulting algorithm has rational vertices.

The remainder of this section is devoted to the proof of Theorem 56. In Section 5.5.2 we recall the definition by Aggarwal et al. [1989] of a *supporting polygon*, which is a special type of a nested polygon. As stated in Lemma 53, they show that there is always a supporting polygon that is also minimal, and give an algorithm that outputs such a polygon.

Let $k$ be the number of vertices of a minimal polygon nested between $\mathcal{R}$ and $\mathcal{P}$. Given a vertex $v$ on the boundary of $\mathcal{P}$, there is a uniquely defined supporting polygon $\mathcal{S}_v$ with at most $k+1$ vertices. To determine $\mathcal{S}_v$ one computes the supporting line segments $v_1v_2, \ldots, v_kv_{k+1}$, where $v_1 = v$; see Figure 6.1. Then $\mathcal{S}_v$ is either the

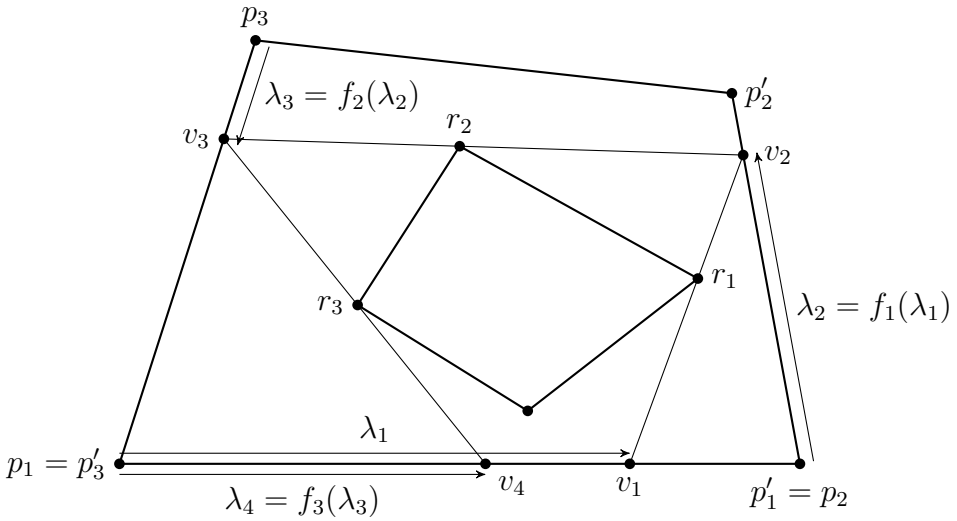



Figure 6.1: Supporting polygon $\mathcal{S}_{v_1}$. For every $i \in [3]$, vertex $v_i$ lies on edge $p_i p'_i$ of $\mathcal{P}$, and $r_i$ is the point where the supporting line segment $v_i v_{i+1}$ touches the inner polygon $\mathcal{R}$ on its left.

polygon with vertices $v_1, \ldots, v_k$ or the polygon with vertices $v_1, \ldots, v_{k+1}$. In the first case, $\mathcal{S}_v$ is minimal. The idea behind the algorithm of Aggarwal et al. [1989] is to search along the boundary of $\mathcal{P}$ for an initial vertex $v$ such that $\mathcal{S}_v$ is minimal.

As a central ingredient for our proof of Theorem 56, we choose a convenient representation of the vertices of supporting polygons. To this end, we assume that the edges of $\mathcal{P}$ are oriented counter-clockwise, and we represent a vertex $v$ on an edge $pq$ of $\mathcal{P}$ by the unique $\lambda \in [0, 1]$ such that $v = (1 - \lambda)p + \lambda q$. We call this the *convex representation* of $v$.

Similar to [Aggarwal et al., 1989], we associate with each supporting line segment $uv$ a *ray function* $f$, such that if $\lambda$ is the convex representation of $u$ then $f(\lambda)$ is the convex representation of $v$. The same ray function applies for supporting line segments $u'v'$ with $u'$ in a small enough interval containing $u$.

To obtain a polynomial time bound, the key lemma is as follows:

**Lemma 57.** *Consider polygons $\mathcal{R} \subseteq \mathcal{P} \subseteq \mathbb{R}^2$ whose vertices are rational and of bit-length $L$. Then the ray function associated with a supporting line segment $uv$*

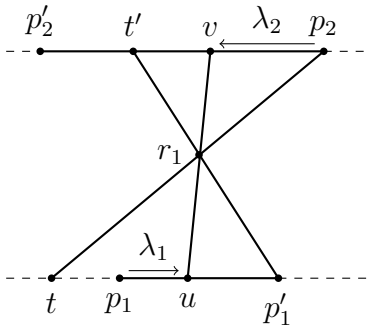



Figure 6.2: Lines $p_1p_1'$ and $p_2p_2'$ are parallel.

*has the form* $f(\lambda) = \frac{a\lambda+b}{c\lambda+d}$*, where the coefficients* $a, b, c, d$ *are rational numbers with bit-length* $O(L)$ *that can be computed in polynomial time.*

*Proof.* Let vertices $u$ and $v$ lie on edges $p_1p_1'$ and $p_2p_2'$ of $\mathcal{P}$, respectively. Let $r_1$ be the point where the supporting line segment $uv$ touches the inner polygon $\mathcal{R}$ on its left. Let $\lambda_1$ and $\lambda_2$ be the convex representations of $u$ and $v$, respectively. That is, $u = (1-\lambda_1)p_1 + \lambda_1 p_1'$ and $v = (1-\lambda_2)p_2 + \lambda_2 p_2'$. By definition of the ray function $f$ we have $\lambda_2 = f(\lambda_1)$.

Let us first consider the case when lines $p_1p_1'$ and $p_2p_2'$ are parallel; see Figure 6.2 for illustration. Let $t$ denote the intersection of lines $p_1p_1'$ and $r_1p_2$, and let $t'$ denote the intersection of lines $p_2p_2'$ and $r_1p_1'$. Since $r_1 \in \mathbb{Q}^2$, we have $t = (1-b)\cdot p_1 + b \cdot p_1'$ and $t' = (1-d)\cdot p_2 + d \cdot p_2'$ for some $b, d \in \mathbb{Q}$. Numbers $b, d$ can be computed from the input vertices $p_1, p_1', p_2, p_2'$ using a constant number of arithmetic operations and therefore have bit-length $O(L)$. Since $\triangle tur_1 \sim \triangle p_2vr_1$ and $\triangle tp_1'r_1 \sim \triangle p_2t'r_1$, we have $\frac{ut}{vp_2} = \frac{tr_1}{r_1p_2}$ and $\frac{tr_1}{r_1p_2} = \frac{p_1't}{t'p_2}$. Hence, $\frac{ut}{vp_2} = \frac{p_1't}{t'p_2}$ and therefore $\frac{\lambda_1-b}{\lambda_2} = \frac{1-b}{d}$. Hence,

$$\lambda_2 = \frac{d}{1-b}\cdot\lambda_1 - \frac{bd}{1-b}$$

where the coefficients $\frac{d}{1-b}, \frac{bd}{1-b} \in \mathbb{Q}$ have bit-length $O(L)$.

Let us now consider the second case: when lines $p_1p_1'$ and $p_2p_2'$ are not parallel. Let $t$ denote their intersection; see Figure 6.3 for illustration. Note that $t \in \mathbb{Q}^2$.

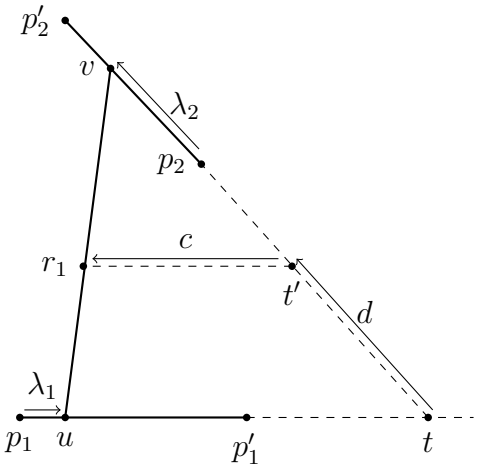



Figure 6.3: Lines $p_1p_1'$ and $p_2p_2'$ intersect at point $t$.

Without loss of generality, let $p_1'$ be a convex combination of $\{p_1, t\}$ and $p_2$ be a convex combination of $\{t, p_2'\}$. Then $p_1' = (1-a)\cdot p_1 + a\cdot t$ and $p_2 = (1-b)\cdot t + b\cdot p_2'$ for some $a, b \in [0,1]$. Numbers $a, b$ are rational as they are each the unique solution of a linear system with rational coefficients. Moreover, $a$ and $b$ can be computed from the input vertices $p_1, p_1', p_2, p_2'$ using a constant number of arithmetic operations and therefore have bit-length $O(L)$. Using the above representation of $p_1'$ as a convex combination of $\{p_1, t\}$, we get

$$u = (1-\lambda_1)\cdot p_1 + \lambda_1\cdot((1-a)\cdot p_1 + a\cdot t) = (1-a\lambda_1)\cdot p_1 + a\lambda_1\cdot t$$

and therefore $u - t = (1-a\lambda_1)\cdot(p_1 - t)$. Similarly, we have

$$v = (1-b-\lambda_2+\lambda_2 b)\cdot t + (b+\lambda_2-\lambda_2 b)\cdot p_2'$$

and therefore $v - t = (b+\lambda_2-\lambda_2 b)\cdot(p_2' - t)$. Let the line through $r_1$ parallel with $p_1t$ intersect $p_2't$ at point $t'$. Note that $r_1 - t' = c\cdot(p_1 - t)$ and $t' - t = d\cdot(p_2' - t)$ for some $c, d \in [0,1]\cap\mathbb{Q}$. Numbers $c$ and $d$ can be computed from vertices $p_1, p_1', p_2, p_2'$ using a constant number of arithmetic operations and thus have bit-length $O(L)$. Since

$\triangle vt'r_1 \sim \triangle vtu$, it holds that $\frac{vt'}{vt} = \frac{r_1t'}{ut}$. We therefore have that

$$\frac{b + \lambda_2 - \lambda_2 b - d}{b + \lambda_2 - \lambda_2 b} = \frac{c}{1 - a\lambda_1}.$$

From the above equation we get that

$$\lambda_2 = \frac{a(b-d) \cdot \lambda_1 + b(c-1) + d}{a(b-1) \cdot \lambda_1 + (b-1)(c-1)}$$

where the coefficients $a(b-d)$, $b(c-1)+d$, $a(b-1)$, and $(b-1)(c-1)$ are rational and have bit-length $O(L)$. □

Suppose that $v_1v_2, \ldots, v_kv_{k+1}$ is a sequence of $k$ supporting line segments, with corresponding ray functions $f_1, \ldots, f_k$. Then $v_1, \ldots, v_k$ are the vertices of a minimal supporting polygon if and only if $(f_k \circ \ldots \circ f_1)(\lambda) \geq \lambda$, where $\lambda$ is the convex representation of $v_1$.

It follows from [Aggarwal et al., 1989] that, for each edge of $\mathcal{P}$, one can compute in polynomial time a partition $\mathcal{I}$ of $[0, 1]$ into intervals with rational endpoints such that if $\lambda_1, \lambda_2$ are in the same interval $I \in \mathcal{I}$ then the points with convex representation $\lambda_1$ and $\lambda_2$ are associated with the same sequence of ray functions $f_1, \ldots, f_k$. Using Lemma 57 we can, for each interval $I \in \mathcal{I}$, compute these ray functions in polynomial time. Furthermore, on each interval $I \in \mathcal{I}$, we can define the *slack function* $s : I \to \mathbb{R}$ such that for every $\lambda \in I$:

$$s(\lambda) = (f_k \circ \ldots \circ f_1)(\lambda) - \lambda. \tag{6.4}$$

By Lemma 58 in §6.3.1, the slack function has the form $s(\lambda) = \frac{a\lambda+b}{c\lambda+d} - \lambda$ for rational numbers $a, b, c, d$ that are computable in polynomial time. Then it is straightforward to check whether $s(\lambda) \geq 0$ has a solution $\lambda \in I$.

Next we show that if such a solution exists, then there exists a rational solution,

which, moreover, can be computed in polynomial time. To this end, let $\lambda^* \in I$ be such that $s(\lambda^*) \geq 0$. If $\lambda^*$ is on the boundary of $I$, then $\lambda^* \in \mathbb{Q}$. If $\lambda^*$ is not on the boundary and is not an isolated solution, then there exists a rational solution in its neighbourhood. Lastly, let $\lambda^*$ be an isolated solution not on the boundary. Then, $\lambda^*$ is a root of both $s$ and its derivative $s'$. For every $\lambda \in I$, we have

$$(c\lambda + d) \cdot s(\lambda) = a\lambda + b - \lambda \cdot (c\lambda + d).$$

Taking the derivative of the above equation with respect to $\lambda$, we get

$$c \cdot s(\lambda) + (c\lambda + d) \cdot s'(\lambda) = a - d - 2c\lambda. \tag{6.5}$$

Since $s(\lambda^*) = s'(\lambda^*) = 0$, from (6.5) we get $0 = a - d - 2c\lambda^*$. Note that $c \neq 0$ since otherwise $s \equiv 0$. Therefore, $\lambda^* = \frac{a-d}{2c} \in \mathbb{Q}$.

It follows that the vertex $v$ represented by $\lambda^*$ has rational coordinates computable in polynomial time. By computing $(f_i \circ \ldots \circ f_1)(\lambda^*)$ for $i \in [k]$, we can compute in polynomial time the convex representation of all vertices of the supporting polygon $\mathcal{S}_v$. Observe, in particular, that all vertices are rational. Hence we have proved Theorem 56.

### 6.3.1 Rational Linear Fractional Transformations

We showed that the ray function $f$ from Lemma 57 has the form $f(\lambda) = \frac{a\lambda+b}{c\lambda+d}$ for some coefficients $a, b, c, d \in \mathbb{Q}$. Such a function is formally called a *rational linear fractional transformation.*

A composition of two rational linear fractional transformations is, clearly, again a rational linear fractional transformation. In the context of our ray functions $f_1, \ldots, f_k$ above, we have the following result:

**Lemma 58.** *Given $f_i(\lambda) = \frac{a_i\lambda+b_i}{c_i\lambda+d_i}$ in terms of $a_i, b_i, c_i, d_i \in \mathbb{Q}$, for all $i \in [k]$, one can compute in polynomial time $a, b, c, d \in \mathbb{Q}$ such that $(f_k \circ \ldots \circ f_2 \circ f_1)(\lambda) = \frac{a\lambda+b}{c\lambda+d}$.*

*Proof.* For every $\lambda$ we have

$$\begin{aligned}(f_2 \circ f_1)(\lambda) &= \frac{a_2\frac{a_1\lambda+b_1}{c_1\lambda+d_1} + b_2}{c_2\frac{a_1\lambda+b_1}{c_1\lambda+d_1} + d_2} \\ &= \frac{a_2(a_1\lambda+b_1) + b_2(c_1\lambda+d_1)}{c_2(a_1\lambda+b_1) + d_2(c_1\lambda+d_1)} = \frac{(a_2a_1+b_2c_1)\lambda + (a_2b_1+b_2d_1)}{(c_2a_1+d_2c_1)\lambda + (c_2b_1+d_2d_1)}.\end{aligned}$$

From here it follows inductively that $(f_k \circ \ldots \circ f_2 \circ f_1)(\lambda) = \frac{a\lambda+b}{c\lambda+d}$, where the coefficients $a, b, c, d \in \mathbb{Q}$ are computable by iterative matrix multiplication:

$$\begin{pmatrix} a & b \\ c & d \end{pmatrix} = \begin{pmatrix} a_k & b_k \\ c_k & d_k \end{pmatrix} \cdot \ldots \cdot \begin{pmatrix} a_2 & b_2 \\ c_2 & d_2 \end{pmatrix} \cdot \begin{pmatrix} a_1 & b_1 \\ c_1 & d_1 \end{pmatrix}.$$

The result follows. □

## 6.4 Restricted NMF Requires Irrationality

We show that the restricted nonnegative ranks over $\mathbb{R}$ and $\mathbb{Q}$ are, in general, different.

**Theorem 59.** *Let*

$$M = \begin{pmatrix} \frac{5}{44} & \frac{5}{11} & \frac{85}{121} & 0 & 0 & 0 \\ 0 & 0 & 0 & \frac{2}{11} & \frac{3}{11} & \frac{7}{33} \\ \frac{1}{11} & \frac{1}{44} & \frac{2}{121} & \frac{1}{44} & \frac{15}{88} & \frac{17}{88} \\ \frac{1}{44} & \frac{1}{44} & \frac{8}{121} & \frac{1}{44} & \frac{19}{88} & \frac{5}{24} \\ \frac{3}{11} & \frac{3}{11} & \frac{12}{121} & \frac{8}{11} & \frac{2}{11} & \frac{2}{33} \\ \frac{1}{2} & \frac{5}{22} & \frac{14}{121} & \frac{1}{22} & \frac{7}{44} & \frac{43}{132} \end{pmatrix} \in \mathbb{Q}_+^{6\times 6}.$$

*Then the restricted nonnegative rank of $M$ over $\mathbb{R}$ is 5, whereas the restricted non-*

*negative rank of $M$ over $\mathbb{Q}$ is* 6.

The rest of this section is devoted to the proof of Theorem 59. Matrix $M$ has a (stochastic) 5-dimensional NMF

$$M = W \cdot H \tag{6.6}$$

with $W$ and $H$ as follows:

$$W = \begin{pmatrix} 0 & \frac{5}{7}+\frac{5\sqrt{2}}{77} & \frac{15+5\sqrt{2}}{77} & 0 & 0 \\ 0 & 0 & 0 & \frac{20+2\sqrt{2}}{77} & \frac{48-8\sqrt{2}}{187} \\ \frac{\sqrt{2}}{11} & 0 & \frac{4-\sqrt{2}}{77} & \frac{3}{14}+\frac{\sqrt{2}}{308} & \frac{14-8\sqrt{2}}{187} \\ \frac{-1+\sqrt{2}}{11} & \frac{4+\sqrt{2}}{77} & 0 & \frac{39}{154}+\frac{5\sqrt{2}}{308} & \frac{21-12\sqrt{2}}{187} \\ \frac{8-4\sqrt{2}}{11} & \frac{12-4\sqrt{2}}{77} & \frac{4}{11} & 0 & \frac{104+28\sqrt{2}}{187} \\ \frac{4+2\sqrt{2}}{11} & \frac{6-2\sqrt{2}}{77} & \frac{30-4\sqrt{2}}{77} & \frac{3}{11}-\frac{\sqrt{2}}{22} & 0 \end{pmatrix}, \tag{6.7}$$

$$H = \begin{pmatrix} \frac{1+\sqrt{2}}{4} & 0 & \frac{\sqrt{2}}{11} & \frac{1}{4}-\frac{\sqrt{2}}{8} & 0 & \frac{1}{6}+\frac{\sqrt{2}}{12} \\ 0 & \frac{1}{2}-\frac{\sqrt{2}}{8} & 1-\frac{\sqrt{2}}{11} & 0 & 0 & 0 \\ \frac{3-\sqrt{2}}{4} & \frac{1}{2}+\frac{\sqrt{2}}{8} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \frac{21}{34}+\frac{7\sqrt{2}}{68} & \frac{5}{6}-\frac{\sqrt{2}}{12} \\ 0 & 0 & 0 & \frac{3}{4}+\frac{\sqrt{2}}{8} & \frac{13}{34}-\frac{7\sqrt{2}}{68} & 0 \end{pmatrix}.$$

Since $\operatorname{rank}(M) = \operatorname{rank}(W) = 4$, the NMF $M = W \cdot H$ is restricted. This implies that the restricted nonnegative rank of $M$ over $\mathbb{R}$ is at most 5. Crucially the entries of the factors $W$, $H$ are irrational, and the core of the proof of Theorem 59 consists in showing that $M$ has no 5-dimensional RNMF over $\mathbb{Q}$.

To this end, in the following Section 6.4.1 we classify all at most 5-dimensional

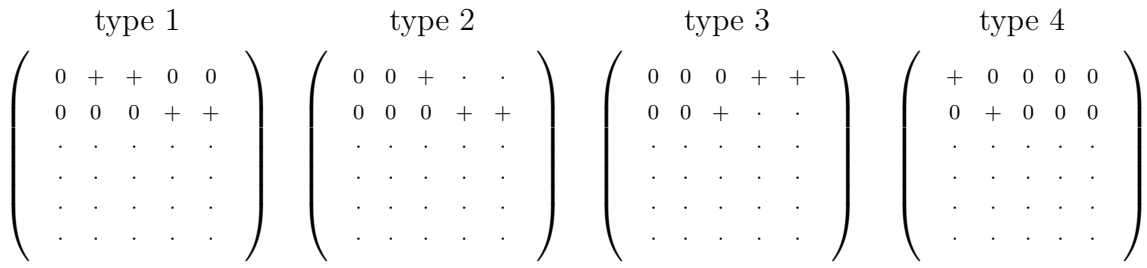



Figure 6.4: Let $M$ be the matrix from Theorem 59. In any 5-dimensional NMF $M = L \cdot R$, matrix $L$ matches one of the four patterns above, up to a permutation of its columns. Here + denotes any strictly positive number.

NMFs of $M$ into four types (illustrated in Figure 6.4) according to the "zero pattern" in the first two rows of the left factor. We then give separate arguments excluding rational RNMFs of $M$ of each of the four types, thereby showing that the restricted nonnegative rank of $M$ over $\mathbb{Q}$ is strictly greater than 5. In fact, we show that factorization (6.6) is the unique at most 5-dimensional RNMF of $M$, thus proving that the restricted nonnegative rank of $M$ over $\mathbb{R}$ is exactly 5.

### 6.4.1 Types of Factorizations

Let $M$ be the matrix from Theorem 59. In Proposition 60 we classify all (stochastic) at most 5-dimensional NMFs of $M$ into four types according to the "zero pattern" in the first two rows of the left factor.

For any stochastic and at most 5-dimensional NMF $L \cdot R$, we introduce the following notation:

- $k$ is the number of columns in $L$ whose first and second coordinates are 0,
- $k_1$ is the number of columns in $L$ whose first coordinate is strictly positive and second coordinate is 0, and
- $k_2$ is the number of columns in $L$ whose second coordinate is strictly positive and first coordinate is 0.

**Proposition 60.** *Let $M$ be the matrix from Theorem 59. Any stochastic NMF $M = L \cdot R$ of inner dimension at most 5 has one of the following four* types*:*

1) $k = 1$, $k_1 = 2$, $k_2 = 2$;

2) $k = 2$, $k_1 = 1$;

3) $k = 2$, $k_2 = 1$;

4) $k = 3$, $k_1 = 1$, $k_2 = 1$.

*Proof.* The NMF $M = L \cdot R$ corresponds to representing each column of $M$ as a convex combination of the columns of $L$, with the coefficients of the convex combination specified by the entries of the right factor $R$. As $L$ has at most 5 columns,

$$k + k_1 + k_2 \leq 5. \tag{6.8}$$

Since the columns $M_{:,1}, M_{:,2}, M_{:,3}$ are affinely independent, matrix $L$ must have at least three columns whose second coordinate is 0. Likewise, since the columns $M_{:,4}, M_{:,5}, M_{:,6}$ are affinely independent, $L$ must have at least three columns whose first coordinate is 0. That is,

$$k + k_1 \geq 3 \quad \text{and} \quad k + k_2 \geq 3. \tag{6.9}$$

Together with (6.8), this implies that $2k \geq 6 - k_1 - k_2 \geq k + 1$ and therefore $k \geq 1$.

The columns $M_{:,1}, M_{:,2}, M_{:,3}$ have first coordinate strictly positive and second coordinate 0, while the columns $M_{:,4}, M_{:,5}, M_{:,6}$ have second coordinate strictly positive and first coordinate 0. Therefore, in order for these columns to be covered by the columns of $L$, we need to have:

$$k_1 \geq 1 \quad \text{and} \quad k_2 \geq 1. \tag{6.10}$$

Together with (6.8), this implies that $k \leq 5 - k_1 - k_2 \leq 3$. We conclude that $k \in \{1, 2, 3\}$. The result now follows from the inequalities (6.8), (6.9), and (6.10). $\square$

The four types of NMFs $M = L \cdot R$ defined in Proposition 60 are illustrated in Figure 6.4 for inner dimension 5. In Section 6.5, we consider each type separately and prove the following proposition:

**Proposition 61.** *Let $M$ be the matrix from Theorem 59 and $W$ the matrix in* (6.7)*.*

(1) *If $M = L \cdot R$ is a type-1 NMF then $L$ is equal to $W$, up to a permutation of its columns. In particular, $L$ is not rational and the NMF $M = L \cdot R$ is restricted.*

(2) *Matrix $M$ has no type-2 NMF.*

(3) *Matrix $M$ has no type-3 NMF.*

(4) *Matrix $M$ has no restricted type-4 NMF.*

Using this proposition we can prove Theorem 59:

*Proof of Theorem 59.* Due to the restricted NMF of $M$ stated in (6.6), the restricted nonnegative rank of $M$ is at most 5. Since $\operatorname{rank}(M) = 4$, the restricted nonnegative rank of $M$ is either 4 or 5. If there existed a 4-dimensional RNMF of $M$ then, as $k + k_1 + k_2 \leq 4$, it would have to have type 2 or 3, but NMFs of those types are excluded by items (2) and (3) of Proposition 61. Hence the restricted nonnegative rank of $M$ equals 5.

Since $M = M \cdot I_6$, the restricted nonnegative rank of $M$ over $\mathbb{Q}$ is at most 6. By Proposition 61, there is no 5-dimensional restricted NMF $M = L \cdot R$ with $L$ rational. Hence, the restricted nonnegative rank of $M$ over $\mathbb{Q}$ equals 6. $\square$

## 6.5 Ruling out Factorizations

In this section, we prove Proposition 61. Let $M$ be the matrix from Theorem 59. To rule out rational (restricted) NMFs of $M$ of each type, we employ geometric arguments concerning nested polygons in the plane (see Section 5.5.2). To set up this geometric connection, first recall from Section 6.2 that the RNMF problem has an interpretation in terms of the nested polytope problem (NPP). More specifically, the RNMF $M = W \cdot H$ from (6.6) can be obtained by a transformation, according to Proposition 54, from a 3-dimensional NPP instance, which we now describe.

Matrix $M$ is stochastic and has rank 4, and hence the columns of $M$ affinely span a 3-dimensional affine subspace $\mathcal{V} \subseteq \mathbb{R}^6$. All vectors in $\mathcal{V}$ are pseudo-stochastic. The properly stochastic (i.e., nonnegative) vectors in $\mathcal{V}$ form a 3-dimensional polytope, say $\mathcal{P}' \subseteq \mathcal{V}$; in other words, $\mathcal{P}'$ is defined as the intersection of $\mathcal{V}$ with $\mathbb{R}^6_+$. Observe that $\text{conv}\{M_{:,1}, \ldots, M_{:,6}\} \subseteq \mathcal{P}'$. For convenience in the proofs, we fix a specific parameterization of $\mathcal{V}$ and $\mathcal{P}'$ in the next subsection.

### Parameterization

Define a function $f : \mathbb{R}^3 \to \mathbb{R}^6$ such that, for each $x \in \mathbb{R}^3$, $f(x) = Cx + d$ where:

$$C = \frac{1}{11} \cdot \begin{pmatrix} 0 & 10 & 0 \\ 0 & 0 & 4 \\ -1 & -2 & \frac{1}{2} \\ -1 & 0 & \frac{5}{2} \\ 4 & 0 & 0 \\ -2 & -8 & -7 \end{pmatrix} \in \mathbb{Q}^{6\times 3} \quad \text{and} \quad d = \frac{1}{11} \cdot \begin{pmatrix} 0 \\ 0 \\ 2 \\ 1 \\ 0 \\ 8 \end{pmatrix} \in \mathbb{Q}^6.$$

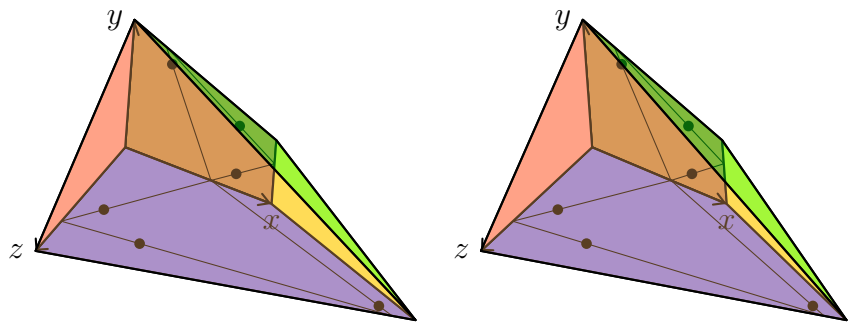



Figure 6.5: Nested polytope problem instance: The two images show orthogonal projections of the outer polytope $\mathcal{P} \subseteq \mathbb{R}^3$. The black dots indicate 6 inner points that span the interior polytope $\mathcal{R}$: 3 points ($r_1$, $r_2$, $r_3$) on the brown $xy$-face, and 3 points ($r_4$, $r_5$, $r_6$) on the blue $xz$-face. The vertices of the two triangles on the $xy$-face and on the $xz$-face indicate the (unique) location of the 5 points that span a nested polytope $\mathcal{Q}$. The two slightly different projections are designed to create a 3-dimensional impression using stereoscopy. The "parallel-eye" technique should be used, see, e.g., [Simanek].

The map $f$ is clearly injective. Let $\mathcal{P} \subseteq \mathbb{R}^3$ be the 3-dimensional polytope defined by $\{\, x \in \mathbb{R}^3 \mid f(x) \geq \mathbf{0} \,\}$. We have

$$f(\mathbb{R}^3) = \mathcal{V} \qquad \text{and} \qquad f(\mathcal{P}) = \mathcal{P}'. \tag{6.11}$$

Indeed, defining

$$r_1 = \begin{pmatrix} \frac{3}{4} \\ \frac{1}{8} \\ 0 \end{pmatrix},\ r_2 = \begin{pmatrix} \frac{3}{4} \\ \frac{1}{2} \\ 0 \end{pmatrix},\ r_3 = \begin{pmatrix} \frac{3}{11} \\ \frac{17}{22} \\ 0 \end{pmatrix},\ r_4 = \begin{pmatrix} 2 \\ 0 \\ \frac{1}{2} \end{pmatrix},\ r_5 = \begin{pmatrix} \frac{1}{2} \\ 0 \\ \frac{3}{4} \end{pmatrix},\ r_6 = \begin{pmatrix} \frac{1}{6} \\ 0 \\ \frac{7}{12} \end{pmatrix},$$

we have $f(r_i) = M_{:,i}$ for each $i \in [6]$.

Figure 6.5 visualises $\mathcal{P}$, which has 6 faces corresponding to the inequalities of the system $Cx + d \geq \mathbf{0}$. In more detail, $\mathcal{P}$ is the intersection of the following half-spaces: $y \geq 0$ (blue), $z \geq 0$ (brown), $-\frac{1}{2}x - y + \frac{1}{4}z + 1 \geq 0$ (green), $-x + \frac{5}{2}z + 1 \geq 0$ (yellow), $x \geq 0$ (pink), $-\frac{1}{4}x - y - \frac{7}{8}z + 1 \geq 0$ (transparent front). The figure also shows

the position of the points $r_1, \ldots, r_6$ (black dots) which span an interior polytope $\mathcal{R}$ . Observe that $r_i \in \mathcal{P}$, as $f(r_i) = M_{:,i} \in \mathcal{P}'$ for all $i \in [6]$.

The columns of $W$ in (6.7) are also in $\mathcal{P}' \subseteq \mathcal{V}$. Indeed, defining

$$q_1^* = \begin{pmatrix} 2-\sqrt{2} \\ 0 \\ 0 \end{pmatrix},\ q_2^* = \begin{pmatrix} \frac{3-\sqrt{2}}{7} \\ \frac{11+\sqrt{2}}{14} \\ 0 \end{pmatrix},\ q_3^* = \begin{pmatrix} 1 \\ \frac{3+\sqrt{2}}{14} \\ 0 \end{pmatrix},\ q_4^* = \begin{pmatrix} 0 \\ 0 \\ \frac{10+\sqrt{2}}{14} \end{pmatrix},\ q_5^* = \begin{pmatrix} \frac{26+7\sqrt{2}}{17} \\ 0 \\ \frac{12-2\sqrt{2}}{17} \end{pmatrix},$$

we have $f(q_i^*) = W_{:,i} \in \mathcal{P}'$, hence $q_i^* \in \mathcal{P}$ for each $i \in [5]$. Applying the inverse of the map $f$ column-wise to the NMF $M = W \cdot H$ of (6.6), we obtain

$$\begin{pmatrix} r_1 & r_2 & r_3 & r_4 & r_5 & r_6 \end{pmatrix} = \begin{pmatrix} q_1^* & q_2^* & q_3^* & q_4^* & q_5^* \end{pmatrix} \cdot H. \tag{6.12}$$

Since matrix $H$ is stochastic, (6.12) implies that the points $r_i$ are contained in the convex hull of the points $q_i^*$. In Figure 6.5, points $q_1^*, q_2^*, q_3^*$ are the vertices of the triangle on the brown $xy$-face, while points $q_1^*, q_4^*, q_5^*$ are the vertices of the triangle on the blue $xz$-face. The former triangle contains the points $r_1, r_2, r_3$, while the latter triangle contains the points $r_4, r_5, r_6$.

### 6.5.1 Type 1

In this subsection we prove Proposition 61 (1), implying that any type-1 NMF of $M$ (restricted or otherwise) requires irrational entries.

We will need the following elementary fact of linear algebra.

**Proposition 62.** *Let $q_1 = (x_1, y_1)$, $q_2 = (x_2, y_2)$, and $q_3 = (x_3, y_3)$ be three distinct*

*points in the plane, and consider the determinant*

$$\Delta = \begin{vmatrix} x_1 & y_1 & 1 \\ x_2 & y_2 & 1 \\ x_3 & y_3 & 1 \end{vmatrix}.$$

*Then* $\Delta = 0$ *if and only if* $q_1$, $q_2$, *and* $q_3$ *are collinear, and* $\Delta > 0$ *if and only if the list of vertices* $q_1, q_2, q_3$ *describes a triangle with anti-clockwise orientation.*

Let us consider a type-1 NMF $M = L \cdot R$. By definition, $k = 1$ and $k_1 = k_2 = 2$. After a suitable permutation of its columns, $L$ matches the pattern

$$L = \begin{pmatrix} 0 & + & + & 0 & 0 \\ 0 & 0 & 0 & + & + \\ \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot \end{pmatrix},$$

where + denotes any strictly positive number. Considering the respective zero patterns of $M$ and $L$ it is clear that: (i) $M_{:,1}$, $M_{:,2}$, $M_{:,3}$ all lie in the convex hull of $L_{:,1}, L_{:,2}, L_{:,3}$, and (ii) $M_{:,4}, M_{:,5}, M_{:,6}$ all lie in the convex hull of $L_{:,1}, L_{:,4}, L_{:,5}$. At a high level, our proof strategy is to use facts (i) and (ii) to respectively derive two inequalities which together force $L = W$, up to a permutation of the columns, for $W$ the matrix in (6.7).

Let us write $\mathcal{V}_0 \subseteq \mathbb{R}^6$ for the affine span of $M_{:,1}, M_{:,2}, M_{:,3}$. We can also characterise $\mathcal{V}_0$ as the image of the $xy$-plane in $\mathbb{R}^3$ under the map $f : \mathbb{R}^3 \to \mathbb{R}^6$. Indeed, we have $f(r_1) = M_{:,1}$, $f(r_2) = M_{:,2}$, and $f(r_3) = M_{:,3}$. Thus, the image of the $xy$-plane under $f$ is a 2-dimensional affine space that includes $\mathcal{V}_0$ and hence is equal to $\mathcal{V}_0$. Define the polygon $\mathcal{P}_0 \subseteq \mathbb{R}^3$ by $\mathcal{P}_0 = \{(x, y, 0)^\top : (x, y, 0)^\top \in \mathcal{P}\}$. Figure 6.6 illustrates

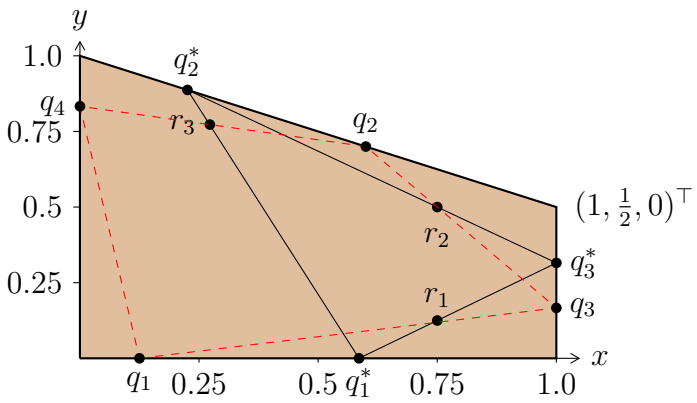



Figure 6.6: The outer polygon is $\mathcal{P}_0$ (after identifying the $xy$-plane in $\mathbb{R}^3$ with $\mathbb{R}^2$). Likewise, the inner polygon is $\triangle r_1 r_2 r_3$. The quadrilateral with dashed boundary is the supporting polygon $\mathcal{S}_{q_1}$, where $q_1 = (\frac{1}{8}, 0, 0)^\top$. The triangle $\triangle q_1^* q_3^* q_2^*$, with solid boundary, is the supporting polygon $\mathcal{S}_{q_1^*}$, where $q_1^* = (2 - \sqrt{2}, 0, 0)^\top$.

polygon $\mathcal{P}_0$. Then, $f$ restricts to a bijection between $\mathcal{P}_0$ and the set of nonnegative vectors in $\mathcal{V}_0$. We have the following lemma:

**Lemma 63.** *Let $\mathcal{R} \subseteq \mathcal{P}_0$ be the triangle with vertices $r_1, r_2, r_3$ (see Figure 6.6). If $q_1 = (u, 0, 0)^\top$, where $0 \leq u < 2 - \sqrt{2}$, then the supporting polygon $\mathcal{S}_{q_1}$ has more than three vertices.*

*Proof.* Towards a contradiction, suppose that $\mathcal{S}_{q_1}$ has three vertices. Moving anticlockwise, let the vertices of $\mathcal{S}_{q_1}$ be $q_1$, $q_3$, and $q_2$. From the assumption that $0 \leq u \leq 2 - \sqrt{2}$ it follows by elementary geometry that: (i) the line segment $q_1 q_3$ passes through $r_1$ and $q_3$ lies on the right edge of $\mathcal{P}_0$, and (ii) the line segment $q_3 q_2$ passes through $r_2$ and $q_2$ lies on the upper edge of $\mathcal{P}_0$. Figure 6.6 shows the supporting polygons in the cases $u = \frac{1}{8}$ and $u = 2 - \sqrt{2}$.

Writing $q_3 = (1, \frac{v}{2}, 0)^\top$ and $q_2 = (1 - w, \frac{1}{2} + \frac{w}{2}, 0)^\top$, where $0 \leq v, w \leq 1$, the

collinearity conditions (i), (ii) and Proposition 62 entail:

$$\begin{vmatrix} u & 0 & 1 \\ 1 & \frac{v}{2} & 1 \\ \frac{3}{4} & \frac{1}{8} & 1 \end{vmatrix} = \frac{1}{2}uv - \frac{1}{8}u - \frac{3}{8}v + \frac{1}{8} \qquad = 0 \qquad \text{and} \tag{6.13}$$

$$\begin{vmatrix} 1 & \frac{v}{2} & 1 \\ 1-w & \frac{1}{2}+\frac{w}{2} & 1 \\ \frac{3}{4} & \frac{1}{2} & 1 \end{vmatrix} = \frac{1}{2}vw - \frac{1}{8}v - \frac{3}{8}w + \frac{1}{8} \qquad = 0\,. \tag{6.14}$$

The assumption that $\mathcal{S}_{q_1}$ is the triangle with vertices $q_1, q_3, q_2$ entails that vertices $q_2, q_1, r_3$ are in anti-clockwise order (unlike in Figure 6.6). This implies:

$$\begin{vmatrix} 1-w & \frac{1}{2}+\frac{w}{2} & 1 \\ u & 0 & 1 \\ \frac{3}{11} & \frac{17}{22} & 1 \end{vmatrix} = -\frac{1}{2}wu + \frac{10}{11}w + \frac{3}{11}u - \frac{7}{11} \geq 0. \tag{6.15}$$

We use the equations in (6.13) and (6.14) to eliminate variables $v, w$ from the inequality (6.15), obtaining:

$$\frac{15}{22(8u-5)} \cdot (u^2 - 4u + 2) \qquad \geq 0.$$

This inequality has no solution $0 \leq u < 2-\sqrt{2}$, yielding the desired contradiction. □

Let us write $\mathcal{V}_1 \subseteq \mathbb{R}^6$ for the affine span of $M_{:,4}, M_{:,5}, M_{:,6}$. We can also characterise $\mathcal{V}_1$ as the image of the $xz$-plane in $\mathbb{R}^3$ under the map $f : \mathbb{R}^3 \to \mathbb{R}^6$. Indeed, we have $f(r_4) = M_{:,4}$, $f(r_5) = M_{:,5}$, and $f(r_6) = M_{:,6}$. Thus, the image of the $xz$-plane under $f$ is a 2-dimensional affine space that includes $\mathcal{V}_1$ and hence is equal to $\mathcal{V}_1$. Define the polygon $\mathcal{P}_1 \subseteq \mathbb{R}^3$ by $\mathcal{P}_1 = \{(x, 0, z)^\top : (x, 0, z)^\top \in \mathcal{P}\}$; see Figure 6.7 for an illustration. Then, $f$ restricts to a bijection between $\mathcal{P}_1$ and the set of nonnegative

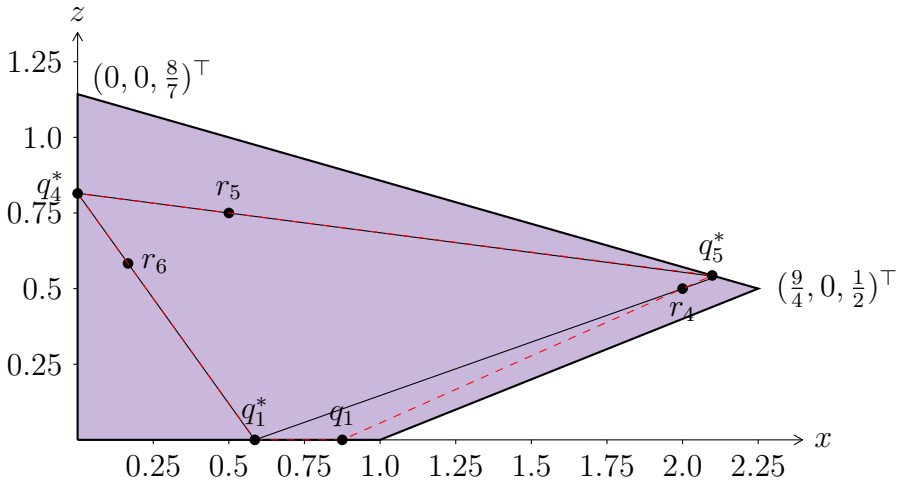



Figure 6.7: The outer polygon is $\mathcal{P}_1$ (after identifying the $xz$-plane in $\mathbb{R}^3$ with $\mathbb{R}^2$). The inner polygon is $\triangle r_4 r_5 r_6$. Writing $q_1 = (\frac{7}{8}, 0, 0)^\top$, the quadrilateral with dashed red boundary is the supporting polygon $\mathcal{S}_{q_1}$. The triangle $\triangle q_1^* q_5^* q_4^*$, with solid black boundary, is the supporting polygon $\mathcal{S}_{q_1^*}$, where $q_1^* = (2 - \sqrt{2}, 0, 0)^\top$.

vectors in $\mathcal{V}_1$. We have the following lemma:

**Lemma 64.** *Let $\mathcal{R} \subseteq \mathcal{P}_1$ be the triangle with vertices $r_4, r_5, r_6$ (see Figure 6.7). If $q_1 = (u, 0, 0)^\top$, where $2 - \sqrt{2} < u \leq 1$, then the supporting polygon $\mathcal{S}_{q_1}$ has more than three vertices.*

*Proof.* Towards a contradiction, suppose that $\mathcal{S}_{q_1}$ has three vertices. Moving anti-clockwise, let the vertices of $\mathcal{S}_{q_1}$ be $q_1$, $q_5$, and $q_4$. From the assumption that $2 - \sqrt{2} \leq u \leq 1$ it follows by elementary geometry that: (i) the line segment $q_1 q_5$ passes through $r_4$ and $q_5$ lies on the upper edge of $\mathcal{P}_1$, and (ii) the line segment $q_5 q_4$ passes through $r_5$ and $q_4$ lies on the left edge of $\mathcal{P}_1$. Figure 6.7 shows the supporting polygons in the cases $u = \frac{7}{8}$ and $u = 2 - \sqrt{2}$.

Writing $q_5 = (\frac{9-9v}{4}, 0, \frac{7+9v}{14})^\top$ and $q_4 = (0, 0, \frac{8-8w}{7})^\top$, where $0 \leq v, w \leq 1$, the

collinearity conditions (i), (ii) and Proposition 62 entail:

$$\begin{vmatrix} u & 0 & 1 \\ \frac{9-9v}{4} & \frac{7+9v}{14} & 1 \\ 2 & \frac{1}{2} & 1 \end{vmatrix} = \frac{9}{14}uv - \frac{135}{56}v + \frac{1}{8} = 0 \quad \text{and} \tag{6.16}$$

$$\begin{vmatrix} \frac{9-9v}{4} & \frac{7+9v}{14} & 1 \\ 0 & \frac{8-8w}{7} & 1 \\ \frac{1}{2} & \frac{3}{4} & 1 \end{vmatrix} = \frac{18}{7}vw - \frac{9}{16}v - 2w + \frac{9}{16} = 0 \tag{6.17}$$

The assumption that $\mathcal{S}_{q_1}$ is the triangle with vertices $q_1, q_5, q_4$ entails that vertices $q_4, q_1, r_6$ are in anti-clockwise order. This implies:

$$\begin{vmatrix} 0 & \frac{8-8w}{7} & 1 \\ u & 0 & 1 \\ \frac{1}{6} & \frac{7}{12} & 1 \end{vmatrix} = \frac{8}{7}wu - \frac{4}{21}w - \frac{47}{84}u + \frac{4}{21} \geq 0 \tag{6.18}$$

We use the equations in (6.16) and (6.17) to eliminate variables $v, w$ from the inequality (6.18), obtaining:

$$\frac{-10}{21(2u-7)} \cdot (u^2 - 4u + 2) \geq 0$$

This inequality has no solution $2-\sqrt{2} < u \leq 1$, yielding the desired contradiction. □

Figure 6.8 provides a combined view of the $xy$-plane and the $xz$-plane.

The affine span of column vectors $L_{:,1}, L_{:,2}, L_{:,3}$ includes $\mathcal{V}_0$ and has dimension at most two, and hence is equal to $\mathcal{V}_0$. In particular, $L_{:,1}, L_{:,2}, L_{:,3}$ must all lie in $\mathcal{V}_0$. Since $L_{:,1}, L_{:,2}, L_{:,3}$ are moreover nonnegative, there are uniquely defined points $q_1, q_2, q_3 \in \mathcal{P}_0$ such that $f(q_i) = L_{:,i}$ for $i \in \{1, 2, 3\}$. Since NMF $M = L \cdot R$ is stochastic, the convex hull of points $q_1, q_2, q_3$ includes points $r_1, r_2, r_3$, i.e., $\triangle q_1 q_3 q_2$ is nested

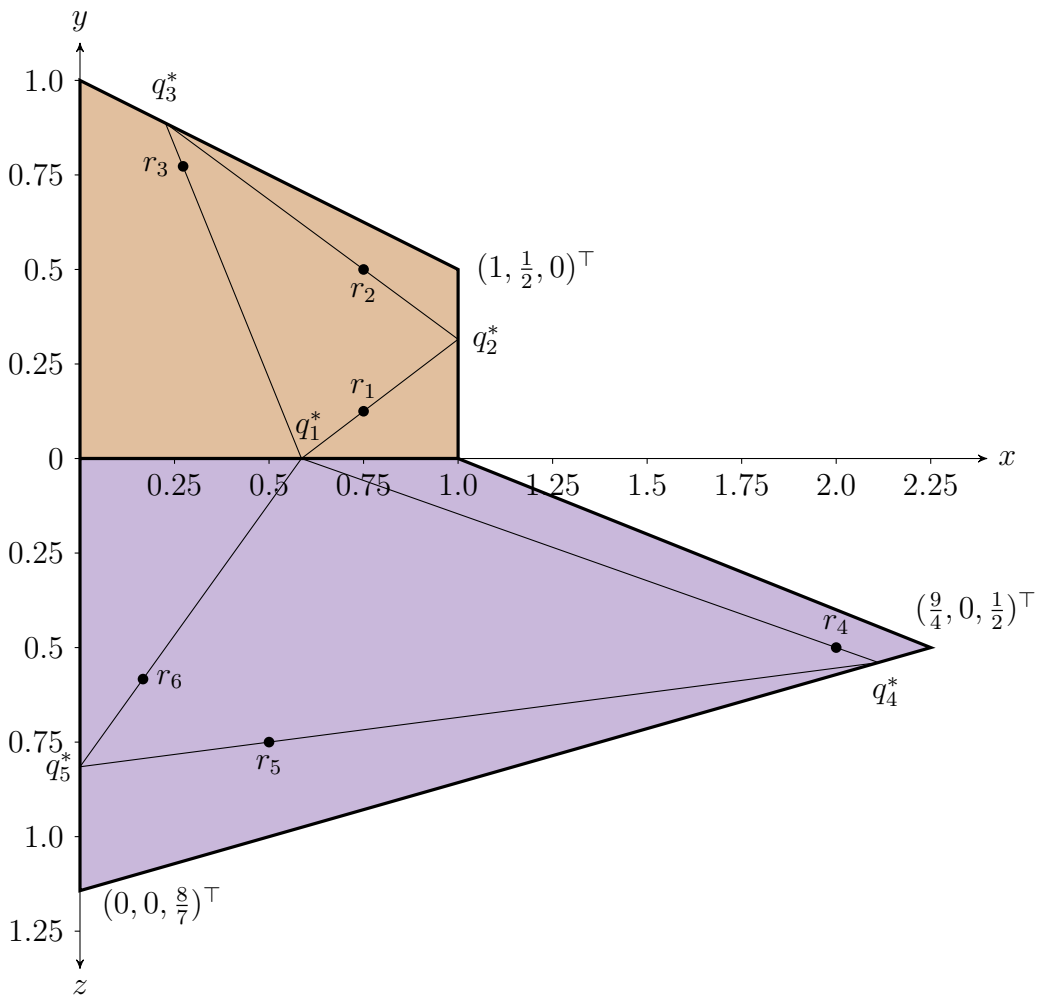



Figure 6.8: Combined view of the $xy$-plane and the $xz$-plane. One may imagine that the figure is folded along the $x$-axis.

between $\triangle r_1 r_2 r_3$ and the polygon $\mathcal{P}_0$. Since $L_{:,1}$ has 0 in its first two coordinates, by inspecting the definition of the map $f$ we see that $q_1 = (u, 0, 0)^\top$ for some $u \in [0, 1]$. By Lemma 53 it follows that the supporting polygon $\mathcal{S}_{q_1}$ has three vertices. Now, Lemma 63 implies that $2 - \sqrt{2} \leq u \leq 1$.

Considering the polygon $\mathcal{P}_1$, we have $q_1 \in \mathcal{P}_1$ (recall that $f(q_1) = L_{:,1}$). Arguing as in the case of $\mathcal{P}_0$, there are uniquely defined points $q_4, q_5 \in \mathcal{P}_1$ such that $f(q_i) = L_{:,i}$ for $i \in \{4, 5\}$. Similarly as before, $\triangle q_1 q_5 q_4$ is nested between $\triangle r_4 r_5 r_6$ and the polygon $\mathcal{P}_1$. Then Lemmas 53 and 64 imply that $0 \leq u \leq 2 - \sqrt{2}$, thus $u = 2 - \sqrt{2}$. This means that $q_1 = (2 - \sqrt{2}, 0, 0)^\top = q_1^*$. Hence $L_{:,1} = f(q_1) = f(q_1^*) = W_{:,1}$.

We recall from Figure 6.6 that the supporting polygon $\mathcal{S}_{q_1^*}$, nested between the triangle $\triangle r_1 r_2 r_3$ and the polygon $\mathcal{P}_0$, corresponds to the triangle $\triangle q_1^* q_3^* q_2^*$. Similarly, as seen in Figure 6.7, the supporting polygon $\mathcal{S}_{q_1^*}$, nested between the triangle $\triangle r_4 r_5 r_6$ and the polygon $\mathcal{P}_1$, is the triangle $\triangle q_1^* q_5^* q_4^*$. We have already shown that $q_1 = q_1^*$. In the following, we show that $q_i = q_i^*$ for each $i \in \{2, 3, 4, 5\}$.

Towards a contradiction, suppose that $q_2 \neq q_2^*$ or $q_3 \neq q_3^*$. Let us consider the case when $q_2 \neq q_2^*$. Observe that triangles $\triangle q_1^* q_3^* q_2^*$ and $\triangle q_1^* q_3 q_2$ are both nested between $\triangle r_1 r_2 r_3$ and $\mathcal{P}_0$. The fact that $\triangle r_1 r_2 r_3 \subseteq \triangle q_1^* q_3 q_2$ implies that vertices $q_3$ and $q_2$ lie to the right of (or on) directed line segments $q_1^* q_3^*$ and $q_2^* q_1^*$, respectively. Since, moreover, $q_3, q_2 \in \mathcal{P}_0$, it holds that vertex $q_3$ lies to the left of (or on) directed line segment $q_3^* q_2^*$, whereas vertex $q_2$ lies strictly to the left of $q_3^* q_2^*$. However, this implies that the point $r_2$ is to the right of directed line segment $q_3 q_2$, which is a contradiction with the assumption that $\triangle r_1 r_2 r_3 \subseteq \triangle q_1^* q_3 q_2$. The case $q_3 \neq q_3^*$ analogously leads to a contradiction. We conclude that $q_2 = q_2^*$ and $q_3 = q_3^*$. Analogously, using Lemma 64 one can show that $q_4 = q_4^*$ and $q_5 = q_5^*$.

Since $f(q_i) = L_{:,i}$ and $f(q_i^*) = W_{:,i}$ for each $i \in \{2, 3, 4, 5\}$, we conclude that $\{L_{:,2}, L_{:,3}\} = \{W_{:,2}, W_{:,3}\}$ and $\{L_{:,4}, L_{:,5}\} = \{W_{:,4}, W_{:,5}\}$. Therefore, the NMF $M = L \cdot R$ coincides with the one given in (6.6), up to a permutation of the columns of $W$

and the rows of $H$. Proposition 61 (1) follows.

### 6.5.2 Type 2

In this subsection we exclude type-2 NMFs of $M$, i.e., we prove Proposition 61 (2). Towards a contradiction, suppose there is a type-2 NMF $M = L \cdot R$. We recall that, by definition, this means that $M = L \cdot R$ is a stochastic and at most 5-dimensional NMF where $k = 2$ and $k_1 = 1$. Up to some permutation of its columns, $L$ matches the pattern

$$L = \begin{pmatrix} 0 & 0 & + & \cdot & \cdot \\ 0 & 0 & 0 & + & + \\ \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot \end{pmatrix} \quad \text{or} \quad L = \begin{pmatrix} 0 & 0 & + & \cdot \\ 0 & 0 & 0 & + \\ \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot \end{pmatrix},$$

according to whether the inner dimension of the NMF $M = L \cdot R$ is 5 or 4, respectively. It follows from the zero pattern in $M$ that, in either case, columns $M_{:,1}, M_{:,2}, M_{:,3}$ all lie in the convex hull of $L_{:,1}, L_{:,2}, L_{:,3}$.

Consider again the affine space $\mathcal{V}_0 \subseteq \mathbb{R}^6$ and the polygon $\mathcal{P}_0 \subseteq \mathbb{R}^3$ from §6.5.1. We recall that $\mathcal{P}_0$ is visualised in Figure 6.6. By reasoning analogously as in §6.5.1, there are uniquely defined points $q_1, q_2, q_3 \in \mathcal{P}_0$ such that $f(q_i) = L_{:,i}$ for $i \in \{1, 2, 3\}$. It follows that the convex hull of $q_1, q_2, q_3$ includes $r_1, r_2, r_3$. Since $L_{:,1}$ and $L_{:,2}$ have 0 in their first two rows, by inspecting the definition of the map $f$ we see that $q_1$ and $q_2$ lie on the $x$-axis in $\mathbb{R}^3$. Let us define points $\hat{q}_1 = (0, 0, 0)^\top$ and $\hat{q}_2 = (1, 0, 0)^\top$. The triangle $\triangle \hat{q}_1 \hat{q}_2 q_3$ contains the triangle $\triangle q_1 q_2 q_3$, hence the convex hull of $\hat{q}_1, \hat{q}_2, q_3$ also includes $r_1, r_2, r_3$. It follows that the vertices $\hat{q}_1, r_3, q_3$ are in anti-clockwise order, and that the vertices $\hat{q}_2, q_3, r_2$ are in anti-clockwise order. By inspecting the location

of these points (see Figure 6.6), one can see that $q_3$ is then outside $\mathcal{P}_0$, which is a contradiction. Thus we have proved Proposition 61 (2).

### 6.5.3 Type 3

In this subsection we exclude type-3 NMFs of $M$, i.e., we prove Proposition 61 (3). Our reasoning is entirely analogous to §6.5.2. Towards a contradiction, suppose there is a type-3 NMF $M = L \cdot R$, i.e., a stochastic and at most 5-dimensional NMF such that $k = 2$ and $k_2 = 1$. Up to some permutation of its columns, $L$ matches the pattern

$$L = \begin{pmatrix} 0 & 0 & 0 & + & + \\ 0 & 0 & + & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot \\ & & & & \\ \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot \end{pmatrix} \quad \text{or} \quad L = \begin{pmatrix} 0 & 0 & 0 & + \\ 0 & 0 & + & \cdot \\ \cdot & \cdot & \cdot & \cdot \\ & & & \\ \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot \end{pmatrix},$$

when the inner dimension of the NMF $M = L \cdot R$ is 5 or 4, respectively. In either case, columns $M_{:,4}, M_{:,5}, M_{:,6}$ all lie in the convex hull of $L_{:,1}, L_{:,2}, L_{:,3}$.

Consider again the affine space $\mathcal{V}_1 \subseteq \mathbb{R}^6$ and the polygon $\mathcal{P}_1 \subseteq \mathbb{R}^3$ from §6.5.1. Recall that $\mathcal{P}_1$ is visualised in Figure 6.7. Analogously to the previous subsection, there are uniquely defined points $q_1, q_2, q_3 \in \mathcal{P}_1$ (such that $f(q_i) = L_{:,i}$ for $i \in \{1, 2, 3\}$) whose convex hull includes $r_4, r_5, r_6$, and $q_1$ and $q_2$ lie on the $x$-axis in $\mathbb{R}^3$. Defining again $\hat{q}_1 = (0, 0, 0)^\top$ and $\hat{q}_2 = (1, 0, 0)^\top$, we obtain that the convex hull of $\hat{q}_1, \hat{q}_2, q_3$ includes $r_4, r_5, r_6$. It follows that the vertices $\hat{q}_1, r_6, q_3$ are in anti-clockwise order, and that the vertices $\hat{q}_2, q_3, r_4$ are in anti-clockwise order. By inspecting the location of these points (see Figure 6.7), one can see that $q_3$ is then outside $\mathcal{P}_1$, which is a contradiction. Thus we have proved Proposition 61 (3).

### 6.5.4 Restricted Type 4

In this subsection we exclude restricted type-4 NMFs of $M$, i.e., we prove Proposition 61 (4). Suppose there is a restricted type-4 NMF $M = L \cdot R$. That is, $M = L \cdot R$ is a stochastic and 5-dimensional restricted NMF such that $k = 3$, $k_1 = 1$, $k_2 = 1$. After a suitable permutation of its columns, $L$ matches the pattern

$$L = \begin{pmatrix} + & 0 & 0 & 0 & 0 \\ 0 & + & 0 & 0 & 0 \\ \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot \end{pmatrix}.$$

Clearly, columns $M_{:,1}, M_{:,2}, M_{:,3}$ all lie in the convex hull of $L_{:,1}, L_{:,3}, L_{:,4}, L_{:,5}$.

Since the NMF $M = L \cdot R$ is restricted, we have $\mathrm{Col}(L) = \mathrm{Col}(M)$ and, in particular, $L_{:,1}, \ldots, L_{:,5} \in \mathrm{Col}(M)$. Since matrices $L$ and $M$ are both stochastic, $L_{:,1}, \ldots, L_{:,5}$ lie in the affine span of the columns of $M$, i.e., the 3-dimensional affine subspace $\mathcal{V} \subseteq \mathbb{R}^6$ defined at the beginning of this section. Moreover since $L$ is stochastic, we have $L_{:,1}, \ldots, L_{:,5} \in \mathcal{V} \cap \mathbb{R}^6_+ = \mathcal{P}'$. Thus, there are uniquely defined points $q_1, \ldots, q_5 \in \mathcal{P}$ such that $f(q_i) = L_{:,i}$ for $i \in [5]$. Since $L_{:,1}$ has 0 in its second row, by inspecting the definition of the map $f$ we see that $q_1 \in \mathcal{P}_0$, where $\mathcal{P}_0$ is visualised in Figure 6.6. Since $L_{:,3}, L_{:,4}, L_{:,5}$ have 0 in their first two rows, by inspecting the definition of the map $f$ we see that $q_3, q_4, q_5$ lie on the $x$-axis in $\mathbb{R}^3$. Recall that the convex hull of $q_1, q_3, q_4, q_5$ includes $r_1, r_2, r_3$. Defining points $\hat{q}_3 = (0, 0, 0)^\top$ and $\hat{q}_4 = (1, 0, 0)^\top$, we get that the triangle $\triangle \hat{q}_3 \hat{q}_4 q_1$ includes $r_1, r_2, r_3$.

Therefore, if there is a restricted type-4 NMF of $M$ then there is a type-2 NMF of $M$. We have shown in Section 6.5.2 that $M$ does not have a type-2 NMF, hence Proposition 61 (4) follows.

## 6.6 Conclusion

In this chapter, we have answered the restricted NMF variant of the Cohen–Rothblum problem negatively by exhibiting a rank-4 rational matrix whose optimal restricted nonnegative factorization requires factors that have irrational entries. Moreover, we have shown that our counterexample is optimal inasmuch as rational matrices of rank 3 or less always have an optimal rational restricted NMF.

# Chapter 7

# Nonnegative Matrix Factorization Requires Irrationality

**Abstract**

*We show that the nonnegative ranks of a nonnegative matrix over the reals and over the rationals may differ. This solves a problem posed by Cohen and Rothblum in 1993. This chapter is an extension of the material in [Chistikov et al., 2017] and [Chistikov et al.].*

---

## 7.1 Introduction

In this chapter we give an example of a matrix whose nonnegative ranks over the reals and over the rationals differ. We thereby answer a longstanding open problem posed by Cohen and Rothblum [1993].

In Chapter 6 we showed a version of this result for the *restricted* nonnegative rank. Specifically, in Section 6.4 we exhibited a rational matrix whose restricted

nonnegative ranks over $\mathbb{R}$ and $\mathbb{Q}$ differ. Our approach in this chapter draws on the ideas from Section 6.4 and extends them in a non-trivial way.

In Section 6.4, we constructed a rational matrix $M$ whose *restricted* NMF $M = W \cdot H$ with minimal inner dimension is *unique* (up to permutation and rescaling of columns of $W$) and has irrational entries in the factors $W$ and $H$. In this chapter, we show that for a larger matrix $M' = (\, M \;\; W_\varepsilon \,)$, where $W_\varepsilon$ is a nonnegative rational matrix which is entry-wise close to $W$, there is no other NMF, restricted or otherwise, of the same inner dimension. This requires ruling out several classes of other hypothetical factorizations of $M'$, while still ensuring that one appropriate NMF (with irrational entries in the factors $W$ and $H$) exists.

The main technical result in this chapter is to exhibit the above-mentioned matrix $W_\varepsilon$ that enables us to rule out type-4 factorizations of $M'$. The fundamental idea here is that, by including all columns of $W$ into the set of columns of $M'$, we could exclude factorizations of type 4. Unfortunately for this construction, the matrix $W$ has irrational entries; however, we can instead take any nonnegative *rational* matrix $W_\varepsilon$ that is sufficiently close to $W$, and undesirable factorizations will still be excluded.

In Section 7.2, we pick a specific matrix $W_\varepsilon$, but also describe a wider class of suitable substitutes. Then in Section 7.3 we show that the matrix $M' = (\, M \;\; W_\varepsilon \,)$ has different nonnegative rank over $\mathbb{R}$ and $\mathbb{Q}$.

## 7.2 Ruling out Type-4 Factorizations

In this section, we give the main technical ingredient needed to show Theorem 67 in the following section. Specifically, we exhibit a nonnegative rational matrix $W_\varepsilon$ which is entry-wise close to the matrix $W$ in (6.7) and has no type-4 NMF.

We begin by defining the matrix

$$W_\varepsilon = \begin{pmatrix} 0 & \frac{133}{165} & \frac{640}{2233} & 0 & 0 \\ \frac{1}{111540} & 0 & 0 & \frac{17209}{58047} & \frac{997}{5082} \\ \frac{114721}{892320} & \frac{1}{146850} & \frac{17}{506} & \frac{385}{1759} & \frac{2921}{203280} \\ \frac{47}{1248} & \frac{413}{5874} & \frac{1}{102718} & \frac{2915}{10554} & \frac{4381}{203280} \\ \frac{36}{169} & \frac{22}{267} & \frac{18674}{51359} & \frac{1}{116094} & \frac{3252}{4235} \\ \frac{276953}{446160} & \frac{1009}{24475} & \frac{16239}{51359} & \frac{1100}{5277} & \frac{1}{101640} \end{pmatrix} \in \mathbb{Q}_+^{6\times 5},$$

which is stochastic and can be seen as a perturbation of $W$. Matrix $W_\varepsilon$ factors through $W$ as it has a stochastic 5-dimensional NMF

$$W_\varepsilon = W \cdot H_\varepsilon \tag{7.1}$$

with $H_\varepsilon$ as follows:

$$\begin{pmatrix} \frac{30419}{40560} + \frac{28679\sqrt{2}}{162240} & \frac{-2728}{46725} + \frac{5791\sqrt{2}}{140175} & \frac{2741}{98049} - \frac{642\sqrt{2}}{32683} & \frac{-689}{10554} + \frac{15595\sqrt{2}}{337728} & \frac{389}{1848} - \frac{5501\sqrt{2}}{36960} \\ 0 & \frac{163318}{140175} - \frac{7277\sqrt{2}}{62300} & \frac{5958}{32683} - \frac{50543\sqrt{2}}{392196} & 0 & 0 \\ 0 & \frac{-2137}{20025} + \frac{6047\sqrt{2}}{80100} & \frac{11062}{14007} + \frac{8321\sqrt{2}}{56028} & 0 & 0 \\ \frac{7443}{8840} - \frac{51313\sqrt{2}}{86190} & 0 & 0 & \frac{148897}{179418} + \frac{172627\sqrt{2}}{1435344} & \frac{-1741}{26180} + \frac{1847\sqrt{2}}{39270} \\ \frac{-408157}{689520} + \frac{1154473\sqrt{2}}{2758080} & 0 & 0 & \frac{7039}{29903} - \frac{318541\sqrt{2}}{1913792} & \frac{134461}{157080} + \frac{1163\sqrt{2}}{11424} \end{pmatrix}$$

In this section, we exclude type-4 NMFs of $W_\varepsilon$. We recall from Section 6.4.1 that these are defined as 5-dimensional stochastic NMFs $W_\varepsilon = L \cdot R$ such that, up to some

permutation of its columns, $L$ matches the zero pattern:

$$L = \begin{pmatrix} + & 0 & 0 & 0 & 0 \\ 0 & + & 0 & 0 & 0 \\ \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot \end{pmatrix}. \tag{7.2}$$

We will then use this result in Section 7.3 to show that a larger matrix $(\ M \ \ W_\varepsilon\ )$ has no type-4 NMF. Our main result is:

**Proposition 65.** *Matrix $W_\varepsilon$ has no type-4 NMF.*

The main technical idea behind the proof of Proposition 65, given in Lemma 66 in Section 7.2.2, is to use constraint propagation to show that no matrix in a suitably small neighbourhood of $W$, that contains $W_\varepsilon$, admits a type-4 NMF.

### 7.2.1 Geometric Intuition

We recall from Section 6.5 that the columns of $M$ affinely span a 3-dimensional affine subspace $\mathcal{V} \subseteq \mathbb{R}^6$, while a 3-dimensional polytope $\mathcal{P}'$ is defined as the intersection of $\mathcal{V}$ with $\mathbb{R}^6_+$. Moreover we recall the parameterization $f : \mathbb{R}^3 \to \mathbb{R}^6$ of $\mathcal{V}$ and $\mathcal{P}'$. Since $W_\varepsilon = W \cdot H_\varepsilon$, we have $\mathrm{Col}(W_\varepsilon) \subseteq \mathrm{Col}(W) = \mathrm{Col}(M)$. Therefore, $(W_\varepsilon)_{:,i} \in \mathcal{P}'$ for each $i \in [5]$. In particular, we have $f(q_i^\varepsilon) = (W_\varepsilon)_{:,i}$ for each $i \in [5]$ where:

$$q_1^\varepsilon = \begin{pmatrix} \frac{99}{169} \\ 0 \\ \frac{1}{40560} \end{pmatrix},\ q_2^\varepsilon = \begin{pmatrix} \frac{121}{534} \\ \frac{133}{150} \\ 0 \end{pmatrix},\ q_3^\varepsilon = \begin{pmatrix} \frac{9337}{9338} \\ \frac{64}{203} \\ 0 \end{pmatrix},\ q_4^\varepsilon = \begin{pmatrix} \frac{1}{42216} \\ 0 \\ \frac{17209}{21108} \end{pmatrix},\ q_5^\varepsilon = \begin{pmatrix} \frac{813}{385} \\ 0 \\ \frac{997}{1848} \end{pmatrix}.$$

Applying the inverse of $f$ column-wise to the NMF $W_\varepsilon = W \cdot H_\varepsilon$, we obtain:

$$\begin{pmatrix} q_1^\varepsilon & q_2^\varepsilon & q_3^\varepsilon & q_4^\varepsilon & q_5^\varepsilon \end{pmatrix} = \begin{pmatrix} q_1^* & q_2^* & q_3^* & q_4^* & q_5^* \end{pmatrix} \cdot H_\varepsilon \,. \tag{7.3}$$

Since matrix $H_\varepsilon$ is stochastic, Eq. (7.3) implies that the points $q_i^\varepsilon$ are contained in the convex hull of the points $q_i^*$. In Figure 6.5, points $q_1^\varepsilon, \ldots, q_5^\varepsilon$ are close to $q_1^*, \ldots, q_5^*$, with $q_2^\varepsilon, q_3^\varepsilon$ being on the $xy$-face and $q_1^\varepsilon, q_4^\varepsilon, q_5^\varepsilon$ being on the $xz$-face, although they are not shown in Figure 6.5.

### 7.2.2 The Proof

In this section we prove Proposition 65. Specifically, in Lemma 66 we use constraint propagation to show that no matrix in a suitably small neighbourhood of the matrix $W$ from (6.7) admits a type-4 NMF. As $W_\varepsilon$ belongs to this neighbourhood, Lemma 66 implies Proposition 65. Indeed, the decimal expansion of the matrix $W_\varepsilon$ can be rounded as follows:

$$W_\varepsilon \approx \begin{pmatrix} 0 & 0.81 & 0.2866 & 0 & 0 \\ 0.9 \cdot 10^{-5} & 0 & 0 & 0.296 & 0.1962 \\ 0.1 & 0.7 \cdot 10^{-5} & 0.03360 & 0.219 & 0.0144 \\ 0.04 & 0.0703 & 0.97 \cdot 10^{-5} & 0.276 & 0.0216 \\ 0.2 & 0.08 & 0.4 & 0.9 \cdot 10^{-5} & 0.7679 \\ 0.621 & 0.04 & 0.316 & 0.208 & 0.98 \cdot 10^{-5} \end{pmatrix}. \tag{7.4}$$

From here it is clear that $W_\varepsilon$ satisfies the system of constraints (7.5).

**Lemma 66.** *For all stochastic matrices $\widetilde{W} \in \mathbb{R}_+^{6\times 5}$ satisfying the following entry-wise*

*constraints:*

$$\widetilde{W} = \begin{pmatrix} 0 & 0.8 \leq \cdot & 0.286 \leq \cdot \leq 0.287 & 0 & 0 \\ \cdot \leq \varepsilon & 0 & 0 & 0.29 \leq \cdot & 0.196 \leq \cdot \\ & \cdot \leq \varepsilon & 0.0335 \leq \cdot & 0.21 \leq \cdot & \cdot \leq 0.015 \\ & 0.07 \leq \cdot & \cdot \leq \varepsilon & 0.27 \leq \cdot & \cdot \leq 0.022 \\ & & & \cdot \leq \varepsilon & 0.767 \leq \cdot \\ 0.62 \leq \cdot & & \cdot \leq 0.32 & \cdot \leq 0.21 & \cdot \leq \varepsilon \end{pmatrix} \tag{7.5}$$

*where* $\varepsilon = 10^{-5}$*, there exists no type-*4 *NMF of* $\widetilde{W}$*.*

*Proof.* Towards a contradiction, assume that there exists some stochastic matrix $\widetilde{W} \in \mathbb{R}_+^{6\times 5}$ that satisfies all constraints in (7.5) and has a 5-dimensional stochastic NMF $\widetilde{W} = L \cdot R$ such that, up to some permutation of its columns, $L$ matches the zero pattern (7.2). We will use constraint propagation to derive lower and upper bounds for various entries of the matrices $L$ and $R$ until we reach a contradiction.

All columns of $\widetilde{W}$ lie in the convex hull of the columns of $L$ since

$$\widetilde{W}_{:,j} = L \cdot R_{:,j} = \sum_{i=1}^{5} R_{i,j} \cdot L_{:,i}$$

for all $j \in [5]$. Consider the columns $\widetilde{W}_{:,4}$ and $\widetilde{W}_{:,5}$. Since $L$ matches the zero pattern (7.2), we have $\widetilde{W}_{2,4} = L_{2,2} \cdot R_{2,4}$ and $\widetilde{W}_{2,5} = L_{2,2} \cdot R_{2,5}$. From here, using (7.5) and the fact that $L$ is stochastic, we obtain the constraints:

$$0.29 \leq \widetilde{W}_{2,4} = L_{2,2} \cdot R_{2,4} \leq R_{2,4} \qquad 0.196 \leq \widetilde{W}_{2,5} = L_{2,2} \cdot R_{2,5} \leq R_{2,5}. \tag{7.6}$$

For every $i \in [6]$ and $j \in [5]$, since $\widetilde{W}_{i,j} = L_{i,:} \cdot R_{:,j}$ and matrices $L, R$ are nonnegative we have the inequality $L_{i,k} \cdot R_{k,j} \leq \widetilde{W}_{i,j}$ for all $k \in [5]$. We could thus

compute an upper bound on $L_{i,k}$ (resp., on $R_{k,j}$) if we had a lower bound on $R_{k,j}$ (resp., on $L_{i,k}$). We refer to this as computing *simple upper bounds* through $\widetilde{W}_{i,j}$.

Since $0.196 \leq R_{2,5}$ by (7.6) and $\widetilde{W}$ satisfies the constraints in (7.5), we can compute the following simple upper bounds through $\widetilde{W}_{3,5}$, $\widetilde{W}_{4,5}$, and $\widetilde{W}_{6,5}$:

$$L_{3,2} \leq \frac{0.015}{0.196} \leq 0.077, \qquad L_{4,2} \leq \frac{0.022}{0.196} \leq 0.12, \qquad L_{6,2} \leq \frac{\varepsilon}{0.196} \leq 6\varepsilon.$$

Moreover, the lower bound $0.29 \leq R_{2,4}$ from (7.6) gives a simple upper bound through $\widetilde{W}_{5,4}$:

$$L_{5,2} \leq \frac{\varepsilon}{0.29} \leq 4\varepsilon.$$

Since $L$ is stochastic, the derived upper bounds on $L_{i,2}$ for $3 \leq i \leq 6$ give:

$$L_{2,2} \geq 1 - (0.077 + 0.12 + 10\varepsilon) \geq 0.8.$$

We now summarise the derived constraints that are used in the next argument:

$$\begin{array}{c} \\ L_{1,:} \\ L_{2,:} \\ L_{3,:} \\ L_{4,:} \\ L_{5,:} \\ L_{6,:} \end{array} \begin{array}{c} \begin{array}{ccccc} L_{:,1} & L_{:,2} & L_{:,3} & L_{:,4} & L_{:,5} \end{array} \\ \begin{pmatrix} & & & & \\ & 0.8 \leq \cdot & & & \\ & & & & \\ & & & & \\ & \cdot \leq 4\varepsilon & & & \\ & \cdot \leq 6\varepsilon & & & \end{pmatrix} \end{array} \begin{array}{c} \\ R_{1,:} \\ R_{2,:} \\ R_{3,:} \\ R_{4,:} \\ R_{5,:} \end{array} \begin{array}{c} \begin{array}{ccccc} R_{:,1} & R_{:,2} & R_{:,3} & R_{:,4} & R_{:,5} \end{array} \\ \begin{pmatrix} & & & & \\ & & & & \\ & & & & 0.196 \leq \cdot \\ & & & & \\ & & & & \\ & & & & \end{pmatrix} \end{array}. \quad (7.7)$$

The column $\widetilde{W}_{:,1}$ lies in the convex hull of the columns $L_{:,2}, L_{:,3}, L_{:,4}, L_{:,5}$. The lower bound $L_{2,2} \geq 0.8$ yields the simple upper bound $R_{2,1} \leq 2\varepsilon$ through $\widetilde{W}_{2,1}$. Assume,

without loss of generality, that $L_{6,3} = \max\{L_{6,3}, L_{6,4}, L_{6,5}\}$. From

$$0.62 \leq \widetilde{W}_{6,1} = L_{6,:} \cdot R_{:,1} \leq L_{6,2} \cdot R_{2,1} + (1 - R_{2,1}) \cdot L_{6,3} \leq L_{6,2} \cdot R_{2,1} + L_{6,3}$$

we get $L_{6,3} \geq 0.62 - 6\varepsilon \cdot 2\varepsilon \geq 0.61$.

The column $\widetilde{W}_{:,5}$ lies in the convex hull of columns $L_{:,2}, L_{:,3}, L_{:,4}, L_{:,5}$. Since $L_{5,2} \leq 4\varepsilon$ and $R_{2,5} \geq 0.196$, we now have

$$0.767 \leq \widetilde{W}_{5,5} = L_{5,:} \cdot R_{:,5} \leq L_{5,2} + (1 - R_{2,5}) \cdot \max\{L_{5,3}, L_{5,4}, L_{5,5}\},$$

yielding the bound

$$\max\{L_{5,3}, L_{5,4}, L_{5,5}\} \geq \frac{0.767 - 4\varepsilon}{1 - 0.196} \geq 0.9539.$$

Since $L_{6,3} \geq 0.61$ and $L$ is stochastic, $\max\{L_{5,3}, L_{5,4}, L_{5,5}\}$ is either $L_{5,4}$ or $L_{5,5}$. Without loss of generality, let $\max\{L_{5,3}, L_{5,4}, L_{5,5}\} = L_{5,4}$. Then, $L_{5,4} \geq 0.9539$.

We recall some of the obtained constraints on the entries of $L$ in the following:

$$\begin{array}{c} \\ L_{1,:} \\ L_{2,:} \\ L_{3,:} \\ L_{4,:} \\ L_{5,:} \\ L_{6,:} \end{array} \begin{array}{c} \begin{array}{ccccc} L_{:,1} & L_{:,2} & L_{:,3} & L_{:,4} & L_{:,5} \end{array} \\ \begin{pmatrix} & & & & \\ & & & & \\ & \cdot \leq 0.077 & & & \\ & \cdot \leq 0.12 & & & \\ & & & 0.9539 \leq \cdot & \\ & & 0.61 \leq \cdot & & \end{pmatrix} \end{array}. \tag{7.8}$$

The column $\widetilde{W}_{:,4}$ lies in the convex hull of columns $L_{:,2}, L_{:,3}, L_{:,4}, L_{:,5}$. By multi-

plying the row vector $\begin{pmatrix} 0 & 0 & 2 & 0 & 0 & -1 \end{pmatrix}$ with $\widetilde{W}_{:,4} = L \cdot R_{:,4}$, we get:

$$2\widetilde{W}_{3,4} - \widetilde{W}_{6,4} = (2L_{3,:} - L_{6,:}) \cdot R_{:,4}.$$

Since $\widetilde{W}_{3,4} \geq 0.21$ and $\widetilde{W}_{6,4} \leq 0.21$, we have $2\widetilde{W}_{3,4} - \widetilde{W}_{6,4} \geq 0.21$. On the other hand, we have $L_{3,4} \leq 0.0461$ because of the lower bound $L_{5,4} \geq 0.9539$. Moreover,

$$2L_{3,3} - L_{6,3} \leq 2\,(1 - L_{6,3}) - L_{6,3} = 2 - 3L_{6,3}.$$

Hence,

$$\begin{aligned} 0.21 &\leq (2L_{3,2} - L_{6,2})R_{2,4} + (2L_{3,3} - L_{6,3})R_{3,4} + (2L_{3,4} - L_{6,4})R_{4,4} + 2L_{3,5} \\ &\leq \max\{\underbrace{2L_{3,2}}_{2\cdot 0.077},\ \underbrace{2 - 3L_{6,3}}_{2-3\cdot 0.61},\ \underbrace{2L_{3,4}}_{2\cdot 0.0461}\} + 2L_{3,5}\ , \end{aligned}$$

which implies that

$$L_{3,5} \geq 0.02. \tag{7.9}$$

Next we derive a lower bound on $L_{4,5}$ by applying a similar argument on the column $\widetilde{W}_{:,4}$, but considering the $4^{\text{th}}$ and $6^{\text{th}}$ rows. First, by multiplying the row vector $\begin{pmatrix} 0 & 0 & 0 & 2 & 0 & -1 \end{pmatrix}$ with $\widetilde{W}_{:,4} = L \cdot R_{:,4}$ we obtain the equality:

$$2\widetilde{W}_{4,4} - \widetilde{W}_{6,4} = (2L_{4,:} - L_{6,:}) \cdot R_{:,4}.$$

We see that $2\widetilde{W}_{4,4} - \widetilde{W}_{6,4} \geq 2 \cdot 0.27 - 0.21 \geq 0.33$. The entry $L_{4,4}$ satisfies the same upper bound as $L_{3,4}$: indeed, $L_{4,4} \leq 1 - L_{5,4} \leq 0.0461$. Moreover,

$$2L_{4,3} - L_{6,3} \leq 2(1 - L_{6,3}) - L_{6,3} = 2 - 3L_{6,3}.$$

We have

$$\begin{aligned} 0.33 &\leq (2L_{4,2} - L_{6,2})R_{2,4} + (2L_{4,3} - L_{6,3})R_{3,4} + (2L_{4,4} - L_{6,4})R_{4,4} + 2L_{4,5} \\ &\leq \max\{\underbrace{2L_{4,2}}_{2\cdot 0.12}, \underbrace{2 - 3L_{6,3}}_{2-3\cdot 0.61}, \underbrace{2L_{4,4}}_{2\cdot 0.0461}\} + 2L_{4,5} \ , \end{aligned}$$

which results in the constraint $L_{4,5} \geq 0.045$.

The column $L_{:,1}$ is the only column of $L$ that has a positive first coordinate; hence it is the only column of $L$ that contributes to the positive first coordinates in the columns $\widetilde{W}_{:,2}$ and $\widetilde{W}_{:,3}$. We obtain the following lower bounds through $\widetilde{W}_{1,2}$ and $\widetilde{W}_{1,3}$:

$$0.8 \leq \widetilde{W}_{1,2} = L_{1,1} \cdot R_{1,2} \text{ implying that } 0.8 \leq L_{1,1} \text{ and } 0.8 \leq R_{1,2} \tag{7.10}$$

$$0.286 \leq \widetilde{W}_{1,3} = L_{1,1} \cdot R_{1,3} \leq R_{1,3}. \tag{7.11}$$

The bounds in (7.10) and (7.11) enable us to derive the simple upper bound $L_{3,1} \leq 2\varepsilon$ through $\widetilde{W}_{3,2}$. We get the following simple upper bounds on the entries of the column $R_{:,3}$:

- $R_{1,3} \leq \frac{0.287}{0.8} \leq 0.36$ through $\widetilde{W}_{1,3}$ since $L_{1,1} \geq 0.8$,
- $R_{3,3} \leq \frac{0.32}{0.61} \leq 0.53$ through $\widetilde{W}_{6,3}$ since $L_{6,3} \geq 0.61$,
- $R_{5,3} \leq \frac{\varepsilon}{0.045} \leq 23\varepsilon$ through $\widetilde{W}_{4,3}$ since $L_{4,5} \geq 0.045$.

These upper bounds constrain the weights of the columns of $R$ in the convex combination that expresses $\widetilde{W}_{:,3}$. Since $\widetilde{W}_{:,3}$ lies in the convex hull of $L_{:,1}, L_{:,3}, L_{:,4}, L_{:,5}$, we then get

$$R_{4,3} \geq 1 - 0.36 - 0.5 - 23\varepsilon \geq 0.1.$$

Using this bound and the bound on $R_{1,3}$ in (7.11), simple upper bounds $L_{4,4} \leq 10\varepsilon$ and $L_{4,1} \leq 4\varepsilon$ are given through $\widetilde{W}_{4,3}$. As $L_{3,5} \geq 0.02$ by (7.9), we have $R_{5,2} \leq 50\varepsilon$ through $\widetilde{W}_{3,2}$.

Here we highlight some of the obtained constraints on the entries of $L$ and $R$:

$$\begin{array}{c} \\ L_{1,:} \\ L_{2,:} \\ L_{3,:} \\ L_{4,:} \\ L_{5,:} \\ L_{6,:} \end{array} \begin{array}{c} \begin{array}{ccccc} L_{:,1} & L_{:,2} & L_{:,3} & L_{:,4} & L_{:,5} \end{array} \\ \begin{pmatrix} & & & & \\ & & & & \\ \cdot \leq 2\varepsilon & & & & \\ \cdot \leq 4\varepsilon & & & \cdot \leq 10\varepsilon & \\ & & & 0.9539 \leq \cdot & \\ & & 0.61 \leq \cdot & & \end{pmatrix} \end{array} \begin{array}{c} \\ R_{1,:} \\ R_{2,:} \\ R_{3,:} \\ R_{4,:} \\ R_{5,:} \end{array} \begin{array}{c} \begin{array}{ccccc} R_{:,1} & R_{:,2} & R_{:,3} & R_{:,4} & R_{:,5} \end{array} \\ \begin{pmatrix} & 0.8 \leq \cdot & 0.286 \leq \cdot & & \\ & & & & \\ & & & & \\ & & & & \\ & \cdot \leq 50\varepsilon & \cdot \leq 23\varepsilon & & \end{pmatrix} \end{array}.$$

The column $\widetilde{W}_{:,2}$ lies in the convex hull of $L_{:,1}, L_{:,3}, L_{:,4}, L_{:,5}$. Using the bounds recalled above, we get:

$$\begin{aligned} 0.07 \leq \widetilde{W}_{4,2} = L_{4,:} \cdot R_{:,2} &= L_{4,1}R_{1,2} + L_{4,3}R_{3,2} + L_{4,4}R_{4,2} + L_{4,5}R_{5,2} \\ &\leq \underbrace{L_{4,1}}_{4\varepsilon} + L_{4,3}\underbrace{(1 - R_{1,2})}_{0.2} + \underbrace{L_{4,4}}_{10\varepsilon} + \underbrace{R_{5,2}}_{50\varepsilon}, \end{aligned}$$

resulting in the lower bound $0.346 \leq L_{4,3}$. By a similar argument on the column $\widetilde{W}_{:,3}$, and by using the lower bound on $R_{1,3}$ recalled above, we get:

$$\begin{aligned} 0.0335 \leq \widetilde{W}_{3,3} = L_{3,:} \cdot R_{:,3} &= L_{3,1}R_{1,3} + L_{3,3}R_{3,3} + L_{3,4}R_{4,3} + L_{3,5}R_{5,3} \\ &\leq \underbrace{L_{3,1}}_{2\varepsilon} + \max\{L_{3,3}, L_{3,4}\}\underbrace{(1 - R_{1,3})}_{0.714} + \underbrace{R_{5,3}}_{23\varepsilon}. \end{aligned}$$

This gives the lower bound:

$$\max\{L_{3,3}, L_{3,4}\} \geq 0.0465. \tag{7.12}$$

However, observe that $L_{3,3} \leq 1 - L_{4,3} - L_{6,3} \leq 0.044$. Moreover since $L_{5,4} \geq 0.9539$, we have $L_{3,4} \leq 0.0461$. Thus $\max\{L_{3,3}, L_{3,4}\} \leq 0.0461$, which contradicts (7.12).

This completes the proof. □

## 7.3 NMF Requires Irrational Numbers

In this section, we give the main contribution of this chapter: a rational matrix whose respective nonnegative ranks over $\mathbb{R}$ and $\mathbb{Q}$ are different.

**Theorem 67.** *Let* $M' = (\, M \;\; W_\varepsilon \,) \in \mathbb{Q}_+^{6\times 11}$ *where:*

$$M = \begin{pmatrix} \frac{5}{44} & \frac{5}{11} & \frac{85}{121} & 0 & 0 & 0 \\ 0 & 0 & 0 & \frac{2}{11} & \frac{3}{11} & \frac{7}{33} \\ \frac{1}{11} & \frac{1}{44} & \frac{2}{121} & \frac{1}{44} & \frac{15}{88} & \frac{17}{88} \\ \frac{1}{44} & \frac{1}{44} & \frac{8}{121} & \frac{1}{44} & \frac{19}{88} & \frac{5}{24} \\ \frac{3}{11} & \frac{3}{11} & \frac{12}{121} & \frac{8}{11} & \frac{2}{11} & \frac{2}{33} \\ \frac{1}{2} & \frac{5}{22} & \frac{14}{121} & \frac{1}{22} & \frac{7}{44} & \frac{43}{132} \end{pmatrix} \in \mathbb{Q}_+^{6\times 6},$$

$$W_\varepsilon = \begin{pmatrix} 0 & \frac{133}{165} & \frac{640}{2233} & 0 & 0 \\ \frac{1}{111540} & 0 & 0 & \frac{17209}{58047} & \frac{997}{5082} \\ \frac{114721}{892320} & \frac{1}{146850} & \frac{17}{506} & \frac{385}{1759} & \frac{2921}{203280} \\ \frac{47}{1248} & \frac{413}{5874} & \frac{1}{102718} & \frac{2915}{10554} & \frac{4381}{203280} \\ \frac{36}{169} & \frac{22}{267} & \frac{18674}{51359} & \frac{1}{116094} & \frac{3252}{4235} \\ \frac{276953}{446160} & \frac{1009}{24475} & \frac{16239}{51359} & \frac{1100}{5277} & \frac{1}{101640} \end{pmatrix} \in \mathbb{Q}_+^{6\times 5}.$$

*The nonnegative rank of* $M'$ *over* $\mathbb{R}$ *is* 5. *The nonnegative rank of* $M'$ *over* $\mathbb{Q}$ *is* 6.

In Theorem 59 we showed that the respective *restricted* nonnegative ranks of the matrix $M$ over $\mathbb{R}$ and $\mathbb{Q}$ are different. Theorem 67 refers to general nonnegative rank; the rest of this section is devoted to its proof.

Matrix $M'$ is stochastic. As argued in Section 5.3, we only need to consider

stochastic NMFs of $M'$ in order to determine its nonnegative rank. We recall from Equations (6.6) and (7.1) that matrices $M$ and $W_\varepsilon$ have, respectively, stochastic 5-dimensional NMFs $M = W \cdot H$ and $W_\varepsilon = W \cdot H_\varepsilon$ where:

$$W = \begin{pmatrix} 0 & \frac{5}{7} + \frac{5\sqrt{2}}{77} & \frac{15+5\sqrt{2}}{77} & 0 & 0 \\ 0 & 0 & 0 & \frac{20+2\sqrt{2}}{77} & \frac{48-8\sqrt{2}}{187} \\ \frac{\sqrt{2}}{11} & 0 & \frac{4-\sqrt{2}}{77} & \frac{3}{14} + \frac{\sqrt{2}}{308} & \frac{14-8\sqrt{2}}{187} \\ \frac{-1+\sqrt{2}}{11} & \frac{4+\sqrt{2}}{77} & 0 & \frac{39}{154} + \frac{5\sqrt{2}}{308} & \frac{21-12\sqrt{2}}{187} \\ \frac{8-4\sqrt{2}}{11} & \frac{12-4\sqrt{2}}{77} & \frac{4}{11} & 0 & \frac{104+28\sqrt{2}}{187} \\ \frac{4+2\sqrt{2}}{11} & \frac{6-2\sqrt{2}}{77} & \frac{30-4\sqrt{2}}{77} & \frac{3}{11} - \frac{\sqrt{2}}{22} & 0 \end{pmatrix}.$$

Thus, matrix $M'$ has the following stochastic 5-dimensional NMF:

$$M' = W \cdot \begin{pmatrix} H & H_\varepsilon \end{pmatrix}. \tag{7.13}$$

**Remark 68.** *The columns of $M'$ and $W$ span the same vector space. It follows that the restricted nonnegative ranks of $M'$ over $\mathbb{R}$ and $\mathbb{Q}$ are 5 and 6, respectively.*

Factorization (7.13) shows that $M'$ has nonnegative rank at most 5 over $\mathbb{R}$. Observe that the entries of the factors $W$, $H$, and $H_\varepsilon$ are irrational. To prove Theorem 67, it suffices to show that $M'$ has no 5-dimensional NMF over $\mathbb{Q}$.

To this end, we recall from Section 6.4.1 the classification of stochastic and at most 5-dimensional NMFs $M = L \cdot R$ into four types according to the zero pattern in the first two rows of the left factor $L$. (These four types are illustrated in Figure 6.4 for inner dimension 5.) Since $M' = (\, M \;\; W_\varepsilon \,)$, it follows from Proposition 60 that any stochastic NMF $M' = L \cdot R$ of inner dimension at most 5 has type 1, 2, 3, or 4. The following corollary enables us to exclude rational NMFs $M' = L \cdot R$ of each of the four types[1], thereby also showing that $M'$ has nonnegative rank exactly 5 over $\mathbb{R}$.

[1]In fact, we show that factorization (7.13) is the unique 5-dimensional NMF of $M'$.

**Corollary 69.** *Let $M'$ be the matrix from Theorem 67 and $W$ the matrix from* (6.7)*.*

(1) *If $M' = L \cdot R$ is a type-1 NMF then $L$ is equal to $W$, up to a permutation of its columns. In particular, $L$ is not rational.*

(2) *Matrix $M'$ has no type-2 NMF.*

(3) *Matrix $M'$ has no type-3 NMF.*

(4) *Matrix $M'$ has no type-4 NMF.*

*Proof.* We recall that $M' = (\, M \;\; W_\varepsilon \,)$. If $M' = L \cdot R$ is a type-1 NMF, then this yields a type-1 NMF of matrix $M$ where the left factor is $L$. Proposition 61 (1) implies that $L$ is equal to $W$, up to a permutation of its columns. Similarly, we know from Proposition 61 (2) and (3) that $M$ has no type-2 or type-3 NMF. Therefore, $M'$ has no type-2 or type-3 NMF. By Proposition 65, matrix $W_\varepsilon$ has no type-4 NMF. Therefore, matrix $M'$ has no type-4 NMF. □

Using this we can prove Theorem 67:

*Proof of Theorem 67.* Due to the NMF of $M'$ stated in (7.13), the nonnegative rank of $M'$ is at most 5. Corollary 69 implies that the nonnegative rank of $M'$ equals 5.

Since $M' = I_6 \cdot M'$, the nonnegative rank of $M'$ over $\mathbb{Q}$ is at most 6. By Corollary 69 there is no 5-dimensional NMF $M' = L \cdot R$ with $L$ rational. Hence, the nonnegative rank of $M'$ over $\mathbb{Q}$ equals 6. □

## 7.4 Conclusion

In this chapter, we have shown that an optimal nonnegative factorization of a rational matrix may require factors that have irrational entries. This answers an old open problem, due to Cohen and Rothblum [1993], negatively.

Our counterexample matrix has rank 4, thus showing that for matrices of rank 4 or greater the nonnegative ranks over $\mathbb{R}$ and over $\mathbb{Q}$ may differ. On the other hand, nonnegative ranks over $\mathbb{R}$ and $\mathbb{Q}$ are known to coincide for matrices of rank at most 2 [Cohen and Rothblum, 1993] or nonnegative rank at most 3 [Kubjas et al., 2015]. It remains an open question whether nonnegative ranks over $\mathbb{R}$ and over $\mathbb{Q}$ differ for rank-3 matrices whose nonnegative rank is at least 4, or whether our rank-4 counterexample is, in fact, minimal.

# Chapter 8

# Minimization of Labelled Markov Chains and Nonnegative Matrix Factorization

**Abstract**

*We apply our results on nonnegative matrix factorization to answer long-standing open problems concerning minimization of labelled Markov chains. We look at two notions of minimization: coverability and state minimization. In his seminal 1971 textbook, Paz asked the following question: When searching for a minimal covering LMC, can one look only at LMCs that have the same rank as the input LMC? We use restricted NMF to answer Paz's question negatively, thus falsifying a positive answer claimed in 1974. Secondly, we use our answer to the Cohen–Rothblum problem to show that state minimization of labelled Markov chains can require the introduction of irrational transition probabilities. This chapter is based on the material in [Chistikov et al., 2016] and [Chistikov et al., 2017].*

## 8.1 Introduction

Labelled Markov chains (LMCs), defined later in Section 8.2, are a fundamental model, variants of which occur under different names in the literature including (generative) probabilistic automata [Abe and Warmuth, 1992] and hidden Markov chains [Gillman and Sipser, 1994]. These models are widely employed in fields such as speech [Rabiner, 1989] and gesture [Chen et al., 2003] recognition, signal processing [Crouse et al., 1998], climate modelling [Ailliot et al., 2009], and computational biology [Eddy, 2004], e.g., sequence analysis [Durbin, 1998].

In this chapter, we look at minimization of labelled Markov chains. We consider two minimization problems: the problem of finding a minimal covering LMC and the problem of finding a minimal equivalent (initialised) LMC. We relate each of these problems to nonnegative matrix factorization.

The problem of finding a minimal *covering* LMC is to find an LMC that is capable of realizing all the distributions of the original LMC (and possibly more) with as few states as possible. More formally, an LMC $\mathcal{M}'$ *covers* an LMC $\mathcal{M}$ if for any initial distribution over the states of $\mathcal{M}$ there is an initial distribution over the states of $\mathcal{M}'$ such that $\mathcal{M}$ and $\mathcal{M}'$ are equivalent. The notion of coverability was introduced by Paz [1971] in his seminal textbook. There, Paz asked a question about the nature of minimal covering LMCs:

> "When searching for a minimal covering LMC, can one look only at LMCs that have the same rank as the input LMC?"

We will refer to this question as *Paz's conjecture*. Here the *rank* of an LMC is defined to be the rank of its *backward matrix*, which we formally define later in Section 8.3. In 1974, Paz's conjecture was supposedly answered positively by Bancilhon [1974].

In Section 8.3, we establish a tight connection between NMF and the coverability relation in LMCs. Using this connection we show that the correct answer to Paz's

conjecture is negative, thus falsifying the claim in [Bancilhon, 1974]. Instrumental to our counterexample is the observation that restricted nonnegative rank and nonnegative rank can be different. Indeed, the wrong claims in [Bancilhon, 1974] seem to implicitly rely on the opposite assumption, although the notions of nonnegative rank and restricted nonnegative rank had not yet been developed. In Section 8.3.1, we determine where exactly the paper [Bancilhon, 1974] goes wrong.

The second natural minimization problem, appropriate for initialised LMCs, asks for a minimal equivalent LMC. Here we consider the following analogue of the Cohen–Rothblum problem: Given an LMC with rational transition probabilities, is there always a minimal equivalent LMC with rational transition probabilities? We refer to this as the *rationality of LMC minimization question*.

In Section 8.4, we use our counterexample to the Cohen–Rothblum problem of Chapter 7 to give a negative answer to the rationality of LMC minimization question, by showing that there exists an LMC with rational transition probabilities that has no minimal equivalent LMC with rational transition probabilities.

## 8.2 Labelled Markov Chains

A *labelled Markov chain* (*LMC*) is a tuple $(n, \Sigma, \mu)$ where $n \in \mathbb{N}$ is the number of states, $\Sigma$ is a finite alphabet of *labels*, and function $\mu : \Sigma \to [0,1]^{n \times n}$ is such that $\sum_{\sigma \in \Sigma} \mu(\sigma)$ is a row-stochastic matrix. The set of states of the LMC is $[n] = \{1, \ldots, n\}$. For each label $\sigma$, every entry $\mu(\sigma)_{i,j}$ is called a *transition probability*. We view $\mu(\sigma)_{i,j}$ as the probability that, when the LMC is in the state $i$, it emits label $\sigma$ and moves to the state $j$.

We extend the function $\mu$ to words by defining $\mu(\varepsilon) := I_n$ and $\mu(\sigma_1 \ldots \sigma_k) := \mu(\sigma_1) \cdots \mu(\sigma_k)$ for all $k \in \mathbb{N}$ and all $\sigma_1, \ldots, \sigma_k \in \Sigma$. Observe that $\mu(xy) = \mu(x) \cdot \mu(y)$ for all words $x, y \in \Sigma^*$. We view $\mu(w)$ for a word $w \in \Sigma^*$ as follows: If the LMC is in

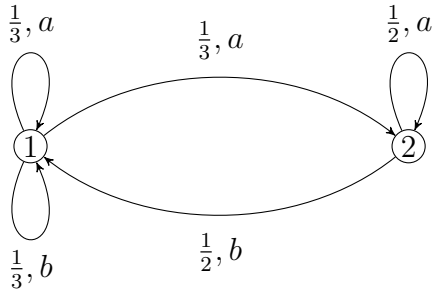



Figure 8.1: Initialised labelled Markov chain $\mathcal{M} = (2, \Sigma, \mu, e_1)$ where $\Sigma = \{a, b\}$, $\mu(a) = \begin{pmatrix} 1/3 & 1/3 \\ 0 & 1/2 \end{pmatrix}$, and $\mu(b) = \begin{pmatrix} 1/3 & 0 \\ 1/2 & 0 \end{pmatrix}$.

state $i$, it emits $w$ and moves to state $j$ in $|w|$ time steps, with probability $\mu(w)_{i,j}$.

An *initialised labelled Markov chain* (*LMC*) is a tuple $\mathcal{M} = (n, \Sigma, \mu, \pi)$ where $(n, \Sigma, \mu)$ is an LMC and $\pi \in [0, 1]^n$ is a stochastic row vector. Vector $\pi$ is called an *initial distribution* on the states of $\mathcal{M}$.

The intuitive behaviour of an initialised LMC $\mathcal{M}$ is as follows: At the start, $\mathcal{M}$ is in an initial state $i$ with probability $\pi_i$. At any time point after that $\mathcal{M}$ is in some state $l$, and at the next time step $\mathcal{M}$ moves to a state $j$ emitting label $\sigma$, with probability $\mu(\sigma)_{l,j}$. For any $k \in \mathbb{N}_0$, after $k$ time steps $\mathcal{M}$ is in some state $j$ having generated a word of length $k$. Thus, $\mathcal{M}$ naturally defines a probability distribution over $\Sigma^k$. Figure 8.1 gives an example of an initialised LMC.

Formally, initialised LMC $\mathcal{M}$ defines a function $\|\mathcal{M}\| : \Sigma^* \to [0, 1]$ where for every $w \in \Sigma^*$,

$$\|\mathcal{M}\|(w) := \sum_{i \in [n]} \sum_{j \in [n]} \pi_i \cdot \mu(w)_{i,j} = \pi \cdot \mu(w) \cdot \mathbf{1}_n,$$

which corresponds to the probability that $\mathcal{M}$ generates the word $w$ in $|w|$ time steps. Hence for any $k \in \mathbb{N}_0$, $\|\mathcal{M}\|_{\Sigma^k} : \Sigma^k \to [0, 1]$ is a probability distribution. For example, in Figure 8.1 we have $\|\mathcal{M}\|(ab) = \frac{1}{3} \cdot \frac{1}{3} + \frac{1}{3} \cdot \frac{1}{2} = \frac{5}{18}$.

## 8.3 Coverability of Labelled Markov Chains

In this section, we establish a connection between RNMF and the coverability relation for labelled Markov chains. We thereby answer an open question posed in 1971 by Paz about the nature of minimal covering labelled Markov chains.

Consider an LMC $\mathcal{M} = (n, \Sigma, \mu)$. (Note that this LMC is uninitialised.) For every $i \in [n]$ and $w \in \Sigma^*$, we write $pr_i^{\mathcal{M}}(w) := e_i \cdot \mu(w) \cdot \mathbf{1}_n$ for the probability that, starting in state $i$, $\mathcal{M}$ emits the word $w$ in $|w|$ time steps. For example, in Figure 8.1 we have $pr_2^{\mathcal{M}}(ab) = \frac{1}{4}$. More generally, for a given *initial distribution* $\pi$ on the set of states $[n]$ (viewed as a stochastic row vector), we write $pr_\pi^{\mathcal{M}}(w) := \pi \cdot \mu(w) \cdot \mathbf{1}_n$ for the probability that $\mathcal{M}$ emits the word $w$ in $|w|$ time steps starting from the state distribution $\pi$. We omit the superscript $\mathcal{M}$ from $pr_\pi^{\mathcal{M}}$ when it is clear from the context.

We say that an LMC $\mathcal{M} = (n, \Sigma, \mu)$ is *covered by* an LMC $\mathcal{M}' = (n', \Sigma, \mu')$, written $\mathcal{M}' \geq \mathcal{M}$, if for every initial distribution $\pi$ on $[n]$ there exists an initial distribution $\pi'$ on $[n']$ such that $pr_\pi^{\mathcal{M}}(w) = pr_{\pi'}^{\mathcal{M}'}(w)$ for every word $w \in \Sigma^*$.

The *backward matrix* of $\mathcal{M}$ is a matrix $\mathrm{Back}(\mathcal{M}) \in \mathbb{R}_+^{[n] \times \Sigma^*}$ where $\mathrm{Back}(\mathcal{M})_{i,w} = pr_i^{\mathcal{M}}(w)$ for every $i \in [n]$ and $w \in \Sigma^*$. The *rank* of $\mathcal{M}$, written $\mathrm{rank}(\mathcal{M})$, is defined to be the rank of $\mathrm{Back}(\mathcal{M})$. Matrix $\mathrm{Back}(\mathcal{M})$ is infinite, but its rank is at most $n$ since it has $n$ rows. It follows easily from the definition [see also Paz, 1971, Theorem 3.1] that $\mathcal{M}' \geq \mathcal{M}$ if and only if there exists a row-stochastic matrix $A$ such that

$$A \cdot \mathrm{Back}(\mathcal{M}') = \mathrm{Back}(\mathcal{M}). \tag{8.1}$$

LMCs can be seen as a special case of stochastic sequential machines, a class of probabilistic automata introduced and studied by Paz [1971]. More specifically, they are stochastic sequential machines with a singleton input alphabet and $\Sigma$ as output alphabet. In his seminal textbook on probabilistic automata, [Paz, 1971] asks the

following question:

**Question 70** (Paz, 1971, p. 38)**.** *If an $n$-state LMC $\mathcal{M}$ is covered by an $n'$-state LMC $\mathcal{M}'$ where $n' < n$, is $\mathcal{M}$ necessarily covered by some $n^*$-state LMC $\mathcal{M}^*$, where $n^* < n$, such that $\mathcal{M}^*$ and $\mathcal{M}$ have the same rank?*

In 1974, a positive answer to Question 70 was claimed by Bancilhon [1974, Theorem 13]. In fact, the paper [Bancilhon, 1974] makes a stronger claim, namely that the answer to Question 70 is yes, even if the inequality $n^* < n$ in Question 70 is replaced by $n^* \leq n'$. Note that this is equivalent to claiming that a minimal covering LMC can always be chosen to have minimal rank. To the contrary, we show:

**Theorem 71.** *The answer to Question 70 is negative.*

Theorem 71 falsifies the claim in [Bancilhon, 1974]. In Section 8.3.1 we discuss in detail the mistake in the latter paper.

To prove Theorem 71 we establish a tight connection between RNMF and LMC coverability:

**Proposition 72.** *Given a nonnegative matrix $M \in \mathbb{Q}_+^{n \times m}$ of rank $r$, one can compute in polynomial time an LMC $\mathcal{M} = (m+2, \Sigma, \mu)$ of rank $r+2$ such that for all $d \in \mathbb{N}$:*

(a) *any $d$-dimensional NMF $M = W \cdot H$ determines an LMC $\mathcal{M}' = (d+2, \Sigma, \mu')$ with $\mathcal{M}' \geq \mathcal{M}$ and $\operatorname{rank}(\mathcal{M}') = \operatorname{rank}(W) + 2$, and*

(b) *any LMC $\mathcal{M}' = (d+2, \Sigma, \mu')$ with $\mathcal{M}' \geq \mathcal{M}$ determines a $d$-dimensional NMF $M = W \cdot H$ with $\operatorname{rank}(\mathcal{M}') = \operatorname{rank}(W) + 2$.*

*In particular, for all $d \in \mathbb{N}$ the inequality $\operatorname{rrank}_+(M) \leq d$ holds if and only if $\mathcal{M}$ is covered by some $(d+2)$-state LMC $\mathcal{M}'$ such that $\mathcal{M}'$ and $\mathcal{M}$ have the same rank.*

Assuming Proposition 72 we can prove Theorem 71:

*Proof of Theorem 71.* Let $M \in \{0,1\}^{6\times 8}$ be the matrix from Example 4 on page 135. Let $\mathcal{M} = (10, \Sigma, \mu)$ be the associated LMC from Proposition 72. Since $M = I_6 \cdot M$ is an NMF with inner dimension 6, by Proposition 72 (a) there is an LMC $\mathcal{M}' = (8, \Sigma, \mu')$ with $\mathcal{M}' \geq \mathcal{M}$. Towards a contradiction, suppose the answer to Question 70 were yes. Then $\mathcal{M}$ is also covered by some $n^*$-state LMC $\mathcal{M}^*$, where $n^* \leq 9$, such that $\mathcal{M}^*$ and $\mathcal{M}$ have the same rank. The last sentence of Proposition 72 then implies that $\operatorname{rrank}_+(M) \leq 7$. But this contradicts the equality $\operatorname{rrank}_+(M) = 8$ from Example 4. Hence, the answer to Question 70 is no. □

To prove Proposition 72 we adapt a reduction from NMF to the trace-refinement problem in Markov decision processes by Fijalkow et al. [2016].

*Proof of Proposition 72.* Let $M \in \mathbb{Q}_+^{n\times m}$ be a nonnegative matrix of rank $r$. As argued in Section 5.3, without loss of generality we may assume that $M$ is stochastic and consider factorizations of $M$ into stochastic matrices only.

We define an LMC $\mathcal{M} = (m+2, \Sigma, \mu)$ with states $\{0, 1, \ldots, m, m+1\}$. The alphabet is $\Sigma = \{a_1, \ldots, a_m\} \cup \{b_1, \ldots, b_n\} \cup \{\checkmark\}$ and the function $\mu$, for all $i \in [m]$ and all $j \in [n]$, is defined by:

$$\mu(a_i)_{0,i} = \tfrac{1}{m}, \qquad \mu(b_j)_{i,m+1} = (M^\top)_{i,j} = M_{j,i}, \qquad \mu(\checkmark)_{m+1,m+1} = 1,$$

and all other entries of $\mu(a_i)$, $\mu(b_j)$, and $\mu(\checkmark)$ are 0. See Figure 8.2 for an example.

For the backward matrix $\operatorname{Back}(\mathcal{M})$, we have for every $i \in [m]$ and $j \in [n]$:

$$\operatorname{Back}(\mathcal{M})_{i,b_j} = \mu(b_j)_{i,:} \cdot \mathbf{1}_{m+2} = \mu(b_j)_{i,m+1} = M_{j,i}.$$

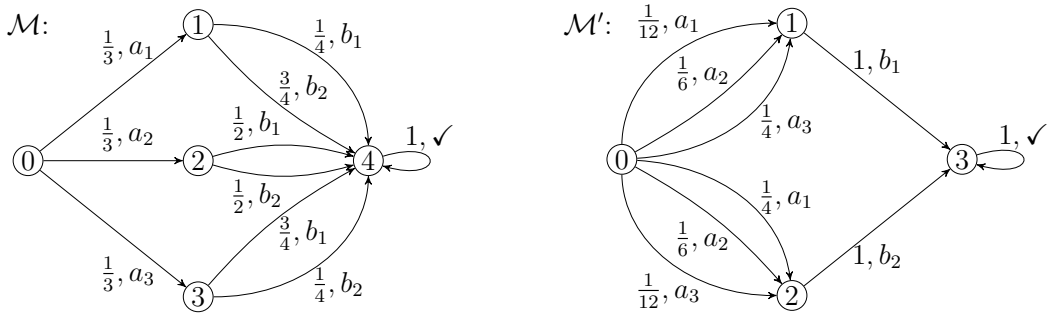



Figure 8.2: LMC $\mathcal{M}$ is constructed from matrix $M = \left(\begin{smallmatrix} 1/4 & 1/2 & 3/4 \\ 3/4 & 1/2 & 1/4 \end{smallmatrix}\right)$ whereas LMC $\mathcal{M}'$ is obtained by NMF $M = I_2 \cdot M$.

From here it is easy to see that for every $w \in \Sigma^*$:

$$\mathrm{Back}(\mathcal{M})_{:,w} = \begin{cases} \mathbf{1} & w = \varepsilon \\ \frac{1}{m} e_0^\top & w = a_i \text{ with } i \in [m] \\ e_{m+1}^\top & w = \checkmark^p \text{ with } p \in \mathbb{N} \\ (0, M_{j,:}, 0)^\top & w = b_j \checkmark^p \text{ with } j \in [n], p \in \mathbb{N}_0 \\ \frac{1}{m} M_{j,i} e_0^\top & w = a_i b_j \checkmark^p \text{ with } i \in [m], j \in [n], p \in \mathbb{N}_0 \\ \mathbf{0} & \text{otherwise;} \end{cases}$$

where $\checkmark^p$ denotes the $p$-fold concatenation of $\checkmark$ by itself. That is,

$$\mathrm{Back}(\mathcal{M}) = \begin{array}{c} \begin{array}{cccccccccccc} \varepsilon & b_1 & \cdots & b_n & \checkmark & a_i & a_i b_j & b_1\checkmark & \cdots & b_n\checkmark & \checkmark^2 & \cdots \end{array} \\ \begin{pmatrix} 1 & 0 & \cdots & 0 & 0 & \frac{1}{m} & \frac{1}{m} M_{j,i} & 0 & \cdots & 0 & 0 & \cdots \\ 1 & M_{1,1} & \cdots & M_{n,1} & 0 & 0 & 0 & M_{1,1} & \cdots & M_{n,1} & 0 & \cdots \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \cdots \\ 1 & M_{1,m} & \cdots & M_{n,m} & 0 & 0 & 0 & M_{1,m} & \cdots & M_{n,m} & 0 & \cdots \\ 1 & 0 & \cdots & 0 & 1 & 0 & 0 & 0 & \cdots & 0 & 1 & \cdots \end{pmatrix} \end{array}.$$

The following full-row submatrix of $\mathrm{Back}(\mathcal{M})$ (corresponding to the initial $n+2$ columns) clearly has the same column space as $\mathrm{Back}(\mathcal{M})$:

$$
\begin{array}{c}
 \\
0 \\
1 \\
\vdots \\
m \\
m{+}1
\end{array}
\begin{array}{c}
\begin{array}{ccccc} \varepsilon & b_1 & \cdots & b_n & \checkmark \end{array} \\
\begin{pmatrix}
1 & 0 & \cdots & 0 & 0 \\
1 & M_{1,1} & \cdots & M_{n,1} & 0 \\
\vdots & \vdots & \ddots & \vdots & \vdots \\
1 & M_{1,m} & \cdots & M_{n,m} & 0 \\
1 & 0 & \cdots & 0 & 1
\end{pmatrix}
\end{array}
$$

This implies that

$$\operatorname{rank}(\mathcal{M}) = \operatorname{rank}(\operatorname{Back}(\mathcal{M})) = \operatorname{rank}(M) + 2 = r + 2. \tag{8.2}$$

For direction (a), let $d \in \mathbb{N}$ and consider any NMF $M = W \cdot H$ where $W \in \mathbb{R}_+^{n \times d}$ and $H \in \mathbb{R}_+^{d \times m}$ are stochastic matrices. Define an LMC $\mathcal{M}' = (d+2, \Sigma, \mu')$ with the set of states $\{0, 1, \ldots, d, d+1\}$. The function $\mu'$, for all $i \in [m]$, $j \in [n]$, and $l \in [d]$, is defined by:

$$\mu'(a_i)_{0,l} = \tfrac{1}{m} H_{l,i}, \qquad \mu'(b_j)_{l,d+1} = W_{j,l}, \qquad \mu'(\checkmark)_{d+1,d+1} = 1,$$

and all other entries of $\mu'(a_i)$, $\mu'(b_j)$, and $\mu'(\checkmark)$ are 0. Since $H$ is stochastic, for every $i \in [m]$ we have

$$\operatorname{Back}(\mathcal{M}')_{:,a_i} = \left( \sum_{l \in [d]} \frac{1}{m} H_{l,i} \right) e_0^\top = \frac{1}{m} e_0^\top.$$

From here it follows easily that for every $w \in \Sigma^*$:

$$\mathrm{Back}(\mathcal{M}')_{:,w} = \begin{cases} \mathbf{1} & w = \varepsilon \\ \frac{1}{m} e_0^\top & w = a_i \text{ with } i \in [m] \\ e_{d+1}^\top & w = \checkmark^p \text{ with } p \in \mathbb{N} \\ (0, W_{j,:}, 0)^\top & w = b_j \checkmark^p \text{ with } j \in [n], p \in \mathbb{N}_0 \\ \frac{1}{m} M_{j,i} e_0^\top & w = a_i b_j \checkmark^p \text{ with } i \in [m], j \in [n], p \in \mathbb{N}_0 \\ \mathbf{0} & \text{otherwise.} \end{cases}$$

The following is a full-row submatrix of $\mathrm{Back}(\mathcal{M}')$ corresponding to the columns indexed by $\varepsilon, b_1, \ldots, b_n, \checkmark$:

$$\begin{array}{c} \\ 0 \\ 1 \\ \vdots \\ d \\ d{+}1 \end{array} \begin{array}{c} \begin{array}{ccccc} \varepsilon & b_1 & \cdots & b_n & \checkmark \end{array} \\ \begin{pmatrix} 1 & 0 & \cdots & 0 & 0 \\ 1 & W_{1,1} & \cdots & W_{n,1} & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 1 & W_{1,d} & \cdots & W_{n,d} & 0 \\ 1 & 0 & \cdots & 0 & 1 \end{pmatrix} \end{array}$$

Since this submatrix has the same column space as $\mathrm{Back}(\mathcal{M}')$,

$$\mathrm{rank}(\mathcal{M}') = \mathrm{rank}(\mathrm{Back}(\mathcal{M}')) = \mathrm{rank}(W) + 2. \tag{8.3}$$

Moreover, if the NMF $M = W \cdot H$ is restricted, then by (8.2) and (8.3) it holds that LMCs $\mathcal{M}$ and $\mathcal{M}'$ have the same rank.

In order to prove direction (a), it remains to show that $\mathcal{M}' \geq \mathcal{M}$. We define a

row-stochastic matrix

$$A := \begin{pmatrix} 1 & 0 & \cdots & 0 & 0 \\ 0 & H_{1,1} & \cdots & H_{d,1} & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & H_{1,m} & \cdots & H_{d,m} & 0 \\ 0 & 0 & \cdots & 0 & 1 \end{pmatrix} \in \mathbb{R}_+^{\{0,1,\ldots,m,m+1\}\times\{0,1,\ldots,d,d+1\}}.$$

For every $i \in [m]$ and $j \in [n]$, we have

$$\begin{aligned}(A \cdot \mathrm{Back}(\mathcal{M}'))_{i,b_j} &= A_{i,:} \cdot \mathrm{Back}(\mathcal{M}')_{:,b_j} \\ &= (H_{:,i})^\top \cdot (W_{j,:})^\top = (W_{j,:} \cdot H_{:,i})^\top = M_{j,i} = \mathrm{Back}(\mathcal{M})_{i,b_j}.\end{aligned}$$

From here it follows easily that $A \cdot \mathrm{Back}(\mathcal{M}') = \mathrm{Back}(\mathcal{M})$. Thus by (8.1), $\mathcal{M}' \geq \mathcal{M}$.

We now prove direction (b). Let LMC $\mathcal{M}' = (d+2, \Sigma, \mu')$ be such that $\mathcal{M}' \geq \mathcal{M}$. By (8.1), there exists a row-stochastic matrix $A$ such that $A \cdot \mathrm{Back}(\mathcal{M}') = \mathrm{Back}(\mathcal{M})$. Then, $M$ has the following $(d+2)$-dimensional NMF:

$$M^\top = \mathrm{Back}(\mathcal{M})_{[m],\{b_1,\ldots,b_n\}} = A_{[m],\{0,1,\ldots,d,d+1\}} \cdot \mathrm{Back}(\mathcal{M}')_{\{0,1,\ldots,d,d+1\},\{b_1,\ldots,b_n\}}. \quad (8.4)$$

Assuming $M$ is nonzero, there exist $i \in [m]$ and $j \in [n]$ such that $M_{j,i} > 0$. Then,

$$\frac{1}{m} M_{j,i} = \mathrm{Back}(\mathcal{M})_{0,a_i b_j} = A_{0,:} \cdot \mathrm{Back}(\mathcal{M}')_{:,a_i b_j} = \sum_{i_1=0}^{d+1} A_{0,i_1} \cdot \mathrm{Back}(\mathcal{M}')_{i_1,a_i b_j} > 0.$$

In particular, there exists $i_1 \in \{0, 1, \ldots, d, d+1\}$ such that $A_{0,i_1} \cdot \mathrm{Back}(\mathcal{M}')_{i_1,a_i b_j} > 0$. Without loss of generality, we may assume that $i_1 = 0$. That is,

$$A_{0,0} \cdot \mathrm{Back}(\mathcal{M}')_{0,a_i b_j} > 0. \quad (8.5)$$

Moreover,

$$1 = \mathrm{Back}(\mathcal{M})_{m+1,\checkmark} = A_{m+1,:} \cdot \mathrm{Back}(\mathcal{M}')_{:,\checkmark} = \sum_{i_2=0}^{d+1} A_{m+1,i_2} \cdot \mathrm{Back}(\mathcal{M}')_{i_2,\checkmark}.$$

In particular, there exists $i_2 \in \{0, 1, \ldots, d, d+1\}$ such that $A_{m+1,i_2} \cdot \mathrm{Back}(\mathcal{M}')_{i_2,\checkmark} > 0$. Note that $i_2 \neq 0$ since otherwise $\mathrm{Back}(\mathcal{M})_{m+1,a_i b_j} \geq A_{m+1,0} \cdot \mathrm{Back}(\mathcal{M}')_{0,a_i b_j} > 0$, which would yield a contradiction. We may therefore, without loss of generality, assume that $i_2 = d + 1$. That is,

$$A_{m+1,d+1} \cdot \mathrm{Back}(\mathcal{M}')_{d+1,\checkmark} > 0. \tag{8.6}$$

**Lemma 73.** *It holds that*

$$M^\top = A_{[m],[d]} \cdot \mathrm{Back}(\mathcal{M}')_{[d],\{b_1,\ldots,b_n\}}, \tag{8.7}$$

*where matrix $A_{[m],[d]}$ is row-stochastic.*

*Proof.* By (8.4), in order to prove (8.7) it suffices to show that

$$A_{[m],\{0,1,\ldots,d,d+1\}} \cdot \mathrm{Back}(\mathcal{M}')_{\{0,1,\ldots,d,d+1\},\{b_1,\ldots,b_n\}} = A_{[m],[d]} \cdot \mathrm{Back}(\mathcal{M}')_{[d],\{b_1,\ldots,b_n\}}.$$

That is, we need to show that for any $l_1 \in [m]$ and $l_2 \in [n]$:

$$A_{l_1,\{0,1,\ldots,d,d+1\}} \cdot \mathrm{Back}(\mathcal{M}')_{\{0,1,\ldots,d,d+1\},b_{l_2}} = A_{l_1,[d]} \cdot \mathrm{Back}(\mathcal{M}')_{[d],b_{l_2}},$$

which is equivalent to:

$$A_{l_1,0} \cdot \mathrm{Back}(\mathcal{M}')_{0,b_{l_2}} + A_{l_1,d+1} \cdot \mathrm{Back}(\mathcal{M}')_{d+1,b_{l_2}} = 0.$$

To show this, it suffices to show that $A_{l_1,0} = 0$ and $A_{l_1,d+1} = 0$. One could analogously

also show that $\text{Back}(\mathcal{M}')_{0,b_{l_2}} = 0$ and $\text{Back}(\mathcal{M}')_{d+1,b_{l_2}} = 0$.

If $A_{l_1,0} > 0$ then it would follow from (8.5) that

$$\text{Back}(\mathcal{M})_{l_1,a_ib_j} = A_{l_1,:} \cdot \text{Back}(\mathcal{M}')_{:,a_ib_j} \geq A_{l_1,0} \cdot \text{Back}(\mathcal{M}')_{0,a_ib_j} > 0,$$

which is a contradiction since $\text{Back}(\mathcal{M})_{l_1,a_ib_j} = 0$ for all $l_1 \in [m]$. Similarly, if $A_{l_1,d+1} > 0$ then by (8.6) we would have that

$$\text{Back}(\mathcal{M})_{l_1,\checkmark} = A_{l_1,:} \cdot \text{Back}(\mathcal{M}')_{:,\checkmark} \geq A_{l_1,d+1} \cdot \text{Back}(\mathcal{M}')_{d+1,\checkmark} > 0,$$

which is a contradiction since $\text{Back}(\mathcal{M})_{l_1,\checkmark} = 0$ for all $l_1 \in [m]$. This implies Eq. (8.7).

Note that we have shown that $A_{[m],0} = A_{[m],d+1} = \mathbf{0}$. Since matrix $A$ is row-stochastic, we conclude that matrix $A_{[m],[d]}$ is also row-stochastic. □

Equation 8.7 yields a $d$-dimensional stochastic NMF $M = W \cdot H$ where $W^\top = \text{Back}(\mathcal{M}')_{[d],\{b_1,\ldots,b_n\}}$ and $H^\top = A_{[m],[d]}$.

Using the structure of $\text{Back}(\mathcal{M})$, the equality $A \cdot \text{Back}(\mathcal{M}') = \text{Back}(\mathcal{M})$, and the inequalities (8.5) and (8.6), we get that the following is a full-row submatrix of $\text{Back}(\mathcal{M}')$ that has the same column space as $\text{Back}(\mathcal{M}')$:

$$\begin{array}{c} \\ 0 \\ 1 \\ \vdots \\ d \\ d{+}1 \end{array} \begin{array}{c} \begin{array}{ccccc} \varepsilon & b_1 & \cdots & b_n & \checkmark \end{array} \\ \begin{pmatrix} 1 & 0 & \cdots & 0 & 0 \\ 1 & W_{1,1} & \cdots & W_{n,1} & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 1 & W_{1,d} & \cdots & W_{n,d} & 0 \\ 1 & 0 & \cdots & 0 & 1 \end{pmatrix} \end{array}.$$

This implies that $\text{rank}(\mathcal{M}') = \text{rank}(\text{Back}(\mathcal{M}')) = \text{rank}(W^\top) + 2 = \text{rank}(W) + 2$.

Lastly, suppose $\mathcal{M}'$ and $\mathcal{M}$ have the same rank. Since $A \cdot \text{Back}(\mathcal{M}') = \text{Back}(\mathcal{M})$,

this implies that $\mathrm{Row}(\mathrm{Back}(\mathcal{M})) = \mathrm{Row}(\mathrm{Back}(\mathcal{M}'))$. Take any $v \in \mathrm{Col}(W)$. Since $W^\top = \mathrm{Back}(\mathcal{M}')_{[d],\{b_1,\ldots,b_n\}}$, there exists $u \in \mathrm{Row}(\mathrm{Back}(\mathcal{M}')) = \mathrm{Row}(\mathrm{Back}(\mathcal{M}))$ such that $v^\top = u_{\{b_1,\ldots,b_n\}}$. It follows that $v \in \mathrm{Col}(M)$, since $M^\top = \mathrm{Back}(\mathcal{M})_{[m],\{b_1,\ldots,b_n\}}$ and $\mathrm{Back}(\mathcal{M})_{0,b_j} = \mathrm{Back}(\mathcal{M})_{m+1,b_j} = 0$ for all $j \in [n]$. Hence, $\mathrm{Col}(W) \subseteq \mathrm{Col}(M)$. The NMF $M = W \cdot H$ is thus restricted and, in particular, $\mathrm{rrank}_+(M) \leq d$. □

### 8.3.1 Discussion of Erroneous Claims in the Literature

As mentioned above, Bancilhon [1974, Theorem 13] claims a statement that implies a positive answer to Paz's conjecture, i.e., Question 70. We have shown in Theorem 71 that the correct answer to Question 70 is negative. In this subsection, we track down where the paper [Bancilhon, 1974] goes wrong. The proof of [Bancilhon, 1974, Theorem 13] offered therein relies on another (wrong) claim about *cones*.

Let $\mathcal{V}$ be a vector space. Let $v_1, \ldots, v_n \in \mathcal{V}$. We recall from Section 5.5 that the polyhedral cone generated by the vectors $v_1, \ldots, v_n$ is the set

$$\left\{ \sum_{i=1}^{n} \lambda_i v_i \;\middle|\; \lambda_1, \ldots, \lambda_n \geq 0 \right\} \subseteq \mathcal{V}\,.$$

**Claim 74** (Bancilhon, 1974, Theorem 2, slightly paraphrased)**.** *Let $\mathcal{V}$ be a vector space. Let $\mathcal{V}' \subseteq \mathcal{V}$ be a vector subspace of $\mathcal{V}$. Let $\mathcal{C} \subseteq \mathcal{V}$ be a polyhedral cone generated by $n$ vectors that also span $\mathcal{V}$. Then, $\mathcal{C} \cap \mathcal{V}'$ is a polyhedral cone generated by at most $n$ vectors.*

Claim 74 is false. For a counterexample, consider the polyhedral cone $\mathcal{C}$ generated

by the following 6 vectors:

$$\begin{pmatrix}1\\0\\0\\1\\0\\0\end{pmatrix}, \begin{pmatrix}0\\0\\0\\1\\0\\0\end{pmatrix}, \begin{pmatrix}0\\1\\0\\0\\1\\0\end{pmatrix}, \begin{pmatrix}0\\0\\0\\0\\1\\0\end{pmatrix}, \begin{pmatrix}0\\0\\1\\0\\0\\1\end{pmatrix}, \begin{pmatrix}0\\0\\0\\0\\0\\1\end{pmatrix}.$$

Polyhedral cone $\mathcal{C}$ can be described equivalently as the conjunction of inequalities $0 \leq x_1 \leq x_4$ and $0 \leq x_2 \leq x_5$ and $0 \leq x_3 \leq x_6$. Let $\mathcal{V}' \subset \mathbb{R}^6$ be the vector space defined by the equalities $x_4 = x_5 = x_6$. Then, the cone $\mathcal{C}' := \mathcal{C} \cap \mathcal{V}'$ can be described by the inequalities $0 \leq x_1, x_2, x_3 \leq x_4 = x_5 = x_6$. The cone $\mathcal{C}'$ is generated by the following 8 vectors:

$$\begin{pmatrix}0\\0\\0\\1\\1\\1\end{pmatrix}, \begin{pmatrix}0\\0\\1\\1\\1\\1\end{pmatrix}, \begin{pmatrix}0\\1\\0\\1\\1\\1\end{pmatrix}, \begin{pmatrix}0\\1\\1\\1\\1\\1\end{pmatrix}, \begin{pmatrix}1\\0\\0\\1\\1\\1\end{pmatrix}, \begin{pmatrix}1\\0\\1\\1\\1\\1\end{pmatrix}, \begin{pmatrix}1\\1\\0\\1\\1\\1\end{pmatrix}, \begin{pmatrix}1\\1\\1\\1\\1\\1\end{pmatrix}.$$

All those vectors are extremal in cone $\mathcal{C}'$. Therefore, $\mathcal{C}'$ cannot be generated by fewer than 8 vectors. Hence, Claim 74 is false.

Let us further examine how Claim 74 was justified in [Bancilhon, 1974]. The proof offered therein starts with the following claim, which is stated there without further justification:

**Claim 75** (proof of Bancilhon, 1974, Theorem 2, slightly paraphrased)**.** *A cone $\mathcal{C}$ is generated by $n$ vectors if and only if it is limited by $n$ hyperplanes.*

Claim 75 is also false. For a counterexample, consider the cone $\mathcal{C} \subseteq \mathbb{R}^4_+$ limited by the following 6 hyperplanes:

$$0 \leq x_1 \leq y, \qquad 0 \leq x_2 \leq y, \qquad 0 \leq x_3 \leq y.$$

In particular, for any $y^* \geq 0$, all vectors in $\mathcal{C}$ with $y = y^*$ form (when projected onto the first three coordinates $x_1, x_2, x_3$) a cube of edge length $y^*$. The following set:

$$\{0,1\} \times \{0,1\} \times \{0,1\} \times \{1\} \quad \subset \quad \mathcal{C}$$

contains 8 vectors, all of which are extremal in $\mathcal{C}$. Hence $\mathcal{C}$ cannot be generated by 6 vectors, contradicting Claim 75.

## 8.4 Labelled Markov Chain Minimization Requires Irrationality

Our solution to the Cohen–Rothblum problem can be applied to the analogous question for labelled Markov chains (LMCs):

> Given an LMC with rational transition probabilities, is there always a minimal equivalent LMC with rational transition probabilities?

In this section, we answer this rationality of LMC minimization question negatively. Note that, since we are considering the notion of equivalence, we are working with *initialised* LMCs, although we refer to them simply as LMCs.

Formally, given two (initialised) LMCs $\mathcal{M}$ and $\mathcal{M}'$ over an alphabet $\Sigma$, we say that $\mathcal{M}$ is *equivalent to* $\mathcal{M}'$ if for any word $w \in \Sigma^*$, $\mathcal{M}$ and $\mathcal{M}'$ generate $w$ with equal probability, i.e., $\|\mathcal{M}\|(w) = \|\mathcal{M}'\|(w)$. An LMC is *minimal* if no equivalent LMC has fewer states. For example, in Figure 8.2 we have $\|\mathcal{M}\|(a_1b_1) = \frac{1}{12} = \|\mathcal{M}'\|(a_1b_1)$.

The reduction in Proposition 76 will be central to our solution to the rationality of LMC minimization question. To construct this reduction, we adapt reductions from NMF to the trace-refinement problem in Markov decision processes [Fijalkow et al., 2016] and to LMC coverability (Proposition 72).

**Proposition 76.** *Given a nonnegative matrix $M \in \mathbb{Q}_+^{n\times m}$, one can compute in polynomial time an LMC $\mathcal{M} = (m+2, \Sigma, \mu, \pi)$ with rational transition probabilities such that for all $d \in \mathbb{N}$:*

(i) *any $d$-dimensional NMF $M = W \cdot H$ determines an LMC $\mathcal{M}' = (d+2, \Sigma, \mu', \pi')$ which is equivalent to $\mathcal{M}$, and*

(ii) *any LMC $\mathcal{M}' = (d+2, \Sigma, \mu', \pi')$ which is equivalent to $\mathcal{M}$ determines a $d$-dimensional NMF $M = W \cdot H$.*

*In both* (i) *and* (ii)*, the NMF $M = W \cdot H$ is rational if and only if the LMC $\mathcal{M}'$ has rational transition probabilities.*

*Proof.* Let $M \in \mathbb{Q}_+^{n\times m}$. As noted in Section 5.3, without loss of generality we may assume that $M$ is stochastic and consider factorizations of $M$ into stochastic matrices only. We define an LMC $\mathcal{M} = (m+2, \Sigma, \mu, \pi)$ as follows: The set of states is $\{0, 1, \ldots, m, m+1\}$ and the alphabet is $\Sigma = \{a_1, \ldots, a_m\} \cup \{b_1, \ldots, b_n\} \cup \{\checkmark\}$. The initial distribution is $\pi = e_0$ and the function $\mu$, for all $i \in [m]$ and all $j \in [n]$, is defined by:

$$\mu(a_i)_{0,i} = \tfrac{1}{m}, \qquad \mu(b_j)_{i,m+1} = (M^\top)_{i,j} = M_{j,i}, \qquad \mu(\checkmark)_{m+1,m+1} = 1,$$

and all other entries of $\mu(a_i)$, $\mu(b_j)$, and $\mu(\checkmark)$ are 0. See Figure 8.2 for an example.

It is easy to see that:

$$\|\mathcal{M}\|(w) = \begin{cases} 1 & w = \varepsilon \\ \frac{1}{m} & w = a_i \text{ with } i \in [m] \\ \frac{1}{m} M_{j,i} & w = a_i b_j \checkmark^p \text{ with } i \in [m], j \in [n], p \in \mathbb{N}_0 \\ 0 & \text{otherwise.} \end{cases}$$

Here $\checkmark^p$ denotes the $p$-fold concatenation of the symbol $\checkmark$ by itself.

We first prove (i). Let $M = W \cdot H$ where $W \in \mathbb{R}_+^{n \times d}$ and $H \in \mathbb{R}_+^{d \times m}$ are stochastic matrices. Define an LMC $\mathcal{M}' = (d{+}2, \Sigma, \mu', \pi')$ where the states are $\{0, 1, \ldots, d, d{+}1\}$, the initial distribution is $\pi' = e_0$, and the function $\mu'$, for all $i \in [m]$, $j \in [n]$, and $l \in [d]$, is defined by:

$$\mu'(a_i)_{0,l} = \frac{1}{m} H_{l,i}, \qquad \mu'(b_j)_{l,d+1} = W_{j,l}, \qquad \mu'(\checkmark)_{d+1,d+1} = 1,$$

and all other entries of $\mu'(a_i)$, $\mu'(b_j)$, and $\mu'(\checkmark)$ are 0. See Figure 8.2 for an example. For every $i \in [m]$, $j \in [n]$, and $p \in \mathbb{N}_0$, we have:

$$\|\mathcal{M}'\|(a_i b_j \checkmark^p) = e_0 \cdot \mu'(a_i) \cdot \mu'(b_j) \cdot \mu'(\checkmark) \cdot \mathbf{1} = \sum_{l \in [d]} \frac{1}{m} H_{l,i} \cdot W_{j,l} = \frac{1}{m} M_{j,i} = \|\mathcal{M}\|(a_i b_j \checkmark^p),$$

$$\|\mathcal{M}'\|(a_i) = e_0 \cdot \mu'(a_i) \cdot \mathbf{1} = \sum_{l \in [d]} \frac{1}{m} H_{l,i} = \frac{1}{m} = \|\mathcal{M}\|(a_i), \quad \text{and}$$

$$\|\mathcal{M}'\|(\varepsilon) = 1 = \|\mathcal{M}\|(\varepsilon).$$

Furthermore, for every $w \in \Sigma^* \setminus \{ \varepsilon, a_i, a_i b_j \checkmark^p \mid i \in [m], j \in [n], p \in \mathbb{N}_0 \}$ we have $\|\mathcal{M}'\|(w) = 0 = \|\mathcal{M}\|(w)$. Hence, $\mathcal{M}'$ is equivalent to $\mathcal{M}$. This proves (i).

To prove (ii), suppose that $\mathcal{M}$ has an equivalent LMC $\mathcal{M}' = (d + 2, \Sigma, \mu', \pi')$. Without loss of generality, let the states of $\mathcal{M}'$ be $\{0, 1, \ldots, d, d + 1\}$ with $\pi' = e_0$.

Let us define a matrix $H \in \mathbb{R}_+^{\{0,1,\ldots,d,d+1\}\times[m]}$ where $(H_{:,i})^\top = m \cdot \mu'(a_i)_{0,:}$ for all $i \in [m]$, and a matrix $W \in \mathbb{R}_+^{[n]\times\{0,1,\ldots,d,d+1\}}$ where $(W_{j,:})^\top = \mu'(b_j) \cdot \mathbf{1}$ for all $j \in [n]$. Since $\mathcal{M}$ and $\mathcal{M}'$ are equivalent, for every $i \in [m]$ and $j \in [n]$ we have

$$\frac{1}{m} M_{j,i} = \|\mathcal{M}\|(a_i b_j) = \|\mathcal{M}'\|(a_i b_j) = \mu'(a_i)_{0,:} \cdot \mu'(b_j) \cdot \mathbf{1} = \frac{1}{m} (H_{:,i})^\top \cdot (W_{j,:})^\top.$$

This implies that $M = W \cdot H$. Assuming $M$ is a nonzero matrix, there exist $i \in [m]$ and $j \in [n]$ such that $\frac{1}{m} M_{j,i} = \mu'(a_i)_{0,:} \cdot \mu'(b_j) \cdot \mathbf{1} > 0$. Since matrices $\mu'(a_i)$, $\mu'(b_j)$ are nonnegative, there exist $k, k' \in \{0, 1, \ldots, d, d+1\}$ such that $\mu'(a_i)_{0,k} \cdot \mu'(b_j)_{k,k'} \cdot 1 > 0$. Without loss of generality, we may assume that $k' = d + 1$. That is,

$$\mu'(a_i)_{0,k} \cdot \mu'(b_j)_{k,d+1} > 0. \tag{8.8}$$

The following lemma shows that the $(d+2)$-dimensional NMF $M = W \cdot H$ is, essentially, $d$-dimensional since one can "ignore" the first and last row of $W$ and column of $H$.

**Lemma 77.** *It holds that* $W \cdot H = W_{[n],[d]} \cdot H_{[d],[m]}$.

*Proof.* Take any $l_1 \in [n]$ and $l_2 \in [m]$. We have

$$(W \cdot H)_{l_1,l_2} = \sum_{l=0}^{d+1} W_{l_1,l} \cdot H_{l,l_2} = W_{l_1,0} \cdot H_{0,l_2} + \left(W_{[n],[d]} \cdot H_{[d],[m]}\right)_{l_1,l_2} + W_{l_1,d+1} \cdot H_{d+1,l_2}.$$

It now suffices to show that $W_{l_1,0} \cdot H_{0,l_2} + W_{l_1,d+1} \cdot H_{d+1,l_2} = 0$, since this would imply that $(W \cdot H)_{l_1,l_2} = (W_{[n],[d]} \cdot H_{[d],[m]})_{l_1,l_2}$ as required. To this end, in the following we show that $W_{l_1,d+1} = 0$ and $H_{0,l_2} = 0$.

Towards a contradiction, suppose that $W_{l_1,d+1} = \mu'(b_{l_1})_{d+1,:} \cdot \mathbf{1} > 0$. By (8.8) we now have

$$\|\mathcal{M}'\|(a_i b_j b_{l_1}) \geq \mu'(a_i)_{0,k} \cdot \mu'(b_j)_{k,d+1} \cdot \mu'(b_{l_1})_{d+1,:} \cdot \mathbf{1} > 0,$$

which is a contradiction with $\mathcal{M}'$ being equivalent to $\mathcal{M}$ since $\|\mathcal{M}\|(a_i b_j b_{l_1}) = 0$. We conclude that $W_{l_1,d+1} = 0$.

Towards a contradiction, suppose that $H_{0,l_2} = m \cdot \mu'(a_{l_2})_{0,0} > 0$. By (8.8), this implies that

$$\|\mathcal{M}'\|(a_{l_2} a_i b_j) \geq \mu'(a_{l_2})_{0,0} \cdot \mu'(a_i)_{0,k} \cdot \mu'(b_j)_{k,d+1} > 0,$$

which is a contradiction since $\|\mathcal{M}\|(a_{l_2} a_i b_j) = 0$. We conclude that $H_{0,l_2} = 0$. □

By Lemma 77, we now have

$$M = W \cdot H = W_{[n],[d]} \cdot H_{[d],[m]},$$

which completes the proof of (ii). □

We are now ready to give a negative answer to the rationality of LMC minimization question from the beginning of this section:

**Corollary 78.** *There exists an LMC with rational transition probabilities such that there is no minimal equivalent LMC with rational transition probabilities.*

*Proof.* Let $M \in \mathbb{Q}_+^{6\times 11}$ be the matrix from Theorem 67. Since $\text{rank}_+(M) = 5$, by Proposition 76 there exists an LMC $\mathcal{M} = (13, \Sigma, \mu, \pi)$ such that any minimal LMC equivalent to $\mathcal{M}$ has 7 states. Towards a contradiction, assume there is an LMC $\mathcal{M}' = (7, \Sigma, \mu', \pi')$ equivalent to $\mathcal{M}$ with rational transition probabilities. Then by Proposition 76, $\mathcal{M}'$ determines a 5-dimensional rational NMF $M = W \cdot H$. This is a contradiction since the nonnegative rank of $M$ over $\mathbb{Q}$ is 6 by Theorem 67. □

**Remark 79.** *Note that we do not need the fact that the reduction in Proposition 76 is polynomial time to prove Corollary 78.*

**Remark 80.** *Even though any minimal LMC equivalent to the LMC $\mathcal{M}$ from the proof of Corollary 78 has irrational transition probabilities, it generates every word with a rational probability because it is equivalent to $\mathcal{M}$. To understand this intuitively, observe that, for any word $w$, the probability $\|\mathcal{M}\|(w) \in \mathbb{Q}$ is equal to the sum of the (possibly irrational) probabilities of the paths consistent with $w$.*

As an immediate corollary of Proposition 76, one can relate the computational complexity of the NMF problem and the following *LMC minimization problem*: Given an LMC $\mathcal{M}$ with rational transition probabilities and $d \in \mathbb{N}$, does there exist an LMC $\mathcal{M}'$ with $d$ states such that $\mathcal{M}$ and $\mathcal{M}'$ are equivalent?

**Corollary 81.** *There is a polynomial-time reduction from the NMF problem to the LMC minimization problem.*

This NMF-hardness result can be adapted easily to (reactive) probabilistic automata [Rabin, 1963]. Since the NMF problem is NP-hard [Vavasis, 2009], minimizing both probabilistic automata and LMCs is NP-hard. In contrast, minimization of weighted automata (which may have negative transition weights) can be done efficiently over $\mathbb{Q}$ [Schützenberger, 1961]. Hence, the nonnegativity constraint makes minimization fundamentally more difficult.

## 8.5 Conclusion

In this chapter, we have developed connections between nonnegative matrix factorization and minimization of labelled Markov chains. We have considered two minimization problems: the problem of finding a minimal covering labelled Markov chain and the problem of finding a minimal equivalent (initialised) labelled Markov chain. We have reduced NMF to each of these minimization problems.

Using the result that restricted nonnegative rank is in general greater than nonnegative rank, we have shown that a minimal covering labelled Markov chain may

have a larger rank than the input labelled Markov chain. We have thus given a negative answer to Paz's conjecture from his seminal 1971 textbook. Moreover, we have falsified a positive answer to Paz's conjecture that was claimed in 1974.

Using our counterexample to the Cohen-Rothblum problem from Chapter 7, we have shown that there exists a labelled Markov chain with rational transition probabilities that has no minimal equivalent labelled Markov chain with rational transition probabilities. We have thus given a negative answer to the rationality of LMC minimization question.

# Part III

# Conclusion and Future Work

# Chapter 9

# Conclusion

In this thesis, we have considered multiplicity-automaton learning and minimization, matrix factorization, and the relationships between these areas.

In the first part of the thesis we looked at learning and minimization of multiplicity automata, which are a powerful algebraic framework that can represent a large class of distributions over words and trees. With regard to learning, we have looked at both active and passive learning frameworks, and have considered both query and computational complexity bounds. With regard to minimization, we have considered multiplicity automata over both words and trees, and have looked at both function and decision minimization problems. For both learning and minimization, we have given upper and lower complexity bounds and have shown relationships to well-known problems in complexity theory, namely the polynomial identity testing problem and Hilbert's Tenth Problem.

In the second part of the thesis we studied nonnegative matrix factorization (NMF), the problem of decomposing a given nonnegative $n \times m$ matrix $M$ into a product of a nonnegative $n \times d$ matrix $W$ and a nonnegative $d \times m$ matrix $H$. An NMF of $M$ is considered optimal when $d$ is minimal. Despite being a widely used dimension-reduction and feature-extraction technique in practice, there are a number

of open questions about the theory of NMF. In this thesis we have tackled a major open problem posed in 1993 by Cohen and Rothblum: Given a nonnegative rational matrix $M$ as input, can one always find an optimal nonnegative factorization of $M$ into rational matrices? We have answered this question negatively, by exhibiting a rank-4 rational matrix $M$ for which $W$ and $H$ require irrational entries.

We have also answered a variant of the Cohen–Rothblum problem for restricted NMF, where the column spaces of $M$ and $W$ are required to coincide, negatively by exhibiting a rank-4 rational matrix whose optimal restricted NMF requires irrational entries. This counterexample is optimal inasmuch as rational matrices of rank 3 or less always have an optimal rational restricted NMF.

There are various connections between automaton minimization and matrix factorization. Indeed, our minimization algorithms for multiplicity automata rely on computing a forward-backward factorization of the Hankel matrix of the automaton, which yields an NMF when the automaton is probabilistic. We have explored this connection in Chapter 8, where we have reduced NMF to minimization of probabilistic automata and labelled Markov chains. We have then applied our insights on NMF to answer longstanding open problems on minimization of labelled Markov chains: Using the fact that restricted nonnegative rank in general differs from nonnegative rank and our solution to the Cohen–Rothblum problem, respectively, we have answered Paz's 1971 conjecture and the rationality of LMC minimization question negatively.

# Chapter 10

# Future Work

In this chapter, we outline some possible directions for future work that arise from the research presented in this thesis.

## Applications of Tree-Automaton Learning

Learning algorithms for multiplicity word automata have been successfully applied to learn many other important classes of functions. For instance, Beimel et al. [2000] apply their exact learning algorithm for multiplicity word automata to show exact learnability of certain classes of polynomials over both finite and infinite fields. They also prove learnability of disjoint DNF formulae and, more generally, disjoint unions of geometric boxes over finite domains. Similarly, Klivans and Shpilka [2006] give an approach to exact learning of algebraic branching programs and arithmetic circuits and formulae using rank bounds for Hankel matrices of polynomials in noncommuting variables.

Since multiplicity tree automata are strictly more expressive than multiplicity word automata, a natural direction for future work would be to try to apply our learning algorithm for multiplicity tree automata of Chapter 3 to derive new results on exact learning of other concept classes, such as probabilistic context-free gram-

mars [Sakakibara, 1990], propositional formulae, and polynomials.

## Other Automaton-Minimization Problems

In this thesis, and Chapter 4 in particular, we focused on analysing the complexity of minimizing multiplicity automata with respect to the number of states. A natural question is minimization with respect to the number of transitions. This is particularly pertinent to the case of multiplicity tree automata, where the number of transitions is potentially exponential in the number of states. This is a difficult question that does not appear to have been addressed in previous research.

## Nonnegative Rank of Hankel Matrices

Let $\mathcal{A}$ be a multiplicity automaton with $n$ states and let $H$ be its Hankel matrix. We have seen in Chapter 4 that $\mathcal{A}$ induces a forward-backward factorization of $H$ with inner dimension $n$, implying that $\operatorname{rank}(H) \leq n$. If $\mathcal{A}$ is a minimal multiplicity automaton, then it follows from Theorem 1 (for word automata) and Theorem 4 (for tree automata) that $\operatorname{rank}(H) = n$.

When $\mathcal{A}$ is a probabilistic automaton, the forward-backward factorization of $H$ is an NMF with inner dimension $n$, implying that $\operatorname{rank}_+(H) \leq n$. The following is a natural open question: If $\mathcal{A}$ is a minimal probabilistic automaton, is $\operatorname{rank}_+(H) = n$?

## Optimal Counterexample to the Cohen–Rothblum Problem

In Chapter 7 we solved the Cohen–Rothblum problem for matrices of rank 4 or greater, showing that their nonnegative ranks over $\mathbb{R}$ and over $\mathbb{Q}$ may differ. It was

already known to Cohen and Rothblum [1993] that nonnegative ranks over $\mathbb{R}$ and $\mathbb{Q}$ coincide for matrices of rank at most 2, while Kubjas et al. [2015] showed that this also holds for matrices of nonnegative rank (over $\mathbb{R}$) at most 3. The remaining open question is whether nonnegative ranks over $\mathbb{R}$ and over $\mathbb{Q}$ differ for rank-3 matrices whose nonnegative rank is at least 4.

## Computational Complexity of the NMF Problem

Recently, Shitov [2016b] claimed that the NMF problem is polynomial-time equivalent to the existential theory of the reals. A natural open question, related to our results in Chapter 7, is whether the problem of deciding if the nonnegative rank over $\mathbb{Q}$ is at most a given number, is equivalent to the problem of deciding the truth of existential sentences over $\mathbb{Q}$. The decidability of the latter problem, which is equivalent to Hilbert's Tenth Problem over $\mathbb{Q}$, is a longstanding open question [Poonen, 2003]. We recall that we have shown in Chapter 4 that Hilbert's Tenth Problem over $\mathbb{Q}$ is logspace equivalent to the minimal consistency problem for $\mathbb{Q}$-multiplicity word automata.

## Probabilistic Automaton Minimization and Nonnegative Matrix Factorization

In Chapter 8 we gave a reduction from NMF to labelled Markov chain (and probabilistic automaton) minimization. In the reverse direction, it would be interesting to investigate whether minimization of labelled Markov chains or probabilistic automata can be reduced to NMF, or be seen as an augmented version of NMF. At the very least, many of the central computational complexity problems for NMF mentioned in this thesis have natural analogues for probabilistic automata. For example, minimiza-

tion of probabilistic word automata was shown to be NP-hard [Kiefer and Wachter, 2014] but, as with NMF, it is not known whether this problem lies in NP nor whether the corresponding problem for tree automata is even harder.

## Automaton Minimization and Matrix Factorization

Going beyond probabilistic automata, one could ask the following, more general question: Does minimization of arbitrary weighted automata reduce to matrix factorization? Some intuition here might be gained from our minimization algorithm for multiplicity automata from Chapter 4, which computes a factorization of a finite full-rank submatrix of the Hankel matrix of the input automaton.

# Bibliography

N. Abe and M. K. Warmuth. On the computational complexity of approximating distributions by probabilistic automata. *Machine Learning*, 9:205–260, 1992.

M. Abrahamsen, A. Adamaszek, and T. Miltzow. Irrational guards are sometimes needed. Technical report, arxiv.org, 2017. Available at `http://arxiv.org/abs/1701.05475`.

A. Aggarwal, H. Booth, J. O'Rourke, S. Suri, and C. K. Yap. Finding minimal convex nested polygons. *Information and Computation*, 83(1):98–110, 1989.

A. V. Aho, J. E. Hopcroft, and J. D. Ullman. *The Design and Analysis of Computer Algorithms*. Addison-Wesley, Reading, MA, 1974.

P. Ailliot, C. Thompson, and P. Thomson. Space-time modelling of precipitation by using a hidden Markov model and censored Gaussian distributions. *Journal of the Royal Statistical Society*, 58(3):405–426, 2009.

J. Albert and J. Kari. Digital image compression. In *Handbook of Weighted Automata*, pages 453–479. Springer, 2009.

E. Allender, P. Bürgisser, J. Kjeldgaard-Pedersen, and P. B. Miltersen. On the complexity of numerical analysis. *SIAM Journal on Computing*, 38(5):1987–2006, 2009.

S. Anantharaman, P. Narendran, and M. Rusinowitch. Closure properties and decision problems of DAG automata. *Information Processing Letters*, 94(5):231–240, 2005.

D. Angluin. Learning regular sets from queries and counterexamples. *Information and Computation*, 75(2):87–106, 1987.

D. Angluin. Queries and concept learning. *Machine Learning*, 2(4):319–342, 1988.

S. Arora and B. Barak. *Computational Complexity: A Modern Approach*. Cambridge University Press, New York, NY, 2009.

S. Arora, R. Ge, R. Kannan, and A. Moitra. Computing a nonnegative matrix factorization - provably. In *Proceedings of the 44th Symposium on Theory of Computing (STOC)*, pages 145–162, 2012a.

S. Arora, R. Ge, and A. Moitra. Learning topic models - going beyond SVD. In *53rd Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 1–10, 2012b.

R. Bailly, F. Denis, and L. Ralaivola. Grammatical inference as a principal component analysis problem. In *Proceedings of the 26th International Conference on Machine Learning (ICML)*, pages 33–40, 2009.

R. Bailly, A. Habrard, and F. Denis. A spectral approach for probabilistic grammatical inference on trees. In *Proceedings of the 21st International Conference on Algorithmic Learning Theory (ALT)*, volume 6331 of *LNCS*, pages 74–88, 2010.

B. Balle and M. Mohri. Spectral learning of general weighted automata via constrained matrix completion. In *Proceedings of the 26th Annual Conference on Neural Information Processing Systems (NIPS)*, pages 2168–2176, 2012.

B. Balle, P. Panangaden, and D. Precup. A canonical form for weighted automata and applications to approximate minimization. In *Proceedings of the 30th Annual ACM/IEEE Symposium on Logic in Computer Science (LICS)*, pages 701–712, 2015.

F. Bancilhon. A geometric model for stochastic automata. *IEEE Trans. Computers*, 23(12):1290–1299, 1974.

A. Beimel, F. Bergadano, N. H. Bshouty, E. Kushilevitz, and S. Varricchio. Learning functions represented as multiplicity automata. *Journal of the ACM*, 47(3):506–530, 2000.

M. W. Berry, N. Gillis, and F. Glineur. Document classification using nonnegative matrix factorization and underapproximation. In *International Symposium on Circuits and Systems (ISCAS)*, pages 2782–2785. IEEE, 2009.

J. Berstel and C. Reutenauer. Recognizable formal power series on trees. *Theoretical Computer Science*, 18(2):115–148, 1982.

L. Bisht, N. H. Bshouty, and H. Mazzawi. On optimal learning algorithms for multiplicity automata. In *Proceedings of the 19th Annual Conference on Learning Theory (COLT)*, volume 4005 of *LNCS*, pages 184–198. Springer, 2006.

B. Borchardt. A pumping lemma and decidability problems for recognizable tree series. *Acta Cybernetica*, 16(4):509–544, 2004.

S. Bozapalidis. Effective construction of the syntactic algebra of a recognizable series on trees. *Acta Informatica*, 28(4):351–363, 1991.

S. Bozapalidis and A. Alexandrakis. Représentations matricielles des séries d'arbre reconnaissables. *Informatique Théorique et Applications (ITA)*, 23(4):449–459, 1989.

S. Bozapalidis and O. Louscou-Bozapalidou. The rank of a formal tree power series. *Theoretical Computer Science*, 27(1):211–215, 1983.

W. S. Brainerd. The minimalization of tree automata. *Information and Control*, 13 (5):484–491, 1968.

S. S. Bucak and B. Günsel. Video content representation by incremental non-negative matrix factorization. In *Proceedings of the International Conference on Image Processing (ICIP)*, pages 113–116. IEEE, 2007.

P. Buneman, M. Grohe, and C. Koch. Path queries on compressed XML. In *Proceedings of the 29th International Conference on Very Large Data Bases (VLDB)*, pages 141–152, 2003.

J. Canny. Some algebraic and geometric computations in PSPACE. In *Proceedings of the 20th Annual ACM Symposium on Theory of Computing (STOC)*, pages 460–467, 1988.

J. W. Carlyle and A. Paz. Realizations by stochastic finite automata. *Journal of Computer and System Sciences*, 5(1):26–40, 1971.

J. Carme, R. Gilleron, A. Lemay, A. Terlutte, and M. Tommasi. Residual finite tree automata. In *Proceedings of the 7th International Conference on Developments in Language Theory (DLT)*, pages 171–182, 2003.

R. Carrasco, J. Daciuk, and M. Forcada. An implementation of deterministic tree automata minimization. In *Proceedings of the 12th International Conference on Implementation and Application of Automata (CIAA)*, pages 122–129, 2007.

W. Charatonik. Automata on DAG representations of finite trees. Research Report MPI-I-1999-2-001, Max-Planck-Institut für Informatik, Saarbrücken, 1999.

F.-S. Chen, C.-M. Fu, and C.-L. Huang. Hand gesture recognition using a real-time tracking method and hidden Markov models. *Image and Vision Computing*, 21(8): 745–758, 2003.

D. Chistikov, S. Kiefer, I. Marušić, M. Shirmohammadi, and J. Worrell. Nonnegative matrix factorization requires irrationality. *SIAM Journal on Applied Algebra and Geometry*. To appear; available at `http://arxiv.org/abs/1605.06848`.

D. Chistikov, S. Kiefer, I. Marušić, M. Shirmohammadi, and J. Worrell. On restricted nonnegative matrix factorization. In *Proceedings of the 43rd International Colloquium on Automata, Languages, and Programming (ICALP)*, pages 103:1–103:14, 2016.

D. Chistikov, S. Kiefer, I. Marušić, M. Shirmohammadi, and J. Worrell. On rationality of nonnegative matrix factorization. In *Proceedings of the 28th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 1290–1305, 2017.

S. Cho and D. T. Huynh. The parallel complexity of finite-state automata problems. *Information and Computation*, 97(1):1–22, 1992.

A. Cichocki, R. Zdunek, A. H. Phan, and S.-i. Amari. *Nonnegative Matrix and Tensor Factorizations: Applications to Exploratory Multi-way Data Analysis and Blind Source Separation*. Wiley Publishing, 2009. ISBN 0470746661, 9780470746660.

H. Cohen. *A Course in Computational Algebraic Number Theory*. Springer-Verlag, Berlin, 1993.

J. E. Cohen and U. G. Rothblum. Nonnegative ranks, decompositions, and factorizations of nonnegative matrices. *Linear Algebra and its Applications*, 190:149–168, 1993.

H. Comon, M. Dauchet, R. Gilleron, C. Löding, F. Jacquemard, D. Lugiez, S. Tison, and M. Tommasi. Tree automata techniques and applications. Available at `http://tata.gforge.inria.fr/`, 2007.

S. A. Cook. A taxonomy of problems with fast parallel algorithms. *Information and Control*, 64(1-3):2–22, 1985.

C. Cortes, M. Mohri, and A. Rastogi. On the computation of some standard distances between probabilistic automata. In *Proceedings of the 11th International Conference on Implementation and Application of Automata (CIAA)*, volume 4094 of *LNCS*, pages 137–149. Springer, 2006.

M. S. Crouse, R. D. Nowak, and R. G. Baraniuk. Wavelet-based statistical signal processing using hidden Markov models. *IEEE Transactions on Signal Processing*, 46(4):886–902, April 1998.

L. Csanky. Fast parallel matrix inversion algorithms. *SIAM Journal on Computing*, 5(4):618–623, 1976.

R. A. DeMillo and R. J. Lipton. A probabilistic remark on algebraic program testing. *Information Processing Letters*, 7(4):193–195, 1978.

F. Denis and A. Habrard. Learning rational stochastic tree languages. In *Proceedings of the 18th International Conference on Algorithmic Learning Theory (ALT)*, volume 4754 of *LNAI*, pages 242–256, 2007.

F. Denis, M. Gybels, and A. Habrard. Dimension-free concentration bounds on Hankel matrices for spectral learning. In *Proceedings of the 31st International Conference on Machine Learning (ICML)*, pages 449–457, 2014.

D. L. Donoho and V. Stodden. When does non-negative matrix factorization give a correct decomposition into parts? In *Advances in Neural Information Processing Systems (NIPS)*, pages 1141–1148, 2003.

F. Drewes and J. Högberg. Query learning of regular tree languages: How to avoid dead states. *Theory of Computing Systems*, 40(2):163–185, 2007.

F. Drewes and H. Vogler. Learning deterministically recognizable tree series. *Journal of Automata, Languages and Combinatorics*, 12(3):332–354, 2007.

F. Drewes, J. Högberg, and A. Maletti. MAT learners for tree series: an abstract data type and two realizations. *Acta Informatica*, 48(3):165–189, 2011.

R. Durbin. *Biological Sequence Analysis: Probabilistic Models of Proteins and Nucleic Acids*. Cambridge University Press, 1998.

S. R. Eddy. What is a hidden Markov model? *Nature Biotechnology*, 22(10):1315–1316, October 2004.

J. Eisner. Simpler and more general minimization for weighted finite-state automata. In *Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology (NAACL HLT)*, volume 1, pages 64–71, 2003.

K. Etessami and M. Yannakakis. Recursive concurrent stochastic games. *Logical Methods in Computer Science*, 4(4), 2008.

K. Etessami and M. Yannakakis. On the complexity of Nash equilibria and other fixed points. *SIAM J. Comput.*, 39(6):2531–2597, 2010.

L. Feng, T. Han, M. Z. Kwiatkowska, and D. Parker. Learning-based compositional verification for synchronous probabilistic systems. In *Proceedings of the 9th International Symposium on Automated Technology for Verification and Analysis (ATVA)*, pages 511–521, 2011.

N. Fijalkow, S. Kiefer, and M. Shirmohammadi. Trace refinement in labelled Markov decision processes. In *Proceedings of the 19th International Conference on Foundations of Software Science and Computation Structures (FoSSaCS)*, volume 9634 of *Lecture Notes in Computer Science*, pages 303–318. Springer, 2016.

B. Fila and S. Anantharaman. Automata for analyzing and querying compressed documents. Research Report RR-2006-03, Laboratoire d'Informatique Fondamentale d'Orléans (LIFO), Université d'Orléans, France, 2006.

M. Fliess. Matrices de Hankel. *Journal de Mathématiques Pures et Appliquées*, 53 (9):197–222, 1974.

M. Frick, M. Grohe, and C. Koch. Query evaluation on compressed trees (extended abstract). In *Proceedings of the 18th Annual IEEE Symposium on Logic in Computer Science (LICS)*, pages 188–197, 2003.

N. Gillis. Sparse and unique nonnegative matrix factorization through data preprocessing. *Journal of Machine Learning Research*, 13:3349–3386, 2012. URL `http://dl.acm.org/citation.cfm?id=2503349`.

N. Gillis. The why and how of nonnegative matrix factorization. In M. Signoretto J.A.K. Suykens and A. Argyriou, editors, *Regularization, Optimization, Kernels, and Support Vector Machines*, Machine Learning and Pattern Recognition Series, pages 257–291. Chapman & Hall/CRC, 2014. Available at `http://arxiv.org/abs/1401.5226`.

N. Gillis and F. Glineur. On the geometric interpretation of the nonnegative rank. *Linear Algebra and its Applications*, 437(11):2685–2712, 2012.

N. Gillis and S. A. Vavasis. Semidefinite programming based preconditioning for more robust near-separable nonnegative matrix factorization. *SIAM Journal on Optimization*, 25(1):677–698, 2015.

D. Gillman and M. Sipser. Inference and minimization of hidden Markov chains. In *Proceedings of the 7th Annual ACM Conference on Computational Learning Theory (COLT)*, pages 147–158, 1994.

E. M. Gold. Complexity of automaton identification from given data. *Information and Control*, 37(3):302–320, 1978.

D. A. Gregory and N. J. Pullman. Semiring rank: Boolean rank and nonnegative rank factorizations. *J. Combin. Inform. System Sci.*, 8(3):223–233, 1983.

M. Gybels, F. Denis, and A. Habrard. Some improvements of the spectral learning approach for probabilistic grammatical inference. In *Proceedings of the 12th International Conference on Grammatical Inference (ICGI)*, pages 64–78, 2014.

A. Habrard and J. Oncina. Learning multiplicity tree automata. In *Proceedings of the 8th International Colloquium on Grammatical Inference: Algorithms and Applications (ICGI)*, volume 4201 of *LNCS*, pages 268–280. Springer, 2006.

D. Hsu, S. M. Kakade, and T. Zhang. A spectral algorithm for learning hidden Markov models. *Journal of Computer and System Sciences*, 78(5):1460–1480, 2012.

O. H. Ibarra, S. Moran, and L. E. Rosier. A note on the parallel complexity of computing the rank of order $n$ matrices. *Information Processing Letters*, 11(4/5): 162, 1980.

T. Jiang and B. Ravikumar. Minimal NFA problems are hard. *SIAM Journal on Computing*, 22(6):1117–1141, 1993.

A. Kasprzik. Four one-shot learners for regular tree languages and their polynomial characterizability. *Theoretical Computer Science*, 485:85–106, 2013.

M. J. Kearns and L. G. Valiant. Cryptographic limitations on learning Boolean formulae and finite automata. *Journal of the ACM*, 41(1):67–95, 1994.

M. J. Kearns and U. V. Vazirani. *An Introduction to Computational Learning Theory.* MIT Press, Cambridge, MA, 1994.

S. Kiefer and B. Wachter. Stability and complexity of minimising probabilistic automata. In *Proceedings of the 41st International Colloquium on Automata, Languages, and Programming (ICALP)*, volume 8573 of *LNCS*, pages 268–279. Springer, 2014.

S. Kiefer, I. Marušić, and J. Worrell. Minimisation of multiplicity tree automata. *Logical Methods in Computer Science.* To appear; available at `http://arxiv.org/abs/1410.5352`.

S. Kiefer, A. Murawski, J. Ouaknine, B. Wachter, and J. Worrell. On the complexity of equivalence and minimisation for $\mathbb{Q}$-weighted automata. *Logical Methods in Computer Science*, 9(1), 2013.

S. Kiefer, I. Marušić, and J. Worrell. Minimisation of multiplicity tree automata. In *Proceedings of the 18th International Conference on Foundations of Software Science and Computation Structures (FoSSaCS)*, pages 297–311, 2015.

A. R. Klivans and A. Shpilka. Learning restricted models of arithmetic circuits. *Theory of Computing*, 2(1):185–206, 2006.

J. Koenigsmann. Undecidability in number theory. In *Model Theory in Algebra, Analysis and Arithmetic*, pages 159–195. Springer, 2014.

K. Kubjas, E. Robeva, and B. Sturmfels. Fixed points of the EM algorithm and nonnegative rank boundaries. *The Annals of Statistics*, 43(1):422–461, 2015.

A. Kumar, V. Sindhwani, and P. Kambadur. Fast conical hull algorithms for near-separable non-negative matrix factorization. In *Proceedings of the 30th International Conference on Machine Learning (ICML)*, page 231239, 2013.

H. Laurberg, M. G. Christensen, M. D. Plumbley, L. K. Hansen, and S. H. Jensen. Theorems on positive data: On the uniqueness of NMF. *Comp. Int. and Neurosc.*, 2008, 2008. doi: 10.1155/2008/764206. URL `http://dx.doi.org/10.1155/2008/764206`.

N. Lauritzen. Lectures on convex sets. *Aarhus University. Available online at http://home. imf. au. dk/niels/lecconset. pdf*, 2009.

D. D. Lee and H. S. Seung. Learning the parts of objects by non-negative matrix factorization. *Nature*, 401(6755):788–791, 1999.

A. Maletti. Learning deterministically recognizable tree series - revisited. In *Proceedings of the 2nd International Conference on Algebraic Informatics (CAI)*, volume 4728 of *LNCS*, pages 218–235, 2007.

A. Maletti. Minimizing deterministic weighted tree automata. *Information and Computation*, 207(11):1284–1299, 2009.

I. Marušić and J. Worrell. Complexity of equivalence and learning for multiplicity tree automata. In *Proceedings of the 39th International Symposium on Mathematical Foundations of Computer Science (MFCS), Part I*, volume 8634 of *LNCS*, pages 414–425. Springer, 2014.

I. Marušić and J. Worrell. Complexity of equivalence and learning for multiplicity tree automata. *Journal of Machine Learning Research*, 16:2465–2500, 2015.

M. Mohri, F. Pereira, and M. Riley. Weighted automata in text and speech processing. In *European Conference on Artificial Intelligence (ECAI), Workshop on Extended Finite State Models of Language*, 1996.

M. Mohri, A. Rostamizadeh, and A. Talwalkar. *Foundations of Machine Learning*. MIT press, 2012.

A. Moitra. An almost optimal algorithm for computing nonnegative rank. *SIAM Journal on Computing*, 45(1):156–173, 2016.

A. Paz. *Introduction to Probabilistic Automata*. Academic Press, 1971.

B. Poonen. Hilbert's tenth problem over rings of number-theoretic interest. *Note from the lecture at the Arizona Winter School on "Number Theory and Logic"*, 2003.

M. O. Rabin. Probabilistic automata. *Information and Control*, 6(3):230–245, 1963.

L. R. Rabiner. A tutorial on hidden Markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286, 1989.

G. Rabusseau, B. Balle, and S. B. Cohen. Low-rank approximation of weighted tree automata. In *Proceedings of the 19th International Conference on Artificial Intelligence and Statistics (AISTATS)*, pages 839–847, 2016.

J. Renegar. On the computational complexity and geometry of the first-order theory of the reals. Parts I–III. *Journal of Symbolic Computation*, 13(3):255–352, 1992.

Y. Sakakibara. Learning context-free grammars from structural data in polynomial time. *Theoretical Computer Science*, 76(2-3):223–242, 1990.

M. P. Schützenberger. On the definition of a family of automata. *Information and Control*, 4(2-3):245–270, 1961.

J. T. Schwartz. Fast probabilistic algorithms for verification of polynomial identities. *Journal of the ACM*, 27(4):701–717, 1980.

H. Seidl. Deciding equivalence of finite tree automata. *SIAM Journal on Computing*, 19(3):424–437, 1990.

L. S. Shapley. Stochastic games. *Proceedings of the national academy of sciences*, 39 (10):1095–1100, 1953.

Y. Shitov. Nonnegative rank depends on the field. Technical report, arxiv.org, 2015. Available at `http://arxiv.org/abs/1505.01893`.

Y. Shitov. Nonnegative rank depends on the field II. Technical report, arxiv.org, 2016a. Available at `http://arxiv.org/abs/1605.07173`.

Y. Shitov. A universality theorem for nonnegative matrix factorizations. Technical report, arxiv.org, 2016b. Available at `http://arxiv.org/abs/1606.09068`.

D. Simanek. How to view 3D without glasses. `https://www.lhup.edu/~dsimanek/3d/view3d.htm`. Online, accessed in April 2016.

A. Sperduti and A. Starita. Supervised neural networks for the classification of structures. *IEEE Transactions on Neural Networks*, 8(3):714–735, 1997.

L. B. Thomas. Rank factorization of nonnegative matrices (by A. Berman and R.J. Plemmons). *SIAM Rev.*, 16(3):393–394, 1974.

E. Tjioe, M. W. Berry, and R. Homayouni. Discovering gene functional relationships using FAUN (feature annotation using nonnegative matrix factorization). *BMC Bioinformatics*, 11(S-6):S14, 2010.

W.-G. Tzeng. A polynomial-time algorithm for the equivalence of probabilistic automata. *SIAM Journal on Computing*, 21(2):216–227, 1992.

S. A. Vavasis. On the complexity of nonnegative matrix factorization. *SIAM Journal on Optimization*, 20(3):1364–1377, 2009.

S. Venkatasubramanian. Computational geometry column 55: New developments in nonnegative matrix factorization. *SIGACT News*, 44(1):70–78, 2013.

N. Vlassis, M. L. Littman, and D. Barber. On the computational complexity of stochastic controller optimization in POMDPs. *TOCT*, 4(4):12, 2012.

M. Yannakakis. Expressing combinatorial optimization problems by linear programs. *J. Comput. Syst. Sci.*, 43(3):441–466, 1991.

T. Yokota, R. Zdunek, A. Cichocki, and Y. Yamashita. Smooth nonnegative matrix and tensor factorizations for robust multi-way data analysis. *Signal Processing*, 113:234–249, 2015.

S. Zhang, W. Wang, J. Ford, and F. Makedon. Learning from incomplete ratings using non-negative matrix factorization. In *Proceedings of the 6th SIAM International Conference on Data Mining*, pages 549–553. SIAM, 2006.

R. E. Zippel. Probabilistic algorithms for sparse polynominals. In *Proceedings of the International Symposium on Symbolic and Algebraic Computation (EUROSAM)*, volume 72 of *LNCS*, pages 216–226. Springer, 1979.