Reconstruction of Exponential Sums from Fourier Data via Rational Approximation

Dissertation

zur Erlangung des mathematisch-naturwissenschaftlichen Doktorgrades

Doctor rerum naturalium

der Georg-August-Universität Göttingen

im Promotionsprogramm *Mathematical Sciences*

der Georg-August University School of Science (GAUSS)

vorgelegt von

Markus Petz

aus Stegersbach

Göttingen, 2022

Betreuungsausschuss

Prof. Dr. Gerlind Plonka-Hoch
Institut für Numerische und Angewandte Mathematik, Georg-August-Universität Göttingen

Prof. Dr. Stephan Huckemann
Institut für Mathematische Stochastik, Georg-August-Universität Göttingen

Prof. Dr. Thorsten Hohage
Institut für Numerische und Angewandte Mathematik, Georg-August-Universität Göttingen

Mitglieder der Prüfungskommission

Referentin
Prof. Dr. Gerlind Plonka-Hoch
Institut für Numerische und Angewandte Mathematik, Georg-August-Universität Göttingen

Korreferent
Prof. Dr. Stefan Kunis
Institut für Mathematik, Universität Osnabrück

Weitere Mitglieder der Prüfungskommission

Prof. Dr. Thorsten Hohage
Institut für Numerische und Angewandte Mathematik, Georg-August-Universität Göttingen

PD Dr. Hartje Kriete
Mathematisches Institut, Georg-August-Universität Göttingen

Prof. Dr. Dominic Schuhmacher
Institut für Mathematische Stochastik, Georg-August-Universität Göttingen

Jun.-Prof. Dr. Anne Wald
Institut für Numerische und Angewandte Mathematik, Georg-August-Universität Göttingen

Tag der mündlichen Prüfung: 30. Juni 2022

Thank You

A lot of hard work went into this dissertation. However, it is not my work alone. Many people helped in its creation, be it directly or indirectly. To them I want to say thank you.

First and foremost I have to thank my supervisor Gerlind Plonka-Hoch. We have shared a long path since she supervised my Bachelor's thesis. I would like to thank her for giving me the opportunity to do a PhD. Without her and her support this project would not have been possible.

I am happy that Stefan Kunis agreed to be the second reviewer for this thesis. He invited me to give a talk in Osnabrück, which, in these pandemic times, provided a rare occasion to talk in person to other researchers about my work. For this I am thankful.

I would also like to thank the members of my thesis advisory committee, Stephan Huckemann and Thorsten Hohage. I very much appreciate that they always took the time to discuss my work and give me valuable feedback.

My sincere thanks go to Diana Sieber, our wonderful secretary. She keeps everything organized and always has an answer to every administrative question.

I was lucky enough to be part of the Mathematical Image and Signal Processing group at our institute. It is a group of kind, funny, understanding and helpful people. I enjoyed our lunch and ice cream breaks, just talking or joking around. But I equally value the mathematical discussions we had and the possibility to approach everyone with any questions. In particular, I am indebted to Nadiia Derevianko, Inge Keller, Hanna Knirsch, Benjamin Kocurov and Raha Razavi for proofreading parts of this thesis.

Inge and Hanna have not just been colleagues over the past years, but friends. Thank you for the countless times I could talk to you about all the small and large things on my mind, both with regard to mathematics and just about everything else.

Special thanks goes to Katharina Müller. Not only did she also proofread parts of this thesis, but, most importantly, despite being on another continent, she was a friend to me, in the truest sense of the word, at a time when I needed one more than ever before.

Lastly, I would like to thank my family, my parents Roswitha and Manfred and my sister Daniela. Thank you for your unconditional support, your kind words, your thoughts, your help, your advice. Thank you for being there when I need you and giving me space when I need it.

Contents

Notations

Sets and Spaces

$\mathbb{N}$	Natural numbers $\{1, 2, 3, \dots\}$
$\mathbb{N}_0$	Non-negative integers $\{0, 1, 2, \dots\}$
$\mathbb{Z}$	Integers
$\mathbb{Q}$	Rational numbers
$\mathbb{R}$	Real numbers
$\mathbb{C}$	Complex numbers.
$\lvert S\rvert$	Number of elements in the finite set S
$\mathbb{K}[x]$	Space of polynomials over a field $\mathbb{K}$
$C^r(T)$	Space of r-times continuously differentiable functions from $T \subseteq \mathbb{R}$ to $\mathbb{C}$

Matrices

$\mathbf{V}_{m,n}(\mathbf{x}, \mathbf{y})$	Vandermonde-type reconstruction matrix for a vector $\mathbf{x}$ of sample points and a vector $\mathbf{y}$ of sample values as in (1.3)
$\mathbf{V}_{m,n}(S)$	Vandermonde-type reconstruction matrix for a sample set S as in (1.7)
$\mathbf{V}_m(\mathbf{x})$	Vandermonde matrix for a vector $\mathbf{x} = (x_1, \dots, x_M) \in \mathbb{C}^M$ given by $\big(x_j^k\big)_{j=1,k=0}^{M,m}$
$\mathbf{L}(S', T')$	Löwner matrix with row array S' and column array T' as in (1.21)
$\operatorname{diag}(\mathbf{x})$	Diagonal matrix in $\mathbb{C}^{M\times M}$ with the entries of $\mathbf{x} \in \mathbb{C}^M$ on its diagonal
$\operatorname{rk}\mathbf{A}$	Rank of a matrix $\mathbf{A}$, i.e., the dimension of its image
$\operatorname{null}\mathbf{A}$	Nullity of a matrix $\mathbf{A}$, i.e., the dimension of its kernel

Other

$\|\mathbf{v}\|_0$	Number of non-zero entries in the vector $\mathbf{v}$
$\deg(p)$	Degree of a polynomial $p \in \mathbb{K}[x]$
$\deg(r)$	MacMillan degree of a rational function $r = \frac{p}{q}$ with $p, q \in \mathbb{K}[x]$ given by $\max\{\deg(p), \deg(q)\}$ as in (1.19)
$\gcd(a, b)$	Greatest common divisor of two elements a and b from a commutative ring, e.g., $\mathbb{Z}$ or $\mathbb{K}[x]$
$a \mid b$	"a divides b" for two elements a and b from a commutative ring
$f \equiv a$	Function which is constantly equal to a
$\lfloor a \rfloor$	Largest integer $n \leq a$
$\lceil a \rceil$	Smallest integer $n \geq a$
$r_{\mathbf{u},T'}$	Barycentric rational function built from the weight vector $\mathbf{u}$ and the sample set T' as in (1.22)
$\hat{f}_P$	P-periodic Fourier transform as in (4.2)
$\hat{\mathbf{f}} = (\hat{f}_n)_{n=0}^{M-1}$	Discrete Fourier transform of a vector $\mathbf{f}$ as in (6.1)

Introduction

A Motivating Example

Let us begin this dissertation by looking at a few examples which motivate what we are about to do. Consider the function

$$f(t) = \mathrm{e}^{2\pi\,\mathrm{i}\cdot\sqrt{2}\cdot t} + \mathrm{e}^{2\pi\,\mathrm{i}\cdot\sqrt{5}\cdot t}\,.$$

Suppose we have measured f and know, that it is of the form

$$\sum_{k=1}^{K} \eta_k \cdot \mathrm{e}^{2\pi\,\mathrm{i}\cdot a_k\cdot t}, \tag{1}$$

but we do not know the parameters $K \in \mathbb{N}$, $\eta_k \in \mathbb{C} \setminus \{0\}$ and $a_k \in \mathbb{R}$ for $k = 1, 2, \ldots, K$. If we want to find these parameters, which is usually described as solving the *parameter identification problem*, then, knowing that f is of the form (1), a natural idea is to compute its Fourier coefficients. Suppose we have measured f over the interval $[0, 1]$, then its Fourier coefficients are given by

$$\hat{f}(n) = \int_0^1 f(t) \cdot \mathrm{e}^{-2\pi\,\mathrm{i}\cdot n\cdot t}\,\mathrm{d}t = \frac{1}{2\pi\,\mathrm{i}} \cdot \left(\frac{1 - \mathrm{e}^{2\pi\,\mathrm{i}\cdot\sqrt{2}}}{n - \sqrt{2}} + \frac{1 - \mathrm{e}^{2\pi\,\mathrm{i}\cdot\sqrt{5}}}{n - \sqrt{5}} \right)$$

for $n \in \mathbb{Z}$. In particular, none of the Fourier coefficients are 0 and they do not reveal the parameters $K = 2$, $\eta_1 = \eta_2 = 1$, $a_1 = \sqrt{2}$ and $a_2 = \sqrt{5}$. In other words, even though f is the sum of only two exponential terms, it corresponds to an infinite Fourier series. Why is that? As we will see in Chapter 3, the function f is not periodic. But for the Fourier series corresponding to a function to be finite, it is necessary that the function is periodic.

Now consider a different function, namely

$$g(t) = \mathrm{e}^{2\pi\,\mathrm{i}\cdot\frac{1}{2}\cdot t} + \mathrm{e}^{2\pi\,\mathrm{i}\cdot\frac{1}{5}\cdot t}\,.$$

Clearly, this function is periodic. However, again assuming we know that g is of the form (1) and we have measured it over the interval $[0,1]$, we find that its Fourier coefficients are

$$\hat{g}(n) = \frac{1}{2\pi\,\mathrm{i}} \cdot \left(\frac{2}{n-\frac{1}{2}} + \frac{1-\mathrm{e}^{2\pi\,\mathrm{i}\cdot\frac{1}{5}}}{n-\frac{1}{5}} \right)$$

for $n \in \mathbb{Z}$. So again, all of these coefficients are different from 0 and they do not reveal that $K=2$, $\eta_1 = \eta_2 = 1$, $a_1 = \frac{1}{2}$, and $a_2 = \frac{1}{5}$. What happened this time? In order to reconstruct a function of the form (1) using Fourier coefficients, it is not sufficient that the function is periodic, but we actually need to know its exact period and compute the Fourier coefficients using measurements of exactly the period length (or a multiple thereof). Here however, g has a period of 10, but we only used data from the interval $[0,1]$.

Let us consider one last example, namely

$$h(t) = \mathrm{e}^{2\pi\,\mathrm{i}\cdot 1\cdot t} + \mathrm{e}^{2\pi\,\mathrm{i}\cdot\frac{1}{2}\cdot t} + \mathrm{e}^{2\pi\,\mathrm{i}\cdot\frac{1}{5}\cdot t}.$$

We find that its Fourier coefficients over the interval $[0,1]$ are given by

$$\hat{h}(n) = \frac{1}{2\pi\,\mathrm{i}} \cdot \left(\frac{2}{n-\frac{1}{2}} + \frac{1-\mathrm{e}^{2\pi\,\mathrm{i}\cdot\frac{1}{5}}}{n-\frac{1}{5}} \right)$$

for $n \in \mathbb{Z} \setminus \{1\}$ and

$$\hat{h}(1) = 1 + \frac{1}{2\pi\,\mathrm{i}} \left(4 + \frac{1-\mathrm{e}^{2\pi\,\mathrm{i}\cdot\frac{1}{5}}}{\frac{4}{5}} \right),$$

i.e., for $n \in \mathbb{Z} \setminus \{1\}$ they are equal to the Fourier coefficients of g and for $n = 1$ they only differ by an added 1. This added 1 arises from the fact that h contains the 1-periodic component $1 \cdot \mathrm{e}^{2\pi\,\mathrm{i}\cdot 1\cdot t}$.

Looking more closely at $\hat{g}(n)$ we note that

$$\begin{aligned}
\hat{g}(n) &= \frac{1}{2\pi\,\mathrm{i}} \cdot \left(\frac{2}{n-\frac{1}{2}} + \frac{1-\mathrm{e}^{2\pi\,\mathrm{i}\cdot\frac{1}{5}}}{n-\frac{1}{5}} \right) \\
&= \frac{1}{2\pi\,\mathrm{i}} \cdot \frac{\left(n-\frac{1}{5}\right)\cdot 2 + \left(1-\mathrm{e}^{2\pi\,\mathrm{i}\cdot\frac{1}{5}}\right)\cdot\left(n-\frac{1}{2}\right)}{\left(n-\frac{1}{2}\right)\cdot\left(n-\frac{1}{5}\right)} \\
&= \frac{1}{2\pi\,\mathrm{i}} \cdot \frac{\left(3-\mathrm{e}^{2\pi\,\mathrm{i}\cdot\frac{1}{5}}\right)\cdot n - \left(\frac{9}{10}-\frac{1}{2}\cdot\mathrm{e}^{2\pi\,\mathrm{i}\cdot\frac{1}{5}}\right)}{n^2-\frac{7}{10}\cdot n+\frac{1}{10}} \\
&=: \frac{1}{2\pi\,\mathrm{i}} \cdot r(n),
\end{aligned}$$

where

$$r(x) = \frac{\left(3 - \mathrm{e}^{2\pi\,\mathrm{i}\cdot\frac{1}{5}}\right)\cdot x - \left(\frac{9}{10} - \frac{1}{2}\cdot \mathrm{e}^{2\pi\,\mathrm{i}\cdot\frac{1}{5}}\right)}{x^2 - \frac{7}{10}\cdot x + \frac{1}{10}}$$

is a rational function. Writing this differently, we see that

$$r(n) = 2\pi\,\mathrm{i}\cdot\hat{g}(n).$$

In other words, the rational function r interpolates the values $2\pi\,\mathrm{i}\cdot\hat{g}(n)$ at the points $n \in \mathbb{Z}$. As we will see, this observation holds true in the general case, i.e., for $P \in \mathbb{R}_{>0}$, the Fourier coefficients of a function of form (1), which does not contain P-periodic components, computed over an interval $[0, P]$, can always be described by a rational function r. Even more, we will see that the parameters K, η_k, and a_k for $k = 1, 2, \ldots, K$ can be reconstructed from r.

But what about $\hat{h}(1) = 1 + r(1)$? It will turn out that if a function contains P-periodic components, where here $P = 1$, there exists a rational function that interpolates all Fourier coefficients not corresponding to these components, while the Fourier coefficients corresponding to P-periodic components give rise to unattainable points.

These observations motivate why the first part of this thesis is dedicated to rational interpolation and, in particular, rational interpolation with unattainable points.

Instead of functions of the form (1) we can more generally consider so-called exponential sums, which are functions of the form

$$\sum_{k=1}^{K} \eta_k \cdot \mathrm{e}^{\phi_k\cdot t}, \tag{2}$$

with $\eta_k \in \mathbb{C}\setminus\{0\}$ and pairwise distinct $\phi_k \in \mathbb{C}$ for $k = 1, 2, \ldots, K$. We will see that the Fourier coefficients of exponential sums can also be described by rational functions.

Applications for the Reconstruction of Exponential Sums

Popular and well-known methods to reconstruct the parameters K, η_k, and ϕ_k for $k = 1, 2, \ldots, K$ of an exponential sum of form (2) are Prony's method [34] and so-called Prony-like methods [57, 99, 103]. In this dissertation, we develop an alternative algorithm to Prony's method, which, following the observations above, is based on rational interpolation of Fourier coefficients. Our algorithm can be used for the same applications as Prony's method.

Exponential sums, and in particular Prony's method as well as Prony-like methods, have been studied extensively in recent years, see for example [7, 41, 72, 96, 97, 111]. Reconstruction methods for exponential sums have been successfully applied in phase retrieval [4], signal approximation [16, 20], and sparse deconvolution in nondestructive testing [19]. The problem of parameter identification of exponential sums also appears in applications such as model reduction in system theory [59], direction of arrival estimation [65], exponential data fitting [88], image super-resolution [112], and reconstruction of signals with finite rate of innovation [7, 111]. For further literature on Prony's and Prony-like methods, as well as their generalizations, we refer to [63, 94] and [92, Chapter 10] and the literature referenced therein.

We will study the problem of recovering exponential sums only in the one-dimensional case. Over the past years, there has also been much interest in higher dimensional versions of Prony's and Prony-like methods, see for example [18, 31, 32, 40, 66, 79].

Organization of this Dissertation

In Chapter 1, we take an in-depth look at rational interpolation. Here we develop most of the theory that we will need throughout this thesis. The content of this chapter can be broken down into three main topics, classical rational interpolation, barycentric rational interpolation, and the AAA algorithm.

In the part about classical rational interpolation, we give some known results, where we put special emphasis on unattainable points. Since we are ultimately interested in reconstructing a rational function r from samples thereof, respectively polynomials p, q, and d such that $r = \frac{p}{q}$ and d describes the unattainable points in the given data set, our presentation of the material differs from the way rational interpolation is usually viewed. In order to reflect this slightly different view, we will often not talk about rational interpolation, but reconstruction of polynomials via rational interpolation.

Barycentric rational interpolation is known to be more stable than methods derived directly from the classical theory. In the corresponding sections, we give the first important result of this dissertation. Theorem 1.15 generalizes a well-known result, which shows how barycentric rational interpolation is connected with special Löwner matrices, to the case that the given data contains unattainable points. Since the classical view on rational interpolation is theoretically more straightforward, we will use it throughout this work to develop our theory. The barycentric approach is then used the develop a numerical algorithm.

The AAA algorithm [80] is an algorithm for rational approximation through successive barycentric rational interpolation. We later use it as part of our new algorithm and show that it always converges in our setting.

In Chapter 2, we consider numerical aspects of the AAA algorithm and barycentric rational interpolation in general. In particular, we develop a new stopping criterion for the AAA algorithm, which is specially tailored to our problem. Further, we numerically compare three different algorithms. One which is directly derived from the classical approach to rational interpolation, the AAA algorithm, and another algorithm based on barycentric rational interpolation. We study how well these algorithms perform depending on different properties of the input functions and optimize the AAA algorithm for our needs.

As noted in the introductory example, periodicity plays an important role when analyzing functions using the Fourier transform. In Chapter 3, we present several concepts, transitioning from classical trigonometric polynomials to exponential sums. This chapter is meant to give some background knowledge and develop an intuition for how the Fourier transform behaves for general exponential sums. At the end, we give a uniqueness proof for exponential sums, showing that they are uniquely defined by their parameters. This is needed in order to guarantee that the problem of reconstructing the function parameters has a unique solution.

In Chapter 4, we thoroughly develop the theory that we already hinted at in the motivating example. Many of the concepts we study will also come up in the same or a very similar form later on. The chapter closes by presenting Main Theorem 4.5, which is the basis of our algorithm for data obtained from the continuous Fourier transform on an interval of arbitrary length.

In Chapter 5, we present an adaptation of the algorithm developed in Chapter 4 for so-called real almost-periodic functions.

Chapter 6 is analogous to Chapter 4 in that the theory which was first established for classical Fourier coefficients is now developed for coefficients computed by the discrete Fourier transform. Many results are the same as in Chapter 4. However, special care is required when translating other results, in particular concerning aliasing. Going from classical Fourier coefficients to DFT coefficients allows us to apply our algorithm in cases where we are given discrete samples of the function we want to reconstruct. This makes our method comparable to Prony's method, which also uses discrete function samples, thereby allowing it to be used for the same applications as described above. At the end of this chapter, we give Main Theorem 6.9, which forms the theoretical foundation for our algorithm to reconstruct exponential sums from discrete function samples.

The dissertation ends by developing a numerically stable version of our algorithm, called ESPIRA, in Chapter 7. We then thoroughly test the algorithm and compare it to the Prony-like Matrix Pencil Method (MPM), which we present in detail in Appendix B. This is done both with exact and noisy data. Without studying this in detail, we also remark on our empirical observations with regard to the ill-posedness of the parameter identification problem, i.e., the problem of obtaining good reconstructions of the parameters K, η_k, and ϕ_k for $k = 1, 2, \ldots, K$, as opposed to just finding a good approximation for the function f.

Part I.

Reconstruction of Polynomials via Rational Interpolation

1. Rational Interpolation

1.1. Classical Rational Interpolation

In this first section, we present some theoretical aspects of rational interpolation. Much has been written about this subject. For general introductions see [23, 102, 107]. More extensive overviews can be found in [77, 114].

Let us begin by recalling the problem of *polynomial interpolation*. Given sets of distinct *sample points* (also called *interpolation points*, *nodes* or *support points*) $x_j \in \mathbb{C}$ and *sample values* $y_j \in \mathbb{C}$, or simply *samples* $(x_j, y_j) \in \mathbb{C}^2$, for $j = 1, 2, \dots, M \in \mathbb{N}$, find a polynomial $p \in \mathbb{C}[x]$ such that $p(x_j) = y_j$ for all $j = 1, 2, \dots, M$.

This problem has a unique solution given by

$$p(x) = \sum_{j=1}^{M} y_j \cdot \ell_j(x),$$

where

$$\ell_j(x) \coloneqq \prod_{\substack{i=1 \\ i \neq j}}^{M} \frac{x - x_i}{x_j - x_i}$$

is the so-called *Lagrange polynomial*, see for example [102, 107], since Lagrange described it in [68], even though he was not the first person to find this interpolation formula, see [76].

Now define the set

$$\mathcal{R}_{m,n} \coloneqq \left\{ \frac{p}{q} : p, q \in \mathbb{C}[x], \deg(p) \leq m, \deg(q) \leq n, q \not\equiv 0 \right\}$$

of *rational functions* with numerator degree at most m and denominator degree at most n. We also say that $\mathcal{R}_{m,n}$ is the set of rational functions of *type* (m, n). Then the problem of *rational interpolation* can be formulated as follows: Given m, n, and a set of $M \in \mathbb{N}$ samples $(x_j, y_j) \in \mathbb{C}^2$ where the sample points x_j are pairwise distinct for $j = 1, 2, \dots, M$, find a rational function $\frac{p}{q} \in \mathcal{R}_{m,n}$ such that

$$\frac{p(x_j)}{q(x_j)} = y_j \quad \text{for all} \quad j = 1, 2, \dots, M. \tag{1.1}$$

This problem goes back to Cauchy [24] and is therefore sometimes also referred to as the *Cauchy interpolation problem.*

A direct consequence of equation (1.1) is that

$$p(x_j) - y_j \cdot q(x_j) = 0 \quad \text{for all} \quad j = 1, 2, \dots, M, \tag{1.2}$$

which is often called the *linearized rational interpolation problem.* Usually, one solves the linearized problem in order to solve the rational interpolation problem. However, a solution of (1.2) does not in general entail a solution to (1.1).

If $q(x_k) = 0$ for some $k \in \{1, 2, \dots, M\}$, then equation (1.2) implies that also $p(x_k) = 0$ and hence (1.2) is satisfied for every choice of y_k. Let $d \coloneqq \gcd(p, q)$ and define $p^* \coloneqq \frac{p}{d}$ and $q^* = \frac{q}{d}$. Then we have

$$\frac{p}{q} = \frac{p^* \cdot d}{q^* \cdot d} = \frac{p^*}{q^*},$$

where p^* and q^* are coprime, i.e., they do not possess any common zeros. Then (1.2) entails (1.1) if and only if $y_j = \frac{p^*(x_j)}{q^*(x_j)}$ for all $j = 1, 2, \dots, M$. In particular, if p and q are coprime, then (1.2) always entails (1.1).

For certain choices of sample points $x_1, x_2, \dots, x_M$ and sample values $y_1, y_2, \dots, y_M$ it is then possible that there are pairs (x_k, y_k) for $1 \leq k \leq M$ such that $\frac{p(x_k)}{q(x_k)} = y_k$ cannot be satisfied. These are called *unattainable points.* Therefore, unlike the problem of polynomial interpolation, the problem of rational interpolation need not have a solution. Note, however, that in the rational interpolation problem, other than in the polynomial interpolation problem, we have specified a maximal degree for the polynomials p and q. Were we to allow for arbitrary degrees, then every solution to the polynomial interpolation problem would also be a solution to the rational interpolation problem with $q \equiv 1$.

Let $\mathbf{p} \coloneqq (p_0, p_1, \dots, p_m)^T \in \mathbb{C}^{m+1}$ and $\mathbf{q} \coloneqq (q_0, q_1, \dots, q_n)^T \in \mathbb{C}^{n+1}$ denote the coefficient vectors corresponding to the polynomials p and q, i.e.,

$$p(x) = \sum_{j=0}^{m} p_j \cdot x^j \qquad \text{and} \qquad q(x) = \sum_{j=0}^{n} q_j \cdot x^j,$$

where, if $\deg(p) < m$, we simply have $p_{\deg(p)+1} = \ldots = p_m = 0$ and analogously for q. Then (1.2) gives rise to the following system of linear equations:

$$\underbrace{\begin{pmatrix} 1 & x_1 & x_1^2 & \ldots & x_1^m & y_1 & y_1 \cdot x_1 & y_1 \cdot x_1^2 & \ldots & y_1 \cdot x_1^n \\ 1 & x_2 & x_2^2 & \ldots & x_2^m & y_2 & y_2 \cdot x_2 & y_2 \cdot x_2^2 & \ldots & y_2 \cdot x_2^n \\ \vdots & \vdots & \vdots & & \vdots & \vdots & \vdots & \vdots & & \vdots \\ 1 & x_M & x_M^2 & \ldots & x_M^m & y_M & y_M \cdot x_M & y_M \cdot x_M^2 & \ldots & y_M \cdot x_M^n \end{pmatrix}}_{=:\mathbf{V}_{m,n}(\mathbf{x},\mathbf{y}) \in \mathbb{C}^{M \times m+n+2}} \cdot \begin{pmatrix} \mathbf{p} \\ -\mathbf{q} \end{pmatrix} = \mathbf{0}, \tag{1.3}$$

where $\mathbf{x} := (x_1, x_2, \ldots, x_M)^T \in \mathbb{C}^M$ and $\mathbf{y} := (y_1, y_2, \ldots, y_M)^T \in \mathbb{C}^M$ denote the vectors of sample points and sample values. We will call $\mathbf{V}_{m,n}(\mathbf{x}, \mathbf{y})$ an *interpolation matrix*, later also *reconstruction matrix*. Note that equations (1.2) and (1.3) are equivalent formulations of the same problem. Whenever p and q are polynomials with $\deg(p) \leq m$ and $\deg(q) \leq n$ solving (1.2), their corresponding coefficient vectors solve (1.3) and vice versa. Therefore, we can always interchange the notion of $(\mathbf{p}^T, -\mathbf{q}^T)^T \in \ker \mathbf{V}_{m,n}(\mathbf{x}, \mathbf{y})$ and p and q solving (1.2).

In general, the system of equations (1.3) is only guaranteed to have a non-trivial solution if $M \leq m + n + 1$. However, in order to develop the theory we need, we will assume here that equation (1.2) holds regardless of M and, therefore, that there always exists a non-trivial solution to $\mathbf{V}_{m,n}(\mathbf{x}, \mathbf{y}) \cdot \mathbf{v} = \mathbf{0}$, i.e., one with $\mathbf{v} \neq \mathbf{0}$. Put differently, we assume that for the *nullity* of the matrix $\mathbf{V}_{m,n}(\mathbf{x}, \mathbf{y})$ we always have $\operatorname{null} \mathbf{V}_{m,n}(\mathbf{x}, \mathbf{y}) := \dim \ker \mathbf{V}_{m,n}(\mathbf{x}, \mathbf{y}) \geq 1$.

Moreover, we are interested in solutions to (1.2) where $p \not\equiv 0$, which means that we always want for $(\mathbf{p}^T, -\mathbf{q}^T)^T \in \ker \mathbf{V}_{m,n}(\mathbf{x}, \mathbf{y})$ that $\mathbf{p} \neq \mathbf{0}$. In the next lemma, we formulate a condition for this to hold, but first, we need to introduce some notation.

For $N \in \mathbb{N}$, let $\mathbf{v} = (v_1, v_2, \ldots, v_N)^T \in \mathbb{C}^N$ be a vector. Then we define $\|\mathbf{v}\|_0 := |\{j \in \{1, 2, \ldots, N\} : v_j \neq 0\}|$, i.e., $\|\mathbf{v}\|_0$ denotes the number of non-zero entries in $\mathbf{v}$. Although this is not a norm, it is often referred to as the 0-*norm*. Further, for a matrix $\mathbf{A} = (\mathbf{a}_1^T, \mathbf{a}_2^T, \ldots, \mathbf{a}_N^T)^T$ with rows $\mathbf{a}_j$ for $j = 1, 2, \ldots, N$ and a set $J \subseteq \{1, 2, \ldots, N\}$, we define the submatrix $\mathbf{A}|_J = (\mathbf{a}_j)_{j \in J}$, i.e., $\mathbf{A}|_J$ is a matrix that consists of the rows of $\mathbf{A}$ indexed in J.

Lemma 1.1. *Let $m, n \in \mathbb{N}$ and $M \geq m+n+1$. Further, let $\mathbf{x} := (x_1, x_2, \ldots, x_M)^T \in \mathbb{C}^M$ with pairwise distinct x_j for $j = 1, 2, \ldots, M$ and $\mathbf{y} := (y_1, y_2, \ldots, y_M)^T \in \mathbb{C}^M$. Define $\mathbf{V}_{m,n}(\mathbf{x}, \mathbf{y})$ as in equation* (1.3) *and assume that* $\operatorname{null} \mathbf{V}_{m,n}(\mathbf{x}, \mathbf{y}) \geq 1$.

Then $(\mathbf{0}^T, -\mathbf{q}^T)^T \in \ker \mathbf{V}_{m,n}(\mathbf{x},\mathbf{y})$ *with* $\mathbf{q} \neq \mathbf{0}$ *if and only if* $\|\mathbf{y}\|_0 \leq n$. *Moreover, if* $\|\mathbf{y}\|_0 \leq n$, *then all non-trivial elements in the kernel of* $\mathbf{V}_{m,n}(\mathbf{x},\mathbf{y})$ *are of the form* $(\mathbf{0}^T, -\mathbf{q}^T)^T$.

Proof. (i) Suppose that $(\mathbf{0}^T, -\mathbf{q}^T)^T \in \ker \mathbf{V}_{m,n}(\mathbf{x},\mathbf{y})$ with $\mathbf{q} \neq \mathbf{0}$. This implies that

$$y_j \cdot q(x_j) = 0 \quad \text{for all} \quad j = 1, 2, \ldots, M, \tag{1.4}$$

where q is the polynomial corresponding to $\mathbf{q}$ with $\deg(q) \leq n$. But q has at most n zeros and, hence, for equation (1.4) to hold, we must have $y_j = 0$ for at least $M - n$ indices $j \in \{1, 2, \ldots, M\}$. Put differently, this means that $\|\mathbf{y}\|_0 \leq M - M + n = n$.

(ii) Assume that $\|\mathbf{y}\|_0 \leq n$. We have

$$\mathbf{V}_{m,n}(\mathbf{x},\mathbf{y}) = \left(\underbrace{\begin{matrix} 1 & x_1 & x_1^2 & \ldots & x_1^m \\ 1 & x_2 & x_2^2 & \ldots & x_2^m \\ \vdots & \vdots & \vdots & & \vdots \\ 1 & x_M & x_M^2 & \ldots & x_M^m \end{matrix}}_{=:\mathbf{V}_m(\mathbf{x})} \quad \underbrace{\begin{matrix} y_1 & y_1 \cdot x_1 & y_1 \cdot x_1^2 & \ldots & y_1 \cdot x_1^n \\ y_2 & y_2 \cdot x_2 & y_2 \cdot x_2^2 & \ldots & y_2 \cdot x_2^n \\ \vdots & \vdots & \vdots & & \vdots \\ y_M & y_M \cdot x_M & y_M \cdot x_M^2 & \ldots & y_M \cdot x_M^n \end{matrix}}_{=:\mathbf{Y}\cdot\mathbf{V}_n(\mathbf{x})}\right),$$

where $\mathbf{V}_m(\mathbf{x}) \in \mathbb{C}^{M\times(m+1)}$ and $\mathbf{V}_n(\mathbf{x}) \in \mathbb{C}^{M\times(n+1)}$ are Vandermonde matrices and $\mathbf{Y} = \operatorname{diag}(\mathbf{y})$. Since the x_j are pairwise distinct for $j = 1, 2, \ldots, M$ and $M \geq m+n+1$, it follows that $\mathbf{V}_m(\mathbf{x})$ and $\mathbf{V}_n(\mathbf{x})$ have full column rank $m+1$ respectively $n+1$, see for example [6, Chapter 11].

Now define the index sets

$$J_1 := \{j \in \{1, 2, \ldots, M\} : y_j = 0\} := \left\{j_1^1, j_2^1, \ldots, j_{M-\|\mathbf{y}\|_0}^1\right\}$$

and

$$J_2 := \{1, 2, \ldots, M\} \setminus J_1 := \left\{j_1^2, j_2^2, \ldots, j_{\|\mathbf{y}\|_0}^2\right\}$$

and denote with $\mathbf{v}_j$ for $j = 1, 2, \ldots, M$ the rows of $\mathbf{V}_{m,n}(\mathbf{x},\mathbf{y})$. Then we have $\mathbf{v}_{j_k^1} = (1, x_{j_k^1}, x_{j_k^1}^2, \ldots, x_{j_k^1}^m, 0, \ldots, 0) \in \mathbb{C}^{m+n+2}$ for $k = 1, 2, \ldots, M - \|\mathbf{y}\|_0$ and therefore

$$\mathbf{V}_{m,n}(\mathbf{x},\mathbf{y})|_{J_1} = \left(\mathbf{v}_{j_k^1}\right)_{k=1}^{M-\|\mathbf{y}\|_0} = \begin{pmatrix} 1 & x_{j_1^1} & x_{j_1^1}^2 & \ldots & x_{j_1^1}^m & 0 & \ldots & 0 \\ 1 & x_{j_2^1} & x_{j_2^1}^2 & \ldots & x_{j_2^1}^m & 0 & \ldots & 0 \\ \vdots & \vdots & \vdots & & \vdots & \vdots & & \vdots \\ 1 & x_{j_{M-\|\mathbf{y}\|_0}^1} & x_{j_{M-\|\mathbf{y}\|_0}^1}^2 & \ldots & x_{j_{M-\|\mathbf{y}\|_0}^1}^m & 0 & \ldots & 0 \end{pmatrix}. \tag{1.5}$$

In particular, we have

$$\mathbf{V}_m(\mathbf{x})|_{J_1} = \begin{pmatrix} 1 & x_{j_1^1} & x_{j_1^1}^2 & \dots & x_{j_1^1}^m \\ 1 & x_{j_2^1} & x_{j_2^1}^2 & \dots & x_{j_2^1}^m \\ \vdots & \vdots & \vdots & & \vdots \\ 1 & x_{j^1_{M-\|\mathbf{y}\|_0}} & x_{j^1_{M-\|\mathbf{y}\|_0}}^2 & \dots & x_{j^1_{M-\|\mathbf{y}\|_0}}^m \end{pmatrix}.$$

Since $\|\mathbf{y}\|_0 \leq n$, it follows that $M - \|\mathbf{y}\|_0 \geq m + n + 1 - n = m + 1$. Therefore, $\mathbf{V}_m(\mathbf{x})|_{J_1}$ has full column rank $m + 1$.

Suppose that $(\mathbf{p}^T, -\mathbf{q}^T)^T \in \ker \mathbf{V}_{m,n}(\mathbf{x}, \mathbf{y})$. Then it follows that $(\mathbf{p}^T, -\mathbf{q}^T)^T \in \ker \mathbf{V}_{m,n}(\mathbf{x}, \mathbf{y})|_{J_1}$ and subsequently, by equation (1.5), $\mathbf{p} \in \ker \mathbf{V}_m(\mathbf{x})|_{J_1}$. But $\mathbf{V}_m(\mathbf{x})|_{J_1}$ has full column rank. Therefore, we must have $\mathbf{p} = \mathbf{0}$. Hence, all vectors in the kernel of $\mathbf{V}_{m,n}(\mathbf{x}, \mathbf{y})$ must be of the form $(\mathbf{0}^T, -\mathbf{q}^T)^T$.

Finally, we show that $\operatorname{null} \mathbf{V}_n(\mathbf{x})|_{J_2} \geq 1$. We have

$$(\mathbf{Y} \cdot \mathbf{V}_n(\mathbf{x}))\big|_{J_2} = \operatorname{diag}\left(\left(y_{j_k^2}\right)_{k=1}^{\|\mathbf{y}\|_0}\right) \cdot \mathbf{V}_n(\mathbf{x})|_{J_2} \in \mathbb{C}^{\|\mathbf{y}\|_0 \times (n+1)},$$

where $\operatorname{diag}\big((y_{j_k^2})_{k=1}^{\|\mathbf{y}\|_0}\big) \in \mathbb{C}^{\|\mathbf{y}\|_0 \times \|\mathbf{y}\|_0}$ has full rank and $\mathbf{V}_n(\mathbf{x})|_{J_2} \in \mathbb{C}^{\|\mathbf{y}\|_0 \times (n+1)}$ has full row rank $\|\mathbf{y}\|_0$. But this entails that $\operatorname{null} \mathbf{V}_n(\mathbf{x})|_{J_2} = n + 1 - \|\mathbf{y}\|_0 \geq n + 1 - n = 1$. Using equation (1.5), it follows that if $\mathbf{q} \in \ker \mathbf{V}_n(\mathbf{x})|_{J_2}$, then $(\mathbf{0}^T, -\mathbf{q}^T)^T \in \ker \mathbf{V}_{m,n}(\mathbf{x}, \mathbf{y})$. □

The following theorem is well known and can be found in similar form in the literature, see for example [5, 26, 74, 84, 102, 114]. For the convenience of the reader, we will present and prove it here in the form in which we will later use it. It characterizes the kernel of $\mathbf{V}_{m,n}(\mathbf{x}, \mathbf{y})$.

Theorem 1.2. *Let $m, n \in \mathbb{N}$ and $M \geq m+n+1$. Further, let $\mathbf{x} := (x_1, x_2, \dots, x_M)^T \in \mathbb{C}^M$ with pairwise distinct x_j for $j = 1, 2, \dots, M$ and $\mathbf{y} := (y_1, y_2, \dots, y_M)^T \in \mathbb{C}^M$ with $\|\mathbf{y}\|_0 \geq n + 1$. Define $\mathbf{V}_{m,n}(\mathbf{x}, \mathbf{y})$ as in equation (1.3) and assume that $\operatorname{null} \mathbf{V}_{m,n}(\mathbf{x}, \mathbf{y}) \geq 1$. Moreover, denote by $(x_{j_1}, y_{j_1}), (x_{j_2}, y_{j_2}), \dots, (x_{j_\nu}, y_{j_\nu})$ the unattainable points in the set $\{(x_j, y_j) : j = 0, 1, \dots, M\}$ and define $d^*(x) := (x - x_{j_1}) \cdot (x - x_{j_2}) \cdot \ldots \cdot (x - x_{j_\nu})$.*

Then there exist, up to multiplication with a non-zero scalar, uniquely defined co-prime polynomials $p^* \not\equiv 0$ and $q^* \not\equiv 0$ such that $(\mathbf{p}^T, -\mathbf{q}^T)^T \in \ker \mathbf{V}_{m,n}(\mathbf{x}, \mathbf{y})$ if and only if the corresponding polynomials p and q are of the form $p = p^* \cdot d^* \cdot s$ and $q = q^* \cdot d^* \cdot s$ for some polynomial s with $\deg(s) \leq \min\{m - \deg(p^*), n - \deg(q^*)\} - \nu$. In particular, we have $\operatorname{null} \mathbf{V}_{m,n}(\mathbf{x}, \mathbf{y}) = \min\{m - \deg(p^*), n - \deg(q^*)\} - \nu + 1$.

Proof. (i) Let $\mathbf{0} \neq (\mathbf{p}^T, -\mathbf{q}^T)^T, (\tilde{\mathbf{p}}^T, -\tilde{\mathbf{q}}^T)^T \in \ker \mathbf{V}_{m,n}(\mathbf{x}, \mathbf{y})$ with corresponding polynomials p, q, $\tilde{p}$ and $\tilde{q}$. Since $\|\mathbf{v}\|_0 \geq n+1$, we have by Lemma 1.1 that $p \not\equiv 0$ and $\tilde{p} \not\equiv 0$. Define $d := \gcd(p, q)$, $\tilde{d} := \gcd(\tilde{p}, \tilde{q})$ and set $p^* := \frac{p}{d}$, $q^* := \frac{q}{d}$, $\tilde{p}^* := \frac{\tilde{p}}{\tilde{d}}$ as well as $\tilde{q}^* := \frac{\tilde{q}}{\tilde{d}}$. First, we will show that $p^* = \tilde{p}^*$ and $q^* = \tilde{q}^*$ up to multiplication with a non-zero scalar.

Note that p^* and q^* as well as $\tilde{p}^*$ and $\tilde{q}^*$ are coprime by construction. Since both p and q as well as $\tilde{p}$ and $\tilde{q}$ satisfy equation (1.2), we have for $j = 1, 2, \ldots, M$ that

$$\begin{aligned} p(x_j) - y_j \cdot q(x_j) = \tilde{p}(x_j) - y_j \cdot \tilde{q}(x_j) = 0 \\ \Rightarrow \tilde{q}(x_j) \cdot \big(p(x_j) - y_j \cdot q(x_j)\big) = q(x_j) \cdot \big(\tilde{p}(x_j) - y_j \cdot \tilde{q}(x_j)\big) \\ \Leftrightarrow \tilde{q}(x_j) \cdot p(x_j) = q(x_j) \cdot \tilde{p}(x_j). \end{aligned}$$

But $\deg(\tilde{q} \cdot p) = \deg(\tilde{q}) + \deg(p) \leq n + m \leq M - 1$ and similarly $\deg(q \cdot \tilde{p}) \leq n + m \leq M - 1$. So $p \cdot \tilde{q}$ and $\tilde{p} \cdot q$ are polynomials of degree at most $m + n$ which agree on $M \geq m + n + 1$ points. Therefore, it follows that $\tilde{p} \cdot q = p \cdot \tilde{q}$.

Now we have

$$\begin{aligned} &\tilde{p}^* \cdot \tilde{d} \cdot q^* \cdot d = p^* \cdot d \cdot \tilde{q}^* \cdot \tilde{d} \\ \Leftrightarrow\ &\tilde{p}^* \cdot q^* = p^* \cdot \tilde{q}^* \\ \Leftrightarrow\ &\tilde{p}^* = \frac{p^* \cdot \tilde{q}^*}{q^*} \\ \Leftrightarrow\ &q^* \mid \tilde{q}^*, \end{aligned}$$

where we have used in the last step that p^* and q^* are coprime. It now follows that there must exist a polynomial $s^* \not\equiv 0$ such that

$$\tilde{q}^* = q^* \cdot s^*$$

and hence also

$$\tilde{p}^* \cdot q^* = p^* \cdot \tilde{q}^* = p^* \cdot q^* \cdot s^* \;\Leftrightarrow\; \tilde{p}^* = p^* \cdot s^*.$$

But since $\tilde{p}^*$ and $\tilde{q}^*$ are coprime, it follows that s^* must be a non-zero scalar.

Hence, every vector $(\mathbf{p}^T, -\mathbf{q}^T)^T \in \ker \mathbf{V}_{m,n}(\mathbf{x}, \mathbf{y})$ corresponds to polynomials p and q of the form $p = p^* \cdot d$ and $q = q^* \cdot d$ for some polynomial d. Since $\deg(p) \leq m$ and $\deg(q) \leq n$, it follows that $\deg(d) \leq \min\{m - \deg(p^*), n - \deg(q^*)\}$.

If now (x_k, y_k) for $1 \leq k \leq M$ is an unattainable point, then both p and q and therefore also d must contain the factor $(x - x_k)$. This is true for every unattainable point. Hence, we must have $d = (x - x_{j_1}) \cdot \ldots \cdot (x - x_{j_\nu}) \cdot s = d^* \cdot s$ with $\deg(s) \leq \min\{m - \deg(p^*), n - \deg(q^*)\} - \nu$.

Note that, since $(\mathbf{p}^T, -\mathbf{q}^T)^T \in \ker \mathbf{V}_{m,n}(\mathbf{x}, \mathbf{y})$ is equivalent to p and q satisfying equation (1.2) and we assumed that $\operatorname{null} \mathbf{V}_{m,n}(\mathbf{x}, \mathbf{y}) \geq 1$, it follows from the above argument that there can be no more than $\min\{m - \deg(p^*), n - \deg(q^*)\}$ unattainable points, i.e., $\nu \leq \min\{m - \deg(p^*), n - \deg(q^*))\}$.

(ii) Now let s be an arbitrary polynomial with $\deg(s) \leq \min\{m - \deg(p^*), n - \deg(q^*)\} - \nu$. Then clearly $p \coloneqq p^* \cdot d^* \cdot s$ and $q \coloneqq q^* \cdot d^* \cdot s$ with $\deg(p) \leq m$ and $\deg(q) \leq n$ satisfy (1.2). Therefore, it follows that $(\mathbf{p}^T, -\mathbf{q}^T)^T \in \ker \mathbf{V}_{m,n}(\mathbf{x}, \mathbf{y})$.

(iii) Finally, define $\mu \coloneqq \min\{m - \deg(p^*), n - \deg(q^*)\} - \nu$ and let $\{1, x, x^2, \ldots, x^\mu\}$ be the standard basis for polynomials of degree at most μ. Then the polynomials $p^* \cdot d^*$, $x \cdot p^* \cdot d^*$, $x^2 \cdot p^* \cdot d^*, \ldots,$ $x^\mu \cdot p^* \cdot d^*$ are linearly independent (and analogously for q^*). Define $\mathbf{p}_j$ as the coefficient vector corresponding to $x^j \cdot p^* \cdot d^*$ and $\mathbf{q}_j$ as the coefficient vector corresponding to $x^j \cdot q^* \cdot d^*$ for $0 \leq j \leq \mu$. Then we have

$$\mathbf{p}_j = (\underbrace{0, \ldots, 0}_{j}, p_0, p_1, \ldots, p_{\deg(p^*)+\nu}, \underbrace{0, \ldots, 0}_{m-\deg(p^*)-\nu-j})^T \in \mathbb{C}^{m+1}$$

and

$$\mathbf{q}_j = (\underbrace{0, \ldots, 0}_{j}, q_0, q_1, \ldots, q_{\deg(q^*)+\nu}, \underbrace{0, \ldots, 0}_{n-\deg(q^*)-\nu-j})^T \in \mathbb{C}^{n+1}.$$

By (ii), we have $(\mathbf{p}_j^T, -\mathbf{q}_j^T)^T \in \ker \mathbf{V}_{m,n}(\mathbf{x}, \mathbf{y})$ for $0 \leq j \leq \mu$. Further, as is apparent from their structure, these vectors are linearly independent.

Lastly, for $s(x) = s_0 + s_1 \cdot x + \ldots + s_\mu \cdot x^\mu \in \mathbb{C}[x]$, we find that $p^* \cdot d^* \cdot s$ corresponds to $s_0 \cdot \mathbf{p}_0 + s_1 \cdot \mathbf{p}_1 + \ldots + s_\mu \cdot \mathbf{p}_\mu$ (and analogously for $q^* \cdot d^* \cdot s$). Hence, every element in the kernel of $\mathbf{V}_{m,n}(\mathbf{x}, \mathbf{y})$ can be written as

$$\alpha_0 \cdot (\mathbf{p}_0^T, -\mathbf{q}_0^T)^T + \alpha_1 \cdot (\mathbf{p}_1^T, -\mathbf{q}_1^T)^T + \ldots + \alpha_\mu \cdot (\mathbf{p}_\mu^T, -\mathbf{q}_\mu^T)^T$$

with $\alpha_0, \alpha_1, \ldots, \alpha_\mu \in \mathbb{C}$ and therefore $(\mathbf{p}_j^T, -\mathbf{q}_j^T)^T$ for $0 \leq j \leq \mu$ is a basis of $\ker \mathbf{V}_{m,n}(\mathbf{x}, \mathbf{y})$ consisting of $\mu + 1 = \min\{m - \deg(p^*), n - \deg(q^*)\} - \nu + 1$ vectors. $\square$

Remark 1.3. The quantity $\min\{m - \deg(p^*), n - \deg(q^*)\}$ in Theorem 1.2 is known as the *defect* of a rational function in $\mathcal{R}_{m,n}$ see for example [109]. Further, the polynomial d^* is also called the *deficiency polynomial*, see for example [52].

1.2. Reconstruction of Polynomials via Classical Rational Interpolation

The traditional rational interpolation problem is described in (1.1). Further, in many applications of rational interpolation it is assumed that the given data set does not contain unattainable points. However, here we are interested in a slightly different problem, namely exactly finding the polynomials p^*, q^*, and d^* from Theorem 1.2. In particular, we will be interested in finding and reconstructing unattainable points, which is equivalent to reconstructing d^*. To emphasize that we do not just want to solve problem (1.1), which is not solvable if there are unattainable points in the given sample set anyway, but want to explicitly find p^*, q^*, and d^* we talk about the *reconstruction of polynomials via rational interpolation* or simply the *polynomial reconstruction problem.*

The problem we want to solve is the following. For pairwise different sample points $x_1, x_2, \dots, x_M \in \mathbb{C}$, we are given sample values of the form

$$y_j = y_j^* + \tilde{y}_j \tag{1.6}$$

such that

$$y_j^* = \frac{p^*(x_j)}{q^*(x_j)}$$

for $j = 1, 2, \dots, M$, where p^* and q^* are coprime polynomials with $\deg(p^*) = m^*$ and $\deg(q^*) = n^*$. Further, we assume that $\tilde{y}_j \neq 0$ for ν sample values $y_{j_1}, y_{j_2}, \dots, y_{j_\nu}$. Our goal is to reconstruct the polynomials p^* and q^* (up to multiplication with a non-zero scalar) and $d^*(x) \coloneqq (x - x_{j_1}) \cdot (x - x_{j_2}) \cdot \ldots \cdot (x - x_{j_\nu})$ from $M \geq m^* + n^* + 2\nu + 1$ samples of the form (1.6). We will say that y_{j_k} are *distorted sample values*, or that (x_{j_k}, y_{j_k}) are *distorted samples* for $k = 1, 2, \dots, \nu$.

Before we go on, we introduce some notation. Let

$$S \coloneqq \{(x_j, y_j) : j = 1, 2, \dots, M\}$$

denote the set of given samples, where y_j is as in equation (1.6), for $j = 1, 2, \dots, M$. Further, define the set

$$D \coloneqq \{(x_j, y_j) \in S : \tilde{y}_j \neq 0\}.$$

From now on we will write

$$\mathbf{V}_{m,n}(S) \coloneqq \mathbf{V}_{m,n}(\mathbf{x}, \mathbf{y}) \tag{1.7}$$

for $\mathbf{x} = (x_1, x_2, \dots, x_M)^T$ and $\mathbf{y} = (y_1, y_2, \dots, y_M)^T$ with $(x_j, y_j) \in S$ for $j = 1, 2, \dots, M$. Moreover, define for any set of sample points and values $\tilde{S} = \{(x_{j_k}, y_{j_k}) : k = 1, 2, \dots, |\tilde{S}|\} \subseteq S$ the polynomial

$$d_{\tilde{S}}(x) \coloneqq (x - x_{j_1}) \cdot (x - x_{j_2}) \cdot \ldots \cdot (x - x_{j_{|\tilde{S}|}}). \tag{1.8}$$

Of particular interest for us will be the polynomial d_D.

Now note that for sample values as in equation (1.6) we have

$$p^*(x_j) \cdot d_D(x_j) - y_j \cdot q^*(x_j) \cdot d_D(x_j) = 0 \quad \text{for} \quad j = 1, 2, \dots, M.$$

This means that the elements of D are precisely the unattainable points in the given sample set. Moreover, the polynomials p^*, q^* and d_D are exactly the polynomials which characterize the kernel of $\mathbf{V}_{m,n}(S)$ for $m \geq m^* + \nu$ and $n \geq n^* + \nu$, i.e., $d_D = d^*$ in Theorem 1.2.

The following algorithm outlines a procedure to reconstruct the polynomials p^*, q^*, and d^*.

Algorithm 1.4 (Reconstruction of Polynomials from Structured Samples)**.**

Input: $S = \{(x_j, y_j) : j = 1, 2, \dots, M\}$. A set of samples as in equation (1.6) with $M \geq m^* + n^* + 2\nu + 1$.

$m \geq m^* + \nu$ and $n \geq n^* + \nu$. Estimates for $m^* = \deg(p^*)$, $n^* = \deg(q^*)$, and the number ν of unattainable points in S.

(i) Construct the matrix $\mathbf{V}_{m,n}(S)$.

(ii) Find $(\mathbf{p}^T, -\mathbf{q}^T)^T \in \ker \mathbf{V}_{m,n}(S)$ with corresponding polynomials p and q.

(iii) Compute $d = \gcd(p, q)$ and set $p^* = \frac{p}{d}$ as well as $q^* = \frac{q}{d}$.

(iv) Compute $\tilde{y}_j = y_j - \frac{p^*(x_j)}{q^*(x_j)}$ for all $j = 1, 2, \dots, M$.

(v) Set $d^*(x) = (x - x_{j_1}) \cdot (x - x_{j_2}) \cdot \ldots \cdot (x - x_{j_\nu})$ for j_k such that $\tilde{y}_{j_k} \neq 0$ for $k = 1, 2, \dots, \nu$.

Output: p^*, q^* and d^*.

Let us take a closer look at the solvability of the polynomial reconstruction problem.

Lemma 1.5. *Let p^* and q^* be coprime polynomials with* $\deg(p^*) = m^*$ *and* $\deg(q^*) = n^*$. *Further, let $m, n \in \mathbb{N}$ and $M \geq m+n+2$. Suppose we are given sets S and D as above, i.e., S contains M samples with sample values of the form* (1.6) *and $D \subseteq S$ contains all distorted samples, where $|D| = \nu$. If* $\operatorname{null} \mathbf{V}_{m,n}(S) = 0$ *then $m < m^* + \nu$ or $n < n^* + \nu$.*

Proof. Since $M \geq m+n+2$, there need not exist a non-trivial solution to $\mathbf{V}_{m,n}(S)\cdot\mathbf{v} = \mathbf{0}$. However, if $m \geq m^* + \nu$ and $n \geq n^* + \nu$, then it follows directly that the vector $\left((\mathbf{p}^*)^T, -(\mathbf{q}^*)^T\right)^T \in \mathbb{C}^{m+n+2}$ corresponding to $p^* \cdot d_D$ and $q^* \cdot d_D$ is in the kernel of $\mathbf{V}_{m,n}(S)$. Therefore, $\operatorname{null} \mathbf{V}_{m,n}(S) = 0$ can only be true if $m < m^* + \nu$ or $n < n^* + \nu$. □

In the context of reconstruction of polynomials, this result can be understood as follows. If the reconstruction matrix with at least as many rows as columns has only a trivial kernel, then we have underestimated the degree of the polynomial $p^* \cdot d_D$ or $q^* \cdot d_D$ which we wish to reconstruct. This could also mean that we do not have enough undistorted samples from $\frac{p^*}{q^*}$.

We collect a few more facts about polynomial reconstruction.

Remark 1.6. (i) We can use Lemma 1.5 in order to modify Algorithm 1.4. Given a set S of M samples, choose m and n such that $m+n+2 \leq M$. If then $\operatorname{null} \mathbf{V}_{m,n}(S) = 0$, we know that we have either made a wrong choice for m and n or that we do not have enough samples to reconstruct p^* and q^*.

(ii) Of course this leaves out the question how to estimate m and n. However, later on we will use the results of this section in order to reconstruct polynomials where we know that $m^* \leq n^*-1$, even $m^* = n^*-1$ almost surely. Given M samples we therefore choose $n = \left\lfloor \frac{M-1}{2} \right\rfloor$ and $m = n-1$ such that $m+n = 2\cdot\left\lfloor \frac{M-1}{2} \right\rfloor - 1 \leq 2\cdot\frac{M-1}{2} - 1 = M-2$. If then $\operatorname{null} \mathbf{V}_{m,n}(S) = 0$, we can be sure that we have too few samples.

(iii) Suppose we are given a sample set S with sample values of the form (1.6) containing ν unattainable points and where $\deg(p^*) = m^*$ and $\deg(q^*) = n^*$. According to Theorem 1.2, it suffices to have $M = m^*+n^*+2\nu+1$ samples in order to reconstruct p^*, q^* and d^*. However, in this case the matrix $\mathbf{V}_{m^*+\nu,n^*+\nu}(S) \in \mathbb{C}^{M\times(m^*+n^*+2\nu+2)}$ has always a non-trivial kernel. Therefore, unless we know m^*, n^* and ν beforehand, we have no way of knowing whether the polynomials corresponding the the kernel vector of $\mathbf{V}_{m^*+\nu,n^*+\nu}(S)$ are indeed equal to p^*, q^* and d^*. Following (i), we should thus always take at least $M = m^* + n^* + 2\nu + 2$ samples. However, even if we only consider reconstruction matrices with at least as many rows as columns, this is no guarantee that we can either reconstruct the polynomials from which our samples stem correctly or find that we have too few samples, as we discuss next.

(iv) Unfortunately, the reverse of Lemma 1.5 is not true. If $\operatorname{null} \mathbf{V}_{m,n}(S) \geq 1$, where we have $M \geq m+n+2$ samples, then it does not follow that $m \geq m^* + \nu$ and $n \geq n^* + \nu$, which implies that we can correctly reconstruct the polynomials p^* and q^*. This follows from the fact that if we have too few samples to begin with, it is always possible that there are polynomials $\tilde{p}$ and $\tilde{q}$ with $\deg(\tilde{p}) < m^*$ and $\deg(\tilde{q}) < n^*$ such that $\frac{\tilde{p}}{\tilde{q}}$ interpolates the given (attainable) points. We will illustrate this in Example 1.7. However, if we know m^* and n^* in advance, then we can formulate a reverse result of Lemma 1.5.

(v) Let $r = \frac{p^*}{q^*}$ be a rational function of type (m^*, n^*), where p^* and q^* are coprime. The *MacMillan degree* of this function, which we will simply denote by $\deg(r)$, is given by $\max\{m^*, n^*\}$. If $m \geq \deg(r)$ and we have a sample set $S = \{(x_j, r(x_j)) : j = 1, 2, \ldots, M\}$ with $M \geq m^* + n^* + 1$, then the rank (or equivalently the nullity) of $\mathbf{V}_{m,m}(S)$ encodes $\deg(r)$, since, by Theorem 1.2, we have $\operatorname{null} \mathbf{V}_{m,m}(S) = \min\{m - m^*, m - n^*\} - \nu + 1 = m - \deg(r) - 0 + 1 = m + 1 - \deg(r)$, or equivalently $\operatorname{rk} \mathbf{V}_{m,m}(S) = 2 \cdot (m+1) - (m+1) + \deg(r) = m + 1 + \deg(r)$. Later on, a similar observation, namely that the degree of a rational function r is encoded in the rank of a special Löwner matrix associated to r, will become very important.

Example 1.7. Consider the rational function

$$\frac{p^*(x)}{q^*(x)} = \frac{10x^3 - 43x^2 + 51x - 12}{4x^3 - 18x^2 + 22x - 6},$$

where p^* and q^* are coprime. Since $\deg(p^*) = \deg(q^*) = 3$, we need by Theorem 1.2 at least 7 samples in order to reconstruct p^* and q^*. Suppose we are given a set of $M = 8$ samples

$$S = \left\{(0,2), \left(\tfrac{1}{2}, 4\right), (1,3), (2,1), \left(\tfrac{5}{2}, -3\right), (3,4), \left(4, \tfrac{72}{25}\right), \left(5, \tfrac{19}{7}\right)\right\}.$$

Then we get

$$\mathbf{V}_{3,3}(S) = \begin{pmatrix} 1 & 0 & 0 & 0 & 2 & 0 & 0 & 0 \\ 1 & \frac{1}{2} & \frac{1}{4} & \frac{1}{8} & 4 & 2 & 1 & \frac{1}{2} \\ 1 & 1 & 1 & 1 & 3 & 3 & 3 & 3 \\ 1 & 2 & 4 & 8 & 1 & 2 & 4 & 8 \\ 1 & \frac{5}{2} & \frac{25}{4} & \frac{125}{8} & -3 & -\frac{15}{2} & -\frac{75}{4} & -\frac{375}{8} \\ 1 & 3 & 9 & 27 & 4 & 12 & 36 & 108 \\ 1 & 4 & 16 & 64 & \frac{72}{25} & \frac{288}{25} & \frac{1152}{25} & \frac{4608}{25} \\ 1 & 5 & 25 & 125 & \frac{19}{7} & \frac{95}{7} & \frac{475}{7} & \frac{2375}{7} \end{pmatrix}$$

with $\operatorname{null} \mathbf{V}_{3,3}(S) = 1$ where the kernel is spanned by the vector

$$(-12, 51, -43, 10, 6, -22, 18, -4)^T = \left((\mathbf{p}^*)^T, (-\mathbf{q}^*)^T\right)^T.$$

The matrix

$$\mathbf{V}_{4,2}(S) = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 2 & 0 & 0 \\ 1 & \frac{1}{2} & \frac{1}{4} & \frac{1}{8} & \frac{1}{16} & 4 & 2 & 1 \\ 1 & 1 & 1 & 1 & 1 & 3 & 3 & 3 \\ 1 & 2 & 4 & 8 & 16 & 1 & 2 & 4 \\ 1 & \frac{5}{2} & \frac{25}{4} & \frac{125}{8} & \frac{625}{16} & -3 & -\frac{15}{2} & -\frac{75}{4} \\ 1 & 3 & 9 & 27 & 81 & 4 & 12 & 36 \\ 1 & 4 & 16 & 64 & 128 & \frac{72}{25} & \frac{288}{25} & \frac{1152}{25} \\ 1 & 5 & 25 & 125 & 625 & \frac{19}{7} & \frac{95}{7} & \frac{475}{7} \end{pmatrix},$$

on the other, hand has full rank.

Now suppose that we have only 4 samples $S_1 = \left\{(0,2), \left(\frac{1}{2}, 4\right), (1,3), \left(5, \frac{19}{7}\right)\right\}$. Then the matrix

$$\mathbf{V}_{1,1}(S_1) = \begin{pmatrix} 1 & 0 & 2 & 0 \\ 1 & \frac{1}{2} & 4 & 2 \\ 1 & 1 & 3 & 3 \\ 1 & 5 & \frac{19}{7} & \frac{95}{7} \end{pmatrix}$$

is singular and its kernel is spanned by the vector $(2, -8, -1, 3)^T$, which corresponds to the polynomial

$$\frac{\tilde{p}(x)}{\tilde{q}(x)} = \frac{8x-2}{3x-1}.$$

This polynomial does indeed agree with $\frac{p^*}{q^*}$ at the points $0, \frac{1}{2}, 1$ and 5. Note, however, that this only happens because we made an unfortunate choice for our sample set. If we take, for example, the set $S_2 = \{(0,2),(1,3),(2,1),(3,4)\}$, then

$$\mathbf{V}_{1,1}(S_2) = \begin{pmatrix} 1 & 0 & 2 & 0 \\ 1 & 1 & 3 & 3 \\ 1 & 2 & 1 & 2 \\ 1 & 3 & 4 & 12 \end{pmatrix}$$

has full rank.

This underpins what we have said before. Given some knowledge about the degrees m^* and n^* of the numerator and denominator polynomials p^* and q^*, for example that $m^* = n^* - 1$, we can infer from $\operatorname{null} \mathbf{V}_{m,n}(S) = 0$ for $|S| \geq m + n + 2$ that we have too few samples. If, on the other hand, $\operatorname{null} \mathbf{V}_{m,n}(S) \geq 1$ then this is no guarantee that we have enough samples to reconstruct p^* and q^*.

1.3. Barycentric Rational Interpolation

The methods for rational interpolation and reconstruction of polynomials presented in the previous sections have the great advantage of being quite straightforward. Therefore, they lend themselves very well to theoretical considerations, which is what we will use them for later on. However, it is well established that they suffer greatly from numerical instability, which we will also see in the next chapter. This is mainly due to the usually very bad condition of the Vandermonde matrices involved, see [87]. We devote the following sections to presenting a more stable method for rational interpolation and reconstruction of polynomials, so-called barycentric rational interpolation.

The first barycentric formula, as we will later call it, was already known by Jacobi in 1825, see [61]. The second barycentric formula appears in the 1940s, see [39, 108]. Some early sources for barycentric interpolation are [23, 53, 113]. From the 1980s onward, there has been a growing interest in this topic, see for example [9, 43, 54, 64, 101, 109] as well as the references therein and the literature mentioned in this section. The stability of barycentric interpolation has been investigated in [55]. A particularly good overview is given in [14]. We will follow this paper in introducing barycentric rational interpolation.

As before, we start by looking at the problem of polynomial interpolation. Recall that the unique polynomial of degree $M - 1$ interpolating the sample values $y_j \in \mathbb{C}$

at distinct sample points $x_j \in \mathbb{C}$ for $j = 1, 2, \ldots, M$ is given by

$$p(x) = \sum_{j=1}^{M} y_j \cdot \ell_j(x) \tag{1.9}$$

with

$$\ell_j(x) := \prod_{\substack{i=1\\ i\neq j}}^{M} \frac{x - x_i}{x_j - x_i}.$$

We now rewrite this formula. To this end we define

$$\ell(x) := \prod_{i=1}^{M} (x - x_i) \tag{1.10}$$

and the *barycentric weights*

$$w_j := \prod_{\substack{i=1\\ i\neq j}}^{M} \frac{1}{x_j - x_i} \tag{1.11}$$

for $j = 1, 2, \ldots, M$. Then we can write

$$\ell_j(x) = \ell(x) \cdot \frac{w_j}{x - x_j}.$$

Inserting this representation into equation (1.9) gives

$$p(x) = \ell(x) \sum_{j=1}^{M} \frac{w_j}{x - x_j} \cdot y_j. \tag{1.12}$$

This is called the "first form of the barycentric interpolation formula" [100]. As this quote suggests, there is also a "second (true) form of the barycentric formula" [100] and it is derived as follows. The function which constantly takes the value 1, in other words, the polynomial of degree 0 given by $p \equiv 1$, can be written as

$$1 = \sum_{j=1}^{M} \ell_j(x) = \ell(x) \sum_{j=1}^{M} \frac{w_j}{x - x_j}. \tag{1.13}$$

Here, the first equation is a special case of equation (1.9), where $y_j = 1$ for all $j = 1, 2, \ldots, M$. It follows from the fact that $\sum_{j=1}^{M} \ell_j(x)$ is (formally) a polynomial of degree $M - 1$ which coincides with the degree 0 polynomial $p \equiv 1$ on M points.

Dividing (1.12) by (1.13) and canceling the common factor $\ell(x)$ then gives us

$$p(x) = \frac{\sum_{j=1}^{M} \frac{w_j}{x-x_j} \cdot y_j}{\sum_{j=1}^{M} \frac{w_j}{x-x_j}}, \tag{1.14}$$

which is simply called the *barycentric formula.*

One of the great advantages of the barycentric formula is that, even if the weights w_j are not exactly as defined in (1.11), as long as $w_j \neq 0$, the function p will still interpolate the value y_j at the point x_j for $j = 1, 2, \ldots, M$. This is true since we have

$$\begin{aligned} \lim_{x \to x_k} p(x) &= \lim_{x \to x_k} \frac{\sum_{j=1}^{M} \frac{w_j}{x-x_j} \cdot y_j}{\sum_{j=1}^{M} \frac{w_j}{x-x_j}} \\ &= \lim_{x \to x_k} \frac{\prod_{j=1}^{M} \frac{1}{x-x_j} \sum_{j=1}^{M} w_j \cdot y_j \cdot \prod_{\substack{i=1 \\ i \neq j}}^{M} (x - x_i)}{\prod_{j=1}^{M} \frac{1}{x-x_j} \sum_{j=1}^{M} w_j \cdot \prod_{\substack{i=1 \\ i \neq j}}^{M} (x - x_i)} \\ &= \frac{w_k \cdot y_k \cdot \prod_{\substack{i=1 \\ i \neq k}}^{M} (x_k - x_i)}{w_k \cdot \prod_{\substack{i=1 \\ i \neq k}}^{M} (x_k - x_i)} = y_k \end{aligned} \tag{1.15}$$

for $k = 1, 2, \ldots, M$. However, unless the weights are defined as in (1.11), the resulting interpolation function will usually no longer be a polynomial but a rational function. Since we are interested in interpolation with rational functions, let us take a closer look at this.

It has been shown in [12, Lemma 2.1] that every rational function $r \in \mathcal{R}_{M-1,M-1}$ can be written in barycentric form. Since the proof of this result is constructive, we repeat it here. Let $r = \frac{p}{q} \in \mathcal{R}_{M-1,M-1}$ with p and q coprime and $r(x_j) = y_j$ for $j = 1, 2, \ldots, M$. The first barycentric form of q is given by

$$q(x) = \ell(x) \sum_{j=1}^{M} \frac{w_j}{x - x_j} \cdot q(x_j),$$

where $\ell(x)$ and w_j are defined as in equations (1.10) and (1.11). This equality follows from the fact that on the right hand side of the equation we have a polynomial of degree at most $M-1$ which coincides with the polynomial q, which also has at most degree $M-1$, in M points. Now define

$$u_j := w_j \cdot q(x_j) \tag{1.16}$$

for $j = 1, 2, \dots, M$. Since $\frac{p(x_j)}{q(x_j)} = y_j$, it follows that $p(x_j) = y_j \cdot q(x_j)$. Therefore, the first barycentric form of p is given by

$$p(x) = \ell(x) \sum_{j=1}^{M} \frac{w_j}{x - x_j} \cdot y_j \cdot q(x_j) = \ell(x) \sum_{j=1}^{M} \frac{u_j}{x - x_j} \cdot y_j.$$

Putting everything back together and canceling the common factors, we then find that

$$r(x) = \frac{p(x)}{q(x)} = \frac{\sum_{j=1}^{M} \frac{u_j}{x-x_j} \cdot y_j}{\sum_{j=1}^{M} \frac{u_j}{x-x_j}}. \tag{1.17}$$

Equations (1.16) and (1.17) show another advantage of the barycentric representation. Given a set of samples $\{(x_j, y_j) : j = 1, 2, \dots, M\}$ for some $M \in \mathbb{N}$, a point (x_k, y_k) for some $1 \leq k \leq M$ is an unattainable point if and only if $u_k = 0$, since in this case $q(x_k) = 0$, see for example [11, 104].

As noted before, the interpolation condition $r(x_j) = y_j$ is satisfied for any choice of weights $u_j \neq 0$ for $j = 1, 2, \dots, M$. However, only if $u_j = w_j \cdot q(x_j)$ with w_j as in equation (1.11) for $j = 1, 2, \dots, M$ do we get $r = \frac{p}{q}$, i.e., with other weights, we get a different rational function. This raises the question how to find the correct weights given samples of $\frac{p}{q}$. Several methods have been proposed for this task. In [13, 104, 116] the weights are computed as solutions to homogeneous linear equations. Particularly, in [13], it is shown that the vector of weights $\mathbf{u} = (u_1, u_2, \dots, u_M)^T$ such that the barycentric representation satisfies equation (1.17) is (up to multiplication with a non-zero scalar) a unique solution to

$$\underbrace{\begin{pmatrix} 1 & 1 & \dots & 1 \\ x_1 & x_2 & \dots & x_M \\ x_1^2 & x_2^2 & \dots & x_M^2 \\ \vdots & \vdots & & \vdots \\ x_1^{m-1} & x_2^{m-1} & \dots & x_M^{m-1} \\ y_1 & y_2 & \dots & y_M \\ y_1 \cdot x_1 & y_2 \cdot x_2 & \dots & y_M \cdot x_M \\ y_1 \cdot x_1^2 & y_2 \cdot x_2^2 & \dots & y_M \cdot x_M^2 \\ \vdots & \vdots & & \vdots \\ y_1 \cdot x_1^{n-1} & y_2 \cdot x_2^{n-1} & \dots & y_M \cdot x_M^{n-1} \end{pmatrix}}_{=:\tilde{\mathbf{V}}_{m-1,n-1}(\mathbf{x},\mathbf{y}) \in \mathbb{C}^{m+n \times M}} \cdot \mathbf{u} = \mathbf{0}, \tag{1.18}$$

where $\mathbf{x} = (x_1, x_2, \dots, x_M)^T$, $\mathbf{y} = (y_1, y_2, \dots, y_M)^T$, $m = \deg(p)$, $n = \deg(q)$, and $M = m+n+1$. In [13], it is assumed that $n \leq m$. In [11], the case that $m < n$ is also discussed. The authors also consider the case that $m > \deg(p)$ or $n > \deg(q)$, which leads to $\mathbf{u}$ not being unique anymore, since then $\operatorname{null} \tilde{\mathbf{V}}_{m-1,n-1}(\mathbf{x}, \mathbf{y}) > 1$. This can lead to the barycentric representation as in (1.17) corresponding to a rational function where the numerator and denominator polynomials contain a common factor. In other words, using the language from Theorem 1.2, there exists a polynomial s with $\deg(s) \geq 1$ such that

$$\ell(x) \sum_{j=1}^{M} \frac{u_j}{x - x_j} \cdot y_j = p(x) \cdot s(x) \quad \text{and} \quad \ell(x) \sum_{j=1}^{M} \frac{u_j}{x - x_j} = q(x) \cdot s(x).$$

In order to obtain a unique kernel vector, the authors suggest in this case to restart the algorithm with $n-1$ instead of n, i.e., computing the kernel of $\tilde{\mathbf{V}}_{m-1,n-2}(\mathbf{x}, \mathbf{y})$ instead of $\tilde{\mathbf{V}}_{m-1,n-1}(\mathbf{x}, \mathbf{y})$. However, this relies on the assumption that $n \leq m$ and might not be computationally feasible if the degrees of p and q are much smaller than the initial guesses for m and n.

Note the similarity between the systems of equations in (1.3) and (1.18). As a matter of fact we have $\tilde{\mathbf{V}}_{m-1,n-1}(\mathbf{x}, \mathbf{y}) = \mathbf{V}_{m-1,n-1}(\mathbf{x}, \mathbf{y})^T$ for $M = m+n+1$. However, in order to solve the classical rational interpolation problem with $\deg(p) = m$ and $\deg(q) = n$ we need the matrix $\mathbf{V}_{m,n}(\mathbf{x}, \mathbf{y})$. Moreover, while $\mathbf{V}_{m,n}(\mathbf{x}, \mathbf{y})$ describes interpolation conditions, $\tilde{\mathbf{V}}_{m-1,n-1}(\mathbf{x}, \mathbf{y})$ imposes a degree condition, i.e., if $\mathbf{u}$ satisfies equation (1.18), then it is ensured in the corresponding barycentric representation that $\deg(p) \leq m$ and $\deg(q) \leq n$, where the first m rows of $\tilde{\mathbf{V}}_{m-1,n-1}(\mathbf{x}, \mathbf{y})$ correspond to the degree of q and the last n rows correspond to the degree of p.

Although the authors claim in [13] that the approach presented above is numerically stable, it still involves solving a homogeneous Vandermonde-like system of equations. Moreover, it faces the same problems as the classical approach if the degrees of p and q are unknown. Hence, the only advantage of this approach over the classical approach is that unattainable points are automatically taken care of by virtue of the barycentric representation. We will therefore consider another method for finding the barycentric weights, which will lead to an iterative procedure, thereby also avoiding the awkward re-running of the algorithm with reduced n as suggested in [13].

Before we continue, let us recall and clarify some important terms. First, we need to distinguish between a rational function r and its *representation* as fraction of polynomials $\frac{p}{q}$. We will say that two rational functions r_1 and r_2 are equal (in an algebraic sense) and write $r_1 = r_2$, if there exist coprime polynomials p^* and q^*

such that $r_1 = \frac{p^* \cdot d_1}{q^* \cdot d_1}$ and $r_2 = \frac{p^* \cdot d_2}{q^* \cdot d_2}$ for some polynomials d_1 and d_2. This means that while r_1 and r_2 might have different representations, they are indeed the same rational function. We need this distinction in order to define two important properties of rational functions. As already introduced, the *type* of a rational function $r = \frac{p}{q}$ is given by $\big(\deg(p), \deg(q)\big)$. If $\deg(p) = \deg(q) = m$ we will simply say that r is of type m. The type of a rational function depends on its representation. Another important concept is the *degree* of a rational function r. There are several ways of defining the degree of a rational function. We will use the so-called MacMillan degree, which was already mentioned in Remark 1.6 (v). It is defined as

$$\deg(r) \coloneqq \max\{\deg(p^*), \deg(q^*)\}, \tag{1.19}$$

where p^* and q^* are coprime and $r = \frac{p^* \cdot d}{q^* \cdot d}$ for some polynomial d, i.e., the degree of r is independent of its representation. To illustrate the difference between type and degree consider r_1 and r_2 as above. The type of r_1 is $\big(\deg(p^* \cdot d_1), \deg(q^* \cdot d_1)\big)$ and the type of r_2 is $\big(\deg(p^* \cdot d_2), \deg(q^* \cdot d_2)\big)$, which need not be the same. On the other hand, we have $\deg(r_1) = \deg(r_2) = \max\{\deg(p^*), \deg(q^*)\}$. A similar notion is the *exact type* of a rational function, given by $\big(\deg(p^*), \deg(q^*)\big)$. We will not need this concept here though. When we want to emphasize that the numerator and denominator polynomials of a representation of a rational function r are coprime, we will say that r is *fully reduced* or *minimal* (as shorthand for *of minimal type*), i.e., its type is equal to its exact type.

The distinction between the degree and the type of a rational function will become important later on. The notions of degree and equality of rational functions we employ represent an algebraic point of view on the problem at hand (algebraically, there is no difference between the functions r_1 and r_2 above). However, once we will concern ourselves with the numerics of the problem, the type of a rational function will play an important role.

Now back to finding a suitble weight vector for a barycentric rational interpolation function r. Antoulas and Anderson consider in [1] an implicit representation of this function. If r is a rational function which interpolates a given sample set $S = \{(x_j, y_j) : j = 1, 2, \ldots, M\}$ with $M \in \mathbb{N}$ then

$$\sum_{j=1}^{M} u_j \cdot \frac{r(x) - y_j}{x - x_j} = 0, \tag{1.20}$$

where $u_j \neq 0$ for $j = 1, 2, \ldots, M$. This formulation is indeed equivalent to the formulation in (1.17), since we have

$$\begin{aligned}
0 &= \sum_{j=1}^{M} u_j \cdot \frac{r(x) - y_j}{x - x_j} \\
&= r(x) \sum_{j=1}^{M} \frac{u_j}{x - x_j} - \sum_{j=1}^{M} \frac{u_j}{x - x_j} \cdot y_j \\
\Leftrightarrow r(x) &= \frac{\sum_{j=1}^{M} \frac{u_j}{x-x_j} \cdot y_j}{\sum_{j=1}^{M} \frac{u_j}{x-x_j}}.
\end{aligned}$$

One has to be careful though that the equivalence above is not to be understood pointwise. If $x = x_k$ for some $1 \leq k \leq M$ then the right hand side of the second equation is of the form $\infty - \infty$. However, as shown in equation (1.15), in this case we have $r(x_k) = y_k$, as long as $u_k \neq 0$. If, on the other hand,

$$\sum_{j=1}^{M} \frac{u_j}{\tilde{x} - x_j} = \sum_{j=1}^{M} \frac{u_j}{\tilde{x} - x_j} \cdot y_j = 0$$

for some $\tilde{x} \in \mathbb{C}$, then r has either a zero, a pole, or a removable singularity at $\tilde{x}$. In any case, this implies that the numerator and denominator polynomials have a factor in common.

The goal of [1] is to study all minimal solutions of the rational interpolation problem $r(x_j) = y_j$ for $j = 1, 2, \ldots, M$. (Beware though that Antoulas and Anderson use a different terminology than we do. They talk about finding a "minimal degree" solution and define the MacMillan degree analogously to our definition of the type.) As noted before, the function

$$r(x) = \frac{\sum_{j=1}^{M} \frac{u_j}{x-x_j} \cdot y_j}{\sum_{j=1}^{M} \frac{u_j}{x-x_j}}$$

will interpolate the samples (x_j, y_j) as long as $u_j \neq 0$ for $j = 1, 2, \ldots, M$. Moreover, r will generically be of type $M - 1$. The task now is to find a "lower complexity barycentric representation" [10] of r, denoted by r', such that

$$r'(x) = \frac{\sum_{j=1}^{M'} \frac{u_j}{x-x_j} \cdot y_j}{\sum_{j=1}^{M'} \frac{u_j}{x-x_j}}$$

is of minimal type $M'-1 \leq M-1$ and satisfies the interpolation conditions $r'(x_j) = y_j$ for all $j = 1, 2, \ldots, M$.

By construction, we have $r'(x_j) = y_j$ if $u_j \neq 0$ for $j = 1, 2, \ldots, M'$. Now let $(x_k, y_k) \in S' := \{(x_j, y_j) : j = M'+1, M'+2, \ldots, M\}$. If $r'(x_k) = y_k$ then it follows from the implicit equation for r' analogously to (1.20) that

$$0 = \sum_{j=1}^{M'} u_j \cdot \frac{r'(x_k) - y_j}{x_k - x_j} = \sum_{j=1}^{M'} u_j \cdot \frac{y_k - y_j}{x_k - x_j} = \left(\frac{y_k - y_1}{x_k - x_1}, \frac{y_k - y_2}{x_k - x_2}, \ldots, \frac{y_k - y_{M'}}{x_k - x_{M'}} \right) \cdot \begin{pmatrix} u_1 \\ u_2 \\ \vdots \\ u_{M'} \end{pmatrix}.$$

Therefore, if $r'(x_j) = y_j$ for $j = M'+1, M'+2, \ldots, M$, we must have

$$\underbrace{\begin{pmatrix} \frac{y_{M'+1}-y_1}{x_{M'+1}-x_1} & \frac{y_{M'+1}-y_2}{x_{M'+1}-x_2} & \cdots & \frac{y_{M'+1}-y_{M'}}{x_{M'+1}-x_{M'}} \\ \frac{y_{M'+2}-y_1}{x_{M'+2}-x_1} & \frac{y_{M'+2}-y_2}{x_{M'+2}-x_2} & \cdots & \frac{y_{M'+2}-y_{M'}}{x_{M'+2}-x_{M'}} \\ \vdots & \vdots & & \vdots \\ \frac{y_M-y_1}{x_M-x_1} & \frac{y_M-y_2}{x_M-x_2} & \cdots & \frac{y_M-y_{M'}}{x_M-x_{M'}} \end{pmatrix}}_{=:\mathbf{L}} \cdot \begin{pmatrix} u_1 \\ u_2 \\ \vdots \\ u_{M'} \end{pmatrix} = \mathbf{0}.$$

The matrix $\mathbf{L}$ is called a *divided-differences* or *Löwner matrix*, after Karl Löwner, who studied the connection between these matrices and rational interpolation in [73].

Note that the ordering of elements in $S = \{(x_j, y_j) : j = 1, 2, \ldots, M\}$ does not matter. The argument above works exactly the same if we take any subset $T' \subseteq S$ with $|T'| = M' \leq M$ in order to build r'. In [1], the set T' is called *column array* and the set $S' := S \setminus T'$ is called *row array*. The sets S' and T' form a *partition* of S. Let $I_{S'} := \{j \in \{1, 2, \ldots, |S|\} : (x_j, y_j) \in S'\}$ be the index set corresponding to S' and define $I_{T'}$ analogously. We will denote a Löwner matrix with row array S' and column array T' by

$$\mathbf{L}(S', T') := \left(\frac{y_s - y_t}{x_s - x_t} \right)_{s \in I_{S'}, t \in I_{T'}}. \tag{1.21}$$

Given a Löwner matrix $\mathbf{L}(S', T')$ which possesses a non-trivial kernel vector $\mathbf{u} = (u_1, u_2, \ldots, u_{|T'|})^T$, we can attach a rational function to this matrix via

$$r_{\mathbf{u},T'}(x) := \frac{\sum_{j=1}^{|T'|} \frac{u_j}{x - x_j} \cdot y_j}{\sum_{j=1}^{|T'|} \frac{u_j}{x - x_j}}, \tag{1.22}$$

where we assumed that $I_{T'} = \{1, 2, \ldots, |T'|\}$ in order to simplify notation. We know already that this function interpolates all $(x_j, y_j) \in T'$ for which $u_j \neq 0$. Moreover, this function interpolates all points $(x_k, y_k) \in S'$ for which $\sum_{j=1}^{|T'|} \frac{u_j}{x_k - x_j} \neq 0$, since in this case we have by construction that

$$\begin{aligned}
0 &= \sum_{j=1}^{|T'|} u_j \cdot \frac{y_k - y_j}{x_k - x_j} \\
&= y_k \sum_{j=1}^{|T'|} \frac{u_j}{x_k - x_j} - \sum_{j=1}^{|T'|} \frac{u_j}{x_k - x_j} \cdot y_j \qquad (1.23) \\
\Leftrightarrow y_k &= \frac{\sum_{j=1}^{|T'|} \frac{u_j}{x_k - x_j} \cdot y_j}{\sum_{j=1}^{|T'|} \frac{u_j}{x_k - x_j}} = r_{\mathbf{u},T'}(x_k).
\end{aligned}$$

If, on the other hand, $\sum_{j=1}^{|T'|} \frac{u_j}{x_k - x_j} = 0$, then the left hand side of the above equivalence implies that also $\sum_{j=1}^{|T'|} \frac{u_j}{x_k - x_j} \cdot y_j = 0$, which means that (x_k, y_k) is an unattainable point. This result is also provided in [1, Corollary 2.15]. Note that this does not contradict the fact that if (x_j, y_j) is an unattainable point, then $u_j = 0$ for $1 \leq j \leq |T'|$, since the point $(x_k, y_k) \in S'$ is not used in the construction of $r_{\mathbf{u},T'}$.

One of the main results (and for our purpose the most important result) of [1] is the following.

Theorem 1.8 ([1, Main Lemma 2.5]). *Let $S = \{(x_j, y_j) : j = 1, 2, \ldots, M\}$ for $M \in \mathbb{N}$ be a set of samples of a fully reduced rational function r, i.e., $r(x_j) = y_j$ with pairwise distinct x_j for $j = 1, 2, \ldots, M$ and the numerator and denominator polynomials of r are coprime. Further, let S' and T' be a partition of S. If $|S'|, |T'| \geq \deg(r)$ then* $\operatorname{rk} \mathbf{L}(S', T') = \deg(r)$.

Put differently, this means that, given sufficiently many samples of a rational function r and a suitable partition of the sample set, the rank of the associated Löwner matrix encodes the MacMillan degree of r. Recall that a similar result holds for classical rational interpolation, see Remark 1.6 (v). Moreover, it does not matter how the partition of S is chosen, as long as both S' and T' contain at least $\deg(r)$ elements, see [1, Corollary 2.24].

In [1], the results above are proved in the general case of multiple interpolation points, which is similar to Hermite interpolation. This means that it is possible that for some $j \in \{1, 2, \ldots, M\}$ there are several sample values $y_{j,0}, y_{j,1}, \ldots, y_{j,\mu_j}$ corresponding to a single sample point x_j. Here, μ_j is called the *multiplicity* of the point x_j and a rational function r is said to interpolate the points $(x_j, y_{j,k})$ for $j = 1, 2, \ldots, M$ and $k = 0, 1, \ldots, \mu_j$ if and only if $r^{(k)}(x_j) = y_{j,k}$, where $r^{(k)}$ denotes the k-th derivative of r.

An earlier proof of these results in the case of distinct sample points, i.e., $\mu_j = 0$ for all $j = 1, 2, \dots, M$, can be found in [5, Theorem 2]. Another proof in this simpler setting is presented in [58, Theorem 1.1].

It is also shown in [1, Corollary 2.6] that every $\deg(r) \times \deg(r)$ submatrix of $\mathbf{L}(S', T')$ has full rank. Later on we will use a similar result.

Corollary 1.9. *Under the assumptions of Theorem 1.8, any* $\deg(r)$ *rows or columns of the matrix* $\mathbf{L}(S', T')$ *are linearly independent.*

Proof. Let S'' be an arbitrary subset of S' such that $|S''| = \deg(r)$. Then $\mathbf{L}(S'', T')$ has rank $\deg(r)$ by Theorem 1.8. Since the column rank of a matrix is equal to its row rank, it follows that any $\deg(r)$ rows of $\mathbf{L}(S', T')$ are linearly independent.

Now let T'' be an arbitrary subset of T' such that $|T''| = \deg(r)$. Then $\mathbf{L}(S', T'')$ has full rank $\deg(r)$ by Theorem 1.8 and the second claim follows. $\square$

The next lemma will play an important role in later proofs.

Lemma 1.10. *Let* r_1 *and* r_2 *be rational functions with* $\deg(r_1) = \delta_1$ *and* $\deg(r_2) = \delta_2$*. If* r_1 *and* r_2 *agree on* $\delta_1 + \delta_2 + 1$ *points* $x_1, x_2, \dots, x_{\delta_1+\delta_2+1}$*, then* $r_1 = r_2$*.*

Proof. Let $r_1 = \frac{p_1}{q_1}$ and $r_2 = \frac{p_2}{q_2}$, where we assume that for both functions the numerator and denominator polynomials are coprime. Then $\deg(p_1), \deg(q_1) \leq \delta_1$ and $\deg(p_2), \deg(q_2) \leq \delta_2$. We have

$$p_1(x_j) \cdot q_2(x_j) = p_2(x_j) \cdot q_1(x_j) \quad \text{for} \quad j = 1, 2, \dots, \delta_1 + \delta_2 + 1$$

and $\deg(p_1 \cdot q_2), \deg(p_2 \cdot q_1) \leq \delta_1 + \delta_2$. Hence $p_1 \cdot q_2 = p_2 \cdot q_1$, which entails $r_1 = \frac{p_1}{q_1} = \frac{p_2}{q_2} = r_2$. $\square$

Remark 1.11. Recall that we understand equality of rational functions in an algebraic sense, i.e., that in their fully reduced form the numerator and denominator polynomials of rational functions are the same (up to a non-zero scalar).

The following result is the key to reconstructing barycentric rational functions. It can be found in a similar form in [1, Lemma 2.17].

Lemma 1.12. *Under the assumptions of Theorem 1.8, if* $\mathbf{0} \neq \mathbf{u} \in \ker \mathbf{L}(S', T')$*, then* $r_{\mathbf{u},T'} = r$*.*

Proof. First, note that we need to have $|T'| \geq \deg(r) + 1 := \delta + 1$, since otherwise we would have $\operatorname{null} \mathbf{L}(S', T') = 0$, by Theorem 1.8. By construction, $r_{\mathbf{u},T'}$ interpolates $\|\mathbf{u}\|_0$ points in T' and we have $\deg(r_{\mathbf{u},T'}) := \delta_{\mathbf{u},T'} \leq \|\mathbf{u}\|_0 - 1 \leq |T'| - 1$.

Let $r_{\mathbf{u},T'} = \frac{p_{\mathbf{u},T'}}{q_{\mathbf{u},T'}}$. If $q_{\mathbf{u},T'}(x_k) = 0$ for some $(x_k, y_k) \in S'$, then it follows from $\mathbf{u} \in \ker \mathbf{L}(S', T')$ that also $p_{\mathbf{u},T'}(x_k) = 0$, see equation (1.23). But this means that $p_{\mathbf{u},T'}$ and $q_{\mathbf{u},T'}$ have a factor in common, which implies that $\delta_{\mathbf{u},T'} \leq \|\mathbf{u}\|_0 - 2$. In general, if we define $\mu \coloneqq |\{(x_k, y_k) \in S' : q_{\mathbf{u},T'}(x_k) = 0\}|$, then it follows that $\delta_{\mathbf{u},T'} \leq \|\mathbf{u}\|_0 - 1 - \mu$.

By equation (1.23), we also find that $r_{\mathbf{u},T'}$ interpolates $|S'| - \mu$ points in S'. Therefore, $r_{\mathbf{u},T'}$ coincides with r on at least $|S'| - \mu + \|\mathbf{u}\|_0 \geq \delta + \|\mathbf{u}\|_0 - \mu \geq \delta + \delta_{\mathbf{u},T'} + 1$ points. Using Lemma 1.10, we then find $r_{\mathbf{u},T'} = r$. □

Antoulas and Anderson are only interested in the theoretical aspects of parameterizing all minimal degree solutions of the rational interpolation problem. They do not, however, concern themselves with the question of how to actually compute such a solution. This problem is treated in [58], where a straightforward algorithm is proposed.

Algorithm 1.13 (Lagrange Rational Interpolation in Barycentric Form [58, Algorithm 1.1])**.**

Input: $S = \{(x_j, y_j) : j = 1, 2, \ldots, M\}$. A set of $M \geq 2\deg(r) + 1$ samples from a rational function r.

(i) Partition S into S' and T' such that $|S'| = \left\lfloor \frac{M}{2} \right\rfloor$ and $|T'| = \left\lceil \frac{M}{2} \right\rceil$.

(ii) Compute $\delta = \operatorname{rk} \mathbf{L}(S', T')$.

(iii) Choose a new partition S'', T'' with $|S''| = M - \delta - 1$ and $|T''| = \delta + 1$.

(iv) Solve $\mathbf{L}(S'', T'') \cdot \mathbf{u} = \mathbf{0}$.

Output: $r_{\mathbf{u},T''} = r$. A minimal rational function interpolating the samples in S.

Remark 1.14. (i) It follows from Theorem 1.8 that the rank of $\mathbf{L}(S', T')$ in step (ii) coincides with the degree of the rational function from which the data was obtained, i.e., $\delta = \deg(r)$.

(ii) We would like to emphasize that this algorithm assumes that there exists a rational function interpolating the given data, i.e., the set of sample points does not contain any unattainable points. We will consider this problem in the next section.

(iii) The case that the data set contains unattainable points is briefly touched in [58, Section 1.4], but not discussed in depth. A consequence of the presence of unattainable points is that not all $\deg(r) \times \deg(r)$ submatrices of $\mathbf{L}(S', T')$ have full rank anymore.

(iv) It is an important question how to partition the sample set in steps (i) and (iii) of Algorithm 1.13. This problem is shortly discussed in [58, Section 2.1], where it is demonstrated that the chosen partition has an enormous impact on the condition of the reconstruction matrix and, therefore, also the results of the reconstruction. A heuristic approach to improving the condition of the involved Löwner matrices is presented, but the problem is not studied systematically. However, this heuristic approach leads to good results for the reconstruction. The idea is to choose an alternating partition of the sample set. This means that, assuming the sample set $S = \{(x_j, y_j) : j = 1, 2, \ldots, M\}$ is ordered with respect to the sample points, i.e., $x_1 < x_2 < \ldots < x_M$ or $x_1 > x_2 > \ldots > x_M$, which of course assumes that the sample points are real, we choose $S' = \left\{(x_{2\cdot j}, y_{2\cdot j}) : j = 1, 2, \ldots, \left\lfloor \frac{M}{2} \right\rfloor\right\}$ and $T' = \left\{(x_{2\cdot j-1}, y_{2\cdot j-1}) : j = 1, 2, \ldots, \left\lceil \frac{M}{2} \right\rceil\right\}$.

1.4. Barycentric Rational Interpolation with Unattainable Points

Both [1] and [58] are concerned with finding a rational interpolant to some data set where the underlying assumption is that there exists a rational function interpolating all the given data. We, on the other hand, are explicitly dealing with (possibly) distorted data, i.e., data containing unattainable points. Therefore, we need a few more results tailored to this task.

We have seen in equation (1.22) how to attach a barycentric function to a Löwner matrix with non-trivial kernel. Lemma 1.12 also tells us that the rational function thus obtained is actually the function that provided the sample data from which the Löwner matrix was constructed. In this section, we will make extensive use of these results.

As in Section 1.2, we will again consider samples of the form

$$y_j = y_j^* + \tilde{y}_j = r(x_j) + \tilde{y}_j \quad \text{for} \quad j = 1, 2, \ldots, M$$

at pairwise different sample points $x_1, x_2, \ldots, x_M \in \mathbb{C}$, where r is a completely reduced rational function with $\deg(r) = \delta$ such that $r(x_j) = y_j^*$ for $j = 1, 2, \ldots, M$ and we assume that $\tilde{y}_j \neq 0$ exactly ν-times. Further, we will always assume that $M \geq 2\delta + 2\nu + 1$.

The following theorem is our main result regarding the interpolation of distorted data via barycentric rational functions. We have given proofs of similar results, where the given data has some additional structure, in [36, Theorem 3.3] and [90, Theorem 5.1]. To our knowledge, there is no other result linking Löwner matrices built from data containing unattainable points to barycentric rational functions.

Theorem 1.15. *Let $S = \{(x_j, y_j^* + \tilde{y}_j) : j = 1, 2, \ldots, M\}$ be a set of partially distorted samples, where $y_j^* = r(x_j)$ for all $j = 1, 2, \ldots, M$ and a rational function r with $\deg(r) = \delta$. Further, assume that $\tilde{y}_k \neq 0$ for exactly ν indices $k \in \{1, 2, \ldots, M\}$ and suppose that $M \geq 2\delta + 2\nu + 1$. Let S', T' be a partition of S. If $|S'| \geq \delta + \nu$ and $|T'| \geq \delta + \nu + 1$, then $\operatorname{rk} \mathbf{L}(S', T') = \delta + \nu$, i.e., $\operatorname{null} \mathbf{L}(S', T') = |T'| - \delta - \nu \geq 1$ and we have $r_{\mathbf{u},T'} = r$ for any $\mathbf{u} \in \ker \mathbf{L}(S', T')$.*

Proof. (i) Define $D := \{(x_j, y_j^* + \tilde{y}_j) \in S : \tilde{y}_j \neq 0\}$ as well as $D_{S'} := D \cap S'$ and $D_{T'} := D \cap T'$ with $|D_{S'}| = \nu_1$ and $|D_{T'}| = \nu_2$, i.e., $\nu_1 + \nu_2 = \nu$.

As before, denote by $I_{S'}$ and $I_{T'}$ the index sets corresponding to S' and T'. Then we have

$$\mathbf{L}(S', T') = \underbrace{\left(\frac{y_s^* - y_t^*}{x_s - x_t}\right)_{s \in I_{S'}, t \in I_{T'}}}_{=:\mathbf{R}} + \underbrace{\left(\frac{\tilde{y}_s}{x_s - x_t}\right)_{s \in I_{S'}, t \in I_{T'}}}_{=:\mathbf{D}_{S'}} + \underbrace{\left(\frac{-\tilde{y}_t}{x_s - x_t}\right)_{s \in I_{S'}, t \in I_{T'}}}_{=:\mathbf{D}_{T'}}.$$

By Theorem 1.8, $\operatorname{rk} \mathbf{R} = \delta$, i.e., $\operatorname{null} \mathbf{R} = |T'| - \delta \geq \nu + 1$, since $|S'| \geq \delta + \nu$ and $|T'| \geq \delta + \nu + 1$. In particular, any $\delta + 1$ columns of $\mathbf{R}$ are linearly dependent. Therefore, there exist scalars $u_1^{(1)}, u_2^{(1)}, \ldots, u_{\delta+1}^{(1)}, u_1^{(2)}, u_2^{(2)}, \ldots, u_{\delta}^{(|T'|-\delta)}, u_{\delta+1}^{(|T'|-\delta)} \neq 0$ such that the vectors

$$\begin{aligned}
\mathbf{u}^{(1)} &= (u_1^{(1)}, u_2^{(1)}, \ldots, u_{\delta+1}^{(1)}, \underbrace{0, 0, \ldots, 0}_{(|T'|-\delta-1)-\text{times}})^T, \\
\mathbf{u}^{(2)} &= (0, u_1^{(2)}, u_2^{(2)}, \ldots, u_{\delta+1}^{(2)}, \underbrace{0, 0, \ldots, 0}_{(|T'|-\delta-2)-\text{times}})^T, \\
&\vdots \\
\mathbf{u}^{(|T'|-\delta)} &= (\underbrace{0, 0, \ldots, 0}_{(|T'|-\delta-1)-\text{times}}, u_1^{(|T'|-\delta)}, u_2^{(|T'|-\delta)}, \ldots, u_{\delta+1}^{(|T'|-\delta)})^T
\end{aligned}$$

are in the kernel of $\mathbf{R}$. Clearly, the vectors $\mathbf{u}^{(1)}, \mathbf{u}^{(2)}, \ldots, \mathbf{u}^{(|T'|-\delta)}$ are linearly independent. Hence, they form a basis of $\ker \mathbf{R}$. Note that it is necessary that $u_i^{(j)} \neq 0$ for all $i = 1, 2, \ldots, \delta+1$ and $j = 1, 2, \ldots, |T'| - \delta$ since any δ columns of $\mathbf{R}$ are linearly independent by Corollary 1.9.

Further, $\mathbf{D}_{S'}$ contains exactly ν_1 non-zero rows (all other rows only contain zeros) and $\mathbf{D}_{T'}$ contains exactly ν_2 non-zero columns. More precisely, we have

$$\mathbf{D}_{S'} = \operatorname{diag}\big((\tilde{y}_s)_{s\in I_{S'}}\big) \cdot \underbrace{\left(\frac{1}{x_s - x_t}\right)_{s\in I_{S'}, t\in I_{T'}}}_{=:\mathbf{C}(S',T')}$$

and

$$\mathbf{D}_{T'} = \mathbf{C}(S',T') \cdot \operatorname{diag}\big((-\tilde{y}_t)_{t\in I_{T'}}\big),$$

where $\mathbf{C}(S',T')$ is a so-called Cauchy matrix. It is well known that a square Cauchy matrix $\big((a_i - b_j)^{-1}\big)_{i,j=1}^{m}$ constructed from pairwise different values $a_1, a_2, \dots, a_m$, $b_1, b_2, \dots, b_m \in \mathbb{C}$, $m \in \mathbb{N}$, has full rank, see for example [98, Section 7]. Therefore, any $\min\{|S'|, |T'|\} \geq \delta + \nu$ columns or rows of $\mathbf{C}(S',T')$ are linearly independent. Hence, the rows of $\mathbf{D}_{S'}$, respectively the columns of $\mathbf{D}_{T'}$, are linearly independent, i.e., $\operatorname{rk}\mathbf{D}_{S'} = \nu_1$ and $\operatorname{rk}\mathbf{D}_{T'} = \nu_2$. Thus, it follows immediately that

$$\operatorname{rk}\mathbf{L}(S',T') = \operatorname{rk}(\mathbf{R} + \mathbf{D}_{S'} + \mathbf{D}_{T'}) \leq \operatorname{rk}\mathbf{R} + \operatorname{rk}\mathbf{D}_{S'} + \operatorname{rk}\mathbf{D}_{T'} = \delta + \nu_1 + \nu_2 = \delta + \nu$$

and it remains to show that equality holds.

(ii) First, assume that $D \subseteq T'$. Then $\mathbf{D}_{S'} = \mathbf{0}$ and $\mathbf{D}_{T'}$ contains exactly $\nu_2 = \nu$ non-zero columns. Suppose w.l.o.g. that the first ν columns of $\mathbf{D}_{T'}$ are non-zero. Then the vectors $\mathbf{u}^{(\nu+1)}, \mathbf{u}^{(\nu+2)}, \dots, \mathbf{u}^{(|T'|-\delta)}$ are all in the kernel of $\mathbf{L}(S',T') = \mathbf{R} + \mathbf{D}_{T'}$. This follows easily since these vectors are by definition in the kernel of $\mathbf{R}$ and the first ν entries of every vector are zero. Hence, they are also in the kernel of $\mathbf{D}_{T'}$. Therefore, $\operatorname{null}\mathbf{L}(S',T') \geq |T'| - \delta - \nu$.

We now show that there cannot be any more linearly independent vectors in the kernel of $\mathbf{L}(S',T')$. Suppose that there exists a vector $\tilde{\mathbf{u}}$ in the kernel such that γ of its first ν entries are non-zero, i.e., $\tilde{\mathbf{u}} = (\tilde{u}_1, \tilde{u}_2, \dots, \tilde{u}_{|T'|})^T$ and $\tilde{u}_j \neq 0$ for γ indices $1 \leq j \leq \nu$. This vector is certainly linearly independent from $\mathbf{u}^{(\nu+1)}, \mathbf{u}^{(\nu+2)}, \dots, \mathbf{u}^{(|T'|-\delta)}$. By construction, the rational function $r_{\tilde{\mathbf{u}},T'}$ coincides with r on at least $\|\tilde{\mathbf{u}}\|_0 - \gamma$ points in T'. As in the proof of Lemma 1.12, assume that the numerator and denominator polynomials of $r_{\tilde{\mathbf{u}},T'}$ are zero for μ points $x_{j_1}, x_{j_2}, \dots, x_{j_\mu}$ in S'. Then, by equation (1.23), $r_{\tilde{\mathbf{u}},T'}$ interpolates $|S'| - \mu$ points in S' and $\deg(r_{\tilde{\mathbf{u}},T'}) \leq \|\tilde{\mathbf{u}}\|_0 - \mu - 1$. Hence, $r_{\tilde{\mathbf{u}},T'}$ coincides with r on at least $|S'| - \mu + \|\tilde{\mathbf{u}}\|_0 - \gamma \geq \delta + \nu - \gamma + \|\tilde{\mathbf{u}}\|_0 - \mu \geq \delta + \deg(r_{\tilde{\mathbf{u}},T'}) + 1$ points (recall that $\gamma \leq \nu$). By Lemma 1.10, it follows that $r_{\tilde{\mathbf{u}},T'} = r$. But this is a contradiction, since $r_{\tilde{\mathbf{u}},T'}$ interpolates by construction γ unattainable points. Therefore, there cannot exist a kernel vector such that any of its entries corresponding to a non-zero column of $\mathbf{D}_{T'}$ are not 0.

Now consider the matrix $\hat{\mathbf{R}}$ which contains the $(\nu+1)$-st through $|T'|$-th columns of $\mathbf{R}$. This matrix has $|T'| - \nu \geq \delta + 1$ columns and therefore, by Theorem 1.8, $\operatorname{rk}(\hat{\mathbf{R}}) = \delta$, i.e., $\operatorname{null}\hat{\mathbf{R}} = |T'| - \delta - \nu$. By the above argument, a vector $\mathbf{u}$ is in the kernel of $\mathbf{L}(S', T')$ if and only if $\mathbf{u} = (\mathbf{0}^T, \hat{\mathbf{u}}^T)^T$ with $\hat{\mathbf{u}} \in \ker \hat{\mathbf{R}}$. Hence, we must have $\operatorname{null}\mathbf{L}(S', T') = \operatorname{null}\hat{\mathbf{R}} = |T'| - \delta - \nu$.

(iii) Suppose now that the row array S' also contains unattainable points, i.e., $D_{S'} \neq \emptyset$. We have

$$\operatorname{rk}\mathbf{L}(S', T') = \operatorname{rk}(\mathbf{R} + \mathbf{D}_{S'} + \mathbf{D}_{T'}) \leq \operatorname{rk}(\mathbf{R} + \mathbf{D}_{T'}) + \operatorname{rk}\mathbf{D}_{S'}.$$

We have already seen that $\operatorname{rk}\mathbf{D}_{S'} = \nu_1$. Further, (ii) shows that $\operatorname{rk}(\mathbf{R}+\mathbf{D}_{T'}) = \delta+\nu_2$. Showing that the inequality above is indeed an equality requires a little more work. First, we show, using a very similar argument as in (ii), that if a vector is in the kernel of $\mathbf{L}(S', T')$, it must be of the form $(\mathbf{0}^T, \hat{\mathbf{u}}^T)^T$ with $\hat{\mathbf{u}} \in \ker(\hat{\mathbf{R}}+\hat{\mathbf{D}}_{S'})$. Here, we assume w.l.o.g. that only the first ν_2 columns of $\mathbf{D}_{T'}$ are non-zero and $\hat{\mathbf{R}}$, respectively $\hat{\mathbf{D}}_{S'}$, are formed by taking the (ν_2+1)-st through $|T'|$-th columns of $\mathbf{R}$ and $\mathbf{D}_{S'}$ respectively.

For this, suppose, as in (ii), that there exists a vector $\tilde{\mathbf{u}} = (\tilde{u}_1, \tilde{u}_2, \ldots, \tilde{u}_{|T'|})^T \in \ker\mathbf{L}(S', T')$ such that $\tilde{u}_j \neq 0$ for γ indices $1 \leq j \leq \nu_2$. Then $r_{\tilde{\mathbf{u}},T'}$ coincides with r on at least $\|\tilde{\mathbf{u}}\|_0 - \gamma$ points in T'. Assume that the numerator and denominator polynomials of $r_{\tilde{\mathbf{u}},T'}$ are zero for μ points in S'. Then $r_{\tilde{\mathbf{u}},T'}$ interpolates at least $|S'|-\mu$ points in S'. In particular, since there are ν_1 unattainable points in S', it interpolates $|S'| - \mu - \nu_1$ attainable points in S', and $\deg(r_{\tilde{\mathbf{u}},T'}) \leq \|\tilde{\mathbf{u}}\|_0 - \mu - 1$. Therefore, $r_{\tilde{\mathbf{u}},T'}$ agrees with r on at least $|S'| - \mu - \nu_1 + \|\tilde{\mathbf{u}}\|_0 - \gamma \geq \delta + \nu - \nu_1 - \gamma + \|\tilde{\mathbf{u}}\|_0 - \mu \geq \delta + \deg(r_{\tilde{\mathbf{u}},T'}) + 1$ points (recall that $\gamma \leq \nu_2$ and therefore $\nu - \nu_1 - \gamma \geq 0$). Hence, we have, again with Lemma 1.10, that $r_{\tilde{\mathbf{u}},T'} = r$, which is a contradiction, since $r_{\tilde{\mathbf{u}},T'}$ interpolates γ unattainable points, and the claim follows.

Now note that $\hat{\mathbf{R}}+\hat{\mathbf{D}}_{S'} = \mathbf{L}(S', T' \backslash D_{T'})$. Since $|T' \backslash D_{T'}| \geq \delta+\nu+1-\nu_2 = \delta+\nu_1+1$ we still have $\operatorname{rk}\hat{\mathbf{R}} = \delta$. Further, we find that $\mathbf{L}(S', T' \setminus D_{T'})^T = \mathbf{L}(T' \setminus D_{T'}, S')$. Therefore, we can repeat the argument from (ii) verbatim with $(\hat{\mathbf{R}} + \hat{\mathbf{D}}_{S'})^T = \hat{\mathbf{R}}^T + \hat{\mathbf{D}}_{S'}^T$ instead of $\mathbf{R} + \mathbf{D}_{T'}$ to find that $\operatorname{rk}(\hat{\mathbf{R}} + \hat{\mathbf{D}}_{S'})^T = \delta + \nu_1$. Since $\operatorname{rk}(\hat{\mathbf{R}} + \hat{\mathbf{D}}_{S'})^T = \operatorname{rk}(\hat{\mathbf{R}} + \hat{\mathbf{D}}_{S'})$, we then have, similar as in (ii), that $\operatorname{null}\mathbf{L}(S', T') = \operatorname{null}(\hat{\mathbf{R}} + \hat{\mathbf{D}}_{S'}) = |T' \setminus D_{T'}| - \delta - \nu_1 = |T'| - \nu_2 - \delta - \nu_1 = |T'| - \delta - \nu$.

We can even explicitly write down a basis for $\ker\mathbf{L}(S', T')$. Since $\operatorname{rk}\mathbf{D}_{S'} = \nu_1$, any ν_1+1 vectors $\mathbf{D}_{S'}\mathbf{u}^{(j)}$, for $j = 1, 2, \ldots, |T'| - \delta$, are linearly dependent. In particular, there exist scalars $w_1^{(1)}, w_2^{(1)}, \ldots, w_{\nu_1+1}^{(1)}, w_1^{(2)}, w_2^{(2)}, \ldots, w_{\nu_1}^{(|T'|-\delta-\nu_1)}, w_{\nu_1+1}^{(|T'|-\delta-\nu_1)} \neq 0$

such that the vectors

$$\mathbf{w}^{(k)} = \sum_{j=1}^{\nu_1+1} w_j^{(k)} \cdot \mathbf{u}^{(k+j-1)}$$

for $k = 1, 2, \ldots, |T'| - \delta - \nu_1$ are in the kernel of $\mathbf{D}_{S'}$. These vectors are clearly linearly independent and also in the kernel of $\mathbf{R}$. Further, for $k = \nu_2+1, \nu_2+2, \ldots, |T'|-\delta-\nu_1$ they are also in the kernel of $D_{T'}$ (we still assume that only the first ν_2 columns of $D_{T'}$ are non-zero). Therefore, we have found $|T'| - \delta - \nu$ linearly independent vectors in the kernel of $\mathbf{L}(S', T')$.

(iv) Finally, we show that for $\mathbf{u}$ to be in $\ker \mathbf{L}(S', T')$, it is necessary that $\mathbf{u} \in \ker \mathbf{R} \cap \ker \mathbf{D}_{S'} \cap \ker \mathbf{D}_{T'}$. We already showed that we must have $\mathbf{u} \in \ker \mathbf{D}_{T'} \cap \ker(\mathbf{R} + \mathbf{D}_{S'})$, i.e., $\mathbf{u} = (\mathbf{0}^T, \hat{\mathbf{u}}^T)^T$ with $\hat{\mathbf{u}} \in \ker(\hat{\mathbf{R}} + \hat{\mathbf{D}}_{\mathbf{S}'})$, where we use the same notation as above.

To show the claim, we will again employ a similar argument as in the proof of Lemma 1.12. Although we have already used this idea twice in this proof, we will give the full argument for completeness' sake.

If $\hat{\mathbf{u}} \in \ker(\hat{\mathbf{R}} + \hat{\mathbf{D}}_{S'})$, then the rational function $r_{\hat{\mathbf{u}}, T' \setminus D_{T'}} := \hat{r} = \frac{\hat{p}}{\hat{q}}$ interpolates by construction $\|\hat{\mathbf{u}}\|_0$ points in $T' \setminus D_{T'}$ and $|S'| - \mu$ points in S', where μ denotes the number of points $(x_s, y_s) \in S'$ for which $\hat{p}(x_s) = \hat{q}(y_s) = 0$. Hence, since S' contains ν_1 unattainable points, $\hat{r}$ agrees with r on at least $|S'| - \mu - \nu_1 + \|\hat{\mathbf{u}}\|_0 \geq \delta + \nu - \nu_1 + \|\hat{\mathbf{u}}\|_0 - \mu \geq \delta + \deg(\hat{r}) + 1$ points, since $\nu \geq \nu_1$ and $\deg(\hat{r}) \leq \|\hat{\mathbf{u}}\|_0 - \mu - 1$. Therefore, we find with Lemma 1.10 that $\hat{r} = r$. But this implies that $\hat{p}(x_s) = \hat{q}(x_s) = 0$ for all $(x_s, y_s) \in D_{S'}$, which in turn entails that $\hat{\mathbf{u}} \in \ker \hat{\mathbf{D}}_{S'}$. Since $\hat{\mathbf{u}} \in \ker(\hat{\mathbf{R}} + \hat{\mathbf{D}}_{S'})$, we must then also have $\hat{\mathbf{u}} \in \ker \hat{\mathbf{R}}$.

In particular, we have thus shown that every vector $\mathbf{u} \in \mathbf{L}(S', T')$ must also be in the kernel of $\mathbf{R}$ and therefore, by Lemma 1.12, $r_{\mathbf{u}, T'} = r$. □

Corollary 1.16. *If in the setting of Theorem 1.15 we have $|T'| \leq \delta + \nu$ and $|S'| \geq \delta + \nu$, then it follows that* $\operatorname{null} \mathbf{L}(S', T') = 0$. *In particular, we have* $\operatorname{rk} \mathbf{L}(S', T') = \delta + \nu$ *whenever $|S'|, |T'| \geq \delta + \nu$.*

Proof. (i) If there exists a non-trivial vector $\mathbf{u} = (u_1, u_2, \ldots, u_{|T'|})^T \in \ker \mathbf{L}(S', T')$, then $r_{\mathbf{u}, T'}$ has degree at most $\|\mathbf{u}\|_0 - \mu - 1$, where, as in the proof of Theorem 1.15, μ is the number of points in S' corresponding to common zeros of the numerator and denominator polynomial of $r_{\mathbf{u}, T'}$. Further, $r_{\mathbf{u}, T'}$ agrees with r on at least $|S'| - \mu - \nu_1 + \|\mathbf{u}\|_0 - \nu_2 \geq \delta + \nu - \nu_1 - \nu_2 + \|\mathbf{u}\|_0 - \mu = \delta + \deg(r_{\mathbf{u}, T'}) + 1$ points. Hence, by Lemma 1.10, $r_{\mathbf{u}, T'} = r$. This equality also implies that $\mu \geq \nu_1$.

As shown in part (ii) of the proof of Theorem 1.15, all entries in $\mathbf{u}$ corresponding to a non-zero column of $\mathbf{D}_{T'}$ must be zero. Therefore, $\|\mathbf{u}\|_0 \leq |T'| - \nu_2$. We now have that $\deg(r_{\mathbf{u},T'}) \leq \|\mathbf{u}\|_0 - \mu - 1 \leq |T'| - \nu_2 - \nu 1 - 1 \leq \delta - 1$, which is a contradiction to $r_{\mathbf{u},T'} = r$.

(ii) We have seen in Theorem 1.15 that $\operatorname{rk} \mathbf{L}(S', T') = \delta + \nu$ for $|S'| \geq \delta + \nu$ and $|T'| \geq \delta + \nu + 1$. Now we showed the same result in the case that $|T'| = \delta + \nu$. □

Remark 1.17. (i) As we have seen in Theorem 1.15, it suffices to have $M = 2\delta + 2\nu + 1$ samples in order to correctly reconstruct a rational function r of degree δ and where ν samples are distorted. However, in practice this would require that we know δ and ν in advance, since if $|T'| \geq |S'| + 1$ then we always have $\operatorname{null} \mathbf{L}(S', T') \geq 1$. If $|S'|, |T'| < \delta + \nu$, this means that we find a vector $\mathbf{u} \in \ker \mathbf{L}(S', T')$ such that $r_{\mathbf{u},T'} \neq r$. Hence, unless we already know δ and ν, we need a sample set S containing $M \geq 2\delta + 2\nu + 2$ samples such that for every partition S', T' of S with $|S'| = \left\lceil \frac{M}{2} \right\rceil$ and $|T'| = \left\lfloor \frac{M}{2} \right\rfloor$ we get $\operatorname{rk} \mathbf{L}(S', T') = \delta + \nu$ and $r_{\mathbf{u},T'} = r$ for every $\mathbf{u} \in \ker \mathbf{L}(S', T')$. See also Remark 1.6 (iii), where we discuss the analogue result for classical rational interpolation.

(ii) While, in the setting of Theorem 1.15, for every $\mathbf{u} \in \ker \mathbf{L}(S', T')$ we find that $r_{\mathbf{u},T'} = r$ in the sense of equality between rational functions, the representation that we get is $r_{\mathbf{u},T'} = \frac{p^* \cdot d_{D_{S'}}}{q^* \cdot d_{D_{S'}}}$, where, as in Section 1.2, $d_{D_{S'}}(x) = \prod_{k=1}^{|D_{S'}|}(x - x_{j_k})$ for $(x_{j_k}, y^*_{j_k} + \tilde{y}_{j_k}) \in D_{S'}$. In other words, if the data set S contains unattainable points, then it is not guaranteed that the representation of the rational function $r_{\mathbf{u},T'}$ obtained from a vector $\mathbf{u} \in \ker \mathbf{L}(S', T')$, where $|S'| \geq \delta + \nu$ and $|T'| \geq \delta + \nu + 1$, is minimal. We will take a closer look at this when we deal with numerical issues. Note that step (iii) in Algorithm 1.13 guarantees that we get a minimal solution. However, this only works since we assume that the data contains no unattainable points. If there are unattainable points in the given data, then we could only get a minimal solution if $D = D_{T'}$.

(iii) Using the previous two remarks, we can devise a new algorithm for reconstructing polynomials from distorted data assuming that we have $M \geq 2\delta + 2\nu + 2$ samples. First, we employ Algorithm 1.13, where we choose a partition for the Löwner matrix in step (i) of the algorithm according to the first remark here, in order to find a rational function $r_{\mathbf{u},T'}$. Then, we apply steps (iii) through (v) of Algorithm 1.4 in order to reconstruct polynomials p^*, q^* and d^*. However, we will later see a different method, which will allow us to use this algorithm on sample sets containing distorted data.

(iv) Suppose we have a set S containing M samples and a partition S', T' of S with $|S'| = \left\lceil \frac{M}{2} \right\rceil$ and $|T'| = \left\lfloor \frac{M}{2} \right\rfloor$. If M is even and $\operatorname{null} \mathbf{L}(S', T') = 0$, then this means that we have too few samples in order to reconstruct r. If M is odd and $\operatorname{null} \mathbf{L}(S', T') = 0$ then this is a strong sign, although not a guarantee, that we have too few samples. However, as already discussed in Remark 1.6 (iv), if we have too few samples, then it is possible that there is a vector $\mathbf{u} \in \ker \mathbf{L}(S', T')$ such that $r_{\mathbf{u},T'} \neq r$, see Example 1.18.

Example 1.18. Consider the minimal rational function

$$r(x) = \frac{\frac{2}{x} + \frac{3}{x-1} + \frac{1}{x-2} + \frac{4}{x-3}}{\frac{1}{x} + \frac{1}{x-1} + \frac{1}{x-2} + \frac{1}{x-3}} = \frac{10x^3 - 43x^2 + 51x - 12}{4x^3 - 18x^2 + 22x - 6},$$

which we have already seen in Example 1.7. Suppose that we have a set of only $M = 4$ samples $S_1 = \left\{(0,2), \left(\frac{1}{2}, 4\right), (1,3), \left(5, \frac{19}{7}\right)\right\}$, which we partition into $S_1' = \left\{\left(\frac{1}{2}, 4\right), \left(5, \frac{19}{7}\right)\right\}$ and $T_1' = \{(0,2), (1,3)\}$. Then we have

$$\mathbf{L}(S_1', T_1') = \begin{pmatrix} \frac{r\left(\frac{1}{2}\right) - r(0)}{\frac{1}{2}} & \frac{r\left(\frac{1}{2}\right) - r(1)}{\frac{1}{2} - 1} \\ \frac{r(5) - r(0)}{5} & \frac{r(5) - r(1)}{5 - 1} \end{pmatrix} = \begin{pmatrix} 4 & -2 \\ \frac{1}{7} & -\frac{1}{14} \end{pmatrix}.$$

Clearly, we have $\mathbf{u} = (1,2)^T \in \ker \mathbf{L}(S_1', T_1')$ but

$$r_{\mathbf{u},T'}(x) = \frac{\frac{1 \cdot 2}{x} + \frac{2 \cdot 3}{x-1}}{\frac{1}{x} + \frac{2}{x-1}} = \frac{8x - 2}{3x - 1} \neq r(x).$$

This shows that if we have too few samples it is possible that the square Löwner matrix associated to the samples is singular. However, note that, as mentioned in Example 1.7, $\mathbf{L}(S_1', T_1')$ is only singular because of an unfortunate choice of samples. If instead the given data is $S_2 = \{(0,2), (1,3), (2,1), (3,4)\}$, which we partition into $S_2' = \{(0,2), (1,3)\}$ and $T_2' = \{(2,1), (3,4)\}$, then

$$\mathbf{L}(S_2', T_2') = \begin{pmatrix} \frac{r(0) - r(2)}{-2} & \frac{r(0) - r(3)}{-3} \\ \frac{r(1) - r(2)}{1 - 2} & \frac{r(1) - r(3)}{1 - 3} \end{pmatrix} = \begin{pmatrix} -\frac{1}{2} & \frac{2}{3} \\ -2 & \frac{1}{2} \end{pmatrix}$$

has full rank.

Taking two more samples $(2,1)$ and $(3,4)$ and choosing a partition $S_3' = \left\{\left(\frac{1}{2},4\right), \left(5,\frac{19}{7}\right), (3,4)\right\}$ and $T_3' = \{(0,2),(1,3),(2,1)\}$ gives us

$$\mathbf{L}(S_3',T_3') = \begin{pmatrix} \frac{r\left(\frac{1}{2}\right)-r(0)}{\frac{1}{2}} & \frac{r\left(\frac{1}{2}\right)-r(1)}{\frac{1}{2}-1} & \frac{r\left(\frac{1}{2}\right)-r(2)}{\frac{1}{2}-2} \\ \frac{r(5)-r(0)}{5} & \frac{r(5)-r(1)}{5-1} & \frac{r(5)-r(2)}{5-2} \\ \frac{r(3)-r(0)}{3} & \frac{r(3)-r(1)}{3-1} & \frac{r(3)-r(2)}{3-2} \end{pmatrix} = \begin{pmatrix} 4 & -2 & -2 \\ \frac{1}{7} & -\frac{1}{14} & \frac{4}{7} \\ \frac{2}{3} & \frac{1}{2} & 3 \end{pmatrix},$$

which has full rank, as we would expect from Theorem 1.8.

Finally, if we take two further samples $\left(\frac{5}{2},-3\right), \left(4,\frac{72}{25}\right)$ and partition the data into $S_4' = \left\{\left(\frac{1}{2},4\right), \left(\frac{5}{2},-3\right), \left(4,\frac{72}{25}\right), \left(5,\frac{19}{7}\right)\right\}$ and $T_4' = \{(0,2),(1,3),(2,1),(3,4)\}$, we get

$$\mathbf{L}(S_4',T_4') = \begin{pmatrix} \frac{r\left(\frac{1}{2}\right)-r(0)}{\frac{1}{2}} & \frac{r\left(\frac{1}{2}\right)-r(1)}{\frac{1}{2}-1} & \frac{r\left(\frac{1}{2}\right)-r(2)}{\frac{1}{2}-2} & \frac{r\left(\frac{1}{2}\right)-r(3)}{\frac{1}{2}-3} \\ \frac{r\left(\frac{5}{2}\right)-r(0)}{\frac{5}{2}} & \frac{r\left(\frac{5}{2}\right)-r(1)}{\frac{5}{2}-1} & \frac{r\left(\frac{5}{2}\right)-r(2)}{\frac{5}{2}-2} & \frac{r\left(\frac{5}{2}\right)-r(3)}{\frac{5}{2}-3} \\ \frac{r(4)-r(0)}{4} & \frac{r(4)-r(1)}{4-1} & \frac{r(4)-r(2)}{4-2} & \frac{r(4)-r(3)}{4-3} \\ \frac{r(5)-r(0)}{5} & \frac{r(5)-r(1)}{5-1} & \frac{r(5)-r(2)}{5-2} & \frac{r(5)-r(3)}{5-3} \end{pmatrix} = \begin{pmatrix} 4 & -2 & -2 & 0 \\ -2 & -4 & -8 & 14 \\ \frac{11}{50} & -\frac{1}{25} & \frac{47}{50} & -\frac{28}{25} \\ \frac{1}{7} & -\frac{1}{14} & \frac{4}{7} & -\frac{9}{14} \end{pmatrix}.$$

We have $\operatorname{null}\mathbf{L}(S_4',T_4') = 1$ and the kernel is spanned by $(1,1,1,1)^T$, which is exactly the weight vector corresponding to r with respect to T_4'.

Note that if we partition the data differently, we will get a different weight vector. But since we then also have different sample points, we will get the same function in the end. For example, if we choose $S_4'' = \left\{\left(\frac{1}{2},4\right), \left(\frac{5}{2},-3\right), (3,4), \left(5,\frac{19}{7}\right)\right\}$ and $T_4'' = \left\{(0,2),(1,3),(2,1),\left(4,\frac{72}{25}\right)\right\}$, we get

$$\mathbf{L}(S_4'',T_4'') = \begin{pmatrix} \frac{r\left(\frac{1}{2}\right)-r(0)}{\frac{1}{2}} & \frac{r\left(\frac{1}{2}\right)-r(1)}{\frac{1}{2}-1} & \frac{r\left(\frac{1}{2}\right)-r(2)}{\frac{1}{2}-2} & \frac{r\left(\frac{1}{2}\right)-r(4)}{\frac{1}{2}-4} \\ \frac{r\left(\frac{5}{2}\right)-r(0)}{\frac{5}{2}} & \frac{r\left(\frac{5}{2}\right)-r(1)}{\frac{5}{2}-1} & \frac{r\left(\frac{5}{2}\right)-r(2)}{\frac{5}{2}-2} & \frac{r\left(\frac{5}{2}\right)-r(4)}{\frac{5}{2}-4} \\ \frac{r(3)-r(0)}{3} & \frac{r(3)-r(1)}{3-1} & \frac{r(3)-r(2)}{3-2} & \frac{r(3)-r(4)}{3-4} \\ \frac{r(5)-r(0)}{5} & \frac{r(5)-r(1)}{5-1} & \frac{r(5)-r(2)}{5-2} & \frac{r(5)-r(4)}{5-4} \end{pmatrix} = \begin{pmatrix} 4 & -2 & -2 & -\frac{8}{25} \\ -2 & -4 & -8 & \frac{98}{25} \\ \frac{2}{3} & \frac{1}{2} & 3 & -\frac{28}{25} \\ \frac{1}{7} & -\frac{1}{14} & \frac{4}{7} & -\frac{29}{175} \end{pmatrix},$$

the kernel of which is spanned by $\mathbf{u} = (9,8,6,25)^T$. Hence, we get

$$r_{\mathbf{u},T_4''}(x) = \frac{\frac{9\cdot 2}{x} + \frac{8\cdot 3}{x-1} + \frac{6\cdot 1}{x-2} + \frac{72}{x-4}}{\frac{9}{x} + \frac{8}{x-1} + \frac{6}{x-2} + \frac{25}{x-4}} = \frac{10x^3 - 43x^2 + 51x - 12}{4x^3 - 18x^2 + 22x - 6} = r(x),$$

which shows that it does not matter how we partition the given sample set, as long as the row and column array are large enough.

Example 1.19. In this example, we show the effect of adding unattainable points. Consider the minimal rational function

$$r(x) = \frac{\frac{1\cdot 2}{x} + \frac{2\cdot 3}{x-1}}{\frac{1}{x} + \frac{2}{x-1}} = \frac{8x-2}{3x-1},$$

which, as we have seen in Example 1.18, can be reconstructed from the sample set $S = \left\{(0,2), \left(\frac{1}{2}, 4\right), (1,3), \left(5, \frac{19}{7}\right)\right\}$. Now let us see what happens if we distort one sample, which means that $\nu = 1$, and use the sample set $S_1 = \left\{(0,2), \left(\frac{1}{2}, 4\right), (1,3), (5,3)\right\}$, i.e., we replaced the sample $\left(5, \frac{19}{7}\right)$ by $(5,3) = \left(5, r(5) + \frac{2}{7}\right)$. Using the partition $S_1' = \left\{\left(\frac{1}{2}, 4\right), (5,3)\right\}$ and $T_1' = \{(0,2), (1,3)\}$ we get

$$\mathbf{L}(S_1', T_1') = \begin{pmatrix} \frac{4-2}{\frac{1}{2}} & \frac{4-3}{\frac{1}{2}-1} \\ \frac{3-2}{5} & \frac{3-3}{5-1} \end{pmatrix} = \begin{pmatrix} 4 & -2 \\ \frac{1}{5} & 0 \end{pmatrix},$$

which has clearly full rank, showing that in the presence of distorted samples it does not suffice to take just $M = 2\deg(r) + 2$ samples.

We now take two more samples, so in total $M = 2\deg(r) + 2\nu + 2 = 6$ samples, and consider the sample set $S_2 = \left\{(0,2), \left(\frac{1}{2}, 4\right), (1,3), \left(2, \frac{14}{5}\right), \left(3, \frac{11}{4}\right), (5,3)\right\}$, which we first partition into $S_2' = \left\{\left(\frac{1}{2}, 4\right), \left(2, \frac{14}{5}\right), \left(3, \frac{11}{4}\right)\right\}$ and $T_2' = \{(0,2), (1,3), (5,3)\}$, i.e., the distorted sample is in the column array T_2'. This gives us

$$\mathbf{L}(S_2', T_2') = \begin{pmatrix} \frac{4-2}{\frac{1}{2}} & \frac{4-3}{\frac{1}{2}-1} & \frac{4-3}{\frac{1}{2}-5} \\ \frac{\frac{14}{5}-2}{2} & \frac{\frac{14}{5}-3}{2-1} & \frac{\frac{14}{5}-3}{2-5} \\ \frac{\frac{11}{4}-2}{3} & \frac{\frac{11}{4}-3}{3-1} & \frac{\frac{11}{4}-3}{3-5} \end{pmatrix} = \begin{pmatrix} 4 & -2 & -\frac{2}{9} \\ \frac{2}{5} & -\frac{1}{5} & \frac{1}{15} \\ \frac{1}{4} & -\frac{1}{8} & \frac{1}{8} \end{pmatrix}.$$

This matrix has a kernel of dimension 1 spanned by $(1,2,0)^T$, which is exactly the weight vector corresponding to r with respect to T_2'. Note that the last column of $\mathbf{L}(S_2', T_2')$ corresponds to the distorted sample $(5,3)$. Consequently, the last entry of the kernel vector, which is the weight corresponding to $(5,3)$, is 0.

Now let us just flip the row and column array to get the partition $S_2'' = \{(0,2), (1,3), (5,3)\}$ and $T_2'' = \left\{\left(\frac{1}{2}, 4\right), \left(2, \frac{14}{5}\right), \left(3, \frac{11}{4}\right)\right\}$. Then the distorted sample is in the row array and we get

$$\mathbf{L}(S_2'', T_2'') = \begin{pmatrix} \frac{2-4}{-\frac{1}{2}} & \frac{2-\frac{14}{5}}{-2} & \frac{2-\frac{11}{4}}{-3} \\ \frac{3-4}{1-\frac{1}{2}} & \frac{3-\frac{14}{5}}{1-2} & \frac{3-\frac{11}{4}}{1-3} \\ \frac{3-4}{5-\frac{1}{2}} & \frac{3-\frac{14}{5}}{5-2} & \frac{3-\frac{11}{4}}{5-3} \end{pmatrix} = \begin{pmatrix} 4 & \frac{2}{5} & \frac{1}{4} \\ -2 & -\frac{1}{5} & -\frac{1}{8} \\ -\frac{2}{9} & \frac{1}{15} & \frac{1}{8} \end{pmatrix} = \mathbf{L}(S_2', T_2')^T.$$

Since $\mathbf{L}(S_2'', T_2'') = \mathbf{L}(S_2', T_2')^T$ this matrix has a kernel of dimension 1. The kernel is spanned by the vector $\mathbf{u} = (3, -50, 32)^T$, which gives us the rational function

$$\begin{aligned} r_{\mathbf{u},T_2''}(x) &= \frac{\frac{3\cdot 4}{x-\frac{1}{2}} + \frac{-50\cdot\frac{14}{5}}{x-2} + \frac{32\cdot\frac{11}{4}}{x-3}}{\frac{3}{x-\frac{1}{2}} + \frac{-50}{x-2} + \frac{32}{x-3}} \\ &= \frac{8x^2 - 42x + 10}{3x^2 - 16x + 5} \\ &= \frac{(8x-2)\cdot(x-5)}{(3x-1)\cdot(x-5)} \\ &= \frac{p^*(x)\cdot d_{D_{S_2''}}(x)}{q^*(x)\cdot d_{D_{S_2''}}(x)}, \end{aligned}$$

as we would expect from Remark 1.17 (ii).

1.5. The AAA Algorithm

In [80], the so-called *Adaptive Antoulas-Anderson* (or *AAA* for short) algorithm is introduced. The algorithm is further developed in [42] and [81]. However, for our purposes the core AAA algorithm, as presented in [80], is completely sufficient. The authors developed the algorithm as a method for barycentric rational approximation in which iteratively more and more values of a given data set are used for interpolation, while the other values are approximated using a least-squares approach. The algorithm terminates if either a prescribed accuracy or a prescribed maximal degree of the approximant is attained. In this section, we will show that the AAA algorithm is capable of exactly reconstructing a given rational function from partially distorted samples. We already published these results in a similar form in [36, 90]. In the presentation of the AAA algorithm, we closely follow the original source [80] but employ the notation we have used so far.

For now, we will only consider the theoretical aspects of the AAA algorithm. Suppose we are given a set $S = \{(x_j, y_j^* + \tilde{y}_j) : j = 1, 2, \ldots, M\}$ of partially distorted samples, where $y_j^* = r(x_j)$ for all $j = 1, 2, \ldots, M$ and a rational function r with $\deg(r) = \delta$. Further, assume that $\tilde{y}_j \neq 0$ for exactly ν indices $j \in \{1, 2, \ldots, M\}$ and that $M \geq 2\delta + 2\nu + 2$. We will write $y_j = y_j^* + \tilde{y}_j$ for all $j = 1, 2, \ldots, M$.

The iteration process of the AAA algorithm then works as follows. First, we have to initialize the algorithm, i.e., we need to pick the first sample we want to interpolate. There are several options to do this. In [80], the algorithm is initialized using the sample (x_k, y_k), $1 \leq k \leq M$, for which $\left|\frac{1}{M}\sum_{j=1}^{M} y_j - y_k\right|$ is maximal. In [36], we simply used (x_k, y_k) such that $|y_k| = \max_{j=1,2,\ldots,M} |y_j|$. In our experiments we found that the first variant gives slightly better numerical results. Now we set $T^{(1)} := \{(x_k, y_k)\}$ and $S^{(1)} := S \setminus S^{(1)}$, i.e., $T^{(1)}$ contains the sample chosen for interpolation and $S^{(1)}$ contains all other samples.

The general iteration step consists of two parts. Suppose that at step m we are given a set $T^{(m)}$, consisting of m samples which we want to interpolate, and a set $S^{(m)} := S \setminus T^{(m)}$. Denote by

$$I_{T^{(m)}} = \{j \in \{1, 2, \ldots, M\} : (x_j, y_j) \in T^{(m)}\} := \{t_1, t_2, \ldots, t_m\}$$

the index set corresponding to the points in $T^{(m)}$ and define

$$I_{S^{(m)}} := \{s_1, s_2, \ldots, s_{M-m}\}$$

analogously. Our goal is to choose a weight vector $\mathbf{u}^{(m)} = (u_1, u_2, \ldots, u_m)^T$ in such a way that the function

$$r_{\mathbf{u}^{(m)}, T^{(m)}}(x) = \frac{\sum_{k=1}^{m} \frac{u_k}{x - x_{t_k}} \cdot y_{t_k}}{\sum_{k=1}^{m} \frac{u_k}{x - x_{t_k}}}$$

approximates the given data well. In other words, we want to have

$$r_{\mathbf{u}^{(m)}, T^{(m)}}(x_j) = \frac{\sum_{k=1}^{m} \frac{u_k}{x_j - x_{t_k}} \cdot y_{t_k}}{\sum_{k=1}^{m} \frac{u_k}{x_j - x_{t_k}}} \approx y_j \quad \text{for} \quad (x_j, y_j) \in S.$$

Linearizing this equation we aim for

$$\sum_{k=1}^{m} \frac{u_k}{x_j - x_{t_k}} \cdot y_{t_k} \approx y_j \cdot \sum_{k=1}^{m} \frac{u_k}{x_j - x_{t_k}} \quad \text{for} \quad (x_j, y_j) \in S^{(m)}. \tag{1.24}$$

Note that in the linearized equation we only consider $S^{(m)}$ instead of S. This has two reasons. First, $r_{\mathbf{u}^{(m)}, T^{(m)}}$ will generically interpolate all points in $T^{(m)}$ anyway. However, we want to remark that this is not necessarily true, i.e., it is possible that some entries in $\mathbf{u}^{(m)}$ are zero, which means that some points in $T^{(m)}$ are not interpolated. If there are unattainable points in $T^{(m)}$, this behavior is even desired. Second, the functions

$$\sum_{k=1}^{m} \frac{u_k}{x - x_{t_k}} \cdot y_{t_k} \quad \text{and} \quad \sum_{k=1}^{m} \frac{u_k}{x - x_{t_k}}$$

generically have poles at the points in $T^{(m)}$.

It follows from equation (1.24) that we want to choose $\mathbf{u}^{(m)}$ such that it minimizes the error

$$\sum_{l=1}^{M-m} \left| \sum_{k=1}^{m} \frac{u_k \cdot y_{s_l}}{x_{s_l} - x_{t_k}} - \sum_{k=1}^{m} \frac{u_k \cdot y_{t_k}}{x_{s_l} - x_{t_k}} \right|^2 = \sum_{l=1}^{M-m} \left| \sum_{k=1}^{m} u_k \cdot \frac{y_{s_l} - y_{t_k}}{x_{s_l} - x_{t_k}} \right|^2,$$

where we demand additionally that $\left\|\mathbf{u}^{(m)}\right\|_2 := \left(\sum_{k=1}^{m} |u_k|^2\right)^{\frac{1}{2}} = 1$. Since for any $1 \leq l \leq M - m$ we have

$$\sum_{k=1}^{m} u_k \cdot \frac{y_{s_l} - y_{t_k}}{x_{s_l} - x_{t_k}} = \left(\frac{y_{s_l} - y_{t_1}}{x_{s_l} - x_{t_1}}, \frac{y_{s_l} - y_{t_2}}{x_{s_l} - x_{t_2}}, \dots, \frac{y_{s_l} - y_{t_m}}{x_{s_l} - x_{t_m}} \right) \cdot \begin{pmatrix} u_1 \\ u_2 \\ \vdots \\ u_m \end{pmatrix},$$

we can rewrite this minimization problem as

$$\mathbf{u}^{(m)} \in \operatorname*{argmin}_{\substack{\mathbf{u} \in \mathbb{C}^m \\ \|\mathbf{u}\|_2 = 1}} \left\| \mathbf{L}\left(S^{(m)}, T^{(m)}\right) \cdot \mathbf{u} \right\|_2, \tag{1.25}$$

with $\mathbf{L}\left(S^{(m)}, T^{(m)}\right) \in \mathbb{C}^{(M-m)\times m}$ defined as in (1.21). Going forward, we will use the simpler notation

$$\mathbf{L}^{(m)} := \mathbf{L}\left(S^{(m)}, T^{(m)}\right)$$

for the Löwner matrix obtained in the m-th iteration step of the AAA algorithm. The vector $\mathbf{u}^{(m)}$ need not be unique. But even if it is not, this does not change the outcome of the algorithm. If $\mathbf{u}^{(m)}$ is a solution to (1.25) then we write

$$r^{(m)} := r_{\mathbf{u}^{(m)}, T^{(m)}}.$$

Note that this minimization problem only makes sense as long as $\left|S^{(m)}\right| \geq \left|T^{(m)}\right|$, i.e., $m \leq \left\lfloor \frac{M}{2} \right\rfloor$, since otherwise the matrix $\mathbf{L}^{(m)}$ would certainly possess a non-trivial kernel vector, which would then solve the minimization problem.

The second part of the iteration step consists of choosing the next interpolation point. This is done in a greedy manner by finding

$$(x_{t_{m+1}}, y_{t_{m+1}}) \in \underset{(x,y)\in S^{(m)}}{\operatorname{argmax}} \left| r^{(m)}(x) - y \right|. \tag{1.26}$$

Again, this problem might not have a unique solution, which nonetheless does not affect the outcome of the algorithm. Lastly, the row and column arrays are updated via $T^{(m+1)} := T^{(m)} \cup \{(x_{t_{m+1}}, y_{t_{m+1}})\}$ and $S^{(m+1)} := S^{(m)} \setminus \{(x_{t_{m+1}}, y_{t_{m+1}})\}$.

Finally, we need a stopping criterion for the iteration process. We postpone a more detailed discussion of this question to later when we also consider numerical aspects of the AAA algorithm. For the theoretical investigation in this section, we will stop the algorithm once we found a vector $\mathbf{u}^{(m)}$ such that $\mathbf{L}^{(m)} \cdot \mathbf{u}^{(m)} = \mathbf{0}$, or equivalently $\operatorname{null} \mathbf{L}^{(m)} = 1$. Note that this is different from the criteria proposed in [80].

At this point, we have to ask ourselves whether the AAA algorithm terminates in the case that the given data stems from (possibly distorted) samples of a rational function r and whether it will correctly reconstruct r. However, this is readily seen from what we have already proven. As mentioned before, we have given different proofs of this result in [36] and [90], where we assume that the given data has some additional structure.

Theorem 1.20. *Let $S = \{(x_j, y_j^* + \tilde{y}_j) : j = 1, 2, \ldots, M\}$ be a given set of partially distorted samples, where $y_j^* = r(x_j)$ for all $j = 1, 2, \ldots, M$ and a rational function r with $\deg(r) = \delta$. Further, let $\tilde{y}_j \neq 0$ for exactly ν indices $j \in \{1, 2, \ldots, M\}$. If $M \geq 2\delta + 2\nu + 2$, then the AAA algorithm terminates after $\delta + \nu + 1$ iterations and $r^{(\delta+\nu+1)} = r$.*

Proof. By Corollary 1.16, we have $\operatorname{null} \mathbf{L}^{(m)} = 0$ as long as $m \leq \delta + \nu$. By Theorem 1.15, we have $\operatorname{null} \mathbf{L}^{(m)} = 1$ for $m = \delta + \nu + 1$ and $r^{(\delta+\nu+1)} = r$. □

Remark 1.21. Note that the convergence of the AAA algorithms solely relies on the properties of the Löwner matrix it uses. The greedy process for choosing the next interpolation points does not play a role here. However, this process improves the numerical stability of the algorithm, as we will see in the next chapter.

Before we go on to look closer at numerical issues in the next chapter, let us present the AAA algorithm in a clear way.

Algorithm 1.22 (AAA Algorithm for Reconstruction of Rational Functions [80])**.**

Input: $S = \{(x_j, y_j) : j = 1, 2, \dots, M\}$. A set of samples, where $y_j = y_j^* + \tilde{y}_j$ such that there exists a rational function r with $\deg(r) = \delta$ and $y_j^* = r(x_j)$ for all $j = 1, 2, \dots, M$ as well as $\tilde{y}_j \neq 0$ exactly ν times and $M \geq 2\delta + 2\nu + 2$.

```
(i) Initialize the algorithm by choosing a starting point $(x_{t_1}, y_{t_1})$, $1 \leq t_1 \leq M$, for
    example such that $\left|\frac{1}{M}\sum_{j=1}^{M} y_j - y_{t_1}\right|$ is maximal or $|y_{t_1}| = \max_{j=1,2,\dots,M} |y_j|$.
    Set $T^{(1)} = \{(x_{t_1}, y_{t_1})\}$ and $S^{(1)} = S \setminus \{(x_{t_1}, y_{t_1})\}$.
(ii) FOR $m \geq 1$:
    (a) Construct the matrix $\mathbf{L}^{(m)}$ and find
        $$\mathbf{u}^{(m)} \in \operatorname*{argmin}_{\substack{\mathbf{u} \in \mathbb{C}^m \\ \|\mathbf{u}\|_2 = 1}} \left\| \mathbf{L}^{(m)} \cdot \mathbf{u} \right\|_2 .$$
    (b) IF $\mathbf{L}^{(m)} \cdot \mathbf{u}^{(m)} = \mathbf{0}$, then STOP. ELSE choose
        $$(x_{t_{m+1}}, y_{t_{m+1}}) \in \operatorname*{argmax}_{(x,y) \in S^{(m)}} \left| r^{(m)}(x) - y \right|$$
        and set $T^{(m+1)} = T^{(m)} \cup \{(x_{t_{m+1}}, y_{t_{m+1}})\}$ as well as
        $S^{(m+1)} = S^{(m)} \setminus \{(x_{t_{m+1}}, y_{t_{m+1}})\}$.
```

Output: $r^{(\delta+\nu+1)} = r$.

Remark 1.23. (i) If in the iteration process of the algorithm $\mathbf{L}^{(m)} \cdot \mathbf{u}^{(m)} \neq \mathbf{0}$ until $m \geq \left\lfloor \frac{M}{2} \right\rfloor + 1$, then this is a strong indicator that we have too few samples. Therefore, instead of the for-loop in this pseudo code, in our implementation of the AAA algorithm we use a while-loop with the condition that $m \leq \left\lfloor \frac{M}{2} \right\rfloor$, which is equivalent to $|T^{(m)}| \leq |S^{(m)}|$ and return an error if the algorithm does not terminate until this condition is violated. See also Remark 1.17 (iv).

(ii) Note that in this section we explicitly only covered the reconstruction of rational functions and not, as is our eventual goal, the reconstruction of polynomials p^*, q^* and d^* such that $r = \frac{p^*}{q^*}$ is minimal and $d^* = d_D$, where $D = \{(x_j, y_j) \in S : \tilde{y}_j \neq 0\}$. We will come back to this problem a little later.

2. Numerical Aspects of Barycentric Rational Interpolation

So far, all our considerations have been of purely theoretical, i.e., algebraic, nature. Now we turn to more practical considerations and the question of how to solve the rational interpolation, respectively polynomial reconstruction problem, in a numerically stable and efficient way.

2.1. Implementation of the AAA Algorithm

Let us spell out the numerics involved in the AAA algorithm. In this section, we will again closely follow [80]. First, we look at how to construct a Löwner matrix $\mathbf{L}^{(m)}$ efficiently. For this we introduce the matrix

$$\mathbf{C}^{(m)} := \mathbf{C}\left(S^{(m)}, T^{(m)}\right) := \left(\frac{1}{x_s - x_t}\right)_{s\in I_{S^{(m)}}, t\in I_{T^{(m)}}},$$

where, as in the previous chapter, $S^{(m)}$ and $T^{(m)}$ form a partition of a given sample set $S = \{(x_j, y_j) : j = 1, 2, \ldots, M\}$ at the m-th step of the AAA algorithm and $I_{S^{(m)}}$ as well as $I_{T^{(m)}}$ are the corresponding index sets. The matrix $\mathbf{C}^{(m)}$ is a so-called *Cauchy matrix*, since Cauchy studied them in [25]. We have already seen Cauchy matrices in the proof of Theorem 1.15. Further, define $\mathbf{y}_{S^{(m)}} := (y_s)_{s\in I_{S^{(m)}}}$ and $\mathbf{Y}_{S^{(m)}} := \operatorname{diag}(\mathbf{y}_{S^{(m)}})$ as well as $\mathbf{y}_{T^{(m)}} := (y_t)_{t\in I_{T^{(m)}}}$ and $\mathbf{Y}_{T^{(m)}} := \operatorname{diag}(\mathbf{y}_{T^{(m)}})$. We can now write

$$\mathbf{L}^{(m)} = \left(\frac{y_s - y_t}{x_s - x_t}\right)_{s\in I_{S^{(m)}}, t\in I_{T^{(m)}}} = \mathbf{Y}_{S^{(m)}} \cdot \mathbf{C}^{(m)} - \mathbf{C}^{(m)} \cdot \mathbf{Y}_{T^{(m)}}.$$

The minimization problem (1.25) is solved by taking the final right singular vector corresponding to the smallest singular value of a singular value decomposition $\mathbf{L}^{(m)} = \mathbf{U\Sigma V}^*$, where $\mathbf{V}^* = \overline{\mathbf{V}}^T \in \mathbb{C}^{m\times m}$. This means that if $\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_m$ denote the columns of $\mathbf{V}$, i.e., the singular vectors corresponding to the singular values $\sigma_1 \geq$

$\sigma_2 \geq \ldots \geq \sigma_m$, then we take $\mathbf{u}^{(m)} = \mathbf{v}_m$. Note that by definition $\|\mathbf{v}_m\|_2 = 1$. For the singular value decomposition, or SVD for short, see [110, Lectures 4, 5 and 31].

Now lets look at how to compute $r^{(m)}(x_s)$ for all $s \in I_{S(m)}$, which we need in order to find the next interpolation point as in (1.26). For this, we first introduce some notation. Let $\mathbf{v}, \mathbf{w} \in \mathbb{C}^n$ for $n \in \mathbb{N}$ be vectors. Then $\mathbf{v} \odot \mathbf{w}$ and $\mathbf{v} \oslash \mathbf{w}$ denote the pointwise (or elementwise) multiplication, respectively division, of these vectors, i.e., if $\mathbf{v} = (v_1, v_2, \ldots, v_n)^T$ and $\mathbf{w} = (w_1, w_2, \ldots, w_n)^T$, then $\mathbf{v} \odot \mathbf{w} = (v_1 \cdot w_1, v_2 \cdot w_2, \ldots, v_n \cdot w_n)^T$ and $\mathbf{v} \oslash \mathbf{w} = \left(\frac{v_1}{w_1}, \frac{v_2}{w_2}, \ldots, \frac{v_n}{w_n}\right)^T$. Now compute

$$\mathbf{p}^{(m)} := \mathbf{C}^{(m)} \cdot \left(\mathbf{u}^{(m)} \odot \mathbf{y}_{T(m)}\right)$$

and

$$\mathbf{q}^{(m)} := \mathbf{C}^{(m)} \cdot \mathbf{u}^{(m)}.$$

Then $\mathbf{r}^{(m)} := \left(r^{(m)}(x_s)\right)_{s \in I_{S(m)}}$ is given by

$$\mathbf{r}^{(m)} = \mathbf{p}^{(m)} \oslash \mathbf{q}^{(m)}.$$

Finally, we need to look at the stopping criterion. We again defer a more in depth analysis of this topic to Section 2.2 and choose a numerical version of the criterion proposed in the previous section. For a predetermined value $\epsilon > 0$ let the algorithm stop once $\sigma_m(\mathbf{L}^{(m)}) \leq \epsilon$, where $\sigma_m(\mathbf{L}^{(m)})$ denotes the m-th and thereby smallest singular value of $\mathbf{L}^{(m)}$.

Before we state the AAA algorithm in full detail, a quick word on its computational complexity. The factor dominating the computational cost is the computation of the SVDs. At step m we have to compute an SVD of an $(M - m) \times m$ matrix, which has a complexity of $\mathcal{O}\left((M - m)m^2\right) = \mathcal{O}(Mm^2)$, see [110, Lecture 31]. Since the AAA algorithm will (usually) terminate after $\delta + \nu + 1$ steps, where, as always, δ denotes the degree of the rational function we want to reconstruct and ν the number of unattainable points in the given sample set, we get an overall computational complexity of $\mathcal{O}\left(M(\delta + \nu)^3\right)$.

An implementation of the AAA algorithm is included in the *Chebfun* package for Matlab. In our computations, we used our own implementation of the algorithm for various practical reasons, such as to incorporate our stopping criterion from above. The problems discussed here present the main challenges in implementing the AAA algorithm. Of course, much more detailed thought needs to go into an actual implementation, like how to update $\mathbf{C}^{(m)}$ cleverly. For more information on this we refer to [38, 80] and `www.chebfun.org`.

We sum up the findings of this section in a numerically more accurate presentation of the AAA algorithm.

Algorithm 2.1 (Numerical AAA Algorithm for Reconstruction of Rational Functions [80])**.**

Input: $S = \{(x_j, y_j) : j = 1, 2, \ldots, M\}$. A set of samples, where $y_j = y_j^* + \tilde{y}_j$ such that there exists a rational function r with $\deg(r) = \delta$ and $y_j^* = r(x_j)$ for all $j = 1, 2, \ldots, M$ as well as $\tilde{y}_j \neq 0$ exactly ν times and $M \geq 2\delta + 2\nu + 2$.

$\epsilon > 0$. Accuracy for the stopping criterion.

(i) Initialize the algorithm.

  (a) Choose $(x_{t_1}, y_{t_1}) \in S$ such that $\left|\frac{1}{M}\sum_{j=1}^{M} y_j - y_{t_1}\right|$ is maximal.

  (b) Set $T^{(1)} = \{(x_{t_1}, y_{t_1})\}$ and $S^{(1)} = S \setminus \{(x_{t_1}, y_{t_1})\}$.

(ii) WHILE $m \leq \left\lfloor \frac{M}{2} \right\rfloor$:

  (a) Construct $\mathbf{C}^{(m)}$, $\mathbf{y}_{S^{(m)}}$, $\mathbf{Y}_{S^{(m)}}$, $\mathbf{y}_{T^{(m)}}$, $\mathbf{Y}_{T^{(m)}}$ and

$$\mathbf{L}^{(m)} = \mathbf{Y}_{S^{(m)}} \cdot \mathbf{C}^{(m)} - \mathbf{C}^{(m)} \cdot \mathbf{Y}_{T^{(m)}}.$$

  (b) Compute the SVD $\mathbf{L}^{(m)} = \mathbf{U}\boldsymbol{\Sigma}\mathbf{V}^*$.

  (c) Set $\mathbf{u}^{(m)} = \mathbf{v}_m$, the final right singular vector.

  (d) IF $\sigma_m(\mathbf{L}^{(m)}) \leq \epsilon$, then STOP. ELSE compute

$$\begin{aligned}
\mathbf{p}^{(m)} &= \mathbf{C}^{(m)} \cdot \left(\mathbf{u}^{(m)} \odot \mathbf{y}_{T^{(m)}}\right), \\
\mathbf{q}^{(m)} &= \mathbf{C}^{(m)} \cdot \mathbf{u}^{(m)}, \\
\mathbf{r}^{(m)} &= \mathbf{p}^{(m)} \oslash \mathbf{q}^{(m)},
\end{aligned}$$

  and

$$\mathbf{e}^{(m)} = \mathbf{r}^{(m)} - \mathbf{y}_{S^{(m)}} = (e_s)_{s \in I_{S^{(m)}}}.$$

(e) Set

$$t_{m+1} = \operatorname*{argmax}_{s \in I_{S^{(m)}}} |e_s|$$

and update

$$T^{(m+1)} = T^{(m)} \cup \{(x_{t_{m+1}}, y_{t_{m+1}})\}$$

as well as

$$S^{(m+1)} = S^{(m)} \setminus \{(x_{t_{m+1}}, y_{t_{m+1}})\}.$$

(iii) If the algorithm did not terminate while $m \leq \left\lfloor \frac{M}{2} \right\rfloor$, return an error: Too few samples.

Output: $\mathbf{u}^{(\delta+\nu+1)} \in \ker \mathbf{L}^{(m)}$. Vector of barycentric weights for the construction of $r^{(\delta+\nu+1)}$.

$T^{(\delta+\nu+1)}$. Set of sample points and values for the construction of $r^{(\delta+\nu+1)}$.

Remark 2.2. As mentioned in Remark 1.23, the iteration process only makes sense as long as $|S^{(m)}| \geq |T^{(m)}|$, which is equivalent to $m \leq \left\lfloor \frac{M}{2} \right\rfloor$.

2.2. Stopping Criteria for the AAA Algorithm

After postponing this twice already, we finally talk about the choice of stopping criteria for the AAA algorithm. In [80], two different stopping criteria are suggested The first is that a maximal number of iterations $m_{\max}$ is specified beforehand, which will generically (although not necessarily) coincide with the type of the resulting rational approximant. For the second criterion, an approximation accuracy $\epsilon > 0$ is prescribed and the algorithm stops if at the m-th iteration step we have

$$\max_{s \in I_{S^{(m)}}} \left| r^{(m)}(x_s) - y_s \right| \leq \epsilon. \tag{2.1}$$

In the implementation of the AAA algorithm, a relative tolerance is used, given by $\epsilon = \max_{1 \leq j \leq M} |y_j| \cdot \tilde{\epsilon}$ with the default value $\tilde{\epsilon} = 10^{-13}$. We need to be careful though, since if one or more samples are near a pole, then $\max_{1 \leq j \leq M} |y_j|$ and therefore also ϵ can become quite large. This then might lead to the algorithm stopping after only a few steps and far too early, since the maximal error is already below the relative tolerance. This actually happened in some of our tests. However, this problem can be easily tackled by searching for outliers in the sample data and removing them before running the algorithm.

We will now look closer at the stopping criterion used in the preceding section, which is closer to the theory developed in Chapter 1. It follows from Theorem 1.15 that we should stop the algorithm once $\operatorname{null} \mathbf{L}^{(m)} = 1$. Numerically, this condition can be implemented by choosing a threshold $\epsilon > 0$ and stopping the algorithm as soon as the smallest singular value of the Löwner matrix at the m-th step satisfies

$$\sigma_m \left(\mathbf{L}^{(m)} \right) \leq \epsilon, \tag{2.2}$$

as we already suggested in the previous section. This is similar to the way MATLAB's `rank` function, which we actually used in our computations, works, where ϵ is chosen adaptively depending on $\mathbf{L}^{(m)}$.

As before, in the following δ denotes the degree of the rational function we wish to reconstruct and ν the number of unattainable points in the given sample set

$$S = \{(x_j, y_j^* + \tilde{y}_j) : j = 1, 2, \ldots, M\},$$

i.e., $r(x_j) = y_j^*$ for all $j = 1, 2, \ldots, M$ with $\deg(r) = \delta$ and $\tilde{y}_j \neq 0$ for exactly ν indices $j \in \{1, 2, \ldots, M\}$. As always, we write $y_j = y_j^* + \tilde{y}_j$. Further, as in the previous sections, in particular the proof of Theorem 1.15, we define the sets

$$\begin{aligned} D &= \{(x_j, y_j^* + \tilde{y}_j) \in S : \tilde{y}_j \neq 0\}, \\ D_{S^{(m)}} &= D \cap S^{(m)}, \end{aligned}$$

and

$$D_{T^{(m)}} = D \cap T^{(m)},$$

where $S^{(m)}$ and $T^{(m)}$ denote the partition of S obtained by the AAA algorithm in the m-th step. Also as before in (1.8), for any set $\tilde{S} \subseteq S$ with corresponding index set $I_{\tilde{S}} \subseteq \{1, 2, \ldots, M\}$ let $d_{\tilde{S}}(x) = \prod_{j \in I_{\tilde{S}}} (x - x_j)$ with the special case $d_\emptyset \equiv 1$.

There is a crucial difference between the stopping criteria (2.1) and (2.2). Since this difference is of algebraic nature and does not only arise in the numerical context, we will discuss it in a theoretical setting. While $\max_{s \in I_{S^{(m)}}} |r^{(m)}(x_s) - y_s| = 0$ implies that $\operatorname{null} \mathbf{L}^{(m)} \geq 1$, the reverse is not necessarily true. In particular, $\operatorname{null} \mathbf{L}^{(m)} = 1$ need not imply that $\max_{s \in I_{S^{(m)}}} |r^{(m)}(x_s) - y_s| = 0$. Let us look at this more closely.

If at the $(\delta+\nu+1)$-st step of the AAA algorithm we have found a vector $\mathbf{u}^{(\delta+\nu+1)} \in \ker \mathbf{L}^{(\delta+\nu+1)}$, then this implies that $r^{(\delta+\nu+1)} = r$. In particular, we have that

$$r^{(\delta+\nu+1)}(x_j) = y_j^* \quad \text{for all} \quad (x_j, y_j^* + \tilde{y}_j) \in S.$$

Now suppose that $D_{S^{(\delta+\nu+1)}} \neq \emptyset$, i.e., $|D_{S^{(\delta+\nu+1)}}| = \nu_1 \neq 0$. This implies, as we showed in the proof of Theorem 1.15 and demonstrated in Example 1.19, that the numerator and denominator polynomials of $r^{(\delta+\nu+1)}$ have ν_1 zeros in common. Hence,

$$r^{(\delta+\nu+1)} = \frac{p^* \cdot d_{D_{S^{(\delta+\nu+1)}}}}{q^* \cdot d_{D_{S^{(\delta+\nu+1)}}}}$$

for coprime polynomials p^* and q^* such that $r = \frac{p^*}{q^*}$. In other words, $r^{(\delta+\nu+1)}$ is not minimal.

On the other hand, we have

$$\max_{s \in I_{S^{(\delta+\nu+1)}}} \left| r^{(\delta+\nu+1)}(x_s) - y_s \right| = \max_{s \in I_{S^{(\delta+\nu+1)}}} |\tilde{y}_s| \neq 0.$$

Consequently, if we choose stopping criterion (2.1), the AAA algorithm will continue for ν_1 more steps until $D_{S^{(\delta+\nu+\nu_1+1)}} = \emptyset$. This follows from the fact that the greedy selection of interpolation points will only choose distorted samples from step $\delta+\nu+1$ onward, since $r^{(\delta+\nu+n+1)} = r$ for all $n = 0, 1, \ldots, \nu_1$. After $\delta + \nu + \nu_1 + 1$ steps the AAA algorithm will stop and return a weight vector $\mathbf{u}^{(\delta+\nu+\nu_1+1)} \in \ker \mathbf{L}^{(\delta+\nu+\nu_1+1)}$. Note that $\mathbf{u}^{(\delta+\nu+\nu_1+1)} \in \mathbb{C}^{(\delta+\nu+\nu_1+1)}$ and $\dim \ker \mathbf{L}^{(\delta+\nu+\nu_1+1)} = \nu_1 + 1$. As we have seen in the proof of Theorem 1.15, the ν entries of this vector corresponding to the elements of $D_{T^{(\delta+\nu+\nu_1+1)}} = D$ will be zero. It is possible that $\|\mathbf{u}^{(\delta+\nu+\nu_1+1)}\|_0 = \delta + 1$. However, this is very unlikely and we will usually have $\|\mathbf{u}^{(\delta+\nu+\nu_1+1)}\|_0 = \delta + \nu_1 + 1$. But this implies that $r^{(\delta+\nu+\nu_1+1)}$ has type $\delta + \nu_1$, i.e.,

$$r^{(\delta+\nu+\nu_1+1)} = \frac{p^* \cdot d}{q^* \cdot d}$$

for some polynomial d with $\deg(d) = \nu_1$.

These considerations show why stopping criterion (2.2) is more appealing in our context. Occasionally the AAA algorithm chooses all distorted samples within the first $\delta + \nu + 1$ steps. However, in practice this rarely happens. In this case, criterion (2.2) stops the algorithm anyway after $\delta+\nu+1$ steps and it returns a function of type $\delta + \nu_1$ (since the ν_2 weights corresponding to $D_{T^{(\delta+\nu+1)}}$ are zero). Criterion (2.1), on the other hand, makes the algorithm go on for ν_1 more iterations before quite possibly also returning a function of type $\delta + \nu_1$. Note that in the case that $\nu = 0$ the two criteria are theoretically equivalent.

2.3. Froissart Doublets

Now the question naturally arises how to deal with the fact that the AAA algorithm might return a weight vector corresponding to a non-minimal function. First of all, let us give our problem a name. We will call a point $\tilde{x} \in \mathbb{C}$ such that for the representation of a rational function $r = \frac{p}{q}$ we have $p(\tilde{x}) = q(\tilde{x}) = 0$ a *Froissart doublet*, after Marcel Froissart, who first studied this problem systematically in [44]. Froissart doublets have mostly been studied in connection with Padé Approximation, see for example [46, 47, 106]. In [21], they are shortly discussed in connection with barycentric rational interpolation.

In a more general sense, one says that r contains Froissart doublets if it has poles lying very close to zeros, i.e., if there are points $\tilde{x}$ such that

$$q(\tilde{x}) = 0 \quad \text{and} \quad p(\tilde{x} + \epsilon) = 0 \tag{2.3}$$

for some small $|\epsilon| > 0$. The point $\tilde{x}$ is also called a *spurious pole*, or it is said that a rational function has a *spurious pole-zero pair*. Clearly, these points cause problems when performing numerical calculations with rational functions.

We will not go too deep into the theory of Froissart doublets here. Let us just mention that this problem not only occurs in numerical calculations but also in an exact setting. To illustrate this, consider the following example taken from [109]. Suppose we are given sample data $\{(-1, 1+\epsilon), (0, 1), (1, 1+2\epsilon)\}$ for some $\epsilon > 0$. Then

$$r(x) = \frac{(3+4\epsilon)x - 1}{3x - 1}$$

is a minimal rational interpolant to this data. Clearly, r has a pole at $\frac{1}{3}$ and a zero at $\frac{1}{3+4\epsilon}$. The distance of the pole and the zero, however, is only

$$\left|\frac{1}{3} - \frac{1}{3+4\epsilon}\right| = \frac{4\epsilon}{9+12\epsilon} < \frac{\epsilon}{2}.$$

In our case, i.e., if there are unattainable points in the given data set, the common factors in the numerator and denominator would cancel out in an algebraic setting, but numerically we will be in the situation of equation (2.3). If, as in our case, the Froissart doublets arise from numerical computations as opposed to them stemming from the underlying mathematical problem, as in the previous example, this is sometimes emphasized by calling them *numerical Froissart doublets.*

We will consider two ways in which one can deal with numerical Froissart doublets. The first follows [80] and the implementation of the AAA algorithm provided by *Chebfun*. The second is a new idea tailored to our problem at hand.

In order to present the first approach, we need to make a quick detour into complex analysis. Recall that if a function f has a simple pole at a point $c \in \mathbb{C}$ and admits a *Laurent series expansion* in a pointed neighborhood of c, then it can be represented in said neighborhood as

$$f(z) = \sum_{n=-1}^{\infty} a_n \cdot (z-c)^n$$

with coefficients $a_n \in \mathbb{C}$, see [69, Chapter 5]. The uniquely defined number a_{-1} is called the *residue* of f at c and denoted by

$$\operatorname*{Res}_{c} f \coloneqq a_{-1},$$

see [69, Chapter 6].

We can now make the observation that spurious poles of rational functions are characterized by having residues, which are small in absolute value. This can be seen as follows. Let $r = \frac{p}{q}$ be a rational function and assume that $q(\tilde{x}) = 0$ is a simple zero and $p(\tilde{x}+\epsilon) = 0$ for some $|\epsilon| > 0$, i.e., $\tilde{x}$ is a simple pole of r, and write $q(x) = \tilde{q}(x) \cdot (x - \tilde{x})$ and $p(x) = \tilde{p}(x) \cdot (x - \tilde{x} - \epsilon)$. Note that, as polynomials, p and q are entire functions, i.e., holomorphic on the whole complex plane. We now use [69, Lemma 1.3] to obtain

$$\operatorname*{Res}_{\tilde{x}} r = \operatorname*{Res}_{\tilde{x}} \frac{p}{q} = \frac{p(\tilde{x})}{q'(\tilde{x})} = \frac{\tilde{p}(\tilde{x}) \cdot (\tilde{x} - \tilde{x} - \epsilon)}{\tilde{q}'(\tilde{x}) \cdot (\tilde{x} - \tilde{x}) + \tilde{q}(\tilde{x})} = -\epsilon \frac{\tilde{p}(\tilde{x})}{\tilde{q}(\tilde{x})},$$

which, since $\tilde{q}(\tilde{x}) \neq 0$, will usually be very small.

This observation is then used to first locate all Froissart doublets by finding the poles of the function $r^{(m)}$ returned by the AAA algorithm after m steps and computing their residues. We will consider the problem of finding the poles of $r^{(m)}$ in the next section. If the residue of a pole $\tilde{x}$ is smaller than a predetermined threshold, the next step is to find the index of the support point in the row array $T^{(m)}$ which lies closest to it, i.e.,

$$k = \operatorname*{argmin}_{t \in I_{T^{(m)}}} |\tilde{x} - x_t|.$$

Then this point is removed from the data set, i.e., we set $T_1^{(m)} = T^{(m)} \setminus \{(x_k, y_k)\}$. This process is repeated for all poles with sufficiently small residues. If the algorithm detects ν Froissart doublets this means that we end up with a new column array $T_\nu^{(m)}$ with $|T_\nu^{(m)}| = m - \nu$.

Finally, the matrix $\mathbf{L}\Big(S^{(m)}, T_\nu^{(m)}\Big) := \tilde{\mathbf{L}}^{(m)} \in \mathbb{C}^{(M-m)\times(m-\nu)}$ is constructed, an SVD $\tilde{\mathbf{L}}^{(m)} = \mathbf{U}\boldsymbol{\Sigma}\mathbf{V}^*$ is computed and the new weight vector for the rational function is set to $\tilde{\mathbf{u}}^{(m)} = \mathbf{v}_{m-\nu}$ the final right singular vector of $\mathbf{V}$ corresponding to the smallest singular value $\sigma_{m-\nu}\big(\tilde{\mathbf{L}}^{(m)}\big)$.

The threshold used for identifying Froissart doublets in the *Chebfun* implementation of the AAA algorithm is 10^{-13}, which we also used in our implementation.

Now let us look at how this Froissart doublet removal process works together with the two stopping criteria investigated in the previous section. Before the process of removing Froissart doublets starts, all points in $T^{(m)}$ which were assigned the weight 0 are removed. Curiously, in the *Chebfun* implementation of the AAA algorithm, a point is only removed if its weight is exactly zero, which in actual numerical computation will almost never happen. However, for our considerations, let us assume that before the Froissart doublet removal process starts all samples corresponding to nearly zero weights have been discarded, i.e., removed from $T^{(m)}$. We denote this new set by $\tilde{T}^{(m)} := T^{(m)} \setminus D_{T^{(m)}}$. Practically, this is done by determining some threshold and considering all weights with an absolute value below this threshold to be zero. Recall that a weight being zero implies that it corresponds to an unattainable point. However, if the weight is exactly zero, then this unattainable point will not give rise to a Froissart doublet, since it is not considered in the construction of a corresponding rational function. Loosely speaking, such a point is invisible to the function.

We first consider stopping criterion (2.2). For ease of reading we will from now on work in an exact algebraic setting, i.e., we will not consider numerical errors. Then (2.2) means that the algorithm stopped as soon as we reached $\operatorname{null} \mathbf{L}^{(m)} = 1$, which, by Theorem 1.15, implies that $m = \delta + \nu + 1$, where, as always, δ is the degree of the rational function we want to reconstruct and ν is the number of unattainable points in the sample set. We will assume throughout this section that $\nu \neq 0$. As we have seen in the previous section, the rational function we obtain from the kernel vector of $\mathbf{L}^{(\delta+\nu+1)}$ is represented as

$$r^{(\delta+\nu+1)} = \frac{p^* \cdot d_{D_{S^{(\delta+\nu+1)}}}}{q^* \cdot d_{D_{S^{(\delta+\nu+1)}}}}. \tag{2.4}$$

Since the ν_2 unattainable points in $D_{T^{(\delta+\nu+1)}}$ are not considered in the construction of

$r^{(\delta+\nu+1)}$, removing the unattainable points from the column set, i.e., using $\tilde{T}^{(\delta+\nu+1)} = T^{(\delta+\nu+1)} \setminus D_{T^{(\delta+\nu+1)}}$ to construct a new Löwner matrix and compute a weight vector from it, will result in exactly the same function, respectively representation.

The function $r^{(\delta+\nu+1)}$ contains $|D_{S^{(\delta+\nu+1)}}| = \nu_1$ Froissart doublets. If $\nu_1 \neq 0$, the process described above would try to remove these by removing points from $\tilde{T}^{(\delta+\nu+1)}$. Suppose now that ν_1 points are removed resulting in a new column array $\tilde{T}_{\nu_1}^{(\delta+\nu+1)}$. Now, since $|\tilde{T}_{\nu_1}^{(\delta+\nu+1)}| = \delta + \nu + 1 - \nu_1 - \nu_2 = \delta + 1$ and $|S^{(\delta+\nu+1)}| = M - \delta - \nu - 1 \geq 2\delta + 2\nu + 2 - \delta - \nu - 1 = \delta + \nu + 1$, it follows with Corollary 1.16 that $\operatorname{null} \mathbf{L}\Big(S^{(\delta+\nu+1)}, \tilde{T}_{\nu_1}^{(\delta+\nu+1)}\Big) = 0$, since $S^{(\delta+\nu+1)}$ still contains ν_1 unattainable points. This means that the Froissart doublet removal process discussed above does not work in connection with stopping criterion (2.2).

Note that if we were to look for the points closest to a pole of $r^{(\delta+\nu+1)}$ not just in $\tilde{T}^{(\delta+\nu+1)}$ but in the whole set S, then we would find and remove the correct points, namely all points in $D_{S^{(\delta+\nu+1)}}$. Therefore, we could modify the Froissart doublet removal process in this way. However, we will later present a method tailored to stopping criterion (2.2) which does not even require the computation of the poles of $r^{(\delta+\nu+1)}$.

Now let us consider stopping criterion (2.1). As discussed above, this will stop the algorithm after $\delta + \nu + \nu_1 + 1$ steps, where $\nu_1 = |D_{S^{(\delta+\nu+1)}}|$. Moreover, we will have $D_{T^{(\delta+\nu+\nu_1+1)}} = D$, i.e., $S^{(\delta+\nu+\nu_1+1)}$ will not contain any unattainable points. Like before, let $\tilde{T}^{(\delta+\nu+\nu_1+1)} = T^{(\delta+\nu+\nu_1+1)} \setminus D$ with $|\tilde{T}^{(\delta+\nu+\nu_1+1)}| = \delta + \nu_1 + 1$. As mentioned above, $r^{(\delta+\nu+\nu_1+1)}$ will generically contain ν_1 Froissart doublets. Therefore, ν_1 points will be removed from $\tilde{T}^{(\delta+\nu+\nu_1+1)}$ resulting in $|\tilde{T}_{\nu_1}^{(\delta+\nu+\nu_1+1)}| = \delta + 1$. Then neither $S^{(\delta+\nu+\nu_1+1)}$ nor $\tilde{T}_{\nu_1}^{(\delta+\nu+\nu_1+1)}$ contain any unattainable points and it follows with Theorem 1.8 that

$$\operatorname{null} \mathbf{L}\Big(S^{(\delta+\nu+\nu_1+1)}, \tilde{T}_{\nu_1}^{(\delta+\nu+\nu_1+1)}\Big) = 1,$$

which means that the rational function constructed from the weight vector in the kernel of this matrix and $\tilde{T}_{\nu_1}^{(\delta+\nu+\nu_1+1)}$ is exactly the minimal representation of r. Hence, the combination of stopping criterion (2.1) and the Froissart doublet removal process implemented by the standard AAA algorithm will give us the desired outcome.

Note that in theory we could remove any ν_1 points from $\tilde{T}$ and get the same outcome. However, choosing to remove points close to Froissart doublets increases the numerical stability of the procedure.

We have seen that stopping criterion (2.1) in combination with the Froissart doublet removal process implemented in the AAA algorithm gives us the desired outcome. However, we will usually need to compute more SVDs (of larger matrices) for the stopping criterion and the removal process requires the computation of the poles of the obtained rational function and their residuals. Let us now present a Froissart doublet removal process for stopping criterion (2.2), which will only require one very simple computation.

As mentioned above, once the algorithm stops, we get a function which is represented as in equation (2.4). In other words, the numerical Froissart doublets will exactly correspond to the points in $D_{S^{(\delta+\nu+1)}}$. Looking closer at this set we can write

$$D_{S^{(\delta+\nu+1)}} = \{(x_s, y_s) \in S^{(\delta+\nu+1)} : |r^{(\delta+\nu+1)}(x_s) - y_s| = |\tilde{y}_s| > 0\}.$$

The condition defining the set $D_{S^{(\delta+\nu+1)}}$ is easy to check numerically by just replacing the 0 in the inequality above with a suitable ϵ. We can then consider the set $\tilde{S}^{(\delta+\nu+1)} := S^{(\delta+\nu+1)} \setminus D_{S^{(\delta+\nu+1)}}$. Then $\tilde{S}^{(\delta+\nu+1)}$ and $\tilde{T}^{(\delta+\nu+1)}$ do not contain unattainable points and $|\tilde{S}^{(\delta+\nu+1)}| = M - \delta - \nu - \nu_1 - 1 \geq \delta + \nu_2 + 1$.

However, we still have $|\tilde{T}^{(\delta+\nu+1)}| = \delta + \nu_1 + 1$. Hence, we want to remove ν_1 points from $\tilde{T}^{(\delta+\nu+1)}$. Note that the number ν_1 is known at this point, since we already found $D_{S^{(\delta+\nu+1)}}$ and $|D_{S^{(\delta+\nu+1)}}| = \nu_1$. As above, from a mathematical point of view it does not matter which points we remove. In our numerical experiments, we tried two methods. First, we removed the ν_1 samples (x_t, y_t) from $\tilde{T}^{(\delta+\nu+1)}$ for which $|y_t|$ was largest. This is similar to before, since if a point x_t for $t \in I_{\tilde{T}^{(\delta+\nu+1)}}$ is close to a pole of $r^{(\delta+\nu+1)}$, then we would expect y_t to be large. Second, we simply removed the last ν_1 elements from $\tilde{T}^{(\delta+\nu+1)}$, i.e., the elements which were added last. The idea behind this ansatz is that the last elements in $\tilde{T}^{(\delta+\nu+1)}$ are those where the approximation error was smallest. In practice, both ideas lead to the very same outcome. For the set $\tilde{T}_{\nu_1}^{(\delta+\nu+1)}$, we thus obtain $|\tilde{T}_{\nu_1}^{(\delta+\nu+1)}| = \delta + 1$. Therefore, we find

$$\operatorname{null} \mathbf{L}\Big(\tilde{S}^{(\delta+\nu+1)}, \tilde{T}_{\nu_1}^{(\delta+\nu+1)}\Big) = 1,$$

which again lets us reconstruct r exactly.

Let us present our Froissart doublet removal process as an algorithm.

Algorithm 2.3 (Froissart Doublet Removal)**.**

Input: $S^{(\delta+\nu+1)}$, $T^{(\delta+\nu+1)}$. Partition of the sample set S obtained by Algorithm 2.1.
$\mathbf{u}^{(\delta+\nu+1)} = (u_1, u_2, \ldots, u_{\delta+\nu+1})^T$. Weight vector obtained by Algorithm 2.1.
$\epsilon_S, \epsilon_T > 0$. Threshold for the detection of unattainable points in $S^{(\delta+\nu+1)}$ and $T^{(\delta+\nu+1)}$.

(i) Construct $\mathbf{C}^{(\delta+\nu+1)}$, $\mathbf{y}_{S^{(\delta+\nu+1)}}$, $\mathbf{Y}_{S^{(\delta+\nu+1)}}$, $\mathbf{y}_{T^{(\delta+\nu+1)}}$, $\mathbf{Y}_{T^{(\delta+\nu+1)}}$ as in Section 2.1 and compute

$$\begin{aligned}\mathbf{p}^{(\delta+\nu+1)} &= \mathbf{C}^{(\delta+\nu+1)} \cdot \left(\mathbf{u}^{(\delta+\nu+1)} \odot \mathbf{y}_{T^{(\delta+\nu+1)}}\right),\\ \mathbf{q}^{(\delta+\nu+1)} &= \mathbf{C}^{(\delta+\nu+1)} \cdot \mathbf{u}^{(\delta+\nu+1)},\\ \mathbf{r}^{(\delta+\nu+1)} &= \mathbf{p}^{(\delta+\nu+1)} \oslash \mathbf{q}^{(\delta+\nu+1)}\end{aligned}$$

and

$$\mathbf{e}^{(\delta+\nu+1)} = \mathbf{r}^{(\delta+\nu+1)} - \mathbf{y}_{S^{(\delta+\nu+1)}} = (e_s)_{s\in I_{S^{(\delta+\nu+1)}}}.$$

(ii) Set

$$D_{S^{(\delta+\nu+1)}} = \left\{(x_s, y_s) \in S^{(\delta+\nu+1)} : |e_s| > \epsilon_S, s \in I_{S^{(\delta+\nu+1)}}\right\}$$

and

$$D_{T^{(\delta+\nu+1)}} = \left\{(x_{t_j}, y_{t_j}) \in T^{(\delta+\nu+1)} : |u_j| < \epsilon_T, 1 \le j \le \delta+\nu+1\right\}$$

as well as

$$\tilde{S}^{(\delta+\nu+1)} = S^{(\delta+\nu+1)} \setminus D_{S^{(\delta+\nu+1)}} \quad \text{and} \quad \tilde{T}^{(\delta+\nu+1)} = T^{(\delta+\nu+1)} \setminus D_{T^{(\delta+\nu+1)}}.$$

(iii) Remove $\nu_1 = |D_{S^{(\delta+\nu+1)}}|$ points from $\tilde{T}^{(\delta+\nu+1)}$ to obtain $\tilde{T}_{\nu_1}^{(\delta+\nu+1)}$.

(iv) Construct $\mathbf{C}\left(\tilde{S}^{(\delta+\nu+1)}, \tilde{T}_{\nu_1}^{(\delta+\nu+1)}\right)$, $\mathbf{y}_{\tilde{S}^{(\delta+\nu+1)}}$, $\mathbf{Y}_{\tilde{S}^{(\delta+\nu+1)}}$, $\mathbf{y}_{\tilde{T}_{\nu_1}^{(\delta+\nu+1)}}$, $\mathbf{Y}_{\tilde{T}_{\nu_1}^{(\delta+\nu+1)}}$ and

$$\begin{aligned}\mathbf{L}\left(\tilde{S}^{(\delta+\nu+1)}, \tilde{T}_{\nu_1}^{(\delta+\nu+1)}\right) = \mathbf{Y}_{\tilde{S}^{(\delta+\nu+1)}} &\cdot \mathbf{C}\left(\tilde{S}^{(\delta+\nu+1)}, \tilde{T}_{\nu_1}^{(\delta+\nu+1)}\right)\\ &- \mathbf{C}\left(\tilde{S}^{(\delta+\nu+1)}, \tilde{T}_{\nu_1}^{(\delta+\nu+1)}\right) \cdot \mathbf{Y}_{\tilde{T}_{\nu_1}^{(\delta+\nu+1)}}.\end{aligned}$$

(v) Compute the SVD $\mathbf{L}\left(\tilde{S}^{(\delta+\nu+1)}, \tilde{T}_{\nu_1}^{(\delta+\nu+1)}\right) = \mathbf{U\Sigma V}^*$.

(vi) Set $\tilde{\mathbf{u}} = \mathbf{v}_{\delta+1}$ the final right singular vector.

Output: $\tilde{\mathbf{u}} \in \ker \mathbf{L}\left(\tilde{S}^{(\delta+\nu+1)}, \tilde{T}_{\nu_1}^{(\delta+\nu+1)}\right)$. Vector of barycentric weights to construct r.
$\tilde{T}_{\nu_1}^{(\delta+\nu+1)}$. Set of sample points and values for the construction of r.

Remark 2.4. As already hinted at in Remark 1.17, this process can also be used in connection with Algorithm 1.13, which means that this algorithm can be employed for data containing unattainable points.

2.4. Zeros and Poles of Rational Functions

Finding the poles of a rational function is not only of importance for the Froissart doublet removal process as proposed in the original AAA algorithm, it is a topic which is also interesting in its own right and the key to reconstructing the numerator and denominator polynomials. In fact, later on, we will be faced with the problem of finding the poles of a fully reduced rational function. Since finding the poles of a rational function means finding the zeros of its denominator polynomial, the problem of finding the zeros of such a function, i.e., the zeros of the numerator polynomial, can be dealt with in the same fashion.

We again follow [80], which refers to [64], and consider the procedure suggested there. Note, however, that the method we will be looking at here was not developed in [64]. But more on this later. First, we have to introduce two concepts.

Let $\mathbf{A}, \mathbf{B} \in \mathbb{C}^{m \times n}$. For $x \in \mathbb{C}$, the expression

$$\mathbf{A} - x \cdot \mathbf{B}$$

is called a *(linear) matrix pencil* and denoted by $(\mathbf{A}, \mathbf{B})$. If $m = n$, i.e., $\mathbf{A}$ and $\mathbf{B}$ are square matrices, and if $\det(\mathbf{A} - x \cdot \mathbf{B}) \not\equiv 0$, i.e., there exist values $\tilde{x} \in \mathbb{C}$ for which $\det(\mathbf{A} - \tilde{x} \cdot \mathbf{B}) \neq 0$, then the matrix pencil is called *regular*. For a regular matrix pencil $(\mathbf{A}, \mathbf{B})$, the values $\hat{x} \in \mathbb{C}$ for which $\det(\mathbf{A} - \hat{x} \cdot \mathbf{B}) = 0$ are called the *eigenvalues* of the pencil and the problem of finding them is called a *generalized eigenvalue problem*. If $\mathbf{B} = \mathbf{I}$, where $\mathbf{I} \in \mathbb{C}^{n \times n}$ denotes the identity matrix, then this is equivalent to the usual eigenvalue problem. More on matrix pencils can be found in [45, chapter 12] and [67].

Note that conventionally matrix pencils are written in the form $\mathbf{A} - x \cdot \mathbf{B}$, while eigenvalue problems are formulated as $x \cdot \mathbf{I} - \mathbf{A}$. However, since $\det(x \cdot \mathbf{I} - \mathbf{A}) = (-1)^n \cdot \det(\mathbf{A} - x \cdot \mathbf{I})$ both formulations are equivalent. For the rest of this section we will follow the convention for eigenvalue problems.

The other concept we need to look at are so-called *companion matrices*. Given a monic polynomial p, i.e., a polynomial with leading coefficient 1, with $\deg(p) = n$, the companion matrix of p is the matrix $\mathbf{C}_p \in \mathbb{C}^{n \times n}$ such that p is the characteristic and minimal polynomial of $\mathbf{C}_p$, i.e., $p(x) = \det(x \cdot \mathbf{I} - \mathbf{C}_p)$. In particular, the zeros of p are the eigenvalues of $\mathbf{C}_p$, see [56, Section 3.3]. We will use this idea in order to compute the zeros and poles of a rational function given in barycentric form.

Given a polynomial $p(x) = \sum_{j=0}^{n} p_j \cdot x^j$ with $p_j \in \mathbb{C}$ for all $j = 0, 1, \ldots, n$ and $p_n = 1$, we have

$$\mathbf{C}_p = \begin{pmatrix} 0 & 1 & & & \\ & 0 & 1 & & \\ & & \ddots & \ddots & \\ & & & 0 & 1 \\ -p_0 & -p_1 & \ldots & -p_{n-2} & -p_{n-1} \end{pmatrix} \in \mathbb{C}^{n \times n},$$

where every blank spot corresponds to a 0 entry. It can be easily seen that in fact $p(x) = \det(x \cdot \mathbf{I} - \mathbf{C}_p)$, for example, by computing the Laplace expansion of $x \cdot \mathbf{I} - \mathbf{C}_p$ along the last row [95, Section 4.2].

If p is not monic, i.e., $p_n \neq 1$, then we find that $p(x) = \det(x \cdot \mathbf{B}_p - \mathbf{C}_p)$, where

$$\mathbf{B}_p = \begin{pmatrix} 1 & & & \\ & \ddots & & \\ & & 1 & \\ & & & p_n \end{pmatrix} \in \mathbb{C}^{n \times n},$$

see [48, Section 7.2]. This means that while finding the zeros of a monic polynomial corresponds to a classical eigenvalue problem, finding the zeros of an arbitrary polynomial leads to a generalized eigenvalue problem. Therefore, the matrix pencil $(\mathbf{C}_p, \mathbf{B}_p)$ is also called a *generalized companion matrix pencil* [29].

Usually, the polynomial p is considered to be represented in the monomial basis, as above. However, this is not necessary and the form of the companion matrix depends on the basis in which p is written, see for example [3, 50]. Now let us consider a polynomial p written in the first barycentric form as in (1.12), i.e.,

$$p(x) = \ell(x) \sum_{j=1}^{n} \frac{w_j}{x - x_j} \cdot y_j, \tag{2.5}$$

where $x_j, y_j \in \mathbb{C}$ and $w_j = \prod_{\substack{i=1 \\ i \neq j}}^{M} \frac{1}{x_j - x_i}$ as in (1.11) for $j = 1, 2, \ldots, n$ as well as $\ell(x) = \prod_{i=1}^{M} (x - x_i)$ as in (1.10). It has been shown in [28], which references the earlier but unpublished [30] and has later been presented in a more concise way in [29], that the zeros of p can be directly computed using the given parameters x_j, y_j and w_j for

$j = 1, 2, \dots, n$ as finite eigenvalues of the generalized companion matrix pencil

$$x \cdot \underbrace{\begin{pmatrix} 1 & & & & \\ & 1 & & & \\ & & \ddots & & \\ & & & 1 & \\ & & & & 0 \end{pmatrix}}_{=:\mathbf{B}_p^L} - \underbrace{\begin{pmatrix} x_1 & & & & y_1 \\ & x_2 & & & y_2 \\ & & \ddots & & \vdots \\ & & & x_n & y_n \\ w_1 & w_2 & \dots & w_n & 0 \end{pmatrix}}_{=:\mathbf{C}_p^L} \in \mathbb{C}^{(n+1)\times(n+1)}. \tag{2.6}$$

The L in the superscript of the matrices stands for Lagrange, since this is a matrix pencil in the Lagrange basis. Again, it is easily verified that $p(x) = \det(x \cdot \mathbf{B}_p^L - \mathbf{C}_p^L)$ by computing the Laplace expansion of this pencil along the last row. The numerical stability of this procedure has been investigated in [70].

Note that the zeros of p are the zeros of $\sum_{j=1}^n \frac{w_j}{x-x_j} \cdot y_j$, while, by construction, the zeros of $\ell(x)$ are not zeros of p. Considering the generalized eigenvalue problem

$$\mathbf{C}_p^L \cdot \mathbf{v} = \lambda \cdot \mathbf{B}_p^L \cdot \mathbf{v},$$

we find that if $\hat{x}$ is a zero of p, and hence of $\sum_{j=1}^n \frac{w_j}{x-x_j} \cdot y_j$, the corresponding eigenvector is given by $\left(\frac{y_1}{\hat{x}-x_1}, \frac{y_2}{\hat{x}-x_2}, \dots, \frac{y_n}{\hat{x}-x_n}, 1\right)$. This can be seen directly by computing

$$\begin{aligned} \begin{pmatrix} x_1 & & & y_1 \\ & \ddots & & \vdots \\ & & x_n & y_n \\ w_1 & \dots & w_n & 0 \end{pmatrix} \cdot \begin{pmatrix} \frac{y_1}{\hat{x}-x_1} \\ \vdots \\ \frac{y_n}{\hat{x}-x_n} \\ 1 \end{pmatrix} &= \begin{pmatrix} \frac{x_1 \cdot y_1}{\hat{x}-x_1} + y_1 \\ \vdots \\ \frac{x_n \cdot y_n}{\hat{x}-x_n} + y_n \\ \sum_{j=1}^n \frac{w_j}{\hat{x}-x_j} \cdot y_j \end{pmatrix} = \begin{pmatrix} \frac{y_1 \cdot (x_1 + \hat{x} - x_1)}{\hat{x}-x_1} \\ \vdots \\ \frac{y_n \cdot (x_n + \hat{x} - x_n)}{\hat{x}-x_n} \\ 0 \end{pmatrix} \\ &= \hat{x} \cdot \begin{pmatrix} \frac{y_1}{\hat{x}-x_1} \\ \vdots \\ \frac{y_n}{\hat{x}-x_n} \\ 0 \end{pmatrix} = \hat{x} \cdot \begin{pmatrix} 1 & & & \\ & \ddots & & \\ & & 1 & \\ & & & 0 \end{pmatrix} \cdot \begin{pmatrix} \frac{y_1}{\hat{x}-x_1} \\ \vdots \\ \frac{y_n}{\hat{x}-x_n} \\ 1 \end{pmatrix}. \end{aligned} \tag{2.7}$$

Now observe that $p(x) = \det(x \cdot \mathbf{B}_p^L - \mathbf{C}_p^L)$ is only true if p is exactly of the form as in (2.5), in particular, $w_j = \prod_{\substack{i=1 \\ i \neq j}}^M \frac{1}{x_j - x_i}$. Equation (2.7), on the other hand, holds true for any weights $u_1, u_2, \dots, u_n$. This means that the finite eigenvalues of a matrix pencil as in (2.7) can give us the zeros of any function $\sum_{j=1}^n \frac{u_j}{x-x_j} \cdot y_j$ in partial fraction form.

Given a rational function in barycentric form,

$$r(x) = \frac{\sum_{j=1}^n \frac{u_j}{x-x_j} \cdot y_j}{\sum_{j=1}^n \frac{u_j}{x-x_j}} =: \frac{p(x)}{q(x)},$$

it is now easy to see that its zeros can be computed by solving the generalized eigenvalue problem $\mathbf{C}_p^L \cdot \mathbf{v} = \lambda \cdot \mathbf{B}_p^L \cdot \mathbf{v}$, where $\mathbf{C}_p^L$ is defined as in (2.6) with the parameters x_j, y_j and u_j for $j = 1, 2, \dots, n$. The poles of r are then given by the zeros of q which can be similarly computed with $\mathbf{C}_q^L$ defined by the parameters x_j, u_j and $y_j = 1$ for $j = 1, 2, \dots, n$.

We mentioned before that the finite eigenvalues of the generalized companion matrix pencil give us the zeros of a polynomial. This statement implies that the matrix pencil also possesses infinite eigenvalues and indeed it does. In particular, a matrix pencil as in (2.6) has two eigenvalues at $\pm\infty$ and all other $n-1$ eigenvalues are finite. The infinite eigenvalues can be understood in a heuristic way as follows. If in (2.7) we let $\hat{x} \to \pm\infty$, then we get

$$\lim_{\hat{x}\to\pm\infty} \begin{pmatrix} x_1 & & & y_1 \\ & \ddots & & \vdots \\ & & x_n & y_n \\ w_1 & \dots & w_n & 0 \end{pmatrix} \cdot \begin{pmatrix} \frac{y_1}{\hat{x}-x_1} \\ \vdots \\ \frac{y_n}{\hat{x}-x_n} \\ 1 \end{pmatrix} = \lim_{\hat{x}\to\pm\infty} \hat{x} \cdot \begin{pmatrix} \frac{y_1}{\hat{x}-x_1} \\ \vdots \\ \frac{y_n}{\hat{x}-x_n} \\ 0 \end{pmatrix}.$$

Taking the limit separately inside and outside of the vector on the right hand side of this equation, this becomes

$$\begin{pmatrix} x_1 & & & y_1 \\ & \ddots & & \vdots \\ & & x_n & y_n \\ w_1 & \dots & w_n & 0 \end{pmatrix} \cdot \begin{pmatrix} 0 \\ \vdots \\ 0 \\ 1 \end{pmatrix} = \pm\infty \cdot \begin{pmatrix} 0 \\ \vdots \\ 0 \\ 0 \end{pmatrix} = \pm\infty \cdot \begin{pmatrix} 1 & & & \\ & \ddots & & \\ & & 1 & \\ & & & 0 \end{pmatrix} \cdot \begin{pmatrix} 0 \\ \vdots \\ 0 \\ 1 \end{pmatrix}.$$

This means that the eigenvector $(0, 0, \dots, 0, 1)^T$ corresponds to the two eigenvalues $\pm\infty$. A more detailed discussion of the infinite eigenvalues can be found in [28].

Another heuristic way to understand the infinite eigenvalues, is by invoking their link to the zeros of a function in partial fraction form and noting that

$$\lim_{\hat{x}\to\pm\infty} \sum_{j=1}^n \frac{u_j}{\hat{x}-x_j} \cdot y_j = 0.$$

Finally, if we know the zeros and poles of a rational function, we know the zeros of its numerator and denominator polynomial, which allows us to reconstruct them up to a scaling factor.

Remark 2.5. (i) In [60], the poles of a rational function are also computed as the eigenvalues of a matrix pencil. However, the generalized eigenvalue problem constructed there is completely different from the one discussed here.

(ii) In order to solve a generalized eigenvalue problem, usually the so-called QZ Method, see [49, Section 7.7], is employed. This is also the standard method used by MATLAB's `eig` function.

2.5. Numerical Examples

Now let us put all the theoretical considerations so far into practice. In this section, we will compare the performance of the classical Vandermonde approach with that of Algorithm 1.13, which we will from now on call Ioniță's algorithm, and the AAA algorithm. For the AAA algorithm, we will also compare the different stopping criteria and Froissart doublet removal processes.

All algorithms are implemented in MATLAB and wherever possible we make use of MATLAB's internal routines or preexisting code. For example, instead of choosing a fixed threshold and saying that a matrix has become singular once its smallest singular value lies below this threshold, as proposed in stopping criterion (2.2), we use MATLAB's `rank` function, which adaptively sets the threshold value for every matrix anew. We also make use of *Chebfun* functions such as `reval` and `prz` to evaluate a rational function given in barycentric form and compute its poles, residues, and zeros. Note that `prz` implements the methods for finding zeros and poles discussed in Section 2.4.

However, as mentioned before, we do not use the *Chebfun* implementation of the AAA algorithm. The main reason for this is that using our own implementation makes it easier to adapt it to our needs and implement, for example, our stopping criterion (2.2) and the corresponding Froissart doublet removal process. For ease of reading we will refer to stopping criterion (2.1) as the error criterion and to (2.2) as the rank criterion.

For Ioniță's algorithm, as noted earlier in Remark 1.14 (iii), we need to decide on how to partition the sample set. We tried three types of partitioning. Given M samples and assuming that the sample points are ordered, i.e., $x_1 < x_2 < \ldots < x_M$ these are a linear partition, in which the column array T' contains the first $\left\lfloor \frac{M}{2} \right\rfloor$ samples, a random partition, in which T' contains $\left\lfloor \frac{M}{2} \right\rfloor$ randomly chosen samples, and an alternating partition, in which T' contains every other sample, i.e., $(x_1, y_1), (x_3, y_3), \dots, (x_{2\cdot\lfloor M/2\rfloor}, y_{2\cdot\lfloor M/2\rfloor})$. In each case the row array S' contains all other samples.

In [58], it is recommended to use an alternating partition and shortly discussed why this is beneficial. Our experiments confirm that the alternating partition gives the best results. For all comparisons in this section, we therefore use Ioniță's algorithm with an alternating partition.

The implementation of the Vandermonde algorithm we use here differs from Algorithm 1.4. Instead of taking any kernel vector from the reconstruction matrix and then computing the greatest common divisor of the numerator and denominator polynomial in order to get a unique solution, which is famously ill-posed, see [82, 86, 105], we use a method inspired by Ioniță's algorithm. First we build the matrix $\mathbf{V}_{\lfloor M/2\rfloor,\lfloor M/2\rfloor}(S)$ as in equation (1.3), where S denotes the given sample set of size M, and compute its rank. By Remark 1.6 (v), we know that $\operatorname{rk}\mathbf{V}_{\lfloor M/2\rfloor,\lfloor M/2\rfloor}(S) = \lfloor M/2\rfloor + 1 + \deg(r)$ and hence $\deg(r) = \operatorname{rk}\mathbf{V}_{\lfloor M/2\rfloor,\lfloor M/2\rfloor}(S) - \lfloor M/2\rfloor - 1$. We then build $\mathbf{V}_{\deg(r),\deg(r)}(S)$ which, by Theorem 1.2, has a kernel of dimension 1. Note that this assumes that there are no unattainable points in the sample set. However, in the following, we will not use the Vandermonde algorithm for data with unattainable points anyway.

2.5.1. Test Data and Parameters

All experiments reported here are carried out by collecting data from 10.000 randomly created rational functions $r = \frac{p}{q}$. These functions are constructed by first fixing the degree of the denominator polynomial q, which will then also be the degree of r. For each sample function, the degree of the numerator polynomial p is chosen randomly in the range $2 \leq \deg(p) \leq \deg(q)$.

Next we fix the range in which the zeros of p and q should lie, i.e., we choose a number r_{zer} such that the zeros lie in the interval $[-r_{\text{zer}}, r_{\text{zer}}]$. In particular, this means that we only consider real polynomials. Then we uniformly draw $\deg(p)$, respectively $\deg(q)$, numbers from this interval. We make sure that we never draw the same number twice for any polynomial in order to ensure that the rational function does not have multiple zeros or poles as well as that the sets of zeros of p and q have no elements in common so that p and q are guaranteed to be coprime. The polynomials p and q are than created using MATLAB's `poly` command.

Finally, we build the sample data. For this, we have to fix two more parameters, namely the number of unattainable, or distorted, points n_{dist} contained in the sample set and the number of additional sample points n_{add}. As we mentioned in Remark 1.6 (iv) and Remark 1.17 (i), we need at least $2 \cdot (\deg(r) + n_{\text{dist}} + 1)$ values in order to reconstruct a function r from distorted sample values. The number n_{add} determines how many more samples than that we have given. We therefore set $M = 2 \cdot (\deg(r) + n_{\text{dist}} + 1) + n_{\text{add}}$ and simply choose the sample points to be $1, 2, \ldots, M$. In order to avoid very bad reconstruction results, we check whether some of the sample points are close to a pole. More precisely, we make sure that the distance between a sample point and the nearest pole is always at least 10^{-15}. We then evaluate r at the sample points. If $n_{\text{dist}} = 0$, there is nothing more to do. Otherwise, we randomly choose n_{dist} indices and add a distortion to the corresponding sample values. For the amplitude of the distortion we first compute the median $\bar{y}$ of all sample values from the given function and then randomly choose the distortion to lie in the set $[-1.5\cdot\bar{y}, -0.5\cdot\bar{y}] \cup [0.5\cdot\bar{y}, 1.5\cdot\bar{y}]$. Summing up, the parameters determining the sample data are:

1. The degree of the denominator polynomial $\deg(q)$, which coincides with the degree of the rational function $\deg(r)$.
2. The range of the zeros of the polynomials r_{zer}.
3. The number of additional points n_{add}.
4. The number of distorted points in the sample set n_{dist}.

2.5.2. Condition Numbers of Reconstruction Matrices

Let us start our investigation by looking at the condition numbers of the reconstruction matrices. Of course, since our goal in every algorithm is to find a kernel vector of the reconstruction matrix, it makes no sense to talk about the condition numbers of these very matrices. In order to talk sensibly about condition numbers, we use the following observations.

Let $\mathbf{A}$ be the reconstruction matrix with $\operatorname{null} \mathbf{A} = 1$ obtained from any of the three algorithms. Further, define $\mathbf{A}^{(m \times n)}$ as the submatrix containing the first m rows and the first n columns of $\mathbf{A}$. If $\mathbf{A}$ is a Löwner matrix, i.e., if we used the AAA or Ioniță's algorithm, then it follows from Theorem 1.8 and [1, Corollary 2.6] that $\mathbf{A}^{(\operatorname{rk}\mathbf{A} \times \operatorname{rk}\mathbf{A})}$ has full rank.

Now consider $\mathbf{A} = \mathbf{V}_{\deg(r),\deg(r)}(S) \in \mathbb{C}^{M \times (2\cdot\deg(r)+2)}$ with S being a sample set of size M. This matrix has rank $2 \cdot \deg(r) + 1$. According to Theorem 1.2, we have that $\mathbf{A}^{(2\cdot\deg(r)+1\times 2\cdot\deg(r)+2)} = \mathbf{A}^{(\operatorname{rk}\mathbf{A}\times\operatorname{rk}\mathbf{A}+1)}$ has also rank $2\cdot\deg(r)+1$, or, put differently, $\operatorname{null}\mathbf{A}^{(\operatorname{rk}\mathbf{A}\times\operatorname{rk}\mathbf{A}+1)} = 1$. Let now $(\mathbf{p}^T, -\mathbf{q}^T)^T \in \ker\mathbf{A}^{(\operatorname{rk}\mathbf{A}\times\operatorname{rk}\mathbf{A}+1)}$, where $\mathbf{p}, \mathbf{q} \in \mathbb{C}^{\deg(r)+1}$. Then, since $\deg(r) = \deg(q)$ by construction of the samples, we have for $\mathbf{q} = (q_0, q_1, \ldots, q_{\deg(r)})^T$ that $q_{\deg(r)} \neq 0$. Now suppose that $\operatorname{null}\mathbf{A}^{(\operatorname{rk}\mathbf{A}\times\operatorname{rk}\mathbf{A})} \geq 1$, i.e., there exists a vector $\mathbf{v} \in \ker\mathbf{A}^{(\operatorname{rk}\mathbf{A}\times\operatorname{rk}\mathbf{A})}$. But then we would have $(\mathbf{v}^T, 0)^T \in \ker\mathbf{A}^{(\operatorname{rk}\mathbf{A}\times\operatorname{rk}\mathbf{A}+1)}$, which, since $(\mathbf{p}^T, -\mathbf{q}^T)^T$ and $(\mathbf{v}^T, 0)^T$ are clearly linearly independent, contradicts $\operatorname{null}\mathbf{A}^{(\operatorname{rk}\mathbf{A}\times\operatorname{rk}\mathbf{A}+1)} = 1$. Hence, $\mathbf{A}^{(\operatorname{rk}\mathbf{A}\times\operatorname{rk}\mathbf{A})}$ must have full rank.

These considerations show that finding the unique kernel vector of $\mathbf{A}$ is equivalent to finding the unique solution to $\mathbf{A}^{(\operatorname{rk}\mathbf{A}\times\operatorname{rk}\mathbf{A})}\cdot\mathbf{x} = \mathbf{a}$, where $\mathbf{a}$ is the last column of $\mathbf{A}^{(\operatorname{rk}\mathbf{A}\times\operatorname{rk}\mathbf{A}+1)}$. Therefore, we say that the condition of the reconstruction matrix $\mathbf{A}$ is the condition of $\mathbf{A}^{(\operatorname{rk}\mathbf{A}\times\operatorname{rk}\mathbf{A})}$.

To produce the following data we set $r_{\text{zer}} = 10$, $n_{\text{dist}} = 0$, and $n_{\text{add}} = 0$, which means that we had $M = 2\cdot\deg(r)+2$ samples, and varied the degree of the rational functions. For the AAA algorithm, we used the rank stopping criterion (2.2). We only computed the condition of the found reconstruction matrix if it actually had a kernel of dimension 1. In the next section, we will shortly talk about why this is not always the case. Also, we only show data from matrices where the expected rank was found correctly, i.e., as we would expect from the theory.

Figure 2.1 shows the mean of the condition numbers for all matrices where the rank was found correctly.

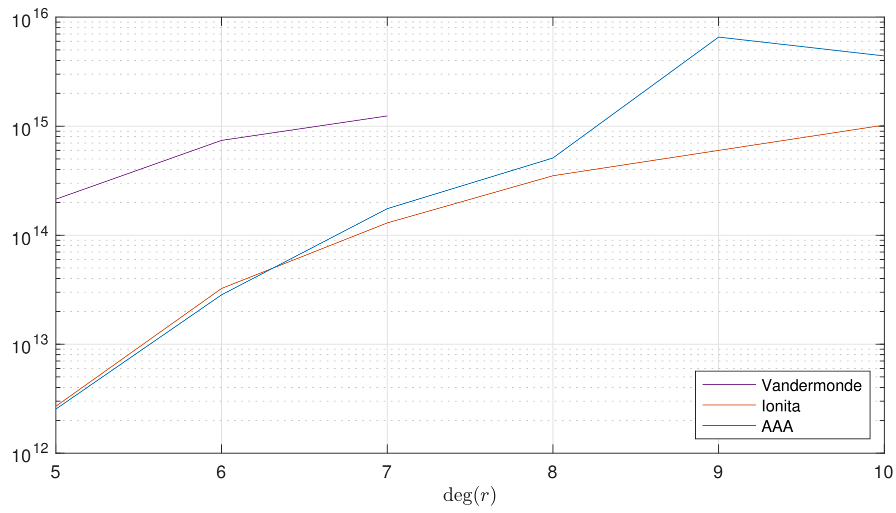

Figure 2.1.: Mean condition numbers of different reconstruction matrices for growing degrees of the original function with $r_{\text{zer}} = 10$ and $n_{\text{add}} = 0$.

The first thing to note about this graph is that for degrees higher than 7, there is no data for the Vandermonde algorithm. This is because for degree 8 there was only 1 function, for which the degree is detected correctly and none for the higher degrees. But more on that in the next section.

Other than that we, see that the condition of the Vandermonde-type matrices is one to two orders of magnitudes worse than the condition of the Löwner matrices. The matrices obtained from Ioniță's and the AAA algorithm display a very similar behavior, where for higher degrees the matrices from Ioniță's algorithm have a lower condition number and behave more consistently. Overall, we observe that the condition numbers grow about a half to one order of magnitude per degree, which certainly, at least among other factors, has to do with the fact that the matrices simply grow in size.

Now let us look at some data in more detail. In Table 2.1 we display the mean, median, minimum, and maximum of the condition numbers per degree for the matrices where the rank was detected correctly.

deg	Stat	Vander	Ioniță	AAA
5	Mean	2.1371e+14	2.6930e+12	2.5373e+12
	Med	2.5785e+12	3.3775e+07	5.8265e+07
	Min	3.2806e+06	1.6026e+01	6.2545e+01
	Max	3.2554e+16	3.1966e+15	4.6068e+15
7	Mean	1.2403e+15	1.2946e+14	1.7488e+14
	Med	4.4057e+14	5.9992e+11	6.8350e+11
	Min	2.7377e+11	2.0204e+04	5.9020e+04
	Max	5.7809e+16	4.8462e+15	2.1686e+17
9	Mean		5.9905e+14	6.5554e+15
	Med		2.0915e+14	1.5173e+14
	Min		2.7897e+08	2.4392e+08
	Max		7.2663e+15	7.5603e+18

Table 2.1.: Mean, median, minimum, and maximum condition numbers for different reconstruction matrices and degrees of the original function with $r_{\text{zer}} = 10$ and $n_{\text{add}} = 0$.

This more detailed data mostly confirms our observations from the previous graph. The numbers for the Vandermonde-type matrices are always orders of magnitudes worse than those for the Löwner matrices. Further, we see again that the data from Ioniță's and the AAA algorithm is very similar.

A few more notes on the Löwner matrices. First observe that, while for any of those matrices the condition numbers are very high, the minimal condition, especially at low degrees, is quite good. For example, it is around 16 respectively 62 for degree 5. Looking at the median, again, especially for the lower degrees, we see that it is orders of magnitude better than the mean. This tells us that while some matrices have an extremely large condition number, most of them seem to be moderately conditioned.

Finally, a few words on the data for degree 9. As we have seen already in Figure 2.1, the mean condition of the matrices from the AAA algorithm is an order of magnitude worse than that of the matrices from Ioniță's algorithm. However, the median value for the AAA algorithm is even slightly better than that for Ioniță's algorithm. Looking at the maximum condition number explains where this discrepancy comes from. Further, note that here the mean and median values are roughly of the same size. This is due to the fact that of the 10.000 functions which were evaluated, for less than 300 the rank was found correctly, which means that the set from which we computed this data is very small.

2.5.3. Degree of the Reconstructed Function

While looking at the condition numbers of the involved matrices gives us a first hint at how our problem behaves, it does not give us a very deep insight. Ultimately, our goal is to reconstruct a rational function r from given sample data and not just approximate some given data by a rational function $\tilde{r}$, which, although very similar, are different problems after all. Therefore, it is of utmost importance that we find the correct degree of the rational function r from which our data stems.

For our first observations, we use the very same data as in the previous section. Figure 2.2 shows the number of times each algorithm detects the correct rank of the reconstruction matrix.

This impressively confirms what we could only have guessed after looking at the condition numbers. Ioniță's and the AAA algorithm perform very similarly, with Ioniță's algorithm being slightly better, while the naive Vandermonde algorithm falls far behind.

In Table 2.2, we not only report the exact numbers of how often the rank is detected correctly but also how often the detected rank is too low and how often it is too high.

Recall that in order to produce this data we use $M = 2 \cdot \deg(r) + 2$ samples, i.e., the minimal number of samples necessary to reconstruct the functions. This also means that, theoretically, the initial matrices for the Vandermonde and Ioniță's

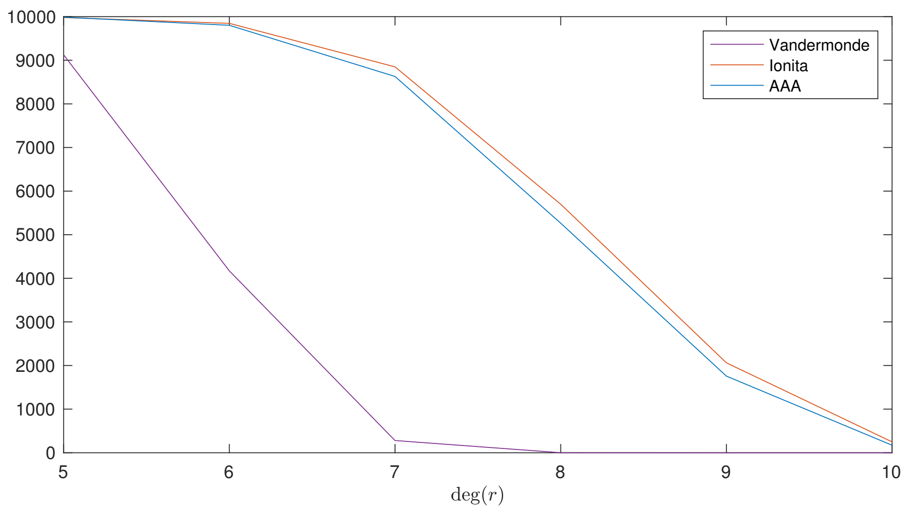

Figure 2.2.: Number of times the rank of different reconstruction matrices is detected correctly for growing degrees of the original function with $r_{\text{zer}} = 10$ and $n_{\text{add}} = 0$.

algorithm should have a nullity of 1 and the AAA algorithm should stop in the last step before the column array gets larger than the row array. Looking at the data it is not surprising to see that the Vandermonde algorithm often underestimates the rank of the reconstruction matrix since it is well known that the larger a Vandermonde matrix gets, the more likely it is that it gets numerically rank deficient. Interestingly, we also see the reverse phenomenon for the Löwner matrices. If the rank of the reconstruction matrix is too high, then this means for Ioniță's algorithm that the initial matrix has numerically full rank and for the AAA algorithm that the iteration process does not terminate before the column array got too large. Note, however, that this is only true here since we use the minimum necessary number of samples. We will talk a little bit more about this when we look at the impact of additional samples.

So far we considered the effect of changing the degree of the function we want to reconstruct. We now shortly investigate the influence of increasing the second parameter r_{zer}, i.e., the range of the zeros and poles of the function. Figure 2.3 again shows the number of times the rank of the reconstruction matrix is detected correctly depending on the parameter r_{zer}. The degree of the functions is fixed to be 5 and again $n_{\text{add}} = 0$.

deg	rank	Vand	Ioniță	AAA
5	c	9124	9980	9988
	l	876	6	7
	h	0	14	5
6	c	4171	9846	9802
	l	5829	148	198
	h	0	0	6
7	c	280	8849	8629
	l	9720	1147	1368
	h	0	4	3
8	c	1	5699	5264
	l	9999	4298	4736
	h	0	3	0
9	c	0	2062	1757
	l	10000	7935	8243
	h	0	3	0
10	c	0	249	172
	l	10000	9751	9828
	h	0	0	0

Table 2.2.: Number of times the rank of different reconstruction matrices is found correctly (c), too low (l), or too high (h), for growing degrees of the original function with $r_{\text{zer}} = 10$ and $n_{\text{add}} = 0$.

The impact of increasing r_{zer} on the two methods based on Löwner matrices is almost neglectable here, where the AAA algorithm performs a little better than Ioniță's algorithm. We will not go deeper into analyzing the exact numbers, but, for $r_{\text{zer}} = 10$, the AAA algorithm finds the correct rank 9988 times and Ioniță's algorithm 9980 times. For $r_{\text{zer}} = 100$, these numbers are 9272 and 9173. Again, the Vandermonde algorithm performs extremely poorly. For $r_{\text{zer}} = 10$, the rank is detected correctly 9124 times. This number goes down to just 314 for $r_{\text{zer}} = 100$.

However, for larger degrees the impact of increasing r_{zer} appears to be greater. Figure 2.4 shows data analogous to the previous graph, but this time for $\deg(r) = 7$ and only the AAA as well as Ioniță's algorithm.

Interestingly, here the number of correctly detected ranks increases at first, before then going down considerably. Now Ioniță's algorithm is notably better. For $r_{\text{zer}} = 10$, we find for Ioniță's algorithm that the rank is detected correctly 8849 times and for the AAA algorithm 8629 times. For $r_{\text{zer}} = 100$, we have 2439, respectively 1748.

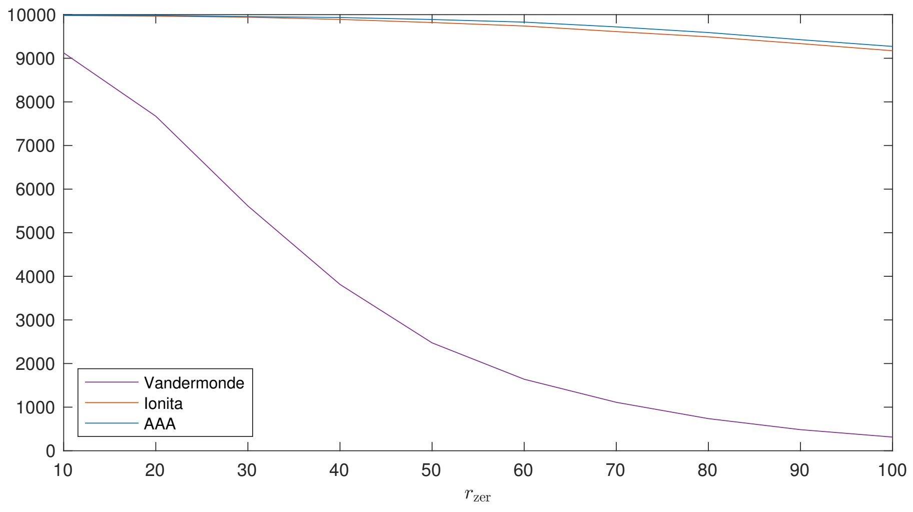

Figure 2.3.: Number of times the rank of different reconstruction matrices for functions of degree 5 is found correctly for growing r_{zer} with $n_{\mathrm{add}} = 0$.

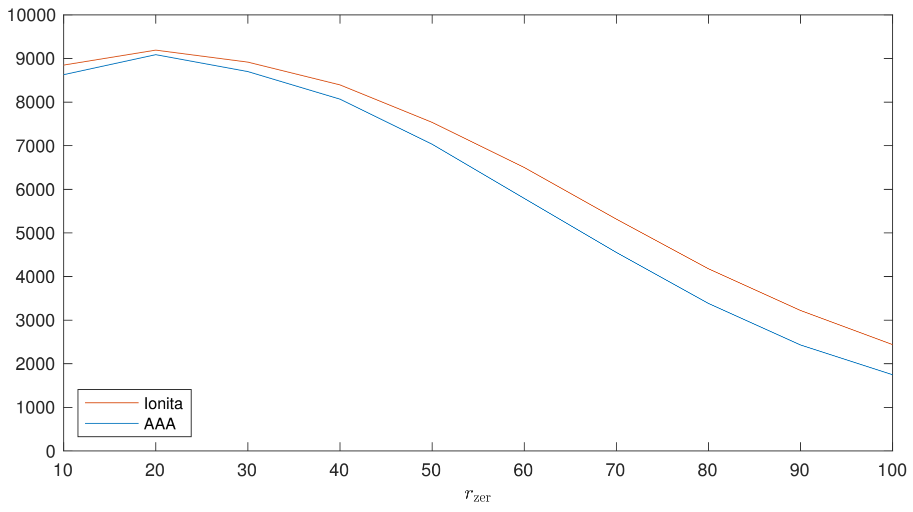

Figure 2.4.: Number of times the rank of different reconstruction matrices for functions of degree 7 is found correctly for growing r_{zer} with $n_{\mathrm{add}} = 0$.

Let us quickly summarize our findings so far. By now it should be sufficiently clear that the Vandermonde algorithm is not suited for any practical application. Considering only the Löwner-matrix-based methods, they both perform very similarly. Increasing the degree of the wanted function and increasing the range of its zeros and poles both negatively affect the reconstruction (in terms of finding the correct degree). However, the effect of increasing the degree is much larger.

Now let us look at the parameter n_{add}, the number of added samples. From now on, we will not consider the Vandermonde algorithm anymore. This is not just because the algorithm performed badly until now, but even for $\deg(r) = 5$ and $r_{\text{zer}} = 10$ it sufficed to add just 9 samples and the algorithm did not find the correct rank a single time.

Before we look at the data, we need to discuss some fundamental differences in how the two Löwner-matrix-based algorithms work. Ioniță's algorithm first creates a matrix from the given data, computes its rank, and then, if necessary, removes columns from the matrix such that the resulting new matrix has (at least theoretically) a nullity of 1. For every column which gets removed, a row is added.

The AAA algorithm works just the other way around. It adds columns to a matrix while at the same time removing rows until some stopping criterion is met (at the moment, since we use the rank stopping criterion (2.2), until the matrix becomes singular). Another very important distinction is that while for Ioniță's algorithms it is necessary that the kernel of the final matrix has exactly dimension 1, so that we get a unique result, this is not the case for the AAA algorithm. It will always return the singular vector to the smallest singular value regardless of the numerical rank of the final matrix. Even if the smallest singular value is not unique, the AAA algorithm returns just one weight vector.

Note that it is very well possible that for the error stopping criterion (2.1) the final matrix in the AAA algorithm has numerically a full rank. It is also possible for both stopping criteria that the final matrix has a kernel of dimension larger than 1. In particular, for the rank stopping criterion, the kernel of the final matrix can at most have dimension 2. This follows since the algorithm stops as soon as the matrix does not have full rank anymore. But this means that in the penultimate step the matrix has full rank. In the last step, one column is added to the matrix and one row removed, which can increase the nullity by at most 2.

These differences have the following effect on the outcome of the algorithms. In order to get a weight vector of the correct size $\deg(r)$, it is necessary that Ioniță's algorithm detects the rank of the Löwner matrix correctly, so that the reshaped matrix has the right size. However, additionally, this reshaped matrix needs to have a (numerical) nullity of 1 since otherwise there is either no or not a unique kernel vector. For the AAA algorithm, on the other hand, the final matrix only has to have the right column size (at least as long as there are no unattainable points in the data set, but we will deal with this later).

Until now, as noted before, we worked with the minimal necessary number of samples $M = 2 \cdot \deg(r) + 2$, which, as also discussed earlier, means that the rank of the final reconstruction matrices can only be too high if the initial matrix for Ioniță's algorithm has full rank or the AAA algorithm does not terminate before the column set gets too large. Adding more samples, we start to see some new phenomena. For Ioniță's algorithm, as before, the detected rank can be correct, too low, or too high. However, if before the rank was detected correctly, then this automatically meant that the matrix had a kernel of dimension one. With added points, it is possible that the rank is initially detected correctly but the nullity of the reshaped matrix can be 0, i.e., too low, or larger than 1, i.e., too high. For the AAA algorithm, we will from now on focus more on the size of the final matrix than its rank.

The following data was produced using the parameters $\deg(r) = 7$, $r_{\text{zer}} = 50$ and by varying n_{add}. First, let us look at $n_{\text{add}} = 0$ in Table 2.3 separately in order to see from where we start and illustrate the differences between this case and $n_{\text{add}} > 0$ discussed here.

	rank correct			rank too low			rank too high		
nullity	1	0	> 1	1	0	> 1	1	0	> 1
Ioniță	7536	0	0	2418	11	16	0	19	0
AAA	7035	0	0	2836	0	126	0	3	0

	size correct			size too low			size too high		
rank	c	l	h	c	l	h	c	l	h
AAA	7035	123	3	0	2839	0	0	0	0

Table 2.3.: Number of times the rank of different reconstruction matrices is (initially - for Ioniță's algorithm) found correctly, too low, or too high, as well as whether the (reshaped - for Ioniță's algorithm) matrices have a kernel of dimension 1, 0, or larger than 1. Number of times the size of the final matrix from the AAA algorithm is correct, too low, or too high and whether its rank is correct (c), too low (l), or too high (h). Original functions of degree 7 with $r_{\text{zer}} = 50$ and $n_{\text{add}} = 0$.

In case of Ioniță's algorithm, the rank in this table refers to whether the rank of the reconstruction matrix is initially found correctly, i.e., whether step (ii) in Algorithm 1.13 delivers the correct result. The nullity refers to the nullity of the reshaped matrix, used in step (iv) of Algorithm 1.13. Let us put this data into perspective. First, note that the 19, respectively 3, times the detected rank is too high mean that, since $n_{\text{add}} = 0$, the initial matrix has full rank, respectively the algorithm does not terminate before the column set gets too large. As soon as we start adding samples these particular phenomena stop occurring, i.e., the initial matrix never has full rank and the AAA algorithm always terminates properly. Also, due to the fact that $n_{\text{add}} = 0$, the matrix size cannot grow too large in the AAA algorithm. The most interesting numbers, as discussed above, are the number of times the Ioniță algorithm finds the correct rank and the reshaped matrix, which in this case is the initial matrix, has the right size and the number of times the AAA algorithm finds the correct matrix size. Here, these numbers are 7536 and 7161, although the latter contains the 3 times the algorithm does not terminate.

One last remark on this data. While, as noted above, the AAA algorithm always returns a weight vector, Ioniță's algorithm only returned 9954 results, namely only those for which the kernel of the (possibly reshaped) matrix has dimension 1.

Before we compare the two algorithms, we look a little closer at the data from Ioniță's algorithm in Table 2.4, where we will see an interesting pattern.

	rank correct			rank too low			rank too high		
n_{add} / null	1	0	> 1	1	0	> 1	1	0	> 1
50	6762	0	868	257	0	29	6	0	2078
100	7132	0	990	189	0	29	0	0	1660
250	7489	0	1160	180	0	38	0	0	1133
500	7645	0	1321	240	0	47	0	0	787

Table 2.4.: Number of times the initial rank of the reconstruction matrix from Ioniță's algorithm is found correctly, too low, or too high, as well as whether the reshaped matrices have a kernel of dimension 1, 0, or larger than 1 for original functions of degree 7 with $r_{\text{zer}} = 50$.

We see that for higher numbers of added points, and with just 6 exceptions for $n_{\text{add}} = 50$, every time the rank of the initial matrix is overestimated, the nullity of the reshaped matrix is too large. Recall that if `rk` is the numerically detected rank of the initial matrix, then a new matrix is constructed with a column array of size `rk` $+ 1$, which, in theory, should have a nullity of 1. However, if the rank is initially computed too large, then `rk`$+1$ will be larger than $\deg(r)+1$, the correct column size. In theory, the reshaped matrix should therefore have a nullity larger than 1 and apparently this is also the case in practice.

This leads to a very simple but effective modification of Ioniță's algorithm. After computing the rank of the initial matrix and creating a new reshaped matrix, we again compute the rank `rk` of this new matrix, i.e., the matrix from step (iv) of Algorithm 1.13. Then, we compare `rk` to the column size of the matrix. If the difference is larger than 1, the nullity will also be larger than 1. In this case, we create a new reshaped matrix with a column array of size `rk`$+1$. If necessary, this process can also be repeated.

This modification adds some complexity to Ioniță's algorithm and also shifts its character from being very static to being a little more dynamic. However, the results of this simple modification speak for themselves. We used the modified Ioniță algorithm on the same input data as used to produce the results in Table 2.4. The following table shows the number of times the final matrix from the modified algorithm has the correct rank and a kernel of dimension 1, i.e., the results displayed for the non-modified algorithm in the first column of Table 2.4.

n_{add}	50	100	250	500
	8696	8671	8529	8368

Without showing the actual numbers, let us present a few more observations from this data. First, with just very few exceptions for small n_{add}, the final matrices had usually either the correct rank, or the rank was too low, i.e., there were usually no matrices with too high rank. Second, with at most 2 exceptions among 10.000 evaluated functions, the final matrices had almost always a nullity of 1. And lastly, for large n_{add}, the number of matrices for which the rank was initially detected correctly in the non-modified algorithm was actually larger than the number of final matrices in the modified algorithm with correct rank. However, the number of matrices with the correct rank and a kernel of dimension 1 was still considerably larger for the modified algorithm.

In our experiments, the modified Ioniță algorithm always delivered better results than the non-modified. Therefore, we will from now on only use the modified version for our comparisons. Going forward, we will also include the AAA algorithm with the error stopping criterion (2.1) in our investigations. Recall that for data without unattainable points both stopping criteria are theoretically equivalent.

Figure 2.5 shows the behavior of the original and modified Ioniță algorithm as well as the AAA algorithm with the two different stopping criteria under increasing n_{add}. For Ioniță's algorithm, the data displayed is the number of times the rank is detected correctly and the final matrix has a kernel of dimension 1. For the AAA algorithm, we show the number of times the final matrix has the correct size.

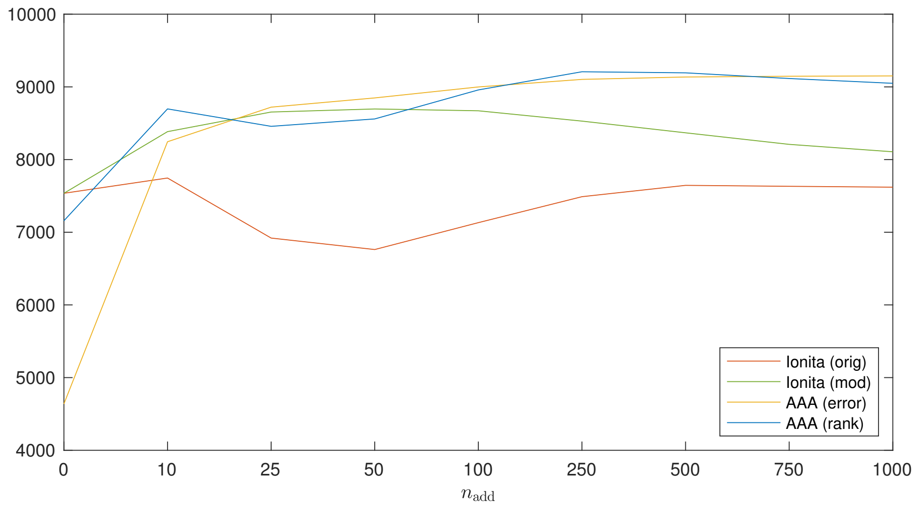

Figure 2.5.: Number of times the rank of the reconstruction matrix is correctly found and the final matrix has a kernel of dimension 1 for Ioniță's algorithm and the number of times the reconstruction matrix has the correct size for the AAA algorithm for growing n_{add}. Original functions of degree 7 with $r_{\text{zer}} = 50$.

First, we note again that the original Ioniță algorithm does not only perform much worse than the AAA algorithm but also than the modified version. Only if there are no added points, Ioniță's algorithm gives better results than both AAA variants. Interestingly, in this case, the AAA algorithm with the error stopping criterion performs extremely badly with just 4638 times where the size is found correctly and one time where the algorithm does not terminate.

For added numbers in the range of 10 to 100, the two AAA variants and the modified Ioniță algorithm give very similar results. However, going beyond 100 added points, the performance of Ioniță's algorithm starts to get worse, approaching the performance of the original variant. At the same time, the results of both AAA variants are very close to each other and do not seem to be affected a lot by further increasing the number of added points.

We ran some more tests, in which we looked at the effect of added samples for different degrees of the function we want to reconstruct. What we observed was that the lower the degree of the function is, the better Ioniță's algorithm performs, i.e., the lower the degree, the more added samples it takes until the AAA algorithm reaches the same or a better performance than Ioniță's algorithm. On the other hand, the higher the degree of the function, the fewer added samples it takes until the AAA algorithm outperforms Ioniță's algorithm.

Overall, adding more sample points clearly improves the results of the AAA algorithm. Let us again consider the example reported in Table 2.2, where we found that if the function we want to reconstruct has a degree of 10, the AAA algorithm with the rank stopping criterion finds the correct rank only 172 times. Adding 250 samples, but otherwise using the same parameters, this number goes up to 985 and the correct size of the matrix is found 987 times. Even more impressive, using the error stopping criterion, the correct rank is found 1102 times and the correct size even 1920 times.

The increased performance of the AAA algorithm with a growing number of added points can heuristically be explained as follows. Recall that a greedy algorithm is used to choose the next interpolation point. If we now have more samples, this means that there are more options to choose from which point is taken next. This in turn means that the algorithm can, in some sense, make better choices.

As mentioned before, Ioniță's algorithm starts with a matrix having a large column size, especially if there are many added points, which is then decreased (in case of the modified algorithm possibly more than once). The AAA algorithm, on the other hand, increases the column size of the matrix one by one. It is now tempting to try and explain the worse performance of Ioniță's algorithm by the fact that it works with, in terms of the column size, larger matrices. In order to test this hypothesis, we also tried a sort of hybrid method, one could also call it an adaptive Ioniță algorithm, in which we iteratively added more points to the column array and removed them from the row array, until the matrix became singular, as in the AAA algorithm with the rank stopping criterion. However, other than in the AAA algorithm, we chose the next interpolation point in such a way that we ended up with an alternating partition, as used by Ioniță algorithm. The results from this algorithm were never better than the results from Ioniță's algorithm and usually just slightly worse. This might be due to the way MATLAB's `rank` function works. It computes the rank of a matrix by counting how many singular values are above a certain threshold and uses the maximum of the row and column size as a factor in determining this threshold. Since the row sizes of the matrices in the adaptive algorithm are higher than the row size for the initial matrix of Ioniță's algorithm and the row size is the maximum of row and column sizes, this means that the thresholds for a matrix to be considered singular are higher in the adaptive algorithm. This is possibly also the reason why the number of final matrices with correct rank in the modified Ioniță algorithm is lower than the number of matrices where the rank is initially detected correctly in the original version.

It is an interesting question in its own right, which is just briefly touched in [58], how to choose an optimal partition of the sample set if such a partition even exists. As already mentioned, the alternating partition we use for Ioniță's algorithm is clearly better than a linear or random partition. However, the results from the last paragraph imply that the partition found by the AAA algorithm is even better.

We finish this section by taking a closer look at the differences between the two stopping criteria for the AAA algorithm in Table 2.5.

		size correct				size too low			size too high			
crit	n_{add}	c	l	h	sum	c	l	sum	c	l	h	sum
rank	50	8556	3	0	8559	0	456	456	5	0	980	985
	100	8958	0	0	8958	0	413	413	2	0	627	629
	500	9193	0	0	9193	0	600	600	0	0	207	207
	1000	9049	0	0	9049	0	812	812	3	0	136	139
	1500	8916	0	0	8916	0	978	978	2	0	104	106
error	50	8468	7	373	8848	86	436	522	1	0	629	630
	100	8855	45	99	8999	12	345	357	22	0	622	644
	500	8843	293	0	9136	0	218	218	264	7	375	646
	1000	8703	448	0	9151	0	199	199	379	10	261	650
	1500	8627	529	0	9156	0	193	193	435	14	202	651

Table 2.5.: Comparison of the two stopping criteria for the AAA algorithm. Number of times the size of the final matrix is found correctly, too low, or too high, as well as how often the rank of the final matrix is correct (c), too low(l), or too high (h). Original functions of degree 7 with $r_{\text{zer}} = 50$.

First, note that it is not possible that the column size of the final matrix is too low while its rank is too high. This follows since, if the column size n is too low, it means that $n \leq \deg(r)$, which in turn means that the rank of this matrix can be at most $\deg(r)$. But this is the correct rank.

As we have already seen in Figure 2.5, the total number of matrices for which the size is detected correctly is very similar for both stopping criteria, with the error criterion appearing to perform slightly better for larger n_{add} (with the exception of

$n_{\text{add}} = 250, 500$). However, it seems that the number of correctly detected sizes changes more with the rank stopping criterion while for the error stopping criterion this number is almost constant for $n_{\text{add}} \geq 500$.

Unsurprisingly, the number of times the size is detected correctly correlates much stronger with the number of times the rank is detected correctly for the rank criterion than for the error criterion. Moreover, for the rank criterion, it cannot happen that the size is correct but the rank too high, or the size too low but the rank correct, or the size too high but the rank too low. The first and second cases both would imply that the final matrix has full rank. In the third case, the column size s of the final matrix being too large means $s \geq \deg(r) + 2$, while the rank being too low means that it is strictly smaller than $\deg(r)$, which implies that the nullity is strictly larger than 2 and this is not possible, as noted above.

For the rank criterion, we observe that, with the exception of $n_{\text{add}} = 50$, the number of times the size and therefore also the rank is too low increases constantly as n_{add} increases, while at the same time the number of times the size and rank are too high decreases. This might again be explained by the way MATLAB's `rank` function uses the size of a matrix for determining its numerical rank. We experimented with different thresholds in which the size of the matrix is given less weight. However, this does not result in an overall better performance. Instead, whenever a parameter leads to a decrease in the number of times the rank is detected too low, it usually increases the number of times the rank is detected too high by roughly the same amount.

Now let us look at the error criterion. First, note that at least for larger n_{add} there are again no matrices where either the size is detected correctly but the rank too high, or the size is detected too low but the rank correctly. However, there the similarities between the two criteria already end.

For $n_{\text{add}} \geq 500$, the total number of times the size is detected correctly, too low, or too high does not appear to change a lot. However, it is interesting how the distribution of the ranks of the final matrices changes. This also tells us more about how this stopping criterion works.

Differently than for the rank stopping criterion, the number of times the size and rank of the final matrix are too low constantly goes down. This is probably best explained as follows. Recall that the error stopping criterion stops the iteration process once a rational function is found which approximates the data not yet interpolated sufficiently well. This means that if the algorithm stops before the matrix has the correct size, it has found a rational function of smaller degree than the function we want to reconstruct which already approximates the given sample data quite well. Now it is clear that the fewer samples we have, the more likely it is that there exists such a function. Or, put the other way around, the more samples we have, the less likely it becomes that a function of smaller degree than that of the wanted function approximates all the data well. Hence, the number of times the algorithm terminates before the size of the final matrix is large enough decreases.

Looking at the matrices where the size was found correctly, we see that the number of times the rank is too low, which implies that the matrix has a kernel of dimension larger than 1, increases with the number of added points. This might again be explained by the way MATLAB's `rank` function works. However, it is very interesting that the number of times the size is found too high but the rank correct increases at roughly the same rate as the number of times the size is found correctly but the rank too low. First of all, this explains why the overall number of times the size is found correctly is about the same for both stopping criteria. But we can deduce more from this.

If, with the error stopping criterion, the column size of the final matrix is too high, then this means that the given data could not be approximated sufficiently well by a rational function of the correct degree. This implies that in these cases the numerical errors are too large. There are two more things to note regarding the fact that the rank of the final matrix is very often correct anyway. First, despite the fact that the sample values do not seem to be approximated very well, the structure of the data still shows the degree of the underlying function. And second, and this will become important when we consider data with unattainable points, for those samples where the final matrix has the correct rank but its size is too large, the rank criterion would have terminated correctly, however, even though in these cases the degree of the reconstructed function would be correct, it would still approximate the given data badly.

The very few times, less than 0.2%, the size of the final matrix is too large while its rank is too low, are neglectable, and probably again due to MATLAB's `rank` function.

2.5.4. Reconstruction Errors

We now look at how well the different algorithms, respectively variants, reconstruct a given rational function $r = \frac{p}{q}$. Here, we only consider numerical errors. We will consider the overall reconstruction error for the rational function and the reconstruction error for the poles, i.e., the zeros of the denominator polynomial p.

It is clear why we should be interested in the reconstruction error for the poles of the function. If we know the zeros of q, we can then reconstruct it, as noted before. Moreover, finding the poles of a rational function as accurately as possible will play an important role later on in this work.

In the same way, we could argue that we should look at the reconstruction error for the zeros of the rational function, which are also the zeros of the numerator polynomial p. However, if $\tilde{r}$ denotes the reconstructed rational function, then we will usually have, that $\tilde{r}$ is of type $(\deg(\tilde{r}), \deg(\tilde{r}))$ due to numerical errors. In particular, we will have for the reconstructed numerator polynomial $\tilde{p}$ that $\deg(\tilde{p}) = \deg(\tilde{r})$. Recall from Section 2.5.1 that the test data we use is constructed such that $\deg(q) = \deg(r)$ and $2 \leq \deg(p) \leq \deg(q)$. If now for the original functions r and p we have $\deg(r) > \deg(p)$ and assuming that $\tilde{r}$ has the correct degree, i.e., $\deg(\tilde{r}) = \deg(r)$, this means that $\tilde{p}$ has too many zeros. In our experiments, we observed that if the algorithms found the correct degree of the rational function (by producing a matrix of the right size), then the set of zeros of $\tilde{p}$ usually contained good reconstructions of the correct zeros alongside some spurious zeros. Often in these cases, the superfluous zeros were either very large, i.e., several orders of magnitudes larger than the true zeros, or appeared as complex conjugate numbers (recall that all our test data was real), implying that $\tilde{p}$ has an irreducible factor of degree 2. However, the second observation can only be used to reliably remove superfluous zeros if the input data is real. The first observation seems to deliver a relatively good condition for superfluous zeros, but, in our experiments, we saw still too many exceptions to derive a reliable rule. Moreover, we did not test if this behavior can also be observed for complex input data. Note that a large magnitude of the superfluous zeros implies that we actually have a good approximation for the numerator polynomial. Let us illustrate this with a small example.

Suppose we want to reconstruct the function $x - 5$ and the reconstruction we get is $\epsilon_1 \cdot x^2 + (\epsilon_1 \cdot (5 + \epsilon_2) + 1) \cdot x + (5 + \epsilon_2)$. For small values of ϵ_1 and ϵ_2, this is a good approximation to the original function. However, the reconstructed polynomial can be factorized as

$$\epsilon_1 \cdot \left(x - \tfrac{1}{\epsilon_1}\right) \cdot (x - (5 + \epsilon_2)).$$

This means that we have a good reconstruction of the original zero 5 and a superfluous zero $\frac{1}{\epsilon_1}$, which can be arbitrarily large.

In general, we note that without any prior knowledge about the numerator polynomial, such as whether it is real or an estimate for the range of its zeros, it appears to be very hard to reconstruct p accurately. Fortunately, we will later use rational interpolation to reconstruct functions for which we know that $\deg(p) = \deg(q) - 1$ almost surely. We will then be able to use this knowledge to enforce that the reconstructed function has type $(\deg(q) - 1, \deg(q))$, which will give us a good reconstruction of p as well. Therefore, we do not study the problem of finding the correct degree of the numerator polynomial here.

In order to measure the reconstruction error, we use the ∞-norm for the error in the poles to get a worst-case estimate. For the reconstruction error of the rational function, we proceed as follows. We compute the relative error in the 2-norm, which better shows how well the function is approximated overall than the ∞-norm, i.e., if r is the original function and $\tilde{r}$ the reconstructed, we look at the error

$$\frac{\|\tilde{r} - r\|_2}{\|r\|_2}.$$

More precisely, we sample both the original and reconstructed function in the interval $[-r_{\text{zer}}, r_{\text{zer}}]$ with a step size of 0.1, which results in sample vectors $\mathbf{r}$ and $\tilde{\mathbf{r}}$. Now note that if we have sampled the original function close to a pole, or maybe even close to several poles, then $\mathbf{r}$ will contain several very large entries, which means that $\|\mathbf{r}\|_2$ will also be very large. This might lead to the relative error being quite small, thereby obscuring the true quality of the approximation. Therefore, we decided to remove values from $\mathbf{r}$ which are likely to be too close to a pole. There are several ways to do this which all give slightly different results. In the end, we decided to use Matlab's `isoutlier` function with its default criterion to detect outliers in the entries of $\mathbf{r}$ and remove them to obtain the new vector $\mathbf{r}_c$. We then remove the corresponding entries from $\tilde{\mathbf{r}}$ to get $\tilde{\mathbf{r}}_c$ and computed the error

$$\frac{\|\tilde{\mathbf{r}}_c - \mathbf{r}_c\|_2}{\|\mathbf{r}_c\|_2}.$$

We would like to remark that this way of measuring the reconstruction error is rather uncommon. In the literature, this error is often measured in an interval which does not contain any poles. We, on the other hand, use the interval $[-r_{\text{zer}}, r_{\text{zer}}]$, which contains all poles and zeros of the original function.

Reconstruction Error for Rational Functions

Now let us look at the reconstruction errors for rational functions. The following data was produced by fixing $r_{\text{zer}} = 50$ and varying the degree and the number of added points. We only show data for samples, where the size of the reconstruction matrix was detected correctly and, in case of Ioniță's algorithm, the nullity was 1. In the first line of Table 2.6, we report the respective numbers.

We first comment on some trends we can see in this data, which confirm what we have observed so far. As already discussed, increasing the degree of the rational function we want to reconstruct leads to worse performance of all algorithms, not just regarding the number of times the correct degree of the original function, or equivalently the correct size of the final matrix, is found, but also regarding the reconstruction error. Further, concerning the matrix size, increasing the number of added points will lead to a better performance of the AAA algorithm, while the performance of Ioniță's algorithm gets worse. Note again that the two AAA variants deliver very similar results, with one major exception. For degree 9 and $n_{\text{add}} = 500$, the AAA algorithm with the error stopping criterion finds the correct size of the matrix 530 times more than the algorithm with the rank stopping criterion.

Before we look closer at the reconstruction errors, let us just quickly look at this discrepancy between the two AAA variants in a little more detail. For degrees 5 and 7, we see that the number of times the degree of the original function is correctly found keeps increasing with increasing n_{add} for both variants. However, for degree 9 and the rank stopping criterion the number first increases and then decreases again slightly. At the same time, the number keeps increasing for the error stopping criterion. Now look back at Table 2.5, where we compared the two variants in more detail with regard to increasing n_{add}. Recall that the data there was for degree 7. We also see there that while the number of correctly found degrees keeps increasing for the error stopping criterion, the number for the rank stopping criterion peaks at $n_{\text{add}} = 500$ and starts to decrease again when adding more samples. Looking at Figure 2.5 we see that the peak actually already occurs at $n_{\text{add}} = 250$. These observations might hint at a general behavior of the rank stopping criterion. Namely that the number of correctly found degrees peaks at some point and then decreases again. The higher the degree, the earlier this point seems to occur. We tested this hypothesis on smaller sample sets and the results seem to confirm it. This behavior might be, at least in part, again due to the way MATLAB's `rank` function works, but there might also be other factors at play. We also looked at the behavior of the error stopping criterion for more degrees and values of n_{add}. There, it seemed that the number of correctly found degrees increases until a certain point and then remains stable, i.e., it neither decreases nor increases in any meaningful way. However, more testing would be necessary to confirm these results.

		deg = 5			7			9		
Stat	Meth	$n_{\text{add}} = 50$	200	500	50	200	500	50	200	500
Size	Ion	9967	9953	9928	8696	8582	8368	4682	4554	4197
	Ark	9482	9780	9863	8559	9182	9193	6049	6804	6668
	Aerr	9741	9746	9740	8848	9083	9136	6129	6977	7198
Mean	Ion	7.1445e−02	1.7795e−02	1.0393e−02	7.5634e−01	1.7711e−01	2.1448e−01	2.7554e+00	1.4063e+00	2.3740e−01
	Ark	2.2164e−03	3.0220e−04	2.7025e−04	1.4788e−01	4.2495e−02	5.1400e−02	2.9371e+00	1.1727e+00	1.3163e−01
	Aerr	2.6582e−03	3.2223e−04	3.0542e−04	8.5365e−01	6.8793e−02	8.7834e−02	2.3970e+00	1.3200e+00	2.0143e−01
Med	Ion	1.3501e−07	9.4433e−08	1.3480e−07	1.7237e−04	9.1213e−05	7.6437e−05	7.2960e−03	2.9654e−03	8.2821e−03
	Ark	8.4785e−09	2.5608e−09	1.9480e−09	1.0116e−05	2.2507e−06	1.3913e−06	1.7522e−03	3.3581e−04	5.0938e−04
	Aerr	8.9742e−09	2.5312e−09	1.9142e−09	1.0682e−05	2.3126e−06	1.5305e−06	1.6307e−03	4.7052e−04	9.5377e−04
Min	Ion	9.7062e−14	5.6770e−14	7.5801e−14	2.5579e−12	2.3076e−12	8.3379e−13	2.4239e−10	1.1448e−10	7.0215e−09
	Ark	1.8543e−15	2.0023e−15	2.7005e−15	1.0452e−14	3.5413e−14	1.4942e−14	1.8601e−13	7.3217e−14	4.9070e−11
	Aerr	1.8543e−15	2.0023e−15	2.7005e−15	1.0452e−14	3.5413e−14	1.4942e−14	1.8601e−13	7.3217e−14	4.9070e−11
Max	Ion	3.5391e+02	5.1028e+01	1.6038e+01	1.8643e+03	2.7780e+02	3.6356e+02	2.0444e+03	2.8863e+03	1.3435e+01
	Ark	4.9129e+00	3.3107e−01	5.4962e−01	1.4570e+02	8.7254e+01	1.2252e+02	6.5803e+03	2.7361e+03	3.3731e+01
	Aerr	4.9129e+00	3.3107e−01	5.4962e−01	6.2636e+03	8.7254e+01	1.2252e+02	6.5803e+03	2.7361e+03	3.3731e+01

Table 2.6.: Correctly found sizes of the reconstruction matrix as well as mean, median, minimum, and maximum of the reconstruction errors for functions with $r_{\text{zer}} = 50$.

We see something very interesting when we compare the mean and median errors of the two AAA variants. Although the errors are very similar for both variants, two-thirds of the time the algorithm with the rank stopping criterion produces a slightly smaller error than the algorithm with the error stopping criterion. Intuitively, one would expect the error criterion to perform better. Unfortunately, we do not have an explanation for this behavior. It might have to do with the fact that some of the final matrices produced with the error stopping criterion have a numerical nullity larger than 1. However, we were not able to definitively establish a connection here.

Generally, as noted above, we observe that the higher the degree, the larger the reconstruction error gets. The median error, for example, gets worse roughly two to three orders of magnitude each time the degree is increased by 2. We also see that adding additional samples generally decreases the reconstruction error. This is even true for Ioniță's algorithm, where sometimes the error is best for $n_{\text{add}} = 200$ and sometimes for $n_{\text{add}} = 500$, but for both these numbers the error is always better than for $n_{\text{add}} = 50$. The only two exceptions to this are for degree 9 the median and minimum error for $n_{\text{add}} = 500$ and the maximum error for $n_{\text{add}} = 200$.

Looking at the overall error, especially the median error, we see that all algorithms achieve quite good results. What is interesting, though, is that the median error is much smaller than the mean error. In fact, it is up to six orders of magnitude smaller and usually three to four orders of magnitude better than the mean error. This implies that the reconstruction is very good most of the time with relatively few but large exceptions. And indeed, this suspicion seems to be confirmed once we look at the minimum and maximum errors.

Considering the minimum and maximum errors, first note that, with one notable exception, both variants of the AAA algorithm have the same errors, implying that these errors are achieved for the same sample. The exception is for degree 7 and $n_{\text{add}} = 50$. We looked closer at this example and found that the sample which produced the largest error for the rank stopping criterion actually produced the second-largest error for the error stropping criterion, while the sample which produced the largest error for the error criterion was not even among the samples with the ten largest errors for the rank criterion. However, we did not find any particular reasons why this sample would produce such a large error for the error stopping criterion.

One thing that immediately stands out is the huge difference between the minimum and maximum errors. Up to 17 orders of magnitude lie in between the two errors for both AAA variants at degree 9 and $n_{\text{add}} = 200$ and for the AAA algorithm with error stopping criterion at the before mentioned degree 7 and $n_{\text{add}} = 50$. The latter example also constitutes the example with the largest maximal error with 6.2636e+03. Recall that this number refers to the relative error in the 2-norm. This means that for this sample the error was more than 6000 times larger than the original function.

At this point, we would like to briefly examine a little more how we acquired that data. Recall that we sample the original and reconstructed functions and then remove points using MATLAB's `isoutlier` function on the sampled values of the original function. If we do not do this, i.e., if we do not remove any sample values, then we get for degree 7 and $n_{\text{add}} = 50$ a maximum error of 6.0623e+02. We already mentioned that removing particularly large samples values before computing the error might lead to larger errors since $\|\mathbf{r}_c\|_2 \leq \|\mathbf{r}\|_2$, i.e., the number we divide by in order to obtain the relative error gets smaller. However, usually, not removing too large values from $\mathbf{r}$ results in worse errors. For example, the mean and median errors are one to two orders of magnitudes worse when we do not remove outliers. This can be explained as follows. Suppose we have a good approximation to the original function in the sense that the poles of the reconstruction lie close to the poles of the original. Now, regardless of how close the poles of the functions are to each other, as long as they are not identical, the difference between the functions gets arbitrarily large the closer we get to the pole, which of course influences the overall error. As a short illustration of this, simply consider the function $\frac{1}{x}$ and $\frac{1}{x+\epsilon}$. Their difference

$$\frac{1}{x} - \frac{1}{x+\epsilon} = \frac{\epsilon}{x^2 + \epsilon \cdot x}$$

gets arbitrarily large as we get closer to 0. If we now remove a large value from $\mathbf{r}$, i.e., one, which probably lies very close to a pole, then the corresponding value in $\tilde{\mathbf{r}}$ lies probably also very close to a pole and the difference between the values of the original and reconstructed function is very large.

We examined the data more closely in order to understand better how the very large reconstruction errors come about. Unfortunately, we were not able to answer this question conclusively. Some factors which might play a role are for example the errors in the poles. One would imagine that if the reconstructed function has a pole in the wrong place, this leads to large reconstruction errors. However, we found examples for functions where the poles are badly reconstructed and yet the overall reconstruction error is very good. Another factor might be if the original function has some poles which lie very close together or pairs of poles and zeros which are very close. But again, we found examples where this is the case but the overall reconstruction is very good.

On the other hand, we found examples where the reconstruction error is quite bad, yet the original parameters of the function are reconstructed very well. We illustrate this with an example. The following table shows the exact values of the original and reconstructed poles and zeros of a function of degree 7 with $n_{\text{add}} = 50$ from the AAA algorithm with the rank stopping criterion.

orig. zero	rec. zero	orig. pole	rec. pole
−47.1414	−46.6976	−47.7147	−47.3314
33.6049	33.6050	−42.3808	−42.3013
		−38.1955	−38.2141
		−27.2396	−27.2393
		−20.2959	−20.2959
		−8.1935	−8.1935
		21.9015	21.9015

The overall reconstruction error for this example is 36.5772, which is the sixth-largest error in the whole data set. However, the individual parameters of the function are reconstructed very well. The algorithm even detects the correct degree of the numerator polynomial. In this case, the large overall reconstruction error might be due to the fact that the original function has a pole and a zero which lie very close together. Although this is a particularly outstanding example, it is not uncommon that for examples with a large reconstruction error the zeros and poles are actually reconstructed quite well. This shows us that the overall reconstruction error might not be the best measure to determine the quality of the reconstruction. This will also become clear when we look at the reconstruction errors for the poles in more detail.

Reconstruction Error for Poles

Table 2.7 shows the reconstruction errors for the poles of the same samples as in Table 2.6.

Again we see much of the behavior we already know. Increasing the degree of the original function decreases the performance while adding more samples generally increases it. Overall, the AAA variants perform very similarly but better than Ioniță's algorithm. The median error is again much better than the mean, which leads to the same conclusions as before.

Looking at the minimal reconstruction errors we find once more that both AAA variants always produce the same outcome. As an interesting side note, for degree 5 the minimal error stays the same regardless of the number of added points.

		deg = 5			7			9		
Stat	Meth	$n_{\text{add}} = 50$	200	500	50	200	500	50	200	500
Mean	Ion	9.6485e−03	3.5528e−03	2.5286e−03	1.1959e−01	5.7243e−02	4.4336e−02	5.4573e−01	2.9086e−01	2.3740e−01
	Ark	8.2277e−04	2.5353e−04	1.4018e−04	2.3924e−01	2.1614e−02	1.1718e−02	9.7177e−01	2.4122e−01	1.3163e−01
	Aerr	9.6088e−04	2.8266e−04	1.8474e−04	8.5365e−01	2.2924e−02	1.6953e−02	8.2969e−01	2.6922e−01	2.0143e−01
Med	Ion	7.8474e−07	5.0581e−07	7.0482e−07	6.3553e−04	3.1997e−04	2.6617e−04	2.5070e−02	1.0573e−02	8.2821e−03
	Ark	4.3557e−08	1.0782e−08	7.7889e−09	3.4183e−05	6.7144e−06	3.9263e−06	6.8826e−03	1.0509e−03	5.0938e−04
	Aerr	4.3535e−08	1.0680e−08	7.7422e−09	3.5010e−05	7.1081e−06	4.4011e−06	6.4183e−03	1.4853e−03	9.5377e−04
Min	Ion	1.9540e−13	1.2079e−13	1.0658e−13	2.2965e−11	3.5110e−12	1.4268e−11	3.9525e−09	3.6642e−09	7.0215e−09
	Ark	1.4211e−14	1.4211e−14	1.4211e−14	2.7001e−13	2.629e0−13	2.7356e−13	3.0296e−10	2.1977e−11	4.9070e−11
	Aerr	1.4211e−14	1.4211e−14	1.4211e−14	2.7001e−13	2.6290e−13	2.7356e−13	3.0296e−10	2.1977e−11	4.9070e−11
Max	Ion	1.1535e+01	1.6318e+00	7.3093e−01	6.0280e+01	7.2735e+00	6.0818e+00	3.9935e+01	1.1983e+01	1.3435e+01
	Ark	1.1498e+00	7.2246e−01	4.9442e−01	1.5875e+03	1.5642e+01	1.0896e+01	9.6566e+02	1.5967e+02	3.3731e+01
	Aerr	1.5006e+00	7.2246e−01	4.9442e−01	5.4029e+02	9.4464e+00	8.4593e+00	9.6566e+02	5.5138e+01	4.3411e+01

Table 2.7.: Mean, median, minimum, and maximum of the reconstruction errors for the poles of functions with $r_{\text{zer}} = 50$.

More interesting things are going on with the maximal errors. First, note that for degree 9 Ioniță's algorithm produces the best maximal error. However, this is just due to the fact that the set for which this data is computed, i.e., the samples where the correct rank is found and the matrix has a nullity of 1 is much smaller than the set of samples where the AAA variants find the correct size. Running the AAA algorithms only on those samples used to compute the errors for Ioniță's algorithm, we get far better results than from Ioniță's algorithm. From now on, we will therefore only compare the two AAA variants.

Other than for the overall maximal reconstruction error, the two AAA variants only have the same maximal reconstruction error for the poles three times. We now look at how these large errors come about. But first an interesting observation. One would expect that large errors when reconstructing the poles lead to large overall reconstruction errors. However, this does not seem to be the case. We looked at the 100 samples which produce the largest overall reconstruction error and the 100 samples which produce the largest error in the poles. The following table shows the number of samples which are in both sets for both stopping criteria.

	deg = 5			deg = 7			deg = 9		
n_{add}	50	200	500	50	200	500	50	200	500
rank	71	66	57	32	44	45	9	23	29
error	69	65	61	29	41	40	8	19	19

This observation suggests that there is no strong correlation between the error in the poles and the overall reconstruction error. We looked further and found that not even once the sample which produces the largest overall reconstruction error is also the sample with the largest error in the poles. Even more, with the exception of degree 5 and $n_{\text{add}} = 50$, the sample which produces the largest error in the poles is not even among the 100 samples which produce the largest overall error. This is true for both AAA variants. On the other hand, with the exception of degree 9 for both variants and degree 7 with $n_{\text{add}} = 50$ for the rank stopping criterion, the examples which produce the largest overall reconstruction error are not among the 100 examples with the largest errors in the poles.

Before, we saw an example where even though the zeros and poles of the original function were reconstructed very well, the overall reconstruction error was quite large. Now let us look at an example where the overall reconstruction error is very good but reconstruction of the poles is bad. Again, the following table shows the results for a function of degree 7 with $n_{\text{add}} = 50$, which was reconstructed using the AAA algorithm with the rank stopping criterion.

orig. zero	rec. zero	orig. pole	rec. pole
−48.1312	−48.1320	−30.1814	−37.2469
−37.8951	−37.8552	−14.5528	−14.5528
−30.3711	−37.2900	−1.1518	−1.1518
−30.1289	−30.3147	13.3378	13.3378
−87.8575e−03	−87.8575e−03	20.4308	20.4308
17.4070	17.4070	34.8323	34.8323
19.4064	19.4064	39.1053	39.1053

We see that the algorithm completely fails to reconstruct the leftmost pole. The error here is 7.0655, which is the seventh-largest of all reconstruction errors for the poles. However, the overall reconstruction error for the function is just 5.7359e-05, which is the same order of magnitude as the median reconstruction error. This shows again impressively that there does not seem to be a close connection between the overall reconstruction error and the error in the reconstructed poles. A fact which is further supported by the observation that the overall reconstruction error for samples where the degree of the original function was not correctly detected (which means that the reconstruction has either too few or too many poles) was usually just slightly worse than if the degree was detected correctly.

However, we will not take a closer look at samples where the degree was not detected correctly. Instead, we like to remark that the above example is somewhat typical. Note that, except for the leftmost, all poles were reconstructed exactly, at least up to four digits after the comma. The three rightmost, i.e., largest, zeros were also reconstructed with the same precision. When looking closer at those examples where the reconstruction error for the poles was very large we noted that most of the time the largest error occurred at the leftmost pole.

Recall now that for reconstruction we used the $M = 2 \cdot (\deg(q) + 1) + n_{\text{add}}$ sample points $1, 2, \ldots, M$. This means that the leftmost pole is also the one farthest away from the sample points. To test whether this is the reason for large errors in the poles we ran the two AAA variants again, using the symmetric sample points $-\left\lceil \frac{M}{2} \right\rceil, -\left\lceil \frac{M}{2} \right\rceil + 1, \ldots, \left\lfloor \frac{M}{2} \right\rfloor$. We report our findings for degree 7 in Table 2.8.

Comparing these results to those reported in Table 2.6 and Table 2.7, we see that the performance of the algorithms increases dramatically. To illustrate this, we would

		n_{add}	50	200	500
	Size	Ark	9279	8793	9289
		Aerr	9522	8731	8735
Fct. Err.	Mean	Ark	1.5621e−04	3.8310e−07	1.1888e−06
		Aerr	7.7107e−04	4.4779e−07	8.7342e−07
	Med	Ark	5.2461e−10	3.5667e−12	4.5805e−12
		Aerr	5.9833e−10	3.5397e−12	3.5307e−12
	Min	Ark	4.4917e−15	3.6346e−15	3.4138e−15
		Aerr	4.4917e−15	3.6346e−15	3.4138e−15
	Max	Ark	3.5393e−01	1.7034e−03	4.6522e−03
		Aerr	5.5419e+00	1.7034e−03	4.6522e−03
Pol. Err.	Mean	Ark	1.3301e−04	5.9877e−08	9.9371e−07
		Aerr	8.6050e−04	4.1797e−07	6.8278e−07
	Med	Ark	1.7019e−09	1.5064e−12	1.9718e−12
		Aerr	1.8382e−09	1.4921e−12	1.4992e−12
	Min	Ark	7.1054e−15	7.1054e−15	3.5527e−15
		Aerr	7.1054e−15	7.1054e−15	3.5527e−15
	Max	Ark	3.5356e−01	1.3454e−04	5.4953e−03
		Aerr	6.1733e+00	3.1182e−03	5.4953e−03

Table 2.8.: Correctly found sizes and reconstruction errors for the overall function and the poles of a function of degree 7 with $r_{\text{zer}} = 50$ for symmetric sample data.

like to point out that for $n_{\text{add}} = 200$ and $n_{\text{add}} = 500$ all maximum errors are even better than the mean errors for non-symmetric samples. We also computed the data for degrees 5 and 9. In both cases, we saw similar improvements.

The only surprising observation is that the number of correctly found degrees is considerably lower for $n_{\text{add}} = 200$ for both variants and even more so for the error stopping criterion and $n_{\text{add}} = 500$. We close this section by once more looking at the number of correctly found degrees. We present the data for non-symmetric and symmetric samples in Table 2.9.

Let us first point out what is probably most striking. If n_{add} is large, then Ioniță's algorithm fails (almost) completely for symmetric data, while for $n_{\text{add}} = 50$ the results for symmetric data are better for degree 7 and 9. Especially for degree 9, the performance of Ioniță's algorithm with symmetric data is only slightly worse than that of the AAA algorithm with the error stopping criterion for non-symmetric data. While adding more points always decreases the performance of Ioniță's algorithm, we unfortunately cannot explain why this is so dramatic for symmetric data.

		non-symmetric			symmetric		
deg	n_{add}	50	200	500	50	200	500
5	Ion	9967	9953	9928	9859	9530	1099
	Ark	9482	9780	9863	9741	9541	9775
	Aerr	9741	9746	9740	9889	9542	9719
7	Ion	8696	8582	8368	9196	373	0
	Ark	8559	9182	9193	9279	8793	9289
	Aerr	8848	9083	9136	9522	8731	8735
9	Ion	4682	4554	4197	7152	0	0
	Ark	6049	6804	6668	8588	7991	8594
	Aerr	6129	6977	7198	8888	7731	7741

Table 2.9.: Correctly found degrees of functions with $r_{\text{zer}} = 50$ for symmetric and non-symmetric samples.

Let us comment on some subtler observations. We will only consider the two AAA variants from now on. For $n_{\text{add}} = 50$ the performance of both variants is always better for symmetric data. For $n_{\text{add}} = 500$, this is true for degree 9 for both variants and for degree 7 only for the rank stopping criterion. Moreover, for $n_{\text{add}} \geq 200$, the rank stopping criterion always performs better than the error stopping criterion, with the small exception of degree 5 and $n_{\text{add}} = 200$ for symmetric data, where the error stopping criterion finds the correct degree one time more.

Interestingly, the performance of both variants goes down for $n_{\text{add}} = 200$, before then increasing again. This effect is far more pronounced for the rank stopping criterion. For the error stopping criterion, the numbers stay almost the same. This is especially interesting in contrast to the non-symmetric data, where we saw that the performance of the algorithm with error stopping criterion constantly increased.

We investigated this phenomenon further. Let $r_{\text{s}} = \deg(q) + 1 + \lceil \frac{n_{\text{add}}}{2} \rceil$ denote the range of symmetric sample points, i.e., all sample points are in the interval $[-r_{\text{s}}, r_{\text{s}}]$. We looked at the number of correctly found degrees for different degrees and combinations of r_{zer} and r_{s}. Independent of the degree we observed, as already mentioned above, that choosing $n_{\text{add}} > 0$ improves the performance. However, after initially going up, the number of correctly found degrees goes down as r_{s} gets closer to r_{zer}. For $r_{\text{s}} \approx r_{\text{zer}}$, we observed the worst performance. Note that $r_{\text{s}} \approx r_{\text{zer}}$ means that the interval from which we take the samples coincides almost exactly with the interval in which all poles and zeros of the rational function lie. As $r_{\text{s}} > r_{\text{zer}}$ gets larger, the performance then increases again. But even for very large numbers of n_{add} the number of correctly found degrees is usually much lower then the best observations we made. These best results usually occur for $r_\mathrm{s} \approx \frac{r_\mathrm{zer}}{4}$. We got comparable results when using a sample distance of 2 instead of 1, which means that for the same r_s we had roughly half the number of added sample points. This implies that, at least for symmetric sample points, the range in which we sample is even more important than the number of additional samples.

We also tried to sample only outside of $[-r_\mathrm{zer}, r_\mathrm{zer}]$. Regardless of n_add this gives better results than for $r_\mathrm{s} \approx r_\mathrm{zer}$. However, it usually takes more additional samples to achieve the same performance as for $r_\mathrm{s} \approx \frac{r_\mathrm{zer}}{4}$. We suspect that the poor performance for $r_\mathrm{s} \approx r_\mathrm{zer}$ has something to do with the fact that in this case there is a relatively large fraction of sample points which lie in some sense close to a pole. However, we did not investigate this any further.

2.5.5. Tolerance for the Error Stopping Criterion

Since from now on we will only be looking at the two AAA variants, there is one more parameter we have not yet talked about, namely the tolerance used for the error stopping criterion. Until now, we performed all our experiments using the relative tolerance $\max_{1\leq j\leq M} |y_j| \cdot \tilde{\epsilon}$ with $\tilde{\epsilon} = 10^{-13}$, where $y_1, y_2, \ldots, y_M$ denote the given sample values. This relative tolerance is suggested in [80].

We tried different values for $\tilde{\epsilon}$ and found that in our setting $\tilde{\epsilon} = 10^{-12}$ often gives the best results, where we again measured how good the algorithm performs in terms of the number of times the degree of the original function is correctly determined. To make this statement more precise, for symmetric samples, we always get better results for $\tilde{\epsilon} = 10^{-12}$. For non-symmetric data, the picture is not as clear. Our experiments seem to point to the conclusion that the strongest influence on which tolerance works best is the magnitude of r_zer. The lower r_zer, the better the performance of $\tilde{\epsilon} = 10^{-13}$ seems to be compared to $\tilde{\epsilon} = 10^{-12}$. This phenomenon gets more prominent for larger degrees and lower numbers of n_add. Let us have a closer look at some data to see in more detail what is happening. The upper half of Table 2.10 displays the same data as the lower half of Table 2.5.

As we would expect, we see that lowering the tolerance to $\tilde{\epsilon} = 10^{-12}$ leads to more final matrices where the column size is too small, in fact more than twice as many as for $\tilde{\epsilon} = 10^{-13}$. On the other hand, the number of matrices where the columns size is too large for $\tilde{\epsilon} = 10^{-12}$ is only about a quarter of the number for $\tilde{\epsilon} = 10^{-13}$. This explains why the number of correctly found sizes is, with the sole exception of $n_\mathrm{add} = 50$, always larger for $\tilde{\epsilon} = 10^{-12}$. Interestingly though, for both $\tilde{\epsilon} = 10^{-13}$ and

		size correct				size too low			size too high			
$\tilde{\epsilon}$	n_{add}	c	l	h	sum	c	l	sum	c	l	h	sum
10^{-13}	50	8468	7	373	8848	86	436	522	1	0	629	630
	100	8855	45	99	8999	12	345	357	22	0	622	644
	500	8843	293	0	9136	0	218	218	264	7	375	646
	1000	8703	448	0	9151	0	199	199	379	10	261	650
	1500	8627	529	0	9156	0	193	193	435	14	202	651
10^{-12}	50	8013	0	809	8822	570	456	1026	0	0	152	152
	100	8625	0	468	9093	335	416	751	0	0	156	156
	500	9212	49	78	9339	21	482	503	4	0	154	158
	1000	9140	195	37	9372	0	467	467	9	0	152	161
	1500	9086	284	14	9384	0	455	455	17	0	144	161

Table 2.10.: Comparison of tolerances for the error stopping criterion. Number of times the size of the reconstruction matrix is correct, too low, or too high, as well as how often the rank of the matrix is correct (c), too low (l), or too high (h) for original functions of degree 7 with $r_{\text{zer}} = 50$.

$\tilde{\epsilon} = 10^{-12}$, the total number of samples where the final matrix has too many columns stays relatively constant regardless of n_{add}.

Regarding the distribution of ranks, we see similar trends for both tolerances. In both cases, we can observe different correlations of the rank and the final size of the reconstruction matrix depending on n_{add}. If n_{add} is high, then there seems to be a strong correlation between the size of the final matrix being too low and the rank being too low. Generally, the size and the rank being too low seem to be stronger correlated for $\tilde{\epsilon} = 10^{-13}$. On the other hand, if n_{add} is low, then there seems to be a strong correlation between the size and the rank of the final matrix being too high. Here, the correlation, regardless of n_{add}, seems to be stronger for $\tilde{\epsilon} = 10^{-12}$. For matrices where the size was detected correctly, we see that for both tolerances the number of times the rank is too high goes down with increasing n_{add}, while the number of times the rank is too low goes up. For $\tilde{\epsilon} = 10^{-12}$, the first effect is stronger and the second weaker. This leads to a stronger correlation between the correctly found rank and size. Overall, for $n_{\text{add}} \geq 500$, there seems to be a stronger correlation between the rank and the size of the final matrix for $\tilde{\epsilon} = 10^{-12}$ than for $\tilde{\epsilon} = 10^{-13}$. This puts the former variant closer to the rank criterion.

Now note that while lower values of r_{zer} tend to lead to better performance, higher degrees and lower numbers of n_{add} tend to lead to worse performance, both regarding the number of correctly found degrees and the overall reconstruction error. We observed that, especially for low r_{zer}, increasing the degree leads to an increased number of final matrices with too small column size. Naturally, this increase is stronger for $\tilde{\epsilon} = 10^{-12}$. On the other hand, the increase of final matrices with column size too large is stronger for $\tilde{\epsilon} = 10^{-13}$. However, even for $\tilde{\epsilon} = 10^{-13}$, the number of matrices which have a too small column size has a greater impact on the overall performance. Therefore, the slower increase of the number of these matrices leads to $\tilde{\epsilon} = 10^{-13}$ performing better at higher degrees. We show this effect in the following Table 2.11.

degree	$\tilde{\epsilon}$	s. cor.	s. low	s. high
5	10^{-13}	9917	5	78
	10^{-12}	9962	32	6
6	10^{-13}	9804	72	124
	10^{-12}	9765	223	12
7	10^{-13}	9386	374	240
	10^{-12}	9158	809	33
8	10^{-13}	8432	1253	315
	10^{-12}	7857	2091	52

Table 2.11.: Comparison of tolerances for the error stopping criterion regarding the size of the reconstruction matrix for a function with $r_{\text{zer}} = 30$ and $n_{\text{add}} = 50$ for non-symmetric samples.

For comparison, we show the same examples again, this time with symmetrical data, in Table 2.12.

degree	$\tilde{\epsilon}$	s. cor.	s. low	s. high
5	10^{-13}	9852	0	148
	10^{-12}	9985	0	15
6	10^{-13}	9724	0	276
	10^{-12}	9967	0	33
7	10^{-13}	9512	0	488
	10^{-12}	9921	0	79
8	10^{-13}	9210	0	790
	10^{-12}	9858	8	134

Table 2.12.: Comparison of tolerances for the error stopping criterion regarding the size of the reconstruction matrix for a function with $r_{\text{zer}} = 30$ and $n_{\text{add}} = 50$ for symmetric samples.

For symmetric samples, $\tilde{\epsilon} = 10^{-12}$ clearly performs better. However, what this shows is that without additional knowledge, for example about the distribution of the sample points, it is hard to give a general recommendation on which tolerance should be used.

When comparing the reconstruction errors for the two tolerances, we found that they are very similar. Interestingly, if n_{add} is low, then $\tilde{\epsilon} = 10^{-12}$ produces slightly smaller errors. This is in line with the observation we made before when comparing the rank and error criterion and found that the rank criterion actually produces better errors. However, for large n_{add} we found that $\tilde{\epsilon} = 10^{-13}$ produces slightly better errors.

2.5.6. Froissart Doublet Removal and Reconstruction of Unattainable Points

First, let us look at how adding a Froissart doublet removal process affects the AAA algorithm in the case where there are no distorted values in the data set. Recall that both stopping criteria sometimes lead to the final matrix having a too large column size, see Table 2.5. This implies that, at least in theory, the function reconstructed from these matrices will contain Froissart doublets. We would therefore expect that adding a Froissart doublet removal process will increase the performance of both variants in terms of how often they find the correct degree of the original function. And indeed, for the error stopping criterion, this is what we observe regardless of how the parameters, i.e., the degree of the original function, r_{zer}, or n_{add}, are chosen, or whether we use symmetrical or unsymmetrical data. For the rank stopping criterion, we see a different picture. To illustrate this, we use the same examples as in Table 2.6 and Table 2.7 and show how often the correct degree is found in Table 2.13.

		deg = 5			7			9		
Meth	fdr	50	200	500	50	200	500	50	200	500
Ark	no	9482	9780	9863	8559	9182	9193	6049	6804	6668
	yes	9483	9781	9863	8562	9187	9182	6059	6808	6591
Aerr	no	9741	9746	9740	8848	9083	9136	6129	6977	7198
	yes	9771	9778	9775	9043	9279	9332	6533	7418	7652

Table 2.13.: Number of times the AAA variants find the correct degree of a function with $r_{\text{zer}} = 50$ for non-symmetric samples and a tolerance of $\tilde{\epsilon} = 10^{-13}$ for the error stopping criterion with and without Froissart doublet removal (fdr) and different n_{add}.

As mentioned before, we see an increased performance for the error stopping criterion throughout. For the rank stopping criterion, we see just slightly better performance for $n_{\text{add}} = 50$ and $n_{\text{add}} = 200$. For $n_{\text{add}} = 500$, we get the same results for degree 5 and even worse results for degree 7 and 9. This can be explained as follows.

Recall that we have seen in Table 2.5 that for growing n_{add}, the error stopping criterion produces more matrices where the column size is too large than the rank criterion. The rank criterion, on the other hand, produces fewer matrices where the column size is too large and therefore more where the column size is too low. Now, as mentioned above, Froissart doublets will usually arise from matrices where the column size is too large.

And indeed, looking at the data in more detail, we found that the Froissart doublet removal process always decreased the number of matrices where the column array was too large. This effect was more prominent for the error stopping criterion as well. So not only were there more matrices for which Froissart doublets could be removed but there were, relatively speaking, also more matrices where this was actually done. This is probably due to the fact that, while the overall number of matrices where the column size is too large stays relatively constant for the error criterion, the number of matrices which at the same time possess the correct rank grows as n_{add} gets larger. Hence, the nullity of these matrices gets larger than 1. Now, in an exact algebraic setting, if a Löwner matrix has a nullity of n and we construct a rational function from a kernel vector which has only non-zero entries then this function will have $n-1$ Froissart doublets. It is likely that this is what leads to the observation that for growing numbers of n_{add} more and more matrices which initially have a too large column size are resized during the Froissart doublet removal process.

Using symmetrical data even increases this effect. Recall that we saw in Table 2.9 that for symmetrical data and larger n_{add} the error stopping criterion performed noticeably worse than the rank criterion. Adding the Froissart doublet removal process leads to both criteria performing very similarly.

For both variants, we observed that sometimes the number of matrices where the column size was too low also grew in the process of removing Froissart doublets. This means that some matrices for which the size was initially found correctly had also elements removed from the column array or that too many Froissart doublets were removed from a matrix which was initially too large. However, the increase of matrices where the column size was too small was much lower than the decrease of matrices where the column size was initially too large. Moreover, as discussed earlier, Froissart doublets do not only arise in a numerical context but can be embedded into the original function. So it is possible that a function for which the degree was initially found correctly simply happened to have one or more pole-zero pairs lying close together, which were then removed.

Another factor that influences how the rank stopping criterion works is the tolerance we use in order to detect unattainable points in the row array. Recall from Section 2.2 that for the error stopping criterion all unattainable points are contained in the column array once the algorithm stops. For the rank criterion, on the other hand, it can happen that some unattainable points are still in the row array. These are found, as described in Algorithm 2.3, by choosing a threshold and sorting out all elements of the row array which are too far away from the value of the reconstructed function at the same point. In order to produce the data presented above we chose this threshold to be $\max_{1 \leq j \leq M} |y_j| \cdot \tilde{\epsilon}$ with $\tilde{\epsilon} = 10^{-12}$, where $y_1, y_2, \ldots, y_M$ denote the given sample values. Bear in mind that we used the same threshold with $\tilde{\epsilon} = 10^{-13}$ for the error stopping criterion. As it turns out, this threshold is too restrictive. It even happened for higher degrees and higher numbers of n_{add} that the algorithm classified more points in the row array as unattainable, as there were in total left in the column array after removing points for which the weight was too small. Recall that we need to remove the same number of points from the column array as there are unattainable points in the row array. For example, for degree 9 and $n_{\text{add}} = 500$ this even happened 241 times.

Now it is important to find a good balance for the threshold for removing unattainable points from the row array. If we choose it too low, then too many points will be classified as unattainable, since the overall reconstruction error is not as low as the threshold, as has happened here. Choosing this parameter too high might lead to unattainable points in the row array being overlooked. We found that $\tilde{\epsilon} = 10^{-8}$ gives satisfactory results and seems to be a good balance. In fact, using this parameter the number of times the degree of the original function is found correctly in the above example is always the same with and without Froissart doublet removal.

We also have to shortly talk about the threshold for the weights in order to sort out unattainable points in the column array. The above data was produced using a threshold of $10^{-8} \cdot \frac{1}{\dim(\mathbf{w})}$. Here, $\dim(\mathbf{w})$ denotes the size of the weight vector $\mathbf{w}$, in MATLAB terms, if `w` is the weight vector, then this quantity is `length(w)`. The reasoning behind this scaling is that the SVD returns normed singular vectors, which means that the larger the vector is, the smaller its entries (in absolute terms) tend to be. Generally, we observed that for data without or with just a few unattainable points lower thresholds worked very well. For more distorted values as well as higher degrees, we needed a higher threshold. We found that $10^{-5} \cdot \frac{1}{\dim(\mathbf{w})}$ works very well and strikes a good balance here.

Regarding the reconstruction errors of the overall function and the poles, there is not much to say. We did not observe any significant differences there. Hence, we do not report these errors here in detail.

But now let us turn to the question how well the algorithms perform if the data contains unattainable points. It turns out that adding more distorted points decreases the performance of the algorithms, again measured in terms of how often they find the correct degree of the given function. We will show this behavior in the following example. In order to produce this data, we again used functions of degree 7 with $r_{\text{zer}} = 50$. Further we chose $n_{\text{add}} = 100$ and, in order to ensure a meaningful comparison to all our previous examples, used non-symmetric data and $\tilde{\epsilon} = 10^{-13}$ for the error stopping criterion. We randomly chose n_{dist} samples, which were then distorted by a randomly selected value in the set $[-1.5 \cdot \bar{y}, -0.5 \cdot \bar{y}] \cup [0.5 \cdot \bar{y}, 1.5 \cdot \bar{y}]$, where $\bar{y}$ denotes the median of the given sample values.

Table 2.14 shows first the number of functions for which the degree of the original function is detected correctly, then the number of functions where the unattainable points are found correctly, i.e., the distorted samples are correctly identified as such, and finally the number of functions for which both the degree and the unattainable points are found correctly. This is followed by some statistics on the reconstruction errors of the distorted values, i.e., if $y = y^* + \tilde{y}$ is a sample value, where y^* is the value of the reconstructed function and $\tilde{y}$ some added distortion, we look at how well $\tilde{y}$ is reconstructed. For each function, the error reported is the maximal distance between $\tilde{y}$ and its reconstruction, i.e., the error is measured in the ∞-norm. For these statistics, we used all samples where the unattainable points were correctly identified and not just those, where also the rank was found correctly.

We first observe that with growing n_{dist} the number of correctly found degrees goes down. This effect appears to be stronger for the error stopping criterion. So while for $n_{\text{dist}} \leq 4$ the error criterion delivers a better performance, for $n_{\text{dist}} \geq 5$, the rank criterion finds the correct degree more often. We also see that the total number of correctly identified unattainable points is extremely high and close to the total number of samples. Here, the error stopping criterion always performs better than the rank criterion. Also, the number of times both the degree and the unattainable points

	n_{dist}	0	1	2	3	4	5	6
s. cor.	Ark	8957	8953	8942	8915	8873	8809	8743
	Aerr	9199	9159	9098	9038	8946	8784	8628
dist. cor.	Ark		9957	9910	9857	9796	9681	9599
	Aerr		9984	9971	9938	9926	9880	9826
both	Ark		8946	8934	8899	8848	8759	8689
	Aerr		9155	9090	9011	8913	8731	8534
Mean	Ark		8.8341e−10	1.9653e−09	4.4324e−08	8.0976e−08	1.7451e−02	1.8028e−02
	Aerr		2.3929e−09	1.5986e−09	2.1437e−08	6.0529e−08	8.7353e−08	2.5591e−07
Med	Ark		9.3601e−18	3.7229e−17	6.8901e−17	9.7145e−17	1.3260e−16	1.9039e−16
	Aerr		9.3807e−18	3.5670e−17	6.0776e−17	8.7425e−17	1.1396e−16	1.4823e−16
Min	Ark		0	0	0	3.7482e−25	3.8774e−26	1.4863e−25
	Aerr		0	0	0	3.7482e−25	1.4217e−25	4.3944e−25
Max	Ark		4.2528e−06	4.5810e−06	3.3743e−04	3.5575e−04	1.6894e+02	1.7305e+02
	Aerr		1.5671e−05	4.2397e−06	7.8437e−05	2.4836e−04	6.2367e−04	1.5568e−03

Table 2.14.: Number of times the column size is found correctly, the distorted points are identified correctly and both simultaneously as well as mean, median, minimum, and maximum errors of the reeconstructed distortion value for correctly identified unattainable points.

are detected correctly is very close to the number of times just the degree is found correctly.

The reconstruction errors are extremely low. Especially the minimal error is even 0 for $n_{\text{dist}} \leq 3$. Interestingly, the maximum error for the rank criterion for $n_{\text{dist}} \geq 5$ is extremely large. Unfortunately, we do not have a good explanation for this. However, looking at the corresponding median errors, we see that these cases are clearly outliers. Moreover, these large errors completely disappear if we use symmetrical samples.

Two more final remarks. Regarding the overall reconstruction error and the error in the poles, we did not see any significant differences whether we added distorted values or not. And lastly, unsurprisingly, the magnitude of the distortion values influences how well unattainable points are detected. The larger the distortion, the better the performance. We also experimented with adding distortions where we used the mean of the given sample values as a reference instead of the median. Since the mean of the sample values is usually larger than the median, we observed that the unattainable points are correctly identified even more often than in the above-reported cases.

2.5.7. Conclusions

After this long and detailed chapter let us briefly sum up our most important findings. What is absolutely clear by now is that the algorithms based on Löwner matrices, i.e., Ioniță's algorithm and the AAA algorithm, are far more stable and deliver a much better performance than the naive algorithm using a Vandermonde-type reconstruction matrix.

Generally speaking, increasing r_{zer} and the degree of the function we want to reconstruct decreases the performance of all algorithms both with regard to the number of times the correct degree is found as well as with regard to the reconstruction errors. Increasing n_{add} increases the performance of the AAA algorithm and, at the same time, decreases the performance of Ioniță's algorithm. In particular, for $n_{\text{add}} = 0$, Ioniță's algorithm is usually better, while ,for $n_{\text{add}} > 0$, the AAA algorithm delivers better results. However, increasing n_{add} indefinitely does not keep increasing its performance. Moreover, the partition of the sample set constructed by the AAA algorithm for $n_{\text{add}} > 0$ seems to be better than the alternating partition used by Ioniță's algorithm.

Something we have not studied here in any depth, but which is of course always important when we consider interpolation, is how to cleverly choose the sample points. We saw that the sample points have a large impact on how well the reconstruction process works. In particular, we saw that since the functions we wanted to reconstruct had both positive as well as negative poles and zeros, choosing both positive and negative sample points gives better results. Even more, we saw that the quality of the reconstruction seems to be related to how far the range of the sample points covers the range of the zeros and poles. However, to our knowledge, there does not exist a theory on how to best sample rational functions. In practice, usually equispaced or Chebychev points are used, see for example [64].

While we could observe clear trends for the parameters discussed so far, this was not always the case. It is not completely clear how to choose a suitable tolerance for the error stopping criterion of the AAA algorithm, since the outcome depends strongly on the previously mentioned parameters. Moreover, while both AAA variants display a very similar performance, we cannot say if one is generally better than the other. Again, depending on the above parameters, sometimes the one and sometimes the other gives better results.

Part II.

Reconstruction of Exponential Sums from Fourier Coefficients

3. From Trigonometric Polynomials to Exponential Sums

3.1. Periodicity

Periodicity, or rather lack thereof, will play an important role later on when we consider the Fourier series expansion, respectively the Fourier coefficients, of exponential sums. We therefore shortly review some properties of periodic functions.

Definition 3.1. We say that a non-constant function $f : \mathbb{R} \to \mathbb{C}$ is *periodic* if there exists a number $P \in \mathbb{R}_{>0}$ such that

$$f(t + P) = f(t) \quad \forall t \in \mathbb{R}.$$

The smallest $P \in \mathbb{R}_{>0}$ such that this equation holds is called the *fundamental period* of f.

Lemma 3.2 (Properties of Periodic Functions)**.** *Let f be a periodic function with fundamental period P. Then the following hold:*

(i) f is $n \cdot P$-periodic for any $n \in \mathbb{N}$.
(ii) If f is R-periodic for some $R \in \mathbb{R}_{>0}$, then $R = n \cdot P$ for some $n \in \mathbb{N}$.
(iii) The function $f(\bullet + c)$ for any fixed $c \in \mathbb{R}$ has also fundamental period P.

Proof. (i) and (iii) are obvious.

(ii) Since by assumption P is the smallest number such that $f(t) = f(t + P)$ for all $t \in \mathbb{R}$, we must have $R \geq P$. Further, by (i),

$$f(t) = f(t - n \cdot P + n \cdot P) = f(t - n \cdot P) = f(t - n \cdot P + R)$$

for any $n \in \mathbb{N}$. Therefore f is $(R - n \cdot P)$-periodic for any $n \in \mathbb{N}$ such that $R - n \cdot P > 0$. Now choose $\tilde{n} \in \mathbb{N}$ such that $0 \leq R - \tilde{n} \cdot P < P$. It cannot happen that $0 < R - \tilde{n} \cdot P < P$ since this contradicts the assumption that f has fundamental period P. Hence, we must have $R = \tilde{n} \cdot P$. □

Theorem 3.3. *Let f and g be continuous periodic functions with fundamental periods P and Q respectively, i.e., $f(t) = f(t+P)$ and $g(t) = g(t+Q)$ for all $t \in \mathbb{R}$, and suppose that $f+g$ is not constant. Then the function $f+g$ is periodic if and only if there exist $n_P, n_Q \in \mathbb{N}$ such that $n_P \cdot P = n_Q \cdot Q$.*

Proof. (i) Suppose there exist $n_P, n_Q \in \mathbb{N}$ such that $n_P \cdot P = n_Q \cdot Q$ and define $R \coloneqq n_P \cdot P = n_Q \cdot Q$. Since, by Lemma 3.2(i), both f and g are R-periodic, we have

$$\begin{aligned}(f+g)(t) &= f(t) + g(t)\\ &= f(t+R) + g(t+R)\\ &= (f+g)(t+R),\end{aligned}$$

i.e., $f+g$ is R-periodic.

(ii) Suppose $f+g$ is periodic with fundamental period $R \in \mathbb{R}_{>0}$, i.e.,

$$f(t) + g(t) = (f+g)(t) = (f+g)(t+R) = f(t+R) + g(t+R)$$

for all $t \in \mathbb{R}$. We can rewrite this equation as

$$f(t) - f(t+R) = g(t+R) - g(t). \tag{3.1}$$

Now distinguish two cases. First, suppose that both sides of equation (3.1) are constant, i.e., $f(t) - f(t+R) \equiv g(t+R) - g(t) \equiv c$ for some $c \in \mathbb{R}$. In particular this implies that $f(t) = f(t+R) + c$. But then also $f(t+R) = f(t+2\cdot R) + c$. Inserting this into the first equation we find that $f(t) = f(t+2\cdot R) + 2\cdot c$ and it follows inductively that $f(t) = f(t+n\cdot R) + n\cdot c$ for any $n \in \mathbb{N}$.

If $c \neq 0$, we conclude that f is unbounded. But, since f is periodic, this entails that it has at least one pole in every interval $[0, P)$, which contradicts the assumption that f is continuous. If, on the other hand, $c = 0$ it follows that $f(t) = f(t+R)$ as well as $g(t) = g(t+R)$ and hence, by Lemma 3.2(ii), $R = n_P \cdot P$ and $R = n_Q \cdot Q$, i.e., $n_P \cdot P = n_Q \cdot Q$, for some $n_P, n_Q \in \mathbb{N}$.

Now suppose that both sides of equation (3.1) are not constant. Since f and, by Lemma 3.2(iii), $f(\cdot + R)$ have both fundamental period P, it follows with (i) that $f(t) - f(t+R)$ is P-periodic and analogously $g(t+R) - g(t)$ is Q-periodic. Let $\tilde{P}$ be the fundamental period of $f(t) - f(t+R)$ and $g(t+R) - g(t)$. Then it follows from Lemma 3.2(ii) that there exist $n_P, n_Q \in \mathbb{N}$ such that $P = n_Q \cdot \tilde{P}$ and $Q = n_P \cdot \tilde{P}$, i.e., $\tilde{P} = \frac{P}{n_Q} = \frac{Q}{n_P}$ and hence $n_P \cdot P = n_Q \cdot Q$. □

Remark 3.4. The condition that $n_P \cdot P = n_Q \cdot Q$ in Theorem 3.3 implies that $\frac{P}{Q} = \frac{n_Q}{n_P} \in \mathbb{Q}_{>0}$. If two numbers $x, y \in \mathbb{R} \setminus \{0\}$ satisfy $\frac{x}{y} \in \mathbb{Q} \setminus \{0\}$, then they are called *commensurable.* A set of more than two numbers is called commensurable if all its elements are pairwise commensurable.

Similarly, we have $P = \frac{n_Q}{n_P} \cdot Q$, where $\frac{n_Q}{n_P} \in \mathbb{Q}_{>0}$. More generally, if for $x_1, x_2, \dots, x_K \in \mathbb{R}$ with $K \in \mathbb{N}$ the equation $\sum_{k=1}^{K} q_k \cdot x_k = 0$ has a solution with $q_k \in \mathbb{Q} \setminus \{0\}$ for $k = 1, 2, \dots, K$, then these numbers are called *rationally dependent.*

Commensurability entails rational dependence. However, for $K \geq 2$ the reverse is not true. For example, the numbers $\sqrt{2}$, $\sqrt{2}+\sqrt{3}$, and $\sqrt{3}$ are rationally dependent but not commensurable. Nonetheless, the terms commensurable and rationally dependent are often used interchangeably. Moreover, although these terms are used frequently, especially in the engineering literature, they are hardly ever rigorously defined.

We show an equivalent definition of commensurability, which we will later use.

Lemma 3.5. *A set of numbers $x_1, x_2, \dots, x_K \in \mathbb{R} \setminus \{0\}$ with $K \in \mathbb{N}$ is commensurable if and only if there exists an $a \in (\mathbb{R} \setminus \mathbb{Q}) \cup \{1\}$ such that $x_k = a \cdot q_k$ with $q_k \in \mathbb{Q} \setminus \{0\}$ for all $k = 1, 2, \dots, N$. We also say that $x_1, x_2, \dots, x_K \in a \cdot \mathbb{Q} \setminus \{0\}$.*

Proof. (i) Suppose that there exists an $a \in \mathbb{R} \setminus \mathbb{Q} \cup \{1\}$ such that $x_k = a \cdot q_k$ with $q_k \in \mathbb{Q} \setminus \{0\}$ for all $k = 1, 2, \dots, N$. For any $1 \leq m < n \leq N$ we then have $\frac{x_m}{x_n} = \frac{a \cdot q_m}{a \cdot q_n} = \frac{q_m}{q_n} \in \mathbb{Q} \setminus \{0\}$. Hence, $x_1, x_2, \dots, x_K$ are commensurable.

(ii) Now suppose that $x_1, x_2, \dots, x_K$ are commensurable. Write $x_1 = a \cdot q_1$ for $a \in (\mathbb{R} \setminus \mathbb{Q}) \cup \{1\}$ and $q_1 \in \mathbb{Q} \setminus \{0\}$. Note that any non-zero real number can be written in this form. Since $x_1, x_2, \dots, x_K$ are commensurable, it follows in particular that $\frac{x_k}{x_1} = \tilde{q}_k \in \mathbb{Q} \setminus \{0\}$ for all $k = 2, 3, \dots, K$. Therefore, we have $x_k = x_1 \cdot \tilde{q}_k = a \cdot q_1 \cdot \tilde{q}_k =: a \cdot q_k$, where $q_k \in \mathbb{Q} \setminus \{0\}$. □

Theorem 3.3 is actually well known. We decided to show the proof here anyway because it helps to develop an intuition to better understand the concepts we will present in the next sections. Showing when the sum of an arbitrary number of continuous periodic functions is again periodic requires a little more work and further assumptions. We reference here a result from [78].

Theorem 3.6 ([78, Theorem 2.1])**.** *Let $f_1, f_2 \dots, f_K$ for $K \in \mathbb{N}$ be continuous periodic functions. If all partial sums of these functions with less than K terms are non-constant, then the sum $f_1 + f_2 + \dots + f_K$ is periodic if and only if the periods of the summands are commensurable.*

Remark 3.7. Note that while part (i) of the proof of Theorem 3.3 can be easily generalized to arbitrary sums, the condition that all partial sums must not be constant is needed to generalize the second part of the proof.

3.2. Trigonometric Polynomials and Their Generalizations

One of the most important classes of periodic functions are *(complex) trigonometric polynomials*, which are of the form

$$f(t) \coloneqq \eta_0 + \sum_{k=1}^{K} \eta_k \cdot \mathrm{e}^{2\pi\,\mathrm{i}\cdot\frac{n_k}{P}\cdot t} \tag{3.2}$$

for $t \in \mathbb{R}$ with $P \in \mathbb{R}_{>0}$, $K \in \mathbb{N}$, $\eta_0 \in \mathbb{C}$, $\eta_k \in \mathbb{C} \setminus \{0\}$ and $n_k \in \mathbb{Z} \setminus \{0\}$ for all $k = 1, 2, \ldots, K$. This notation is a little uncommon. However, we use it in order to have a consistent notation throughout this chapter.

It is easily seen that any non-constant component of this function, i.e., $\eta_k \cdot \mathrm{e}^{2\pi\,\mathrm{i}\cdot\frac{n_k}{P}\cdot t}$ for $k = 1, 2, \ldots, K$, is periodic with period P. Therefore, the function f is again periodic with period P. Trigonometric polynomials, or, in the case that $K = \infty$, trigonometric series, lie at the core of classical Fourier analysis. We will now look at ways to generalize trigonometric polynomials.

If P is the fundamental period of a trigonometric polynomial as in (3.2), then we say that this function has a *base frequency* of $\frac{1}{P}$. The first generalization we consider is a function with multiple base frequencies. Let $P_1, P_2, \ldots, P_N \in \mathbb{R}_{>0}$ be incommensurable. A *quasi-periodic function* is of the form

$$f(t) \coloneqq \eta_0 + \sum_{k_1,k_2,\ldots k_N=1}^{K} \eta_{k_1,k_2,\ldots,k_N} \cdot \mathrm{e}^{2\pi\,\mathrm{i}\cdot\left(\frac{n_{k_1}}{P_1}+\frac{n_{k_2}}{P_2}+\ldots+\frac{n_{k_N}}{P_N}\right)\cdot t},$$

with all parameters as above except that now $\eta_{k_1,k_2,\ldots,k_N} \in \mathbb{C}$. These functions occur naturally in Hamiltonian mechanics and are used to describe multi-periodic motions of integrable systems, see [2, Chapter 10, Appendix 8], as well as in nonlinear dynamics, see [62, Chapter 1]. Adding or multiplying trigonometric polynomials with incommensurable base frequencies results in a quasi-periodic function. Since the base frequencies are incommensurable, quasi-periodic functions are not periodic, or, put differently, they have a period of infinity. We will actually prove this shortly in Theorem 3.8.

Going one step further, we arrive at *almost-periodic functions*, which have the form

$$f(t) \coloneqq \eta_0 + \sum_{k=1}^{K} \eta_k \cdot \mathrm{e}^{2\pi\,\mathrm{i}\cdot a_k \cdot t}, \tag{3.3}$$

where again all parameters are as in (3.2) but this time $a_k \in \mathbb{R} \setminus \{0\}$ is arbitrary for $k = 1, 2, \ldots, K$. Almost-periodic functions have been extensively studied, see for example [15, 27]. More generally, the term almost-periodic function refers to a function which is in the closure of the set of all functions of form (3.3) with respect to different metrics. Some results from classical Fourier analysis can be generalized to the setting of almost-periodic functions. They are also used in the theory of differential equations, see for example [71]. If $K = \infty$, the term *nonharmonic Fourier series* is also used, see [85, 115]. Almost-periodic functions will usually be non-periodic, except in the case where the a_k for $k = 1, 2, \ldots, K$ are commensurable. We take a closer look at this in Theorem 3.8. Note that some authors call any function of the form (3.3) a trigonometric polynomial.

We will later also look at *real almost-periodic functions* of the form

$$f(t) \coloneqq \rho_0 + \sum_{k=1}^{K} \rho_k \cdot \cos(2\pi \cdot a_k \cdot t + b_k), \tag{3.4}$$

with $\rho_0 \in \mathbb{R}$, $\rho_k \in \mathbb{R}_{>0}$, $a_k \in \mathbb{R} \setminus \{0\}$, and $b_k \in [0, 2\pi)$ for $k = 1, 2, \ldots, K$. Indeed, writing $\operatorname{Re}(\eta_0) = \rho_0$ and $\eta_k = \rho_k \cdot \mathrm{e}^{\mathrm{i}\, b_k}$ for $k = 1, 2, \ldots, K$ in (3.3), we see that a real almost-periodic function is just the real part of an almost-periodic function.

W.l.o.g. we can further assume that $a_k \in \mathbb{R}_{>0}$ for all $k = 1, 2, \ldots, K$ since for $a < 0$ and $b \in [0, 2\pi)$ we have $\cos(a \cdot t + b) = \cos(-a \cdot t - b) = \cos(-a \cdot t + 2\pi - b)$, where $-a > 0$ and $2\pi - b \in (0, 2\pi]$. But whether we use the interval $(0, 2\pi]$ or $[0, 2\pi)$ for the phase shift does not matter. Even more, we can assume w.l.o.g. that all a_k are pairwise distinct since for $a, \rho, \gamma \in \mathbb{R}_{>0}$ and $b, c \in [0, 2\pi)$ with $b \neq c$ we find that

$$\rho \cdot \cos(a \cdot t + b) + \gamma \cdot \cos(a \cdot t + c) = \Gamma \cdot \cos\left(a \cdot t + \tfrac{b+c}{2} + \varphi\right)$$

with

$$\Gamma = \sqrt{\rho^2 + \gamma^2 + 2\rho\gamma \cdot \cos(b - c)}$$

and

$$\varphi = \operatorname{atan2}\left((\rho - \gamma) \cdot \sin\left(\tfrac{b-c}{2}\right), (\rho + \gamma) \cdot \cos\left(\tfrac{b-c}{2}\right)\right),$$

where atan2 is defined in (5.5). Since the proof of this is rather technical, yet not very insightful, we put it into Appendix A.

While it is clear that trigonometric polynomials are periodic, seeing that almost-periodic functions are generally non-periodic requires that we also check the additional condition from Theorem 3.6, i.e., that all partial sums are non-constant. It turns out that this condition is always satisfied for almost-periodic functions, as we will see in Lemma 3.10. However, we will give a direct proof of when an almost-periodic function is periodic or non-periodic in order to get a better understanding of their structure. As we will shortly discuss in the next section, this proof also applies to real almost-periodic functions.

Theorem 3.8. *A function of the form* $f(t) = \eta_0 + \sum_{k=1}^{K} \eta_k \cdot \mathrm{e}^{2\pi\,\mathrm{i}\cdot a_k \cdot t}$ *for* $\eta_0 \in \mathbb{C}$, $\eta_k \in \mathbb{C} \setminus \{0\}$*, and pairwise distinct* $a_k \in \mathbb{R} \setminus \{0\}$ *for* $k = 1, 2, \ldots, K$ *is periodic if and only if the set of* a_k *is commensurable.*

Proof. (i) If the set of a_k is commensurable, then, by Lemma 3.5, there exists an $a \in \mathbb{R} \setminus \mathbb{Q} \cup \{1\}$ such that $a_k \in a \cdot \mathbb{Q} \setminus \{0\}$ for $k = 1, 2, \ldots, K$. We can therefore rewrite f as

$$f(t) = \eta_0 + \sum_{k=1}^{K} \eta_k \cdot \mathrm{e}^{2\pi\,\mathrm{i}\cdot a_k \cdot t} = \eta_0 + \sum_{k=1}^{K} \eta_k \cdot \mathrm{e}^{2\pi\,\mathrm{i}\cdot a \cdot q_k \cdot t}.$$

Assume w.l.o.g. that all denominators of q_k for $k = 1, 2, \ldots, K$ are positive and let $\tilde{P} > 0$ be their least common multiple. Then $q_k \cdot \tilde{P} := n_k \in \mathbb{Z} \setminus \{0\}$. Setting $P := \frac{\tilde{P}}{a}$ we can then write

$$f(t) = \eta_0 + \sum_{k=1}^{K} \eta_k \cdot \mathrm{e}^{2\pi\,\mathrm{i}\cdot \frac{n_k}{P} \cdot t},$$

which is just a trigonometric polynomial and hence periodic. In particular, f has fundamental period P.

(ii) Now suppose that f is periodic with fundamental period P. Then we have

$$f(t) = \eta_0 + \sum_{k=1}^{K} \eta_k \cdot \mathrm{e}^{2\pi\,\mathrm{i}\cdot a_k \cdot t} = \eta_0 + \sum_{k=1}^{K} \eta_k \cdot \mathrm{e}^{2\pi\,\mathrm{i}\cdot a_k \cdot (t+P)} = f(t+P).$$

Rewriting this equation gives us

$$f(t) - f(t+P) = \sum_{k=1}^{K} \eta_k \cdot \left(1 - \mathrm{e}^{2\pi\,\mathrm{i}\cdot a_k \cdot P}\right) \cdot \mathrm{e}^{2\pi\,\mathrm{i}\cdot a_k \cdot t} \equiv 0. \tag{3.5}$$

We will see in Lemma 3.10 that, since all a_k are pairwise distinct, the functions $\mathrm{e}^{2\pi\,\mathrm{i}\cdot a_1 \cdot t}$,

$e^{2\pi\,\mathrm{i}\cdot a_2\cdot t}, \dots, e^{2\pi\,\mathrm{i}\cdot a_K\cdot t}$ are linearly independent over $\mathbb{C}$. Therefore, equation (3.5) can only hold if $e^{2\pi\,\mathrm{i}\cdot a_k\cdot P} = 1$ for all $k = 1, 2, \dots, K$. But this means that $a_k \cdot P \in \mathbb{Z} \setminus \{0\}$, which implies that for any $1 \le m < n \le K$ we have $\frac{a_m\cdot P}{a_n\cdot P} = \frac{a_m}{a_n} \in \mathbb{Q} \setminus \{0\}$, i.e., the set of a_k is commensurable. □

For completeness' sake let us give a justification for the term almost-periodic function. The following result holds for the more general set of functions called *uniformly almost-periodic functions*. These are functions contained in the closure of the set of all functions of form (3.3) with respect to the metric $d(f, g) := \sup_{t\in\mathbb{R}} |f(t) - g(t)|$, where f and g are functions mapping from $\mathbb{R}$ to $\mathbb{C}$.

Theorem 3.9. *A continuous function f is uniformly almost-periodic if and only if for every $\epsilon > 0$ there exists a number $l(\epsilon) > 0$ with the property that every interval of length $l(\epsilon)$ of the real line contains at least one number $\tau(\epsilon)$ such that*

$$|f(t + \tau(\epsilon)) - f(t)| < \epsilon \quad \textit{for all} \quad t \in \mathbb{R}.$$

The number $\tau(\epsilon)$ is called a translation number *or an* ϵ-almost-period *of f.*

Proof. The proof can be found by combining Theorems 1.9, 1.10, and 1.11 from [27]. □

Theorem 3.9 can be used as an alternative definition for uniformly almost-periodic functions. In fact, this is the definition used by Harald Bohr in [17], where he originally introduced the concept of almost-periodic functions.

Although all the concepts we have seen so far are fascinating subjects on their own, we do not go any deeper into the theory surrounding them. Rather, we use them as stepping stones on the way to *exponential sums*. These are functions of the form

$$f(t) := \eta_0 + \sum_{k=1}^{K} \eta_k \cdot e^{\phi_k\cdot t}, \tag{3.6}$$

where now $\phi_k \in \mathbb{C} \setminus \{0\}$. This is a further generalization of (3.3), since almost-periodic functions can be characterized as exponential sums where we demand that $\phi_k \in \mathrm{i}\,\mathbb{R} \setminus \{0\}$ for all $k = 1, 2, \dots, K$. However, there is a crucial difference between exponential sums and almost-periodic functions. While for the latter every term in the sum, except the constant term, is a periodic function, this is no longer true for exponential sums. If we write $\phi = 2\pi\,\mathrm{i}\cdot a + b$ for $a, b \in \mathbb{R}$, each non-constant summand, or component, of the exponential sum satisfies the equation

$$\mathrm{e}^{\phi\cdot\left(t+\frac{1}{a}\right)} = \mathrm{e}^{\frac{b}{a}} \cdot \mathrm{e}^{\phi\cdot t},$$

i.e., for $b \neq 0$, it is not a periodic function. Nonetheless, all our results from here on out do not only hold for exponential sums but also for almost-periodic functions and hence quasi-periodic functions and trigonometric polynomials.

In the literature, the term exponential sum often refers to a slightly different concept, namely functions of the form

$$f(t) \coloneqq \eta_0 + \sum_{k=1}^{K} \eta_k \cdot z_k^t, \tag{3.7}$$

with $z_k \in \mathbb{C} \setminus \{0, 1\}$. Now note that we can write every $z \in \mathbb{C} \setminus \{0\}$ as $z = \mathrm{e}^{\phi}$, where $\phi \in \mathbb{C}$ with $\operatorname{Im} \phi \in [0, 2\pi)$. Therefore, (3.7) can be viewed as a special case of (3.6). However, we briefly need to talk about some subtleties regarding the behavior of complex exponentials in order to justify this claim. For $\phi \in \mathbb{C}$ we generally have $\left(\mathrm{e}^{\phi}\right)^t \neq \mathrm{e}^{\phi\cdot t}$, which can be easily seen by looking at the example $\mathrm{e}^{2\pi\,\mathrm{i}\cdot t} \neq \left(\mathrm{e}^{2\pi\,\mathrm{i}}\right)^t = 1^t = 1$. In short, this is due to the multibranch nature of the complex logarithm. If we restrict the imaginary part of ϕ to $[0, 2\pi)$, then this effectively means that we fix one branch of the complex logarithm and therefore, for $\phi \in \mathbb{C}$ with $\operatorname{Im} \phi \in [0, 2\pi)$, we do have $\left(\mathrm{e}^{\phi}\right)^t = \mathrm{e}^{\phi\cdot t}$. This argument shows that although we will only work with exponential sums of the form (3.6), all our results also hold for functions of the form (3.7).

3.3. Uniqueness of Exponential Sums

We will spend the rest of this thesis developing methods to reconstruct exponential sums from samples of different nature. By reconstruction we mean the recovery of the parameters η_0, η_k, and ϕ_k for exponential sums as in (3.6) as well as ρ_0, ρ_k, a_k, and b_k for real almost-periodic functions as in (3.4) for $k = 1, 2, \ldots, K$. In order to show that this reconstruction even makes sense, we first have to show that all of these parameters are uniquely determined.

We give the following proofs only for exponential sums, which directly entails that the results also hold for almost-periodic functions. Further, we can write a real almost-periodic function as

$$f(t) = \rho_0 + \sum_{k=1}^{K} \rho_k \cdot \cos(2\pi \cdot a_k \cdot t + b_k) = \frac{\rho_0}{2} + \sum_{k=-K}^{K} \frac{\rho_k}{2} \cdot \mathrm{e}^{\mathrm{i}\, b_k} \cdot \mathrm{e}^{2\pi\,\mathrm{i}\cdot a_k \cdot t} \tag{3.8}$$

with $a_0 = b_0 = 0$, $b_{-j} = -b_j$, $a_{-j} = -a_j$ and $\rho_{-j} = \rho_j$ for $j = 1, 2, \ldots, K$. This is again an almost-periodic function, just with shifted indices, and therefore the proofs are also valid for real almost-periodic functions. In [90], we gave a uniqueness proof for real almost-periodic functions similar to the one presented here.

Lemma 3.10. *Let $\phi_1, \phi_2, \ldots, \phi_K \in \mathbb{C} \setminus \{0\}$ be pairwise distinct. Then the functions $1, \mathrm{e}^{\phi_1 \cdot t}, \mathrm{e}^{\phi_2 \cdot t}, \ldots, \mathrm{e}^{\phi_K \cdot t}$ are linearly independent over $\mathbb{C}$ on any interval $T \subseteq \mathbb{R}$ of non-zero length.*

Proof. We use the fact that $\mathrm{e}^{\phi \cdot t}$ satisfies the ordinary differential equation

$$\frac{\mathrm{d}^n}{\mathrm{d}t^n} \mathrm{e}^{\phi \cdot t} = \phi^n \cdot \mathrm{e}^{\phi \cdot t}$$

for any $n \in \mathbb{N}_0$.

Suppose that the functions 1, $\mathrm{e}^{\phi_k \cdot t}$ for $k = 1, 2, \ldots, K$ are linearly dependent on T. Then there exist $\eta_0, \eta_1, \ldots, \eta_K \in \mathbb{C}$ that are not all zero such that

$$\eta_0 + \sum_{k=1}^{K} \eta_k \cdot \mathrm{e}^{\phi_k \cdot t} \equiv 0$$

on T. W.l.o.g. assume that $\eta_k \neq 0$ for all $k = 0, 1, \ldots, K$, otherwise, i.e., if $\eta_{\tilde{k}} = 0$ for some $0 \leq \tilde{k} \leq K$, just leave out the corresponding function $\mathrm{e}^{\phi_{\tilde{k}} \cdot t}$.

Since the functions $\mathrm{e}^{\phi_k \cdot t}$ are smooth we can differentiate this sum K times and get the following system of equations

$$\begin{aligned} \eta_0 + \sum_{k=1}^{K} \eta_k \cdot \mathrm{e}^{\phi_k \cdot t} &\equiv 0, \\ \sum_{k=1}^{K} \phi_k \cdot \eta_k \cdot \mathrm{e}^{\phi_k \cdot t} &\equiv 0, \\ &\vdots \\ \sum_{k=1}^{K} \phi_k^K \cdot \eta_k \cdot \mathrm{e}^{\phi_k \cdot t} &\equiv 0, \end{aligned}$$

which we can rewrite as a matrix-vector multiplication to get

$$\underbrace{\begin{pmatrix} 1 & 1 & 1 & \dots & 1 \\ 0 & \phi_1 & \phi_2 & \dots & \phi_K \\ 0 & \phi_1^2 & \phi_2^2 & \dots & \phi_K^2 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & \phi_1^K & \phi_2^K & \dots & \phi_K^K \end{pmatrix}}_{=:\mathbf{V}} \cdot \underbrace{\begin{pmatrix} \eta_0 & 0 & 0 & \dots & 0 \\ 0 & \eta_1 & 0 & \dots & 0 \\ 0 & 0 & \eta_2 & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \dots & \eta_K \end{pmatrix}}_{=:\mathbf{E}} \cdot \underbrace{\begin{pmatrix} 1 \\ \mathrm{e}^{\phi_1 \cdot t} \\ \mathrm{e}^{\phi_2 \cdot t} \\ \vdots \\ \mathrm{e}^{\phi_K \cdot t} \end{pmatrix}}_{=:\mathbf{e}(t)} \equiv \mathbf{0}.$$

Now $\mathbf{V}$ is a square Vandermonde matrix and $\mathbf{E}$ is a square diagonal matrix. Hence, we can easily calculate the determinant

$$\det(\mathbf{V} \cdot \mathbf{E}) = \det(\mathbf{V}) \cdot \det(\mathbf{E}) = \prod_{0 \le m < n \le K} (\phi_n - \phi_m) \cdot \prod_{k=0}^{K} \eta_k,$$

where we have set $\phi_0 = 0$. Since $\eta_k \neq 0$ for $k = 0, 1, \dots, K$ and since all ϕ_k are pairwise distinct, we find that $\det(\mathbf{V} \cdot \mathbf{E}) \neq 0$. Hence, $\mathbf{V} \cdot \mathbf{E}$ has full rank and $\mathbf{V} \cdot \mathbf{E} \cdot \mathbf{e}(t) \equiv \mathbf{0}$ can only be true if $\mathbf{e}(t) \equiv \mathbf{0}$ on T. This, however, is clearly not the case. Note that this does not depend on the fact that $\eta_0 \neq 0$, which leads to the first entry of $\mathbf{e}(t)$ being 1, since none of the functions $\mathrm{e}^{\phi_k \cdot t}$ for $k = 1, 2, \dots, K$ constantly vanishes on any interval of non-zero length. $\square$

Theorem 3.11 (Uniqueness of Exponential Sums). *Let*

$$\eta_0 + \sum_{k=1}^{K} \eta_k \cdot \mathrm{e}^{\phi_k \cdot t} = \gamma_0 + \sum_{k=1}^{\tilde{K}} \gamma_k \cdot \mathrm{e}^{\psi_k \cdot t}$$

on an interval $T \subset \mathbb{R}$ of non-zero length, where $\phi_1, \phi_2, \dots, \phi_K \in \mathbb{C} \setminus \{0\}$ as well as $\psi_1, \psi_2, \dots, \psi_{\tilde{K}} \in \mathbb{C} \setminus \{0\}$ are each pairwise distinct, $\eta_0, \gamma_0 \in \mathbb{C}$ and $\eta_k, \gamma_k \in \mathbb{C} \setminus \{0\}$ for all $k = 1, 2, \dots, K$ (respectively $\tilde{K}$). Then $K = \tilde{K}$ and, after a suitable ordering of the terms, $\phi_k = \psi_k$ as well as $\eta_k = \gamma_k$ for all $k = 0, 1, \dots, K$.

Proof. Assume w.l.o.g. that $K \le \tilde{K}$ and $\eta_0 \neq 0$. If

$$\eta_0 + \sum_{k=1}^{K} \eta_k \cdot \mathrm{e}^{\phi_k \cdot t} - \gamma_0 - \sum_{k=1}^{\tilde{K}} \gamma_k \cdot \mathrm{e}^{\psi_k \cdot t} \equiv 0 \tag{3.9}$$

on T, then it follows from Lemma 3.10 and the assumption that $\eta_k, \gamma_k \neq 0$ as well as $\eta_0 \neq 0$, that the functions $1, \mathrm{e}^{\phi_k \cdot t}, \mathrm{e}^{\psi_k \cdot t}$ for $k = 1, 2, \dots, K$ (respectively $\tilde{K}$) are linearly dependent. But then, since all ϕ_k and ψ_k are each pairwise distinct, there must exist indices $1 \leq k_1 \leq K$ and $1 \leq k_2 \leq \tilde{K}$ such that $\phi_{k_1} = \psi_{k_2}$.

W.l.o.g. assume that $k_1 = k_2 = K$. We can then rewrite equation (3.9) and get

$$\eta_0 - \gamma_0 + \sum_{k=1}^{K-1} \eta_k \cdot \mathrm{e}^{\phi_k \cdot t} + (\eta_K - \gamma_K) \cdot \mathrm{e}^{\phi_K \cdot t} - \sum_{\substack{k=1 \\ k \neq K}}^{\tilde{K}} \gamma_k \cdot \mathrm{e}^{\psi_k \cdot t} \equiv 0$$

on T. However, this still implies that the functions $1, \mathrm{e}^{\phi_1 \cdot t}, \mathrm{e}^{\phi_2 \cdot t}, \ldots, \mathrm{e}^{\phi_K \cdot t}, \mathrm{e}^{\psi_1 \cdot t}, \ldots, \mathrm{e}^{\psi_{K-1} \cdot t}, \mathrm{e}^{\psi_{K+1} \cdot t}, \ldots, \mathrm{e}^{\psi_{\tilde{K}} \cdot t}$ are linearly dependent. Since ϕ_K is different from all ϕ_k and ψ_k with $k \neq K$, it follows that there must exist $1 \leq k_3 \leq K - 1$ and $1 \leq k_4 \leq \tilde{K}$ with $k_4 \neq K$ such that $\phi_{k_3} = \psi_{k_4}$. W.l.o.g. assume that $k_3 = k_4 = K - 1$ and rewrite the sum as before. We can continue this reduction process until we arrive at

$$(\eta_0 - \gamma_0) + \sum_{k=1}^{K} (\eta_k - \gamma_k) \cdot \mathrm{e}^{\phi_k \cdot t} - \sum_{k=K+1}^{\tilde{K}} \gamma_k \cdot \mathrm{e}^{\psi_k \cdot t} \equiv 0.$$

But the functions $1, \mathrm{e}^{\phi_1 \cdot t}, \mathrm{e}^{\phi_2 \cdot t}, \ldots, \mathrm{e}^{\phi_K \cdot t}$, $\mathrm{e}^{\psi_{K+1} \cdot t}, \ldots, \mathrm{e}^{\psi_{\tilde{K}} \cdot t}$ are linearly independent by construction. Hence, we must have $\eta_k - \gamma_k = 0$ for all $k = 0, 1, \ldots, K$ and $\gamma_k = 0$ for all $k = K + 1, K + 2, \ldots, \tilde{K}$, which entails that $\tilde{K} = K$, and the claim follows. $\square$

4. Reconstruction of Exponential Sums from Fourier Coefficients

4.1. Periodic Fourier Transform of Exponential Sums

Let $f \in C^1(\mathbb{R})$ be a P-periodic function. Then f can be expressed as a *Fourier series*, i.e.,

$$f(t) = \sum_{n=-\infty}^{\infty} c_n^P(f) \cdot \mathrm{e}^{2\pi\,\mathrm{i}\cdot\frac{n}{P}\cdot t}, \tag{4.1}$$

with *Fourier coefficients*

$$c_n^P(f) \coloneqq \hat{f}_P(n) \coloneqq \frac{1}{P}\int_0^P f(t)\cdot \mathrm{e}^{-2\pi\,\mathrm{i}\cdot\frac{n}{P}\cdot t}\ \mathrm{d}t, \tag{4.2}$$

where $\hat{f}_P$ denotes the *periodic Fourier transform* with period P, see [92, Remark 1.4]. The equality in (4.1) is understood in the sense that for any $t_0 \in \mathbb{R}$ the sequence of partial Fourier sums $\sum_{n=-k}^{k} c_n^P(f)\cdot \mathrm{e}^{2\pi\,\mathrm{i}\cdot\frac{n}{P}\cdot t_0}$ convergs to $f(t_0)$ as $k \to \infty$, i.e., the series converges pointwise. This follows from the Convergence Theorem of Dirichlet-Jordan [92, Theorem 1.34].

Now suppose that f is a P-periodic function with jump discontinuities at 0 and P and $f|_{(0,P)} \in C^1\big((0,P)\big)$, where $f|_{(0,P)}$ denotes the restriction of f to the interval $(0,P)$. Then the Fourier series corresponding to f still converges to f pointwise on each interval of the form $(m\cdot P, (m+1)\cdot P)$ for $m \in \mathbb{Z}$, which also follows from [92, Theroem 1.34]. We can use this fact to compute the Fourier series corresponding to an arbitrary function $f \in C^1(\mathbb{R})$ on an interval $(0,P)$. For this, define the *periodized function*

$$f_P(t) \coloneqq \sum_{m\in\mathbb{Z}} f(t - m\cdot P)\cdot \chi_{[m\cdot P,(m+1)\cdot P)}(t), \tag{4.3}$$

where

$$\chi_T(t) = \begin{cases} 1, & \text{if } t \in T \\ 0, & \text{if } t \notin T \end{cases} \tag{4.4}$$

is the indicator function on the set T. This is a P-periodic function which coincides with f on the interval $[0, P)$. Accordingly, the Fourier series representation of f_P coincides with f on the interval $(0, P)$.

In the following lemma, we look at the Fourier coefficients of exponential sums.

Lemma 4.1. *Let $f(t) = \eta \cdot \mathrm{e}^{\phi \cdot t}$ with $\eta \in \mathbb{C} \setminus \{0\}$ and $\phi \in \mathbb{C}$. Further, let $P > 0$. If $\phi = 2\pi\,\mathrm{i} \cdot \frac{m}{P}$ for some $m \in \mathbb{Z}$ then*

$$\hat{f}_P(n) = \begin{cases} \eta & \text{if } n = m, \\ 0 & \text{else} \end{cases}$$

for $n \in \mathbb{Z}$. Otherwise

$$\hat{f}_P(n) = \frac{1}{P} \cdot \frac{\eta}{2\pi\,\mathrm{i} \cdot \frac{n}{P} - \phi} \cdot \left(1 - \mathrm{e}^{\phi \cdot P}\right)$$

for $n \in \mathbb{Z}$.

Proof. For $n \in \mathbb{Z}$, we have

$$\begin{aligned} \hat{f}_P(n) &= \frac{1}{P} \int_0^P \eta \cdot \mathrm{e}^{\phi \cdot t} \cdot \mathrm{e}^{-2\pi\,\mathrm{i} \cdot \frac{n}{P} \cdot t} \,\mathrm{d}t \\ &= \frac{\eta}{P} \int_0^P \mathrm{e}^{\left(\phi - 2\pi\,\mathrm{i} \cdot \frac{n}{P}\right) \cdot t} \,\mathrm{d}t. \end{aligned}$$

If $\phi = 2\pi\,\mathrm{i} \cdot \frac{n}{P}$ we find then

$$\hat{f}_P(n) = \eta.$$

Otherwise

$$\begin{aligned} \hat{f}_P(n) &= \frac{\eta}{P} \cdot \left[\frac{1}{\phi - 2\pi\,\mathrm{i} \cdot \frac{n}{P}} \cdot \mathrm{e}^{\left(\phi - 2\pi\,\mathrm{i} \cdot \frac{n}{P}\right) \cdot t} \right]_0^P \\ &= \frac{\eta}{P} \cdot \frac{1}{\phi - 2\pi\,\mathrm{i} \cdot \frac{n}{P}} \cdot \left(\mathrm{e}^{\phi \cdot P} \cdot \underbrace{\mathrm{e}^{-2\pi\,\mathrm{i} \cdot n}}_{=1} - 1 \right) \end{aligned}$$

and it follows directly that $\hat{f}_P(n) = 0$ if $\phi = 2\pi\,\mathrm{i} \cdot \frac{m}{P}$ for $m \in \mathbb{Z}$ with $m \neq n$. □

Remark 4.2. Note that if $\phi \neq 2\pi\,\mathrm{i} \cdot \frac{m}{P}$ for any $m \in \mathbb{Z}$, the Fourier coefficients of f decay with a rate of $\mathcal{O}\left(\frac{1}{n}\right)$. This is due to the fact that we are actually computing the Fourier coefficients of the periodized function f_P as in (4.3), which is discontinuous, see [75, Section 2.3.1]. Further, we have in this case that $\hat{f}_P(n) \neq 0$ for all $n \in \mathbb{Z}$.

4.2. Reconstruction of Functions without P-Periodic Components

In this section we will consider exponential sums as in (3.6), i.e., with $\eta_k \in \mathbb{C} \setminus \{0\}$ and pairwise distinct $\phi_k \in \mathbb{C} \setminus \{0\}$, where we additionally assume that $\eta_0 = 0$ and $\phi_k \neq 2\pi\,\mathrm{i} \cdot \frac{n}{P}$ for any $n \in \mathbb{Z}$ and all $k = 1, 2, \dots, K$, i.e., none of the components are P-periodic. Then, by equation (4.2) and Lemma 4.1, the Fourier coefficients of

$$f(t) = \sum_{k=1}^{K} \eta_k \cdot \mathrm{e}^{\phi_k \cdot t}$$

are given by

$$\begin{aligned}\hat{f}_P(n) &= \frac{1}{P} \sum_{k=1}^{K} \frac{\eta_k}{2\pi\,\mathrm{i} \cdot \frac{n}{P} - \phi_k} \cdot \left(1 - \mathrm{e}^{\phi_k \cdot P}\right) \\ &= \frac{1}{P} \cdot \frac{\sum_{k=1}^{K} \eta_k \cdot \left(1 - \mathrm{e}^{\phi_k \cdot P}\right) \cdot \prod_{\substack{l=1 \\ l \neq k}}^{K} \left(2\pi\,\mathrm{i} \cdot \frac{n}{P} - \phi_l\right)}{\prod_{k=1}^{K} \left(2\pi\,\mathrm{i} \cdot \frac{n}{P} - \phi_k\right)}\end{aligned}$$

for $n \in \mathbb{Z}$. This means that we can write

$$\hat{f}_P(n) = \frac{1}{P} \cdot \frac{p\left(2\pi\,\mathrm{i} \cdot \frac{n}{P}\right)}{q\left(2\pi\,\mathrm{i} \cdot \frac{n}{P}\right)} \tag{4.5}$$

with polynomials

$$p(x) \coloneqq \sum_{k=1}^{K} \eta_k \cdot \left(1 - \mathrm{e}^{\phi_k \cdot P}\right) \cdot \prod_{\substack{l=1 \\ l \neq k}}^{K} (x - \phi_l) \tag{4.6}$$

and

$$q(x) \coloneqq \prod_{k=1}^{K} (x - \phi_k). \tag{4.7}$$

We have already published the following result in [91].

Lemma 4.3. *The polynomials p and q in equations* (4.6) *and* (4.7) *have the following properties:*

(i) $q\left(2\pi\,\mathrm{i} \cdot \frac{n}{P}\right) \neq 0$ *for any* $n \in \mathbb{Z}$.

(ii) q *is monic, i.e., its leading coefficient is* 1.

(iii) p and q are coprime.

(iv) $p \not\equiv 0$.

(v) $\deg(q) = K$ and $\deg(p) \leq K - 1$.

(vi) The set of parameters η_k and ϕ_k for $k = 1, 2, \ldots, K$ such that $\deg(p) < K - 1$ is a Lebesgue null set in $\mathbb{C}^{2K}$, where we identify $\mathbb{C}^{2K} \simeq \mathbb{R}^{4K}$. In other words, $\deg(p) = K - 1$ almost surely, which we will write $\deg(p) \overset{\text{a.s.}}{=} K - 1$.

Proof. (i) This follows from the fact that $\phi_k \neq 2\pi\,\mathrm{i} \cdot \frac{n}{P}$ for any $n \in \mathbb{Z}$ and all $k = 1, 2, \ldots, K$.

(ii) This is clear from the definition.

(iii) The polynomial q possesses exactly the zeros $\phi_1, \phi_2, \ldots, \phi_K$ and we have

$$p(\phi_k) = \eta_k \cdot \left(1 - \mathrm{e}^{\phi_k \cdot P}\right) \cdot \prod_{\substack{l=1 \\ l \neq k}}^{K} (\phi_k - \phi_l) \neq 0 \tag{4.8}$$

for $k = 1, 2, \ldots, K$ since $\mathrm{e}^{\phi_k \cdot P} \neq 1$ and all ϕ_k are pairwise distinct.

(iv) This can be seen immediately from equation (4.8). Another way to prove this. is as follows. Suppose that $p \equiv 0$. Since $q\left(2\pi\,\mathrm{i} \cdot \frac{n}{P}\right) \neq 0$, equation (4.5) implies that then $\hat{f}_P(n) = 0$ for all $n \in \mathbb{Z}$. But then it follows that $f \equiv 0$ (see e.g. [92, Theorem 1.3]).

(v) This is clear from the definition.

(vi) The leading coefficient of p is given by

$$p_{K-1} = \sum_{k=1}^{K} \eta_k \cdot \left(1 - \mathrm{e}^{\phi_k \cdot P}\right).$$

We now interpret this as a multivariate function in the parameters $\eta_k \in \mathbb{C} \setminus \{0\} \subset \mathbb{C}$ and $\phi_k \in \mathbb{C} \setminus \{0\} \subset \mathbb{C}$, i.e.,

$$p_{K-1} : \mathbb{C}^{2K} \to \mathbb{C}, \qquad p_{K-1}(\mathbf{z}) = \sum_{k=1}^{K} z_k \cdot \left(1 - \mathrm{e}^{z_{K+k} \cdot P}\right),$$

where $\mathbf{z} = (z_1, z_2, \ldots, z_{2K}) \in \mathbb{C}^{2K}$. Therefore, if the leading coefficient $p_{K-1} = 0$, then $(\eta_1, \ldots, \eta_K, \phi_1, \ldots, \phi_K)$ is a zero of $p_{K-1}(\mathbf{z})$.

By Osgood's Lemma, see [51, Theorem 2], the function p_{K-1} is holomorphic on $\mathbb{C}^{2K}$ since it is continuous on $\mathbb{C}^{2K}$ and holomorphic in each variable separately. Then, by

[51, Corollary 10], the set $\{\mathbf{z} \in \mathbb{C}^{2K} : p_{K-1}(\mathbf{z}) = 0\}$ has Lebesgue measure zero, where we identify $\mathbb{C}^{2K} \simeq \mathbb{R}^{4K}$. Therefore, it follows that the set of parameters η_k and ϕ_k for $k = 1, 2, \ldots, K$ such that $p_{K-1} = 0$, i.e., $\deg(p) < K - 1$, is a Lebesgue null set. $\square$

The fact conveyed in equation (4.5) can be interpreted as follows. The values $P \cdot \hat{f}_P(n)$ sample the rational function $\frac{p}{q}$ at the points $2\pi\,\mathrm{i} \cdot \frac{n}{P}$. Therefore, we can use Theorem 1.2 to conclude that p and q can be uniquely reconstructed from $2K$ Fourier coefficients via Algorithm 1.4. This observation is the core and basis of our new algorithm to reconstruct exponential sums.

However, in order to reconstruct the exponential sum f, we need to extract the parameters η_k and ϕ_k for $k = 1, 2, \ldots, K$ from p and q.

Theorem 4.4. *Given polynomials p and q as in equations* (4.6) *and* (4.7)*, we can reconstruct the parameters η_k and ϕ_k for $k = 1, 2, \ldots, K$ via*

$$q(\phi_k) = 0$$

and

$$\eta_k = p(\phi_k) \cdot \left(1 - \mathrm{e}^{\phi_k \cdot P}\right)^{-1} \cdot \prod_{\substack{l=1 \\ l \neq k}}^{K} (\phi_k - \phi_l)^{-1}.$$

Proof. It is obvious from equation (4.7) that every ϕ_k for $k = 1, 2, \ldots, K$ is a zero of q. Moreover, since $\phi_1, \phi_2, \ldots, \phi_K$ are pairwise distinct, q has only zeros with multiplicity 1. Hence, each zero corresponds exactly to one parameter ϕ_k.

Further, we have for $j = 1, 2, \ldots, K$ that

$$\begin{aligned} p(\phi_j) &= \sum_{k=1}^{K} \eta_k \cdot \left(1 - \mathrm{e}^{\phi_k \cdot P}\right) \cdot \prod_{\substack{l=1 \\ l \neq k}}^{K} (\phi_j - \phi_l) \\ &= \eta_j \cdot \left(1 - \mathrm{e}^{\phi_j \cdot P}\right) \cdot \prod_{\substack{l=1 \\ l \neq j}}^{K} (\phi_j - \phi_l). \end{aligned}$$

Since $\phi_j \neq \phi_l$ for $j \neq l$ and $\phi_j \neq 2\pi\,\mathrm{i} \cdot \frac{n}{P}$ for any $n \in \mathbb{Z}$, i.e., $\mathrm{e}^{\phi_j \cdot P} \neq 1$, the claim for η_k for $k = 1, 2, \ldots, K$ follows. $\square$

4.3. Reconstruction of Functions with P-Periodic Components

We will now consider exponential sums where we allow for $\eta_0 \neq 0$ and $\phi_k = 2\pi\,\mathrm{i}\cdot\frac{n}{P}$ for some $n \in \mathbb{Z}\setminus\{0\}$ and some $1 \leq k \leq K$, i.e.,

$$f(t) = \eta_0 + \sum_{k=1}^{K} \eta_k \cdot \mathrm{e}^{\phi_k \cdot t},$$

with $\eta_0 \in \mathbb{C}$ and $\eta_k, \phi_k \in \mathbb{C}\setminus\{0\}$. In order to increase readability of this section we will use a slightly different notation than above and say that

$$f(t) = \sum_{k=1}^{K} \eta_k \cdot \mathrm{e}^{\phi_k \cdot t}, \tag{4.9}$$

where $\phi_k \in \mathbb{C}$ for all $k = 1, 2, \dots, K$, i.e., we incorporate the the constant term into the sum so that we do not have to deal with it separately.

4.3.1. Functions Containing Periodic and Non-Periodic Components

First, we assume that the function contains both P-periodic and non-P-periodic components. Denote

$$L_{\mathrm{np}} \coloneqq \left\{k \in \{1, 2, \dots, K\} : \phi_k \neq 2\pi\,\mathrm{i}\cdot\tfrac{n}{P} \text{ for any } n \in \mathbb{Z}\right\}$$

and

$$L_{\mathrm{p}} \coloneqq \left\{k \in \{1, 2, \dots, K\} : \phi_k = 2\pi\,\mathrm{i}\cdot\tfrac{n}{P} \text{ for some } n \in \mathbb{Z}\right\}.$$

Then, by equation (4.2) and Lemma 4.1, the Fourier coefficients of such a function are given by

$$\hat{f}_P(n) = \frac{1}{P}\sum_{k\in L_{\mathrm{np}}} \frac{\eta_k}{2\pi\,\mathrm{i}\cdot\frac{n}{P} - \phi_k}\cdot\left(1 - \mathrm{e}^{\phi_k\cdot P}\right) + \sum_{k\in L_{\mathrm{p}}} \eta_k \cdot \chi_{\{\phi_k\}}(2\pi\,\mathrm{i}\cdot\tfrac{n}{P})$$

for $n \in \mathbb{Z}$ with χ as in (4.4). Put differently, we have

$$\hat{f}_P(n) = \frac{1}{P}\cdot\frac{p\left(2\pi\,\mathrm{i}\cdot\frac{n}{P}\right)}{q\left(2\pi\,\mathrm{i}\cdot\frac{n}{P}\right)} + \sum_{k\in L_{\mathrm{p}}} \eta_k \cdot \chi_{\{\phi_k\}}\left(2\pi\,\mathrm{i}\cdot\tfrac{n}{P}\right) \tag{4.10}$$

for $n \in \mathbb{Z}$, where

$$p(x) \coloneqq \sum_{k\in L_{\mathrm{np}}} \eta_k \cdot \left(1 - \mathrm{e}^{\phi_k \cdot P}\right) \cdot \prod_{\substack{l\in L_{\mathrm{np}}\\ l\neq k}} (x - \phi_l) \tag{4.11}$$

and

$$q(x) \coloneqq \prod_{k\in L_{\mathrm{np}}} (x - \phi_k) \tag{4.12}$$

are coprime polynomials with $\deg(p) \overset{\text{a.s.}}{=} |L_{\mathrm{np}}| - 1$ and $\deg(q) = |L_{\mathrm{np}}|$ as well as $q\left(2\pi\,\mathrm{i}\cdot\frac{n}{P}\right) \neq 0$ for all $n \in \mathbb{Z}$, by Lemma 4.3.

Comparing equation (4.10) with equation (1.6), we find that this is the same situation as considered in Section 1.2. More precisely, with

$$d(x) \coloneqq \prod_{k\in L_{\mathrm{p}}} (x - \phi_k) \tag{4.13}$$

we get

$$p\left(2\pi\,\mathrm{i}\cdot\tfrac{n}{P}\right) \cdot d\left(2\pi\,\mathrm{i}\cdot\tfrac{n}{P}\right) - P \cdot \hat{f}_P(n) \cdot q\left(2\pi\,\mathrm{i}\cdot\tfrac{n}{P}\right) \cdot d\left(2\pi\,\mathrm{i}\cdot\tfrac{n}{P}\right) = 0 \tag{4.14}$$

for $n \in \mathbb{Z}$, where $\deg(p \cdot d) \overset{\text{a.s.}}{=} |L_{\mathrm{np}}| + |L_{\mathrm{p}}| - 1 = K - 1$ and $\deg(q \cdot d) = |L_{\mathrm{np}}| + |L_{\mathrm{p}}| = K$. Therefore, we can reconstruct p, q, and d from $M \geq 2K$ suitable samples using Algorithm 1.4.

Given a set

$$S = \left\{\left(2\pi\,\mathrm{i}\cdot\tfrac{n_j}{P},\, P \cdot \hat{f}_P(n_j)\right) : n_j \in \mathbb{Z},\ j = 1, 2, \ldots, M\right\} \tag{4.15}$$

of $M \geq 2K$ sample points and values containing the set

$$D \coloneqq \left\{\left(2\pi\,\mathrm{i}\cdot\tfrac{n^*}{P},\, P \cdot \hat{f}_P(n^*)\right) : 2\pi\,\mathrm{i}\cdot\tfrac{n^*}{P} = \phi_k \text{ for some } k \in L_{\mathrm{p}} \text{ and } n^* \in \mathbb{Z}\right\}, \tag{4.16}$$

we see from equation (4.10) that the points in D are the unattainable points in the problem of finding a rational function of type $(K-1, K)$ that interpolates S. In other words, the P-periodic components of the function correspond precisely to the unattainable points of this rational interpolation problem.

In particular, if $k \in L_{\mathrm{p}}$ and $\phi_k = 2\pi\,\mathrm{i}\cdot\frac{n^*}{P}$ for some $n^* \in \mathbb{Z}$ then we have

$$\hat{f}_P(n^*) - \frac{1}{P} \cdot \frac{p\left(2\pi\,\mathrm{i}\cdot\frac{n^*}{P}\right)}{q\left(2\pi\,\mathrm{i}\cdot\frac{n^*}{P}\right)} = \eta_k. \tag{4.17}$$

From this, we can directly recover the parameter η_k for $k \in L_{\mathrm{p}}$. This means that the value by which the unattainable point in the rational interpolation problem is distorted corresponds to the coefficient of the P-periodic component. Further, the parameter ϕ_k is then simply given by $\phi_k = 2\pi\,\mathrm{i} \cdot \frac{n^*}{P}$.

4.3.2. Functions Containing only Periodic Components

There is one more issue we need to address. Until now, we have assumed that the function f contains at least one component which is not P-periodic. But what happens if f only consists of P-periodic components, i.e., f is just a trigonometric polynomial and the Fourier samples $\hat{f}_P(n)$ already provide the correct decomposition?

Suppose that we have $M \geq 2\tilde{K}$ samples of an exponential sum f as in (4.9), where $\tilde{K} \geq K$. Let $\mathbf{x}$ and $\mathbf{y}$ be vectors of sample points and values, i.e.,

$$\mathbf{x} = (x_1, x_2, \dots, x_M)^T = 2\pi\,\mathrm{i} \cdot \left(\tfrac{n_1}{P}, \tfrac{n_2}{P}, \dots, \tfrac{n_M}{P}\right)^T$$

and

$$\mathbf{y} = (y_1, y_2, \dots, y_M)^T = P \cdot \left(\hat{f}_P(n_1), \hat{f}_P(n_2), \dots, \hat{f}_P(n_M)\right)^T.$$

If f contains only P-periodic components, then, using Lemma 4.1, we get

$$y_j = \begin{cases} P \cdot \eta_k & \text{if } \phi_k = 2\pi\,\mathrm{i} \cdot \frac{n_j}{P} \text{ for some } k \in \{1, 2, \dots, K\}, \\ 0 & \text{else} \end{cases}$$

for all $j = 1, 2, \dots, M$. In particular, we have $\|\mathbf{y}\|_0 = K \leq \tilde{K}$. But then it follows from Lemma 1.1 that every vector in the kernel of $\mathbf{V}_{\tilde{K}-1,\tilde{K}}(\mathbf{x}, \mathbf{y})$ is of the form $(\mathbf{0}^T, -\mathbf{q}^T)^T$.

If, on the other hand, f contains at least one non-P-periodic component, then $y_j = 0$ for at most $|L_{\mathrm{np}}| - 1 + |L_{\mathrm{p}}| = K - 1$ indices $j \in \{1, 2, \dots, M\}$. This can be seen from equation (4.10) as follows. We have for $n \in \mathbb{N}$ that

$$\frac{1}{P} \cdot \frac{p\left(2\pi\,\mathrm{i} \cdot \frac{n}{P}\right)}{q\left(2\pi\,\mathrm{i} \cdot \frac{n}{P}\right)} + \sum_{k \in L_{\mathrm{p}}} \eta_k \cdot \chi_{\{\phi_k\}}\left(2\pi\,\mathrm{i} \cdot \tfrac{n}{P}\right) = 0$$

if either

$$\frac{p\left(2\pi\,\mathrm{i} \cdot \frac{n}{P}\right)}{q\left(2\pi\,\mathrm{i} \cdot \frac{n}{P}\right)} = 0 \quad \text{and} \quad \phi_k \neq 2\pi\,\mathrm{i} \cdot \frac{n}{P}$$

for all $k = 1, 2, \dots, K$ or

$$\frac{p\left(2\pi\,\mathrm{i} \cdot \frac{n}{P}\right)}{q\left(2\pi\,\mathrm{i} \cdot \frac{n}{P}\right)} = -\eta_k$$

for some $1 \leq k \leq K$ such that $\phi_k = 2\pi\,\mathrm{i} \cdot \frac{n}{P}$. Since $\deg(p) \leq |L_{\mathrm{np}}| - 1$ and $q\left(2\pi\,\mathrm{i} \cdot \frac{n}{P}\right) \neq 0$, the first case happens at most $|L_{\mathrm{np}}| - 1 = K - |L_{\mathrm{p}}| - 1$ times. The second case occurs at most $|L_{\mathrm{p}}|$ times. So in total we have $\hat{f}_P(n) = 0$ at most $K - |L_{\mathrm{p}}| - 1 + |L_{\mathrm{p}}| = K - 1$ times.

Therefore, if f contains at least one P-periodic component, $\|\mathbf{y}\|_0 \geq M - K + 1 \geq K + 1$ and hence every vector in the kernel of $\mathbf{V}_{\tilde{K}-1,\tilde{K}}(\mathbf{x}, \mathbf{y})$ is of the form $(\mathbf{p}^T, -\mathbf{q}^T)^T$ where $\mathbf{p} \neq \mathbf{0}$ and $\mathbf{q} \neq \mathbf{0}$. It follows that if any element of $\ker \mathbf{V}_{\tilde{K}-1,\tilde{K}}(\mathbf{x}, \mathbf{y})$ is of the form $(\mathbf{0}^T, -\mathbf{q}^T)^T$, which, by Lemma 1.1, is equivalent to all elements in the kernel being of this form, then we know that $\phi_k = 2\pi\,\mathrm{i} \cdot \frac{n_j}{P}$ for all $k = 1, 2, \ldots, K$ and certain $j \in \{1, 2, \ldots, M\}$. In this case we can directly recover ϕ_k and η_k from the sample values $\hat{f}_P(n_j)$.

However, the above argument gives us an even simpler criterion to determine whether the given exponential sum contains only P-periodic components. Since, if the function contains non-P-periodic components we have $\|\mathbf{y}\|_0 \geq M - K + 1 \geq \left\lceil \frac{M}{2} \right\rceil + 1$, it follows immediately that if $\|\mathbf{y}\|_0 \leq \left\lceil \frac{M}{2} \right\rceil$, it can only contain P-periodic components.

4.4. Main Theorem and Algorithm for Classical Fourier Coefficients

Summing up our findings so far gives us one of the main results of this thesis.

Main Theorem 4.5 (Reconstruction of Exponential Sums from Fourier Coefficients)**.** *Given an exponential sum $f(t) = \sum_{k=1}^{K} \eta_k \cdot \mathrm{e}^{\phi_k \cdot t}$ with $\eta_k \in \mathbb{C} \setminus \{0\}$ and pairwise distinct $\phi_k \in \mathbb{C}$ for all $k = 1, 2, \ldots, K$, the parameters η_k and ϕ_k can be uniquely reconstructed from $2K$ Fourier coefficients $\hat{f}_P(n)$ with arbitrary $P > 0$ and suitable $n \in \mathbb{Z}$.*

Parameter K?

Proof. From equation (4.14) and Lemma 4.3(iii), (v), it follows with Theorem 1.2 that the polynomials p, q, and d as in (4.11), (4.12), and (4.13) can be reconstructed from $2K$ samples using Algorithm 1.4. Since q and d are monic and P is known, we can uniquely reconstruct all polynomials. Using Theorem 4.4 and equation (4.17), we can reconstruct η_k and ϕ_k for $k = 1, 2, \ldots, K$. By Theorem 3.11, these parameters uniquely determine the exponential sum f. □

An immediate consequence of this theorem is the following.

Corollary 4.6. *Let f and g be exponential sums consisting of at most K components. Given samples $\left(2\pi\mathrm{i}\cdot\frac{n_j}{P},\, P\cdot\hat{f}_P(n_j)\right)$ and $\left(2\pi\mathrm{i}\cdot\frac{n_j}{P},\, P\cdot\hat{g}_P(n_j)\right)$ for $n_j\in\mathbb{Z}$ and $j=1,2,\dots,M$ with $M\geq 2K$ containing all Fourier coefficients corresponding to P-periodic components of f and g, if $\hat{f}_P(n_j)=\hat{g}_P(n_j)$ for all $j=1,2,\dots,M$, then $f=g$, i.e., the two functions are defined by the same parameters.*

We can now formulate a procedure to reconstruct exponential sums from Fourier samples.

Algorithm 4.7 (Reconstruction of Exponential Sums from Fourier Coefficients)**.**
Input: $S=\left\{\left(2\pi\mathrm{i}\cdot\frac{n_j}{P},\, P\cdot\hat{f}_P(n_j)\right) : n_j\in\mathbb{Z},\ j=1,2,\dots,M\right\}$. A set of M Fourier samples from an exponential sum f with K components.

(i) IF $\left\|\left(\hat{f}_P(n_j)\right)_{j=1}^{M}\right\|_0 \leq \left\lceil\frac{M}{2}\right\rceil$, then STOP. The function f contains only P-periodic components. Reconstruct the parameters η_k and ϕ_k for $k=1,2,\dots,K$ directly from $\hat{f}_P(n_j)$.

(ii) Set $n=\left\lfloor\frac{M-1}{2}\right\rfloor$ and $m=n-1$ and construct the matrix $\mathbf{V}_{m,n}(S)$ as in (1.7).

(iii) Find $(\mathbf{p}^T,-\mathbf{q}^T)^T\in\ker\mathbf{V}_{m,n}(S)$.

(iv) IF $\operatorname{null}\mathbf{V}_{m,n}(S)=0$, then STOP. There are too few samples.

(v) ELSE set p and q the polynomials corresponding to $\mathbf{p}$ and $\mathbf{q}$. Compute $d=\gcd(p,q)$ and set $p^*=c\cdot\frac{p}{d}$ as well as $q^*=c\cdot\frac{q}{d}$, where $c\in\mathbb{R}$ is chosen such that q^* is monic.

(vi) Find the roots ϕ_k of q^* and compute

$$\eta_k = p^*(\phi_k)\cdot\left(1-\mathrm{e}^{\phi_k\cdot P}\right)^{-1}\cdot\prod_{\substack{l=1\\ l\neq k}}^{\deg(q^*)}(\phi_k-\phi_l)^{-1}$$

for $k=1,2,\dots,\deg(q^*)$.

(vii) Compute

$$\tilde{y}_j = P\cdot\hat{f}_P(n_j)-\frac{p^*\left(2\pi\mathrm{i}\cdot\frac{n_j}{P}\right)}{q^*\left(2\pi\mathrm{i}\cdot\frac{n_j}{P}\right)}$$

for all $j=1,2,\dots,M$. IF $\tilde{y}_j\neq 0$, set $\phi_k=2\pi\mathrm{i}\cdot\frac{n_j}{P}$ and $\eta_k=\frac{\tilde{y}_j}{P}$ for $k=\deg(q^*)+1,\deg(q^*)+2,\dots,K$.

Output: Parameters K, η_k, ϕ_k for $k=1,2,\dots,K$ and possibly η_0 defining f.

Remark 4.8. (i) Note that step (iv) in the above algorithm is justified by Lemma 1.5, respectively Remark 1.6 (i). Further, as already noted in Remark 1.6 (iv), it is possible that for $M < 2K$ we have $\operatorname{null} \mathbf{V}_{m,n}(S) \geq 1$.

(ii) If f contains P-periodic components, we can only reconstruct them if we have the correct samples, i.e., if $\phi_k = 2\pi\,\mathrm{i} \cdot \frac{n}{P}$ for some $n \in \mathbb{Z}$ and some $1 \leq k \leq K$, then we need to have the sample $\Big(2\pi\,\mathrm{i} \cdot \frac{n}{P},\ P \cdot \hat{f}_P(n)\Big)$.

(iii) If, on the other hand, f does not contain any P-periodic components, then it does not matter which samples we take, as long as we have enough of them.

5. Reconstruction of Real Almost-Periodic Functions from Fourier Coefficients

5.1. Periodic Fourier Transform of Real Almost-Periodic Functions

As we have seen in equation (3.8), a real almost-periodic function consisting of K components can always be written as a (complex) almost-periodic function consisting of $2K$ components. Therefore, if we want to reconstruct a real almost-periodic function, we need twice as many samples as for a complex one, i.e., $4K$. However, the Fourier coefficients of real almost-periodic functions have a special structure, which we can use to develop a modification of our Algorithm 4.7 for real almost-periodic functions. This modified algorithm will only need $2K$ samples in order to reconstruct a real almost-periodic function with K components. We have published the results from this chapter in [90]. The following is the analogue of Lemma 4.1.

Lemma 5.1. *Let $f(t) = \rho \cdot \cos(2\pi \cdot a \cdot t + b)$ with $\rho, a \in \mathbb{R}_{>0}$ and $b \in [0, 2\pi)$ or $(\rho, a, b) \in \mathbb{R} \times \{(0,0)\}$, i.e., $f(t) \equiv \rho$. Further, let $P > 0$. If $a = \frac{m}{P}$ for some $m \in \mathbb{N}_0$, then*

$$\hat{f}_P(n) = \begin{cases} \rho, & \text{if } n = m = 0, \\ \frac{\rho}{2} \cdot \big(\cos(b) \pm \mathrm{i} \cdot \sin(b)\big), & \text{if } n = \pm m \neq 0, \\ 0, & \text{else} \end{cases}$$

for $n \in \mathbb{Z}$. Otherwise

$$\hat{f}_P(n) = \frac{1}{\pi \cdot P} \cdot \frac{\rho \cdot \sin(\pi \cdot a \cdot P)}{a^2 - \frac{n^2}{P^2}} \cdot \big(a \cdot \cos(\pi \cdot a \cdot P + b) + \mathrm{i} \cdot \tfrac{n}{P} \cdot \sin(\pi \cdot a \cdot P + b)\big)$$

for $n \in \mathbb{Z}$.

Proof. We will use that

$$\begin{aligned}\hat{f}_P(n) &= \frac{1}{P}\int_0^P f(t)\cdot \mathrm{e}^{-2\pi\,\mathrm{i}\cdot\frac{n}{P}\cdot t}\ \mathrm{d}t\\ &= \frac{1}{P}\int_0^P f(t)\cdot\cos\left(2\pi\cdot\tfrac{n}{P}\cdot t\right)\mathrm{d}t - \mathrm{i}\,\frac{1}{P}\int_0^P f(t)\cdot\sin\left(2\pi\cdot\tfrac{n}{P}\cdot t\right)\mathrm{d}t\end{aligned}$$

and compute the real and imaginary part of $\hat{f}_P(n)$ separately.

First, we compute $\operatorname{Re}\hat{f}_P(n)$. Suppose that $a = \frac{m}{P}$ for some $m \in \mathbb{N}_0$. If $m = n = 0$ we have

$$\operatorname{Re}\hat{f}_P(n) = \frac{1}{P}\int_0^P \rho\cdot\cos(b)\cdot 1\mathrm{d}t = \rho\cdot\cos(b) = \rho\cdot 1$$

since we assumed that if $a = 0$ then also $b = 0$.

Using the trigonometric identity

$$\cos(x)\cdot\cos(y) = \frac{1}{2}\cdot\big(\cos(x+y)+\cos(x-y)\big)$$

we find

$$\begin{aligned}\operatorname{Re}\hat{f}_P(n) &= \frac{1}{P}\int_0^P \rho\cdot\cos(2\pi\cdot a\cdot t+b)\cdot\cos\left(2\pi\cdot\tfrac{n}{P}\cdot t\right)\mathrm{d}t\\ &= \frac{1}{P}\cdot\frac{\rho}{2}\int_0^P \cos\left(2\pi\cdot\left(a+\tfrac{n}{P}\right)\cdot t+b\right)+\cos\left(2\pi\cdot\left(a-\tfrac{n}{P}\right)\cdot t+b\right)\mathrm{d}t.\end{aligned}$$

For $m = \pm n \neq 0$, i.e., $a = \pm\frac{n}{P} \neq 0$, we get

$$\begin{aligned}\operatorname{Re}\hat{f}_P(n) &= \frac{1}{P}\cdot\frac{\rho}{2}\int_0^P \cos\left(\pm 4\pi\cdot\tfrac{n}{P}\cdot t+b\right)+\cos(b)\mathrm{d}t\\ &= \frac{1}{P}\cdot\frac{\rho}{2}\cdot\left(\left[\pm\tfrac{1}{4\pi\cdot\frac{n}{P}}\cdot\sin\left(\pm 4\pi\cdot\tfrac{n}{P}\cdot t+b\right)\right]_0^P + P\cdot\cos(b)\right)\\ &= \frac{\rho}{2}\cdot\cos(b).\end{aligned}$$

Now suppose that $a \neq \pm\frac{n}{P}$. Then

$$\begin{aligned}\operatorname{Re}\hat{f}_P(n) = \frac{1}{P}\cdot\frac{\rho}{2}\cdot\Bigg(&\left[\tfrac{1}{2\pi\cdot\left(a+\frac{n}{P}\right)}\cdot\sin\left(2\pi\cdot\left(a+\tfrac{n}{P}\right)\cdot t+b\right)\right]_0^P\\ &+\left[\tfrac{1}{2\pi\cdot\left(a-\frac{n}{P}\right)}\cdot\sin\left(2\pi\cdot\left(a-\tfrac{n}{P}\right)\cdot t+b\right)\right]_0^P\Bigg)\\ = \frac{1}{P}\cdot\frac{\rho}{2}\cdot\Bigg(&\tfrac{1}{2\pi\cdot\left(a+\frac{n}{P}\right)}\cdot\big(\sin(2\pi\cdot a\cdot P+b)-\sin(b)\big)\end{aligned}$$

$$
\begin{aligned}
&\quad + \tfrac{1}{2\pi\cdot\left(a-\frac{n}{P}\right)} \cdot \big(\sin(2\pi\cdot a\cdot P+b)-\sin(b)\big)\Big) \\
&= \tfrac{\rho\cdot a}{2\pi\cdot P\cdot\left(a^2-\frac{n^2}{P^2}\right)} \cdot \big(\sin(2\pi\cdot a\cdot P+b)-\sin(b)\big).
\end{aligned}
$$

If $a=\frac{m}{P}$ for some $m\in\mathbb{N}_0$, $m\neq\pm n$, then it follows immediately that $\operatorname{Re}\hat{f}_P(n)=0$. Otherwise, using the trigonometric identity

$$\sin(x+y)-\sin(x-y)=2\cdot\cos(x)\cdot\sin(y) \tag{5.1}$$

we find

$$\sin(2\pi\cdot a\cdot P+b)-\sin(b)=2\cdot\sin(\pi\cdot a\cdot P)\cdot\cos(\pi\cdot a\cdot P+b),$$

which leads to

$$\operatorname{Re}\hat{f}_P(n)=\frac{1}{\pi\cdot P}\cdot\frac{\rho\cdot\sin(\pi\cdot a\cdot P)}{a^2-\frac{n^2}{P^2}}\cdot a\cdot\cos(\pi\cdot a\cdot P+b).$$

Similarly, we calculate $\operatorname{Im}\hat{f}_P(n)$. Using the trigonometric identity (5.1) again, we find

$$
\begin{aligned}
\operatorname{Im}\hat{f}_P(n) &= -\frac{1}{P}\int_0^P \rho\cdot\cos(2\pi\cdot a\cdot t+b)\cdot\sin\left(2\pi\cdot\tfrac{n}{P}\cdot t\right)\mathrm{d}t \\
&= -\frac{1}{P}\cdot\frac{\rho}{2}\int_0^P \sin\left(2\pi\cdot\left(a+\tfrac{n}{P}\right)\cdot t+b\right)-\sin\left(2\pi\cdot\left(a-\tfrac{n}{P}\right)\cdot t+b\right)\mathrm{d}t.
\end{aligned}
$$

For $n=0$, we immediately see that $\operatorname{Im}\hat{f}_P(n)=0$. For $a=\pm\frac{n}{P}\neq 0$, we get

$$\operatorname{Im}\hat{f}_P(n)=-\frac{1}{P}\cdot\frac{\rho}{2}\int_0^P \pm\sin\left(\pm4\pi\cdot\tfrac{n}{P}\cdot t+b\right)\mp\sin(b)\mathrm{d}t=\pm\frac{\rho}{2}\cdot\sin(b).$$

For $a\neq\pm\frac{n}{P}$, we have

$$
\begin{aligned}
\operatorname{Im}\hat{f}_P(n) &= -\frac{1}{P}\cdot\frac{\rho}{2}\cdot\left(\left[\tfrac{-1}{2\pi\cdot\left(a+\frac{n}{P}\right)}\cdot\cos\left(2\pi\cdot\left(a+\tfrac{n}{P}\right)\cdot t+b\right)\right]_0^P\right. \\
&\quad \left. - \left[\tfrac{-1}{2\pi\cdot\left(a-\frac{n}{P}\right)}\cdot\cos\left(2\pi\cdot\left(a-\tfrac{n}{P}\right)\cdot t+b\right)\right]_0^P\right) \\
&= -\frac{1}{P}\cdot\frac{\rho}{2}\cdot\Big(\tfrac{-1}{2\pi\cdot\left(a+\frac{n}{P}\right)}\cdot\big(\cos(2\pi\cdot a\cdot P+b)-\cos(b)\big) \\
&\quad + \tfrac{1}{2\pi\cdot\left(a-\frac{n}{P}\right)}\cdot\big(\cos(2\pi\cdot a\cdot P+b)-\cos(b)\big)\Big)
\end{aligned}
$$

$$= -\frac{\rho\cdot\frac{n}{P}}{2\pi\cdot P\cdot\left(a^2-\frac{n^2}{P^2}\right)}\cdot\big(\cos(2\pi\cdot a\cdot P+b)-\cos(b)\big).$$

If $a = \frac{m}{P}$ for some $m \in \mathbb{N}_0$, $m \neq \pm n$, then it follows that $\operatorname{Im}\hat{f}_P(n) = 0$. Otherwise, using the trigonometric identity

$$\cos(x+y)-\cos(x-y) = -2\cdot\sin(x)\cdot\sin(y)$$

gives us

$$\cos(2\pi\cdot a\cdot P+b)-\cos(b) = -2\cdot\sin(\pi\cdot a\cdot P)\cdot\sin(\pi\cdot a\cdot P+b),$$

i.e.,

$$\operatorname{Im}\hat{f}_P(n) = \frac{1}{\pi\cdot P}\cdot\frac{\rho\cdot\sin(\pi\cdot a\cdot P)}{a^2-\frac{n^2}{P^2}}\cdot\tfrac{n}{P}\cdot\sin(\pi\cdot a\cdot P+b).$$

□

Remark 5.2. Note that for $n \in \mathbb{N}_0$ we have $\hat{f}_P(-n) = \overline{\hat{f}_P(n)}$, which comes from the fact that f is real valued, see [92, p.14].

5.2. Reconstruction of Functions without P-Periodic Components

We again start by considering functions without P-periodic components, i.e.,

$$f(t) = \sum_{k=1}^{K}\rho_k\cdot\cos(2\pi\cdot a_k\cdot t+b_k),$$

where $\rho_k, a_k \in \mathbb{R}_{>0}$ and $b_k \in [0,2\pi)$ with $a_k \neq \frac{n}{P}$ for any $n \in \mathbb{N}_0$ and all $k = 1,2,\ldots,K$. Then we find, similar as in Section 4.2, that

$$\hat{f}_P(n) = \frac{-1}{\pi\cdot P}\sum_{k=1}^{K}\frac{\rho_k\cdot\sin(\pi\cdot a_k\cdot P)}{\frac{n^2}{P^2}-a_k^2}\cdot\Big(a_k\cdot\cos(\pi\cdot a_k\cdot P+b_k) + \mathrm{i}\cdot\tfrac{n}{P}\cdot\sin(\pi\cdot a_k\cdot P+b_k)\Big)$$

for $n \in \mathbb{Z}$. This can again be written as

$$\hat{f}_P(n) = \frac{-1}{\pi\cdot P}\cdot\frac{p\big(\frac{n}{P}\big)}{q\big(\frac{n}{P}\big)},$$

with

$$p(x) := \sum_{k=1}^{K} \rho_k \cdot \sin(\pi \cdot a_k \cdot P) \cdot \Big(a_k \cdot \cos(\pi \cdot a_k \cdot P + b_k) \\ + \mathrm{i} \cdot x \cdot \sin(\pi \cdot a_k \cdot P + b_k)\Big) \cdot \prod_{\substack{l=1 \\ l \neq k}}^{K} (x^2 - a_l^2)$$

and

$$q(x) := \prod_{k=1}^{K} (x^2 - a_k^2),$$

where p and q are coprime polynomials and q is monic with $q\big(\frac{n}{P}\big) \neq 0$ for all $n \in \mathbb{Z}$. However, other than in Section 4.2, now $\deg(p) \leq 2K - 1$ and $\deg(q) = 2K$. Hence, by Theorem 1.2, we need at least $4K$ samples in order to reconstruct p and q.

Remark 5.3. The fact that we need twice as many samples in the real case than in the complex case can be explained heuristically by observing that a real almost-periodic function contains, in some sense, only half as much information as a complex one. After all, the real function is just the real part of a complex function, i.e., the complex function also contains the information of the imaginary part.

Through a simple trick though, we will be able to reduce the number of required samples to only $2K$. For $n \in \mathbb{Z} \setminus \{0\}$, we define the *modified Fourier coefficients* by

$$\begin{aligned} \tilde{c}_n^P &:= \operatorname{Re} \hat{f}_P(n) + \mathrm{i}\, \tfrac{P}{n} \cdot \operatorname{Im} \hat{f}_P(n) \\ &= \frac{-1}{\pi \cdot P} \sum_{k=1}^{K} \frac{\rho_k \cdot \sin(\pi \cdot a_k \cdot P)}{\frac{n^2}{P^2} - a_k^2} \cdot \Big(a_k \cdot \cos(\pi \cdot a_k \cdot P + b_k) + \mathrm{i} \cdot \sin(\pi \cdot a_k \cdot P + b_k)\Big). \end{aligned}$$

Since $\tilde{c}_n^P = \tilde{c}_{-n}^P$, we will from now on only consider $n \in \mathbb{N}$.

In order to simplify notation we will write

$$\begin{aligned} C_k &:= \rho_k \cdot \sin(\pi \cdot a_k \cdot P) \cdot \cos(\pi \cdot a_k \cdot P + b_k) \quad \text{and} \\ S_k &:= \rho_k \cdot \sin(\pi \cdot a_k \cdot P) \cdot \sin(\pi \cdot a_k \cdot P + b_k) \end{aligned} \tag{5.2}$$

for $k = 1, 2, \ldots, K$. Defining the polynomials

$$\tilde{p}(x) := \sum_{k=1}^{K} (a_k \cdot C_k + \mathrm{i}\, S_k) \cdot \prod_{\substack{l=1 \\ l \neq k}}^{K} (x - a_l^2) \tag{5.3}$$

and

$$\tilde{q}(x) \coloneqq \prod_{k=1}^{K} (x - a_k^2), \tag{5.4}$$

we then find that

$$\tilde{c}_n^P = \frac{-1}{\pi \cdot P} \cdot \frac{\tilde{p}\left(\frac{n^2}{P^2}\right)}{\tilde{q}\left(\frac{n^2}{P^2}\right)}.$$

Moreover, $\tilde{p}$ and $\tilde{q}$ possess the same properties as p and q above, i.e., they are coprime and $\tilde{q}$ is monic with $\tilde{q}\left(\frac{n^2}{P}\right) \neq 0$. But now we have $\deg(\tilde{p}) \leq K - 1$ and $\deg(\tilde{q}) = K$. This means that we can reconstruct $\tilde{p}$ and $\tilde{q}$ from just $2K$ modified samples of the form $\left(\frac{n^2}{P^2},\, -\pi \cdot P \cdot \tilde{c}_n^P\right)$ for $n \in \mathbb{N}$. However, it is important to note that these samples must not be symmetric with respect to 0 since $\left(\frac{(-n)^2}{P^2},\, -\pi \cdot P \cdot \tilde{c}_{-n}^P\right) = \left(\frac{n^2}{P^2},\, -\pi \cdot P \cdot \tilde{c}_n^P\right)$, i.e., if we were to sample symmetrically around 0, we would just get two times the same samples.

Remark 5.4. Recall that for a real signal f we have $\hat{f}_P(-n) = \overline{\hat{f}_P(n)}$ for $n \in \mathbb{N}_0$, see Remark 5.2. We can use this fact to heuristically explain why $2K$ samples suffice to reconstruct the polynomials corresponding to Fourier coefficients of a real almost-periodic signal. Since we sample asymmetrically, i.e., we have $\hat{f}_P(n)$ for certain $n \in \mathbb{N}$, we automatically also know $\hat{f}_P(-n) = \overline{\hat{f}_P(n)}$. This means that although we only took $2K$ samples, using this symmetry property, we actually know $4K$ samples.

As in Section 4.2, we find that $\tilde{p} \not\equiv 0$, where the proof is exactly the same as in Lemma 4.3 (iv). Furthermore, using the same argument as in Lemma 4.3 (vi), we find that the set of parameters $\rho_k, a_k \in \mathbb{R}_{>0}$ and $b_k \in [0, 2\pi)$ for $k = 1, 2, \ldots, K$ such that the leading coefficient of $\tilde{p}$ satisfies

$$\tilde{p}_{K-1} = \sum_{k=1}^{K} \left(a_k \cdot C_k + \mathrm{i}\, S_k\right) = 0$$

is a Lebesgue null set, i.e., $\deg(\tilde{p}) \overset{\text{a.s.}}{=} K - 1$.

The next theorem is the analogous result to Theorem 4.4. However, before we state it, we need to introduce the so-called *two-argument* or *four-quadrant arctangent* function. This function probably originated from the FORTRAN programming language, see [83]. While for numbers $x, y \in \mathbb{R}$ such that $(x, y) \neq (0, 0)$ the function $\arctan\left(\frac{x}{y}\right)$ returns a value in the interval $\left(-\frac{\pi}{2}, \frac{\pi}{2}\right]$, the two-argument arctangent will take into account the signs of x and y and return a value in the range $(-\pi, \pi]$. For $(x, y) \neq (0, 0)$, it is defined in terms of the arctangent as

$$\operatorname{atan2}(x,y) \coloneqq \begin{cases} \arctan\left(\frac{x}{y}\right) & \text{if } y > 0, \\ \arctan\left(\frac{x}{y}\right) + \pi & \text{if } y < 0 \text{ and } x \geq 0, \\ \arctan\left(\frac{x}{y}\right) - \pi & \text{if } y < 0 \text{ and } x < 0, \\ +\frac{\pi}{2} & \text{if } y = 0 \text{ and } x > 0, \\ -\frac{\pi}{2} & \text{if } y = 0 \text{ and } x < 0, \end{cases} \tag{5.5}$$

see also [92, p.12].

Theorem 5.5. *Given polynomials $\tilde{p}$ and $\tilde{q}$ as in equations* (5.3) *and* (5.4)*, we can reconstruct the parameters ρ_k, a_k, and b_k for $k = 1, 2, \ldots, K$ via*

$$\begin{aligned} \tilde{q}(a_k^2) &= 0, \\ C_k &= \frac{\operatorname{Re} \tilde{p}(a_k^2)}{a_k \cdot \prod_{\substack{l=1 \\ l \neq k}}^{K} (a_k^2 - a_l^2)}, \\ S_k &= \frac{\operatorname{Im} \tilde{p}(a_k^2)}{\prod_{\substack{l=1 \\ l \neq k}}^{K} (a_k^2 - a_l^2)}, \\ \rho_k &= \frac{\sqrt{C_k^2 + S_k^2}}{|\sin(\pi \cdot a_k \cdot P)|}, \end{aligned}$$

and

$$b_k = \operatorname{atan2}\left(\frac{S_k}{\rho_k \cdot \sin(\pi \cdot a_k \cdot P)}, \frac{C_k}{\rho_k \cdot \sin(\pi \cdot a_k \cdot P)}\right) - \pi \cdot a_k \cdot P \mod 2\pi.$$

Proof. First, we can recover a_k^2 for $k = 1, 2, \ldots, K$ by computing the zeros r_k of $\tilde{q}$. Since $a_k > 0$ for all $k = 1, 2, \ldots, K$, this immediately gives us $a_k = \sqrt{r_k}$. On account of all a_k being pairwise distinct, we recover these parameters uniquely.

Once we know a_k, we can compute C_k and S_k as in (5.2) for $k = 1, 2, \ldots, K$ via

$$\begin{aligned} & \tilde{p}(a_k^2) = (a_k \cdot C_k + \mathrm{i}\, S_k) \cdot \prod_{\substack{l=1 \\ l \neq k}}^{K} (a_k^2 - a_l^2) \\ \Leftrightarrow \quad & \frac{\tilde{p}(a_k^2)}{\prod_{\substack{l=1 \\ l \neq k}}^{K} (a_k^2 - a_l^2)} = a_k \cdot C_k + \mathrm{i}\, S_k. \end{aligned}$$

This means that

$$C_k = \frac{\operatorname{Re} \tilde{p}(a_k^2)}{a_k \cdot \prod_{\substack{l=1 \\ l\neq k}}^{K} (a_k^2 - a_l^2)}$$

and

$$S_k = \frac{\operatorname{Im} \tilde{p}(a_k^2)}{\prod_{\substack{l=1 \\ l\neq k}}^{K} (a_k^2 - a_l^2)}$$

for $k = 1, 2, \dots, K$. Writing out C_k and S_k as in (5.2) we find that

$$C_k^2 + S_k^2 = \rho_k^2 \cdot \sin^2(\pi \cdot a_k \cdot P) \cdot \big(\underbrace{\cos^2(\pi \cdot a_k \cdot P + b_k) + \sin^2(\pi \cdot a_k \cdot P + b_k)}_{=1}\big).$$

Recall that $a_k \neq \frac{n}{P}$ for any $n \in \mathbb{N}$ and hence $\sin(\pi \cdot a_k \cdot P) \neq 0$. Since a_k as well as P are known and $\rho_k > 0$, we then get

$$\rho_k = \frac{\sqrt{C_k^2 + S_k^2}}{|\sin(\pi \cdot a_k \cdot P)|}.$$

Further, for $k = 1, 2, \dots, K$ we have

$$\frac{S_k}{C_k} = \frac{\sin(\pi \cdot a_k \cdot P + b_k)}{\cos(\pi \cdot a_k \cdot P + b_k)} = \tan(\pi \cdot a_k \cdot P + b_k).$$

Hence, we find

$$b_k = \left(\operatorname{atan2}\left(\frac{S_k}{\rho_k \cdot \sin(\pi \cdot a_k \cdot P)}, \frac{C_k}{\rho_k \cdot \sin(\pi \cdot a_k \cdot P)}\right) - \pi \cdot a_k \cdot P\right) \bmod 2\pi.$$

Note that we have to use $\frac{S_k}{\rho_k \cdot \sin(\pi \cdot a_k \cdot P)}$ and $\frac{C_k}{\rho_k \cdot \sin(\pi \cdot a_k \cdot P)}$ instead of just S_k and C_k since atan2 uses the signs of the arguments to determine a unique solution.

These results hold true even in the case that either $C_k = 0$ or $S_k = 0$ for all $k = 1, 2, \dots, K$. □

5.3. Reconstruction of Functions with P-Periodic Components

Now let us consider functions containing P-periodic components. First, note that the case that the function only contains P-periodic components is completely analogous to the complex case discussed in Section 4.3.2. Therefore, we do not need to consider it separately and only look at the case that the function contains both P-periodic and non-P-periodic components.

Similarly as in Section 4.3.1, let

$$L_{\mathrm{np}} \coloneqq \big\{k \in \{1, 2, \dots, K\} : a_k \neq \tfrac{n}{P} \text{ for any } n \in \mathbb{N}\big\}$$

and

$$L_{\mathrm{p}} \coloneqq \big\{k \in \{1, 2, \dots, K\} : a_k = \tfrac{n}{P} \text{ for some } n \in \mathbb{N}\big\}.$$

Then, according to Lemma 5.1, the Fourier coefficients of the real almost-periodic function are given by

$$\begin{aligned}\hat{f}_P(n) = \frac{-1}{\pi \cdot P} \sum_{k \in L_{\mathrm{np}}} \frac{a_k \cdot C_k + \mathrm{i} \cdot \frac{n}{P} \cdot S_k}{\frac{n^2}{P^2} - a_k^2} + \rho_0 \cdot \chi_{\{0\}}\left(\tfrac{n}{P}\right) \\ + \sum_{k \in L_{\mathrm{p}}} \frac{\rho_k}{2} \cdot \big(\cos(b_k) \pm \mathrm{i} \cdot \sin(b_k)\big) \cdot \chi_{\{\pm a_k\}}\left(\tfrac{n}{P}\right)\end{aligned}$$

for $n \in \mathbb{Z}$ with χ as in (4.4) and C_k, S_k as in (5.2). As in the previous section, we have for $n \in \mathbb{N}$ the modified Fourier coefficients

$$\begin{aligned}\tilde{c}_n^P &= \operatorname{Re} \hat{f}_P(n) + \mathrm{i}\, \tfrac{P}{n} \cdot \operatorname{Im} \hat{f}_P(n) \\ &= \frac{-1}{\pi \cdot P} \sum_{k \in L_{\mathrm{np}}} \frac{a_k \cdot C_k + \mathrm{i}\, S_k}{\frac{n^2}{P^2} - a_k^2} + \sum_{k \in L_{\mathrm{p}}} \frac{\rho_k}{2} \cdot \Big(\cos(b_k) + \mathrm{i} \cdot \tfrac{P}{n} \cdot \sin(b_k)\Big) \cdot \chi_{\{a_k\}}\big(\tfrac{n}{P}\big).\end{aligned}$$

Then

$$\tilde{c}_n^P = \frac{-1}{\pi \cdot P} \cdot \frac{\tilde{p}\Big(\frac{n^2}{P^2}\Big)}{\tilde{q}\Big(\frac{n^2}{P^2}\Big)} + \sum_{k \in L_{\mathrm{p}}} \frac{\rho_k}{2} \cdot \Big(\cos(b_k) + \mathrm{i} \cdot \tfrac{P}{n} \cdot \sin(b_k)\Big) \cdot \chi_{\{a_k\}}\left(\tfrac{n}{P}\right),$$

with

$$\tilde{p}(x) \coloneqq \sum_{k \in L_{\mathrm{np}}} \left(a_k \cdot C_k + \mathrm{i}\, S_k\right) \cdot \prod_{\substack{l \in L_{\mathrm{np}} \\ l \neq k}} \left(x - a_l^2\right)$$

and

$$\tilde{q}(x) \coloneqq \prod_{k \in L_{\mathrm{np}}} (x - a_k^2),$$

where $\tilde{p}$ and $\tilde{q}$ are coprime with $\deg(\tilde{p}) \leq |L_{\mathrm{np}}| - 1$ and $\deg(\tilde{q}) = |L_{\mathrm{np}}|$ as well as $\tilde{q}\Big(\frac{n^2}{P^2}\Big) \neq 0$ for all $n \in \mathbb{N}$. Note that at this point we ignore the constant term ρ_0. We will deal with it separately later.

As for exponential sums, this means that the P-periodic components of the function f correspond to the unattainable points in the rational interpolation problem we have to solve in order to find $\tilde{p}$ and $\tilde{q}$. Therefore, as in Section 4.3.1, if $a_k = \frac{n^*}{P}$ for some $n^* \in \mathbb{N}$ and some $1 \leq k \leq K$, we get

$$\tilde{c}_{n^*}^P + \frac{1}{\pi \cdot P} \cdot \frac{\tilde{p}\left(\frac{(n^*)^2}{P^2}\right)}{\tilde{q}\left(\frac{(n^*)^2}{P^2}\right)} = \frac{\rho_k}{2} \cdot \left(\cos(b_k) + \mathrm{i} \cdot \tfrac{P}{n^*} \cdot \sin(b_k)\right) =: \tilde{y}_{n^*}.$$

Then, since $\rho_k > 0$, we find

$$\rho_k = 2 \cdot \sqrt{\left(\operatorname{Re} \tilde{y}_{n^*}\right)^2 + \left(\tfrac{n^*}{P} \cdot \operatorname{Im} \tilde{y}_{n^*}\right)^2}$$

and

$$b_k = \operatorname{atan2}\left(\tfrac{2n^*}{\rho_k \cdot P} \cdot \operatorname{Im} \tilde{y}_{n^*}, \tfrac{2}{\rho_k} \cdot \operatorname{Re} \tilde{y}_{n^*}\right).$$

Finally, let us consider the constant term ρ_0. We have

$$\hat{f}_P(0) = \frac{1}{\pi \cdot P} \sum_{k \in L_{\mathrm{np}}} \frac{a_k \cdot C_k}{a_k^2} + \rho_0 = \frac{\operatorname{Re} \tilde{p}(0)}{\tilde{q}(0)} + \rho_0,$$

which means that if $\hat{f}_P(0)$ is given, we can easily recover ρ_0 via

$$\rho_0 = \hat{f}_P(0) - \frac{\operatorname{Re} \tilde{p}(0)}{\tilde{q}(0)}.$$

However, if we are given a function and not just Fourier samples, we can alternatively calculate ρ_0 by first computing a reconstruction without taking the constant term into account and then comparing this reconstruction with the given function. If they differ by a constant, then we have found ρ_0. This means that in the case that a function is given it is not necessary to know $\hat{f}_P(0)$.

Lastly, we note that Main Theorem 4.5 holds analogously for real almost-periodic functions, i.e., they are uniquely determined by $2K$ modified Fourier coefficients.

We can now present a dedicated algorithm to reconstruct real almost-periodic functions.

Algorithm 5.6 (Exact Reconstruction of Real Almost-Periodic Signals)**.**
Input: $S = \left\{ \left(\frac{n_j^2}{P^2}, -\pi \cdot P \cdot \tilde{c}_{n_j}^P \right) : n_j \in \mathbb{N},\ j = 1, 2, \ldots, M \right\}$. A set of M modified Fourier samples from a real almost-periodic function f with K components.
$(0, \hat{f}_P(0))$. Fourier sample corresponding to the constant term of f (optional).

(i) IF $\left\| (\tilde{c}_{n_j}^P)_{j=1}^M \right\|_0 \leq \left\lceil \frac{M}{2} \right\rceil$, then STOP. The function f contains only P-periodic components. Reconstruct the parameters ρ_k, a_k, and b_k for $k = 1, 2, \ldots, K$ directly from $\hat{f}_P(n_j)$.

(ii) Set $n = \left\lfloor \frac{M-1}{2} \right\rfloor$ and $m = n - 1$ and construct the matrix $\mathbf{V}_{m,n}(S)$ as in (1.7).

(iii) Find $(\mathbf{p}^T, -\mathbf{q}^T)^T \in \ker \mathbf{V}_{m,n}(S)$.

(iv) IF $\operatorname{null} \mathbf{V}_{m,n}(S) = 0$, then STOP. There are too few samples.

(v) ELSE set p and q the polynomials corresponding to $\mathbf{p}$ and $\mathbf{q}$. Compute $d = \gcd(p, q)$ and set $p^* = c \cdot \frac{p}{d}$ as well as $q^* = c \cdot \frac{q}{d}$, where $c \in \mathbb{R}$ is chosen such that q^* is monic.

(vi) Find the roots r_k of q^* and set $a_k = \sqrt{r_k}$ for $k = 1, 2, \ldots, \deg(q^*)$. Compute

$$C_k = \frac{\operatorname{Re} p^*(a_k^2)}{a_k \cdot \prod_{\substack{l=1 \\ l \neq k}}^K (a_k^2 - a_l^2)} \qquad \text{and} \qquad S_k = \frac{\operatorname{Im} p^*(a_k^2)}{\prod_{\substack{l=1 \\ l \neq k}}^K (a_k^2 - a_l^2)}$$

and set

$$\rho_k = \frac{\sqrt{C_k^2 + S_k^2}}{|\sin(\pi \cdot a_k \cdot P)|}$$

as well as

$$b_k = \operatorname{atan2}\left(\frac{S_k}{\rho_k \cdot \sin(\pi \cdot a_k \cdot P)}, \frac{C_k}{\rho_k \cdot \sin(\pi \cdot a_k \cdot P)} \right) - \pi \cdot a_k \cdot P \bmod 2\pi$$

for $k = 1, 2, \ldots, \deg(q^*)$.

(vii) Compute

$$\tilde{y}_j = \tilde{c}_{n_j}^P + \frac{1}{\pi \cdot P} \cdot \frac{p^*\left(\frac{n_j^2}{P^2}\right)}{q^*\left(\frac{n_j^2}{P^2}\right)}$$

for all $j = 1, 2, \ldots, M$. IF $\tilde{y}_j \neq 0$ set $a_k = \frac{n_j}{P}$ as well as

$$\rho_k = 2 \cdot \sqrt{\left(\operatorname{Re} \tilde{y}_{n_j}\right)^2 + \left(\tfrac{n_j}{P} \cdot \operatorname{Im} \tilde{y}_{n_j}\right)^2}$$

and

$$b_k = \operatorname{atan2}\left(\tfrac{2n_j}{\rho_k \cdot P} \cdot \operatorname{Im} \tilde{y}_{n_j}, \tfrac{2}{\rho_k} \cdot \operatorname{Re} \tilde{y}_{n_j}\right)$$

for $k = \deg(\tilde{q}) + 1, \ldots, K$.

(viii) IF $\hat{f}_P(0)$ is given, compute

$$\rho_0 = \hat{f}_P(0) - \frac{\operatorname{Re} \tilde{p}(0)}{\tilde{q}(0)}$$

Output: Parameters K, a_k, b_k, ρ_k for $k = 1, 2, \ldots, K$ and possibly ρ_0 defining f.

Remark 5.7. As thoroughly discussed in Part I, both Algorithm 4.7 and Algorithm 5.6 are not suited for actual numerical computations. They rather serve as demonstrations that we can reconstruct exponential sums and real almost-periodic functions from Fourier data. We will look at a numerically more feasible algorithm in Chapter 7.

Part III.

Reconstruction of Exponential Sums from DFT Coefficients

6. Reconstruction of Exponential Sums from DFT Coefficients

6.1. Discrete Fourier Transform of Exponential Sums

The algorithms developed in the previous part work with classical Fourier coefficients. They can be used in practice if, for example, the given measurements are already from the Fourier transform of the function we wish to reconstruct. However, when we are given measurements of a function, we usually cannot compute the continuous Fourier transform. Instead, we can compute the discrete Fourier transform of a sample vector. We have published most of the results from this chapter in [36].

Let $\mathbf{f} = (f_m)_{m=0}^{M-1} \in \mathbb{C}^M$ for $M \in \mathbb{N}$ be a vector of equispaced samples from a function f. Then the *discrete Fourier transform* of $\mathbf{f}$ is defined as $\hat{\mathbf{f}} \coloneqq (\hat{f}_n)_{n=0}^{M-1} \in \mathbb{C}^M$ with *discrete Fourier coefficients*

$$\hat{f}_n \coloneqq \frac{1}{M} \sum_{m=0}^{M-1} f_m \cdot \mathrm{e}^{-2\pi\,\mathrm{i}\cdot n \cdot \frac{m}{M}}, \tag{6.1}$$

see for example [92, Chapter 3]. Note that many authors, including [92], as well as many programming library functions, such as Matlab's `fft`, define the discrete Fourier coefficients without the normalization factor $\frac{1}{M}$. However, this does not effect any of our results.

The following lemma is the DFT analogue to Lemma 4.1.

Lemma 6.1. *Let $f(t) = \eta \cdot \mathrm{e}^{\phi \cdot t}$ with $\eta \in \mathbb{C} \setminus \{0\}$ and $\phi \in \mathbb{C}$. Further, let $P > 0$ and suppose that we have $M \in \mathbb{N}$ equispaced samples $(f_m)_{m=0}^{M-1}$ of f from the interval $[0, P)$, i.e., $f_m = f\left(\frac{m \cdot P}{M}\right)$ for $m = 0, 1, \ldots, M-1$. If $\phi = 2\pi\,\mathrm{i} \cdot \frac{l}{P}$ for some $l \in \mathbb{Z}$ then*

$$\hat{f}_n = \begin{cases} \eta, & \textit{if } n \equiv l \mod M \\ 0, & \textit{else} \end{cases}$$

for $n = 0, 1, \dots, M-1$. *Otherwise*

$$\hat{f}_n = \frac{\eta}{M} \cdot \frac{1 - \mathrm{e}^{\phi \cdot P}}{1 - \mathrm{e}^{\frac{\phi \cdot P - 2\pi\,\mathrm{i}\cdot n}{M}}} \tag{6.2}$$

for $n = 0, 1, \dots, M-1$.

Proof. We have

$$\begin{aligned}\hat{f}_n &= \frac{1}{M}\sum_{m=0}^{M-1} f_m \cdot \mathrm{e}^{-2\pi\,\mathrm{i}\cdot n\cdot\frac{m}{M}} \\ &= \frac{1}{M}\sum_{m=0}^{M-1} \eta \cdot \mathrm{e}^{\phi\cdot\frac{m\cdot P}{M}} \cdot \mathrm{e}^{-2\pi\,\mathrm{i}\cdot n\cdot\frac{m}{M}} \\ &= \frac{\eta}{M}\sum_{m=0}^{M-1}\left(\mathrm{e}^{\frac{\phi\cdot P-2\pi\,\mathrm{i}\cdot n}{M}}\right)^m.\end{aligned}$$

If $\phi = 2\pi\,\mathrm{i}\cdot\frac{l}{P}$ for $l \in \mathbb{Z}$ such that $n \equiv l \mod M$, i.e., $l - n = j \cdot M$ for some $j \in \mathbb{Z}$, then

$$\mathrm{e}^{\frac{\phi\cdot P-2\pi\,\mathrm{i}\cdot n}{M}} = \mathrm{e}^{2\pi\,\mathrm{i}\cdot\frac{l-n}{M}} = \mathrm{e}^{2\pi\,\mathrm{i}\cdot\frac{j\cdot M}{M}} = \mathrm{e}^{2\pi\,\mathrm{i}\cdot j} = 1$$

and hence

$$\hat{f}_n = \frac{\eta}{M}\sum_{m=0}^{M-1} 1^m = \frac{\eta}{M}\cdot M = \eta.$$

Otherwise we find

$$\begin{aligned}\hat{f}_n &= \frac{\eta}{M}\cdot\frac{1-\mathrm{e}^{\phi\cdot P-2\pi\,\mathrm{i}\cdot n}}{1-\mathrm{e}^{\frac{\phi\cdot P-2\pi\,\mathrm{i}\cdot n}{M}}} \\ &= \frac{\eta}{M}\cdot\frac{1-\mathrm{e}^{\phi\cdot P}\cdot\overbrace{\mathrm{e}^{-2\pi\,\mathrm{i}\cdot n}}^{=1}}{1-\mathrm{e}^{\frac{\phi\cdot P-2\pi\,\mathrm{i}\cdot n}{M}}},\end{aligned}$$

where we have used that

$$\sum_{m=0}^{M-1} z^m = \frac{1-z^M}{1-z}$$

for any $z \in \mathbb{C} \setminus \{1\}$. Note that $\mathrm{e}^{\frac{\phi\cdot P-2\pi\,\mathrm{i}\cdot n}{M}} = 1$ only if $\phi = 2\pi\,\mathrm{i}\cdot\frac{l}{P}$ for $l \in \mathbb{Z}$ such that $n \equiv l \mod M$. In particular, if $\phi = 2\pi\,\mathrm{i}\cdot\frac{l}{P}$ for some $l \in \mathbb{Z}$ with $n \not\equiv l \bmod M$, it follows that $l - n \in \mathbb{Z} \setminus M \cdot \mathbb{Z}$ and therefore $\mathrm{e}^{\frac{\phi\cdot P-2\pi\,\mathrm{i}\cdot n}{M}} = \mathrm{e}^{2\pi\,\mathrm{i}\cdot\frac{l-n}{M}} \neq 1$ as well as $1 - \mathrm{e}^{\phi\cdot P} = 1 - \mathrm{e}^{2\pi\,\mathrm{i}\cdot l} = 0$, which means that in this case $\hat{f}_n = 0$. □

Remark 6.2. (i) In the following, we will use a rewritten form of (6.2), namely

$$\hat{f}_n = \frac{\eta}{M} \cdot \frac{1 - \mathrm{e}^{\phi \cdot P}}{1 - \mathrm{e}^{\phi \cdot \frac{P}{M}} \cdot \mathrm{e}^{-2\pi\,\mathrm{i} \cdot \frac{n}{M}}} = \mathrm{e}^{2\pi\,\mathrm{i} \cdot \frac{n}{M}} \cdot \frac{\eta}{M} \cdot \frac{1 - \mathrm{e}^{\phi \cdot P}}{\mathrm{e}^{2\pi\,\mathrm{i} \cdot \frac{n}{M}} - \mathrm{e}^{\phi \cdot \frac{P}{M}}}. \tag{6.3}$$

(ii) Let us briefly consider the special case that $f_m = f(m)$, which is equivalent to assuming that $P = M$. As a matter of fact, we can even assume this w.l.o.g. since given any function f as well as numbers P and M we can instead of f use the function $\tilde{f} := f\left(\frac{P}{M} \cdot \cdot\right)$ such that $f\left(\frac{m \cdot P}{M}\right) = \tilde{f}(m)$. Then (6.3) becomes

$$\hat{f}_n = \mathrm{e}^{2\pi\,\mathrm{i} \cdot \frac{n}{M}} \cdot \frac{\eta}{M} \cdot \frac{1 - \mathrm{e}^{\phi \cdot M}}{\mathrm{e}^{2\pi\,\mathrm{i} \cdot \frac{n}{M}} - \mathrm{e}^{\phi}}. \tag{6.4}$$

This also means that the condition that $\phi \neq 2\pi\,\mathrm{i} \cdot \frac{l}{P}$, which is the same as demanding that $\mathrm{e}^{\phi \cdot t}$ is not P-periodic, becomes $\phi \neq 2\pi\,\mathrm{i} \cdot \frac{l}{M}$ for any $l \in \mathbb{Z}$. Writing $z = \mathrm{e}^{\phi}$ with $\operatorname{Im} \phi \in [0, 2\pi)$, as in (3.7), this implies that $z \neq \mathrm{e}^{2\pi\,\mathrm{i} \cdot \frac{l}{M}}$ for any $l = 0, 1, \ldots, M-1$, i.e., z is not an *M-th root of unity*, and (6.4) becomes

$$\hat{f}_n = \mathrm{e}^{2\pi\,\mathrm{i} \cdot \frac{n}{M}} \cdot \frac{\eta}{M} \cdot \frac{1 - z^M}{\mathrm{e}^{2\pi\,\mathrm{i} \cdot \frac{n}{M}} - z}. \tag{6.5}$$

We will see in Remark 6.4 (ii) that $f_m = f(m)$ entails that we must have $\operatorname{Im} \phi \in [0, 2\pi)$. Therefore, assuming that $f_m = f(m)$ automatically means that we can only consider exponential sums as in (3.7).

Since (6.3) is more general in the sense that we can choose P and M independently and (6.4), respectively (6.5), are just special cases, we will throughout this chapter always consider (6.3), i.e., we will assume that f_m are equispaced samples, but not necessarily $f_m = f(m)$ for $m = 0, 1, \ldots, M-1$ and $M \in \mathbb{N}$.

6.2. Reconstruction of Functions without P-Periodic Components

As in in Chapters 4 and 5 we will first consider exponential sums as in (3.6) which do not contain P-periodic components, i.e., functions of the form

$$f(t) = \sum_{k=1}^{K} \eta_k \cdot \mathrm{e}^{\phi_k \cdot t}$$

with $\eta_k, \phi_k \in \mathbb{C} \setminus \{0\}$ such that the ϕ_k are pairwise distinct and $\phi_k \neq 2\pi\,\mathrm{i}\cdot\frac{l}{P}$ for any $l \in \mathbb{Z}$ and all $k = 1, 2, \ldots, K$. For $P > 0$ and $M \in \mathbb{N}$, let $\mathbf{f} = \left(f\left(\frac{m\cdot P}{M}\right)\right)_{m=0}^{M-1}$ be a vector of equispaced samples from f. Then, by equations (6.1) and (6.3), the discrete Fourier transform of $\mathbf{f}$ is given by $\hat{\mathbf{f}} \coloneqq \left(\hat{f}_n\right)_{n=0}^{M-1}$ with

$$\begin{aligned}\hat{f}_n &= \frac{\mathrm{e}^{2\pi\,\mathrm{i}\cdot\frac{n}{M}}}{M} \sum_{k=1}^{K} \eta_k \cdot \frac{1 - \mathrm{e}^{\phi_k \cdot P}}{\mathrm{e}^{2\pi\,\mathrm{i}\cdot\frac{n}{M}} - \mathrm{e}^{\phi_k\cdot\frac{P}{M}}} \\ &= \frac{\mathrm{e}^{2\pi\,\mathrm{i}\cdot\frac{n}{M}}}{M} \cdot \frac{\sum_{k=1}^{K} \eta_k \cdot \left(1 - \mathrm{e}^{\phi_k\cdot P}\right) \cdot \prod_{\substack{l=1\\ l\neq k}}^{K} \left(\mathrm{e}^{2\pi\,\mathrm{i}\cdot\frac{n}{M}} - \mathrm{e}^{\phi_l\cdot\frac{P}{M}}\right)}{\prod_{k=1}^{K} \left(\mathrm{e}^{2\pi\,\mathrm{i}\cdot\frac{n}{M}} - \mathrm{e}^{\phi_k\cdot\frac{P}{M}}\right)}\end{aligned}$$

for $n = 0, 1, \ldots, M-1$. Thus, we find similarly as in Chapter 4 that

$$\hat{f}_n = \frac{\mathrm{e}^{2\pi\,\mathrm{i}\cdot\frac{n}{M}}}{M} \cdot \frac{p\left(\mathrm{e}^{2\pi\,\mathrm{i}\cdot\frac{n}{M}}\right)}{q\left(\mathrm{e}^{2\pi\,\mathrm{i}\cdot\frac{n}{M}}\right)} \tag{6.6}$$

with polynomials

$$p(z) \coloneqq \sum_{k=1}^{K} \eta_k \cdot \left(1 - \mathrm{e}^{\phi_k\cdot P}\right) \cdot \prod_{\substack{l=1\\ l\neq k}}^{K} \left(z - \mathrm{e}^{\phi_l\cdot\frac{P}{M}}\right) \tag{6.7}$$

and

$$q(z) \coloneqq \prod_{k=1}^{K} \left(z - \mathrm{e}^{\phi_k\cdot\frac{P}{M}}\right). \tag{6.8}$$

As in the case of classical Fourier coefficients, we find that $q\left(\mathrm{e}^{2\pi\,\mathrm{i}\cdot\frac{n}{M}}\right) \neq 0$ for any $n \in \mathbb{Z}$, $\deg(q) = K$, $\deg(p) \leq K-1$, and q is monic, see Lemma 4.3 (i), (ii) and (v). However, differently from the classical case, the question whether p and q are coprime, see Lemma 4.3 (iii), needs to be addressed separately.

Lemma 6.3. *If $\phi_k \not\equiv \phi_l \bmod 2\pi\,\mathrm{i}\cdot\frac{M}{P}$ for all $1 \leq k < l \leq K$ then the polynomials p and q in* (6.7) *and* (6.8) *are coprime.*

Proof. First note that q possesses exactly the zeros $\mathrm{e}^{\phi_k\cdot\frac{P}{M}}$ for $k = 1, 2, \ldots, K$ and we have

$$p\left(\mathrm{e}^{\phi_k\cdot\frac{P}{M}}\right) = \eta_k \cdot \left(1 - \mathrm{e}^{\phi_k\cdot P}\right) \cdot \prod_{\substack{l=1\\ l\neq k}}^{K} \left(\mathrm{e}^{\phi_k\cdot\frac{P}{M}} - \mathrm{e}^{\phi_l\cdot\frac{P}{M}}\right).$$

Since by assumption $\eta_k \neq 0$ and $\phi_k \neq 2\pi\,\mathrm{i}\cdot\frac{l}{P}$ for any $l \in \mathbb{Z}$, i.e., $1 - \mathrm{e}^{\phi_k\cdot P} \neq 0$, we must have for $p\left(\mathrm{e}^{\phi_k \cdot \frac{P}{M}}\right)$ to be zero that

$$\mathrm{e}^{\phi_k \cdot \frac{P}{M}} - \mathrm{e}^{\phi_l \cdot \frac{P}{M}} = 0$$

for some $l \in \{1, 2, \ldots, K\}$, $l \neq k$, which is equivalent to

$$\phi_k \equiv \phi_l \bmod 2\pi\,\mathrm{i} \cdot \frac{M}{P} \tag{6.9}$$

and the claim follows. $\square$

Remark 6.4. (i) The condition that $\phi_k \not\equiv \phi_l \bmod 2\pi\,\mathrm{i} \cdot \frac{M}{P}$ for all $1 \leq k < l \leq K$ in Lemma 6.3 is rather technical. However, from equation (6.9), we can deduce a more intuitive condition. For this note that (6.9) is equivalent to $\operatorname{Re}\phi_k = \operatorname{Re}\phi_l$ and $\operatorname{Im}\phi_k \equiv \operatorname{Im}\phi_l \bmod 2\pi \cdot \frac{M}{P}$ or equivalently $\operatorname{Im}\phi_k \cdot \frac{P}{2\pi} \equiv \operatorname{Im}\phi_l \cdot \frac{P}{2\pi} \bmod M$. Assuming that $\operatorname{Im}\phi_k \geq 0$ for all $k = 1, 2, \ldots, K$ this means that if

$$M > \max_{k=1,2,\ldots,K} \{\operatorname{Im}\phi_k\} \cdot \frac{P}{2\pi}, \tag{6.10}$$

it follows that p and q are coprime given the ϕ_k are pairwise distinct, which we always assume anyway. If we allow for $\operatorname{Im}\phi_k \in \mathbb{R}$, then (6.10) becomes

$$M > \max_{k=1,2,\ldots,K} \{|\operatorname{Im}\phi_k|\} \cdot \frac{P}{\pi}.$$

As we will see in Theorem 6.5, this condition is required in any case to be able to uniquely reconstruct the parameters ϕ_k for $k = 1, 2, \ldots, K$. Put differently, this condition is needed in order to prevent *aliasing*. Another way to view this is that we need to sample the function we want to reconstruct with a sampling distance of less than $\frac{2\pi}{\max_{k=1,2,\ldots,K}\{\operatorname{Im}\phi_k\}}$, respectively $\frac{\pi}{\max_{k=1,2,\ldots,K}\{|\operatorname{Im}\phi_k|\}}$.

(ii) Assuming that $f_m = f(m)$, which, as noted in Remark 6.2, is equivalent to $P = M$, the condition (6.10) becomes

$$2\pi > \max_{k=1,2,\ldots,K} \{\operatorname{Im}\phi_k\}.$$

This is just the condition we had for exponential sums with knots $\mathrm{e}^{\phi_k} = z_k \in \mathbb{C}$ for $k = 1, 2, \ldots, K$ as in (3.7). Moreover, in this case the condition that $\phi_k \not\equiv \phi_l \bmod 2\pi\,\mathrm{i} \cdot \frac{M}{P}$ for all $1 \leq k < l \leq K$ from Lemma 6.3 simply becomes $z_k \neq z_l$ for $1 \leq k < l \leq K$.

(iii) Under the additional assumption from Lemma 6.3, properties (iv) and (vi) from Lemma 4.3 hold analogously as for classical Fourier coefficients, i.e., $p \not\equiv 0$ and the set of parameters η_k and ϕ_k for $k = 1, 2, \dots, K$ such that $\deg(p) < K - 1$ is a Lebesgue null set. Since the leading coefficient of p is here the same as in the exact case, the proof of the latter statement can be repeated verbatim. The first assertion follows with Lemma 6.3 by noting that $p\Big(\mathrm{e}^{\phi_k \cdot \frac{P}{M}}\Big) \neq 0$ for $k = 1, 2, \dots, K$.

Before we continue, we need to introduce the so-called *argument function*. Let $z = r \cdot \mathrm{e}^{\mathrm{i}\,\varphi} = x + \mathrm{i}\,y \in \mathbb{C} \setminus \{0\}$ be a complex number, with $r \in \mathbb{R}_{>0}$, $\varphi \in [0, 2\pi)$ and $(x, y) \in \mathbb{R}^2 \setminus \{(0, 0)\}$. Then the argument function is defined as

$$\arg(z) \coloneqq \operatorname{atan2}(y, x) = \varphi,$$

see for example [22, p.16]. Note that we define arg as a single valued function with a value in the interval $[0, 2\pi)$. In the literature, it is sometimes defined as a set valued function mapping z to the set $\{\varphi + 2\pi \cdot n : z = r \cdot \mathrm{e}^{\mathrm{i}\,\varphi}, \varphi \in [0, 2\pi), n \in \mathbb{Z}\}$ or as a function mapping into $[-\pi, \pi)$, which is also the definition used by MATLAB's `angle` function.

The following theorem tells us, analogously to Theorem 4.4, how we can reconstruct the parameters η_k and ϕ_k for $k = 1, 2, \dots, K$ from p and q. For better readability, we will only cover the case that $\operatorname{Im} \phi_k \geq 0$ and remark on the case $\operatorname{Im} \phi_k \in \mathbb{R}$ for $k = 1, 2, \dots, K$ later.

Theorem 6.5. *Given polynomials p and q as in equations* (6.7) *and* (6.8) *with* $\operatorname{Im} \phi_k \geq 0$ *for all* $k = 1, 2, \dots, K$, *where* $M > \max_{k=1,2,\dots,K}\{\operatorname{Im} \phi_k\} \cdot \frac{P}{2\pi}$, *we can reconstruct the parameters* η_k *and* ϕ_k *for* $k = 1, 2, \dots, K$ *via*

$$\operatorname{Im} \phi_k = \frac{M}{P} \cdot \arg(r_k) \quad \textit{and} \quad \operatorname{Re} \phi_k = \frac{M}{P} \cdot \ln\left(|r_k|\right) \qquad \textit{for} \qquad q(r_k) = 0,$$

where ln *denotes the natural logarithm, and*

$$\eta_k = p(r_k) \cdot \left(1 - \mathrm{e}^{\phi_k \cdot P}\right)^{-1} \cdot \prod_{\substack{l=1 \\ l \neq k}}^{K} (r_k - r_l)^{-1}.$$

Proof. Since $\phi_k \neq \phi_l$ and $M > \max_{k=1,2,\dots,K}\{\operatorname{Im} \phi_k\} \cdot \frac{P}{2\pi}$, it follows with Remark 6.4(i) as in the proof of Lemma 6.3 that

$$\phi_k \cdot \frac{P}{M} \not\equiv \phi_l \cdot \frac{P}{M} \bmod 2\pi\,\mathrm{i}$$

for $1 \leq k < l \leq K$. Therefore, q possesses the distinct zeros $r_k = e^{\phi_k \cdot \frac{P}{M}}$ for $k = 1, 2, \ldots, K$. Since $\operatorname{Im} \phi_k \cdot \frac{P}{M} < 2\pi$, we can uniquely reconstruct $\operatorname{Im} \phi_k = \frac{M}{P} \cdot \arg(r_k)$ and $\operatorname{Re} \phi_k = \ln\left(|r_k|^{\frac{M}{P}}\right) = \frac{M}{P} \cdot \ln\left(|r_k|\right)$ for $k = 1, 2, \ldots, K$.

Further, we have for $j = 1, 2, \ldots, K$ that

$$\begin{aligned} p(r_j) &= \sum_{k=1}^{K} \eta_k \cdot \left(1 - \mathrm{e}^{\phi_k \cdot P}\right) \cdot \prod_{\substack{l=1 \\ l \neq k}}^{K} (r_j - r_l) \\ &= \eta_j \cdot \left(1 - \mathrm{e}^{\phi_j \cdot P}\right) \cdot \prod_{\substack{l=1 \\ l \neq j}}^{K} (r_j - r_l). \end{aligned}$$

Since $r_j \neq r_l$ for $j \neq l$ and $\phi_k \neq 2\pi\,\mathrm{i} \cdot \frac{n}{P}$ for any $n \in \mathbb{Z}$, i.e., $\mathrm{e}^{\phi_j \cdot P} \neq 1$, the claim for η_k follows. $\square$

Remark 6.6. If we allow in Theorem 6.5 that $\operatorname{Im} \phi_k \in \mathbb{R}$ for $k = 1, 2, \ldots, K$, then, by Remark 6.4(i), we need to demand that $M > \max_{k=1,2,\ldots,K} \left\{|\operatorname{Im} \phi_k|\right\} \cdot \frac{P}{\pi}$. This implies that $|\operatorname{Im} \phi_k| \cdot \frac{P}{M} < |\operatorname{Im} \phi_k| \cdot \frac{\pi}{\max_{k=1,2,\ldots,K}\{|\operatorname{Im} \phi_k|\}} \leq \pi$ for all $k = 1, 2, \ldots, K$. Therefore, if in this case $\arg(r_k) \in [\pi, 2\pi)$ for some $1 \leq k \leq K$, we know that $\operatorname{Im} \phi_k < 0$. This is consistent with defining the arg function as mapping into $[-\pi, \pi)$.

We can now again interpret equation (6.6) to express that the values $\frac{M}{\mathrm{e}^{2\pi\,\mathrm{i} \cdot \frac{n}{M}}} \cdot \hat{f}_n$ sample the rational function $\frac{p}{q}$ at the points $\mathrm{e}^{2\pi\,\mathrm{i} \cdot \frac{n}{M}}$ for $n = 0, 1, \ldots, M-1$. Theorem 1.2 then tells us that we need at least $2K$ DFT values in order to reconstruct p and q via Algorithm 1.4.

However, Lemma 6.3, respectively Remark 6.4, and Theorem 6.5 also imply that in order to ensure that p and q are coprime and the parameters ϕ_k for $k = 1, 2, \ldots, K$ are uniquely determined by these polynomials, we need that $M > \max_{k=1,2,\ldots,K}\{\operatorname{Im} \phi_k\} \cdot \frac{P}{2\pi}$, respectively $M > \max_{k=1,2,\ldots,K}\{|\operatorname{Im} \phi_k|\} \cdot \frac{P}{\pi}$. Define

$$\phi_{\max} := \begin{cases} \max\limits_{k=1,2,\ldots,K} \{\operatorname{Im} \phi_k\} \cdot \frac{P}{2\pi} & \text{if } \operatorname{Im} \phi_k \geq 0 \text{ for } k = 1, 2, \ldots, K, \\ \max\limits_{k=1,2,\ldots,K} \{|\operatorname{Im} \phi_k|\} \cdot \frac{P}{\pi} & \text{if } \operatorname{Im} \phi_k \in \mathbb{R} \text{ for } k = 1, 2, \ldots, K. \end{cases} \tag{6.11}$$

Then this means that in fact we need $M \geq \max\{2K, \lfloor \phi_{\max} \rfloor + 1\}$ samples in order to reconstruct a given function.

This opens up the question of how to either estimate $\phi_{\max}$ a priori, or how to handle aliasing. However, we will not address these questions here. For a way to deal with the latter problem, we refer to [33].

6.3. Reconstruction of Functions with P-Periodic Components

Analogously to Section 4.3, we will now look at exponential sums as in (3.6), i.e.,

$$f(t) = \eta_0 + \sum_{k=1}^{K} \eta_k \cdot \mathrm{e}^{\phi_k \cdot t},$$

where we now allow for $\eta_0 \neq 0$ and $\phi_k = 2\pi\,\mathrm{i} \cdot \frac{m}{P}$ for some $m \in \mathbb{Z}$ and $1 \leq k \leq K$. For increased readability, we will, again as in Section 4.3, use the notation

$$f(t) = \sum_{k=1}^{K} \eta_k \cdot \mathrm{e}^{\phi_k \cdot t},$$

with $\eta_k \in \mathbb{C} \setminus \{0\}$ and pairwise distinct $\phi_k \in \mathbb{C}$ for all $k = 1, 2, \ldots, K$. Since we allow that $\phi_k = 0$ for some $1 \leq k \leq K$, this means that the constant term is incorporated into the sum. Further, we will assume throughout this section that $\operatorname{Im} \phi_k \geq 0$ for all $k = 1, 2, \ldots, K$ and $M > \max_{k=1,2,\ldots,K}\{\operatorname{Im} \phi_k\} \cdot \frac{P}{2\pi}$.

First, we note that, by Lemma 6.1, the case that the function contains only P-periodic components is the same as in section 4.3.2, i.e., if f contains only P-periodic components, then and only then do we have $\|\mathbf{y}\|_0 \leq \left\lceil \frac{M}{2} \right\rceil$, where

$$\mathbf{y} = M \cdot \left(\frac{\hat{f}_n}{\mathrm{e}^{2\pi\,\mathrm{i} \cdot \frac{n}{M}}} \right)_{n=0}^{M-1}$$

denotes the vector of given sample values used for rational interpolation.

The case that f contains both P-periodic and non-P-periodic components is also very similar to the classical case. Therefore, we will only briefly study it here. For more details, see section 4.3.1.

Let

$$L_{\mathrm{np}} \coloneqq \left\{k \in \{1, 2, \ldots, K\} : \phi_k \neq 2\pi\,\mathrm{i} \cdot \tfrac{m}{P} \text{ for any } m \in \mathbb{Z}\right\} \tag{6.12}$$

and

$$L_{\mathrm{p}} \coloneqq \left\{k \in \{1, 2, \ldots, K\} : \phi_k = 2\pi\,\mathrm{i} \cdot \tfrac{m}{P} \text{ for some } m \in \mathbb{Z}\right\}. \tag{6.13}$$

Then, by equation (6.1) and Lemma 6.1 together with equation (6.3), the values of the discrete Fourier transform $\hat{\mathbf{f}}$ of an equispaced sample vector $\mathbf{f}$ of f are given by

$$\hat{f}_n = \frac{\mathrm{e}^{2\pi\,\mathrm{i}\cdot\frac{n}{M}}}{M} \sum_{k\in L_{\mathrm{np}}} \eta_k \cdot \frac{1-\mathrm{e}^{\phi_k\cdot P}}{\mathrm{e}^{2\pi\,\mathrm{i}\cdot\frac{n}{M}} - \mathrm{e}^{\phi_k\cdot\frac{P}{M}}} + \sum_{k\in L_{\mathrm{p}}} \eta_k \cdot \chi_{\{\phi_k\}}\left(2\pi\,\mathrm{i}\cdot\tfrac{n}{P}\right)$$

for $n = 0, 1, \ldots, M-1$. Now assume that we have a set of samples

$$S = \left\{ \left(\mathrm{e}^{2\pi\,\mathrm{i}\cdot\frac{n}{M}},\, \frac{M}{\mathrm{e}^{2\pi\,\mathrm{i}\cdot\frac{n}{M}}} \cdot \hat{f}_n \right) : n = 0, 1, \ldots, M-1 \right\}.$$

Since $M > \max_{k=1,2,\ldots,K}\{\operatorname{Im}\phi_k\} \cdot \frac{P}{2\pi}$, the set S automatically contains all points

$$\left(\mathrm{e}^{\phi_k\cdot\frac{P}{M}},\, \frac{M}{\mathrm{e}^{\phi_k\cdot\frac{P}{M}}} \cdot \hat{f}_{\phi_k\cdot\frac{P}{2\pi\,\mathrm{i}}} \right) \tag{6.14}$$

for $k \in L_{\mathrm{p}}$, i.e., all the samples corresponding to P-periodic components of f are guaranteed to be contained in the sample set.

We then find that

$$p\left(\mathrm{e}^{2\pi\,\mathrm{i}\cdot\frac{n}{M}}\right) \cdot d\left(\mathrm{e}^{2\pi\,\mathrm{i}\cdot\frac{n}{M}}\right) - \frac{M}{\mathrm{e}^{2\pi\,\mathrm{i}\cdot\frac{n}{M}}} \cdot q\left(\mathrm{e}^{2\pi\,\mathrm{i}\cdot\frac{n}{M}}\right) \cdot d\left(\mathrm{e}^{2\pi\,\mathrm{i}\cdot\frac{n}{M}}\right) = 0 \tag{6.15}$$

for $n = 0, 1, \ldots, M-1$, with

$$p(z) \coloneqq \sum_{k\in L_{\mathrm{np}}} \eta_k \cdot \left(1 - \mathrm{e}^{\phi_k\cdot P}\right) \cdot \prod_{\substack{l\in L_{\mathrm{np}}\\ l\neq k}}^{K} \left(z - \mathrm{e}^{\phi_l\cdot\frac{P}{M}}\right), \tag{6.16}$$

$$q(z) \coloneqq \prod_{k\in L_{\mathrm{np}}} \left(z - \mathrm{e}^{\phi_k\cdot\frac{P}{M}}\right), \tag{6.17}$$

and

$$d(z) \coloneqq \prod_{k\in L_{\mathrm{p}}} \left(z - \mathrm{e}^{\phi_k\cdot\frac{P}{M}}\right), \tag{6.18}$$

where $\deg(p\cdot d) \overset{\text{a.s.}}{=} K-1$ and $\deg(q\cdot d) = K$. Therefore we can reconstruct p, q, and d from $M \geq \max\{2K, \lfloor\phi_{\max}\rfloor + 1\}$ samples, with $\phi_{\max}$ as in (6.11), using Algorithm 1.4.

In particular, if $k \in L_{\mathrm{p}}$ and $\phi_k = 2\pi\,\mathrm{i}\cdot\frac{n^*}{P}$ for some $0 \leq n^* \leq M-1$, then

$$\hat{f}_{n^*} - \frac{e^{2\pi\,\mathrm{i}\cdot\frac{n^*}{M}}}{M} \cdot \frac{p\left(\mathrm{e}^{2\pi\,\mathrm{i}\cdot\frac{n^*}{M}}\right)}{q\left(\mathrm{e}^{2\pi\,\mathrm{i}\cdot\frac{n^*}{M}}\right)} = \eta_k. \tag{6.19}$$

From this we can directly recover the parameters η_k and ϕ_k for $k \in L_{\mathrm{p}}$.

Remark 6.7. We shortly comment on the case that we allow for $\operatorname{Im}\phi_k \in \mathbb{R}$ for $k = 1, 2, \ldots, K$ and accordingly demand that $M > \max_{k=1,2,\ldots,K}\{|\operatorname{Im}\phi_k|\} \cdot \frac{P}{\pi}$. Assume that $k \in L_\mathrm{p}$ and $\operatorname{Im}\phi_k \leq 0$, i.e., $\phi_k = -2\pi\,\mathrm{i}\cdot\frac{n^*}{P}$ for some $n^* \in \mathbb{N}$. Note that since $|\operatorname{Im}\phi_k| < \frac{\pi\cdot M}{P}$, it follows that $n^* < \frac{M}{2}$.

We first consider the samples corresponding to P-periodic components as in (6.14). In this setting, the P-periodic component $\eta_k \cdot \mathrm{e}^{\phi_k \cdot t}$ corresponds to the DFT value $\hat{f}_{M+\phi_k\cdot\frac{P}{2\pi\,\mathrm{i}}} = \hat{f}_{M-n^*}$.

Analogously, we find that (6.19) becomes

$$\hat{f}_{M-n^*} - \frac{e^{2\pi\,\mathrm{i}\cdot\frac{M-n^*}{M}}}{M} \cdot \frac{p\left(\mathrm{e}^{2\pi\,\mathrm{i}\cdot\frac{M-n^*}{M}}\right)}{q\left(\mathrm{e}^{2\pi\,\mathrm{i}\cdot\frac{M-n^*}{M}}\right)} = \hat{f}_{M-n^*} - \frac{e^{-2\pi\,\mathrm{i}\cdot n^*}}{M} \cdot \frac{p\left(\mathrm{e}^{-2\pi\,\mathrm{i}\cdot n^*}\right)}{q\left(\mathrm{e}^{-2\pi\,\mathrm{i}\cdot n^*}\right)} = \eta_k.$$

Finally, we want to remark on the reconstruction of real almost-periodic functions from DFT data.

Remark 6.8. As we have seen in equation (3.8), we can always interpret a real almost-periodic function containing K components as an exponential sum containing $2K$ components, implying that we need twice as many, i.e., $4K$, samples for reconstruction. This is true for both the continuous and the discrete case. In Chapter 5, we saw that for classical Fourier coefficients it sufficed to have $2K$ samples in order to reconstruct a real almost-periodic function. We explained this heuristically by the fact that we are able to collect the data asymmetrically and use that $\hat{f}_P(-n) = \overline{\hat{f}_P(n)}$, see Remark 5.4. We also looked for an analogous result for DFT data but could not find one. We believe that it is not possible to obtain such a result. A heuristic explanation for this is that for real input signals, the DFT values always satisfy $\hat{f}_{M-n} = \overline{\hat{f}_n}$ for $n = 1, 2, \ldots, M-1$ and $\hat{f}_0 \in \mathbb{R}$, see [92, Chapter 3]. This means that the discrete Fourier transform always returns symmetric data, which in turn means that we cannot exploit any asymmetry in the data, as we did in Chapter 5. In some sense, the DFT of length M of a real input function only returns $\frac{M}{2}$ independent data points. Put differently, the DFT of length $2K$ of a real almost-periodic function does not give us enough information, namely only K independent data points, in order to reconstruct the given function, which requires $2K$ data points.

6.4. Main Theorem and Algorithm for DFT Coefficients

We now come to the second main result of this thesis.

Main Theorem 6.9 (Reconstruction of Exponential Sums from DFT Coefficients)**.** *Let $f(t) = \sum_{k=1}^{K} \eta_k \cdot \mathrm{e}^{\phi_k \cdot t}$ be an exponential sum with $\eta_k \in \mathbb{C} \setminus \{0\}$ and pairwise distinct $\phi_k \in \mathbb{C}$ for all $k = 1, 2, \ldots, K$. Further, let $P > 0$ be arbitrary and $M \geq \max\{2K, \lfloor \phi_{\max} \rfloor + 1\}$ with $\phi_{\max}$ as in (6.11). Then the parameters η_k and ϕ_k can be uniquely reconstructed from M samples of the form $f_m = f\left(\frac{m \cdot P}{M}\right)$ using the values of the discrete Fourier transform $\hat{f}_n$ for $n = 0, 1, \ldots, M-1$.*

Parameter K?

Proof. From equation (6.15) and Lemma 6.3 together with Remark 6.4 as well as Lemma 4.3(v), it follows by Theorem 1.2 that the polynomials p, q, and d as in equations (6.16), (6.17), and (6.18) can be reconstructed from $2K$ samples using Algorithm 1.4. Since q and d are monic and P is known, we can uniquely reconstruct all polynomials. Using Theorem 6.5 and equation (6.19), we can reconstruct η_k and ϕ_k for $k = 1, 2, \ldots, K$. By Theorem 3.11, these parameters uniquely determine the exponential sum f. □

A corollary of this result is a new uniqueness theorem for exponential sums. Recall that Theorem 3.11 states that two exponential sums are equal, i.e., defined by the same parameters, if they agree on any interval $T \subseteq \mathbb{R}$ of non-zero length.

Theorem 6.10 (Uniqueness of Exponential Sums)**.** *Let*

$$f(t) = \sum_{k=1}^{K} \eta_k \cdot \mathrm{e}^{\phi_k \cdot t} \quad \text{and} \quad g(t) = \sum_{k=1}^{\tilde{K}} \gamma_k \cdot \mathrm{e}^{\psi_k \cdot t}$$

be exponential sums, where $\eta_k, \gamma_k \in \mathbb{C} \setminus \{0\}$ for all $k = 1, 2, \ldots, K$, respectively $\tilde{K}$, and $\phi_1, \phi_2, \ldots, \phi_K \in \mathbb{C} \setminus \{0\}$ as well as $\psi_1, \psi_2, \ldots, \psi_{\tilde{K}} \in \mathbb{C} \setminus \{0\}$ are each pairwise distinct. Further, let $P > 0$ be arbitrary and $M \geq \max\left\{2K, 2\tilde{K}, \lfloor \phi_{\max} \rfloor + 1, \lfloor \psi_{\max} \rfloor + 1\right\}$, where $\phi_{\max}$ and $\psi_{\max}$ are defined as in (6.11). If $f\left(\frac{m \cdot P}{M}\right) = g\left(\frac{m \cdot P}{M}\right)$ for $m = 0, 1, \ldots, M-1$ then $K = \tilde{K}$ and, after a suitable ordering of the terms, $\phi_k = \psi_k$ as well as $\eta_k = \gamma_k$ for all $k = 1, 2, \ldots, K$.

This result is not new, in the sense that there exist algorithms which can recover exponential sums from discrete samples, implying the statement of Theorem 6.10. We will briefly touch on these algorithms in the next chapter and in Appendix B.

We can now present an algorithm for the reconstruction of exponential sums from discrete function samples.

Algorithm 6.11 (Reconstruction of Exponential Sums from Discrete Samples)**.**

Input: $\mathbf{f} = (f_m)_{m=0}^{M-1} = \left(f\left(\frac{m \cdot P}{M}\right)\right)_{m=0}^{M-1} \in \mathbb{C}^M$. Vector of equispaced samples from an exponential sum f with K components, where $P > 0$ and $M \geq \lfloor \phi_{\max} \rfloor + 1$.

(i) Compute the DFT $\hat{\mathbf{f}} = (\hat{f}_n)_{n=0}^{M-1}$ of $\mathbf{f}$ and set

$$S = \left\{ \left(\mathrm{e}^{2\pi\,\mathrm{i}\cdot\frac{n}{M}},\ \frac{M}{\mathrm{e}^{2\pi\,\mathrm{i}\cdot\frac{n}{M}}} \cdot \hat{f}_n \right) : n = 0, 1, \dots, M-1 \right\}.$$

(ii) IF $\left\| (\hat{f}_n)_{n=0}^{M-1} \right\|_0 \leq \left\lceil \frac{M}{2} \right\rceil$, then STOP. The function f contains only P-periodic components. Reconstruct the parameters ϕ_k and η_k for $k = 1, 2, \dots, K$ directly from $\hat{f}_n$.

(iii) Construct the matrix $\mathbf{V}_{\lfloor \frac{M-1}{2} \rfloor - 1, \lfloor \frac{M-1}{2} \rfloor}(S)$ as in (1.7).

(iv) Find $(\mathbf{p}^T, -\mathbf{q}^T)^T \in \ker \mathbf{V}_{m,n}(S)$.

(v) IF $\operatorname{null} \mathbf{V}_{\lfloor \frac{M-1}{2} \rfloor - 1, \lfloor \frac{M-1}{2} \rfloor}(S) = 0$, then STOP. There are too few samples.

(vi) ELSE set p and q the polynomials corresponding to $\mathbf{p}$ and $\mathbf{q}$. Compute $d = \gcd(p, q)$ and set $p^* = c \cdot \frac{p}{d}$ as well as $q^* = c \cdot \frac{q}{d}$, where $c \in \mathbb{R}$ is chosen such that q^* is monic.

(vii) Find the roots r_k of q^* and compute $\operatorname{Im} \phi_k = \frac{M}{P} \cdot \arg(r_k)$, $\operatorname{Re} \phi_k = \frac{M}{P} \cdot \ln(|r_k|)$ as well as

$$\eta_k = p^*(r_k) \cdot \left(1 - \mathrm{e}^{\phi_k \cdot P}\right)^{-1} \cdot \prod_{\substack{l=1 \\ l \neq k}}^{\deg(q^*)} (r_k - r_l)^{-1}$$

for $k = 1, 2, \dots, \deg(q^*)$.

(viii) Compute

$$\tilde{y}_n = \frac{M}{\mathrm{e}^{2\pi\,\mathrm{i}\cdot\frac{n}{M}}} \cdot \hat{f}_n - \frac{p^*\left(\mathrm{e}^{2\pi\,\mathrm{i}\cdot\frac{n}{M}}\right)}{q^*\left(\mathrm{e}^{2\pi\,\mathrm{i}\cdot\frac{n}{M}}\right)}$$

for all $n = 0, 1, \dots, M-1$. IF $\tilde{y}_n \neq 0$, set $\phi_k = 2\pi\,\mathrm{i} \cdot \frac{n}{P}$ and $\eta_k = \frac{\mathrm{e}^{2\pi\,\mathrm{i}\cdot\frac{n}{M}}}{M} \cdot \tilde{y}_n$ for $k = \deg(q^*) + 1, \dots, K$.

Output: Parameters K, η_k, and ϕ_k for $k = 1, 2, \ldots, K$ defining f.

Remark 6.12. (i) Note that in step (viii) of Algorithm 6.11 we tacitly assumed that $\operatorname{Im} \phi_k \geq 0$ for all $k = 1, 2, \ldots, K$. If we allow for $\operatorname{Im} \phi_k \in \mathbb{R}$, then we have to change this step according to Remark 6.7.

(ii) If the number of given samples is less than $\lfloor \phi_{\max} \rfloor + 1$ then we cannot reconstruct the imaginary parts of all ϕ_k correctly. As mentioned before, this problem is discussed in more depth in [33]. In the case that, instead of just samples, we are given a function from which we can sample at will, we can always compare the outcome of Algorithm 6.11 to the given function and, if they differ, rerun the algorithm with more samples. However, this only works if we have access to the actual function.

7. The ESPIRA Algorithm

7.1. The Algorithm

Finally, we can harvest the fruits of our labor so far. In the previous chapter, we developed an algorithm to reconstruct exponential sums from function samples using the discrete Fourier transform and rational interpolation. In Chapters 1 and 2, we looked in detail at a numerically stable method for rational interpolation. Now we combine our findings to obtain a numerically stable algorithm for the reconstruction of exponential sums from discrete sample values. Although we will only present the algorithm for DFT data, it can in principle, with minor changes, also be applied to data from classical Fourier coefficients as well as modified Fourier coefficients from real almost-periodic functions. We have published the results of this section in [36].

We already have almost all ingredients for our algorithm. Given equispaced samples f_m for $m = 0, 1, \ldots, M-1$, with sufficiently large M, from an exponential sum f, we first compute the DFT values $\hat{f}_n$ for $n = 0, 1, \ldots, M-1$. Then we use the AAA algorithm as discussed in Sections 1.5 and 2.1 in order to find a rational interpolant for the given data

$$S = \left\{ \left(\mathrm{e}^{2\pi\,\mathrm{i}\cdot\frac{n}{M}}, \, \frac{M}{\mathrm{e}^{2\pi\,\mathrm{i}\cdot\frac{n}{M}}} \cdot \hat{f}_n \right) : n = 0, 1, \ldots, M-1 \right\}.$$

Using a Froissart doublet removal process as in Section 2.3, we then find the unattainable points in the given data, which, by equation (6.15), correspond to the P-periodic components of the given function.

Let $\tilde{S}$ denote the subset of the sample set S, where all unattainable points have been removed. We thus obtain a minimal rational function in barycentric form, given by a weight vector $\mathbf{u}$ and a partition S', T' of $\tilde{S}$, which interpolates all points in $\tilde{S}$. We will denote the set of indices of the points in $\tilde{S}$ by $I_{\tilde{S}}$.

Now we know by (6.6) and Lemma 1.10 that

$$\begin{aligned} r_{\mathbf{u},T'}(z) &= \frac{\sum_{j=1}^{|T'|} \frac{u_j}{z-\mathrm{e}^{2\pi\,\mathrm{i}\cdot\frac{n_j}{M}}} \cdot M \cdot \mathrm{e}^{-2\pi\,\mathrm{i}\cdot\frac{n_j}{M}} \cdot \hat{f}_{n_j}}{\sum_{j=1}^{|T'|} \frac{u_j}{z-\mathrm{e}^{2\pi\,\mathrm{i}\cdot\frac{n_j}{M}}}} \\ &= \frac{\sum_{k\in L_{\mathrm{np}}} \eta_k \cdot \left(1-\mathrm{e}^{\phi_k\cdot P}\right) \cdot \prod_{\substack{l\in L_{\mathrm{np}}\\ l\neq k}}^{K} \left(z-\mathrm{e}^{\phi_l\cdot\frac{P}{M}}\right)}{\prod_{k\in L_{\mathrm{np}}} \left(z-\mathrm{e}^{\phi_k\cdot\frac{P}{M}}\right)} \\ &= \sum_{k\in L_{\mathrm{np}}} \frac{\eta_k\cdot\left(1-\mathrm{e}^{\phi_k\cdot P}\right)}{z-\mathrm{e}^{\phi_k\cdot\frac{P}{M}}}, \end{aligned}$$

where n_j for $j = 1, 2, \dots, |T'|$ denote the indices of the samples in T' and L_{np} denotes the indices of the non-P-periodic components of f as in (6.12). It follows that the zeros of $q(z) = \prod_{k\in L_{\mathrm{np}}} \left(z-\mathrm{e}^{\phi_k\cdot\frac{P}{M}}\right)$ are the poles of $r_{\mathbf{u},T}$. We already discussed how to find the poles of a rational function in barycentric form using a generalized eigenvalue problem in Section 2.4. Once we know the poles $r_k = \mathrm{e}^{\phi_k\cdot\frac{P}{M}}$, we can reconstruct ϕ_k for $k = 1, 2, \dots, K$ as in Theorem 6.5 via $\operatorname{Im}\phi_k = \frac{M}{P}\cdot\arg(r_k)$ and $\operatorname{Re}\phi_k = \frac{M}{P}\cdot\ln\left(|r_k|\right)$. Knowing ϕ_k we can than also compute $\mathrm{e}^{\phi_k\cdot P}$.

We mentioned in Section 2.5.4 that it is hard to reconstruct the numerator polynomial p since, due to numerical errors, it is likely that the reconstructed function has type $(|T'|-1, |T'|-1)$. However, we do not actually need to reconstruct p, as would be necessary to apply Theorem 6.5, in order to find the last missing parameters η_k for $k = 1, 2, \dots, K$. Instead, we note that

$$r_{\mathbf{u},T'}\left(\mathrm{e}^{2\pi\,\mathrm{i}\cdot\frac{n}{M}}\right) = \sum_{k\in L_{\mathrm{np}}} \frac{\eta_k\cdot\left(1-\mathrm{e}^{\phi_k\cdot P}\right)}{\mathrm{e}^{2\pi\,\mathrm{i}\cdot\frac{n}{M}}-\mathrm{e}^{\phi_k\cdot\frac{P}{M}}} = \frac{M}{\mathrm{e}^{2\pi\,\mathrm{i}\cdot\frac{n}{M}}}\cdot\hat{f}_n$$

for all $n \in I_{\tilde{S}}$. This is an overdetermined system with the K unknown parameters η_k and $|I_{\tilde{S}}| = M - |L_{\mathrm{p}}| \geq K + |L_{\mathrm{np}}|$ equations, where L_{p} denotes the indices of the P-periodic components of f as in (6.13). We can therefore compute the least squares solution $\mathbf{x} = (x_k)_{k\in L_{\mathrm{np}}}$ of

$$\left(\frac{1}{\mathrm{e}^{2\pi\,\mathrm{i}\cdot\frac{n}{M}}-\mathrm{e}^{\phi_k\cdot\frac{P}{M}}}\right)_{n\in I_{\tilde{S}},\,k\in L_{\mathrm{np}}} \cdot\mathbf{x} = \left(\frac{M}{\mathrm{e}^{2\pi\,\mathrm{i}\cdot\frac{n}{M}}}\cdot\hat{f}_n\right)_{n\in I_{\tilde{S}}}$$

and find

$$\eta_k = \frac{x_k}{1-\mathrm{e}^{\phi_k\cdot P}}$$

for all $k \in L_{\mathrm{np}}$.

Now we have everything we need in order to present a numerically stable reconstruction algorithm. However, before we do that, we present one last result. In Theorem 1.20, we have seen that the convergence of the AAA algorithm relies on the fact that the Löwner matrix, which lies at the core of the iteration process, becomes singular at some point. We gave a general proof of this fact in Theorem 1.15. If the given data S does not contain unattainable points, i.e., there is a rational function r interpolating all points in S, then Theorem 1.8 tells us that the rank of the Löwner matrix $\mathbf{L}(S', T')$ as in equation (1.21) with a partition of S such that $|S'|, |T'| \geq \deg(r)$ is equal to $\deg(r)$. In [36] and [90], we proved this directly, where we used the structure of the given Fourier data $\hat{f}_P(n)$ respectively $\hat{f}_n$. We present the proof for DFT data here. The continuous case is very similar.

Lemma 7.1. *Let $f = \sum_{k=1}^{K} \eta_k \cdot \mathrm{e}^{\phi_k \cdot t}$ be an exponential sum with $\eta_k, \phi_k \in \mathbb{C} \setminus \{0\}$ for $k = 1, 2, \ldots, K$. Further, let $P > 0$ and $M \geq 2K$ such that $\phi_k \neq 2\pi\,\mathrm{i} \cdot \frac{j}{P}$ for any $j \in \mathbb{Z}$ and all $k = 1, 2, \ldots, K$ as well as $\phi_k \not\equiv \phi_l \bmod 2\pi\,\mathrm{i} \cdot \frac{M}{P}$ for $1 \leq k < l \leq K$. Given the discrete Fourier transform $\hat{\mathbf{f}} = (\hat{f}_n)_{n=0}^{M-1}$ of a vector $\mathbf{f} \in \mathbb{C}^M$ of equispaced samples from f, set*

$$S = \left\{ \left(\mathrm{e}^{2\pi\,\mathrm{i} \cdot \frac{n}{M}}, M \cdot \mathrm{e}^{-2\pi\,\mathrm{i} \cdot \frac{n}{M}} \cdot \hat{f}_n \right) : n = 0, 1, \ldots, M-1 \right\}$$

and let S', T' be a partition of S such that $|S'| \geq K$. If $|T'| \geq K$ then $\operatorname{rk} \mathbf{L}(S', T') = K$ *and if $|T'| < K$ then* $\operatorname{null} \mathbf{L}(S', T') = 0$.

Proof. Denote by $I_{S'}$ and $I_{T'}$ the sets of indices of samples in S', respectively T'. Then we have

$$\begin{aligned}
\mathbf{L}(S', T') &= M \cdot \left(\frac{\mathrm{e}^{-2\pi\,\mathrm{i} \cdot \frac{s}{M}}\, \hat{f}_s - \mathrm{e}^{-2\pi\,\mathrm{i} \cdot \frac{t}{M}}\, \hat{f}_t}{\mathrm{e}^{2\pi\,\mathrm{i} \cdot \frac{s}{M}} - \mathrm{e}^{2\pi\,\mathrm{i} \cdot \frac{t}{M}}} \right)_{s \in I_{S'}, t \in I_{T'}} \\
&= \left(\frac{\sum_{k=1}^{K} \eta_k \left(1 - \mathrm{e}^{\phi_k \cdot P}\right) \cdot \left(\frac{1}{\mathrm{e}^{2\pi\,\mathrm{i} \cdot \frac{s}{M}} - \mathrm{e}^{\phi_k \cdot \frac{P}{M}}} - \frac{1}{\mathrm{e}^{2\pi\,\mathrm{i} \cdot \frac{t}{M}} - \mathrm{e}^{\phi_k \cdot \frac{P}{M}}} \right)}{\mathrm{e}^{2\pi\,\mathrm{i} \cdot \frac{s}{M}} - \mathrm{e}^{2\pi\,\mathrm{i} \cdot \frac{t}{M}}} \right)_{s \in I_{S'}, t \in I_{T'}} \\
&= \left(\sum_{k=1}^{K} \eta_k \left(\mathrm{e}^{\phi_k \cdot P} - 1 \right) \cdot \frac{1}{\left(\mathrm{e}^{2\pi\,\mathrm{i} \cdot \frac{s}{M}} - \mathrm{e}^{\phi_k \cdot \frac{P}{M}} \right) \cdot \left(\mathrm{e}^{2\pi\,\mathrm{i} \cdot \frac{t}{M}} - \mathrm{e}^{\phi_k \cdot \frac{P}{M}} \right)} \right)_{s \in I_{S'}, t \in I_{T'}}
\end{aligned}$$

$$= \left(\frac{1}{\mathrm{e}^{2\pi\,\mathrm{i}\cdot\frac{s}{M}} - \mathrm{e}^{\phi_k\cdot\frac{P}{M}}} \right)_{s\in I_{S'},\,k=1,\dots,K} \cdot \operatorname{diag}\Big(\Big(\eta_k \left(\mathrm{e}^{\phi_k\cdot P} - 1\right)\Big)_{k=1}^{K} \Big) \cdot \left(\frac{1}{\mathrm{e}^{2\pi\,\mathrm{i}\cdot\frac{t}{M}} - \mathrm{e}^{\phi_k\cdot\frac{P}{M}}} \right)_{k=1,\dots,K,\,t\in I_{T'}},$$

where we have used (6.3) for the second equality. Since $\phi_k \neq 2\pi\,\mathrm{i}\cdot\frac{j}{P}$ for any $j \in \mathbb{Z}$ and all $k = 1, 2, \dots, K$, it follows that the matrix

$$\mathbf{E} \coloneqq \operatorname{diag}\Big(\Big(\eta_k \left(\mathrm{e}^{\phi_k\cdot P} - 1\right)\Big)_{k=1}^{K} \Big)$$

has full rank. Further, as already mentioned in the proof of Theorem 1.15, the ranks of the Cauchy matrices

$$\mathbf{C}_{S'} \coloneqq \left(\frac{1}{\mathrm{e}^{2\pi\,\mathrm{i}\cdot\frac{s}{M}} - \mathrm{e}^{\phi_k\cdot\frac{P}{M}}} \right)_{s\in I_{S'},\,k=1,\dots,K}$$

and

$$\mathbf{C}_{T'} \coloneqq \left(\frac{1}{\mathrm{e}^{2\pi\,\mathrm{i}\cdot\frac{t}{M}} - \mathrm{e}^{\phi_k\cdot\frac{P}{M}}} \right)_{k=1,\dots,K,\,t\in I_{T'}}$$

are given by $\min\{|I_{S'}|, K\}$ and $\min\{K, |I_{T'}|\}$. Since we assume that $|I_{S'}| \geq K$, it follows that $\mathbf{C}_{S'}$ has full column rank and therefore the matrix $\mathbf{C}_{S'} \cdot \mathbf{E}$ has also full column rank K. If now $|I_{T'}| < K$ then $\mathbf{C}_{T'}$ too has full column rank and we have $\operatorname{null}(\mathbf{C}_{S'} \cdot \mathbf{E} \cdot \mathbf{C}_{T'}) = \operatorname{null} \mathbf{L}(S', T') = 0$. However, if $|I_{T'}| \geq K$ then $\operatorname{rk} \mathbf{C}_{T'} = K$ and it follows that $\operatorname{rk} \mathbf{L}(S', T') = K$. □

Finally, we present our algorithm. In [36], we named this algorithm ESPIRA for Estimation of Signal Parameters by Iterative Rational Approximation.

Algorithm 7.2 (ESPIRA).
Input: $\mathbf{f} = (f_m)_{m=0}^{M-1} = \left(f\left(\frac{m\cdot P}{M}\right)\right)_{m=0}^{M-1} \in \mathbb{C}^M$. Vector of equispaced samples from an exponential sum f with K components, where $P > 0$ and $M \geq \max\{2K, \lfloor\phi_{\max}\rfloor + 1\}$.

(i) Compute the DFT $\hat{\mathbf{f}} = (\hat{f}_n)_{n=0}^{M-1}$ of $\mathbf{f}$ and set

$$S = \left\{ \left(\mathrm{e}^{2\pi\,\mathrm{i}\cdot\frac{n}{M}}, \frac{M}{\mathrm{e}^{2\pi\,\mathrm{i}\cdot\frac{n}{M}}} \cdot \hat{f}_n \right) : n = 0, 1, \dots, M-1 \right\}.$$

(ii) IF $\left\|\left(\hat{f}_n\right)_{n=0}^{M-1}\right\|_0 \leq \left\lceil \frac{M}{2} \right\rceil$, then STOP. The function f contains only P-periodic components. Reconstruct the parameters ϕ_k and η_k for $k = 1, 2, \ldots, K$ directly from $\hat{f}_n$.

(iii) ELSE use Algorithm 2.1 with a Froissart doublet removal process to reconstruct a rational function r interpolating the DFT values of the non-periodic part of f and find the indices $k_j \in \{0, 1, \ldots, M-1\}$ for $j = 1, 2, \ldots, \nu$ corresponding to the periodic components of f.

(iv) Set $\phi_{k_j} = 2\pi\,\mathrm{i} \cdot \frac{k_j}{P}$ and $\eta_{k_j} = \hat{f}_{k_j} - \frac{\mathrm{e}^{2\pi\,\mathrm{i}\cdot\frac{k_j}{M}}}{M} \cdot r\left(\mathrm{e}^{2\pi\,\mathrm{i}\cdot\frac{k_j}{M}}\right)$ for $j = 1, 2, \ldots, \nu$.

(v) Find the poles ξ_{k_j} of r and reconstruct ϕ_{k_j} via $\operatorname{Im}\phi_{k_j} = \frac{M}{P} \cdot \arg(\xi_{k_j})$ and $\operatorname{Re}\phi_{k_j} = \frac{M}{P} \cdot \ln(|\xi_{k_j}|)$ for $j = \nu+1, \nu+2, \ldots, K$.

(vi) Compute the least squares solution $\mathbf{x} = \left(x_{k_j}\right)_{j=\nu+1}^{K}$ of

$$\left(\frac{1}{\mathrm{e}^{2\pi\,\mathrm{i}\cdot\frac{n}{M}} - \mathrm{e}^{\phi_{k_j}\cdot\frac{P}{M}}}\right)_{n\in I_{\tilde{S}},\, j=\nu+1,\ldots,K} \cdot \mathbf{x} = M \cdot \left(\mathrm{e}^{-2\pi\,\mathrm{i}\cdot\frac{n}{M}} \cdot \hat{f}_n\right)_{n\in I_{\tilde{S}}},$$

where $I_{\tilde{S}}$ denotes the index set of the attainable sample points in S, and set

$$\eta_{k_j} = \frac{x_{k_j}}{1 - \mathrm{e}^{\phi_{k_j}\cdot P}}$$

for $j = \nu+1, \nu+2, \ldots, K$.

Output: Parameters K, η_k, and ϕ_k for $k = 1, 2, \ldots, K$ defining f.

Remark 7.3. Like before in step (viii) of Algorithm 6.11, we have tacitly assumed in step (iv) of Algorithm 7.2 that $\operatorname{Im}\phi_k \geq 0$ for all $k = 1, 2, \ldots, K$. As already mentioned in Remark 6.12, if we allow for $\operatorname{Im}\phi_k \in \mathbb{R}$, then we have to change this step according to Remark 6.7.

7.2. Numerical Examples

We finish this thesis by looking at how well our algorithm works in practice. In order to assess its performance, we will compare it to the *Matrix Pencil Method (MPM)* [57]. This is a so-called *Prony-like* method. *Prony's Method* [34] is a well-known and popular method to reconstruct exponential sums from discrete samples. However, the classical method is known to be unstable, see [94]. Therefore, several more stable

Prony-like methods have been proposed. Besides MPM, these are for example MUSIC [103] and ESPRIT [99]. For our comparisons in this section, we will use an improved, state-of-the-art version of the MPM algorithm as proposed in [97]. We present the MPM algorithm in more detail in Appendix B. In [36], we also gave a short review of MPM and ESPRIT.

As in Chapter 2, we will look at the performance on large sets of randomly generated functions, but we will also consider specific examples in more detail. The test functions

$$f(t) = \sum_{k=1}^{K} \eta_k \cdot \mathrm{e}^{\phi_k \cdot t}$$

are generated by fixing the number of components K and the period P over which we sample, i.e., we sample from the interval $[0, P)$, as well as the ranges for $\operatorname{Re}\eta_k$, $\operatorname{Im}\eta_k$, $\operatorname{Re}\phi_k$ and $\operatorname{Im}\phi_k$ for $k = 1, 2, \dots, K$. Let $r_{\operatorname{Re}\eta}$ be the range for $\operatorname{Re}\eta_k$ for $k = 1, 2, \dots, K$. We then choose $\operatorname{Re}\eta_k$ uniformly from the interval $[-r_{\operatorname{Re}\eta}, r_{\operatorname{Re}\eta}]$ and proceeded in the same way for $\operatorname{Im}\eta_k, \operatorname{Re}\phi_k$ and $\operatorname{Im}\phi_k$ for all $k = 1, 2, \dots, K$. Additionally, we make sure that the parameters ϕ_k for $k = 1, 2, \dots, K$ are always pairwise distinct. We always use 1.000 randomly chosen functions for our experiments. Further, unless stated otherwise, we use

$$M = \max\left\{2K + 2, \left\lfloor r_{\operatorname{Im}\phi} \cdot \tfrac{P}{\pi} \right\rfloor + 1\right\} \tag{7.1}$$

samples for the reconstruction. This is the minimal number of samples needed in order to guarantee that the function can be reconstructed, see Remark 1.17 (i), and there is no aliasing, i.e., ϕ_k for $k = 1, 2, \dots, K$ can always be reconstructed, see Remark 6.4(i).

The reported errors are calculated as follows. For the parameters η_k and ϕ_k, let $\tilde{\eta}_j$ and $\tilde{\phi}_j$ be the output of the respective algorithms for $j, k = 1, 2, \dots, K$. First, we match the output to the given parameters by finding

$$\tilde{\eta}_k = \underset{j=1,2,\dots,K}{\operatorname{argmin}} |\eta_k - \tilde{\eta}_j| \qquad \text{and} \qquad \tilde{\phi}_k = \underset{j=1,2,\dots,K}{\operatorname{argmin}} |\phi_k - \tilde{\phi}_j| \tag{7.2}$$

for $k = 1, 2, \dots, K$. We then compute the relative errors

$$\Delta\eta := \frac{\max_{k=1,2,\dots,K} |\eta_k - \tilde{\eta}_k|}{\max_{k=1,2,\dots,K} |\eta_k|} \qquad \text{and} \qquad \Delta\phi := \frac{\max_{k=1,2,\dots,K} |\phi_k - \tilde{\phi}_k|}{\max_{k=1,2,\dots,K} |\phi_k|}.$$

In order to compute the error for the reconstruction $\tilde{f}$ of f, we sample both functions in the interval $[0, P - 0.01]$ with a sampling distance of 0.01, which means that we get

$N = 100 \cdot \lfloor P \rfloor$ samples f_n and $\tilde{f}_n$. The error is then computed as

$$\Delta f := \frac{\max_{n=1,2,\ldots,N} |f_n - \tilde{f}_n|}{\max_{n=1,2,\ldots,N} |f_n|}.$$

We quickly need to talk about the matching process in (7.2). Since the parameters ϕ_k are always pairwise distinct, we can, at least in principle, always match the reconstructed value $\tilde{\phi}_k$ uniquely to the original value ϕ_k for all $k = 1, 2, \ldots, K$. For $\tilde{\eta}_k$ and η_k, on the other hand, this is not necessarily true since we never demand that the original η_k have to be pairwise distinct for $k = 1, 2, \ldots, K$. Therefore, it would make sense to only match $\tilde{\phi}_k$ and ϕ_k and then match $\tilde{\eta}_k$ and η_k accordingly. We can do this since the algorithms always return the reconstructed parameters in pairs $(\tilde{\eta}_j, \tilde{\phi}_j)$ for $j = 1, 2, \ldots, K$. However, in practice it can and does happen that $\operatorname{argmin}_{j=1,2,\ldots,K} |\phi_k - \tilde{\phi}_j|$ is not unique for all $k = 1, 2, \ldots, K$. In other words, it can happen that two (or possibly even more, which is highly unlikely though) reconstructed values $\tilde{\phi}_j$ lie closer to the same $\phi_{\tilde{k}}$ for $1 \leq \tilde{k} \leq K$ than to all other original values ϕ_k for $k = 1, 2, \ldots, K$, $k \neq \tilde{k}$. If we now match several $\tilde{\phi}_j$ to the same $\phi_{\tilde{k}}$ and accordingly match several $\tilde{\eta}_j$ to the same $\eta_{\tilde{k}}$, this can result in an unreasonably large error $\Delta\eta$, which does not reflect how well the parameters were actually reconstructed. We therefore decided to match the parameters $\tilde{\eta}_k$ and η_k as well as $\tilde{\phi}_k$ and ϕ_k for $k = 1, 2, \ldots, K$ separately as in (7.2). Moreover, since we choose the parameters η_k randomly, it is extremely unlikely, that two or more of them are equal, which means that usually it should be possible to match them uniquely. We also looked at how often it happens that the parameters $\tilde{\phi}_k$ and ϕ_k for $k = 1, 2, \ldots, K$ are not matched uniquely and found that this happens very rarely. Generally, in cases where the error $\Delta\phi$ was overall larger, it happened more often that there was no unique matching. This observation is not very surprising. However, even in the worst cases we observed, it happened less than 2% of the time that there was no unique matching. Even more, we computed the error $\Delta\eta$ both using the matching described in this paragraph as well as (7.2). We found that although in the cases where the matching of $\tilde{\phi}_k$ and ϕ_k was not unique the method described here produced larger maximum errors, and hence a large mean error, the median error over all $\Delta\eta$ was always the same as when using (7.2).

Recall that in Chapter 2 we found that for the error stopping criterion of the AAA algorithm a relative tolerance with the factor 10^{-13} worked best. However, in our experiments, we found that this tolerance is not ideal here. Instead, a relative tolerance with the factor 10^{-10} provided the best outcome. Moreover, while, in the setting of Chapter 2, the Froissart doublet removal process worked best if we chose a relative tolerance with the factor 10^{-5} in order to remove elements corresponding to small weights from the column array, here, a factor of 10^{-8} proved to be better.

In this section, we will compare the MPM algorithm with the ESPIRA algorithm in two variants. These variants refer to the stopping criterion employed in the AAA algorithm. ESPIRA (err) or (error) denotes the variant with the error stopping criterion (2.1) and ESPIRA (rank) the variant with the rank stopping criterion (2.2).

7.2.1. Exact Data

We will proceed in this section as follows. We start with a given set of parameters and then vary one parameter at a time and observe the effects. We found in our tests that varying the size of the parameters η_k for $k = 1, 2, \ldots, K$ does not seem to have much of an effect at all. Therefore, we will not consider this parameter any further. For all experiments reported here, we fix $r_{\mathrm{Re}\,\eta} = r_{\mathrm{Im}\,\eta} = 10$. As in Chapter 2, when looking at the reconstruction errors, we always only consider data from functions where the number K of components was detected correctly.

We first consider the effect of increasing the number K of components in the exponential sum. Our starting point is given by the parameters $K = 5$, $P = 5$, $r_{\mathrm{Im}\,\phi} = 2\pi\cdot 10$ and $r_{\mathrm{Re}\,\phi} = 0$, i.e., $\mathrm{Re}\,\phi_k = 0$ for all $k = 1, 2, \ldots, K$. In Table 7.1, we report the results for these parameters.

Method	K	Par	Mean	Median	Minimum	Maximum
ESPIRA (err)	1000	$\Delta\eta$	5.3457e−11	5.3452e−14	9.3725e−15	3.7613e−08
		$\Delta\phi$	1.1079e−14	3.6694e−16	1.1549e−16	3.4706e−12
		Δf	7.4256e−14	3.8495e−14	1.1745e−14	3.0916e−12
ESPIRA (rank)	1000	$\Delta\eta$	5.3457e−11	5.3452e−14	9.3725e−15	3.7613e−08
		$\Delta\phi$	1.1079e−14	3.6694e−16	1.1549e−16	3.4706e−12
		Δf	7.4256e−14	3.8495e−14	1.1745e−14	3.0916e−12
MPM	970	$\Delta\eta$	7.3958e−11	5.4662e−14	1.0082e−14	2.8989e−08
		$\Delta\phi$	2.0052e−14	5.5091e−16	1.4683e−16	4.1585e−12
		Δf	3.9108e−14	3.6789e−14	1.4372e−14	1.0534e−13

Table 7.1.: Number of times the number K of components is found correctly as well as mean, median, minimum, and maximum of the relative errors for $K = 5$, $P = 5$, $r_{\mathrm{Re}\,\eta} = r_{\mathrm{Im}\,\eta} = 10$, $r_{\mathrm{Im}\,\phi} = 2 \cdot \pi \cdot 10$ and $r_{\mathrm{Re}\,\phi} = 0$.

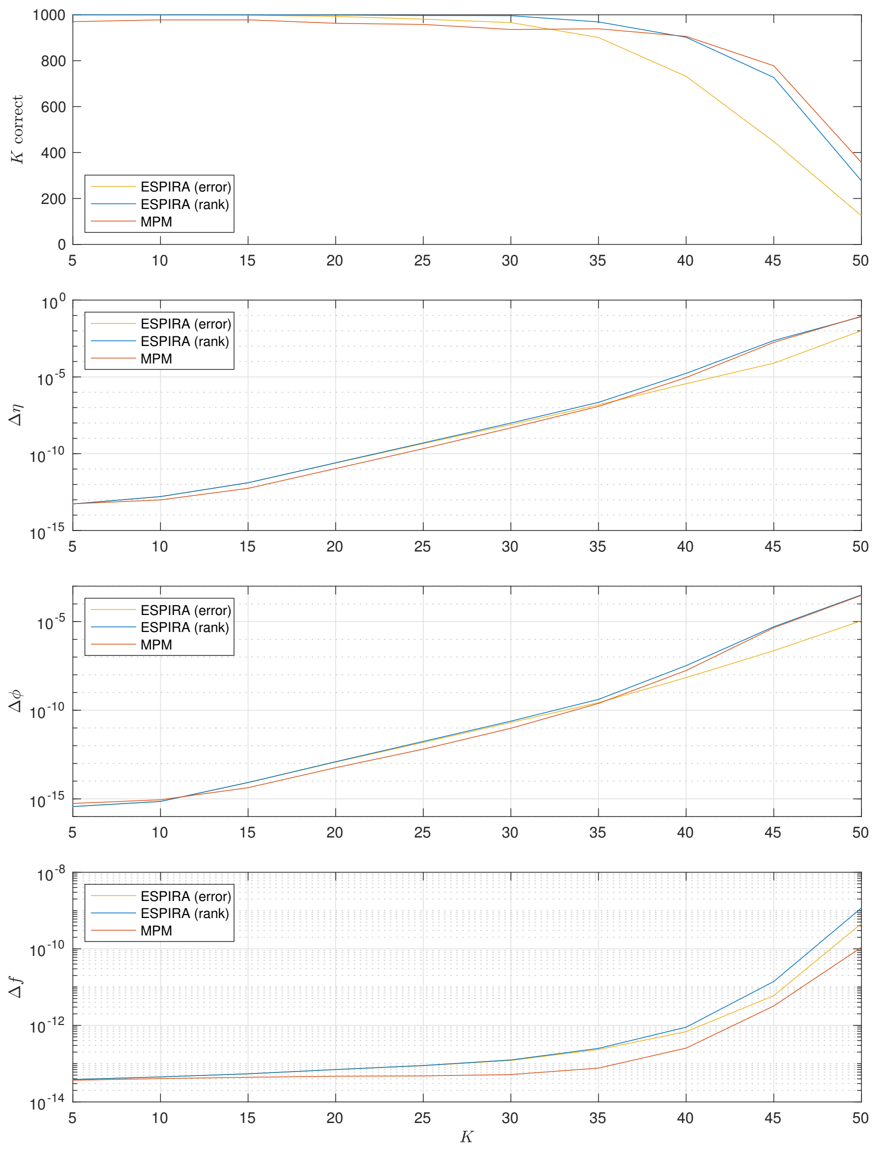

Figure 7.1.: Number of times the number K of components is detected correctly as well as the median of $\Delta\eta$, $\Delta\phi$ and Δf for increasing K with $P = 5$, $r_{\mathrm{Re}\,\eta} = r_{\mathrm{Im}\,\eta} = 10$, $r_{\mathrm{Im}\,\phi} = 2 \cdot \pi \cdot 10$ and $r_{\mathrm{Re}\,\phi} = 0$.

As we can see, the reconstruction errors for the functions as well as the coefficient and exponent values are very similar for all methods. However, while both ESPIRA variants detect the number of components correctly every time, MPM does so only for 97% of the samples. Figure 7.1 shows how the algorithms perform for increasing K.

Regarding the errors, all three algorithms perform very similarly. For $\Delta\eta$ and $\Delta\phi$, ESPIRA (error) seems to be slightly better for large K. However, this might just be due to the fact that for large K this algorithm finds the correct number of components far less often than the other two algorithms, meaning that the presented data is obtained from far fewer samples. Moreover, we would like to note that for $K = 50$ the Cauchy matrix in step (v) of Algorithm 7.2 with the error stopping criterion had an `Inf` entry for two samples where the number of components was detected correctly. This means that the parameters η_k for $k = 1, 2, \ldots, K$ could not be recovered. We therefore manually removed these samples before computing the errors.

Looking at Δf, we see that MPM outperforms both ESPIRA variants. Especially for large K, it is up to an order of magnitude better.

Now, let us consider how often the number of components is found correctly. For $K \leq 35$, all algorithms perform relatively reliably with ESPIRA (rank) performing best. In fact, for K from 5 to 30 it finds the correct number of components between 996 and 1000 times. For $K > 35$, all algorithms quickly become less accurate. Especially ESPIRA (error) performs poorly here. For K larger than 40, MPM again performs better than ESPIRA (rank).

We want to remark here that $K = 50$ means that the rational function which we use in ESPIRA to reconstruct the function parameters has degree 50. Recall that we have seen in Chapter 2 that the AAA algorithm does not find the correct degree for more than 25% of the samples already at degree 9 and with 500 added sample points. This is roughly the same performance as ESPIRA (rank) delivers for $K = 45$. This observation becomes even more impressive if we consider that we use $M = \max\left\{2K + 2, 2\pi \cdot 10 \cdot \frac{5}{\pi} + 1\right\}$ samples. In other words, for $K = 45$ we take 101 samples, which is just 9 more than the minimal number of samples we need to take in order to be able to solve the rational interpolation problem in the ESPIRA algorithm. For $K = 50$, we even only take $2K + 2 = 102$ samples. This shows that the structure inherent to the DFT data we use for rational interpolation greatly stabilizes the interpolation process.

One would now naturally assume that adding more samples would increase the performance, at least of the ESPIRA algorithms. However, we tried doing this and although the performance increased, it did so only very slightly. This is most likely due to the fact that adding more sample points while keeping P constant, means that the sampling distance decreases, potentially increasing the condition numbers of the involved Löwner matrices, which then offsets the advantage gained by having more sample points. We find a much larger effect by increasing P, which, by equation (7.1), in this case also increases the number of samples. Before we show the effect of increasing P in Figure 7.2, we give a detailed overview over the data for $K = 50$ in Table 7.2.

Method	K	Par	Mean	Median	Minimum	Maximum
ESPIRA (err)	125	$\Delta\eta$	9.0638e−02	1.0177e−02	1.1846e−08	6.6889e−01
		$\Delta\phi$	7.5508e−02	1.1149e−05	4.3994e−11	1.1691e+00
		Δf	1.4705e−03	4.5658e−10	2.7860e−13	1.4142e−01
ESPIRA (rank)	277	$\Delta\eta$	1.5979e−01	8.3435e−02	1.1846e−08	1.0281e+00
		$\Delta\phi$	1.1020e−02	3.2485e−04	4.3994e−11	6.3159e−01
		Δf	3.3009e−07	1.1650e−09	2.7860e−13	1.8318e−05
MPM	357	$\Delta\eta$	1.6004e−01	8.8952e−02	2.4797e−09	1.5380e+00
		$\Delta\phi$	1.4878e−02	3.0459e−04	9.3963e−12	3.9709e−01
		Δf	1.6989e−02	1.0950e−10	6.6280e−14	1.0000e+00

Table 7.2.: Number of times the number K of components is found correctly as well as mean, median, minimum, and maximum of the relative errors for $K = 50$, $P = 5$, $r_{\mathrm{Re}\,\eta} = r_{\mathrm{Im}\,\eta} = 10$, $r_{\mathrm{Im}\,\phi} = 2 \cdot \pi \cdot 10$, and $r_{\mathrm{Re}\,\phi} = 0$. For ESPIRA (error) only 123 samples were considered for computing the errors (see text).

First, let us note that we saw similar effects as shown in Figure 7.2 for smaller values of K. However, since for smaller K the performance of the algorithms is already quite good, the improvement was not as impressive as for the displayed data. We start by looking at how the number of correctly detected K develops. For both ESPIRA variants, the performance increases drastically. However, while ESPIRA (error) stagnates at about 930 from $P = 9$ onward, ESPIRA (rank) finds the correct K between 994 and 999 times for P larger than 10 and 985 times for $P = 9$.

Something curious happens with the MPM algorithm. Until $P = 9$, it is slightly better than ESPIRA (rank), with the exception of $P = 8$, but for $P = 10$ its performance drops dramatically. For $P = 11$, the performance is then comparable to ESPIRA (error) before there is another drop for $P = 12$. Finally, the algorithm seems to stabilize at around 960. We are not sure why this happens, but we observed that

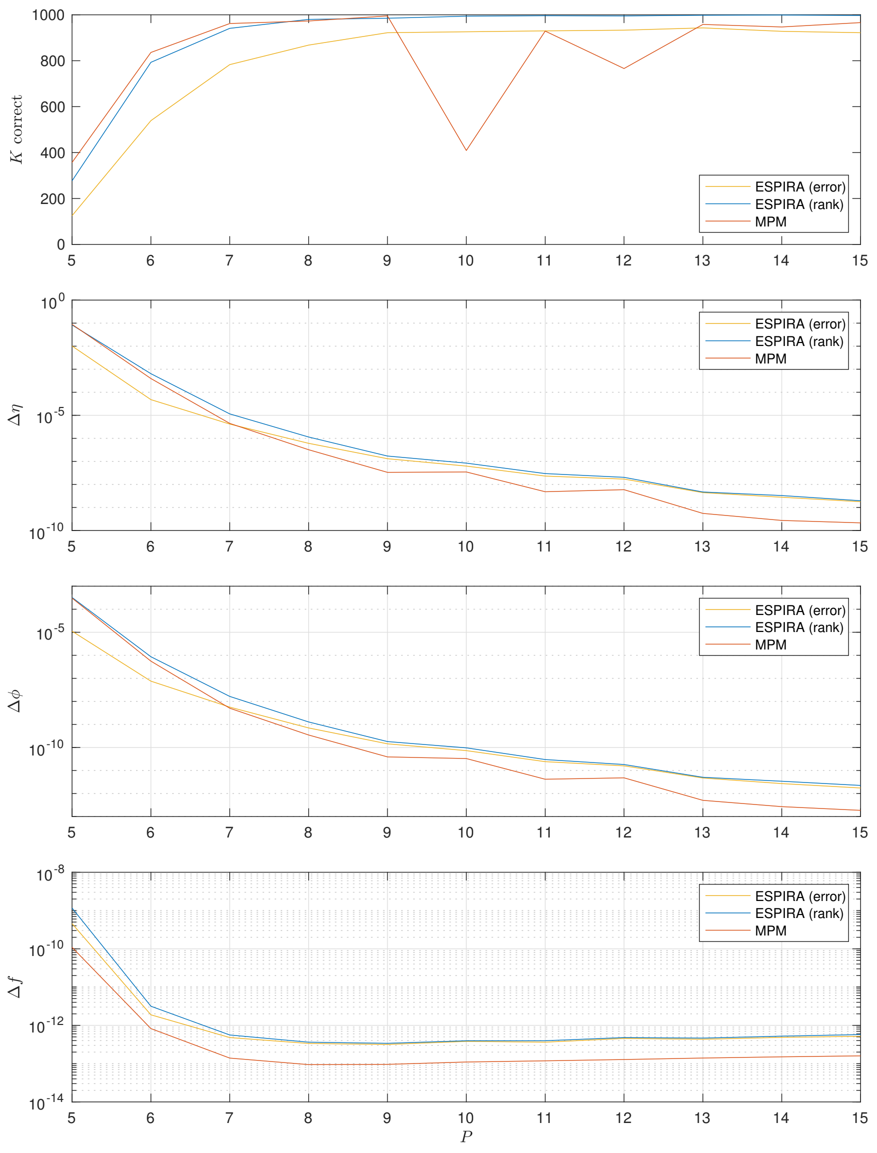

Figure 7.2.: Number of times the number K of components is detected correctly as well as the median for $\Delta\eta$, $\Delta\phi$ and Δf for increasing P with $K = 50$, $r_{\mathrm{Re}\,\eta} = r_{\mathrm{Im}\,\eta} = 10$, $r_{\mathrm{Im}\,\phi} = 2 \cdot \pi \cdot 10$ and $r_{\mathrm{Re}\,\phi} = 0$.

MPM performs very poorly for $P = 10$, regardless of K or $r_{\mathrm{Im}\,\phi}$. For $P = 12$, we saw that MPM performs poorly regardless of K, but, changing $r_{\mathrm{Im}\,\phi}$, we could not observe this effect again. This strange behavior is something we could observe often in our experiments. While the ESPIRA algorithms behaved very consistently and as one would expect, as seen here by finding the correct K more often for larger P, the MPM algorithm often displayed unexpected results for particular parameter values and only those particular values. In this sense, we believe that the ESPIRA algorithms and in particular the algorithm with the rank stopping criterion are more robust than MPM.

Looking at the errors we see that, as we would expect, all errors go down when we increase P. However, there are a few more observations to make. First, note that considering Δf, the error first drops considerably, but the minimal error is actually attained at $P = 8$ for MPM respectively $P = 9$ for both ESPIRA variants. After that, the errors get just slightly worse with increasing P. This is despite the fact that $\Delta\eta$ and $\Delta\phi$ keep decreasing. This observation gives us a first hint that the problem of parameter identification, i.e., finding good approximations for the parameters ϕ_k and η_k for $k = 1, 2, \ldots, K$ as opposed to only finding a good approximation for f, is ill-posed. Another hint can be found if we compare the errors of ESPIRA (error) to MPM. While for $P \leq 7$ the errors $\Delta\eta$ and $\Delta\phi$ for ESPIRA are smaller than or very comparable to the errors of MPM, the error Δf is clearly smaller for MPM.

In Table 7.3 we present the detailed data for $P = 15$.

Method	K	Par	Mean	Median	Minimum	Maximum
ESPIRA (err)	922	$\Delta\eta$	6.9532e−04	1.7736e−09	5.5502e−13	3.3064e−01
		$\Delta\phi$	1.1959e−07	1.7395e−12	1.1668e−15	9.0424e−05
		Δf	3.8323e−12	5.1777e−13	1.1613e−13	2.3920e−09
ESPIRA (rank)	997	$\Delta\eta$	1.4956e−03	1.9566e−09	5.5502e−13	3.3064e−01
		$\Delta\phi$	1.3655e−03	2.2461e−12	1.1668e−15	1.0290e+00
		Δf	1.9941e−04	5.7438e−13	1.1613e−13	1.3387e−01
MPM	966	$\Delta\eta$	1.3115e−03	2.1402e−10	2.8343e−13	6.2775e−01
		$\Delta\phi$	5.7118e−07	1.8611e−13	1.0367e−15	4.9673e−04
		Δf	1.6519e−13	1.5955e−13	8.8891e−14	1.7844e−12

Table 7.3.: Number of times the number K of components is found correctly as well as mean, median, minimum, and maximum of the relative errors for $K = 50$, $P = 15$, $r_{\mathrm{Re}\,\eta} = r_{\mathrm{Im}\,\eta} = 10$, $r_{\mathrm{Im}\,\phi} = 2 \cdot \pi \cdot 10$ and $r_{\mathrm{Re}\,\phi} = 0$.

Let us now look at the effect of changing $r_{\operatorname{Im}\phi}$. It is hard to quantify this isolated. Generally, if $r_{\operatorname{Im}\phi}$ is small then P needs to be large in order to reconstruct the given function and its parameters. Heuristically, this can be explained as follows. If $r_{\operatorname{Im}\phi}$ is small, then the frequencies of all components of the function are small, which means that their period is large. If now P is small as well, then any component performs just a few complete cycles, if any at all, in $[0, P)$. This of course makes it harder to detect the components. The number of components K also plays an important role here. The larger K, the larger P has to be. We found in our experiments that the impact of increasing K is severe, as can be seen in Table 7.4.

As we have seen already, given K and $r_{\operatorname{Im}\phi}$, increasing P will decrease the reconstruction errors. Since this is nothing new, we will not consider errors here. Instead, we fix K and decrease $r_{\operatorname{Im}\phi}$. For any $r_{\operatorname{Im}\phi}$, we report the smallest $P \in \mathbb{N}$ such that K is detected correctly at least once. This data is shown in Table 7.4.

		P		
K	$r_{\operatorname{Im}\phi}$	ESPIRA (err)	ESPIRA (rank)	MPM
10	$2\pi \cdot 10$	1	1	1
	$2\pi \cdot 9$	1	1	1
	$2\pi \cdot 8$	1	1	1
	$2\pi \cdot 7$	1	1	1
	$2\pi \cdot 6$	1	1	1
	$2\pi \cdot 5$	1	1	1
	$2\pi \cdot 4$	1	1	1
	$2\pi \cdot 3$	1	1	1
	$2\pi \cdot 2$	2	1	1
	$2\pi \cdot 1$	3	2	2
25	$2\pi \cdot 10$	2	2	2
	$2\pi \cdot 9$	2	2	2
	$2\pi \cdot 8$	3	2	2
	$2\pi \cdot 7$	3	3	3
	$2\pi \cdot 6$	4	3	3
	$2\pi \cdot 5$	4	4	4
	$2\pi \cdot 4$	5	4	4
	$2\pi \cdot 3$	7	6	6
	$2\pi \cdot 2$	10	8	8
	$2\pi \cdot 1$	19	16	16

Table 7.4.: Number of times the number K of components is found correctly for $r_{\operatorname{Re}\eta} = r_{\operatorname{Im}\eta} = 10$ and $r_{\operatorname{Re}\phi} = 0$, where $P \in \mathbb{N}$ is the smalles number such that the respective algorithm finds K correctly at least one time.

Since the test data we used so far was generated randomly, it is highly unlikely that any of the test functions contained P-periodic components. In order to test how the algorithms work in that case, we therefore deliberately add P-periodic components to the functions. Before we show the results, we need to discuss one special aspect. Recall that in step (ii) of Algorithm 7.2, we checked if the input data stems from a P-periodic function, i.e., one for which all components are P-periodic. This was done in order to reconstruct the components directly from the DFT data. However, since in practice it is unlikely that a sufficiently large number of DFT coefficients is actually 0, we have to choose a threshold for when to regard a DFT coefficient as being 0. In our experiments, we found that a threshold of 10^{-8} is large enough to reliably identify P-periodic functions and small enough not to mistake non-P-periodic functions for being P-periodic. Nonetheless, for the following data, we look at how ESPIRA handles periodic functions directly. It turns out that if the input functions are P-periodic, performing ESPIRA (rank) on them gives comparable results to reconstructing the function parameters directly from the DFT data. Unfortunately, this is not true for ESPIRA (error). We will demonstrate this in Table 7.5. Figures 7.3 – 7.6 show how the algorithms perform for different numbers of P-periodic components and different values of K.

Let us start by looking at how often the number K of components is found correctly. Here, ESPIRA (rank) delivers an impressive performance. For $K = 10$, it finds the correct K every single time. Similarly, for $K = 25$, it finds the correct K every time except for the purely periodic functions, i.e., functions with 25 P-periodic components, where it finds the correct K 999 times. MPM performs not quite as well here. For $K = 10$, it finds the correct K on average 993 times and just 955 times for $K = 25$. For $K = 50$, MPM finds the correct number of components on average 980 times, while ESPIRA (rank) averages 997 times. Now, let us turn to ESPIRA (error). It is clear from looking at Figure 7.3 that it performs very poorly regardless of K. For $K = 50$ and P-periodic functions, i.e., functions with 50 P-periodic components, it does not find the correct number of components a single time. This is an interesting observation since, looking back at Table 2.14, we found before that the AAA algorithm with the error stopping criterion performs slightly better than the AAA algorithm with the rank stopping criterion for smaller numbers of unattainable points and just slightly worse for larger numbers of unattainable points. Here, on the other hand, the Froissart doublet removal process for the error stopping criterion seems to perform extremely poorly. We tried to improve these results by experimenting with different parameters

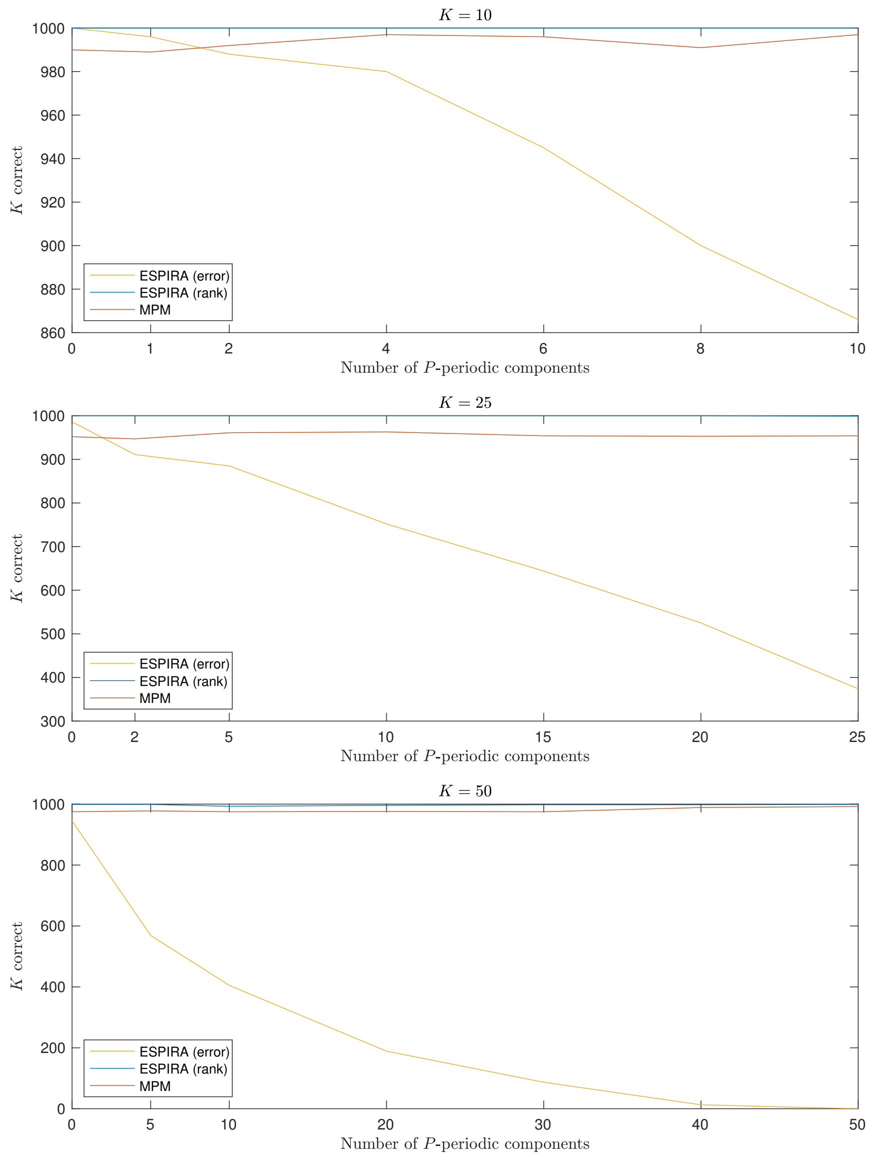

Figure 7.3.: Number of times the number K of components is correctly detected for different values of K and increasing number of P-periodic components with $P = 15$, $r_{\mathrm{Re}\,\eta} = r_{\mathrm{Im}\,\eta} = 10$, $r_{\mathrm{Im}\,\phi} = 2 \cdot \pi \cdot 10$, and $r_{\mathrm{Re}\,\phi} = 0$.

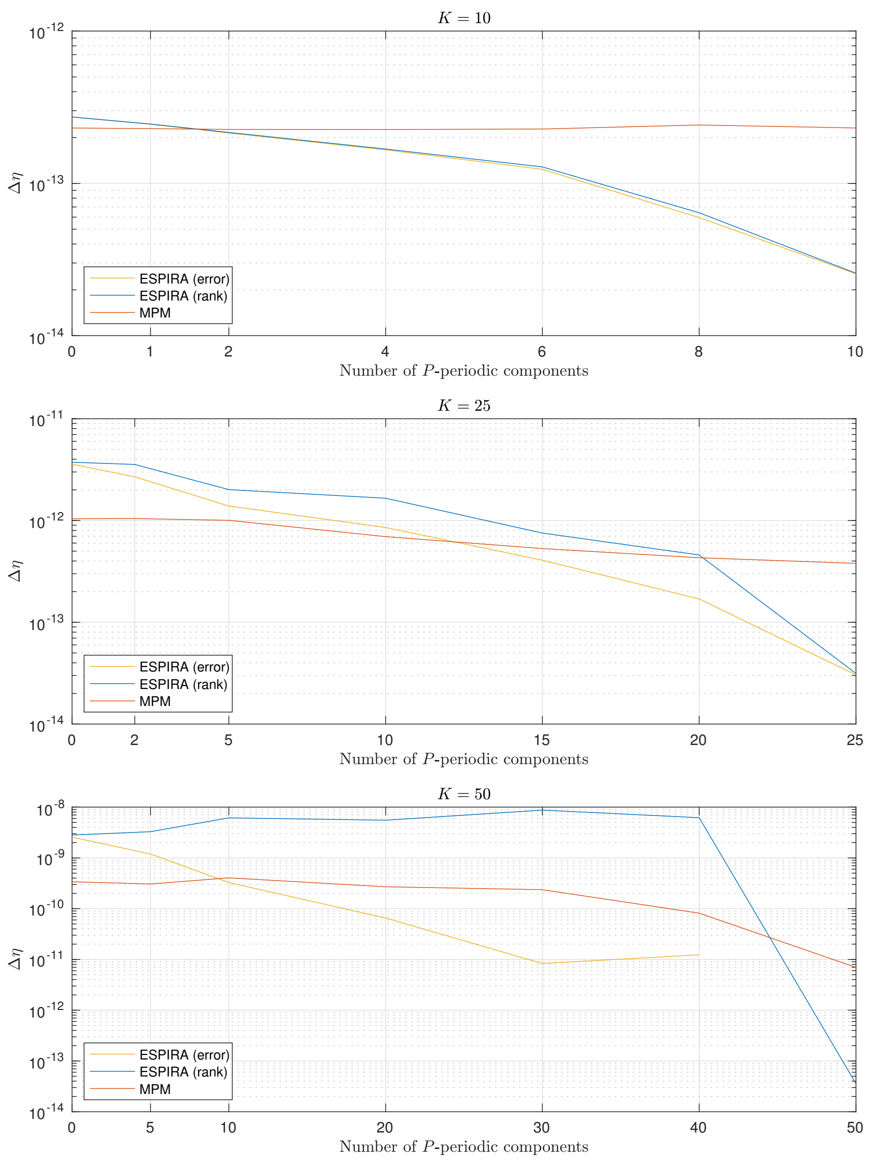

Figure 7.4.: Median of $\Delta\eta$ for different values of K and increasing number of P-periodic components with $P = 15$, $r_{\mathrm{Re}\,\eta} = r_{\mathrm{Im}\,\eta} = 10$, $r_{\mathrm{Im}\,\phi} = 2 \cdot \pi \cdot 10$, and $r_{\mathrm{Re}\,\phi} = 0$.

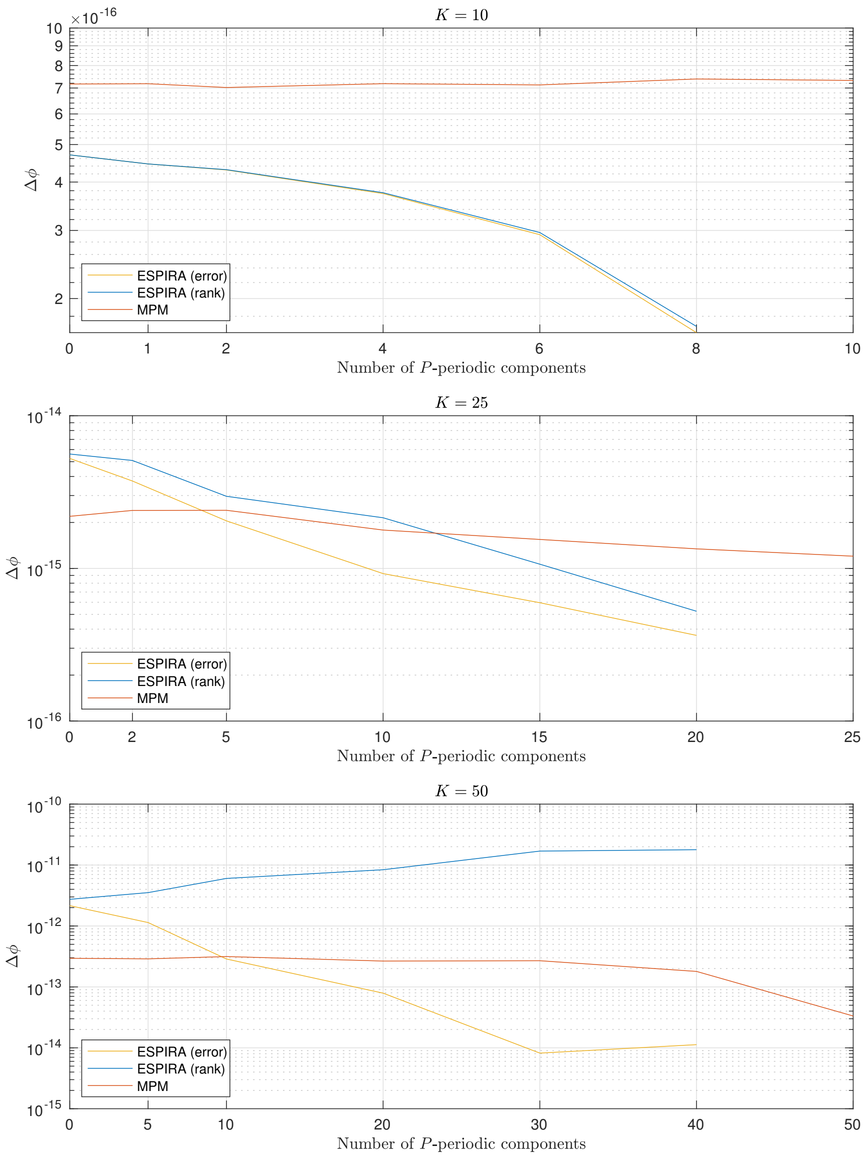

Figure 7.5.: Median of $\Delta\phi$ for different values of K and increasing number of P-periodic components with $P = 15$, $r_{\mathrm{Re}\,\eta} = r_{\mathrm{Im}\,\eta} = 10$, $r_{\mathrm{Im}\,\phi} = 2 \cdot \pi \cdot 10$, and $r_{\mathrm{Re}\,\phi} = 0$.

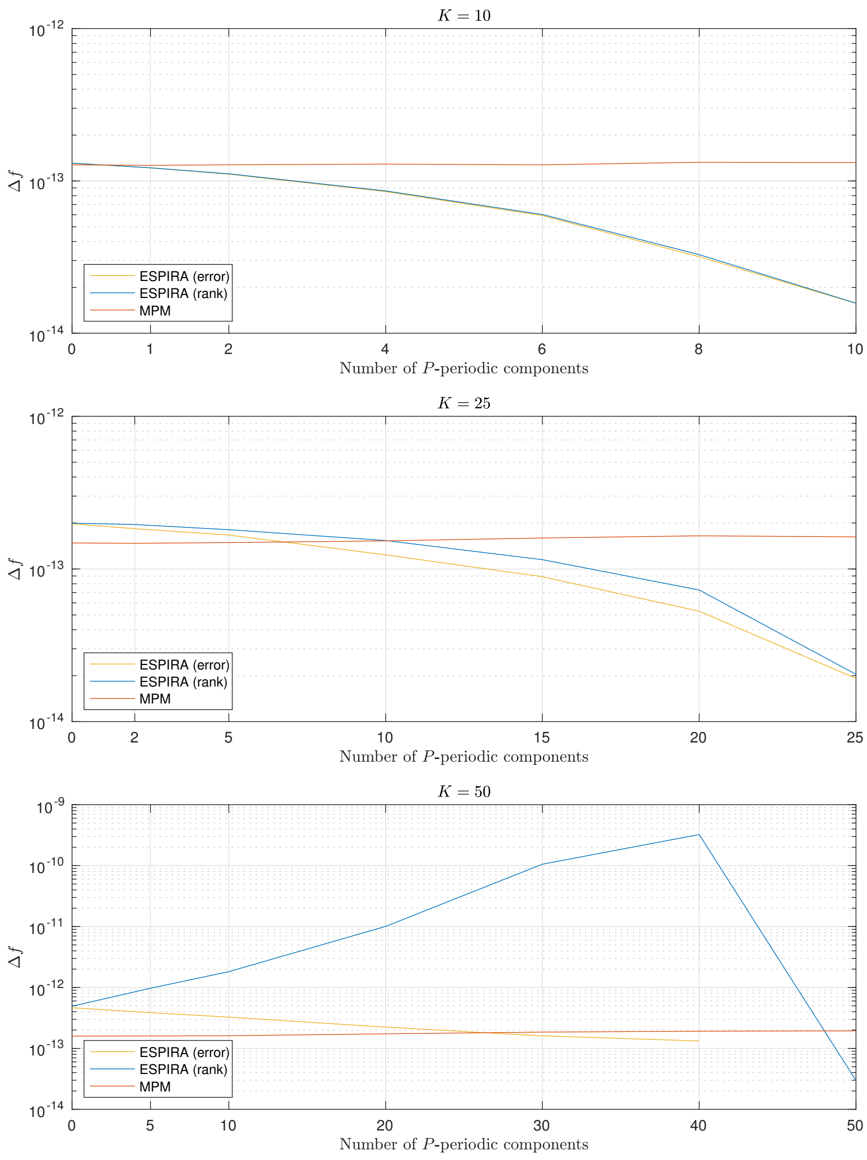

Figure 7.6.: Median of Δf for increasing number of P-periodic components with $P = 15$, $r_{\mathrm{Re}\,\eta} = r_{\mathrm{Im}\,\eta} = 10$, $r_{\mathrm{Im}\,\phi} = 2 \cdot \pi \cdot 10$, and $r_{\mathrm{Re}\,\phi} = 0$.

for the Froissart doublet removal process but with no success. We therefore assume that this process is unsuitable for the data in the given setting.

We now turn to $\Delta\eta$. First, we note that this error is almost constant for the MPM algorithm. Only for $K = 50$, there is a considerable improvement for very large numbers of periodic components. Both ESPIRA variants behave almost identically for $K = 10$, improving considerably for a growing number of P-periodic components. We can also observe this trend for $K = 25$, where ESPIRA (error) performs clearly better. However, always keep in mind that we only compute the errors for those functions where K is detected correctly. Since this number is small for ESPIRA (error) and $K = 25, 50$, it is hard to assess how meaningful the results for the errors are. We will therefore not comment on the performance of ESPIRA (error) for $K = 50$ any further. Regarding ESPIRA (rank) for $K = 50$, we see that $\Delta\eta$ gets slightly worse for an increasing number of P-periodic components, but is very small if the given functions are completely P-periodic. We will look at this case in more detail in Table 7.5. Comparing MPM and ESPIRA (rank), the difference in the errors becomes as large as almost two orders of magnitude. This behavior is interesting since it is in contrast to what we observed for smaller K, where the errors decreased for more P-periodic components. Unfortunately, we do not have a good explanation for why this happens.

Looking at $\Delta\phi$, we observe the same behavior as for $\Delta\eta$. We would just like to note the fact that for the ESPIRA algorithms no data is displayed for P-periodic functions for $K = 10, 25$, which is due to the errors being 0. For $K = 50$ and ESPIRA (rank), this is also because the error is 0, while for ESPIRA (error), this is because it did not detect K correctly at all, see Table 7.5. In any case, note that for $K = 10$, the observed errors are already close to machine precision.

For Δf, there is again nothing new to see. It is remarkable that for MPM the error is even closer to being constant than for $\Delta\eta$ and $\Delta\phi$. For ESPIRA (rank) and $K = 50$, on the other hand, the error grows even faster than the errors for the parameters, such that for 40 P-periodic components it is over three orders of magnitude worse than the error for MPM. Moreover, as we have seen similarly before, for $K = 50$ and 20 P-periodic components, there was one sample where K was detected correctly but the Cauchy matrix in step (v) of Algorithm 7.2 with the rank stopping criterion had an `Inf` entry so that we had to remove this sample manually.

We now look closer at the case that the input functions are indeed P-periodic. In Table 7.5, we compare the two ESPIRA variants to recovering the function parameters

K	Method	K corr	Par	Mean	Median	Minimum	Maximum
10	error	866	$\Delta\eta$	2.6280e−14	2.5479e−14	6.2091e−15	5.6733e−14
			$\Delta\phi$	0	0	0	0
			Δf	1.6851e−14	1.5708e−14	4.4380e−15	5.3230e−14
	rank	1000	$\Delta\eta$	2.6315e−14	2.5695e−14	6.1128e−15	5.6733e−14
			$\Delta\phi$	0	0	0	0
			Δf	1.6884e−14	1.5761e−14	5.2525e−15	5.3230e−14
	DFT	1000	$\Delta\eta$	2.5416e−14	2.4362e−14	6.1612e−15	5.4046e−14
			$\Delta\phi$	0	0	0	0
			Δf	1.4293e−14	1.3731e−14	4.3651e−15	3.0222e−14
25	error	374	$\Delta\eta$	3.0665e−14	3.0402e−14	1.4206e−14	5.9945e−14
			$\Delta\phi$	0	0	0	0
			Δf	2.2387e−14	1.9203e−14	7.3561e−15	7.1195e−14
	rank	999	$\Delta\eta$	3.1722e−14	3.1394e−14	1.1282e−14	6.1643e−14
			$\Delta\phi$	0	0	0	0
			Δf	2.3493e−14	2.0421e−14	7.3561e−15	8.7886e−14
	DFT	1000	$\Delta\eta$	3.0225e−14	2.9935e−14	1.1541e−14	6.1448e−14
			$\Delta\phi$	0	0	0	0
			Δf	1.6931e−14	1.6675e−14	7.2538e−15	2.8760e−14
50	error	0	-	-	-	-	-
	rank	999	$\Delta\eta$	1.2655e−06	3.5681e−14	1.8061e−14	1.2642e−03
			$\Delta\phi$	8.1986e−19	0	0	8.1904e−16
			Δf	1.0819e−07	2.9782e−14	9.5305e−15	1.0808e−04
	DFT	1000	$\Delta\eta$	3.3995e−14	3.3542e−14	1.6893e−14	6.4420e−14
			$\Delta\phi$	0	0	0	0
			Δf	2.0487e−14	2.0187e−14	9.7446e−15	3.4707e−14

Table 7.5.: Comparison of the two ESPIRA variants with direct reconstruction of the parameters from DFT data for P-periodic functions. Number of times the number K of components is found correctly as well as mean, median, minimum, and maximum of the relative errors for $P = 15$, $r_{\mathrm{Re}\,\eta} = r_{\mathrm{Im}\,\eta} = 10$, $r_{\mathrm{Im}\,\phi} = 2 \cdot \pi \cdot 10$, and $r_{\mathrm{Re}\,\phi} = 0$.

directly from the DFT data, using a threshold of 10^{-8} in order to detect if the input data stems from a P-periodic function. Ignoring the fact that ESPIRA (error) performs poorly when it comes to detecting the correct number of components, there is hardly any difference between ESPIRA and directly recovering the parameters for $K = 10, 25$. In particular, the phase parameters ϕ_k for $k = 1, 2, \ldots, K$ are recovered with machine precision. Only for $K = 50$, where ESPIRA (error) fails completely, we find that, with the exception of $\Delta\phi$, ESPIRA (rank) produces larger maximum errors than the direct variant. This means that even if we leave out step (ii) of Algorithm 7.2, we still get good results for P-periodic functions, at least for ESPIRA (rank).

Until now, we have actually just looked at (almost) periodic functions. However, our algorithm as well as MPM were actually developed to deal with exponential sums. Therefore, we now consider functions where $r_{\mathrm{Re}\,\phi} \neq 0$ and show our findings in Figure 7.7.

For $r_{\mathrm{Re}\,\phi} \leq 5$, both ESPIRA variants perform very well. Especially ESPIRA (rank) finds the correct K all the time until $r_{\mathrm{Re}\,\phi} = 6$. However, from $r_{\mathrm{Re}\,\phi} = 6$ onward, the performance of ESPIRA (error) drops again dramatically. Comparing MPM and ESPIRA (rank), we see that the latter algorithm performs better until $r_{\mathrm{Re}\,\phi} = 8$. In particular, within this range, MPM finds the correct K on average 962 times, while ESPIRA (rank) does so 997 times.

Regarding $\Delta\eta$ and $\Delta\phi$, we see a similar behavior as before, where the errors for MPM are slightly better than for both ESPIRA variants. Again, the relatively small errors for ESPIRA (error) and large $r_{\mathrm{Re}\,\phi}$ are most likely only due to the fact that the errors are computed for relatively few samples. What is interesting though, is that the errors decrease at first. Also, note that the impact of increasing $r_{\mathrm{Re}\,\phi}$ on the errors is far larger than what we observed for the other parameters. Increasing $r_{\mathrm{Re}\,\phi}$ by 1 increases the error by roughly an order of magnitude.

Finally, let us look at Δf. Interestingly, it is almost constant and almost the same for all algorithms. The fact that Δf is roughly constant, while both $\Delta\eta$ and $\Delta\phi$ increase drastically for growing $r_{\mathrm{Re}\,\phi}$ shows once more that the parameter identification problem is ill-posed.

Before we consider noisy data, we show three specific examples. Here, we will only use ESPIRA (rank) without spelling it out every time. In order to specify the given functions, we use parameter vectors $\boldsymbol{\eta} := (\eta_1, \eta_2, \ldots, \eta_K)$ and $\boldsymbol{\phi} := (\phi_1, \phi_2, \ldots, \phi_K)$ for $K \in \mathbb{N}$.

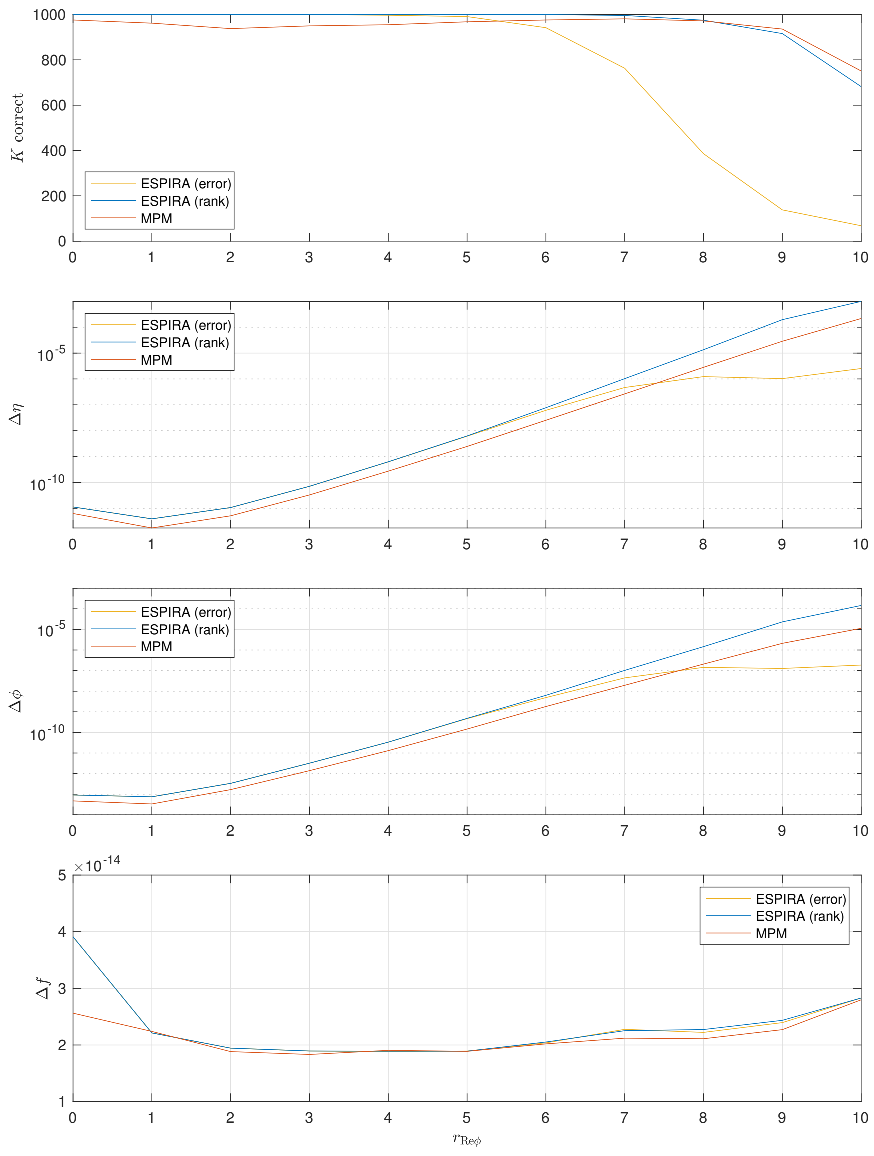

Figure 7.7.: Number of times the number K of components is correctly detected as well as the median for $\Delta\eta$, $\Delta\phi$, and Δf for increasing $r_{\mathrm{Re}\,\phi}$ with $K = 15$, $P = 3$, $r_{\mathrm{Re}\,\eta} = r_{\mathrm{Im}\,\eta} = 10$, and $r_{\mathrm{Im}\,\phi} = 2 \cdot \pi \cdot 10$.

The first example serves two purposes, it demonstrates once more the ill-posed nature of the parameter identification problem and it provides a warning to be careful how to choose P when dealing with exponential sums. We consider the function given by

$$\begin{aligned} \boldsymbol{\eta} &= (3, 2, 1, 5) \qquad \text{and} \\ \boldsymbol{\phi} &= (5.6 + 2\pi\,\mathrm{i}\cdot 0.45, 2.3 + 2\pi\,\mathrm{i}\cdot 2.98, 0.7 + 2\pi\,\mathrm{i}\cdot 3, 2\pi\,\mathrm{i}\cdot 5). \end{aligned} \tag{7.3}$$

This function can be seen on the interval $[0, 1]$ in Figure 7.8. In Table 7.6, we show the reconstruction errors for $P = 1$. In this example, we again use the minimal necessary number M of samples to reconstruct f.

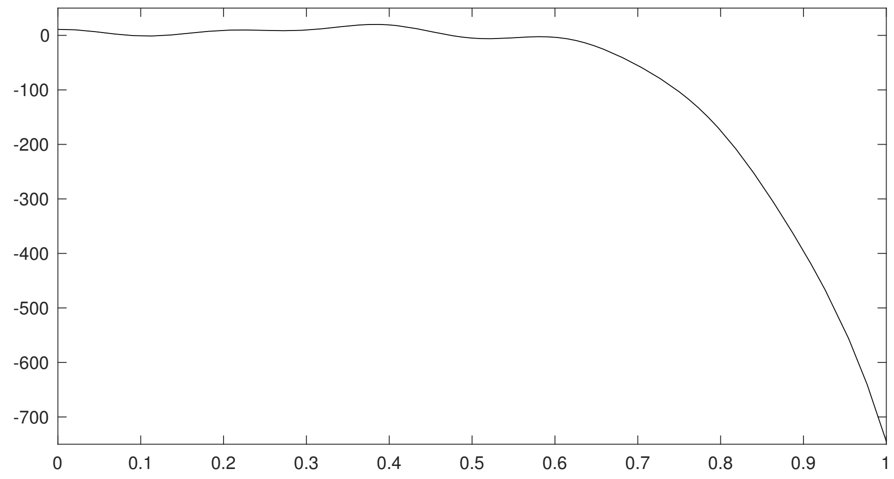

Figure 7.8.: Real part of the function specified by the parameters in (7.3) on the interval $[0, 1]$.

	$\Delta\eta$	$\Delta\phi$	Δf
ESPIRA	3.0354e−12	5.4264e−13	6.6006e−15
MPM	1.3159e−12	2.5666e−13	1.6572e−15

Table 7.6.: Reconstruction errors for the function specified by the parameters in (7.3) for $P = 1$.

The observed results are very much what we would expect from our observations so far. If we now increase P, we see that the errors are getting worse until for $P = 7$, both algorithms fail to find $K = 4$ correctly. Increasing P further, we find that both algorithms reconstruct the function using just one term. This one term is $3 \cdot \mathrm{e}^{(5.6+2\pi\,\mathrm{i}\cdot 0.45)\cdot t}$. We show the function on the interval $[0, 10]$ in Figure 7.9 and record the reconstruction errors for f as well as $\eta_1 = 3$ and $\phi_1 = 5.6 + 2\pi\,\mathrm{i}\cdot 0.45$ in Table 7.7.

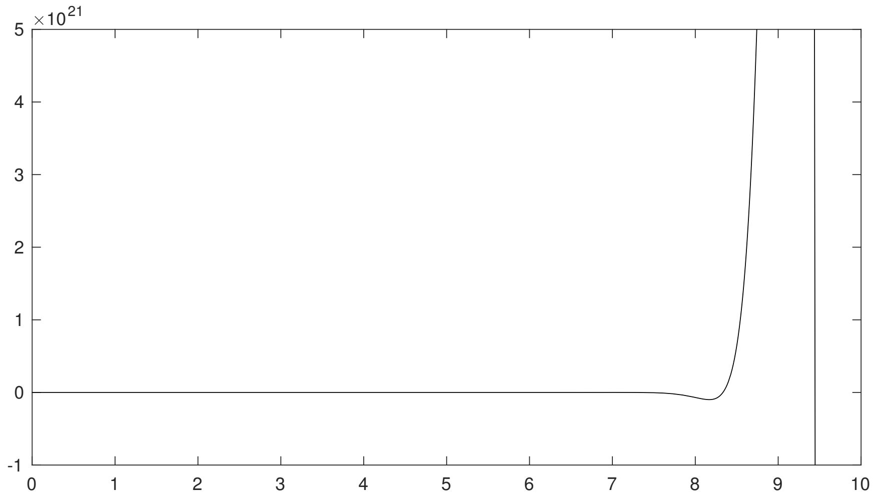

Figure 7.9.: Real part of the function specified by the parameters in (7.3) on the interval $[0, 10]$.

	$\Delta\eta_1$	$\Delta\phi_1$	Δf
ESPIRA	3.4303e−13	5.7897e−15	2.1110e−14
MPM	3.3413e−13	5.6384e−15	2.5088e−14

Table 7.7.: Reconstruction errors for the function specified by the parameters in (7.3) for $P = 10$.

In Table 7.7, we see that not just the parameters η_1 and ϕ_1 are reconstructed very well, but also f as a whole. Looking at Figure 7.9 gives us a hint at why this happens. Since $\operatorname{Re}\phi_1$ is larger than $\operatorname{Re}\phi_k$ for $k = 2, 3, 4$, the term $3 \cdot \mathrm{e}^{(5.6+2\pi\,\mathrm{i}\cdot 0.45)\cdot t}$ dominates the whole function f for large values of t. Moreover, for large P, we also have that $\max_{t\in[0,P]} |f(t)|$ becomes very large, which means that the relative error Δf is likely to become very small. This once more shows that the parameter identification problem is ill-posed. Even though we were only able to reconstruct one component of f, this was enough to get a very good reconstruction of f on the interval $[0, 10]$.

The following two examples will look at clustered frequencies, i.e., frequencies which lie very close to each other. The first example is given by the parameters

$$\begin{aligned}\boldsymbol{\eta} &= (6,5,1,4,5,-1,3,2,3,-4) \qquad \text{and}\\ \boldsymbol{\phi} &= \frac{2\pi\,\mathrm{i}}{1000}\cdot(7,21,51,52,53,200,201,202,543,894).\end{aligned} \tag{7.4}$$

This function has two frequency clusters, one around 52 and one around 201. Having frequencies which lie so close together, we need to sample the function over a relatively large interval $[0,P]$ in order to be able to reconstruct the parameters. Here, we choose $P = 25$ and again M minimal. Table 7.8 shows the results of the reconstruction.

	$\Delta\eta$	$\Delta\phi$	Δf
ESPIRA	1.8934e−02	4.1358e−05	2.3022e−12
MPM	6.5265e−02	1.3728e−04	6.0355e−14

Table 7.8.: Reconstruction errors for the function specified by the parameters in (7.4) for $P = 25$.

Both algorithms reconstructed the function and its parameters well, where the errors for the function parameters are smaller for ESPIRA, while the function itself is reconstructed better by MPM.

The last example in this section is a function which has all frequencies clustered around 200. It is given by

$$\begin{aligned}\boldsymbol{\eta} &= (6,5,1,4,5,-1,3,2,3,-4) \qquad \text{and}\\ \boldsymbol{\phi} &= \frac{2\pi\,\mathrm{i}}{1000}\cdot(196,197,198,199,200,201,202,203,204,205).\end{aligned} \tag{7.5}$$

This time, since all frequencies lie very close together, we need to sample over a much larger interval. Table 7.9 presents the results for $P = 500$ and minimal M.

	$\Delta\eta$	$\Delta\phi$	Δf
ESPIRA	3.1895e−03	6.3496e−05	9.6560e−12
MPM	2.3926e−02	4.4102e−04	6.5838e−13

Table 7.9.: Reconstruction errors for the function specified by the parameters in (7.5) for $P = 500$.

As before, both algorithms perform well, with ESPIRA having smaller errors for the parameters and MPM having a smaller error Δf.

Let us briefly sum up our findings of this section. It has become clear that ESPIRA (error) definitely displays the weakest performance of the three algorithms considered here since it is often not able to find the correct number of components of a given function. ESPIRA (rank) and MPM, on the other hand, perform very similarly, where MPM usually produces smaller errors, while ESPIRA (rank) usually finds the correct number of components more often. We have seen in our experiments that it is not possible to say with certainty which algorithm is better in general since varying the given parameters only slightly can result in the one or the other algorithm delivering better results.

7.2.2. Noisy Data

Lastly, let us consider noisy data. We will not go into as much detail as in the previous section since, generally speaking, all the trends we observed there also hold in the noisy case.

Here, we have to provide the correct number of components K beforehand to the algorithms since neither ESPIRA nor MPM can detect the correct number of components only from noisy data. However, this change can easily be implemented. For ESPIRA, this also means that we do not consider the two different stopping criteria. In a sense, we use a third stopping criterion, namely to let the algorithm run until the rational function provided by the AAA algorithm has degree K. We also do not need to consider Froissart doublet removal, i.e., P-periodic components, in this case since it is highly unlikely that the rational function obtained from the AAA algorithm interpolates all, or even just most, of the given DFT samples from the noisy data. This in turn means that the AAA algorithm would interpret all, or most, of the samples which are not interpolated by the obtained rational function of degree K as Froissart doublets, which clearly makes no sense.

We add noise to the given samples as follows. For each function f in our experiment, let `F` denote the samples taken from f. We add normally distributed noise with mean 0 to the real and imaginary part of the sample values, which we generate by `n = 0.5*std(F)*(randn(size(F)) + 1i*randn(size(F)))`, where `std(F)` is the standard deviation of the samples.

In Figures 7.10 and 7.11, we show how ESPIRA and MPM compare for $K = 10, 25$ and increasing P. We also show the mean of the *signal-to-noise ratio*, or *SNR* for short, which is defined as

$$\mathrm{SNR} := 20 \cdot \log_{10}\left(\frac{\texttt{std(F)}}{\texttt{std(n)}}\right),$$

that results from adding the noise `n`.

Looking at the data, we see the true strength of ESPIRA. It clearly outperforms MPM every time. Especially for $K = 25$, ESPIRA reconstructs η in the median up to more than 3 orders of magnitude better than MPM and Δf is up to over 2 orders of magnitude better.

Another interesting observation is that for $K = 10$ even though the SNR is going down until $P = 8$, meaning that the proportion of the signal in the observed noisy data is decreasing, the results of ESPIRA keep getting better, which is somewhat counterintuitive.

Note that we did not actively implement any denoising measures in the ESPIRA algorithm. However, since we approximate the given function in the frequency domain using only K dominant frequencies in order to reconstruct the function, this naturally leads to a denoising.

This last observation gives rise to a new idea. Recall that in the AAA algorithm, which sits at the core of ESPIRA, we employ a greedy step in order to find the next sample value we want to interpolate, see step (ii)(b) of Algorithm 1.22 or step (ii)(e) of Algorithm 2.1. This step consists of finding the sample for which the difference between the value of the rational function obtained so far and the sample value is largest. Inspired by the observation made in the previous paragraph, we tried to use an even simpler greedy step in the case of noisy data, namely choosing the next interpolation point as the sample which has the largest absolute value among all samples not yet interpolated. However, this does not usually improve the quality of the reconstruction. While simply using the largest sample value as the next interpolation point results in $\Delta\phi$ being up to only half as large as when using the classical variant, it also leads to $\Delta\eta$ being up to twice as large. Generally, $\Delta\phi$ is smaller for the variant described here and $\Delta\eta$ is smaller for the classical variant. Regarding Δf both variants perform very similarly with sometimes the one and sometimes the other variant being better. However, in our experiments, we did not see a clear pattern emerge.

Let us briefly report on some more findings without giving detailed data. We did not only experiment with adding normally distributed noise but also uniformly distributed noise. The results we observed were virtually the same as for normally distributed noise. As already noted in the previous section, increasing the number of additional

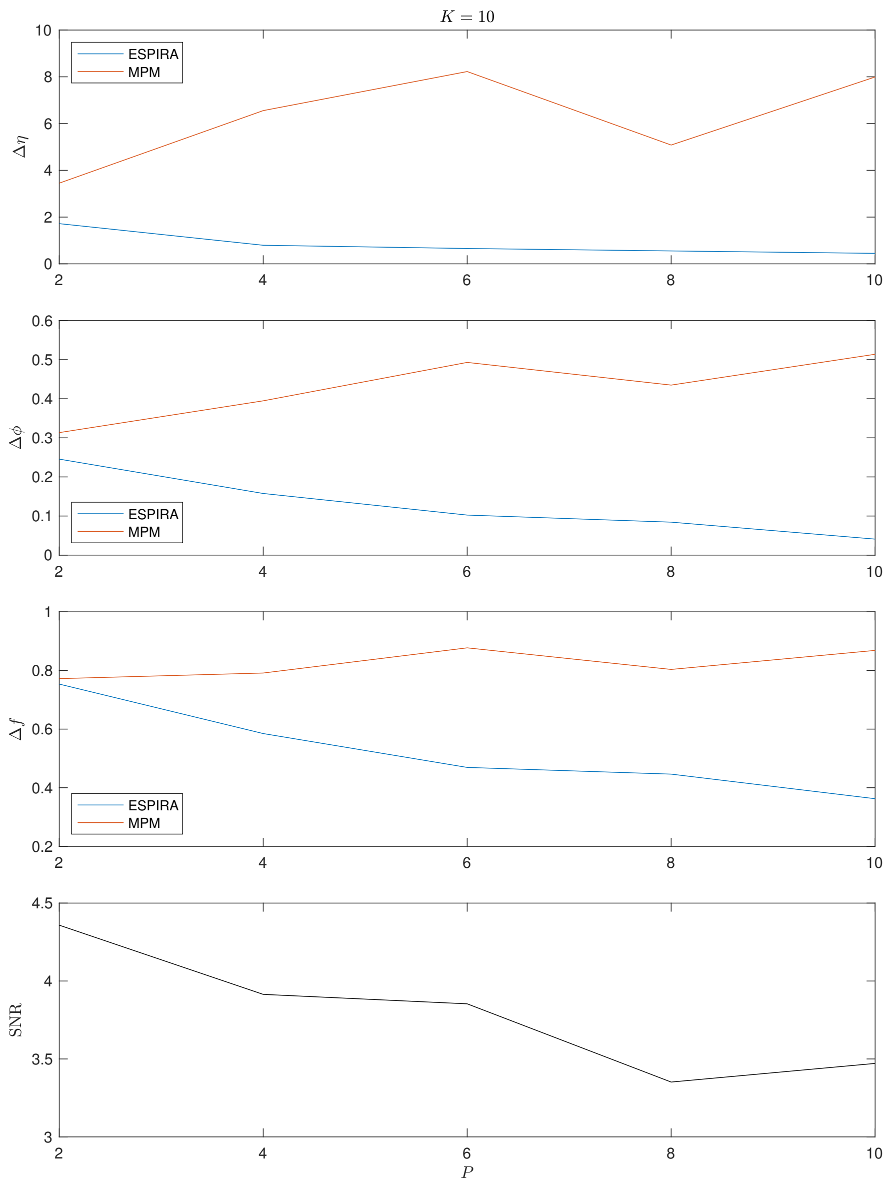

Figure 7.10.: Mean SNR as well as median of $\Delta\eta$, $\Delta\phi$, and Δf for incresing P with $r_{\mathrm{Re}\,\eta} = r_{\mathrm{Im}\,\eta} = 10$, $r_{\mathrm{Im}\,\phi} = 2 \cdot \pi \cdot 10$, and $r_{\mathrm{Re}\,\phi} = 0$.

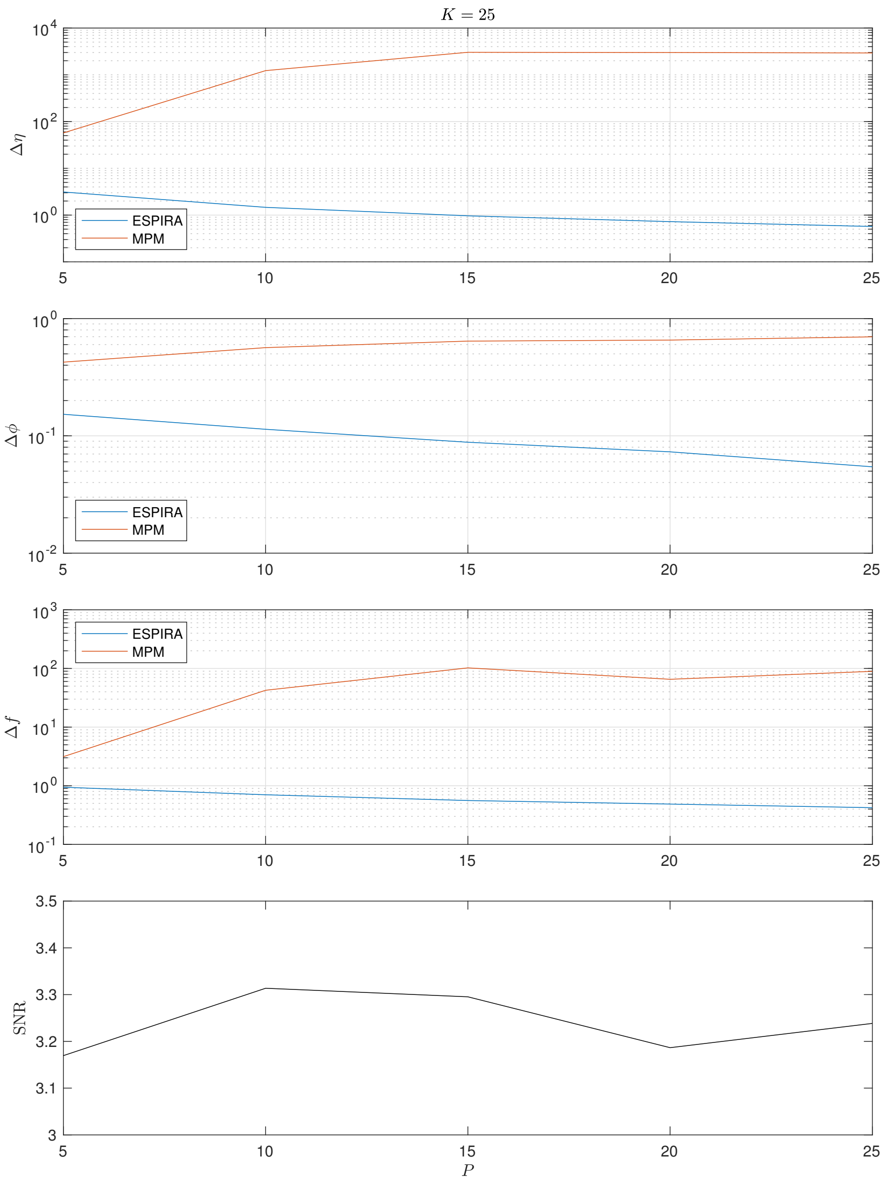

Figure 7.11.: Mean SNR as well as median of $\Delta\eta$, $\Delta\phi$, and Δf for incresing P with $r_{\mathrm{Re}\,\eta} = r_{\mathrm{Im}\,\eta} = 10$, $r_{\mathrm{Im}\,\phi} = 2 \cdot \pi \cdot 10$, and $r_{\mathrm{Re}\,\phi} = 0$.

sample points has far less of an effect than increasing P, i.e., the interval in which we sample. However, adding more sample points usually seems to improve the results of ESPIRA slightly, while it actually slightly worsens the results of MPM. We also tried different levels of noise, i.e., noise produced via `s*std(F)*(randn(size(F)) + 1i*randn(size(F)))` for different values of `s`. This of course results in different SNR values. Comparing the results of ESPIRA and MPM, we found that the smaller the SNR value is, i.e., the more the samples are corrupted by noise, the better ESPIRA performs compared to MPM. As in the previous section, we also looked at the effect of increasing $r_{\mathrm{Re}\,\phi}$. Interestingly, in this case, both algorithms perform very similarly, with $\Delta\eta$ and $\Delta\phi$ usually being smaller for ESPIRA and Δf being smaller for MPM.

We close this chapter by looking at one last specific example. We again consider the function given by the parameters in (7.4), where, as above, we add noise generated by `0.5*std(F)*(randn(size(F)) + 1i*randn(size(F)))`. For this example, we use a far larger sampling interval than before, namely $[0, 1500]$, i.e., P = 1500. We report our findings in Table 7.10 and show the reconstruction results on the interval $[0, 50]$ in Figures 7.12 and 7.13.

	$\Delta\eta$	$\Delta\phi$	Δf
ESPIRA	1.7228e−01	1.1774e−03	1.5463e−01
MPM	2.7701e+01	1.1334e+00	9.2596e−01

Table 7.10.: Reconstruction errors for the function specified by the parameters in (7.4) for $P = 1500$ and SNR 2.98.

In Table 7.10, we see the results that we would have expected. ESPIRA is roughly two orders of magnitude better than MPM for $\Delta\eta$ and three orders of magnitude better for $\Delta\phi$. Looking closely at Δf, we find that ESPIRA is almost one order of magnitude better. However, what this actually means becomes clear once we look at Figures 7.12 and 7.13. There, we can see that even though the noisy samples, marked as dots in the plots, deviate quite a lot from the original function, ESPIRA reconstructs both the real and imaginary part very well. The reconstruction obtained by MPM on the other hand, although clearly following the shape of the original function, is far more off. This reconstruction often seems to lie closer to the noisy sample data than the actual function.

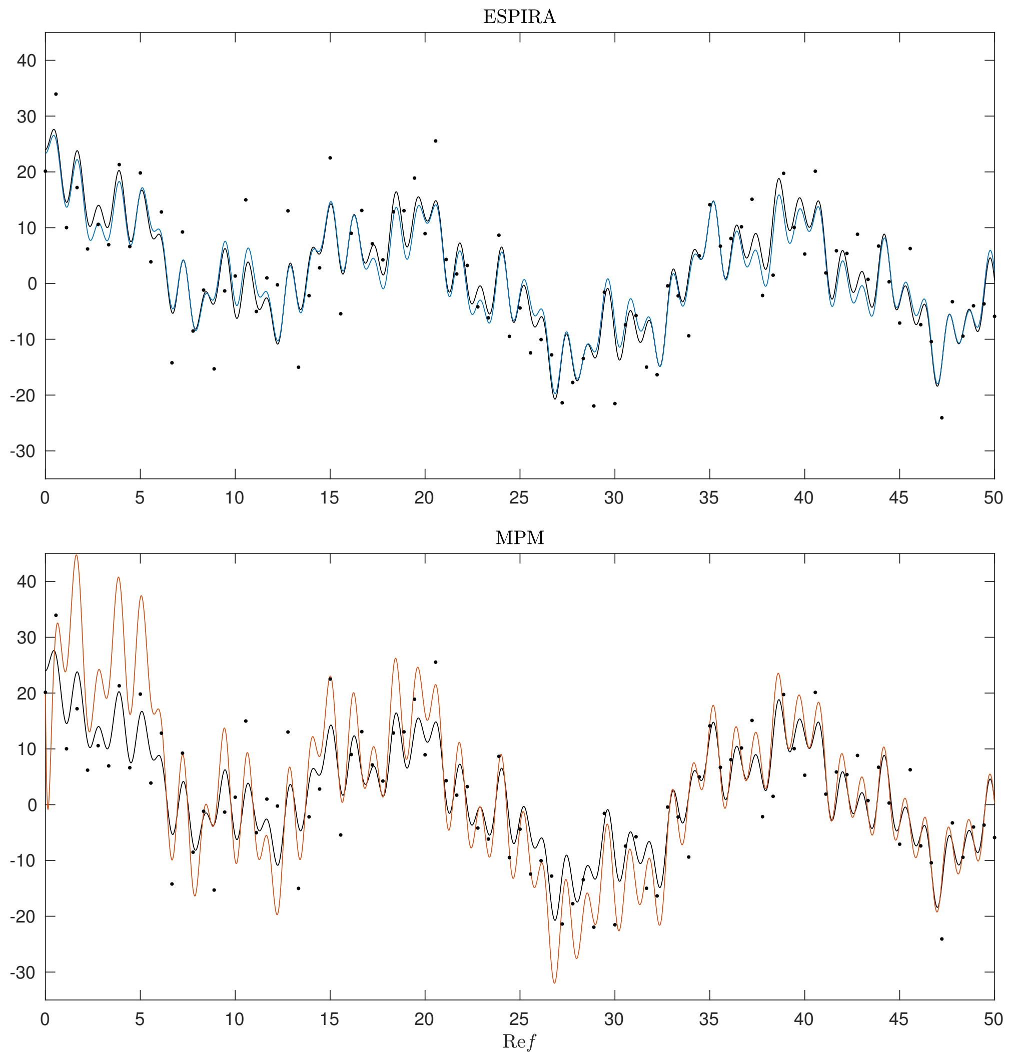

Figure 7.12.: Reconstruction of the real part of the function given by (7.4) by ESPIRA and MPM for noisy input data with SNR 2.98. Original function in black.

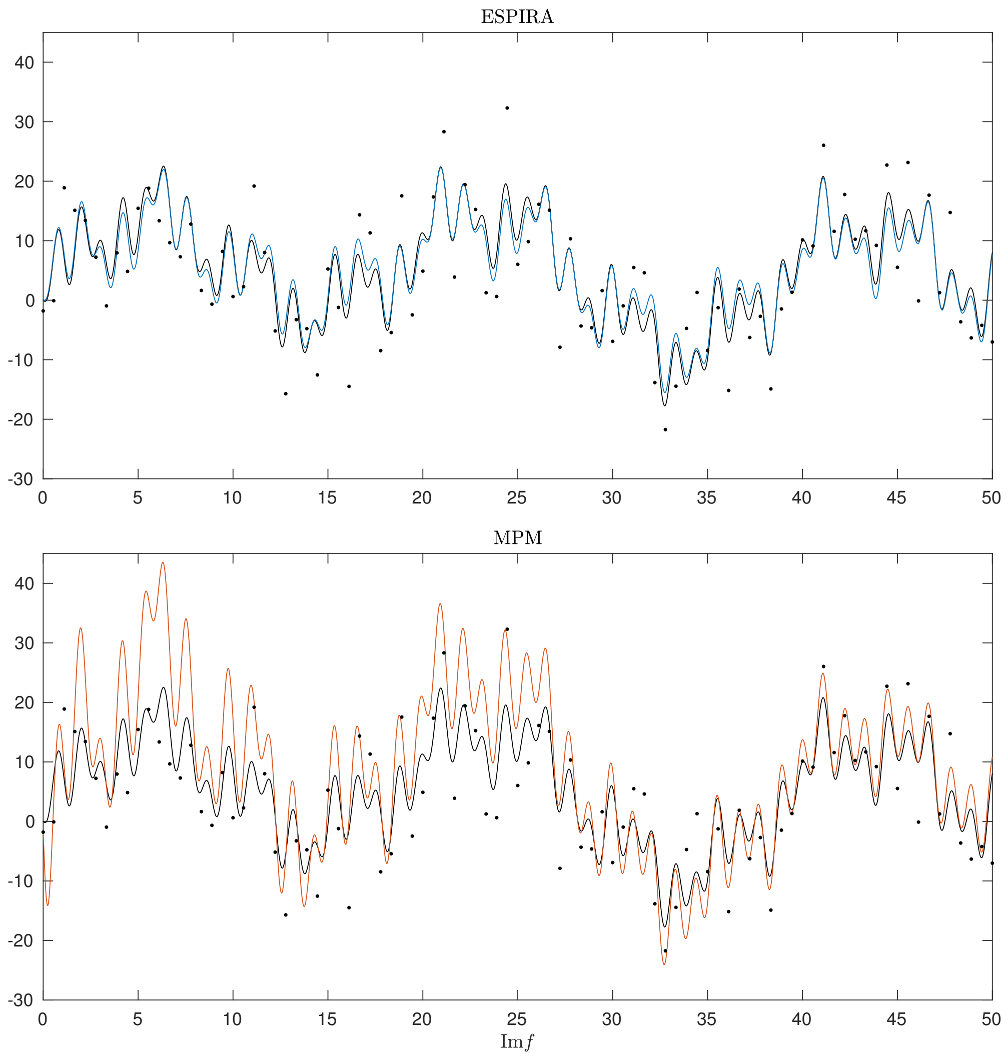

Figure 7.13.: Reconstruction of the imaginary part of the function given by (7.4) by ESPIRA and MPM for noisy input data with SNR 2.98. Original function in black.

Conclusion and Outlook

Let us shortly sum up our findings. We have seen in Chapters 4 and 6 that, both in the continuous and discrete case, the Fourier coefficients with respect to a period P of an exponential sum can be interpreted as samples from a rational function. This rational function can be reconstructed via rational interpolation of the Fourier coefficients. If the function contains P-periodic components, then the set of Fourier coefficients contains unattainable points.

Further, we showed in our Main Theorems 4.5 and 6.9 that the parameters defining an exponential sum can, under suitable conditions, be uniquely recovered from the rational function interpolating the Fourier coefficients. We therefore devised the new Algorithms 4.7 and 6.11 to recover these parameters from Fourier samples using rational interpolation. Since the latter algorithm works with DFT coefficients, it can be used directly on function samples.

In order to obtain a numerically stable algorithm, we used barycentric rational interpolation, more precisely the AAA algorithm. We showed in Chapter 1 that in our case the AAA algorithm always converges. In particular, we demonstrated that this is also true if the given data contains unattainable points, i.e., the given function includes P-periodic components. To do this, we used Theorem 1.15. It generalizes a well-known result, which links barycentric rational interpolation to certain Löwner matrices, to the case that the data from which the Löwner matrix is built contains unattainable points. We also developed a new stopping criterion based on this theorem that is more suitable to our problem than the original stopping criterion for the AAA algorithm.

Finally, we thoroughly tested our new ESPIRA algorithm and compared it to the well-established MPM algorithm. We found that both algorithms deliver very good results. Usually, MPM produced slightly better errors. However, in many cases, ESPIRA found the correct number of terms in the exponential sum more often than MPM. Overall it appeared that ESPIRA behaves more consistently with regard to the input data, i.e., it behaves in a more predictable fashion than MPM. At this point, we once more want to point out, that MPM has been around for more than 30 years and has seen considerable improvements over time. In particular, as we discussed in Appendix B, we used a version of MPM including a preconditioning step that greatly enhances its performance. Our algorithm, on the other hand, has not yet seen such refinement. Nonetheless, even in its current form, we saw that, if we have noisy input data, ESPIRA clearly outperforms MPM.

This leads to the question of what can be done next. As just mentioned, we are certain that it is possible to improve ESPIRA by way of improving its implementation. For example, we found that the overdetermined system of equations involving a Cauchy matrix that is used to reconstruct the coefficient parameters η_k for $k = 1, 2, \ldots, K$ of an exponential sum with K terms is sometimes a source of relatively large errors. Improving upon this step would certainly benefit ESPIRA in general.

Some research expanding on topics of this thesis has already been conducted. In [35], the findings of Chapter 4 are generalized to so called *Extended Exponential Sums*. These are functions of the form

$$\sum_{k=1}^{K} \left(\sum_{j=1}^{J_k} \gamma_{j,k} \cdot t^j \right) \cdot \mathrm{e}^{\phi_k \cdot t},$$

i.e., exponential sums where the constant coefficient η_k for $k = 1, 2, \ldots, K$ has been replaced by a polynomial.

We argued in Remark 6.8 that it is probably not possible to modify the algorithm for DFT data similarly as in Chapter 5, i.e., if the input data stems from a real almost-periodic function, we cannot modify the algorithm such that it only needs two times as many samples as the number of components of the function, as opposed to four times as many. However, if instead of the discrete Fourier transform the discrete cosine transform is used, this is possible, as has been shown in [37].

An idea we have not looked into at all is to consider a different method for rational interpolation than the AAA algorithm. In the case of noisy data, one might even consider not using rational interpolation but rather rational approximation, for example by employing the RKFIT algorithm [8].

Another path to follow could be to emulate developments in Prony's method. For example, in [89] the generalized Prony method has been introduced, which was further improved in [93]. This method can be used to reconstruct functions of the form

$$\sum_{k=1}^{K} \eta_k \cdot v_{\lambda_k},$$

where $\eta_k \in \mathbb{C} \setminus \{0\}$ and v_{λ_k} for $k = 1, 2, \ldots, K$ are eigenfunctions of a linear operator on a normed vector space corresponding to pairwise distinct elements λ_k in the point spectrum of the operator. We believe that ESPIRA can also be generalized to be used in this setting.

Since all our results were obtained in the one-dimensional case, a natural question to ask is whether they can be generalized to higher dimensions. Such generalizations exist for Prony's and Prony-like methods, see for example [18, 32, 31, 40, 66, 79].

Appendix

A. Sums of Cosines with Common Frequency

Lemma A.1. *For $a, \rho, \gamma \in \mathbb{R}_{>0}$ and $b, c \in [0, 2\pi)$ with $b \neq c$ we have*

$$\rho \cdot \cos(a \cdot t + b) + \gamma \cdot \cos(a \cdot t + c) = \Gamma \cdot \cos\left(a \cdot t + \tfrac{b+c}{2} + \varphi\right),$$

where

$$\Gamma = \sqrt{\rho^2 + \gamma^2 + 2\rho\gamma \cdot \cos(b-c)}$$

and

$$\varphi = \operatorname{atan2}\left((\rho - \gamma) \cdot \sin\left(\tfrac{b-c}{2}\right), (\rho + \gamma) \cdot \cos\left(\tfrac{b-c}{2}\right)\right).$$

Proof. Throughout this proof we will use the trigonometric identities

$$\cos(x \pm y) = \cos(x) \cdot \cos(y) \mp \sin(x) \cdot \sin(y)$$

and

$$\frac{1}{2} \cdot \left(\cos(x+y) + \cos(x-y)\right) = \cos(x) \cdot \cos(y),$$

where the second identity is a straightforward consequence of the first. We have

$$\begin{aligned}
&\rho \cdot \cos(a \cdot t + b) + \gamma \cdot \cos(a \cdot t + c) \\
&\quad = \tfrac{\rho+\gamma}{2} \cdot 2 \cdot \tfrac{1}{2} \cdot \left(\cos\left(a \cdot t + \tfrac{b+c}{2} + \tfrac{b-c}{2}\right) + \cos\left(a \cdot t + \tfrac{b+c}{2} - \tfrac{b-c}{2}\right)\right) \\
&\qquad + \tfrac{\rho-\gamma}{2} \cdot 2 \cdot \tfrac{1}{2} \cdot \left(\cos\left(a \cdot t + \tfrac{b+c}{2} + \tfrac{b-c}{2}\right) - \cos\left(a \cdot t + \tfrac{b+c}{2} - \tfrac{b-c}{2}\right)\right) \\
&\quad = \underbrace{\tfrac{\rho+\gamma}{2} \cdot 2 \cdot \cos\left(\tfrac{b-c}{2}\right)}_{=:\Gamma_1} \cdot \cos\left(a \cdot t + \tfrac{b+c}{2}\right) - \underbrace{\tfrac{\rho-\gamma}{2} \cdot 2 \cdot \sin\left(\tfrac{b-c}{2}\right)}_{=:\Gamma_2} \cdot \sin\left(a \cdot t + \tfrac{b+c}{2}\right)
\end{aligned}$$

and

$$\begin{aligned}
\Gamma^2 &= \rho^2 + \gamma^2 + 2\rho\gamma\cos(b-d) \\
&= \rho^2 + \gamma^2 + 2\rho\gamma \cdot \cos\left(\tfrac{b-d}{2} + \tfrac{b-d}{2}\right) \\
&= (\rho^2 + \gamma^2) \cdot \left(\cos^2\left(\tfrac{b-d}{2}\right) + \sin^2\left(\tfrac{b-d}{2}\right)\right) \\
&\qquad + 2\rho\gamma \cdot \left(\cos^2\left(\tfrac{b-d}{2}\right) - \sin^2\left(\tfrac{b-d}{2}\right)\right) \\
&= (\rho+\gamma)^2 \cdot \cos^2\left(\tfrac{b-d}{2}\right) + (\rho-\gamma)^2 \cdot \sin^2\left(\tfrac{b-d}{2}\right) \\
&= \Gamma_1^2 + \Gamma_2^2.
\end{aligned}$$

Hence

$$\left(\frac{\Gamma_1}{\Gamma}\right)^2 + \left(\frac{\Gamma_2}{\Gamma}\right)^2 = 1,$$

and therefore, we can find $\varphi \in [0, 2\pi)$ such that $\frac{\Gamma_1}{\Gamma} = \cos(\varphi)$ and $\frac{\Gamma_2}{\Gamma} = \sin(\varphi)$. Since

$$\frac{\Gamma_2}{\Gamma_1} = \frac{\frac{\Gamma_2}{\Gamma}}{\frac{\Gamma_1}{\Gamma}} = \frac{\sin(\varphi)}{\cos(\varphi)} = \tan(\varphi),$$

we get

$$\varphi = \operatorname{atan2}(\Gamma_2, \Gamma_1).$$

Finally we find

$$\begin{aligned}
&\Gamma \cdot \cos\left(a \cdot t + \tfrac{b+c}{2} + \varphi\right) \\
&\quad = \Gamma \cdot \cos\left(a \cdot t + \tfrac{b+c}{2}\right) \cdot \cos(\varphi) - \Gamma \cdot \sin\left(a \cdot t + \tfrac{b+c}{2}\right) \cdot \sin(\varphi) \\
&\quad = (\rho+\gamma) \cdot \cos\left(a \cdot t + \tfrac{b+c}{2}\right) \cdot \cos\left(\tfrac{b-c}{2}\right) - (\rho-\gamma) \cdot \sin\left(a \cdot t + \tfrac{b+c}{2}\right) \cdot \sin\left(\tfrac{b-c}{2}\right) \\
&\quad = \rho \cdot \cos(a \cdot t + b) + \gamma \cdot \cos(a \cdot t + c).
\end{aligned}$$

□

B. The Matrix Pencil Method

We gave an overview of Prony's method and the Matrix Pencil Method similar to the presentation here in [36]. In this chapter, we work with exponential sums of the form (3.7), i.e.,

$$f(t) = \sum_{k=1}^{K} \eta_k \cdot z_k^t,$$

with $\eta_k, z_k \in \mathbb{C} \setminus \{0\}$ where the z_k are pairwise distinct for $k = 1, 2, \ldots, K$, since this is the form commonly used in the literature. However, all results can be generalized to hold for exponential sums of the form (3.6).

We start by presenting the classical Prony method as it can be found, for example, in [94]. First, define the Prony polynomial

$$p(z) \coloneqq \prod_{k=1}^{K} (z - z_k) = z^K + \sum_{j=0}^{K-1} p_j \cdot z^j.$$

Let $M \geq 2K$ and define $f_m = f(m)$ for $m = 0, 1, \ldots, M-1$. Then we find that

$$\begin{aligned}\sum_{j=0}^{K-1} p_j \cdot f_{n+j} &= \sum_{j=0}^{K-1} p_j \sum_{k=1}^{K} \eta_k \cdot z_k^{n+j} = \sum_{k=1}^{K} \eta_k \cdot z_k^n \sum_{j=0}^{K-1} p_j \cdot z_k^j \\ &= \sum_{k=1}^{K} \eta_k \cdot z_k^n \cdot \big(\underbrace{p(z_k)}_{=0} - z_k^K\big) = -\sum_{k=1}^{K} \eta_k \cdot z_k^{n+K} = -f_{n+K}\end{aligned}$$

for $n = 0, 1, \ldots, M-K-1$. This leads to the linear system

$$(f_{n+j})_{n,j=0}^{M-K-1,K-1} \cdot \mathbf{p} = -(f_{n+K})_{n=0}^{M-K-1} \quad \Leftrightarrow \quad \underbrace{(f_{n+j})_{n,j=0}^{M-K-1,K}}_{=:\mathbf{H}_{M-K-1,K}} \cdot \begin{pmatrix} \mathbf{p} \\ 1 \end{pmatrix} = \mathbf{0}, \qquad \text{(B.1)}$$

where $\mathbf{p} = (p_0, p_1, \ldots, p_{K-1})^T$ and

$$\mathbf{H}_{M-K-1,K} = \begin{pmatrix} f_0 & f_1 & f_2 & \cdots & f_K \\ f_1 & f_2 & & \iddots & \vdots \\ f_2 & & \iddots & & \vdots \\ \vdots & \iddots & & & \vdots \\ f_K & & & & \vdots \\ \vdots & & & & f_{M-2} \\ f_{M-K-1} & \cdots & \cdots & f_{M-2} & f_{M-1} \end{pmatrix} \tag{B.2}$$

is a so called *Hankel matrix*. Therefore, we can reconstruct p by solving (B.1) and find z_k for $k = 1, 2, \ldots, K$ by computing the roots of p.

Finally, we note that

$$\sum_{k=1}^{K} \eta_k \cdot z_k^m = f_m$$

for $m = 1, 2, \ldots, M-1$. This means that, writing $\mathbf{z} = (z_1, z_2, \ldots, z_K)^T$, we can compute η_k for $k = 1, 2, \ldots, K$ by solving

$$\mathbf{V}_{M-1}(\mathbf{z})^T \cdot (\eta_k)_{k=1}^{K} = (f_m)_{m=0}^{M-1},$$

with a Vandermonde matrix

$$\mathbf{V}_{M-1}(\mathbf{z}) = \begin{pmatrix} 1 & z_1 & z_1^2 & \cdots & z_1^{M-1} \\ 1 & z_2 & z_2^2 & \cdots & z_2^{M-1} \\ \vdots & \vdots & \vdots & & \vdots \\ 1 & z_K & z_K^2 & \cdots & z_K^{M-1} \end{pmatrix}.$$

It is well known that a Hankel matrix of the form (B.2), i.e., where the entries are equispaced samples of an exponential sum, can be factorized as

$$\mathbf{H}_{M-K-1,K} = \mathbf{V}_{M-K-1}(\mathbf{z})^T \cdot \operatorname{diag}\Big((\eta_k)_{k=1}^{K}\Big) \cdot \mathbf{V}_K(\mathbf{z}).$$

More general, if $L \geq K$ is an estimate for the number K of components and we have $M \geq 2L$ samples given, then the Hankel matrix $\mathbf{H}_{M-L-1,L}$ can be factorized as

$$\mathbf{H}_{M-L-1,L} = \mathbf{V}_{M-L-1}(\mathbf{z})^T \cdot \operatorname{diag}\Big((\eta_k)_{k=1}^{K}\Big) \cdot \mathbf{V}_L(\mathbf{z}). \tag{B.3}$$

Since $\mathbf{V}_{M-L-1}(\mathbf{z})^T$ has full column rank K and $\mathbf{V}_L(\mathbf{z})$ has full row rank K, this implies that $\operatorname{rk}\mathbf{H}_{M-L-1,L} = K$, i.e., the rank of $\mathbf{H}_{M-L-1,L}$ encodes the number of terms in the exponential sum.

Using (B.3). we now derive the Matrix Pencil Method. This method was first introduced in [57]. However, in our presentation we will loosely follow [97]. Define

$$\mathbf{H}^0_{M-L-1,L-1} \coloneqq (f_{n+j})_{n,j=0}^{M-L-1,L-1} \quad \text{and} \quad \mathbf{H}^1_{M-L-1,L-1} \coloneqq (f_{n+j+1})_{n,j=0}^{M-L-1,L-1},$$

where the first matrix is obtained from $\mathbf{H}_{M-L-1,L}$ by removing the last column and the second by removing the first column. It can be easily verified that

$$\mathbf{H}^0_{M-L-1,L-1} = \mathbf{V}_{M-L-1}(\mathbf{z})^T \cdot \operatorname{diag}\Big((\eta_k)_{k=1}^K\Big) \cdot \mathbf{V}_{L-1}(\mathbf{z})$$

and

$$\mathbf{H}^1_{M-L-1,L-1} = \mathbf{V}_{M-L-1}(\mathbf{z})^T \cdot \operatorname{diag}\Big((\eta_k \cdot z_k)_{k=1}^K\Big) \cdot \mathbf{V}_{L-1}(\mathbf{z}).$$

The Matrix Pencil Method is based on the observation that the rectangular matrix pencil

$$z \cdot \mathbf{H}^0_{M-L-1,L-1} - \mathbf{H}^1_{M-L-1,L-1} = \mathbf{V}_{M-L-1}(\mathbf{z})^T \cdot \operatorname{diag}\Big((\eta_k \cdot (z - z_k))_{k=1}^K\Big) \cdot \mathbf{V}_{L-1}(\mathbf{z})$$

has rank K for all $z \in \mathbb{C} \setminus \{z_k : k = 1, 2, \ldots, K\}$ and rank $K-1$ for $z = z_1, z_2, \ldots, z_K$. Note that in the special case that $L = K$ and $M = 2K$ the matrices $\mathbf{H}^0_{M-L-1,L-1} = \mathbf{H}^0_{K-1,K-1}$ and $\mathbf{H}^1_{K-1,K-1}$ are square matrices with full rank. This means that z_k for $k = 1, 2, \ldots, K$ are the eigenvalues of the regular matrix pencil $z \cdot \mathbf{H}^0_{K-1,K-1} - \mathbf{H}^1_{K-1,K-1}$ and finding them is a generalized eigenvalue problem, see Section 2.4.

We now compute a QR decomposition with column pivot, see [110, Lecture 7] and [49, Section 5.5.6], to obtain

$$\mathbf{H}_{M-L-1,L} \cdot \mathbf{P} = \mathbf{Q} \cdot \mathbf{R},$$

where $\mathbf{Q} \in \mathbb{C}^{(M-L)\times(M-L)}$ is unitary, $\mathbf{P} \in \mathbb{C}^{(L+1)\times(L+1)}$ is a permutation matrix, and

$$\mathbf{R} = \begin{pmatrix} \hat{\mathbf{R}} \\ \mathbf{0} \end{pmatrix} \in \mathbb{C}^{(M-L)\times L+1},$$

with a trapezoidal matrix $\hat{\mathbf{R}} \in \mathbb{C}^{K\times L+1}$ of full row rank K. Constructing $\left(\mathbf{P}^T\right)^0$ and

$\left(\mathbf{P}^T\right)^1$ by removing the last, respectively first, column from $\mathbf{P}^T$, we can then rewrite

$$\begin{aligned} z \cdot \mathbf{H}^0_{M-L-1,L-1} - \mathbf{H}^1_{M-L-1,L-1} &= z \cdot \mathbf{Q} \cdot \mathbf{R} \cdot \left(\mathbf{P}^T\right)^0 - \mathbf{Q} \cdot \mathbf{R} \cdot \left(\mathbf{P}^T\right)^1 \\ &= \mathbf{Q} \cdot \left(z \cdot \mathbf{R} \cdot \left(\mathbf{P}^T\right)^0 - \mathbf{R} \cdot \left(\mathbf{P}^T\right)^1\right). \end{aligned}$$

Since $\mathbf{Q}$ is unitary and using the structure of $\mathbf{R}$, we find that the matrix pencil

$$z \cdot \hat{\mathbf{R}} \cdot \left(\mathbf{P}^T\right)^0 - \hat{\mathbf{R}} \cdot \left(\mathbf{P}^T\right)^1 \in \mathbb{C}^{K \times L} \tag{B.4}$$

has rank K for all $z \in \mathbb{C} \setminus \{z_k : k = 1, 2, \dots, K\}$ and rank $K - 1$ for $z = z_1, z_2, \dots, z_K$.

In [97], the authors suggest to add a further preconditioning step. Let $\hat{\mathbf{R}} = (r_{kl})_{k,l=1}^{K,L+1}$ and define $\mathbf{D} \coloneqq \operatorname{diag}\left((r_{kk})_{k=1}^K\right)$, where, by construction, $r_{kk} \neq 0$ for all $k = 1, 2, \dots, K$. Now write

$$\mathbf{S}^0 \coloneqq \mathbf{D}^{-1} \cdot \hat{\mathbf{R}} \cdot \left(\mathbf{P}^T\right)^0 \quad \text{and} \quad \mathbf{S}^1 \coloneqq \mathbf{D}^{-1} \cdot \hat{\mathbf{R}} \cdot \left(\mathbf{P}^T\right)^1.$$

Then the matrix pencil

$$z \cdot \mathbf{S}^0 - \mathbf{S}^1$$

has the same properties as (B.4) and can be used to find z_k for $k = 1, 2, \dots, K$.

We found in our experiments that using this preconditioning step greatly improves the performance of the MPM algorithm. We also used this variant for our comparisons.

Finally, the values z_k for $k = 1, 2, \dots, K$ are computed as the eigenvalues of

$$\left(\left(\mathbf{S}^0\right)^T\right)^\dagger \cdot \left(\mathbf{S}^1\right)^T \in \mathbb{C}^{K \times K}$$

where $\left(\left(\mathbf{S}^0\right)^T\right)^\dagger$ denotes the *Moore-Penrose pseudoinverse* of $\left(\mathbf{S}^0\right)^T$. This can be seen as follows. Since $z \cdot \mathbf{S}^0 - \mathbf{S}^1$ has full row rank for all $z \in \mathbb{C} \setminus \{z_k : k = 1, 2, \dots, K\}$, it follows that $z \cdot \left(\mathbf{S}^0\right)^T - \left(\mathbf{S}^1\right)^T$ has full column rank for all $z \in \mathbb{C} \setminus \{z_k : k = 1, 2, \dots, K\}$. Put differently, we have $\operatorname{null}\left(z_k \cdot \left(\mathbf{S}^0\right)^T - \left(\mathbf{S}^1\right)^T\right) = 1$ for $k = 1, 2, \dots, K$. In particular, the nullity is exactly 1, which means that there is a unique vector $\mathbf{x}_k \in \mathbb{C}^K$ such that

$$\left(z_k \cdot \left(\mathbf{S}^0\right)^T - \left(\mathbf{S}^1\right)^T\right) \cdot \mathbf{x}_k = \mathbf{0} \tag{B.5}$$

for all $k = 1, 2, \dots, K$. Since $\left(\mathbf{S}^0\right)^T$ has full column rank K, it follows that

$$\left(\left(\mathbf{S}^0\right)^T\right)^\dagger = \left(\mathbf{S}^0 \cdot \left(\mathbf{S}^0\right)^T\right)^{-1} \cdot \mathbf{S}^0, \tag{B.6}$$

see [49, Section 5.5.4]. Putting (B.5) and (B.6) together, we find that

$$\left((\mathbf{S}^0)^T\right)^\dagger \cdot \left(z_k \cdot (\mathbf{S}^0)^T - (\mathbf{S}^1)^T\right) \cdot \mathbf{x}_k = \left(z_k \cdot \mathbf{I} - \left((\mathbf{S}^0)^T\right)^\dagger \cdot (\mathbf{S}^1)^T\right) \cdot \mathbf{x}_k = \mathbf{0},$$

where $\mathbf{I} \in \mathbb{C}^{K\times K}$ denotes the identity matrix. In other words, $\mathbf{x}_k$ is an eigenvector of $\left((\mathbf{S}^0)^T\right)^\dagger \cdot (\mathbf{S}^1)^T$ to the eigenvalue z_k for $k = 1, 2, \dots, K$.

Lastly, the values η_k for $k = 1, 2, \dots, K$ are computed by solving an overdetermined system of linear equations as in the classical Prony algorithm.

It remains to mention that for the MPM algorithm, we need an estimate L of the number of components K. In the implementation we used, we chose $L = \left\lfloor \frac{M}{2} \right\rfloor$.

Bibliography

[1] Antoulas, A. C., and Anderson, B. D. O. On the Scalar Rational Interpolation Problem. *IMA Journal of Mathematical Control and Information 3* (1986), 61–88.

[2] Arnold, V. I. *Mathematical Methods of Classical Mechanics*, second ed., vol. 60 of *Graduate Texts in Mathematics*. Springer, New York, 1989.

[3] Barnett, S. A Companion Matrix Analogue for Orthogonal Polynomials. *Linear Algebra and its Applications 12*, 3 (1975), 197–202.

[4] Beinert, R., and Plonka, G. Sparse Phase Retrieval of One-Dimensional Signals by Prony's Method. *Frontiers in Applied Mathematics and Statistics 3* (2017).

[5] Belevitch, V. Interpolation Matrices. *Philips Research Reports 25* (1970), 337–369.

[6] Bellman, R. *Introduction to Matrix Analysis*, first ed. McGraw-Hill Book Company, New York, 1960.

[7] Berent, J., Dragotti, P. L., and Blu, T. Sampling Piecewise Sinusoidal Signals With Finite Rate of Innovation Methods. *IEEE Transactions on Signal Processing 58*, 2 (2010), 613–625.

[8] Berljafa, M., and Güttel, S. The RKFIT Algorithm for Nonlinear Rational Approximation. *SIAM Journal on Scientific Computing 39*, 5 (2017), A2049–A2071.

[9] Berrut, J.-P. Rational Functions for Guaranteed and Experimentally Well-Conditioned Global Interpolation. *Computers and Mathematics with Applications 15*, 1 (1988), 1–16.

[10] Berrut, J.-P. A Matrix for Determining Lower Complexity Barycentric Representations of Rational Interpolants. *Numerical Algorithms 24* (2000), 17–29.

[11] Berrut, J.-P., Baltensperger, R., and Mittelmann, H. D. Recent Developments in Barycentric Rational Interpolation. In *Trends and Applications in Constructive Approximation*, D. H. Mache, J. Szabados, and M. G. de Bruin, Eds., vol. 151 of *International Series of Numerical Mathematics*. Birkhäuser, Basel, 2005, pp. 27–51.

[12] Berrut, J.-P., and Mittelmann, H. D. Lebesgue Constant Minimizing Linear Rational Interpolation of Continuous Functions over the Interval. *Computers and Mathematics with Applications 33*, 6 (1997), 77–86.

[13] Berrut, J.-P., and Mittelmann, H. D. Matrices for the Direct Determination of the Barycentric Weights of Rational Interpolation. *Journal of Computational and Applied Mathematics 78* (1997), 355–370.

[14] Berrut, J.-P., and Trefethen, L. N. Barycentric Lagrange Interpolation. *SIAM Review 46*, 3 (2004), 501–517.

[15] Besicovitch, A. S. *Almost Periodic Functions.* Cambridge University Press, Cambridge, 1932.

[16] Beylkin, G., and Monzón, L. On Approximation of Functions by Exponential Sums. *Applied and Computational Harmonic Analysis 19*, 1 (2005), 17–48.

[17] Bohr, H. Fastperiodische Funktionen. *Jahresbericht der Deutschen Mathematiker-Vereinigung 34* (1926), 25–40.

[18] Bosner, N. Parallel Prony's Method with Multivariate Matrix Pencil Approach and Its Numerical Aspects. *SIAM Journal on Matrix Analysis and Applications 42*, 2 (2021), 635–658.

[19] Bossmann, F., Plonka, G., Peter, T., Nemitz, O., and Schmitte, T. Sparse Deconvolution Methods for Ultrasonic NDT. *Journal of Nondestructive Evaluation 31* (2012), 225–244.

[20] Braess, D., and Hackbusch, W. Approximation of $1/x$ by exponential sums in $[1, \infty)$. *IMA Journal of Numerical Analysis 25*, 4 (2005), 685–697.

[21] Brezinski, C., and Redivo-Zaglia, M. Padé-Type Rational and Barycentric Interpolation. *Numerische Mathematik 125* (2013), 89–113.

[22] Brown, J. W., and Churchill, R. V. *Complex Variables and Applications*, eighth ed. McGraw-Hill, Boston, 2009.

[23] Burlisch, R., and Rutishauser, H. Interpolation und genäherte Quadratur. In *Mathematische Hilfsmittel des Ingenieurs*, R. Sauer and I. Szabó, Eds., vol. 141 of *Grundlehren der mathematischen Wissenschaften.* Springer, Berlin, 1968, pp. 232–319.

[24] Cauchy, A.-L. Sur la formule de Lagrange relative à l'interpolation. In *Cours d'Analyse de l'École Royale Polytechnique: Analyse Algébrique.* L'Imprimerie Royale, Paris, 1821, pp. 525–529.

[25] Cauchy, A.-L. *Exercices d'Analyse et de Physique Mathématique*, vol. 2. Bachelier, Paris, 1841.

[26] Claessens, G. On the Newton-Padé Approximation Problem. *Journal of Approximation Theory 22* (1978), 150–160.

[27] Corduneanu, C. *Almost Periodic Functions*, second ed. Chelsea Publishing Company, New York, 1989.

[28] Corless, R. M. Generalized Companion Matrices in the Lagrange Basis. In *Proceedings of the EACA*, L. González-Vega and T. Recio, Eds. 2004, pp. 317–322.

[29] Corless, R. M. On a Generalized Companion Matrix Pencil for Matrix Polynomials Expressed in the Lagrange Basis. In *Symbolic-Numeric Computation*, D. Wang and L. Zhi, Eds., Trends in Mathematics. Birkhäuser, Basel, 2007, pp. 1–15.

[30] Corless, R. M., and Litt, G. Generalized Companion Matrices for Polynomials not expressed in Monomial Bases. unpublished.

[31] Cuyt, A., Hou, Y., Knaepkens, F., and Lee, W.-s. Sparse Multidimensional Exponential Analysis with an Application to Radar Imaging. *SIAM Journal on Scientific Computing 42*, 3 (2020), B675–B695.

[32] Cuyt, A., and Lee, W.-s. Multivariate exponential analysis from the minimal number of samples. *Advances in Computational Mathematics 44* (2018), 987—1002.

[33] Cuyt, A., and Lee, W.-s. How To Get High Resolution Results from Sparse and Coarsely Sampled Data. *Applied and Computational Harmonic Analysis 48* (2020), 1066–1087.

[34] de Prony, G. R. Essai Experimental et Analytique sur les Lois de la Delatabilite des Fluides et sur Celles de la Force expansive del la Vapeur de l'Eau el de l'Alkool differentes Temperatures. *Journal l'Ecole Polytechnique 1*, 2 (1795), 24–76.

[35] Derevianko, N., and Plonka, G. Exact Reconstruction of Extended Exponential Sums using Rational Approximation of their Fourier Coefficients. *Analysis and Applications* (2021).

[36] Derevianko, N., Plonka, G., and Petz, M. From ESPRIT to ESPIRA: Estimation of Signal Parameters by Iterative Rational Approximation. *IMA Journal of Numerical Analysis* (2022).

[37] Derevianko, N., Plonka, G., and Razavi, R. ESPRIT versus ESPIRA for Reconstruction of short Cosine Sums and its Application. Preprint.

[38] Driscoll, T. A., Hale, N., and Trefethen, L. N., Eds. *Chebfun Guide.* Pafnuty Publications, Oxford, 2014.

[39] Dupuy, M. Le calcul numérique des fonctions par l'interpolation barycentrique. *Comptes Rendus Hebdomadaires des Séances de l'Académie des Sciences 226* (1948), 158–159.

[40] Ehler, M., Kunis, S., Peter, T., and Richter, C. A Randomized Multivariate Matrix Pencil Method for Superresolution Microscopy. *Electronic Transactions on Numerical Analysis 51* (2019), 63–74.

[41] Filbir, F., Mhaskar, H. N., and Prestin, J. On the Problem of Parameter Estimation in Exponential Sums. *Constructive Approximation 35* (2012), 323–343.

[42] Filip, S.-I., Nakatsukasa, Y., Trefethen, L. N., and Beckermann, B. Rational Minimax Approximation via Adaptive Barycentric Representations. *SIAM Journal on Scientific Computing 40*, 4 (2018), A2427–A2455.

[43] Floater, M. S., and Hormann, K. Barycentric Rational Interpolation with no Poles and High Rates of Approximation. *Numerische Mathematik 107* (2007), 315–331.

[44] Froissaret, M. Approximation de Padé: Application à la physique des particules élémentaires. *Les rencontres physiciens-mathématiciens de Strasbourg - RCP25 9* (1969), 1–13.

[45] Gantmacher, F. R. *The Theory of Matrices*, vol. 2. AMS Chelsea Publishing, Providence, 2000. Reprint.

[46] Gilewicz, J., and Pindor, M. Padé Approximants and Noise: A Case of Geometric Series. *Journal of Computational and Applied Mathematics 87* (1997), 199–214.

[47] Gilewicz, J., and Pindor, M. Padé Approximants and Noise: Rational Functions. *Journal of Computational and Applied Mathematics 105* (1999), 285–297.

[48] Gohberg, I., Lancaster, P., and Leiba, R. *Matrix Polynomials.* Academic Press, New York, 1982.

[49] Golub, G. H., and Van Loan, C. F. *Matrix Computations*, third ed. The Johns Hopkins University Press, Baltimore, 1996.

[50] Good, I. J. The Colleague Matrix, a Chebyshev Analogue of the Companion Matrix. *The Quarterly Journal of Mathematics 12*, 1 (1961), 61–68.

[51] Gunning, R. C., and Rossi, H. *Analytic Functions of Several Complex Variables.* AMS Chelsea Publishing, Providence, 1965.

[52] Gutknecht, M. H. The Rational Interpolation Problem Revisited. *The Rocky Mountain Journal of Mathematics 21*, 1 (1991), 263–280.

[53] Hamming, R. W. *Numerical Methods for Scientists and Engineers*, first ed. McGraw-Hill Book Company, New York, 1962.

[54] Henrici, P. *Essentials of Numerical Analysis.* Wiley, New York, 1982.

[55] Highham, N. J. The Numerical Stability of Barycentric Lagrange Interpolation. *IMA Journal of Numerical Analysis 24*, 4 (2004), 547–556.

[56] Horn, R. A., and Johnson, C. R. *Matrix Analysis.* Cambridge University Press, Cambridge, 1985.

[57] Hua, Y., and Sarkar, T. K. Matrix Pencil Method for Estimating Parameters of Exponetially Damped/Undamped Sinusoids in Noise. *IEEE Transactions on Acoustics, Speech, and Signal Processing 38*, 5 (1990), 814–824.

[58] Ioniță, A. C. *Lagrange Rational Interpolation and its Applications to Approximation of Large-Scale Dynamical Systems.* Ph.D. Thesis. Rice University, Houston, 2013.

[59] Ioniță, A. C., and Antoulas, A. C. Data-Driven Parametrized Model Reduction in the Loewner Framework. *SIAM Journal on Scientific Computing 36*, 3 (2014), A984–A1007.

[60] Ito, S., and Nakatsukasa, Y. Stable Polefinding and Rational Least-Squares Fitting via Eigenvalues. *Numerische Mathematik 139* (2018), 633–682.

[61] Jacobi, C. G. J. *Disquisitiones Analyticae de Fractionibus Simplicibus.* Ph.D. Thesis. Friedrich-Wilhelms-Universität, Berlin, 1825.

[62] Kaplan, D., and Glass, L. *Understanding Nonlinear Dynamics*, vol. 19 of *Texts in Applied Mathematics.* Springer, New York, 1995.

[63] Keller, I., and Plonka, G. Modifications of Prony's Method for the Recovery and Sparse Approximation with Generalized Exponential Sums. In *Approximation Theory XVI*, G. E. Fasshauer, M. Neamtu, and L. L. Schumaker, Eds. Springer, Cham, 2021, pp. 123–152.

[64] Klein, G. *Applications of Linear Barycentric Rational Interpolation.* Ph.D. Thesis. University of Fribourg, Fribourg, 2012.

[65] Knaepkens, F., Cuyt, A., Lee, W.-S., and de Villiers, D. I. L. Regular Sparse Array Direction of Arrival Estimation in one Dimension. *IEEE Transactions on Antennas and Propagation 68*, 5 (2020), 3997–4006.

[66] Kunis, S., Peter, T., Römer, T., and von der Ohe, U. A Multivariate Generalization of Prony's Method. *Linear Algebra and its Applications 490* (2016), 31–47.

[67] Kågström, B., and Ruhe, A., Eds. *Matrix Pencils*, vol. 973 of *Lecture Notes in Mathematics*. Springer, Berlin, 1983.

[68] Lagrange, J. L. d. Leçons Élémentaires sur les Mathématiques, données à l'École Normale en 1795. In *Œuvres de Lagrange*, vol. 7. Gauthier–Villars, Paris, 1877, pp. 183–287.

[69] Lang, S. *Complex Analysis*, fourth ed., vol. 103 of *Graduate Texts in Mathematics*. Springer, New York, 1999.

[70] Lawrence, P. W., and Corless, R. M. Stability of Rootfinding for Barycentric Lagrange Interpolants. *Numerical Algorithms 65*, 3 (2014), 447–464.

[71] Levitan, B. M., and Zhikov, V. V. *Almost Periodic Functions and Differential Equations*. Cambridge University Press, Cambridge, 1982.

[72] Lobos, T., Rezmer, J., and Schegner, P. Parameter Estimation of Distorted Signals Using Prony Method. *IEEE Bologna Power Tech Conference Proceedings 4* (2003).

[73] Löwner, K. Über Monotone Matrixfunktionen. *Mathematische Zeitschrift 38* (1934), 177–216.

[74] Maehly, H. J., and Witzgall, C. Tschebyscheff-Approximationen in kleinen Intervallen II. *Numerische Mathematik 2* (1960), 293–307.

[75] Mallat, S. *A Wavelet Tour of Signal Processing. The Sparse Way.* Elsevier, Amsterdam, 2009.

[76] Meijering, E. A Chronology of Interpolation: From Ancient Astronomy to Modern Signal and Image Processing. *Proceedings of the IEEE 90*, 3 (2002), 319–342.

[77] Meinguet, J. On the Solubility of the Cauchy Interpolation Problem. In *Approximation Teory*, A. Talbot, Ed., Proceedings of a Symposium held at Lancester, July 1969. Academic Press, London, 1970, pp. 137–163.

[78] Mirotin, A. R., and Mirotin, E. A. On Sums and Products of Periodic Funtions. *Real Analysis Exchange 34*, 2 (2009), 347–358.

[79] Mourrain, B. Polynomial–Exponential Decomposition From Moments. *Foundations of Computational Mathematics 18* (2018), 1435–1492.

[80] Nakatsukasa, Y., Sète, O., and Trefethen, L. N. The AAA Algorithm for Rational Approximation. *SIAM Journal on Scientific Computing 40*, 3 (2018), A1494–A1522.

[81] Nakatsukasa, Y., and Trefethen, L. N. An Algorithm for Real and Complex Rational Minimax Approximation. *SIAM Journal on Scientific Computing 42*, 5 (2020), A3157–A3179.

[82] Noda, M.-T., and Sasaki, T. Approximate GCD and its Application to Ill-Conditioned Algebraic Equations. *Journal of Computational and Applied Mathematics 38* (1991), 335–351.

[83] Organick, E. I. *A FORTRAN IV Primer*. Addison-Wesley, Reading, 1966.

[84] Pachón, R., Gonnet, P., and van Deun, J. Fast and Stable Rational Interpolation in Roots of Unity and Chebyshev Points. *SIAM Journal on Numerical Analysis 50*, 3 (2012), 1713–1734.

[85] Paley, R. E. A. C., and Wiener, N. *Fourier Transforms in the Complex Domain*, vol. 19 of *Colloquium Publications*. American Mathematical Society, New York, 1934.

[86] Pan, V. Y. Numerical Computation of a Polynomial GCD and Extensions. *INRIA*, RR-2969 (1996). Technical Report.

[87] Pan, V. Y. How bad are Vandermonde Matrices. *SIAM Journal on Matrix Analysis and Applications 37*, 2 (2016), 676–694.

[88] Pereyra, V., and Scherer, G. Exponential Data Fitting. In *Exponential Data Fitting and its Applications*, V. Pereyra and G. Scherer, Eds. Bentham Science Publishers, Sharjah, 2010, pp. 1–26.

[89] Peter, T., and Plonka, G. A Generalized Prony Method for Reconstruction of Sparse Sums of Eigenfunctions of Linear Operators. *Inverse Problems 29* (2013).

[90] Petz, M., Plonka, G., and Derevianko, N. Exact Reconstruction of Sparse Non-Harmonic Signals from Their Fourier Coefficients. *Sampling Theory, Signal Processing, and Data Analysis 19* (2021).

[91] Petz, M., Plonka, G., and Derevianko, N. Rational Functions for the Reconstructions of Exponentials Sums from their Fourier Coefficients. *Proceedings in Applied Mathematics and Mechanics 21*, 1 (2021).

[92] Plonka, G., Potts, D., Steidl, G., and Tasche, M. *Numerical Fourier Analysis*. Applied and Numerical Harmonic Analysis. Birkhäuser, Cham, 2018.

[93] Plonka, G., Stampfer, K., and Keller, I. Reconstruction of Stationary and Non-Stationary Signals by the Generalized Prony Method. *Analysis and Applications 17*, 02 (2019), 179–210.

[94] Plonka, G., and Tasche, M. Prony Methods for Recovery of Structured Functions. *GAMM-Mitteilungen 37*, 2 (2014), 239–258.

[95] Poole, D. *Linear Algebra - A Modern Introduction*, second ed. Thomson Brooks/Cole, Belmont, 2006.

[96] Potts, D., and Tasche, M. Parameter Estimation for Exponential Sums by Approximate Prony Method. *Signal Processing 90*, 5 (2010), 1631–1642.

[97] Potts, D., and Tasche, M. Parameter Estimation for Nonincreasing Exponential Sums by Prony-like Methods. *Linear Algebra and its Applications 439* (2013), 1024–1039.

[98] Pólya, G., and Szegö, G. *Aufgaben und Lehrsätze aus der Analysis*, fourth ed., vol. 2. Springer, Berlin, 1971.

[99] Roy, R. H., and Kailath, T. ESPRIT - Estimation of Signal Parameters via Rotational Invariance Techniques. *IEEE Transactions on Acoustics, Speech, and Signal Processing 37*, 7 (1989), 984–995.

[100] Rutishauser, H. *Vorlesungen über Numerische Mathematik, Band 1*. Birkhäuser, Basel, 1976. English Translation: Lectures on Numerical Mathematics, Walter Gautschi, ed., Birkhäuser, Boston, 1990.

[101] Salzer, H. E. Rational Interpolation using Incomplete Barycentric Forms. *Zeitschrift für Angewandte Mathematik und Mechanik 61* (1981), 161–164.

[102] Schaback, R., and Werner, H. *Numerische Mathematik*, fourth ed. Springer, Berlin, 1992.

[103] Schmidt, R. O. Multiple Emitter Location and Signal Parameter Estimation. *IEEE Transactions on Antennas and Propagation 34*, 3 (1986), 276–280.

[104] Schneider, C., and Werner, W. Some New Aspects of Rational Interpolation. *Mathematics of Computation 47*, 175 (1986), 285–299.

[105] Schönhage, A. Quasi-GCD Computations. *Journal of Complexity 1* (1985), 118–137.

[106] Stahl, H. Spurious Poles in Padé Approximation. *Journal of Computational and Applied Mathematics 99* (1998), 511–527.

[107] Stoer, J. *Einführung in die Numerische Mathematik I*, fourth ed., vol. 105 of *Heidelberger Taschenbücher*. Springer, Berlin, 1983.

[108] Taylor, W. J. Method of Lagrangian Curvilinear Interpolation. *Journal of Research of the National Bureau of Standards 35* (1945), 151–155.

[109] Trefethen, L. N. *Approximation Theory and Approximation Practice*. Society for Industrial and Applied Mathematics, Philadelphia, 2020.

[110] Trefethen, L. N., and Bau, D. *Numerical Linear Algebra*. Society for Industrial and Applied Mathematics, Philadelphia, 1997.

[111] Vetterli, M., Marziliano, P., and Blu, T. Sampling Signals With Finite Rate of Innovation. *IEEE Transactions on Signal Processing 50*, 6 (2002), 1417–1428.

[112] Wei, X., and Dragotti, P. L. FRESH-FRI-Based Single-Image Super-Resolution Algorithm. *IEEE Transactions on Image Processing 25*, 8 (2016), 3723—3735.

[113] Winrich, L. B. Note on a Comparison of Evaluation Schemes for the Interpolating Polynomial. *The Computer Journal 12*, 2 (1969), 154–155.

[114] Wuytack, L. On Some Aspects of the Rational Interpolation Problem. *SIAM Journal on Numerical Analysis 11*, 1 (1974), 52–60.

[115] Young, R. M. *An Introduction to Nonharmonic Fourier Series.* Academic Press, San Diego, 2001. Revised First Edition.

[116] Zhu, X., and Zhu, G. A Method for Directly Finding the Denominator Values of Rational Interpolants. *Journal of Computational and Applied Mathematics 148* (2002), 341–348.