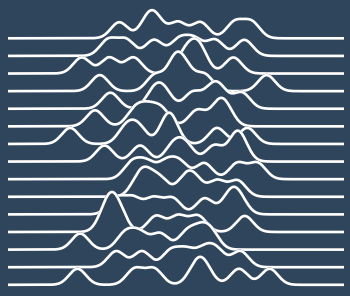
MASTER'S THESIS

# Randomized Estimation of Spectral Densities



*Fabio Matti*

supervised by

Prof. Daniel Kressner & Haoze He

from the

Chair of Numerical Algorithms and High-Performance Computing

at the

École Polytechnique Fédérale de Lausanne

February 25, 2025

# Acknowledgments

I would like to acknowledge the support I was given by my supervisor Prof. Daniel Kressner. His clear guidance and unparalleled expertise were the critical success factor for my thesis. I appreciate him always being very generous with his time despite his full schedule. In every meeting he filled my hands with ideas, inspiration, and appropriate criticism. He assigned me a tasks on which I worked well beyond the usual eight hour workday; not because I had to, but because I wanted to and enjoyed every moment of doing so. I am grateful for all the opportunities and favors he has given me during this time.

My co-supervisor, Haoze He, also contributed heavily to me having a great time these last four months. With his tolerance for my mistakes, quick grasp of complicated concepts, and critical thinking skills he was able to support me well in my work. Our many shared interests led to uncountable entertaining conversations. Our collaboration together with Hysan Lam, who also helped me in countless ways, was to me one of the most exciting parts of my work.

I am thankful for all the amazing people I could meet during my thesis. The many interesting conversations I was able have over lunch with the other members of the ANCHP group, namely David, Peter, Margherita, Angelo, and Nian, always inspired and entertained me. The fact that they treated me as one of them made me feel accepted and gave me the confidence to pursue my goals.

Furthermore, I express my gratitude for my teachers, family, and friends. If it was not for the many teachers who have inspired, mentored, and formed me all my life long, I would not have been motivated to follow this path. My parents were always there for me and stood behind me no matter what I did. Being able to count on their kindness and support helped me overcome multiple challenging situations. Thanks to my friends I could experience many great moments and adventures during the time of working on my thesis, particularly with Lucas, Jozef, Bruno, Irvin, Mateja, Bartul, and Paride.

Finally, I admit that the title image of this thesis was inspired by the cover of Joy Division's studio album called Unknown Pleasures. It is a schematic depiction of the spectral densities of 15 different Gaussian random matrices.

# Summary

We develop a family of methods used to approximate the distribution of the eigenvalues – the spectral density – of large symmetric matrices. These methods are based on the polynomial expansion of a smoothing function in combination with either a randomized trace estimation, a randomized low-rank factorization, or both simultaneously.

Initially, we give an introduction and overview of procedures which are used in literature to compute the spectral density of matrices. Then, we proceed by showing how some matrix functions can efficiently be computed based on their Chebyshev expansion using the discrete cosine transform. Together with the Girard-Hutchinson stochastic trace estimator, we construct a first algorithm for approximating the spectral density: the Delta-Gauss-Chebyshev method. Subsequently, an analysis of the structure of a matrix function involved in the computation motivates the usage of a Nyström low-rank factorization for reducing the dimensionality of the problem, which leads us to the Nyström-Chebyshev method. To circumvent the inefficiency of one and the lack of robustness of the other method they are combined into a third algorithm called the Nyström-Chebyshev++ method.

The techniques employed in these methods are motivated and introduced in a rigorous manner. We present multiple implementation strategies for improved computational speed, accuracy, and stability, and give a theoretical analysis of each method. In various experiments the analysis of our algorithms is numerically confirmed and their effectiveness is compared to other conventionally used methods.

**Keywords:** Spectral density, Chebyshev expansion, discrete cosine transform, stochastic trace estimation, Girard-Hutchinson estimator, randomized low-rank factorization, Nyström approximation, matrix functions, parameter dependent matrices

# Résumé

Nous développons une famille de méthodes utilisées pour calculer approximativement la distribution des valeurs propres – la densité spectrale – de grandes matrices symétriques. Ces méthodes sont basées sur l'expansion polynomiale d'une fonction lisse en combinaison avec une estimation probabiliste de la trace, une factorisation probabiliste de rang bas, ou les deux simultanément.

D'abord, nous présentons une introduction et une vue d'ensemble des procédures utilisées dans la littérature pour calculer la densité spectrale des matrices. Ensuite, nous montrons comment certaines fonctions matricielles peuvent être calculées efficacement sur la base de leur expansion de Chebyshev en calculant les transformées en cosinus discrètes. L'utilisation de l'estimateur stochastique de trace de Girard-Hutchinson mène à un premier algorithme d'approximation de la densité spectrale : la méthode Delta-Gauss-Chebyshev. Ensuite, une analyse de la structure de la fonction matricielle impliquée dans le calcul motive l'utilisation d'une factorisation de Nyström de rang bas pour réduire la dimensionnalité du problème, ce qui nous conduit à la méthode de Nyström-Chebychev. Pour contourner l'inefficacité de l'une et le manque de robustesse de l'autre méthode, elles sont combinées en un troisième algorithme appelé méthode Nyström-Chebychev++.

Les techniques employées dans ces méthodes sont motivées et introduites de manière rigoureuse. Nous présentons plusieurs stratégies d'implémentation pour améliorer la vitesse de calcul, la précision et la stabilité, et nous donnons une analyse théorique de chaque méthode. Dans diverses expériences, l'analyse de nos algorithmes est confirmée numériquement et leur efficacité est comparée à d'autres méthodes conventionnelles.

**Mots-clés :** Densité spectrale, expansion de Chebyshev, transformée en cosinus discrète, estimation probabiliste de trace, estimateur de Girard-Hutchinson, factorisation probabiliste de rang bas, approximation de Nyström, fonctions matricielles, matrices dépendantes des paramètres

# About provable reproducibility

How can scientists unmistakably know whether their results can be reproduced by other people? How can reviewers quickly verify that a certain numerical experiment in an article is correct? And how can collaborators quickly understand how to use the source code you have written for your project?

Provable reproducibility is an initiative I hereby launch, which pursues the goal of making results published in articles, theses, and software packages easier to reproduce and verify. No more ambiguity, misunderstandings, cherry-picked parameters, and hand-crafted results. Every figure, plot, and table in a provably reproducible project can be unequivocally traced back to where it originated from. This goes one step further than the practice of open-sourcing the code of a project, where a reviewer or user still has to go through the tedious and often error-prone process of reproducing your results.

Like in a public demonstration of a scientific experiment, the proof is delivered by remotely – not locally – recreating every aspect of a project, for example in an online archive. Everyone can see what steps are taken to achieve a certain outcome. Making a project provably reproducible includes, but is not limited to, explicitly downloading, importing, and building all the external dependencies which are being used (code, data, ...); running all the computations and generating the corresponding results (plots, tables, ...); and compiling the project outcome (article, report, ...). A reproducibility proof can be achieved without too much effort by using the continuous integration frameworks offered in most contemporary software development and version control services, such as GitHub actions or GitLab CI/CD. Corresponding guidelines and resources will be made available in due time[1].

This document is provably reproducible.

```
> hosted at https://github.com/FMatti/Rand-SD
> built on 2024-02-05 at 18:44:29 UTC from ca391a8
```

[1] https://github.com/FMatti/Re-Pro

# Notation

We try to follow the notation and conventions which are used in contemporary literature in this field. That is,

- we exclusively work with objects over the real numbers $\mathbb{R}$ and non-zero integers $\mathbb{N}$;
- scalar are represented by lower case Greek or Latin letters ($s$, $\varepsilon$, ...), vectors are additionally printed in bold ($\boldsymbol{\nu}$, $\boldsymbol{\mu}$, ...), and matrices are additionally capitalized ($\boldsymbol{A}$, $\boldsymbol{\Omega}$, ...);
- the components of a vector $\boldsymbol{\nu} \in \mathbb{R}^n$ are $\nu_i \in \mathbb{R}, i = 1, \dots, n$, and the elements of a matrix $\boldsymbol{A} \in \mathbb{R}^{n \times n}$ are $a_{ij} \in \mathbb{R}, i, j = 1, \dots, n$;
- the identity matrix $\mathbf{I}_n = \operatorname{diag}(1, \dots, 1) \in \mathbb{R}^{n \times n}$ carries ones on its diagonal and zeros everywhere else. The zero matrix $\mathbf{0}_{n \times m} \in \mathbb{R}^{n \times m}$ consists of only zero entries;
- the eigenvalues of a square matrix $\boldsymbol{A} \in \mathbb{R}^{n \times n}$ are all scalars $\lambda$ which, together with some non-zero vectors $\boldsymbol{\nu}$, satisfy the condition $\boldsymbol{A}\boldsymbol{\nu} = \lambda\boldsymbol{\nu}$. We denote them as $\lambda_1 \geqslant \cdots \geqslant \lambda_n$ or, if we know them to be non-negative, as $\sigma_1 \geqslant \cdots \geqslant \sigma_n \geqslant 0$;
- the trace of a square matrix $\boldsymbol{A} \in \mathbb{R}^{n \times n}$ is the sum of its diagonal elements $\operatorname{Tr}(\boldsymbol{A}) = \sum_{i=1}^{n} a_{ii}$ but also the sum of its eigenvalues $\operatorname{Tr}(\boldsymbol{A}) = \sum_{i=1}^{n} \lambda_i$;
- the transpose of a matrix $\boldsymbol{A}$ is denoted with $\boldsymbol{A}^\top$. Real symmetric matrices satisfy $\boldsymbol{A}^\top = \boldsymbol{A}$ and allow for a spectral decomposition $\boldsymbol{A} = \boldsymbol{U}\boldsymbol{\Lambda}\boldsymbol{U}^\top$ with $\boldsymbol{\Lambda} = \operatorname{diag}(\lambda_1, \dots, \lambda_n) \in \mathbb{R}^{n \times n}$ and $\boldsymbol{U} \in \mathbb{R}^{n \times n}$ such that $\boldsymbol{U}^\top\boldsymbol{U} = \mathbf{I}_n$. You may assume every symbol $\boldsymbol{A}$ you encounter in this thesis to represent a real symmetric matrix;
- matrix norms are denoted with $\|\cdot\|$.. For symmetric matrices, the Schatten norm is $\|\boldsymbol{A}\|_{(p)} = (\sum_{i=1}^{n} |\lambda_i|^p)^{1/p}$. We refer to the spectral norm as $\|\boldsymbol{A}\|_2 = \|\boldsymbol{A}\|_{(\infty)} = \max_{i=1,\dots,n} |\lambda_i|$, the nuclear norm as $\|\boldsymbol{A}\|_* = \|\boldsymbol{A}\|_{(1)} = \sum_{i=1}^{n} |\lambda_i|$, and the Frobenius norm is $\|\boldsymbol{A}\|_F = \|\boldsymbol{A}\|_{(2)} = (\sum_{i=1}^{n} |\lambda_i|^2)^{\frac{1}{2}}$;

- the pseudoinverse of a matrix is denoted with $\mathbf{A}^\dagger$. It satisfies $\mathbf{A}^\dagger = (\mathbf{A}^\top \mathbf{A})^\dagger \mathbf{A}^\top$, and if $\mathbf{A}$ has linearly independent columns, it acts as a left inverse in the sense that $\mathbf{A}^\dagger \mathbf{A} = \mathbf{I}_n$ [33];
- a matrix $\mathbf{B} \in \mathbb{R}^{n \times n}$ is said to be positive semi-definite if its eigenvalues are all non-negative. You may assume every symbol $\mathbf{B}$ you encounter in this thesis to denote a real positive semi-definite matrix;
- closed interval are written as $[a, b] \subset \mathbb{R}$ and open intervals as $(a, b) \subset \mathbb{R}$;
- the $L^p$-norm of a real-valued, integrable function $f$ on $[-1, 1]$ is defined as $\|f\|_p = (\int_{-1}^{1} |f(s)|^p ds)^{1/p}$ with respect to the Lebesgue integral. Further, we define $\|f\|_\infty = \sup_{s \in [-1,1]} |f(s)|$;
- convolutions between two functions $f : \mathbb{R} \to \mathbb{R}$ and $g : \mathbb{R} \to \mathbb{R}$ can be defined as $(f * g)(t) = \int_{-\infty}^{\infty} f(s) g(t - s) ds$;
- the expected value is denoted with $\mathbb{E}$ and the variance with Var;
- Euler's number is printed as $e$ and its corresponding natural logarithm as log;
- Dirac's delta distribution is written as $\delta$ and satisfies $\int_{-\infty}^{\infty} f(s)\delta(t-s) = f(t)$ as well as $\delta(cs) = \frac{1}{|c|}\delta(s)$ for any $c \in \mathbb{R} \setminus 0$ [9, chapter 15]. The Kronecker delta $\delta_{ij}$ is 1 if $i = j$, otherwise 0;
- the trigonometric cosine function is denoted with cos and its inverse with arccos.
- the real part of a complex number $z$ is $\Re(z)$ while its imaginary part is $\Im(z)$. The complex unit is printed as $\imath$;
- we use $\mathcal{O}$ to describe the asymptotic lower and upper bounds on the complexity of a computation. For those more familiar with the Bachmann-Landau notation [7, section 3.2], please allow us – for simplicity – to use $\mathcal{O}$ as though it was $\Theta$;
- $c(n)$ denotes the computational complexity of a matrix-vector product;
- and finally, we call an operation *prohibitively expensive*, if its complexity scales more than quadratically with the size of the problem.

# Contents

# Chapter 1

# Introduction

In many problems in physics, chemistry, engineering, and computer science, the eigenvalues of certain matrices help understand the nature of a system: In electronic structure calculations they represent the allowed energy levels electrons can occupy [10, 16, 27]; in neural network optimization they are indicative of the optimization speed [4, 6, 36]; and in graph processing they can uncover hidden graph motifs [24, 21, 32]. However, computing the eigenvalues of a matrix can be prohibitively expensive. Furthermore, when analyzing these systems, it is often not crucial to know the exact individual eigenvalues, but more so their approximate locations with respect to each other, such as eigenvalue clusters or spectral gaps.

The goal of spectral density theory is to find the approximate distribution of the eigenvalues of large matrices. In this introductory chapter we define the spectral density of a matrix, give an overview of the most common ways of approximating it, and show how the work in this thesis is embedded in current research on spectral density approximations.

## 1.1 Spectral density

In most applications, the studied matrices are real, i.e. $\mathbf{A} \in \mathbb{R}^{n\times n}$, and symmetric, i.e. $\mathbf{A}^\top = \mathbf{A}$, such that their eigenvalues $\lambda_1, \ldots, \lambda_n$ are all real. This allows us to define their spectral densities on the real line.

**Definition 1.1: Spectral density of a matrix**

Let $\mathbf{A} \in \mathbb{R}^{n\times n}$ be a symmetric matrix with eigenvalues $\lambda_i \in \mathbb{R}, i = 1, \ldots, n$. We define the spectral density $\phi$ as the functional

$$\phi(t) = \frac{1}{n}\sum_{i=1}^{n} \delta(t - \lambda_i), \tag{1.1}$$

involving the Dirac delta distribution $\delta$ and a spectral parameter $t \in \mathbb{R}$.

$\phi$ is the probability density [23] of the cumulative empirical spectral measure [6]. Knowing $\phi$ of a matrix would – in theory – allow us to reconstruct its eigenvalues. Furthermore, $\phi$ can for example be used to count the number of eigenvalues of a matrix which lie within an interval $[a, b]$

$$\nu_{[a,b]} = n \int_a^b \phi(t)dt, \tag{1.2}$$

or to compute the trace of a matrix function [27]

$$\mathrm{Tr}(f(\boldsymbol{A})) = \sum_{i=1}^{n} f(\lambda_i) = n \int_{-\infty}^{\infty} f(t)\phi(t)dt. \tag{1.3}$$

Constructing the spectral density $\phi$ (definition 1.1) of a matrix amounts to knowing all its eigenvalues, which are often prohibitively expensive to compute. Since we can neither hope to measure the convergence of a smooth approximation to $\phi$ in any of the conventionally used $L^p$-norms, nor visualize the spectral density in a simple and easily interpretable plot, we work with a smooth version of the spectral density.

## 1.2 SMOOTH SPECTRAL DENSITY

We regularize $\phi$ with a suitable smoothing kernel $g_\sigma$ to define the smooth spectral density $\phi_\sigma$ as the convolution

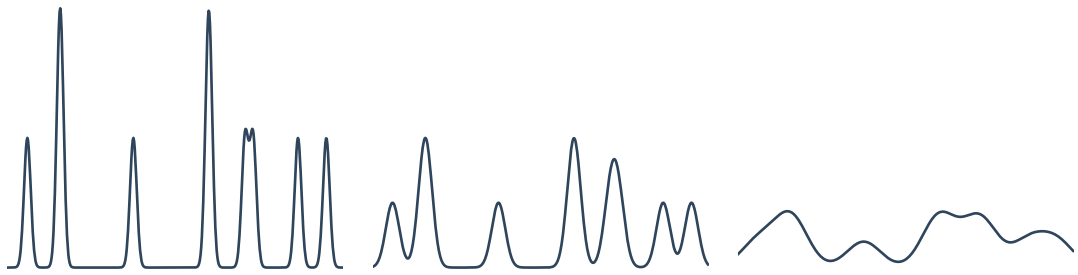
$$\phi_\sigma(t) = (\phi * g_\sigma)(t) = \int_{-\infty}^{\infty} \phi(s) g_\sigma(t-s)ds = \frac{1}{n} \sum_{i=1}^{n} g_\sigma(t-\lambda_i). \tag{1.4}$$

The smoothing parameter $\sigma > 0$ controls by how much $\phi$ is smoothed (see figure 1.1). Typically, large $\sigma$ allow for easier approximations of $\phi_\sigma$ but at the cost of losing many of the finer characteristics of the spectrum.

Commonly, $g_\sigma$ is chosen to be a Gaussian of width $\sigma$

$$g_\sigma(s) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{s^2}{2\sigma^2}}, \tag{1.5}$$

due to its rapidly decaying tail and desirable interpolation properties. There are many other choices possible for $g_\sigma$. Ideally, $g_\sigma$ should be smooth, symmetric, non-negative, and tend – in a weak sense – towards the Dirac delta distribution $\delta$ in the limit of $\sigma \to 0$. Another commonly used kernel is the Lorentzian, which



(A) small $\sigma$ (B) medium $\sigma$ (C) large $\sigma$

FIGURE 1.1 – Schematic depiction of the spectral density of a simple matrix with 10 eigenvalues for different values of the smoothing parameter $\sigma$.

will be discussed in an example in section 5.2.

Because we only consider symmetric matrices, we may represent $\mathbf{A} \in \mathbb{R}^{n\times n}$ using its spectral decomposition $\mathbf{A} = \mathbf{U}\mathbf{\Lambda}\mathbf{U}^\top$ where $\mathbf{\Lambda} \in \mathbb{R}^{n\times n}$ is the diagonal matrix which carries the eigenvalues of $\mathbf{A}$ on its diagonal and $\mathbf{U} \in \mathbb{R}^{n\times n}$ is orthonormal, i.e. $\mathbf{U}^\top\mathbf{U} = \mathbf{I}_n$ [20, theorem 4.1.5]. Thus, if we absorb the $1/n$ factor from (1.4) in $g_\sigma$, use the definition of a smooth function $f$ applied to a symmetric matrix $f(\mathbf{A}) = \mathbf{U}f(\mathbf{\Lambda})\mathbf{U}^\top$, with $f(\mathbf{\Lambda}) = \operatorname{diag}(f(\lambda_1), \dots, f(\lambda_n))$ [19, definition 1.2], and finally use the invariance of the trace under orthonormal transformations, we may write

$$\phi_\sigma(t) = \sum_{i=1}^{n} g_\sigma(t - \lambda_i) = \operatorname{Tr}(g_\sigma(t\mathbf{I}_n - \mathbf{A})). \tag{1.6}$$

In this way, we have just converted the problem of computing the eigenvalues of a matrix to computing the trace of a function applied to the same matrix, for which – we will see – exist many efficient algorithms.

## 1.3 OVERVIEW OF THE METHODS DEVELOPED IN THIS THESIS

The three methods we will develop in this thesis are based on [27]. All of them are closely related to each other. In a first stage, they all expand the matrix function $g_\sigma(t\mathbf{I}_n - \mathbf{A})$ from (1.6) in a truncated basis of Chebyshev polynomials. The Delta-Gauss-Chebyshev (DGC) method then proceeds by approximating the trace in (1.6) with a stochastic trace estimator. The Nyström-Chebyshev (NC) method instead expresses the expanded matrix function as a product of smaller matrices, for which we can directly compute the trace. Finally, the Nyström-Chebyshev++ (NC++) method combines the DGC and NC method by first factorizing the matrix and subsequently correcting for the approximation error by

estimating the trace of the residual of this approximation with a stochastic trace estimator. In fact, if we denote with $\widetilde{\phi}_\sigma^{(m)}$ the approximation of $\phi_\sigma$ computed with the DGC method, $\widehat{\phi}_\sigma^{(m)}$ the one computed with the NC method, and $\check{\phi}_\sigma^{(m)}$ the one with NC++, we can show that

$$\check{\phi}_\sigma^{(m)} = \widetilde{\phi}_\sigma^{(m)} + \widehat{\phi}_\sigma^{(m)} - \text{some additional term.} \tag{1.7}$$

It will turn out that the additional term can be determined by applying the DGC method to the matrix factorization produced in the NC method. A schematic overview of the relation between these methods can be found in figure 1.2.

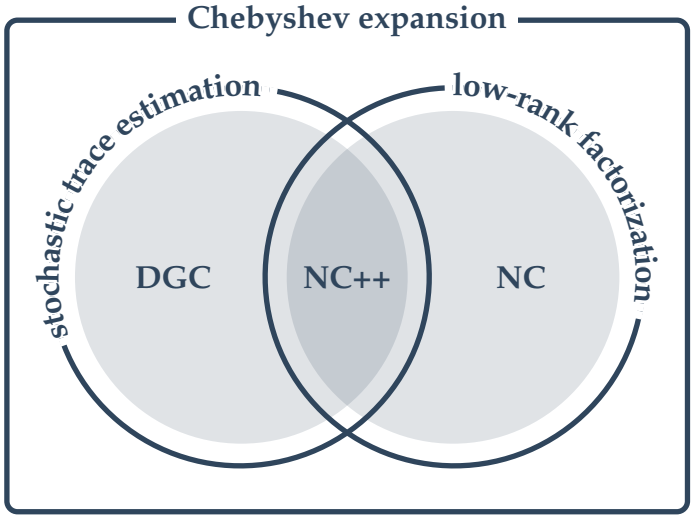



FIGURE 1.2 – Overview of the techniques used in the three methods developed in this thesis: the Delta-Gauss-Chebyshev (DGC), Nyström-Chebyshev (NC), and Nyström-Chebyshev++ (NC++) methods.

The DGC method from [27] is very similar to our version of this method, except that we employ a more rigorous and faster approach when computing the Chebyshev expansion. [27] also proposes the spectrum sweeping (SS) method, as well as the robust and efficient spectrum sweeping (RESS) method. We decide to not discuss the SS method, since we have found an equivalent version of this method which incorporates the "efficient" part from the RESS method at no loss of accuracy. We call it the NC method. Finally, the NC++ method is derived from the RESS method. Both the NC and NC++ methods feature major algorithmic improvements compared to their ancestors.

## 1.4 RELATED WORK

Multiple approaches have been taken for approximating the spectral density $\phi$ of a large symmetric matrix. Most of which can loosely be grouped into two big families of methods (see also figure 1.3 for an illustrated overview of the methods):

1. Lanczos-based methods first partially tridiagonalize the matrix and subsequently either extract Ritz values or directly exploit properties of the smoothing kernel $g_\sigma$ to approximate $\phi$;

2. and expansion-based methods, which either compute a truncated expansion of $\phi$ or $g_\sigma$ in a polynomial basis and subsequently make use of trace estimation to approximate $\phi$.

Among the Lanczos-based methods features the Haydock method [16, 28]. It stochastically estimates the trace of the Lorentzian kernel applied to a matrix by first tridiagonalizing the matrix with a few iterations of the Lanczos algorithm [25] and subsequently evaluating the matrix function using a continued fraction formula. The stochastic Lanczos quadrature [28, 42, 6] also first tridiagonalizes the matrix. It then extracts the nodes and weights for the corresponding Gaussian quadrature to build an approximate spectral density. The expansion-based methods encompass the kernel polynomial methods [37, 43, 44] which involve the formal expansion of a suitable modification of $\phi$ in a basis of orthogonal polynomials. The Delta-Gauss-Legendre [28] and Delta-Gauss-Chebyshev [27] methods instead expand $g_\sigma$ and use this expansion to efficiently evaluate a stochastic trace estimator.

It turns out that these two families are not distinct, and in fact some of these methods can be shown to be equivalent to each other: The Lanczos algorithm can be used as an engine for the kernel polynomial method [5], while smoothing the approximation resulting from the kernel polynomial method will give the same result as the Delta-Gauss-Legendre and Delta-Gauss-Chebyshev methods [28].

Spectral densities of matrices have found use in priming the computation of matrix functions [12] and in parallelized eigenvalue solvers [35, 26]. In recent years, significant progress in the theory of stochastic trace estimation, which is the backbone of most expansion-based methods for approximating spectral densities, has been made [30, 34]. These developments involved demonstrating the reciprocal decrease of the approximation error with the number of matrix-vector multiplication queries used, which is a highly desirable algorithmic property.

## 1.5 Main contributions

Besides a couple of small clarifications to [27], we see our main contributions to be

- the development of a simple and consistent discrete cosine transform (DCT)-based interpolation scheme which allows us to speed up [27, algorithm 5] by orders of magnitude at no loss of accuracy;

- the proposal of multiple algorithmic improvements and speed-ups for all methods from [27];
- the derivation of an a priori guarantee for the number of matrix-vector products needed to get an accurate approximation of the spectral density under certain assumptions;
- proposal of a simple generalization of the presented methods to other commonly used randomized low-rank approximation schemes;
- a fast and rigorously documented implementation of all the algorithms following the notation and conventions of this thesis, which can be used to reproduce every plot, table, and much more[1].

## 1.6 STRUCTURE OF THE THESIS

This paragraph finishes off the introductory chapter 1. In chapter 2 we discuss the Chebyshev expansion, present an efficient way of computing it, and study its convergence. We also take a look at stochastic trace estimation to then construct a first algorithm, the Delta-Gauss-Chebyshev (DGC) method, using these tools. Chapter 3 is dedicated to the use of randomized low-rank factorization of matrices for computing spectral densities, which gives rise to a second algorithm, the Nyström-Chebyshev (NC) method. Putting the ideas from the two previously discussed algorithms together, we end up with a fast and general algorithm, the Nyström-Chebyshev++ (NC++) method, for computing spectral densities of large matrices in chapter 4. In chapter 5, we study the accuracy and computational time of these algorithms on numerous numerical experiments. We conclude the thesis in chapter 6.

[1] A GitHub repository with the code which allows to easily reproduce this entire document is available at https://github.com/FMatti/Rand-SD.

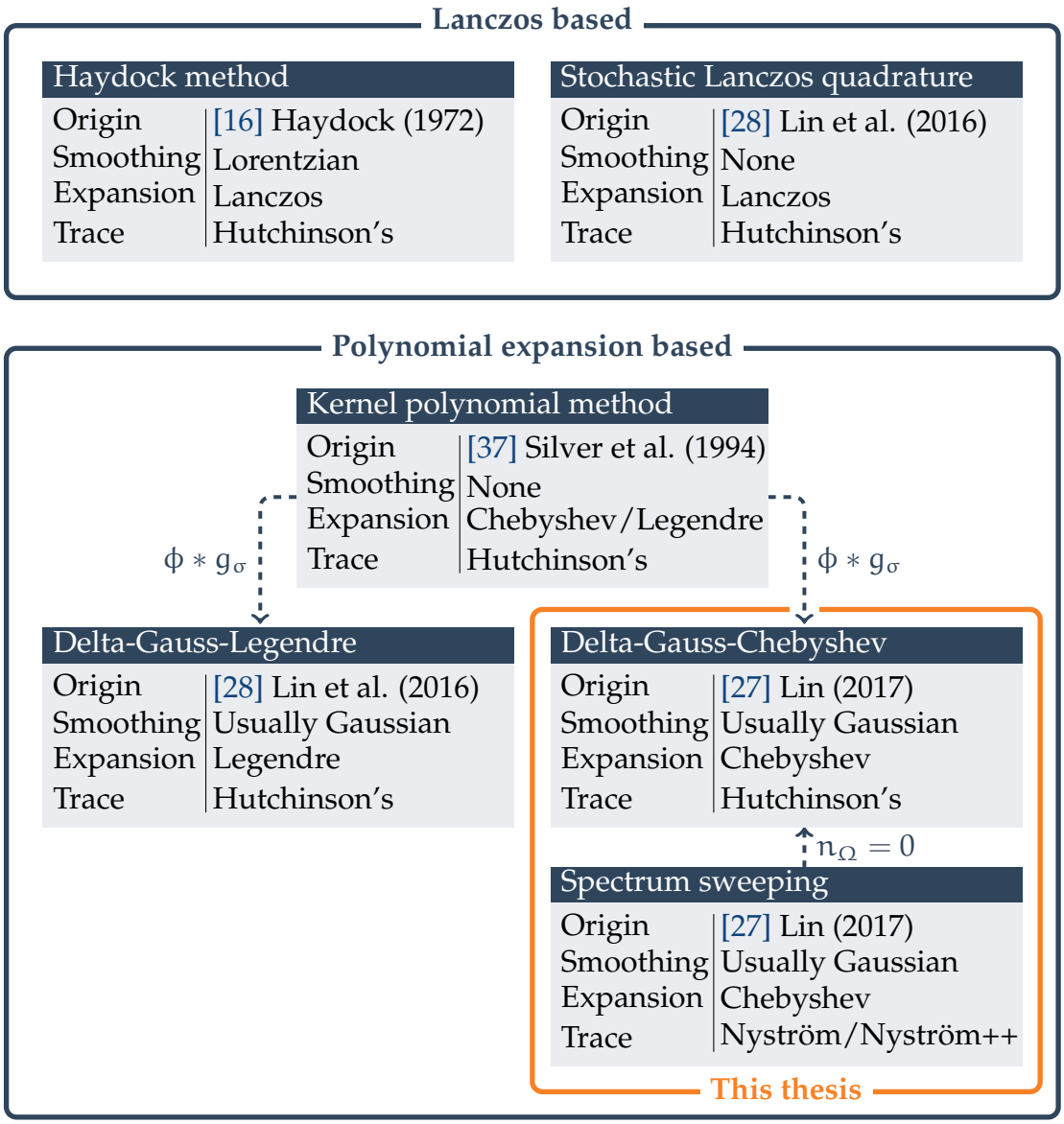



FIGURE 1.3 – An overview of some popular methods for approximating spectral densities of large symmetric matrices. For each method, the publication where we have encountered them in, the type of smoothing it applies, what kind of expansion is used, and how the trace in (1.6) is approximated are mentioned. Relations between the methods are indicated with dashed lines.

# Chapter 2

# Interpolation and trace estimation

For many smoothing kernel $g_\sigma$, as is for example the case with Gaussian smoothing (1.5), we cannot compute the matrix function

$$g_\sigma(t\mathbf{I}_n - \mathbf{A}) \tag{2.1}$$

involved in (1.6) without first diagonalizing $\mathbf{A}$, which can be prohibitively expensive for large $\mathbf{A}$. A way around this problem is to refrain from trying to assemble the matrix function explicitly and instead use the fact that thanks to (1.6) we are only interested in its trace. Approximating the trace can be done by multiplying the matrix with random vectors, to then – for example – evaluate the Girard-Hutchinson trace estimator [22]. The multiplication of a matrix function with a vector can often be determined quite efficiently using Krylov subspace methods [19, chapter 13.2]. Another way in which products of matrix functions with vectors can be computed involves expanding the function in terms of a finite set of orthogonal polynomials and subsequently using a recurrence relation to efficiently construct the result. This approach turns out to be particularly effective when we work with matrix functions which smoothly depend on a parameter within a bounded interval, and if we want to evaluate the function at a large number of values of this parameter. In this chapter we will analyze one such expansion, the Chebyshev expansion, which gives rise to an efficient method for approximating the spectral density, particularly when the number of evaluation points $n_t$ is large.

## 2.1 Chebyshev interpolation

The Chebyshev interpolation framework is best known for its stability, beneficial convergence properties, and simple three-term recurrence relation (2.3) which can be exploited to efficiently compute products of Chebyshev polynomials with vectors.

At the foundation of Chebyshev interpolation lie the Chebyshev polynomials $T_l$. They are defined for all $l \in \mathbb{N}$ as [39, chapter 3]

$$\begin{cases} T_l : [-1,1] \to [-1,1] \\ T_l(s) = \cos(l \arccos(s)). \end{cases} \tag{2.2}$$

They satisfy the three-term recurrence relation

$$\begin{cases} T_0(s) = 1 & l = 0, \\ T_1(s) = s & l = 1, \\ T_l(s) = 2sT_{l-1}(s) - T_{l-2}(s) & l \geqslant 2, \end{cases} \tag{2.3}$$

which can be shown using their definition (2.2) and a standard trigonometric identity.

A function $f : [-1,1] \to \mathbb{R}$ can be expanded in a basis of Chebyshev polynomials up to degree of expansion $m \in \mathbb{N}$ [39, chapter 3]

$$f^{(m)}(s) = \sum_{l=0}^{m} \mu_l T_l(s). \tag{2.4}$$

For functions $f$ which can be analytically extended in a certain neighborhood of $[-1,1]$ in the complex plane, Bernstein's theorem [38, theorem 4.3] establishes that the convergence of this expansion in the $L^\infty$-norm is exponential.

The coefficients $\{\mu_l\}_{l=0}^m$ in (2.4) could be computed using the orthogonality of the Chebyshev polynomials with respect to a certain inner product, and some authors indeed suggest approximating the involved integral using a quadrature rule with evenly spaced nodes [27, equation 8]. However, we cannot find a good theoretical guarantee that this will be accurate. Instead, we use a significantly simpler, faster, and provably accurate way of computing the coefficients of the Chebyshev expansion of a function $f : [-1,1] \to \mathbb{R}$ [39, 1]: If the values $f^{(m)}(s_i)$ at some $m+1$ distinct points $\{s_i\}_{i=0}^m$ are known, the coefficients $\{\mu_l\}_{l=0}^m$ of this polynomial are uniquely determined [13]. For the choice $s_i = \cos(\pi i/m), i = 0, \ldots, m$, (2.4) reads

$$f^{(m)}(s_i) = \sum_{l=0}^{m} \mu_l \cos\left(\frac{\pi i l}{m}\right), \tag{2.5}$$

which coincides with a discrete cosine transform (DCT)[1] of the coefficients $\{\mu_l\}_{l=0}^m$. Thus, if we collect the coefficients $\{\mu_l\}_{l=0}^m$ in a vector $\boldsymbol{\mu} \in \mathbb{R}^{m+1}$ and the function

[1]There exist multiple conventions for the DCT. The one which we use is (up to scaling of the first and last coefficient) referred to as a type I DCT, and is efficiently implemented in the SciPy Python package: https://docs.scipy.org/doc/scipy/reference/generated/scipy.fft.dct.html.

evaluations $\{f^{(m)}(s_i)\}_{i=0}^m$ in another vector $\mathbf{f} \in \mathbb{R}^{m+1}$, we can pass back and forth between the two with

$$\mathbf{f} = \mathrm{DCT}\{\boldsymbol{\mu}\} \implies \boldsymbol{\mu} = \mathrm{DCT}^{-1}\{\mathbf{f}\}. \tag{2.6}$$

In short, computing the coefficients of the Chebyshev expansion (2.4) of the function $f$ amounts to evaluating this function at $m+1$ well chosen points and computing the inverse DCT. The corresponding algorithm can be found in algorithm 2.1. This procedure is usually inexpensive and can be done in $\mathcal{O}(m \log(m))$ operations [29].

**Algorithm 2.1: Chebyshev expansion**

**Input:** Function $f : [-1, 1] \to \mathbb{R}$
**Parameters:** degree of expansion $m$
**Output:** Coefficients $\{\mu_l\}_{l=0}^m$ of the expansion $f^{(m)}(s) = \sum_{l=0}^m \mu_l T_l(s)$

1: Evaluate $f$ at $\{\cos(\pi i/m)\}_{i=0}^m$ and store the results in $\mathbf{f} \in \mathbb{R}^{m+1}$
2: Compute $\boldsymbol{\mu} = \mathrm{DCT}^{-1}(\mathbf{f})$
3: Let $\mu_l = (\boldsymbol{\mu})_l$ for $l = 0, \ldots, m$

To demonstrate the higher efficiency of this DCT-based method, we time it against the corresponding algorithm from [27]. The results can be seen in table 2.1.

TABLE 2.1 – Comparison of the runtime in milliseconds of the two approaches for computing the coefficients of the Chebyshev expansion of a function. We average over 7 runs of the algorithms and repeat these runs 1000 times to form the mean and standard deviation which are given in the below table. We refer to [27, algorithm 1] with "quadrature" and to algorithm 2.1 with "DCT". For each algorithm, we interpolate a Gaussian $g_\sigma$ with $\sigma = 0.05$, at $n_t = 1000$ points, for various values of $m$.

| | $m = 800$ | $m = 1600$ | $m = 2400$ | $m = 3200$ |
|---|---|---|---|---|
| quadrature | $286.2 \pm 2.6$ | $1716.0 \pm 6.0$ | $1048.7 \pm 5.7$ | $1660.2 \pm 6.2$ |
| DCT | $99.8 \pm 33.8$ | $126.8 \pm 1.1$ | $169.5 \pm 1.8$ | $208.8 \pm 1.0$ |

## 2.2 STOCHASTIC TRACE ESTIMATION

Matrix-free stochastic trace estimation is most useful when a matrix is not given explicitly, but products of this matrix with vectors can be computed efficiently. Examples of such scenarios are traces of matrix functions [42, 11] or of implicit matrices which can only be queried through matrix-vector products [4, 36]. Most algorithms for stochastic trace estimation are based on the Girard-Hutchinson

trace estimator, which we will discuss in the following paragraphs.

### 2.2.1 CONSTANT MATRICES

For a symmetric matrix $\mathbf{B} \in \mathbb{R}^{n\times n}$ and a standard Gaussian random vector $\boldsymbol{\psi} \in \mathbb{R}^n$, the quadratic form $\boldsymbol{\psi}^\top \mathbf{B}\boldsymbol{\psi}$ is an unbiased estimate of the trace:

$$\mathbb{E}[\boldsymbol{\psi}^\top \mathbf{B}\boldsymbol{\psi}] = \mathbb{E}\left[\sum_{i=1}^{n}\sum_{j=1}^{n} \psi_i b_{ij} \psi_j\right] = \sum_{i=1}^{n}\sum_{j=1}^{n} b_{ij}\mathbb{E}[\psi_i\psi_j] = \sum_{i=1}^{n} b_{ii} = \operatorname{Tr}(\mathbf{B}). \tag{2.7}$$

Furthermore, the variance of this estimate is bounded by the Frobenius norm of the matrix $\mathbf{B}$:

$$\begin{aligned}
\operatorname{Var}(\boldsymbol{\psi}^\top \mathbf{B}\boldsymbol{\psi}) &= \operatorname{Var}(\boldsymbol{\psi}^\top \mathbf{U}\boldsymbol{\Lambda}\mathbf{U}^\top\boldsymbol{\psi}) && \text{(spectral decomposition of } \mathbf{B}) \\
&= \operatorname{Var}(\widetilde{\boldsymbol{\psi}}^\top \boldsymbol{\Lambda}\widetilde{\boldsymbol{\psi}}) && (\mathbf{U}^\top\boldsymbol{\psi} = \widetilde{\boldsymbol{\psi}} \sim \boldsymbol{\psi}) \\
&= \mathbb{E}[(\widetilde{\boldsymbol{\psi}}^\top \boldsymbol{\Lambda}\widetilde{\boldsymbol{\psi}})^2] - \mathbb{E}[\widetilde{\boldsymbol{\psi}}^\top \boldsymbol{\Lambda}\widetilde{\boldsymbol{\psi}}]^2 && \text{(definition of variance)} \\
&= \mathbb{E}\left[\left(\sum_{i=1}^{n} \widetilde{\psi}_i^2\lambda_i\right)^2\right] - \operatorname{Tr}(\mathbf{B})^2 && (\boldsymbol{\Lambda} \text{ diagonal and (2.7)}) \\
&= \sum_{i=1}^{n}\lambda_i \sum_{j=1}^{n}\lambda_j \mathbb{E}[\widetilde{\psi}_i^2\widetilde{\psi}_j^2] - \operatorname{Tr}(\mathbf{B})^2 && \text{(linearity of } \mathbb{E}) \\
&= \sum_{i=1}^{n}\lambda_i \sum_{j=1}^{n}\lambda_j + 2\sum_{i=1}^{n}\lambda_i^2 - \operatorname{Tr}(\mathbf{B})^2 && (\mathbb{E}[\widetilde{\psi}_i^2] = 1 \text{ and } \mathbb{E}[\widetilde{\psi}_i^4] = 3) \\
&= \operatorname{Tr}(\mathbf{B})^2 + 2\|\mathbf{B}\|_F^2 - \operatorname{Tr}(\mathbf{B})^2 && (\sum_{i=1}^{n}\lambda_i = \operatorname{Tr}(\mathbf{B}),\ \sum_{i=1}^{n}\lambda_i^2 = \|\mathbf{B}\|_F^2) \\
&= 2\|\mathbf{B}\|_F^2.
\end{aligned}$$

The idea of the Girard-Hutchinson trace estimator is to compute multiple such estimates for different, independent random vectors and take the average. This will again be an unbiased estimate of the trace, but with the reduced variance

$$\operatorname{Var}\left(\frac{1}{n_\Psi}\sum_{j=1}^{n_\Psi} \boldsymbol{\psi}_j^\top \mathbf{B}\boldsymbol{\psi}_j\right) = \frac{2}{n_\Psi}\|\mathbf{B}\|_F^2 \tag{2.8}$$

with the number of Hutchinson's queries $n_\Psi \in \mathbb{N}$. Collecting the $n_\Psi$ independent random vectors $\boldsymbol{\psi}_i \in \mathbb{R}^n$ in the standard Gaussian random matrix $\boldsymbol{\Psi} = [\boldsymbol{\psi}_1, \ldots, \boldsymbol{\psi}_{n_\Psi}] \in \mathbb{R}^{n\times n_\Psi}$, we can then rewrite the Girard-Hutchinson trace estimator as

$$H_{n_\Psi}(\mathbf{B}) = \frac{1}{n_\Psi}\operatorname{Tr}(\boldsymbol{\Psi}^\top \mathbf{B}\boldsymbol{\Psi}). \tag{2.9}$$

### 2.2.2 PARAMETER-DEPENDENT MATRICES

In the case where the matrix, or – alternatively said – all its entries, continuously depends on a parameter in a bounded interval, we can analogously define the Girard-Hutchinson estimator for parameter-dependent matrices

$$\mathrm{H}_{n_\Psi}(\mathbf{B}(t)) = \frac{1}{n_\Psi} \operatorname{Tr}(\mathbf{\Psi}^\top \mathbf{B}(t) \mathbf{\Psi}). \tag{2.10}$$

As the counterpart of the variance in the parametrized case, we measure the error of this estimate in the $L^1$-norm, for which we can use a result from [17], which we will state in the following lemma.

**Lemma 2.2: $L^1$-error of parameter-dependent Girard-Hutchinson estimator**

Let $\mathbf{B}(t) \in \mathbb{R}^{n \times n}$ symmetric and continuous in $t \in [a, b]$, $\delta \in (0, e^{-1})$, and $n_\Psi \in \mathbb{N}$. Let $\mathrm{H}_{n_\Psi}(\mathbf{B}(t))$ be the $n_\Psi$-query Girard-Hutchinson estimator (2.10). With the constant $c_\Psi = 24e$, it holds with probability $\geqslant 1 - \delta$

$$\int_a^b |\operatorname{Tr}(\mathbf{B}(t)) - \mathrm{H}_{n_\Psi}(\mathbf{B}(t))| \, dt \leqslant c_\Psi \frac{\log(1/\delta)}{\sqrt{n_\Psi}} \int_a^b \|\mathbf{B}(t)\|_F dt. \tag{2.11}$$

## 2.3 THE DELTA-GAUSS-CHEBYSHEV METHOD

Now we have all the ingredients for constructing a first algorithm to approximate the expression (1.6): the Chebyshev expansion of a function (algorithm 2.1) and the Girard-Hutchinson trace estimator (2.10). For a symmetric matrix $\mathbf{A} \in \mathbb{R}^{n \times n}$ with eigenvalues contained in $[-1, 1]$ we expand the smoothing kernel $g_\sigma$ in terms of Chebyshev polynomials, such that

$$g_\sigma^{(m)}(t\mathbf{I}_n - \mathbf{A}) = \sum_{l=0}^{m} \mu_l(t) T_l(\mathbf{A}). \tag{2.12}$$

Plugging this expansion into (1.6) gives us the expanded spectral density

$$\phi_\sigma^{(m)}(t) = \operatorname{Tr}(g_\sigma^{(m)}(t\mathbf{I}_n - \mathbf{A})). \tag{2.13}$$

By combining the Chebyshev expansion (2.13) with stochastic trace estimation we end up with the Delta-Gauss-Chebyshev (DGC) method [27, algorithm 2], which approximates the smooth spectral density $\phi_\sigma$ as

$$\widetilde{\phi}_\sigma^{(m)}(t) = \mathrm{H}_{n_\Psi}(g_\sigma^{(m)}(t\mathbf{I}_n - \mathbf{A})) = \frac{1}{n_\Psi} \sum_{l=0}^{m} \mu_l(t) \operatorname{Tr}(\mathbf{\Psi}^\top T_l(\mathbf{A}) \mathbf{\Psi}). \tag{2.14}$$

Apparently, it is rather cheap to evaluate $\widetilde{\phi}_\sigma^{(m)}(t)$ at multiple values of $t$, since only the coefficients of the linear combination of $\{\mathrm{Tr}(\boldsymbol{\Psi}^\top T_l(\boldsymbol{A})\boldsymbol{\Psi})\}_{l=0}^m$ change, which can easily be computed using algorithm 2.1.

An efficient implementation can be achieved thanks to the recurrence relation (2.3) which the Chebyshev polynomials satisfy. However, it is usually prohibitively expensive to interpolate a big matrix $\boldsymbol{A}$ as a whole, since alone the matrix-matrix multiplication in each step of the recurrence can cost up to $\mathcal{O}(n^3)$, and the evaluation of the expansion at $n_t$ values of $t$ could cost a further $\mathcal{O}(n^2 m n_t)$ operations. Therefore, in case we are only interested in the result of a linear mapping applied to the interpolant, a significant speed-up can be achieved by directly interpolating the result of this linear mapping applied to the interpolant (line 6 in algorithm 2.3). In figure 2.1 some examples of such linear mappings – which we will make use of later on – are schematically illustrated.

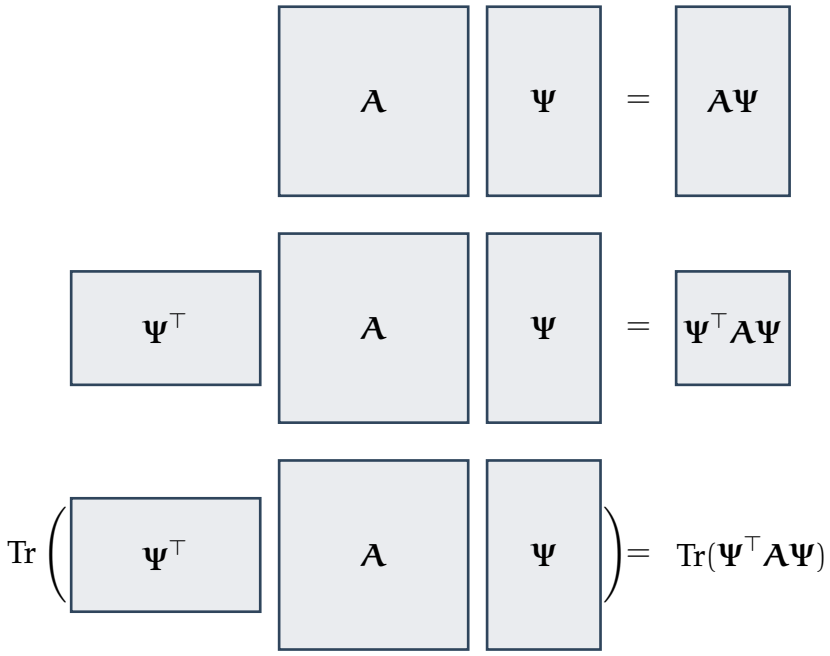



FIGURE 2.1 – Schematic illustration of linear mappings which, applied to a large matrix $\boldsymbol{A}$, reduce the dimensionality of the interpolation problem.

Finally, we can give the pseudocode of this first method in algorithm 2.3.

**Algorithm 2.3: Delta-Gauss-Chebyshev method**

**Input:** Symmetric matrix $\boldsymbol{A} \in \mathbb{R}^{n\times n}$, evaluation points $\{t_i\}_{i=1}^{n_t}$
**Parameters:** Number of Hutchinson's queries $n_\psi$, degree of expansion $m$
**Output:** Approximate evaluations of the spectral density $\{\widetilde{\phi}_\sigma^{(m)}(t_i)\}_{i=1}^{n_t}$

Compute $\{\mu_l(t_i)\}_{l=0}^{m}$ for all $t_i$ using algorithm 2.1
Generate standard Gaussian random matrix $\boldsymbol{\Psi} \in \mathbb{R}^{n \times n_\Psi}$
Initialize $[\boldsymbol{V}_1, \boldsymbol{V}_2, \boldsymbol{V}_3] \leftarrow [\boldsymbol{0}_{n \times n_\Psi}, \boldsymbol{\Psi}, \boldsymbol{0}_{n \times n_\Psi}]$
Set $\widetilde{\phi}_\sigma^{(m)}(t_i) \leftarrow 0$ for all $t_i$
**for** $l = 0, \ldots, m$ **do**
    $x \leftarrow \mathrm{Tr}(\boldsymbol{\Psi}^\top \boldsymbol{V}_2)$ ▷ Linear mapping of interpolant (figure 2.1)
    **for** $i = 1, \ldots, n_t$ **do**
        $\widetilde{\phi}_\sigma^{(m)}(t_i) \leftarrow \widetilde{\phi}_\sigma^{(m)}(t_i) + \mu_l(t_i) x$
    $\boldsymbol{V}_3 \leftarrow (2 - \delta_{l0}) \boldsymbol{A} \boldsymbol{V}_2 - \boldsymbol{V}_1$ ▷ Chebyshev recurrence (2.3)
    $\boldsymbol{V}_1 \leftarrow \boldsymbol{V}_2, \boldsymbol{V}_2 \leftarrow \boldsymbol{V}_3$

Denoting the cost of a matrix-vector product of $\boldsymbol{A} \in \mathbb{R}^{n \times n}$ with $c(n)$, e.g. $\mathcal{O}(c(n)) = n^2$ for dense and $\mathcal{O}(c(n)) = n$ for sparse matrices, we determine the computational complexity of the DGC method to be $\mathcal{O}(m \log(m) n_t + m n_\Psi c(n))$, with $\mathcal{O}(m n_t + n n_\Psi)$ required additional storage.

### 2.3.1 Implementation Details

Expression (2.12) is only well-defined for matrices whose spectra are fully contained in $[-1, 1]$. To also use the DGC method on matrices $\boldsymbol{A}$ whose spectra we know, or can estimate [28, 47], to be within a different interval $[a, b] \subset \mathbb{R}$, we can define a spectral transformation $\tau$ as the linear mapping

$$\begin{cases} \tau : [a, b] \to [-1, 1], \\ \tau(t) = \frac{2t - a - b}{b - a}. \end{cases} \tag{2.15}$$

The DGC method can then be applied to $\bar{\boldsymbol{A}} = \tau(\boldsymbol{A})$ whose spectrum is contained in $[-1, 1]$.

However, retrieving the smooth spectral density $\phi_\sigma$ of the original matrix $\boldsymbol{A}$ after this transformation is not straight-forward and is usually swept under the rug in literature. Let us call $\bar{\phi}$ the spectral density of $\bar{\boldsymbol{A}}$. Based on a derivation from appendix A, it turns out that in order to approximate $\phi_\sigma$ of a general matrix $\boldsymbol{A}$ for the smoothing kernel $g_\sigma$ we consider in this thesis (Gaussian, Lorentzian), we only need to rescale their smoothing parameter $\sigma$ to

$$\bar{\sigma} = \frac{2\sigma}{b - a}, \tag{2.16}$$

run algorithm 2.3 on $\bar{\boldsymbol{A}}$ with $g_{\bar{\sigma}}$ on the transformed evaluation points $\{\tau(t_i)\}_{i=1}^{n_t}$ and finally multiply the resulting approximation with $\frac{2}{b-a}$. In all our examples, this procedure will be used to compute spectral densities of matrices which have eigenvalues outside of $[-1, 1]$.

A speed-up of algorithm 2.3 can be achieved by smartly computing the trace of the product $\mathbf{E}^\top \mathbf{F}$ of two matrices $\mathbf{E}, \mathbf{F} \in \mathbb{R}^{N \times M}$ in $\mathcal{O}(MN)$ instead of $\mathcal{O}(M^2 N)$ time, due to the relation

$$\operatorname{Tr}(\mathbf{E}^\top \mathbf{F}) = \sum_{i=1}^{N} \sum_{j=1}^{M} e_{ij} f_{ij}. \tag{2.17}$$

This reduces the complexity of line 6 in algorithm 2.3 from $\mathcal{O}(n n_\Psi^2)$ to $\mathcal{O}(n n_\Psi)$. Throughout this work, as has already be done for the complexity analysis of algorithm 2.3, we implicitly assume that all traces of this form are computed using this technique.

### 2.3.2 THEORETICAL ANALYSIS

In order to obtain tractable results and because it is the most common case in literature, we choose to restrict the analysis in this section to Gaussian $g_\sigma$ (1.5).

The convergence of the expansion of a Gaussian $g_\sigma$ is exponential and depends on $\sigma$. This can be seen quite well in figure 2.2, and is proven in the following lemma.

**Lemma 2.4: $L^1$-error of Chebyshev expansion for Gaussian smoothing**

Let $\mathbf{A} \in \mathbb{R}^{n \times n}$ be a symmetric matrix whose spectrum is contained in $[-1, 1]$. Then the expansion $g_\sigma^{(m)}$ of the Gaussian smoothing kernel $g_\sigma$ and the corresponding expansion $\phi_\sigma^{(m)}$ of the smooth spectral density $\phi_\sigma$ satisfy

$$\|g_\sigma - g_\sigma^{(m)}\|_\infty \leqslant \frac{\sqrt{2}}{n\sigma^2}(1+\sigma)^{-m}, \tag{2.18}$$

$$\|\phi_\sigma - \phi_\sigma^{(m)}\|_\infty \leqslant \frac{\sqrt{2}}{\sigma^2}(1+\sigma)^{-m}, \tag{2.19}$$

$$\|\phi_\sigma - \phi_\sigma^{(m)}\|_1 \leqslant \frac{2\sqrt{2}}{\sigma^2}(1+\sigma)^{-m}. \tag{2.20}$$

for all $\sigma > 0$.

This result is a consequence of Bernstein's theorem [38, theorem 4.3]. A proof of a similar result can be found in [27, theorem 2]. However, since our result – and more so the proof – deviate from the aforementioned work, we chose to reproduce it hereafter.

*Proof.* From Bernstein's theorem [38, theorem 4.3] and the analyticity of $g_\sigma$ we know that for all $t \in \mathbb{R}$

$$\|g_\sigma(t - \cdot) - g_\sigma^{(m)}(t - \cdot)\|_\infty \leqslant \frac{2}{\chi^m(\chi - 1)} \sup_{z \in \mathcal{E}_\chi} |g_\sigma(t - z)| \tag{2.21}$$

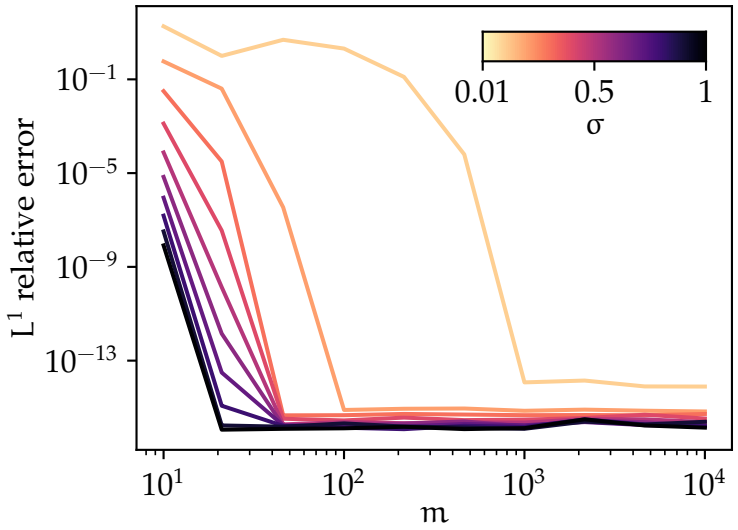



FIGURE 2.2 – The error of the Chebyshev expansion of increasing degree of expansion $m$ for a Gaussian $g_\sigma$ with different values of $\sigma$.

where we can use any ellipse $\mathcal{E}_\chi$ with foci $\{-1, 1\}$ and with sum of half-axes $\chi = a + b > 1$ (see figure 2.3).

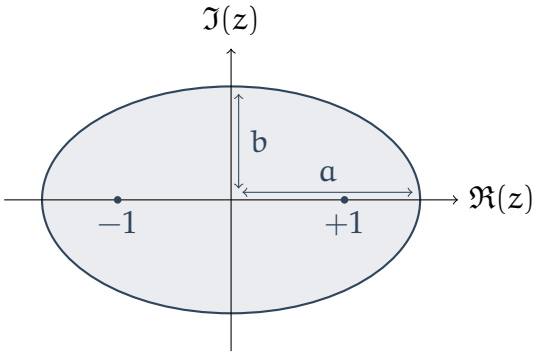



FIGURE 2.3 – A Bernstein ellipse $\mathcal{E}_\chi$ with half axis lengths $a$ and $b$ visualized in the complex plane $\mathbb{C}$.

Writing $z = x + \imath y$ for $x, y \in \mathbb{R}$, we estimate (using $|e^z| = e^{\Re(z)}$)

$$|g_\sigma(t - (x + \imath y))| = \frac{1}{n\sqrt{2\pi\sigma^2}} e^{-\frac{(t-x)^2 - y^2}{2\sigma^2}} \leqslant \frac{1}{n\sqrt{2\pi\sigma^2}} e^{\frac{y^2}{2\sigma^2}}. \tag{2.22}$$

Expressing $\chi = 1 + c\sigma$ for any $c > 0$, we can estimate

$$\chi - \chi^{-1} \leqslant 2c\sigma. \tag{2.23}$$

This can be established by observing $h(\chi) = 2c\sigma - \chi + \chi^{-1} = 2(\chi - 1) - \chi + \chi^{-1} = \chi + \chi^{-1} - 2 \geqslant 0$ because $h(1) = 0$ and $h'(\chi) \geqslant 0$ for all $\chi > 1$. Furthermore,

because $z$ is contained in $\mathcal{E}_\chi$ we know that the absolute value of its imaginary part $y$ is upper bound by the length of the imaginary half axis $b$, which can be expressed in terms of $\chi$ to show with (2.23) that

$$|y| \leqslant b = \frac{\chi - \chi^{-1}}{2} \leqslant c\sigma. \tag{2.24}$$

Consequently, for all $t \in \mathbb{R}$

$$\sup_{z \in \mathcal{E}_\chi} |g_\sigma(t-z)| \leqslant \frac{1}{n\sqrt{2\pi\sigma^2}} e^{\frac{c^2}{2}}. \tag{2.25}$$

Plugging this estimate into (2.21) yields

$$\|g_\sigma(t-\cdot) - g_\sigma^{(m)}(t-\cdot)\|_\infty \leqslant \frac{2}{(1+c\sigma)^m c\sigma} \frac{1}{n\sqrt{2\pi\sigma^2}} e^{\frac{c^2}{2}} \tag{2.26}$$

for every $c > 0$. In particular, for $c = 1$ and with $\sqrt{e/\pi} \leqslant 1$ we have

$$\|g_\sigma(t-\cdot) - g_\sigma^{(m)}(t-\cdot)\|_\infty \leqslant \frac{\sqrt{2}}{n\sigma^2}(1+\sigma)^{-m}, \tag{2.27}$$

which shows the first assertion with $t = 0$.

For the second assertion, we may use basic properties of matrix functions to obtain

$$\begin{aligned}
&\left|\phi_\sigma(t) - \phi_\sigma^{(m)}(t)\right| \\
&= \left|\operatorname{Tr}(g_\sigma(t\mathbf{I}_n - \mathbf{A})) - \operatorname{Tr}(g_\sigma^{(m)}(t\mathbf{I}_n - \mathbf{A}))\right| && \text{(definitions (1.6) and (2.12))} \\
&= \left|\sum_{i=1}^{n} \left(g_\sigma(t-\lambda_i) - g_\sigma^{(m)}(t-\lambda_i)\right)\right| && (\lambda_1, \dots, \lambda_n \text{ eigenvalues of } \mathbf{A}) \\
&\leqslant n \max_{i=1,\dots,n} \left|g_\sigma(t-\lambda_i) - g_\sigma^{(m)}(t-\lambda_i)\right| && \text{(conservative upper bound)} \\
&\leqslant n \max_{s \in [-1,1]} \left|g_\sigma(t-s) - g_\sigma^{(m)}(t-s)\right| && \text{(extension of domain)} \\
&= n\|g_\sigma(t-\cdot) - g_\sigma^{(m)}(t-\cdot)\|_\infty && \text{(definition of } L^\infty\text{-norm)} \\
&\leqslant \frac{\sqrt{2}}{\sigma^2}(1+\sigma)^{-m} && \text{(using (2.27))}
\end{aligned}$$

from which the result follows directly.

Finally, Hölder's inequality [23] allows us to also show the last assertion with what we have found above:

$$\|\phi_\sigma - \phi_\sigma^{(m)}\|_1 \leqslant 2\|\phi_\sigma - \phi_\sigma^{(m)}\|_\infty \leqslant \frac{2\sqrt{2}}{\sigma^2}(1+\sigma)^{-m}. \tag{2.28}$$

□

We now have all the tools at hand to combine the approximation error of the Chebyshev expansion (lemma 2.4) with the trace estimation error (lemma 2.2) to get a tractable theoretical result for the accuracy of the DGC method.

**Theorem 2.5: $L^1$-error of Delta-Gauss-Chebyshev method**

Let $\widetilde{\phi}_\sigma^{(m)}(t)$ be the result from running algorithm 2.3 on a symmetric matrix $\mathbf{A} \in \mathbb{R}^{n\times n}$ with its spectrum contained in $[-1,1]$ using a Gaussian smoothing kernel $g_\sigma$ with smoothing parameter $\sigma > 0$, degree of expansion $m \in \mathbb{N}$, and number of Hutchinson's queries $n_\Psi \in \mathbb{N}$. For $\delta \in (0, e^{-1})$ it holds with probability $\geqslant 1-\delta$, that

$$\|\phi_\sigma - \widetilde{\phi}_\sigma^{(m)}\|_1 \leqslant \frac{\sqrt{2}}{\sigma^2}(1+\sigma)^{-m}\left(2 + c_\Psi \frac{\log(1/\delta)}{\sqrt{n n_\Psi}}\right) + c_\Psi \frac{\log(1/\delta)}{\sqrt{n_\Psi}} \tag{2.29}$$

for $c_\Psi = 24e$.

*Proof.* First, we apply the triangle inequality to get

$$\|\phi_\sigma - \widetilde{\phi}_\sigma^{(m)}\|_1 \leqslant \|\phi_\sigma - \phi_\sigma^{(m)}\|_1 + \|\phi_\sigma^{(m)} - \widetilde{\phi}_\sigma^{(m)}\|_1. \tag{2.30}$$

The first term can be dealt with using lemma 2.4. Lemma 2.2 can be applied to the second term for

$$\begin{aligned}
\|\phi_\sigma^{(m)} - \widetilde{\phi}_\sigma^{(m)}\|_1 &= \int_{-1}^{1} \left|\mathrm{Tr}(g_\sigma^{(m)}(t\mathbf{I}_n - \mathbf{A})) - H_{n_\Psi}(g_\sigma^{(m)}(t\mathbf{I}_n - \mathbf{A}))\right| \mathrm{d}t && \text{(definitions)} \\
&\leqslant c_\Psi \frac{\log(1/\delta)}{\sqrt{n_\Psi}} \int_{-1}^{1} \|g_\sigma^{(m)}(t\mathbf{I}_n - \mathbf{A})\|_F \mathrm{d}t && \text{(lemma 2.2)}
\end{aligned}$$

We proceed with bounding the involved integrand by first applying the triangle inequality, then exploiting properties of the Frobenius norm of a matrix function and the positivity of $g_\sigma$, and finally using the result from the proof of lemma 2.4 and the definition of $\phi_\sigma$:

$$\begin{aligned}
&\|g_\sigma^{(m)}(t\mathbf{I}_n - \mathbf{A})\|_F \\
&\leqslant \|g_\sigma^{(m)}(t\mathbf{I}_n - \mathbf{A}) - g_\sigma(t\mathbf{I}_n - \mathbf{A})\|_F + \|g_\sigma(t\mathbf{I}_n - \mathbf{A})\|_F && \text{(triangle inequality)} \\
&\leqslant \sqrt{n}\|g_\sigma^{(m)} - g_\sigma\|_\infty + \mathrm{Tr}(g_\sigma(t\mathbf{I}_n - \mathbf{A})) && \text{(norm inequalities)} \\
&\leqslant \frac{\sqrt{2}}{\sqrt{n}\sigma^2}(1+\sigma)^{-m} + \phi_\sigma(t) && \text{(lemma 2.4)}
\end{aligned}$$

Putting all things together and using the normalization $\int_{-1}^{1} \phi_\sigma(t)\mathrm{d}t = 1$ we end

up with the desired result:

$$\begin{aligned}\|\phi_\sigma - \widetilde{\phi}_\sigma^{(m)}\|_1 &\leqslant \frac{2\sqrt{2}}{\sigma^2}(1+\sigma)^{-m} + c_\Psi \frac{\log(1/\delta)}{\sqrt{n_\Psi}}\left(\frac{\sqrt{2}}{\sqrt{n}\sigma^2}(1+\sigma)^{-m} + 1\right) \\ &= \frac{\sqrt{2}}{\sigma^2}(1+\sigma)^{-m}\left(2 + c_\Psi \frac{\log(1/\delta)}{\sqrt{n n_\Psi}}\right) + c_\Psi \frac{\log(1/\delta)}{\sqrt{n_\Psi}}.\end{aligned}$$

□

We see that the first term in theorem 2.5 will quickly vanish as $m$ increases. What we are left with is the slowly decaying $\mathcal{O}(n_\Psi^{-1/2})$ term. In fact, this is what bottlenecks the DGC method from achieving better accuracies: The Girard-Hutchinson stochastic trace estimator is not efficient enough for approximating spectral densities. Therefore, we will consider alternative ways of approximating (1.6) in the next two chapters.

# Chapter 3

# Randomized low-rank factorization

We again start at (1.6), but now directly analyze the structure of the matrix function

$$g_\sigma(t\mathbf{I}_n - \mathbf{A}). \tag{3.1}$$

Suppose $\mathbf{A}$ is symmetric and has a spectral decomposition

$$\mathbf{A} = \mathbf{U}\mathbf{\Lambda}\mathbf{U}^\top = \sum_{i=1}^{n} \lambda_i \mathbf{u}_i \mathbf{u}_i^\top. \tag{3.2}$$

Applying the matrix function to this expression, we see that

$$g_\sigma(t\mathbf{I}_n - \mathbf{A}) = \sum_{i=1}^{n} g_\sigma(t - \lambda_i)\mathbf{u}_i \mathbf{u}_i^\top. \tag{3.3}$$

The smoothing kernel $g_\sigma$ typically decays rapidly to zero when its argument deviates from zero, particularly when $\sigma$ is small. This can, for example, be observed when using Gaussian or Lorentzian smoothing (see figure 5.5). Therefore, all terms in the sum (3.3) are suppressed except the ones for which $\lambda_i$ is close to $t$. Thus,

$$g_\sigma(t\mathbf{I}_n - \mathbf{A}) \approx \sum_{i:|t-\lambda_i|/\sigma \text{ small}} g_\sigma(t - \lambda_i)\mathbf{u}_i \mathbf{u}_i^\top \tag{3.4}$$

usually exhibits an approximate low-rank structure for all $t$, unless of course $\sigma$ is chosen exceedingly big or all eigenvalues are clustered. The matrix function can approximately be represented by a product of two or more matrices which are significantly smaller.

The aim of this chapter is to find a good approximate factorization of (3.1) in terms of smaller matrices, which we can efficiently compute and for which we can easily determine the trace (1.6).

## 3.1 THE NUMERICAL RANK OF A MATRIX

To quantify by how much a matrix could potentially be compressed when being accurately represented as the product of smaller matrices, we introduce the rank of a matrix [18, section III.3]. The rank of a matrix $\mathbf{B} \in \mathbb{R}^{n\times n}$ is defined as

$$\operatorname{rank}(\mathbf{B}) = \dim(\operatorname{range}(\mathbf{B})), \tag{3.5}$$

where dim is the dimension and range denotes the set of all elements which can be attained by right-multiplying $\mathbf{B}$ with a vector from $\mathbb{R}^n$.

To account for the finite precision of floating point operations and for matrices which are almost low-rank matrices, we use the notion of the $\varepsilon$-numerical rank $r_{\varepsilon,\cdot}$ [2, definition 1.1].

**Definition 3.1: Numerical rank of a matrix**

The $\varepsilon$-numerical rank $r_{\varepsilon,\cdot}$ of a matrix $\mathbf{B} \in \mathbb{R}^{n\times n}$ in the norm $\|\cdot\|_\cdot$ is the number

$$r_{\varepsilon,\cdot}(\mathbf{B}) = \min\{\operatorname{rank}(\mathbf{C}) : \mathbf{C} \in \mathbb{R}^{n\times n} : \|\mathbf{B}-\mathbf{C}\|_\cdot \leqslant \varepsilon\} \tag{3.6}$$

for a small tolerance $\varepsilon \geqslant 0$.

A matrix is said to be numerically low-rank, if $r_{\varepsilon,\cdot}(\mathbf{B}) \ll n$. $\varepsilon$ is usually taken to be the double machine precision, i.e. $10^{-16}$.

For unitarily invariant norms $\|\cdot\|_\cdot$ and symmetric positive semi-definite (PSD) matrices $\mathbf{B}$, we may use [31, theorem 5] to express $r_{\varepsilon,\cdot}$ in terms of its eigenvalues $\sigma_1 \geqslant \cdots \geqslant \sigma_n \geqslant 0$. In particular, for the spectral norm $\|\cdot\|_2$ we have

$$r_{\varepsilon,2}(\mathbf{B}) = \min\{1 \leqslant r \leqslant n : \sigma_{r+1} \leqslant \varepsilon\}, \tag{3.7}$$

for the nuclear norm $\|\cdot\|_*$

$$r_{\varepsilon,*}(\mathbf{B}) = \min\{1 \leqslant r \leqslant n : \sum_{j=r+1}^{n} \sigma_j \leqslant \varepsilon\}, \tag{3.8}$$

and for the Frobenius norm $\|\cdot\|_F$

$$r_{\varepsilon,F}(\mathbf{B}) = \min\{1 \leqslant r \leqslant n : \sqrt{\sum_{j=r+1}^{n} \sigma_j^2} \leqslant \varepsilon\}. \tag{3.9}$$

## 3.2 LOW-RANK FACTORIZATION

We will now give an introduction to low-rank factorizations of constant matrices.

### 3.2.1 CONSTANT MATRICES

From section 3.1, it is clear that the rank of a symmetric matrix is also equal to the number of its eigenvalues which are non-zero. Hence, for a symmetric PSD matrix $\mathbf{B} \in \mathbb{R}^{n \times n}$ of rank $r \ll n$, we can immediately obtain a factorization of the matrix in terms of smaller matrices by looking at its spectral decomposition

$$\mathbf{B} = \mathbf{V}\mathbf{\Sigma}\mathbf{V}^\top = \begin{bmatrix} \mathbf{V}_1 & \mathbf{V}_2 \end{bmatrix} \begin{bmatrix} \mathbf{\Sigma}_1 & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{bmatrix} \begin{bmatrix} \mathbf{V}_1^\top \\ \mathbf{V}_2^\top \end{bmatrix} = \mathbf{V}_1 \mathbf{\Sigma}_1 \mathbf{V}_1^\top \tag{3.10}$$

with $\mathbf{\Sigma}_1 \in \mathbb{R}^{r \times r}$ containing all the non-zero eigenvalues $\sigma_1, \ldots, \sigma_r$ of $\mathbf{B}$ and $\mathbf{V}_1 \in \mathbb{R}^{n \times r}$ the corresponding eigenvectors. What's more, we see that

$$\mathbf{V}_1 \mathbf{V}_1^\top \mathbf{B} = \mathbf{B}, \tag{3.11}$$

i.e. the columns of the matrix $\mathbf{V}_1$ fully capture the range of $\mathbf{B}$.

However, computing the spectral decomposition is often prohibitively expensive and usually not necessary for obtaining good approximate factorizations of a matrix. There exist multiple factorization methods for low-rank matrices, many of which are based on sketching. Sketching is a widely used technique for approximating the column space of a matrix [15, 46, 27, 41, 40]. Multiplying a low-rank matrix $\mathbf{B} \in \mathbb{R}^{n \times n}$ with a standard Gaussian sketching matrix $\mathbf{\Omega} \in \mathbb{R}^{n \times n_\Omega}$ with $n_\Omega \ll n$, will result in a significantly smaller matrix $\mathbf{B}\mathbf{\Omega}$ – the sketch – whose columns roughly span the same range as the columns of $\mathbf{B}$. Closely following [40, section 2.1], we illustrate the reason why this approach works hereafter.

Suppose $\mathbf{B} \in \mathbb{R}^{n \times n}$ is symmetric PSD and admits a spectral decomposition

$$\mathbf{B} = \mathbf{V}\mathbf{\Sigma}\mathbf{V}^\top = \sum_{i=1}^{n} \sigma_i \boldsymbol{v}_i \boldsymbol{v}_i^\top. \tag{3.12}$$

If we left-multiply $\mathbf{B}$ with a standard Gaussian random vector $\boldsymbol{\omega} \in \mathbb{R}^n$, we obtain

$$\mathbf{B}\boldsymbol{\omega} = \sum_{i=1}^{n} \sigma_i \boldsymbol{v}_i (\boldsymbol{v}_i^\top \boldsymbol{\omega}) = \sum_{i=1}^{n} (\sigma_i z_i) \boldsymbol{v}_i, \tag{3.13}$$

where $z_i = \boldsymbol{v}_i^\top \boldsymbol{\omega} \in \mathbb{R}$ again follows a standard Gaussian distribution, since the $\boldsymbol{v}_i$ have unit norm [23]. The component of $\mathbf{B}\boldsymbol{\omega}$ in the direction of the j-th eigenvector $\boldsymbol{v}_j$ of $\mathbf{B}$ is

$$\boldsymbol{v}_j^\top \mathbf{B}\boldsymbol{\omega} = \sum_{i=1}^{n} (\sigma_i z_i)(\boldsymbol{v}_j^\top \boldsymbol{v}_i) = \sigma_j z_j. \tag{3.14}$$

The larger the eigenvalue $\sigma_j$ of an eigenvector $\boldsymbol{v}_j$ is, the more prominent the component of $\mathbf{B}\boldsymbol{\omega}$ in that direction usually is. That is, $\mathbf{B}\boldsymbol{\omega}$ is roughly aligned with the eigenvectors corresponding to the largest eigenvalues: the dominant eigenvectors. When repeating this procedure for multiple $\boldsymbol{\omega}$ by right-multiplying $\mathbf{B}$ with a standard Gaussian $\boldsymbol{\Omega} \in \mathbb{R}^{n \times n_\Omega}$, it can be shown that $\mathbf{B}\boldsymbol{\Omega}$ approximates the range of $\mathbf{B}$ well, and consequently

$$\widehat{\mathbf{B}} = (\mathbf{B}\boldsymbol{\Omega})(\mathbf{B}\boldsymbol{\Omega})^{\dagger}\mathbf{B} \tag{3.15}$$

is a good approximation of $\mathbf{B}$ [15].

Based on the sketch $\mathbf{B}\boldsymbol{\Omega}$, we can also compute low-rank approximations of the form

$$\widehat{\mathbf{B}} = (\mathbf{B}\boldsymbol{\Omega})\mathbf{K}(\mathbf{B}\boldsymbol{\Omega})^{\top}, \tag{3.16}$$

for some matrix $\mathbf{K} \in \mathbb{R}^{n_\Omega \times n_\Omega}$ [27, section 3.1]. Ideally, we would want $\widehat{\mathbf{B}}$ to be as close to $\mathbf{B}$ as possible. Therefore, by multiplying (3.16) from the left with $\boldsymbol{\Omega}^{\top}$ and from the right with $\boldsymbol{\Omega}$, and by imposing $\widehat{\mathbf{B}} = \mathbf{B}$ we get

$$\boldsymbol{\Omega}^{\top}\mathbf{B}\boldsymbol{\Omega} = (\boldsymbol{\Omega}^{\top}\mathbf{B}\boldsymbol{\Omega})\mathbf{K}(\boldsymbol{\Omega}^{\top}\mathbf{B}\boldsymbol{\Omega})^{\top}. \tag{3.17}$$

By the properties of the pseudoinverse of a matrix we see that $\mathbf{K} = (\boldsymbol{\Omega}^{\top}\mathbf{B}\boldsymbol{\Omega})^{\dagger}$ satisfies this relation. This approximation is often referred to as the Nyström approximation [14]

$$\widehat{\mathbf{B}} = (\mathbf{B}\boldsymbol{\Omega})(\boldsymbol{\Omega}^{\top}\mathbf{B}\boldsymbol{\Omega})^{\dagger}(\mathbf{B}\boldsymbol{\Omega})^{\top}. \tag{3.18}$$

It is shown in [40, lemma 5.2] that for PSD matrices the Nyström approximation is at least as accurate as the approximation in (3.15) when measured in the spectral and Frobenius norms. The following theorem gives an upper bound on the error of the Nyström approximation in the Frobenius norm [34, lemma 3.2].

**Theorem 3.2: Frobenius norm error of Nyström approximation**

Let $\mathbf{B} \in \mathbb{R}^{n \times n}$ be a symmetric PSD matrix. Its Nyström approximation $\widehat{\mathbf{B}}$ (3.18) with standard Gaussian $\boldsymbol{\Omega} \in \mathbb{R}^{n \times n_\Omega}$ of even sketch size $n_\Omega \geqslant 10$ satisfies with probability $\geqslant 1 - 6e^{-n_\Omega/2}$

$$\|\mathbf{B} - \widehat{\mathbf{B}}\|_F \leqslant 542\sqrt{\frac{2}{n_\Omega}}\,\mathrm{Tr}(\mathbf{B}). \tag{3.19}$$

### 3.2.2 PARAMETER-DEPENDENT MATRICES

The Nyström low-rank factorizations can be extended in a straightforward way to the case where the symmetric PSD matrix $\mathbf{B}(t) \in \mathbb{R}^{n \times n}$ depends continuously on a parameter $t \in [a, b]$:

$$\widehat{\mathbf{B}}(t) = (\mathbf{B}(t)\boldsymbol{\Omega})(\boldsymbol{\Omega}^{\top}\mathbf{B}(t)\boldsymbol{\Omega})^{\dagger}(\mathbf{B}(t)\boldsymbol{\Omega})^{\top}. \tag{3.20}$$

What's special in this formulation is that we reuse the same sketching matrix $\boldsymbol{\Omega} \in \mathbb{R}^{n \times n_\Omega}$ for all $t$. This will introduce remarkable gains in terms of computational complexity of the approximation.

In [17], a corresponding result to theorem 3.2 for the parameter-dependent case is shown:

**Lemma 3.3: $L^1$-error of parameter-dependent Nyström approximation**

Let $\mathbf{B}(t) \in \mathbb{R}^{n \times n}$ be symmetric PSD and continuous in $t \in [a, b]$. Then the parameter-dependent Nyström approximation $\widehat{\mathbf{B}}(t)$ (3.20) with standard Gaussian $\boldsymbol{\Omega} \in \mathbb{R}^{n \times n_\Omega}$, even $n_\Omega \geqslant 8 \log(1/\delta)$, and constant $c_\Omega \geqslant 0$, satisfies with probability $\geqslant 1 - \delta$

$$\int_a^b \|\mathbf{B}(t) - \widehat{\mathbf{B}}(t)\|_F dt \leqslant c_\Omega \frac{1}{\sqrt{n_\Omega}} \int_a^b \operatorname{Tr}(\mathbf{B}(t)) dt. \tag{3.21}$$

For the trace of the parameter-dependent Nyström approximation, a much stronger convergence result can be shown for matrix functions of the form $f(\mathbf{A}, t)$ which continuously depend on $t$ and whose eigenvalues decay quickly. An expression for the trace estimation error in the $L^1$-norm is shown in a theorem from [17].

**Theorem 3.4: $L^1$-trace-error of parameter-dependent Nyström approximation**

Let $f(\mathbf{A}, t)$ be a non-negative function of $\mathbf{A} \in \mathbb{R}^{n \times n}$ which also continuously depends on $t \in [a, b]$. Denote with $\sigma_1(t) \geqslant \cdots \geqslant \sigma_n(t) \geqslant 0$ the eigenvalues of $f(\mathbf{A}, t)$ at $t$. Let the standard Gaussian sketching matrix $\boldsymbol{\Omega} \in \mathbb{R}^{n \times n_\Omega}$ used in (3.18) have $n_\Omega = r + p$ columns for some numbers $r \geqslant 2, p \geqslant 4$. Then for all $\gamma \geqslant 1$, the Nyström approximation $\widehat{f}(\mathbf{A}, t)$ satisfies

$$\int_a^b |\operatorname{Tr}(f(\mathbf{A}, t)) - \operatorname{Tr}(\widehat{f}(\mathbf{A}, t))| dt \leqslant \gamma^2 (1 + r) \int_a^b \sum_{i=r+1}^{n} \sigma_i(t) dt \tag{3.22}$$

with probability $\geqslant 1 - \gamma^{-p}$.

Note that the smooth spectral density $\phi_\sigma$ (1.6) is exactly a trace of such a matrix function, which means that we can directly apply this theorem to it.

## 3.3 THE NYSTRÖM-CHEBYSHEV METHOD

Motivated by the observations from the beginning of this chapter, we now present an algorithm for estimating the smooth spectral density $\phi_\sigma$. Since all $g_\sigma$ we consider are non-negative, the matrix function (3.1) is symmetric PSD. There-

fore, we opt for the Nyström approximation (3.20) as our randomized low-rank factorization engine. Before we go into the details of the method, we need to quantify the $\varepsilon$-numerical rank $r_{\varepsilon,\cdot}$ of this matrix function, in order to get an a priori guarantee for the convergence of the method.

When using a Gaussian $g_\sigma$, we can explicitly write the eigenvalues of $g_\sigma(t\mathbf{I}_n - \mathbf{A})$ as

$$\sigma_i(t) = g_\sigma(t - \lambda_{(i)}) = \frac{1}{n\sqrt{2\pi\sigma^2}} e^{-\frac{(t-\lambda_{(i)})^2}{2\sigma^2}} \tag{3.23}$$

where $\lambda_{(1)}, \dots, \lambda_{(n)}$ denote the eigenvalues of $\mathbf{A}$ sorted by increasing distance from $t$, such that $\sigma_1(t) \geqslant \cdots \geqslant \sigma_n(t)$. Consequently, by using the closed-form expression of the eigenvalues (3.23) and inserting it into (3.7), (3.8), and (3.9), we may upper bound the numerical rank of (3.1) for Gaussian $g_\sigma$ as

$$r_{\varepsilon,\cdot}(g_\sigma(t\mathbf{I}_n - \mathbf{A})) \leqslant \#\{1 \leqslant i \leqslant n : |t - \lambda_i| < C_{\varepsilon,\cdot}(\sigma)\} \tag{3.24}$$

with the eigenvalues $\lambda_1, \dots, \lambda_n$ of $\mathbf{A}$ and the constants

$$C_{\varepsilon,2}(\sigma) = \sigma\sqrt{-2\log(\sqrt{2\pi}n\sigma\varepsilon)} \qquad \text{(spectral norm)} \tag{3.25}$$

$$C_{\varepsilon,*}(\sigma) = \sigma\sqrt{-2\log(\sqrt{2\pi n}\sigma\varepsilon)} \qquad \text{(nuclear norm)} \tag{3.26}$$

$$C_{\varepsilon,F}(\sigma) = \sigma\sqrt{-2\log(\sqrt{2\pi}\sigma\varepsilon)} \qquad \text{(Frobenius norm)} \tag{3.27}$$

For the spectral norm, (3.24) even holds as an equality. A derivation of expression (3.24) and the corresponding constants can be found in appendix B.

The expression (3.24) has a very visual interpretation: The $\varepsilon$-numerical rank $r_{\varepsilon,\cdot}$ of a matrix is at most equal to the number of eigenvalues which are closer to $t$ than $C_{\varepsilon,\cdot}(\sigma)$. This is also illustrated in figure 3.1.

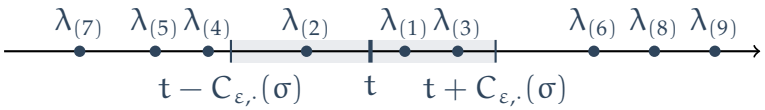
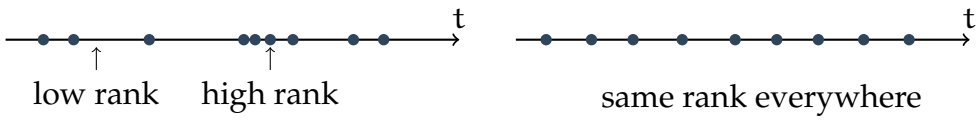



FIGURE 3.1 – The numerical rank of $g_\sigma(t\mathbf{I}_n - \mathbf{A})$ can be approximately computed by counting the number of eigenvalues $\lambda_{(1)}, \dots, \lambda_{(n)}$ of the matrix $\mathbf{A}$ which lie less than a constant $C_{\varepsilon,\cdot}(\sigma)$ away from $t$.

If we additionally assume the eigenvalues of the matrix $\mathbf{A}$ to be evenly distributed within $[a, b]$, that is, in any subinterval of fixed length in $[a, b]$ we can expect to find roughly the same number of eigenvalues (see figure 3.2), then we can estimate the numerical rank of $g_\sigma(t\mathbf{I}_n - \mathbf{A})$ for Gaussian $g_\sigma$ to be

$$r_{\varepsilon,\cdot}(g_\sigma(t\mathbf{I}_n - \mathbf{A})) \lessapprox \frac{2n}{b-a} C_{\varepsilon,\cdot}(\sigma). \tag{3.28}$$




(A) Unevenly distributed spectrum (B) Evenly distributed spectrum

FIGURE 3.2 – Examples of an unevenly and an evenly distributed spectrum with some comments about the expected numerical rank of $g_\sigma(t\mathbf{I}_n - \mathbf{A})$ at certain values of $t$. The dots represent eigenvalues.

In figure 3.3, we numerically check the decay of the eigenvalues for one of our example matrices which we use in the numerical experiments (section 5.1). Clearly, the estimated numerical rank of the matrix is not accurate for some values of $t$ in this case, since the spectrum is not evenly distributed. Nevertheless, it provides us with a good first guess which can be used to decide on what values for $n_\Omega$ to choose in order to ensure a high accuracy.

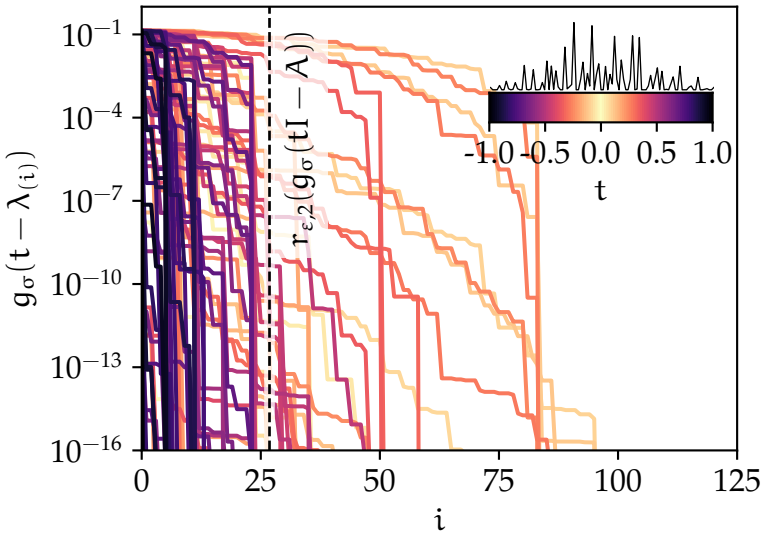



FIGURE 3.3 – Singular value decay of a Gaussian smoothing kernel $g_\sigma$ with $\sigma = 0.05$ applied to the matrix introduced section 5.1 for $c = 1$. For reference, the effective spectral density is plotted above the color bar which assigns a color to the values of $t$.

The backbone of what is also known as the spectrum sweeping (SS) method [27] is the Nyström approximation of the Chebyshev expansion (2.12), hence why we refer to it as the Nyström-Chebyshev (NC) method. For some standard Gaussian $\mathbf{\Omega} \in \mathbb{R}^{n \times n_\Omega}$, the approximation reads

(3.29)
$$\widehat{g}_\sigma^{(m)}(t\mathbf{I}_n - \mathbf{A}) = (g_\sigma^{(m)}(t\mathbf{I}_n - \mathbf{A})\mathbf{\Omega})(\mathbf{\Omega}^\top g_\sigma^{(m)}(t\mathbf{I}_n - \mathbf{A})\mathbf{\Omega})^\dagger (g_\sigma^{(m)}(t\mathbf{I}_n - \mathbf{A})\mathbf{\Omega})^\top.$$

Since we are only interested in the trace of this approximation, we may use the cyclic property of the trace to obtain

$$\widehat{\phi}_\sigma^{(m)}(t) = \operatorname{Tr}\Big((\underbrace{\boldsymbol{\Omega}^\top g_\sigma^{(m)}(t\mathbf{I}_n - \mathbf{A})\boldsymbol{\Omega}}_{=\mathbf{K}_1(t)})^\dagger(\underbrace{\boldsymbol{\Omega}^\top (g_\sigma^{(m)}(t\mathbf{I}_n - \mathbf{A}))^2\boldsymbol{\Omega}}_{=\mathbf{K}_2(t)})\Big). \tag{3.30}$$

The interpolation framework introduced in section 2.1, allows us to interpolate the matrices $\mathbf{K}_1(t) \in \mathbb{R}^{n_\Omega \times n_\Omega}$ and $\mathbf{K}_2(t) \in \mathbb{R}^{n_\Omega \times n_\Omega}$, which appear inside the trace, efficiently. However, when interpolating the squared $g_\sigma$ in a separate expansion

$$(g_\sigma(t\mathbf{I}_n - \mathbf{A})^2)^{(m)} = \sum_{l=0}^{2m} \nu_l(t) T_l(A), \tag{3.31}$$

as is suggested in [27], it is no longer guaranteed that the expansion of the squared matrix function $(g_\sigma(t\mathbf{I}_n - \mathbf{A})^2)^{(m)}$ is equal to the square of the expanded matrix function $g_\sigma^{(m)}(t\mathbf{I}_n - \mathbf{A})$. We have observed that the loss of this square relationship decreases the numerical accuracy noticeably, which is shown in figure 3.4.

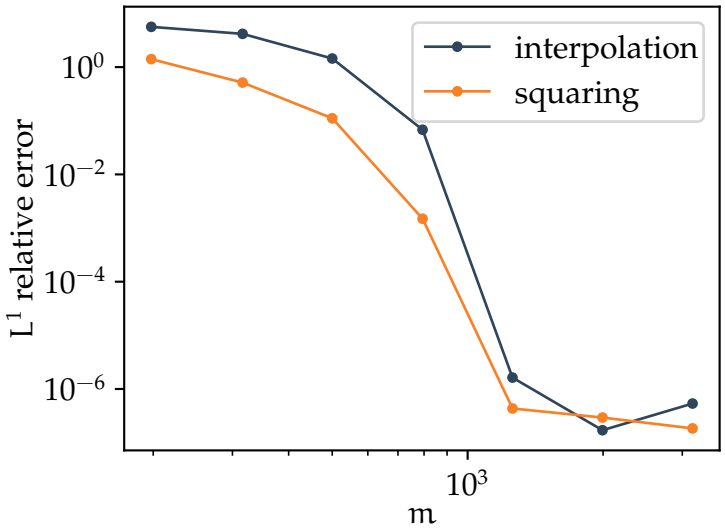



FIGURE 3.4 – Difference in the accuracy when computing (3.30) using a separate expansion (interpolation) versus explicitly squaring the matrix function (squaring). With the NC, which we will introduce on the next few pages, the spectral density of a 2D model problem from section 5.1 is computed for a Gaussian $g_\sigma$ with $\sigma = 0.05$ and fixed $n_\Omega = 80$. An estimate of the relative error in the $L^1$-norm is computed based on $n_t = 100$ evaluation of the spectral density.

In order to circumvent the above mentioned issue, we propose an alternative way of computing an expansion for $(g_\sigma)^2$ which is consistent, more accurate (see figure 3.4) and significantly faster (see table 3.1): The relation of the Chebyshev expansion to the DCT shown in (2.6) can be exploited to design fast and exact multiplication algorithms between polynomials of the form (2.12) [3, proposition 3.1]. In particular, raising a Chebyshev expansion to an integer power can be achieved very efficiently by chaining a DCT with an inverse DCT: Suppose we know the coefficients $\boldsymbol{\mu} \in \mathbb{R}^{m+1}$ of a Chebyshev expansion (2.12). Then we may quickly compute the coefficients $\boldsymbol{\nu} \in \mathbb{R}^{km+1}$ of the same expansion raised to the power $k \geqslant 2$ by first extending $\boldsymbol{\mu}$ with zeros to $\widehat{\boldsymbol{\mu}} \in \mathbb{R}^{km+1}$ and subsequently computing

$$\boldsymbol{\nu} = \mathrm{DCT}^{-1}\left\{\mathrm{DCT}\{\widehat{\boldsymbol{\mu}}\}^k\right\} \tag{3.32}$$

where the exponentiation of a vector is understood element-wise. The corresponding algorithm is presented hereafter.

**Algorithm 3.5: Fast and exact exponentiation of Chebyshev expansions**

**Input:** Coefficients $\{\mu_l\}_{l=0}^m$ of the expansion $\sum_{l=0}^m \mu_l T_l(s)$
**Parameter:** Exponent $k \geqslant 2$
**Output:** Coefficients $\{\nu_l\}_{l=0}^{km}$ such that $\sum_{l=0}^{km} \nu_l T_l(s) = (\sum_{l=0}^m \mu_l T_l(s))^k$

1: Define the vector $\widehat{\boldsymbol{\mu}}$ as

$$(\widehat{\boldsymbol{\mu}})_l = \begin{cases} \mu_l & \text{for } l = 0, \dots, m \\ 0 & \text{for } l = m+1, \dots, km \end{cases} \tag{3.33}$$

2: Compute $\mathbf{f} = \mathrm{DCT}(\widehat{\boldsymbol{\mu}})$
3: Let $(\mathbf{f}^k)_l = (f_l)^k$ for $l = 0, \dots, km$
4: Compute $\boldsymbol{\nu} = \mathrm{DCT}^{-1}(\mathbf{f}^k)$
5: Let $\nu_l = (\boldsymbol{\nu})_l$ for $l = 0, \dots, km$

The complexity of this procedure is $\mathcal{O}(km \log(km))$ [29].

The algorithm gives us the consistent expansion

$$(g_\sigma^{(m)}(t\mathbf{I}_n - \mathbf{A}))^2 = \sum_{l=0}^{2m} \nu_l(t) T_l(A). \tag{3.34}$$

This way of expanding the squared matrix function is exactly equivalent to the spectrum sweeping (SS) algorithm [27, algorithm 5], but usually orders of magnitude faster because the matrices involved in the generalized eigenvalue problem on line 13 of this algorithm do not have to be assembled from the product of two large matrices for every $t$.

TABLE 3.1 – Comparison of the runtime in milliseconds of three approaches with which the coefficients of the Chebyshev expansion of a function can be computed. We average over 7 runs of the algorithms and repeat these runs 100 times to form the mean and standard deviation which are given in the below table. We refer to the interpolation of $(g_\sigma)^2$ with [27, algorithm 1] as "quadrature", to the interpolation of $(g_\sigma)^2$ with algorithm 2.1 as "DCT", and finally to the consistent squaring algorithm algorithm 3.5 as "squaring". For each algorithm, we interpolate a Gaussian $g_\sigma$ with $\sigma = 0.05$, at $n_t = 1000$ points, for various values of $m$.

| | $m = 800$ | $m = 1600$ | $m = 2400$ | $m = 3200$ |
|---|---|---|---|---|
| quadrature | $163.4 \pm 0.9$ | $156.7 \pm 0.9$ | $498.8 \pm 3.5$ | $686.4 \pm 3.1$ |
| DCT | $25.9 \pm 32.9$ | $20.7 \pm 0.2$ | $30.1 \pm 0.6$ | $37.8 \pm 0.5$ |
| squaring | $9.6 \pm 0.1$ | $15.0 \pm 0.1$ | $22.6 \pm 0.1$ | $28.7 \pm 0.2$ |

Putting all things together, we get the Nyström-Chebyshev (NC) method, whose pseudocode can be found in algorithm 3.6.

**Algorithm 3.6:** Nyström-Chebyshev method

**Input:** Symmetric matrix $\mathbf{A} \in \mathbb{R}^{n\times n}$, evaluation points $\{t_i\}_{i=1}^{n_t}$
**Parameters:** Sketch size $n_\Omega$, degree of expansion $m$
**Output:** Approximate evaluations of the spectral density $\{\widehat{\phi}_\sigma^{(m)}(t_i)\}_{i=1}^{n_t}$

Compute $\{\mu_l(t_i)\}_{l=0}^{m}$ for all $t_i$ using algorithm 2.1
Compute $\{\nu_l(t_i)\}_{l=0}^{2m}$ for all $t_i$ using algorithm 3.5
Generate standard Gaussian sketching matrix $\mathbf{\Omega} \in \mathbb{R}^{n\times n_\Omega}$
Initialize $[\mathbf{V}_1, \mathbf{V}_2, \mathbf{V}_3] \leftarrow [\mathbf{0}_{n\times n_\Omega}, \mathbf{\Omega}, \mathbf{0}_{n\times n_\Omega}]$
Initialize $[\mathbf{K}_1(t_i), \mathbf{K}_2(t_i)] \leftarrow [\mathbf{0}_{n_\Omega\times n_\Omega}, \mathbf{0}_{n_\Omega\times n_\Omega}]$ for all $t_i$
Set $\widehat{\phi}_\sigma^{(m)}(t_i) \leftarrow 0$ for all $t_i$
**for** $l = 0, \dots, 2m$ **do**
    $\mathbf{X} \leftarrow \mathbf{\Omega}^\top \mathbf{V}_2$
    **for** $i = 1, \dots, n_t$ **do**
        **if** $l \leqslant m$ **then**
            $\mathbf{K}_1(t_i) \leftarrow \mathbf{K}_1(t_i) + \mu_l(t_i)\mathbf{X}$
        $\mathbf{K}_2(t_i) \leftarrow \mathbf{K}_2(t_i) + \nu_l(t_i)\mathbf{X}$
    $\mathbf{V}_3 \leftarrow (2 - \delta_{l0})\mathbf{A}\mathbf{V}_2 - \mathbf{V}_1$ ▷ Chebyshev recurrence (2.3)
    $\mathbf{V}_1 \leftarrow \mathbf{V}_2, \mathbf{V}_2 \leftarrow \mathbf{V}_3$
**for** $i = 1, \dots, n_t$ **do**
    Compute $\widehat{\phi}_\sigma^{(m)}(t_i) \leftarrow \mathrm{Tr}\left(\mathbf{K}_1(t_i)^\dagger \mathbf{K}_2(t_i)\right)$

Again denoting the cost of a matrix-vector product of $\mathbf{A} \in \mathbb{R}^{n\times n}$ with $c(n)$, e.g. $\mathcal{O}(c(n)) = n^2$ for dense and $\mathcal{O}(c(n)) = n$ for sparse matrices, we find the

computational complexity of the NC method to be $\mathcal{O}(m\log(m)n_t + mn_\Omega^2 n + mn_t n_\Omega^2 + mc(n)n_\Omega + n_t n_\Omega^3)$, with $\mathcal{O}(mn_t + nn_\Omega + n_\Omega^2 n_t)$ required additional storage.

### 3.3.1 IMPLEMENTATION DETAILS

In this section, we propose a sequence of algorithmic improvements to algorithm 3.6, which make the said algorithm more robust and accurate.

**Non-zero checking** In algorithm 3.6, we have that

$$\mathbf{K}_1(t_i) = \boldsymbol{\Omega}^\top g_\sigma^{(m)}(t_i\mathbf{I}_n - \boldsymbol{A})\boldsymbol{\Omega}. \tag{3.35}$$

Hence, if $g_\sigma^{(m)}(t_i\mathbf{I}_n - \boldsymbol{A})$ is close to the zero matrix, which we have seen in section 3.3 to happen when $t_i$ is far away from any of the eigenvalues of $\boldsymbol{A}$, $\mathbf{K}_1(t_i)$ will also be close to the zero matrix. In this case, it may not be a good idea to compute the pseudoinverse of $\mathbf{K}_1(t_i)$ in line 16 of algorithm 3.6. Even less, since in that case we already know that

$$\phi_\sigma^{(m)}(t_i) = \operatorname{Tr}(g_\sigma^{(m)}(t_i\mathbf{I}_n - \boldsymbol{A})) \approx \operatorname{Tr}(\mathbf{0}) = 0. \tag{3.36}$$

Motivated by this observation we use a "non-zero" check which computes the $n_\Omega$-query Girard-Hutchinson estimate (2.9)

$$H_{n_\Omega}(g_\sigma^{(m)}(t_i\mathbf{I}_n - \boldsymbol{A})) = \frac{1}{n_\Omega}\operatorname{Tr}(\mathbf{K}_1(t_i)) \tag{3.37}$$

of $\operatorname{Tr}(g_\sigma^{(m)}(t_i\mathbf{I}_n - \boldsymbol{A}))$ before executing line 16 in algorithm 3.6. If the result is smaller than a fixed non-zero check threshold $\kappa > 0$, it immediately sets $\widetilde{\phi}_\sigma^{(m)}(t_i) = 0$ and skips line 16 for this $t_i$. In order to not accidentally remove parts of the spectrum with eigenvalues, $\kappa$ should not exceed $1/(n\sqrt{2\pi\sigma^2})$. We usually use $\kappa = 10^{-5}$. Since $\mathbf{K}_1(t_i)$ needs to be computed in any case in algorithm 3.6, this check can be incorporated with minimal additional computational cost. The effect this non-zero check mechanism has on the approximation quality can be seen in figure 3.5.

**Pseudoinverse via eigenvalue problem** There exists an alternative way of computing the result of line 16 in algorithm 3.6, namely traces of the form

$$\operatorname{Tr}((\boldsymbol{\Omega}^\top\mathbf{B}\boldsymbol{\Omega})^\dagger(\boldsymbol{\Omega}^\top\mathbf{B}^2\boldsymbol{\Omega})). \tag{3.38}$$

Rather than explicitly forming the pseudoinverse, it converts the problem of computing such a trace into solving a generalized eigenvalue problem by making use of the following theorem [27, theorem 3].

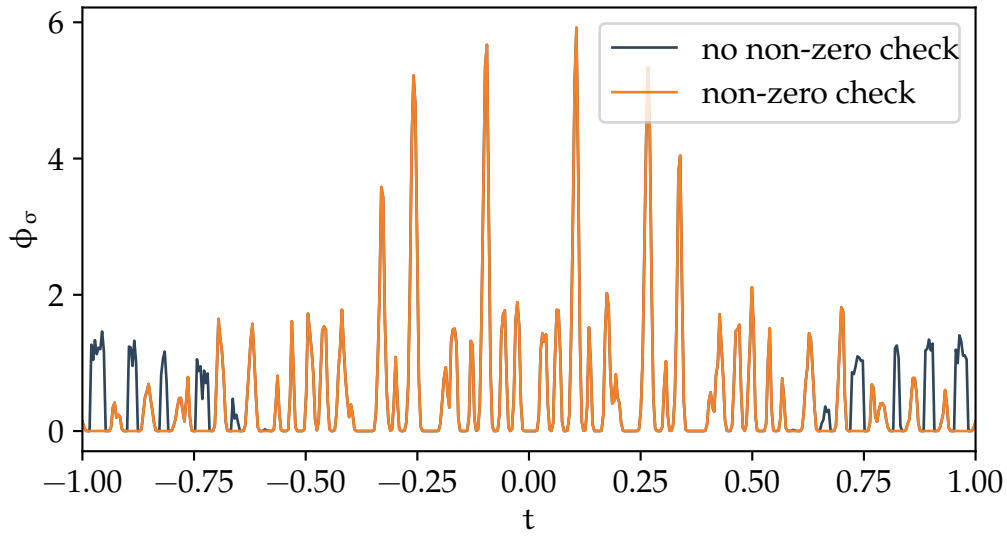



FIGURE 3.5 – The difference the non-zero check can make when approximating a spectral density using algorithm 3.6. Here, we used the matrix described in section 5.1 and ran the NC method with and without non-zero check threshold $\kappa = 10^{-5}$, $m = 2000$, $n_\Omega = 80$, and a Gaussian $g_\sigma$ with $\sigma = 0.05$.

**Theorem 3.7: Generalized eigenvalue problem to compute pseudoinverse**

Let $\mathbf{B} \in \mathbb{R}^{n\times n}$ be symmetric PSD with rank $r < n$ and spectral decomposition (3.10)

$$\mathbf{B} = \begin{bmatrix}\mathbf{V}_1 & \mathbf{V}_2\end{bmatrix}\begin{bmatrix}\mathbf{\Sigma}_1 & \mathbf{0}\\ \mathbf{0} & \mathbf{0}\end{bmatrix}\begin{bmatrix}\mathbf{V}_1^\top\\ \mathbf{V}_2^\top\end{bmatrix} = \mathbf{V}_1\mathbf{\Sigma}_1\mathbf{V}_1^\top, \tag{3.39}$$

where $\mathbf{\Sigma}_1 \in \mathbb{R}^{r\times r}$ and $\mathbf{V}_1 \in \mathbb{R}^{n\times r}$ with $\mathbf{V}_1^\top\mathbf{V}_1 = \mathbf{I}_r$. For $\mathbf{\Omega} \in \mathbb{R}^{n\times n_\Omega}, n_\Omega > r$, such that $\mathbf{\Omega}^\top\mathbf{V}_1$ has linearly independent columns, the solution of the generalized eigenvalue problem

$$(\mathbf{\Omega}^\top\mathbf{B}^2\mathbf{\Omega})\mathbf{C} = (\mathbf{\Omega}^\top\mathbf{B}\mathbf{\Omega})\mathbf{C}\mathbf{\Xi} \tag{3.40}$$

with $\mathbf{C} \in \mathbb{R}^{n_\Omega\times r}$ and $\mathbf{\Xi} \in \mathbb{R}^{r\times r}$ a non-zero diagonal matrix is $\mathbf{C} = (\mathbf{V}_1^\top\mathbf{\Omega})^\dagger\mathbf{\Theta}$ and $\mathbf{\Xi} = \mathbf{\Theta}^\top\mathbf{\Sigma}_1\mathbf{\Theta}$ for a permutation matrix $\mathbf{\Theta} \in \mathbb{R}^{r\times r}$.
Furthermore,

$$\mathrm{Tr}((\mathbf{\Omega}^\top\mathbf{B}\mathbf{\Omega})^\dagger(\mathbf{\Omega}^\top\mathbf{B}^2\mathbf{\Omega})) = \mathrm{Tr}(\mathbf{\Xi}). \tag{3.41}$$

A proof of this theorem can be found in [27, theorem 3]. We choose to still include our own version which goes slightly more into detail.

*Proof.* Since $\mathbf{\Omega}^\top\mathbf{V}_1$ has linearly independent columns, its pseudoinverse is its left inverse, meaning $(\mathbf{\Omega}^\top\mathbf{V}_1)^\dagger(\mathbf{\Omega}^\top\mathbf{V}_1) = (\mathbf{V}_1^\top\mathbf{\Omega})(\mathbf{V}_1^\top\mathbf{\Omega})^\dagger = \mathbf{I}_r$. We use $\mathbf{B} = \mathbf{V}_1\mathbf{\Sigma}_1\mathbf{V}_1^\top$

and insert $\mathbf{C} = (\mathbf{V}_1^\top \boldsymbol{\Omega})^\dagger \boldsymbol{\Theta}^\top$ and $\boldsymbol{\Xi} = \boldsymbol{\Theta}\boldsymbol{\Sigma}_1\boldsymbol{\Theta}^\top$ into the left-hand side of (3.40) to get

$$(\boldsymbol{\Omega}^\top \mathbf{V}_1)\boldsymbol{\Sigma}_1^2 \underbrace{(\mathbf{V}_1^\top \boldsymbol{\Omega})(\mathbf{V}_1^\top \boldsymbol{\Omega})^\dagger}_{=\mathbf{I}_r} \boldsymbol{\Theta} = (\boldsymbol{\Omega}^\top \mathbf{V}_1)\boldsymbol{\Sigma}_1^2\boldsymbol{\Theta}, \tag{3.42}$$

and on the right-hand side of (3.40) for

$$(\boldsymbol{\Omega}^\top \mathbf{V}_1)\boldsymbol{\Sigma}_1 \underbrace{(\mathbf{V}_1^\top \boldsymbol{\Omega})(\mathbf{V}_1^\top \boldsymbol{\Omega})^\dagger}_{=\mathbf{I}_r} \underbrace{\boldsymbol{\Theta}\boldsymbol{\Theta}^\top}_{=\mathbf{I}_r} \boldsymbol{\Sigma}_1\boldsymbol{\Theta} = (\boldsymbol{\Omega}^\top \mathbf{V}_1)\boldsymbol{\Sigma}_1^2\boldsymbol{\Theta}. \tag{3.43}$$

Now that $\boldsymbol{\Omega}^\top \mathbf{V}_1$ has linearly independent columns, $\boldsymbol{\Sigma}_1$ is a non-zero diagonal matrix, and $\boldsymbol{\Theta}$ is a permutation, we conclude that the given $\mathbf{C}$ and $\boldsymbol{\Xi}$ indeed solve the generalized eigenvalue problem. By the uniqueness of generalized eigenvalues and eigenvectors (up to permutation and scaling), these are indeed the only solutions.

(3.41) can be shown by inserting the truncated spectral decomposition (3.39) and using the properties of the pseudoinverse

$$\begin{aligned}
&\operatorname{Tr}((\boldsymbol{\Omega}^\top \mathbf{B}\boldsymbol{\Omega})^\dagger(\boldsymbol{\Omega}^\top \mathbf{B}^2\boldsymbol{\Omega})) \\
&= \operatorname{Tr}((\mathbf{V}_1^\top \boldsymbol{\Omega})^\dagger \boldsymbol{\Sigma}_1^{-1}(\boldsymbol{\Omega}^\top \mathbf{V}_1)^\dagger(\boldsymbol{\Omega}^\top \mathbf{V}_1)\boldsymbol{\Sigma}_1^2(\mathbf{V}_1^\top \boldsymbol{\Omega})) && \text{(spectral decomposition of } \mathbf{B}\text{)} \\
&= \operatorname{Tr}(\boldsymbol{\Sigma}_1^{-1}(\boldsymbol{\Omega}^\top \mathbf{V}_1)^\dagger(\boldsymbol{\Omega}^\top \mathbf{V}_1)\boldsymbol{\Sigma}_1^2(\mathbf{V}_1^\top \boldsymbol{\Omega})(\mathbf{V}_1^\top \boldsymbol{\Omega})^\dagger) && \text{(cyclic property of trace)} \\
&= \operatorname{Tr}(\boldsymbol{\Sigma}_1^{-1} \underbrace{(\boldsymbol{\Omega}^\top \mathbf{V}_1)^\dagger(\boldsymbol{\Omega}^\top \mathbf{V}_1)}_{=\mathbf{I}_r} \boldsymbol{\Sigma}_1^2 \underbrace{(\mathbf{V}_1^\top \boldsymbol{\Omega})(\mathbf{V}_1^\top \boldsymbol{\Omega})^\dagger}_{=\mathbf{I}_r}) && (\boldsymbol{\Omega}^\top \mathbf{V}_1 \text{ independent columns}) \\
&= \operatorname{Tr}(\boldsymbol{\Sigma}_1) && (\boldsymbol{\Sigma}_1^{-1}\boldsymbol{\Sigma}_1^2 = \boldsymbol{\Sigma}_1) \\
&= \operatorname{Tr}(\boldsymbol{\Xi}) && (\boldsymbol{\Sigma}_1 = \boldsymbol{\Xi} \text{ up to permutation})
\end{aligned}$$

□

A standard way of computing the generalized eigenvalue problem (3.40) starts with taking a spectral decomposition of the right-hand side

$$\boldsymbol{\Omega}^\top \mathbf{B}\boldsymbol{\Omega} = \mathbf{W}\boldsymbol{\Gamma}\mathbf{W}^\top = \begin{bmatrix} \mathbf{W}_1 & \mathbf{W}_2 \end{bmatrix} \begin{bmatrix} \boldsymbol{\Gamma}_1 & \mathbf{0} \\ \mathbf{0} & \boldsymbol{\Gamma}_2 \end{bmatrix} \begin{bmatrix} \mathbf{W}_1^\top \\ \mathbf{W}_2^\top \end{bmatrix} \tag{3.44}$$

where $\boldsymbol{\Gamma}_1$ only contains those eigenvalues $\gamma_1, \ldots, \gamma_{n_\Omega}$ which satisfy $\gamma_i \geqslant \zeta \max_{j=1,\ldots,n_\Omega} \gamma_j$ for some threshold factor $\zeta > 0$ and $\mathbf{W}_1$ the corresponding eigenvectors. In our experiments, we choose $\zeta = 10^{-7}$. It allows us to approximate (3.40) with the more stable standard eigenvalue problem

$$\boldsymbol{\Gamma}_1^{-1/2}\mathbf{W}_1^\top(\boldsymbol{\Omega}^\top \mathbf{B}^2\boldsymbol{\Omega})\mathbf{W}_1\boldsymbol{\Gamma}_1^{-1/2}\mathbf{X} = \mathbf{X}\boldsymbol{\Xi}, \tag{3.45}$$

which projects $\boldsymbol{\Omega}^\top \mathbf{B}^2\boldsymbol{\Omega}$ onto the space spanned by the dominant eigenvectors of $\boldsymbol{\Omega}^\top \mathbf{B}\boldsymbol{\Omega}$.

**Filter tolerance** Since by theorem 3.7 the diagonal of $\Xi(t)$ should merely be a permutation of the eigenvalues of $g_\sigma(t\mathbf{I}_n - \mathbf{A})$, we expect its elements to be within the range of $g_\sigma$. Hence, for Gaussian $g_\sigma$ we may remove all elements in $\Xi(t)$ which are outside of $[0, 1/(n\sqrt{2\pi\sigma^2})]$, with size of the matrix $n$ of $\mathbf{A}$ and smoothing parameter $\sigma$ of $g_\sigma$, as suggested in [27]. In this way, we can filter out computational artefacts which for example might result from an inaccurate Chebyshev expansion and furthermore can enforce non-negativity of the resulting approximation of $\phi_\sigma$. However, one needs to be careful not to accidentally remove a valid element from $\Xi(t)$. In the case where $t$ coincides with an eigenvalue of $\mathbf{A}$, one of the elements in $\Xi(t)$ should correctly assume the value $1/(n\sqrt{2\pi\sigma^2})$. Hence, if due to certain numerical inaccuracies this value slightly exceeds the above threshold, it will mistakenly get removed. This problematic case can be avoided by introducing a filter tolerance $\eta > 0$, such that only elements in $\Xi(t)$ which exceed $(1+\eta)/(n\sqrt{2\pi\sigma^2})$ are filtered out. We usually choose $\eta = 10^{-3}$. To ensure non-negativity, the filter tolerance is only applied from above. Figure 3.6 shows the improvement this filter tolerance gives over the standard approach. Close to the edges of the spectrum, there are particularly many inappropriate filterings happening, which are visible as downward-spikes of the spectral density.

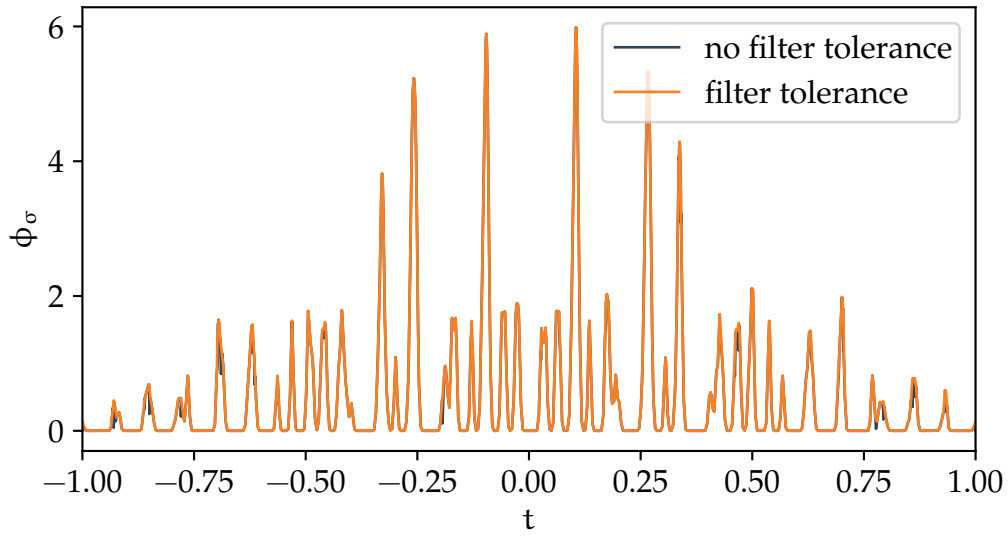



FIGURE 3.6 – The difference the filter tolerance can make when approximating a spectral density using algorithm 3.6. Here, we used the matrix described in section 5.1 and ran the NC method with and without filter tolerance. We use filter tolerance $\eta = 10^{-3}$, $m = 2000$, $n_\Omega = 80$, and a Gaussian $g_\sigma$ with $\sigma = 0.05$.

We can summarize this alternative way of treating line 16 of algorithm 3.6 in the following algorithm.

**Algorithm 3.8: Trace through generalized eigenvalue problem**

**Input:** Matrices $\mathbf{K}_1, \mathbf{K}_2 \in \mathbb{R}^{n_\Omega \times n_\Omega}$
**Parameters:** threshold factor $\zeta$, filter tolerance $\eta$
**Output:** $\mathrm{Tr}(\mathbf{K}_1^\dagger \mathbf{K}_2)$

1: Compute spectral decomposition $\mathbf{K}_1 = \mathbf{W}\mathbf{\Gamma}\mathbf{W}^\top$
2: Let $\mathbf{\Gamma}_1$ only contain the eigenvalues $\gamma_1, \dots, \gamma_{n_\Omega}$ which are larger than $\zeta \max_{i=1,\dots,n_\Omega} \gamma_i$ and $\mathbf{W}_1$ the corresponding eigenvectors
3: Solve eigenvalue problem $\mathbf{\Gamma}_1^{-1/2}\mathbf{W}_1^\top \mathbf{K}_2 \mathbf{W}_1 \mathbf{\Gamma}_1^{-1/2}\mathbf{X} = \mathbf{X}\mathbf{\Xi}$
4: Set all values in $\mathbf{\Xi}$ which are outside of $[0, (1+\eta)/(n\sqrt{2\pi\sigma^2})]$ to zero
5: Compute $\mathrm{Tr}(\mathbf{\Xi})$

In the end, this procedure slightly improves the accuracy and only for small $m$, i.e. when the Chebyshev expansion has not converged yet. This is shown with an example in figure 3.7.

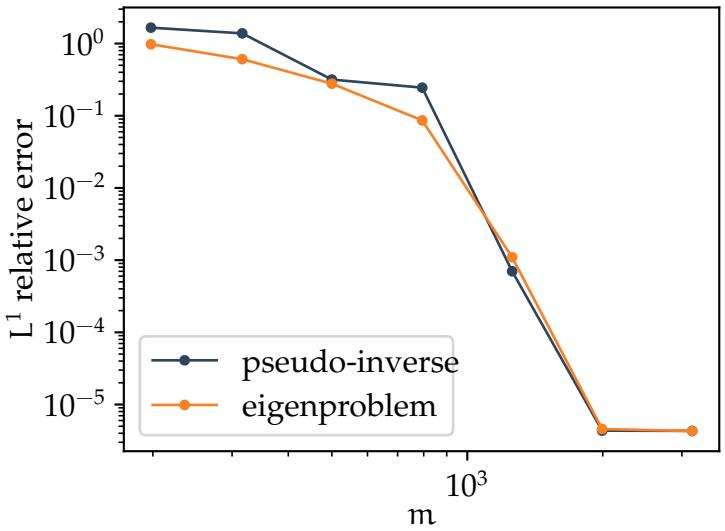



FIGURE 3.7 – The difference between directly computing the pseudoinverse in line 16 of algorithm 2.3 versus solving the eigenvalue problem (3.45). Here, we used the matrix described in section 5.1 and ran the NC method for fixed $n_\Omega = 80$, and a Gaussian $g_\sigma$ with $\sigma = 0.05$.

**Combination of all improvements** In the end, we want to see the combined effect the above proposed algorithmic improvements have. To do so, we observe the accuracy in terms of the $L^1$-error of the approximated spectral density for changing $m$. Once for the raw algorithm 3.6, and once for algorithm 3.6 with non-zero check, computation of the pseudoinverse through an eigenvalue problem, and added filter tolerance.

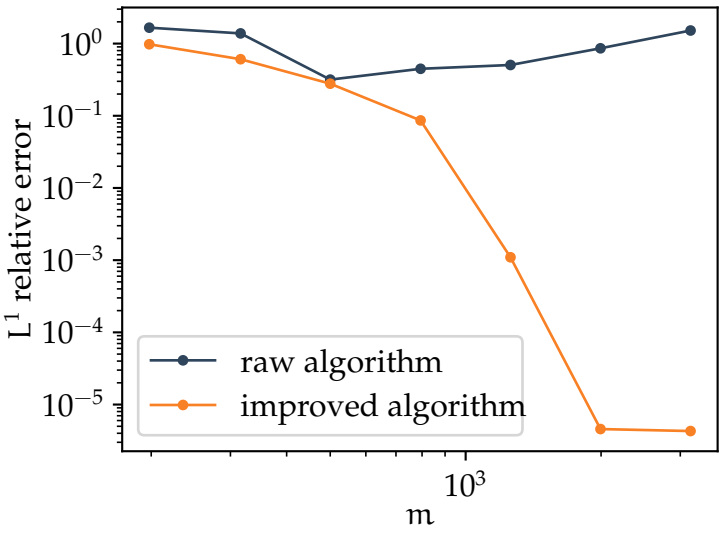



FIGURE 3.8 – The effect the algorithmic improvements have on the accuracy when computing the spectral density. Here, we used the matrix described in section 5.1 and ran the NC method for a Gaussian $g_\sigma$ with $\sigma = 0.05$ and fixed $n_\Omega = 80$. We use threshold factor $\zeta = 10^{-7}$, non-zero check threshold $\kappa = 10^{-5}$, and filter tolerance $\eta = 10^{-3}$.

### 3.3.2 THEORETICAL ANALYSIS

Despite $g_\sigma$ being a non-negative function, its Chebyshev expansion $g_\sigma^{(m)}$ is not guaranteed to retain this property. Therefore, it can happen that the conditions from theorem 3.4 are not verified any more. However, due to lemma 2.4, we know that for large $m$, $g_\sigma^{(m)}$ will be very close to $g_\sigma$. So if we instead interpolate the shifted kernel

$$\underline{g}_\sigma(s) = g_\sigma(s) + \rho, \tag{3.46}$$

we may guarantee that for a large enough shift $\rho \geqslant 0$, its Chebyshev expansion $\underline{g}_\sigma^{(m)}$ will be non-negative. Furthermore, given the shifted smooth spectral density $\underline{\phi}_\sigma$ of a matrix $\mathbf{A} \in \mathbb{R}^{n\times n}$, we may retrieve the smooth spectral density $\phi_\sigma$ using the linearity of the trace [1]:

$$\phi_\sigma(t) = \mathrm{Tr}(g_\sigma(t\mathbf{I}_n - \mathbf{A})) = \mathrm{Tr}(\underline{g}_\sigma(t\mathbf{I}_n - \mathbf{A}) - \rho\mathbf{I}_n) = \underline{\phi}_\sigma(t) - \rho n. \tag{3.47}$$

For the shifted smooth spectral density $\underline{\phi}_\sigma$ we can prove the following result.

[1] For the NC method, this correction will have to be adjusted to $\widehat{\phi}_\sigma(t) = \underline{\phi}_\sigma(t) - \rho n_\Omega$ since in the algorithm we compute the trace of an $n_\Omega \times n_\Omega$ matrix.

**Theorem 3.9: $L^1$-error of Nyström-Chebyshev method with shift**

Let $\widehat{\underline{\phi}}_\sigma^{(m)}$ be the result from running algorithm 3.6 on a symmetric matrix $\mathbf{A} \in \mathbb{R}^{n\times n}$ with its spectrum contained in $[-1,1]$ using a shifted Gaussian smoothing kernel $\underline{g}_\sigma = g_\sigma + \rho$ with smoothing parameter $\sigma > 0$, degree of expansion $m \in \mathbb{N}$, and $n_\Omega = r + p$ for some numbers $r \geqslant 2$, $p \geqslant 4$. If $\rho \geqslant \frac{\sqrt{2}}{n\sigma^2}(1+\sigma)^{-m}$ and $r \geqslant r_{\varepsilon,*}(g_\sigma(t\mathbf{I}_n - \mathbf{A}))$ for all $t \in [-1,1]$, then for all $\gamma \geqslant 1$, the inequality

$$\|\underline{\phi}_\sigma - \widehat{\underline{\phi}}_\sigma^{(m)}\|_1 \leqslant 4\gamma^2(1+r)(\varepsilon + 2\rho(n-r)) + \frac{2\sqrt{2}}{\sigma^2}(1+\sigma)^{-m} \tag{3.48}$$

holds with probability $\geqslant 1 - \gamma^{-p}$.

*Proof.* We assume the conditions of the theorem are satisfied. Then by the triangle inequality

$$\|\underline{\phi}_\sigma - \widehat{\underline{\phi}}_\sigma^{(m)}\|_1 \leqslant \|\underline{\phi}_\sigma - \underline{\phi}_\sigma^{(m)}\|_1 + \|\underline{\phi}_\sigma^{(m)} - \widehat{\underline{\phi}}_\sigma^{(m)}\|_1. \tag{3.49}$$

Since the approximation error of the Chebyshev expansion remains invariant under constant shifts, we deduce with lemma 2.4 that the first term in (3.49) is bounded by

$$\|\underline{\phi}_\sigma - \underline{\phi}_\sigma^{(m)}\|_1 = \|\phi_\sigma - \phi_\sigma^{(m)}\|_1 \leqslant \frac{2\sqrt{2}}{\sigma^2}(1+\sigma)^{-m} \tag{3.50}$$

The second term can be bounded with theorem 3.4:

$$\|\widehat{\underline{\phi}}_\sigma^{(m)} - \underline{\phi}_\sigma^{(m)}\|_1 \leqslant \gamma^2(1+r)\int_{-1}^{1}\sum_{i=r+1}^{n}\underline{\sigma}_i^{(m)}(t)\,dt \tag{3.51}$$

where $\underline{\sigma}_1^{(m)}(t) \geqslant \cdots \geqslant \underline{\sigma}_n^{(m)}(t) \geqslant 0$ are the eigenvalues of $\underline{g}_\sigma^{(m)}(t\mathbf{I}_n - \mathbf{A})$. From the explicit expression for these eigenvalues (3.23) and the condition on $\rho$ based on the approximation error in lemma 2.4 we can deduce that for all $t$

$$\underline{\sigma}_i^{(m)}(t) = \sigma_i^{(m)}(t) + \rho \leqslant \sigma_i(t) + 2\rho \tag{3.52}$$

where $\sigma_i^{(m)}(t)$ stand for the eigenvalues of $g_\sigma^{(m)}(t\mathbf{I}_n - \mathbf{A})$. Consequently

$$\begin{aligned}
\|\widehat{\underline{\phi}}_\sigma - \underline{\phi}_\sigma^{(m)}\|_1 &\leqslant \gamma^2(1+r)\int_{-1}^{1}\sum_{i=r+1}^{n}(\sigma_i(t) + 2\rho)\,dt && \text{((3.52) into (3.51))}\\
&\leqslant \gamma^2(1+r)\left(\int_{-1}^{1}\sum_{i=r+1}^{n}\sigma_i(t)\,dt + 4\rho(n-r)\right) && \text{(linearity)}\\
&\leqslant 2\gamma^2(1+r)\,(2\varepsilon + 4\rho(n-r)) && \text{(choice of } r \geqslant r_{\varepsilon,*}\text{)}
\end{aligned}$$

The result follows by inserting the two derived bounds into (3.49). □

For matrices $\mathbf{A} \in \mathbb{R}^{n \times n}$ whose spectra are approximately evenly distributed, we know from section 3.1 that

$$r_{\varepsilon,*}(g_\sigma(t\mathbf{I}_n - \mathbf{A})) \lessapprox n\sigma\sqrt{-2\log(\sqrt{2\pi n}\sigma\varepsilon)}, \tag{3.53}$$

which gives a tractable expression for the choice of $r$ in theorem 3.9.

As $m$ gets smaller and smaller, the Chebyshev expansion gets more and more accurate, and the shift $\rho$ can be chosen smaller and smaller. However, the non-linearity of most low-rank factorizations make this estimator non-linear, and disallow extending this error guarantee to $\|\phi_\sigma - \widehat{\phi}_\sigma^{(m)}\|$.

From figure 3.9 it is clear that adding the shift harms the convergence of the approximation. Therefore, this shifting operation is only useful to guarantee theoretical results, but in practice it seems like the non-negativity of the smoothing kernel $g_\sigma$ does not have a crucial impact on the performance of the Nyström approximation of the matrix function $g_\sigma(t\mathbf{I}_n - \mathbf{A})$.

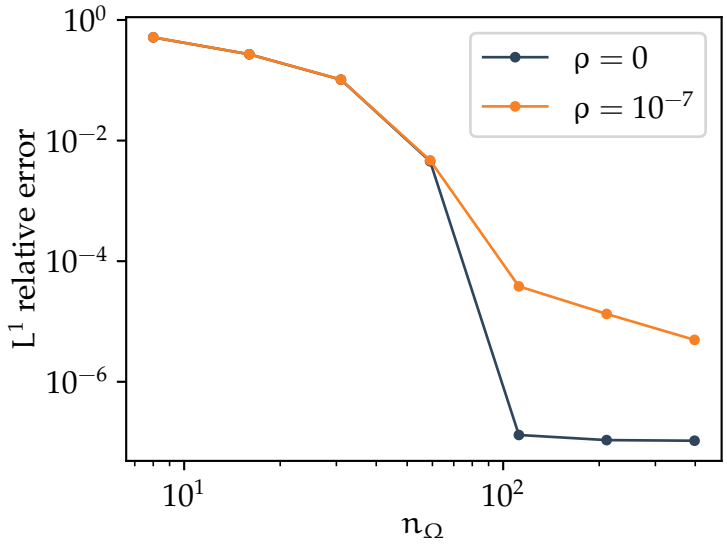



FIGURE 3.9 – For $m = 2400$, we observe the influence of a shift $\rho$ of the kernel on the convergence with respect to $n_\Omega$. Here, we used the matrix described in section 5.1 and ran the NC method (including the algorithmic improvements from section 3.3.1) for a Gaussian $g_\sigma$ with $\sigma = 0.05$ and fixed $n_\Omega = 80$. Once in the usual way and once with $\rho = 10^{-7}$, which is more than sufficient for the shifted kernel $\underline{g}_\sigma$ to be non-negative.

### 3.3.3 EXTENSION TO OTHER LOW-RANK APPROXIMATIONS

Notice that many other low-rank factorization methods [15, 40] also fit into the fast interpolation scheme used to approximate (3.30). In fact, we have found that

if we generalize

$$\mathrm{Tr}^k(\widehat{\mathbf{B}}) = \mathrm{Tr}\left((\boldsymbol{\Omega}^\top \mathbf{B}^k \boldsymbol{\Omega})^\dagger (\boldsymbol{\Omega}^\top \mathbf{B}^{k+1} \boldsymbol{\Omega})\right), \tag{3.54}$$

for $k \in \mathbb{N}$, the underlying low-rank approximation for $k = 1$ is the Nyström approximation which we have discussed in section 3.2. For $k = 2$, it corresponds to the approximation (3.15). To see this, we rewrite it as

$$\widehat{\mathbf{B}} = (\mathbf{B}\boldsymbol{\Omega})((\mathbf{B}\boldsymbol{\Omega})^\top (\mathbf{B}\boldsymbol{\Omega}))^\dagger (\mathbf{B}\boldsymbol{\Omega})^\top \mathbf{B}, \tag{3.55}$$

using the definition of the pseudoinverse. For $k = 3$ we would obtain a scheme which theoretically coincides with the Nyström approximation with subspace iteration [40]

$$\widehat{\mathbf{B}} = (\mathbf{B}^2\boldsymbol{\Omega})(\boldsymbol{\Omega}^\top \mathbf{B}^3 \boldsymbol{\Omega})^\dagger (\mathbf{B}^2\boldsymbol{\Omega})^\top, \tag{3.56}$$

and so on.

Generalizing algorithm 3.6 is straight-forward and due to the efficient exponentiation of Chebyshev polynomials using algorithm 3.5, remains of the same computational and storage complexity, provided we consider $k$ as a small constant.

In practice, however, this extension may not immediately be useful: [40, lemma 5.2] shows that for PSD matrices, approximations corresponding to the case $k = 2$ can be expected not to be as accurate as approximations in the case $k = 1$, and while for $k = 3$ the additional subspace iteration for the Nyström approximation should theoretically produce a better sketch, it suffers from other shortcomings, such as a worse conditioning of the generalized eigenvalue problem (algorithm 3.8). A comparison of the accuracy of the extended methods for $k = 1, 2, 3$ can be found in figure 3.10.

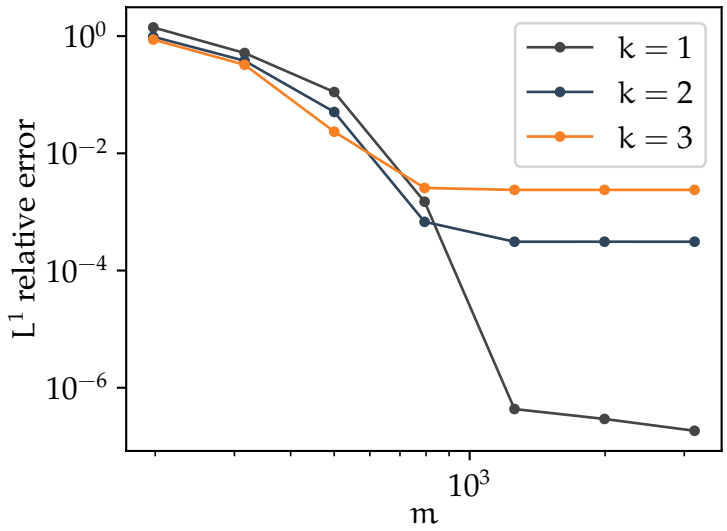



FIGURE 3.10 – Difference in the accuracy when running the NC method developed in this chapter for different types of low-rank factorizations. A 2D model problem from section 5.1 is used for a Gaussian $g_\sigma$ with $\sigma = 0.05$ and we fix $n_\Omega = 80$.

# Chapter 4

# Variance-reduced trace estimation

In section 2.2 we have introduced the classical approach for estimating the trace of a matrix using matrix-vector products. A drawback of this method is the rather slow, reciprocal decrease of the variance with number of Hutchinson's queries $m_\Psi$. Variance-reduced trace estimators try to improve on this. They usually take a "hybrid" approach, combining low-rank factorization with stochastic trace estimation.

This chapter will take the methods we have discussed in chapter 2 and chapter 3, and combine them to an improved trace estimator which we call the Nyström-Chebyshev++ (NC++) method.

## 4.1 Fundamentals of variance-reduced trace estimation

The initial idea from [27] was to perform a low-rank factorization before the stochastic trace estimation to achieve a steeper decrease of the estimation error as the number of computed matrix-vector products increase. Recent theoretical developments confirm the efficacy of this approach [30, 34]. We will first briefly discuss these results which are valid for constant matrices.

### 4.1.1 Constant matrices

Due to the linearity of the trace, we can decompose the trace of any symmetric matrix $\mathbf{B} \in \mathbb{R}^{n\times n}$ into two parts using another matrix $\widehat{\mathbf{B}} \in \mathbb{R}^{n\times n}$

$$\operatorname{Tr}(\mathbf{B}) = \operatorname{Tr}(\widehat{\mathbf{B}}) + \operatorname{Tr}(\mathbf{B} - \widehat{\mathbf{B}}). \tag{4.1}$$

If we manage to find a matrix $\widehat{\mathbf{B}}$, such that the trace of $\widehat{\mathbf{B}}$ can be computed efficiently and $\|\mathbf{B} - \widehat{\mathbf{B}}\|_F$ is small, then we can compute the first term in (4.1) exactly and the second term can be approximated well using the $m_\Psi$-query Girard-

Hutchinson trace estimator $H_{n_\Psi}$ (2.9) due to its small Frobenius norm (2.8). That is,

$$\mathrm{Tr}^{++}(\mathbf{B}) = \mathrm{Tr}(\widehat{\mathbf{B}}) + H_{n_\Psi}(\boldsymbol{\Delta}), \tag{4.2}$$

with the residual $\boldsymbol{\Delta} = \mathbf{B} - \widehat{\mathbf{B}}$, will be an excellent approximation of $\mathrm{Tr}(\mathbf{B})$.

In section 3.2.1 we have introduced some ways in which a matrix $\widehat{\mathbf{B}}$ satisfying the above mentioned criteria can be constructed. For instance, for a symmetric PSD matrix $\mathbf{B} \in \mathbb{R}^{n\times n}$ of rank $r \ll n$, we can compute a factorization of the form $\mathbf{B} = \mathbf{V}_1\boldsymbol{\Sigma}_1\mathbf{V}_1^\top$ (3.10). Since $\mathbf{V}_1$ is orthonormal, $\mathrm{Tr}(\mathbf{B}) = \mathrm{Tr}(\boldsymbol{\Sigma}_1)$ by the cyclic property of the trace, which is easy to compute. On the other hand, $\boldsymbol{\Delta} = \mathbf{B} - \widehat{\mathbf{B}} = \mathbf{0}$ which implies that the Girard-Hutchinson estimator will trivially be exact. Thus, (4.2) would yield an exact estimate. However, this factorization is – in general – prohibitively expensive to compute.

This is where randomized low-rank factorizations come into play. One such example is a factorization of the form (3.15) which is employed in the Hutch++ algorithm [30, algorithm 1]. For PSD matrices this algorithm was shown to give an estimate whose relative deviation from the actual trace is at most $\varepsilon$ by only using $\mathcal{O}(\varepsilon^{-1})$ matrix-vector multiplications, with high probability. The key to this conclusion is the following theorem [30, theorem 1].

**Theorem 4.1: Variance-reduced trace estimation**

Suppose $\mathbf{B} \in \mathbb{R}^{n\times n}$ is symmetric PSD. Let $\widehat{\mathbf{B}}$ and $\boldsymbol{\Delta}$ be any matrices such that

$$\begin{cases} \mathrm{Tr}(\mathbf{B}) = \mathrm{Tr}(\widehat{\mathbf{B}}) + \mathrm{Tr}(\boldsymbol{\Delta}), \\ \|\boldsymbol{\Delta}\|_F \leqslant 2\|\mathbf{B} - \mathbf{B}_{n_\Omega}\|_F. \end{cases} \tag{4.3}$$

where $\mathbf{B}_{n_\Omega}$ is the best rank-$n_\Omega$ approximation to $\mathbf{B}$. For fixed constants $c_1, c_2 > 0$, if $n_\Psi > c_1 \log(1/\delta)$, then with probability $\geqslant 1 - \delta$,

$$|\mathrm{Tr}(\mathbf{B}) - \mathrm{Tr}^{++}(\mathbf{B})| \leqslant 2c_2\sqrt{\frac{\log(1/\delta)}{n_\Psi n_\Omega}}\,\mathrm{Tr}(\mathbf{B}). \tag{4.4}$$

In particular, if $n_\Omega = n_\Psi = \mathcal{O}\left(\sqrt{\log(1/\delta)}/\varepsilon + \log(1/\delta)\right)$, $\mathrm{Tr}^{++}(\mathbf{B})$ is a $(1 \pm \varepsilon)$ error approximation to $\mathrm{Tr}(\mathbf{B})$.

In [30] this theorem is applied to a low-rank factorization of the form (3.15) to prove the $\mathcal{O}(\varepsilon^{-1})$-dependence requirement on the number of matrix-vector multiplications to achieve the relative approximation error $\varepsilon$, which lead to the ubiquitous Hutch++ algorithm. [27] uses and [34] refines an analogous procedure for the case of the Nyström approximation (3.18). In [34, theorem 3.4] it is shown that similarly to the Hutch++ algorithm, the following result holds:

**Theorem 4.2: Error of the Nyström++ trace estimator**

If the trace estimator (4.2) based on the Nyström approximation (3.18) is computed with $n_\Omega = n_\Psi = \mathcal{O}(\sqrt{\log(1/\delta)}/\varepsilon + \log(1/\delta))$ and $\delta \in (0, 1/2)$, then

$$|\operatorname{Tr}(\mathbf{B}) - \operatorname{Tr}^{++}(\mathbf{B})| \leqslant \varepsilon |\operatorname{Tr}(\mathbf{B})| \tag{4.5}$$

holds with probability $\geqslant 1 - \delta$.

### 4.1.2 PARAMETER-DEPENDENT MATRICES

Similarly to the techniques introduced in chapter 2 and chapter 3, also the variance-reduced trace estimation can be extended to the case where the trace of a matrix $\mathbf{B}(t) \in \mathbb{R}^{n\times n}$ which depends continuously on a parameter $t$ needs to be computed.

With a matrix $\widehat{\mathbf{B}}(t)$, the residual $\boldsymbol{\Delta}(t) = \mathbf{B}(t) - \widehat{\mathbf{B}}(t)$, and the parameter-dependent $n_\Psi$-query Girard-Hutchinson estimator $H_{n_\Psi}$ (2.10), we define the estimator

$$\operatorname{Tr}^{++}(\mathbf{B}(t)) = \operatorname{Tr}(\widehat{\mathbf{B}}(t)) + H_{n_\Psi}(\boldsymbol{\Delta}(t)). \tag{4.6}$$

Here, the dependence of $\widehat{\mathbf{B}}(t)$, and therefore also $\boldsymbol{\Delta}(t)$, on a certain number $n_\Omega \in \mathbb{N}$, usually the sketch size $n_\Omega$ of the sketching matrix $\boldsymbol{\Omega}$, is implicitly assumed. Using the result from lemma 2.2, we can – under certain conditions – derive an analogous result to theorem 4.2 for any trace estimate of the form (4.2) in the parameter-dependent case.

**Theorem 4.3: Variance-reduced parameter-dependent trace estimation**

Suppose $\mathbf{B}(t) \in \mathbb{R}^{n\times n}$ is symmetric PSD and continuous in $t \in [a, b]$. Let $\widehat{\mathbf{B}}(t)$ and $\boldsymbol{\Delta}(t)$ be any symmetric PSD and continuous matrices such that

$$\begin{cases} \operatorname{Tr}(\mathbf{B}(t)) = \operatorname{Tr}(\widehat{\mathbf{B}}(t)) + \operatorname{Tr}(\boldsymbol{\Delta}(t)); \\ \int_a^b \|\boldsymbol{\Delta}(t)\|_F dt \leqslant c_\Omega \frac{1}{\sqrt{n_\Omega}} \int_a^b \operatorname{Tr}(\mathbf{B}(t)) dt. \end{cases} \tag{4.7}$$

for some constant $c_\Omega \geqslant 0$. Then, for a fixed constant $c \geqslant 0$ and $\delta \in (0, e^{-1})$, with probability $\geqslant 1 - \delta$,

$$\int_a^b |\operatorname{Tr}(\mathbf{B}(t)) - \operatorname{Tr}^{++}(\mathbf{B}(t))| dt \leqslant c \frac{\log(1/\delta)}{\sqrt{n_\Psi n_\Omega}} \int_a^b \operatorname{Tr}(\mathbf{B}(t)) dt. \tag{4.8}$$

In particular, if $n_\Omega = n_\Psi = \mathcal{O}(\log(1/\delta)/\varepsilon)$, then $\operatorname{Tr}^{++}(\mathbf{B}(t))$ is an $\varepsilon$-error approximation of $\operatorname{Tr}(\mathbf{B}(t))$ in the relative $L^1$-norm.

*Proof.* We may directly bound

$$
\begin{aligned}
&\int_a^b |\operatorname{Tr}(\mathbf{B}(t)) - \operatorname{Tr}^{++}(\mathbf{B}(t))| dt \\
&= \int_a^b |\operatorname{Tr}(\boldsymbol{\Delta}(t)) - H_{n_\Psi}(\boldsymbol{\Delta}(t))| dt && \text{(definition of } \operatorname{Tr}^{++} \text{ and linearity of Tr)} \\
&\leqslant c_\Psi \frac{\log(1/\delta)}{\sqrt{n_\Psi}} \int_a^b \|\boldsymbol{\Delta}(t)\|_F dt && \text{(using lemma 2.2)} \\
&= c_\Psi c_\Omega \frac{\log(1/\delta)}{\sqrt{n_\Psi n_\Omega}} \int_a^b \operatorname{Tr}(\mathbf{B}(t)) dt && \text{(assumption on } \boldsymbol{\Delta}(t))
\end{aligned}
$$

Identifying $c = c_\Psi c_\Omega$, we get the desired result with probability $\geqslant 1 - \delta$. □

When compared to constant matrices (theorem 4.1), the parameter-dependent case (theorem 4.3) requires the residual $\boldsymbol{\Delta}(t)$ to be bounded by $(n_\Omega)^{-1/2}$ times the $L^1$-norm of the trace of $\mathbf{B}(t)$, instead of the best approximation error. We also require an additional factor of $\sqrt{\log(1/\delta)}$ when choosing $n_\Omega$ and $n_\Psi$ such that we can achieve an $\varepsilon$ error.

For the parameter-dependent Nyström approximation in particular (3.20), we can show that it is an approximation which satisfies the conditions of theorem 4.3. The key is to use the following lemma from [17] which guarantees the desired convergence property of the parameter-dependent Nyström approximation.

**Theorem 4.4: $L^1$-error of the parameter-dependent Nyström++ trace estimator**

The parameter-dependent Nyström++ computed with $n_\Omega = n_\Psi = \mathcal{O}(\log(1/\delta)/\varepsilon)$ and even $n_\Omega \geqslant 8\log(1/\delta)$, satisfies for any symmetric PSD matrix $\mathbf{B}(t) \in \mathbb{R}^{n\times n}$ which continuously depends on $t \in [a, b]$, and $\delta \in (0, e^{-1})$, with probability $\geqslant 1 - \delta$

$$
\int_a^b |\operatorname{Tr}(\mathbf{B}(t)) - \operatorname{Tr}^{++}(\mathbf{B}(t))| dt \leqslant \varepsilon \int_a^b \operatorname{Tr}(\mathbf{B}(t)) dt \tag{4.9}
$$

*Proof.* According to theorem 4.3 it is enough to verify that, with the linearity of the trace, the Nyström approximation $\widehat{\mathbf{B}}(t)$ of $\mathbf{B}(t)$ satisfies

$$
\operatorname{Tr}(\mathbf{B}(t)) = \operatorname{Tr}(\widehat{\mathbf{B}}(t) + \mathbf{B}(t) - \widehat{\mathbf{B}}(t)) = \operatorname{Tr}(\widehat{\mathbf{B}}(t)) + \operatorname{Tr}(\boldsymbol{\Delta}(t)) \tag{4.10}
$$

for all $t \in [a, b]$ and with probability $\geqslant 1 - \delta$

$$
\int_a^b \|\boldsymbol{\Delta}(t)\|_F dt = \int_a^b \|\mathbf{B}(t) - \widehat{\mathbf{B}}(t)\|_F dt \leqslant c_\Omega \frac{1}{\sqrt{n_\Omega}} \int_a^b \operatorname{Tr}(\mathbf{B}(t)) dt. \tag{4.11}
$$

The latter follows from lemma 3.3. Finally, the choice of $n_\Omega$ and $n_\Psi$ follows from theorem 4.3 and lemma 3.3. □

Compared to constant matrices (theorem 4.2), we require an additional factor of $\sqrt{\log(1/\delta)}$ in the choice of $n_\Omega$ and $n_\Psi$, and the choice of $\delta$ is slightly more restricted.

## 4.2 THE NYSTRÖM-CHEBYSHEV++ METHOD

Taking the algorithmic developments from chapter 2 and chapter 3, we can easily combine them to a powerful hybrid method which we call the Nyström-Chebyshev++ (NC++) method.

As mentioned at the start of this chapter, this method improves upon the NC method by correcting it with an estimate of the trace of the residual $\boldsymbol{\Delta}(t) = g_\sigma^{(m)}(t\mathbf{I}_n - \boldsymbol{A}) - \widehat{g}_\sigma^{(m)}(t\mathbf{I}_n - \boldsymbol{A})$ of the Nyström approximation. To this purpose, the $n_\Psi$-query Girard-Hutchinson estimator $H_{n_\Psi}$ (2.9) is employed as follows:

$$\begin{aligned}
\check{\phi}_\sigma^{(m)}(t) &= \operatorname{Tr}(\widehat{g}_\sigma^{(m)}(t\mathbf{I}_n - \boldsymbol{A})) + H_{n_\Psi}(\boldsymbol{\Delta}(t)) \\
&= \operatorname{Tr}(\widehat{g}_\sigma^{(m)}(t\mathbf{I}_n - \boldsymbol{A})) + H_{n_\Psi}(g_\sigma^{(m)}(t\mathbf{I}_n - \boldsymbol{A}) - \widehat{g}_\sigma^{(m)}(t\mathbf{I}_n - \boldsymbol{A})) \\
&= \underbrace{\operatorname{Tr}(\widehat{g}_\sigma^{(m)}(t\mathbf{I}_n - \boldsymbol{A}))}_{=\widehat{\phi}_\sigma^{(m)}(t)} + \underbrace{H_{n_\Psi}(g_\sigma^{(m)}(t\mathbf{I}_n - \boldsymbol{A}))}_{=\widetilde{\phi}_\sigma^{(m)}(t)} - H_{n_\Psi}(\widehat{g}_\sigma^{(m)}(t\mathbf{I}_n - \boldsymbol{A})).
\end{aligned} \tag{4.12}$$

It turns out that two of the three terms appearing in (4.12) are already quite familiar to us: In chapter 2 we have seen how to compute $\widetilde{\phi}_\sigma^{(m)}$ with the DGC method, whereas in chapter 3, $\widehat{\phi}_\sigma^{(m)}$ is computed with the NC method. Only the last term is new. Using the standard Gaussian random matrix $\boldsymbol{\Psi}$ from the DGC method, the sketching matrix $\boldsymbol{\Omega}$ from the NC method, and the definition of the Nyström approximation $\widehat{g}_\sigma^{(m)}(t\mathbf{I}_n - \boldsymbol{A})$ (3.29), we may rewrite it as

$$\begin{aligned}
&H_{n_\Psi}(\widehat{g}_\sigma^{(m)}(t\mathbf{I}_n - \boldsymbol{A})) \\
&= \frac{1}{n_\Psi} \operatorname{Tr}\Big((\underbrace{\boldsymbol{\Psi}^\top g_\sigma^{(m)}(t\mathbf{I}_n - \boldsymbol{A})\boldsymbol{\Omega}}_{=\mathbf{L}_1(t)^\top})(\underbrace{\boldsymbol{\Omega}^\top g_\sigma^{(m)}(t\mathbf{I}_n - \boldsymbol{A})\boldsymbol{\Omega}}_{=\mathbf{K}_1(t)})^\dagger(\underbrace{\boldsymbol{\Omega}^\top g_\sigma^{(m)}(t\mathbf{I}_n - \boldsymbol{A})\boldsymbol{\Psi}}_{=\mathbf{L}_1(t)})\Big).
\end{aligned}$$

Notice how the involved matrices $\mathbf{L}_1(t) \in \mathbb{R}^{n_\Omega \times n_\Psi}$ and $\mathbf{K}_1(t) \in \mathbb{R}^{n_\Omega \times n_\Omega}$ again have a form in which they can be expanded efficiently and for all $t$ simultaneously, as we have seen in section 2.3.

The implementation of this new method is similar to the DGC and NC methods (algorithm 2.3 and algorithm 3.6). Although from (4.12) we could see that the result of the Nyström-Chebyshev++ (NC++) method could be obtained by combining the results of the NC and DGC methods, doing so is not efficient in practice. The pseudocode for the Nyström-Chebyshev++ (NC++) method is given in algorithm 4.5.

**Algorithm 4.5:** Nyström-Chebyshev++ method

**Input:** Symmetric matrix $\mathbf{A} \in \mathbb{R}^{n\times n}$, evaluation points $\{t_i\}_{i=1}^{n_t}$
**Parameters:** Sketch size $n_\Omega$, number of Hutchinson's queries $n_\Psi$, degree of expansion $m$
**Output:** Approximate evaluations of the spectral density $\{\breve{\phi}_\sigma^{(m)}(t_i)\}_{i=1}^{n_t}$

Compute $\{\mu_l(t_i)\}_{l=0}^{m}$ for all $t_i$ using algorithm 2.1
Compute $\{\nu_l(t_i)\}_{l=0}^{2m}$ for all $t_i$ using algorithm 3.5
Generate standard Gaussian sketching matrix $\mathbf{\Omega} \in \mathbb{R}^{n\times n_\Omega}$
Generate standard Gaussian random matrix $\mathbf{\Psi} \in \mathbb{R}^{n\times n_\Psi}$
Initialize $[\boldsymbol{V}_1, \boldsymbol{V}_2, \boldsymbol{V}_3] \leftarrow [\mathbf{0}_{n\times n_\Omega}, \mathbf{\Omega}, \mathbf{0}_{n\times n_\Omega}]$
Initialize $[\boldsymbol{W}_1, \boldsymbol{W}_2, \boldsymbol{W}_3] \leftarrow [\mathbf{0}_{n\times n_\Psi}, \mathbf{\Psi}, \mathbf{0}_{n\times n_\Psi}]$
Initialize $[\mathbf{K}_1(t_i), \mathbf{K}_2(t_i)] \leftarrow [\mathbf{0}_{n_\Omega\times n_\Omega}, \mathbf{0}_{n_\Omega\times n_\Omega}]$ for all $t_i$
Initialize $[\mathbf{L}_1(t_i), \ell(t_i)] \leftarrow [\mathbf{0}_{n_\Omega\times n_\Psi}, 0]$ for all $t_i$
Set $\breve{\phi}_\sigma^{(m)}(t_i) \leftarrow 0$ for all $t_i$
**for** $l = 0, \ldots, 2m$ **do**
    $\mathbf{X} \leftarrow \mathbf{\Omega}^\top \boldsymbol{V}_2$
    $\mathbf{Y} \leftarrow \mathbf{\Omega}^\top \boldsymbol{W}_2$
    $z \leftarrow \mathrm{Tr}(\mathbf{\Psi}^\top \boldsymbol{W}_2)$
    **for** $i = 1, \ldots, n_t$ **do**
        **if** $l \leqslant m$ **then**
            $\mathbf{K}_1(t_i) \leftarrow \mathbf{K}_1(t_i) + \mu_l(t_i)\mathbf{X}$
            $\mathbf{L}_1(t_i) \leftarrow \mathbf{L}_1(t_i) + \mu_l(t_i)\mathbf{Y}$
            $\ell(t_i) \leftarrow \ell(t_i) + \mu_l(t_i) z$
        $\mathbf{K}_2(t_i) \leftarrow \mathbf{K}_2(t_i) + \nu_l(t_i)\mathbf{X}$
    $\boldsymbol{V}_3 \leftarrow (2 - \delta_{l0})\mathbf{A}\boldsymbol{V}_2 - \boldsymbol{V}_1$ ▷ Chebyshev recurrence (2.3)
    $\boldsymbol{V}_1 \leftarrow \boldsymbol{V}_2, \boldsymbol{V}_2 \leftarrow \boldsymbol{V}_3$
    $\boldsymbol{W}_3 \leftarrow (2 - \delta_{l0})\mathbf{A}\boldsymbol{W}_2 - \boldsymbol{W}_1$ ▷ Chebyshev recurrence (2.3)
    $\boldsymbol{W}_1 \leftarrow \boldsymbol{W}_2, \boldsymbol{W}_2 \leftarrow \boldsymbol{W}_3$
**for** $i = 1, \ldots, n_t$ **do**
    $\breve{\phi}_\sigma^{(m)}(t_i) \leftarrow \mathrm{Tr}\left(\mathbf{K}_1(t_i)^\dagger \mathbf{K}_2(t_i)\right) + \frac{1}{n_\Psi}\left(\ell(t_i) + \mathrm{Tr}\left(\mathbf{L}_1(t_i)^\top \mathbf{K}_1(t_i)^\dagger \mathbf{L}_1(t_i)\right)\right)$

With the cost of a matrix-vector product denoted by $c(n)$, and supposing we allocate the random vectors equally to the low-rank approximation and the trace estimation, i.e. $n_\Omega \approx n_\Psi$, we determine the computational complexity of the NC++ method to be $\mathcal{O}(m\log(m)n_t + mn_\Omega^2 n + mn_t n_\Omega^2 + mc(n)n_\Omega + n_t n_\Omega^3)$, with $\mathcal{O}(mn_t + nn_\Omega + n_\Omega^2 n_t)$ required additional storage.

It is not hard to extend this method to the other low-rank approximations we have mentioned in section 3.3.3.

### 4.2.1 IMPLEMENTATION DETAILS

All of the implementation details for the DGC method (section 2.3.1) and NC method (section 4.2.1) can be directly translated to the NC++ method.

An interesting and useful observation, which we can make in (3.45), is that by identifying

$$\mathbf{D}(t) = \boldsymbol{W}_1(t)\boldsymbol{\Gamma}_1(t)^{-1/2}\mathbf{X}(t), \tag{4.13}$$

which contains, unlike suggested in [27, algorithm 4], not the generalized eigenvectors from (3.40), we can compute

$$\boldsymbol{\Xi}(t) = \mathbf{D}(t)^\top(\boldsymbol{\Omega}^\top(g_\sigma^{(m)}(t\mathbf{I}_n - \mathbf{A}))^2\boldsymbol{\Omega})\mathbf{D}(t), \tag{4.14}$$

in other words, the matrix whose trace we used in section 3.3.1 to form $\widehat{\phi}_\sigma^{(m)}(t)$ from the Nyström approximation of $\widehat{g}_\sigma^{(m)}(t\mathbf{I}_n - \mathbf{A})$. For consistency between all the terms in (4.12), it is crucial to compute the correction term $H_{n_\Psi}(\widehat{g}_\sigma^{(m)}(t\mathbf{I}_n - \mathbf{A}))$ using the same Girard-Hutchinson estimator, i.e. the same random matrix $\boldsymbol{\Psi}$, which was already used to compute the second term $\widetilde{\phi}_\sigma^{(m)}(t)$, i.e.

$$H_{n_\Psi}(\boldsymbol{\Xi}(t)) = \frac{1}{n_\Psi}\operatorname{Tr}(\boldsymbol{\Psi}^\top\boldsymbol{\Xi}(t)\boldsymbol{\Psi}). \tag{4.15}$$

This can be done quickly and consistently by reusing $\mathbf{D}(t)$ from algorithm 3.8, since by (4.14) and the cyclic property of the trace

$$\begin{aligned}
\operatorname{Tr}(\boldsymbol{\Xi}(t)) &= \operatorname{Tr}(\mathbf{D}(t)^\top(\boldsymbol{\Omega}^\top(g_\sigma^{(m)}(t\mathbf{I}_n - \mathbf{A}))^2\boldsymbol{\Omega})\mathbf{D}(t)) \\
&= \operatorname{Tr}(g_\sigma^{(m)}(t\mathbf{I}_n - \mathbf{A})\boldsymbol{\Omega}\mathbf{D}(t)\mathbf{D}(t)^\top\boldsymbol{\Omega}^\top g_\sigma^{(m)}(t\mathbf{I}_n - \mathbf{A})) \\
&= \mathbb{E}\left[\frac{1}{n_\Psi}\operatorname{Tr}((\underbrace{\boldsymbol{\Psi}^\top g_\sigma^{(m)}(t\mathbf{I}_n - \mathbf{A})\boldsymbol{\Omega}}_{=\mathbf{L}_1(t)^\top})(\mathbf{D}(t)\mathbf{D}(t)^\top)(\underbrace{\boldsymbol{\Omega}^\top g_\sigma^{(m)}(t\mathbf{I}_n - \mathbf{A})\boldsymbol{\Psi}}_{=\mathbf{L}_1(t)}))\right].
\end{aligned}$$

### 4.2.2 THEORETICAL ANALYSIS

Similarly to section 3.3.2, we can again consider the shifted spectral density $\underline{g}_\sigma = g_\sigma + \rho$. For sufficiently large $\rho \geqslant 0$ we may then guarantee that $\underline{g}_\sigma^{(m)}(t\mathbf{I}_n - \mathbf{A})$ is symmetric PSD. Therefore, theorem 4.4 can be employed to get the following result.

**Theorem 4.6: $L^1$-error of Nyström-Chebyshev++ method with shift**

Let $\underline{\breve{\phi}}_\sigma^{(m)}$ be the result from running algorithm 4.5 on a symmetric matrix $\mathbf{A} \in \mathbb{R}^{n\times n}$ with its spectrum contained in $[-1, 1]$ using a shifted Gaussian smoothing kernel $\underline{g}_\sigma = g_\sigma + \rho$ and with the parameters $\sigma > 0$, $m \in \mathbb{N}$, and $n_\Psi + n_\Omega = \mathcal{O}(\log(1/\delta)/\varepsilon)$ with even $n_\Omega \geqslant 8\log(1/\delta)$. If $\rho \geqslant \frac{\sqrt{2}}{n\sigma^2}(1+\sigma)^{-m}$, then

for $\delta \in (0, e^{-1})$ with probability $\geqslant 1 - \delta$

$$\|\underline{\phi}_\sigma - \underline{\breve{\phi}}_\sigma^{(m)}\|_1 \leqslant (1+\varepsilon)\frac{2\sqrt{2}}{\sigma^2}(1+\sigma)^{-m} + \varepsilon(1+2n\rho). \tag{4.16}$$

*Proof.* With the choice of $\rho$ in the assumptions of the theorem, we can conclude from lemma 2.4 that the expansion of the shifted smoothing kernel $\underline{g}_\sigma^{(m)}$ is non-negative. Therefore, the parameter-dependent matrix $\underline{g}_\sigma^{(m)}(t\mathbf{I} - \boldsymbol{A})$ is symmetric PSD, and theorem 4.4 can be applied.

Thus, under the conditions of the theorem, we can conclude

$$\begin{aligned}
&\|\underline{\phi}_\sigma - \underline{\breve{\phi}}_\sigma^{(m)}\|_1 \\
&\leqslant \|\underline{\phi}_\sigma - \underline{\phi}_\sigma^{(m)}\|_1 + \|\underline{\phi}_\sigma^{(m)} - \underline{\breve{\phi}}_\sigma^{(m)}\|_1 && \text{(triangle inequality)} \\
&\leqslant \|\underline{\phi}_\sigma - \underline{\phi}_\sigma^{(m)}\|_1 + \varepsilon\|\underline{\phi}_\sigma^{(m)}\|_1 && \text{(theorem 4.4)} \\
&\leqslant (1+\varepsilon)\|\underline{\phi}_\sigma - \underline{\phi}_\sigma^{(m)}\|_1 + \varepsilon\|\underline{\phi}_\sigma\|_1 && \text{(triangle inequality)} \\
&= (1+\varepsilon)\frac{2\sqrt{2}}{\sigma^2}(1+\sigma)^{-m} + \varepsilon(1+2n\rho) && \text{(lemma 2.4 and } \|\phi_\sigma\|_1 = 1\text{)}
\end{aligned}$$

with probability $\geqslant 1 - \delta$. □

# Chapter 5

# Numerical experiments

In the previous chapters we have developed three methods for approximating the spectral density of a symmetric matrix $\mathbf{A} \in \mathbb{R}^{n \times n}$. All methods compute the trace in (2.13) in slightly different ways. The Delta-Gauss-Chebyshev (DGC) method uses the Girard-Hutchinson stochastic trace estimator (section 2.3), the Nyström-Chebyshev (NC) method computes a randomized low-rank factorization of the matrix function (section 3.3), while the Nyström-Chebyshev++ (NC++) method combines the two aforementioned approaches into a variance-reduced stochastic trace estimator (section 4.2). For each of these methods we have presented a number of algorithmic improvements, which distinguish them from the methods of [27]. In the following numerical experiments – where applicable – we use these improvements with threshold factor $\zeta = 10^{-7}$, filter tolerance $\eta = 10^{-3}$, and non-zero check threshold $\kappa = 10^{-5}$ (section 3.3.1), as we have seen them to be crucial for obtaining a reasonable approximation on the full spectrum figure 3.8.

Our goal is to compare these methods with each other and with other related methods in terms of their accuracy and speed. In order to do so, we apply these algorithms in multiple scenarios. We first consider a model problem from density functional theory [27] and discuss the convergence properties of our algorithms in this setting (section 5.1). Subsequently, we show that the developed methods are also effective for other choices of kernels, for example the Lorentzian kernel (section 5.2). Finally, we test the methods on various other matrices which we have found to be commonly used in literature (section 5.3).

The accuracy is measured in terms of the discrete relative $L^1$-error of the approximated spectral density $\widetilde{\phi}_\sigma^{(m)}$ (also denoted $\widehat{\phi}_\sigma^{(m)}$ and $\check{\phi}_\sigma^{(m)}$) from the spectral

density $\phi_\sigma$ which we obtain using standard eigenvalue solvers[1]:

$$\frac{\sum_{i=1}^{n_t} |\widetilde{\phi}_\sigma^{(m)}(t_i) - \phi_\sigma(t_i)|}{\sum_{i=1}^{n_t} |\phi_\sigma(t_i)|}. \tag{5.1}$$

We use $n_t = 100$ evenly spaced evaluation points which cover the whole spectrum of $\mathbf{A}$. The choice of this metric can be justified by the fact that this error roughly corresponds to the midpoint quadrature rule applied to the continuous $L^1$-norm, for which our theoretical results hold.

## 5.1 MODEL PROBLEM FROM DENSITY FUNCTIONAL THEORY

For our first example, we consider the matrix which arises from the second order finite difference discretization of the Laplace operator $\Delta$ in a potential field $V$,

$$\mathcal{A}u(\mathbf{x}) = -\Delta u(\mathbf{x}) + V(\mathbf{x})u(\mathbf{x}), \tag{5.2}$$

for a uniform mesh of size $h = 0.6$. The potential $V$ results from a lattice whose primitive cell is of side-length $L = 6$ and in whose center a potential

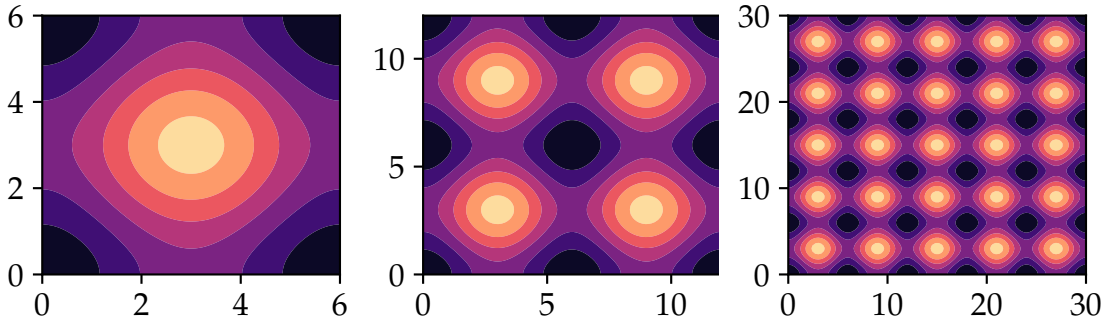
$$\alpha \exp\left(-\frac{\|\mathbf{x}\|_2^2}{2\beta^2}\right) \tag{5.3}$$

with $\alpha = -4$, $\beta = 2$ is located. The computational domain is chosen to span $n_c \in \mathbb{N}$ primitive cells in every spatial dimension, hence, yielding discretization matrices which are growing in size with $n_c$. In our experiments we consider the three-dimensional case, but for visualization purposes, we illustrate the potential in figure 5.1 in two dimensions.

For Gaussian $g_\sigma$ (1.5) with $\sigma = 0.05$ we plot for $n_c = 1$ and two choices of $m$ the convergence of the error with $n_\Omega$ in figure 5.2 and equally for two choices of $n_\Omega + n_\Psi$ the convergence of the error with $m$ in figure 5.2. In our experiments, we always use $n_\Omega = n_\Psi$ for the NC++ method.

In figure 5.2a the Chebyshev expansion is clearly not accurate enough for a good approximation of the spectral density. This is confirmed by figure 5.3: unless a Chebyshev expansion of degree $m \gtrapprox 1000$ is used, we cannot hope for high accuracy approximations. Figure 5.2b allows us to make an interesting observation: The approximation error for the NC++ method first decays quite slowly compared to the NC method. To achieve an $\varepsilon$-error we require $n_\Omega + n_\Psi = \mathcal{O}(\varepsilon^{-1})$ as suggested by theorem 4.3. However, after $n_\Omega + n_\Psi$ exceeds a

[1]We use the standard symmetric eigenvalue solver from the NumPy Python package: https://numpy.org/doc/stable/reference/generated/numpy.linalg.eigvalsh.html.




(A) $n_c = 1$ (B) $n_c = 2$ (C) $n_c = 5$

FIGURE 5.1 – Two dimensional periodic potential $V$ for different sizes $n_c$ of the computational domain.

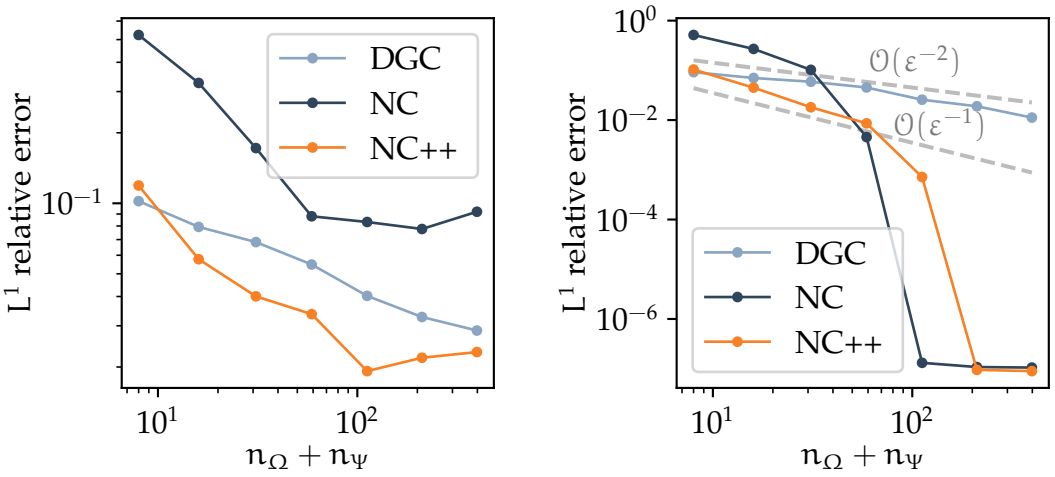
certain value, the approximation error shoots down quickly to where it saturates. The reason is that at this point $n_\Omega$ starts exceeding the $\varepsilon$-numerical rank $r_{\varepsilon,\cdot}$ of the model matrix figure 3.3, which, as a consequence of theorem 3.4, means that the approximation error is expected to be significantly smaller than theorem 4.4 guarantees in general. In fact, it seems that after this point, NC and NC++ behave almost identically, with the exception that the NC uses an $n_\Omega$ which is twice as large as the one in NC++ by design of the experiment, while the contribution from the Girard-Hutchinson correction part in the NC++ method (4.2) seems to be insignificant.

In table 5.1 we list the wall-clock time each method takes to compute an approximate $\phi$ at $n_t = 100$ points for different values of $n_\Omega$ and $m$.

TABLE 5.1 – Comparison of the runtime in seconds of the algorithms applied to the model problem for approximating the smooth spectral density $\phi_\sigma$ with $\sigma = 0.05$ at $n_t = 100$ points for various choices of $m$ and $n_\Omega + n_\Psi$. The mean and standard deviation of 7 runs is given.

| | $m = 800$<br>$n_\Omega + n_\Psi = 40$ | $m = 2400$<br>$n_\Omega + n_\Psi = 40$ | $m = 800$<br>$n_\Omega + n_\Psi = 160$ | $m = 2400$<br>$n_\Omega + n_\Psi = 160$ |
|---|---|---|---|---|
| DGC | $0.295 \pm 0.010$ | $0.866 \pm 0.002$ | $1.067 \pm 0.007$ | $3.184 \pm 0.015$ |
| NC | $1.187 \pm 0.010$ | $3.465 \pm 0.041$ | $4.947 \pm 0.095$ | $14.224 \pm 0.122$ |
| NC++ | $0.940 \pm 0.007$ | $2.773 \pm 0.021$ | $3.329 \pm 0.017$ | $9.992 \pm 0.150$ |

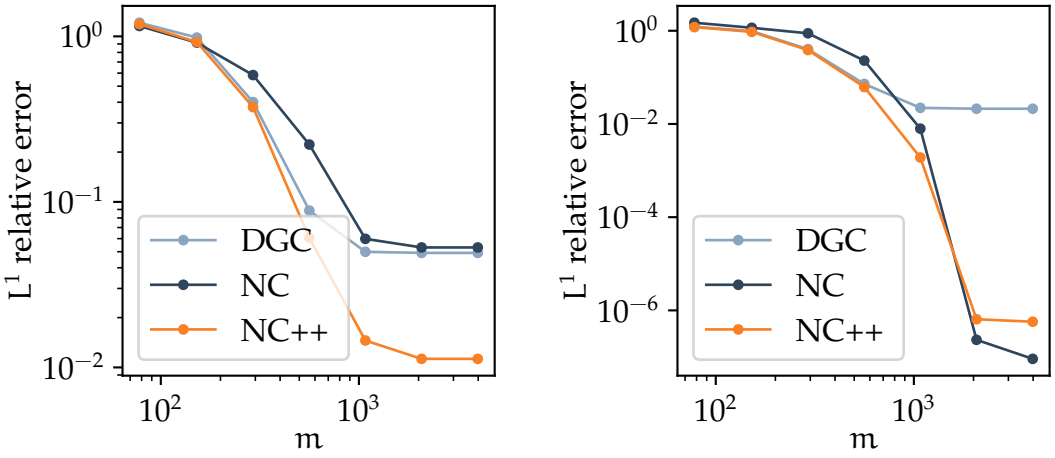
We have seen that the NC++ method is a hybrid method between the DGC and NC methods. In fact, for $n_\Omega = 0$, the NC++ is equivalent to the DGC method, while for $n_\Psi = 0$, it is equivalent to the NC method. Back in chapter 3 we already saw that for small values of $\sigma$ the Nyström approximation will only need a




(A) $m = 800$ (B) $m = 2400$

FIGURE 5.2 – For increasing values of $n_\Omega + n_\Psi$ but fixed $m$ we plot the $L^1$ relative approximation error (5.1) for the model problem with $\sigma = 0.05$. We also indicate orders of $n_\Omega + n_\Psi$ as a function of the relative $L^1$ error $\varepsilon$.

small $n_\Omega$ in order to achieve an accurate approximation. On the other hand, for large choices of $\sigma$ the low-rank approximation will – by itself – not suffice. The interplay between the two parts which make up the NC++ method, on one hand the low-rank approximation and on the other hand the trace estimation on the residual, is illustrated well in figure 5.4. For various values of $\sigma$ and a simultaneously changing $m = 120/\sigma$ to keep an approximately equal expansion accuracy, the behavior of the error for fixed $n_\Omega + n_\Psi = 80$ is plotted.




(A) $n_\Omega + n_\Psi = 40$ (B) $n_\Omega + n_\Psi = 160$

FIGURE 5.3 – For increasing values of $m$ but fixed $n_\Omega + n_\Psi$ we plot the $L^1$ relative approximation error (5.1) for the model problem with $\sigma = 0.05$.

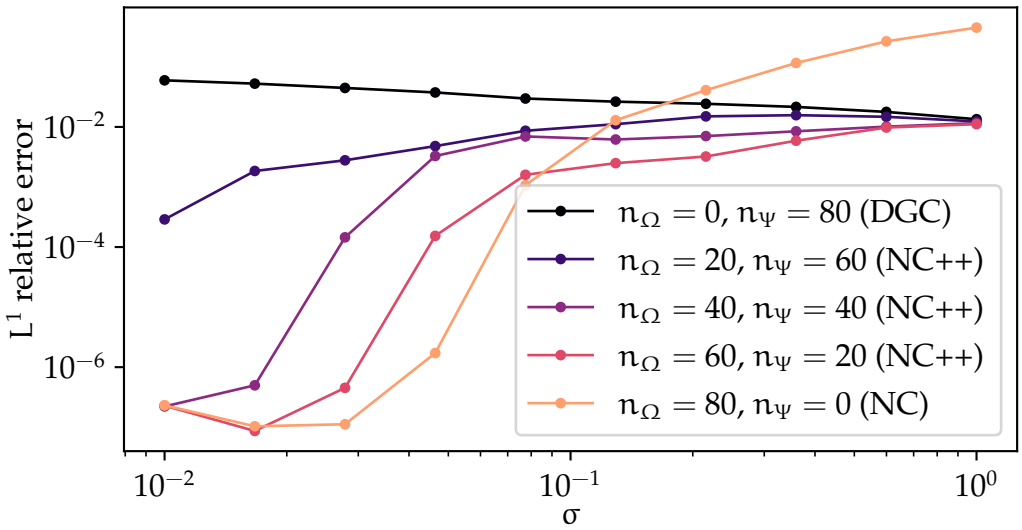



FIGURE 5.4 – The NC++ method for different ways of allocations a total of $n_\Omega + n_\Psi = 80$ random vectors to either the Nyström low-rank approximation or the Girard-Hutchinson trace estimation for the Gaussian $g_\sigma$ with multiple different values of the $\sigma$. We make the approximation error made in the Chebyshev expansion negligible by rescaling $m = 120/\sigma$.

## 5.2 BENCHMARK AGAINST HAYDOCK'S METHOD

Haydock's method [16, 28] is a specialized technique for approximating $\phi_\sigma$ in the case where a Lorentzian smoothing kernel

$$g_\sigma(s) = \frac{1}{\pi}\frac{\sigma}{s^2+\sigma^2} \tag{5.4}$$

is used. A comparison of this kernel with the Gaussian kernel (1.5) is provided in figure 5.5.

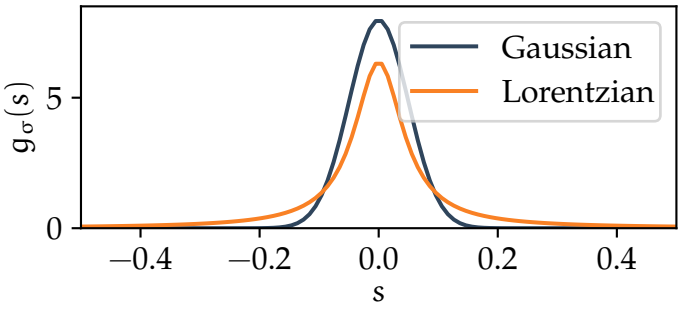
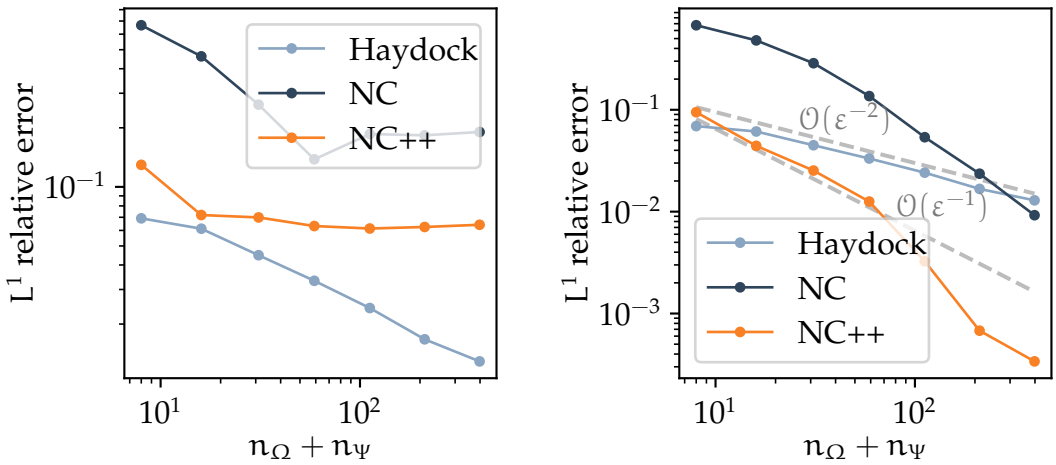



FIGURE 5.5 – Comparison of the Gaussian with the Lorentzian smoothing kernel $g_\sigma$ for $\sigma = 0.05$.

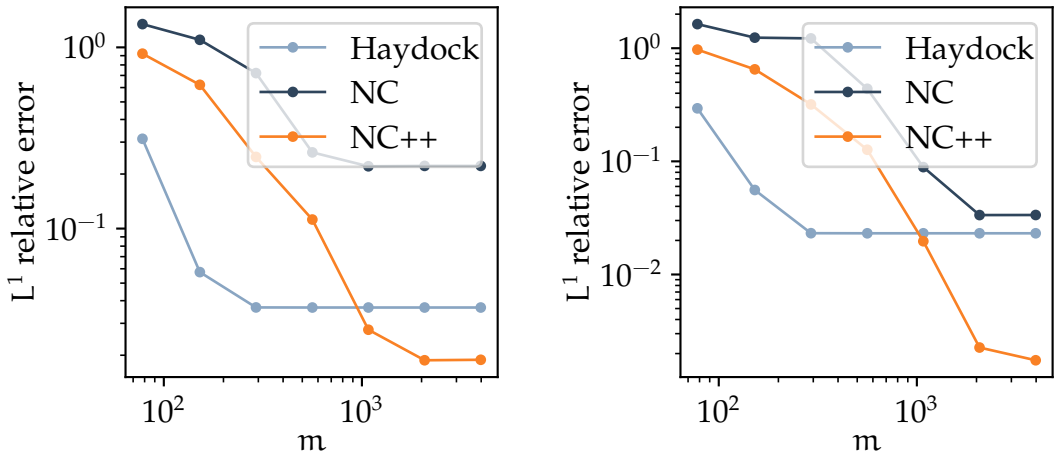
We repeat the same experiments as in section 5.1 but this time for a Lorentzian kernel with Haydock's method to demonstrate that also in the case of non-Gaussian $g_\sigma$ our theoretical guarantees apply. In this method, we use Lanczos with reorthogonalization and fix the number of Lanczos iterations to $m$ and the amount of random vectors used in the Monte-Carlo estimate to $n_\Omega + n_\Psi$. We plot the results in figure 5.6 and figure 5.7, and compare the wall-clock time between the methods in table 5.2.

On one hand the low-rank factorization for the Lorentzian $g_\sigma$ is not as effective as it was for the Gaussian case, since the decay to zero is noticeably slower (see figure 5.5). On the other, the Lorentzian $g_\sigma$ has a pole at $s = \pm\iota$, which has as a consequence that the Chebyshev expansion is not guaranteed to converge as fast as it does in the Gaussian case. Due to these reasons, the convergence of the NC and NC++ methods are slower than they used to be in section 5.1. Nevertheless, this choice of $g_\sigma$ exhibits perfectly the $\mathcal{O}(\varepsilon^{-1})$ and $\mathcal{O}(\varepsilon^{-2})$ dependence of the $L^1$-approximation error on $n_\Psi$ and $n_\Omega$, which the Haydock and NC++ methods respectively show in figure 5.6b.




(A) $m = 800$ (B) $m = 2400$

FIGURE 5.6 – For increasing values of $n_\Omega + n_\Psi$ but fixed $m$ we plot the $L^1$ relative approximation error (5.1) for the model problem from section 5.1 with the Lorentzian kernel with $\sigma = 0.05$. We also indicate orders of $n_\Omega + n_\Psi$ as a function of the relative $L^1$ error $\varepsilon$.




(A) $n_\Omega + n_\Psi = 40$ (B) $n_\Omega + n_\Psi = 160$

FIGURE 5.7 – For increasing values of $m$ but fixed $n_\Omega + n_\Psi$ we plot the $L^1$ relative approximation error (5.1) for the model problem from section 5.1 with the Lorentzian kernel with $\sigma = 0.05$.

TABLE 5.2 – Comparison of the runtime in seconds of the algorithms applied to the model problem from section 5.1 for approximating the smooth spectral density $\phi_\sigma$ with a Lorentzian kernel $g_\sigma$ with $\sigma = 0.05$ at $n_t = 100$ points for various choices of $m$ and $n_\Omega + n_\Psi$. The mean and standard deviation of 7 runs is given.

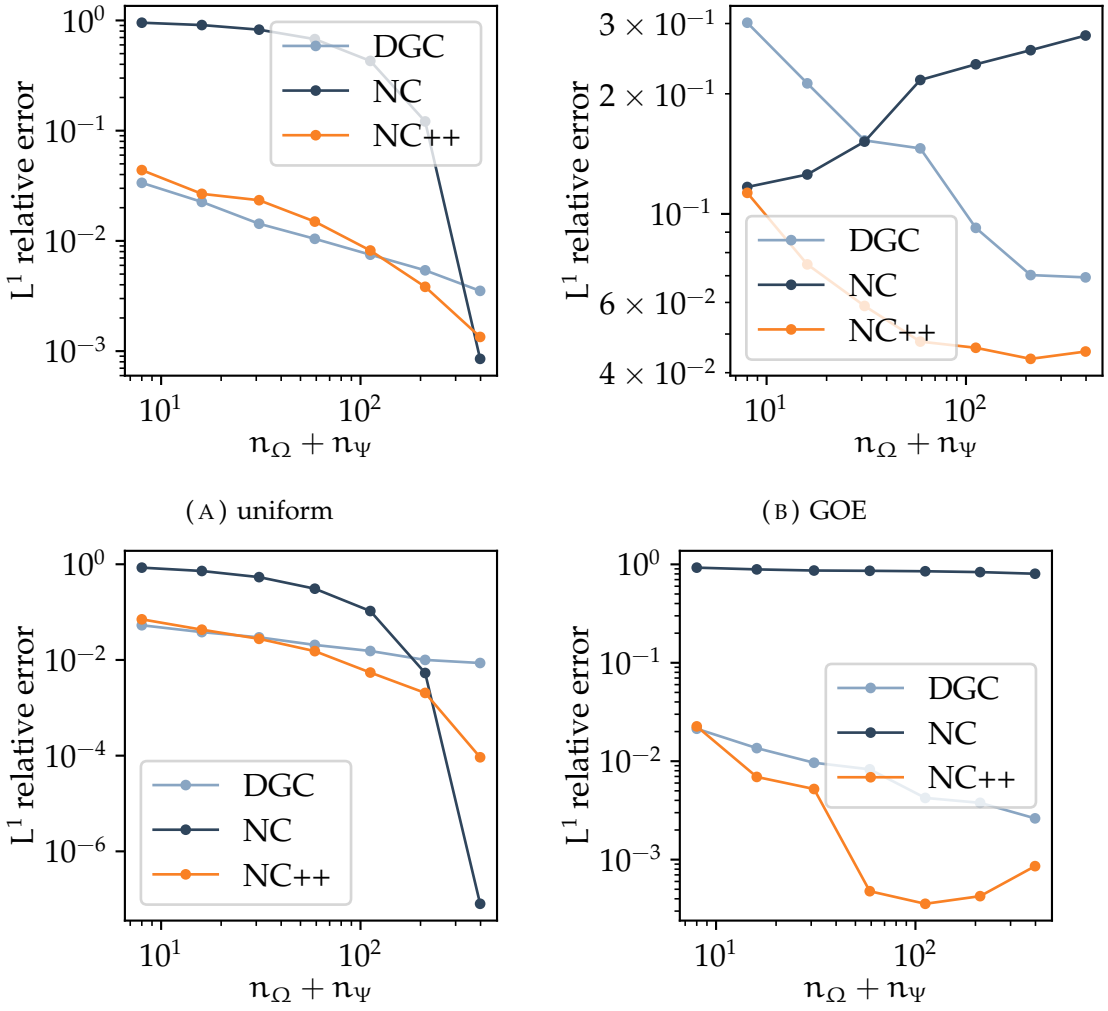
| | $m = 800$<br>$n_\Omega + n_\Psi = 40$ | $m = 2400$<br>$n_\Omega + n_\Psi = 40$ | $m = 800$<br>$n_\Omega + n_\Psi = 160$ | $m = 2400$<br>$n_\Omega + n_\Psi = 160$ |
|---|---|---|---|---|
| Haydock | $5.938 \pm 0.056$ | $12.652 \pm 0.073$ | $24.005 \pm 0.126$ | $50.914 \pm 0.311$ |
| NC | $1.135 \pm 0.013$ | $3.413 \pm 0.083$ | $4.873 \pm 0.077$ | $14.655 \pm 0.187$ |
| NC++ | $0.950 \pm 0.006$ | $2.800 \pm 0.014$ | $3.441 \pm 0.053$ | $9.937 \pm 0.217$ |

## 5.3 EXPERIMENTS WITH VARIOUS MATRICES

We test the algorithms on various problems encountered in literature. We take a synthetic sparse matrix with 2000 uniformly spaced eigenvalues in $[-1, 1]$ [6]; GOE, a matrix $\mathbf{A} = (\mathbf{G} + \mathbf{G}^\top)/\sqrt{2}$ with standard normal $\mathbf{G} \in \mathbb{R}^{1000\times1000}$ from the Gaussian Orthogonal Ensemble; the matrix ModES3D_8, an $8000 \times 8000$ sparse matrix resulting from the same problem as in section 5.1 but with $n_c = 2$, i.e. a larger computational domain [27]; and Erdos992[2], a $6100 \times 6100$ sparse matrix representing the collaboration network of the Hungarian mathematician Pál Erdős from [6]. All these matrices are symmetric. For all of them, we compute, for fixed $m = 2400$ and increasing $n_\Omega + n_\Psi$, the relative $L^1$ approximation error of the spectral density for a Gaussian smoothing kernel $g_\sigma$ with $\sigma = 0.05$. The resulting plots are displayed in figure 5.8.

We observe that for the two matrices which have an evenly distributed spectrum (figure 5.8a) or an approximately evenly distributed spectrum (figure 5.8c), the NC method by itself can achieve a good approximation once $n_\Omega$ exceeds the numerical rank of the matrix. On the other hand, for matrices where the spectrum is very concentrated around a certain point (figure 5.8d) or approximately describes a semi-circle (figure 5.8b) [45], the NC is not as effective, and the correction part in the NC++ makes a significant difference.

[2]Downloaded in the matrix marked format from: https://sparse.tamu.edu/Pajek/Erdos992.




(A) uniform

(B) GOE

(C) ModES3D_8

(D) Erdos992

FIGURE 5.8 – For increasing values of $n_\Omega + n_\Psi$ but fixed $m = 2400$ we plot the $L^1$ relative approximation error (5.1) for multiple different matrices. We use a Gaussian smoothing kernel $g_\sigma$ with $\sigma = 0.05$.

# Chapter 6

# Conclusion

In our work we have developed a family of methods designed for the randomized computation of the spectral density of large matrices, which are closely related to the methods from [27].

We were able to significantly improve many algorithmic aspects of these methods. The development of an alternative expansion framework in section 2.1 allowed us to vastly simplify the Chebyshev expansion stage, which is common to all the studied methods, while obtaining provable accuracy of the expansion, all this in addition to making this stage orders of magnitude faster in most cases. The studied methods were made more robust through a series of well founded implementation strategies section 3.3.1. Furthermore, we give theoretical error guarantees for all the encountered methods building on [17]. All these developments are illustrated and verified in multiple numerical experiments which are all provably reproducible.

We have noticed that the degree of the Chebyshev expansion needs to be quite high to achieve an acceptable expansion accuracy, which slows down our methods considerably. Alternative ways of computing products of matrix functions with random vectors should be taken into consideration [8, 42]. Also, different ways of evaluating line 16 in algorithm 3.6 may still improve the accuracy and stability of our methods. For example [40, algorithm 5.6] may help, but a way of making this compatible with our expansion framework has not been found yet.

On the theoretical side, the incorporation of the Chebyshev expansion into the error bound analogous to theorem 2.5 is of high priority. What hinders such a unification for now is the fact that the Chebyshev expansion is not guaranteed to be non-negative, which in turn breaks the PSD property, which is fundamental to most theoretical results on the Nyström approximation. In section 3.3.2 and section 4.2.2 we have shown a way around this problem by introducing a shift which guarantees the expansion to be non-negative. However, this shift significantly impairs the performance in practice.

# Bibliography


[1] N. Ahmed and P. S. Fisher. Study of algorithmic properties of Chebyshev coefficients. *International Journal of Computer Mathematics*, 2(1-4):307–317, 1968. DOI: 10.1080/00207167008803043.

[2] N. Alon, T. Lee, A. Shraibman, and S. Vempala. The approximate rank of a matrix and its algorithmic applications: Approximate rank. In *Proceedings of the Forty-Fifth Annual ACM Symposium on Theory of Computing*, pages 675–684. Association for Computing Machinery, 2013. DOI: 10.1145/2488608.2488694.

[3] G. Baszenski and M. Tasche. Fast polynomial multiplication and convolutions related to the discrete cosine transform. *Linear Algebra and its Applications*, 252(1):1–25, 1997. DOI: 10.1016/0024-3795(95)00696-6.

[4] G. Behrooz, K. Shankar, and X. Ying. An investigation into neural net optimization via Hessian eigenvalue density. *arXiv*, 2019. DOI: 10.48550/arXiv.1901.10159.

[5] T. Chen. A spectrum adaptive kernel polynomial method. *The Journal of Chemical Physics*, 159(11):114101, 2023. DOI: 10.1063/5.0166678.

[6] T. Chen, T. Trogdon, and S. Ubaru. Analysis of stochastic Lanczos quadrature for spectrum approximation. *arXiv*, 2021. DOI: 10.48550/arXiv.2105.06595.

[7] T. H. Cormen, C. E. Leiserson, and R. L. Rivest. *Introduction to Algorithms*. MIT Press, third edition, 2009. ISBN 978-0-262-53305-8.

[8] A. Cortinovis, D. Kressner, and Y. Nakatsukasa. Speeding up Krylov subspace methods for computing f(a)b via randomization. *arXiv*, 2023. DOI: 10.48550/arXiv.2212.12758.

[9] P. A. M. Dirac. *The Principles of Quantum Mechanics*. Oxford University Press, third edition, 1947.

[10] F. Ducastelle and F. Cyrot-Lackmann. Moments developments and their application to the electronic charge distribution of d bands. *Journal of*

*Physics and Chemistry of Solids*, 31(6):1295–1306, 1970. DOI: 10.1016/0022-3697(70)90134-4.

[11] E. N. Epperly, J. A. Tropp, and R. J. Webber. XTrace: Making the most of every sample in stochastic trace estimation. *arXiv*, 2023. DOI: 10.48550/arXiv.2301.07825.

[12] T. Fan, D. I. Shuman, S. Ubaru, and Y. Saad. Spectrum-adapted polynomial approximation for matrix functions with applications in graph signal processing. *Algorithms*, 13(11), 2020. DOI: 10.3390/a13110295.

[13] C. F. Gauss. *Demonstratio nova theorematis omnem functionem algebraicam rationalem integram unius variabilis in factores reales primi vel secundi gradus resolvi posse*. Nineteenth Century Collections Online (NCCO): Science, Technology, and Medicine: 1780-1925. Fleckeisen, 1799.

[14] A. Gittens and M. W. Mahoney. Revisiting the Nyström method for improved large-scale machine learning. *Journal of Machine Learning Research*, 17(1):3977–4041, 2016. URL http://jmlr.org/papers/v17/gittens16a.html.

[15] N. Halko, P. G. Martinsson, and J. A. Tropp. Finding structure with randomness: Probabilistic algorithms for constructing approximate matrix decompositions. *SIAM Review*, 53(2):217–288, 2011. DOI: 10.1137/090771806.

[16] R. Haydock, V. Heine, and M. J. Kelly. Electronic structure based on the local atomic environment for tight-binding bands. *Journal of Physics C: Solid State Physics*, 5(20):2845, 1972. DOI: 10.1088/0022-3719/5/20/004.

[17] H. He, D. Kressner, H. L. Lam, and F. Matti. Parameter dependent Nystrom++ with application in spectral density function theory. *In preparation*, 2024.

[18] J. Hefferon. *Linear Algebra*. VCU mathematics textbook series. CreateSpace Independent Publishing Platform, 2012. ISBN 979-8362590277.

[19] N. J. Higham. *Functions of Matrices*. Society for Industrial and Applied Mathematics, 2008. DOI: 10.1137/1.9780898717778.

[20] R. A. Horn and C. R. Johnson. *Matrix Analysis*. Cambridge University Press, 1985. DOI: 10.1017/CBO9780511810817.

[21] L. Huang, A. J. Graven, and D. Bindel. Density of states graph kernels. In *Proceedings of the 2021 SIAM International Conference on Data Mining (SDM)*, pages 289–297. Society for Industrial and Applied Mathematics, 2021. DOI: 10.1137/1.9781611976700.33.

[22] M. F. Hutchinson. A stochastic estimator of the trace of the influence matrix for Laplacian smoothing splines. *Communication in Statistics – Simulation and Computation*, 19(2):433–450, 1990. DOI: 10.1080/03610919008812866.

[23] A. Klenke. *Probability Theory*. Springer London, second edition, 2013. DOI: 10.1007/978-1-4471-5361-0.

[24] S. Kruzick and J. M. F. Moura. Graph signal processing: Filter design and spectral statistics. *arXiv*, 2018. DOI: 10.48550/arXiv.1802.10145.

[25] C. Lanczos. An iteration method for the solution of the eigenvalue problem of linear differential and integral operators. *Journal of Research of the National Bureau of Standards*, 45:255–282, 1950. DOI: 10.6028/jres.045.026.

[26] R. Li, Y. Xi, L. Erlandson, and Y. Saad. The eigenvalues slicing library (EVSL): Algorithms, implementation, and software. *SIAM Journal on Scientific Computing*, 41(4):C393–C415, 2019. DOI: 10.1137/18M1170935.

[27] L. Lin. Randomized estimation of spectral densities of large matrices made accurate. *Numerische Mathematik*, 136:183–213, 2017. DOI: 10.1007/s00211-016-0837-7.

[28] L. Lin, Y. Saad, and C. Yang. Approximating spectral densities of large matrices. *SIAM Review*, 58(1):34–65, 2016. DOI: 10.1137/130934283.

[29] J. Makhoul. A fast cosine transform in one and two dimensions. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 28(1):27–34, 1980. DOI: 10.1109/TASSP.1980.1163351.

[30] R. A. Meyer, C. Musco, C. Musco, and D. P. Woodruff. Hutch++: Optimal stochastic trace estimation. *arXiv*, 2021. DOI: 10.48550/arXiv.2010.09649.

[31] L. Mirsky. Symmetric gauge functions and unitarily invariant norms. *The Quarterly Journal of Mathematics*, 11(1):50–59, 1960. DOI: 10.1093/qmath/11.1.50.

[32] G. Patane. Fourier-based and rational graph filters for spectral processing. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PP, 2022. DOI: 10.1109/TPAMI.2022.3177075.

[33] R. Penrose. A generalized inverse for matrices. *Mathematical Proceedings of the Cambridge Philosophical Society*, 51(3):406–413, 1955. DOI: 10.1017/S0305004100030401.

[34] D. Persson, A. Cortinovis, and D. Kressner. Improved variants of the Hutch++ algorithm for trace estimation. *SIAM Journal on Matrix Analysis and Applications*, 43(3):1162–1185, 2022. DOI: 10.1137/21M1447623.

[35] E. Polizzi. Density-matrix-based algorithm for solving eigenvalue problems. *Physical Review B*, 79:115112, 2009. DOI: 10.1103/PhysRevB.79.115112.

[36] A. Sankar, Y. Khasbage, R. Vigneswaran, and V. Balasubramanian. A deeper look at the Hessian eigenspectrum of deep neural networks and its applications to regularization. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35:9481–9488, 05 2021. DOI: 10.1609/aaai.v35i11.17142.

[37] R. N. Silver and H. Röder. Densities of states of mega-dimensional Hamiltonian matrices. *International Journal of Modern Physics C*, 05(04):735–753, 1994. DOI: 10.1142/S0129183194000842.

[38] L. N. Trefethen. Is Gauss quadrature better than Clenshaw-Curtis? *SIAM Review*, 50(1):67–87, 2008. DOI: 10.1137/060659831.

[39] L. N. Trefethen. *Approximation Theory and Approximation Practice*. Society for Industrial and Applied Mathematics, 2019. DOI: 10.1137/1.9781611975949.

[40] J. A. Tropp and R. J. Webber. Randomized algorithms for low-rank matrix approximation: Design, analysis, and applications. *arXiv*, 2023. DOI: 10.48550/arXiv.2306.12418.

[41] J. A. Tropp, A. Yurtsever, M. Udell, and V. Cevher. Practical sketching algorithms for low-rank matrix approximation. *SIAM Journal on Matrix Analysis and Applications*, 38(4):1454–1485, 2017. DOI: 10.1137/17M1111590.

[42] S. Ubaru, J. Chen, and Y. Saad. Fast estimation of $tr(f(a))$ via stochastic Lanczos quadrature. *SIAM Journal on Matrix Analysis and Applications*, 38(4): 1075–1099, 2017. DOI: 10.1137/16M1104974.

[43] L.-W. Wang. Calculating the density of states and optical-absorption spectra of large quantum systems by the plane-wave moments method. *Physical Review B*, 49:10154–10158, 1994. DOI: 10.1103/PhysRevB.49.10154.

[44] A. Weisse, G. Wellein, A. Alvermann, and H. Fehske. The kernel polynomial method. *Reviews of Modern Physics*, 78:275–306, 2006. DOI: 10.1103/RevModPhys.78.275.

[45] E. P. Wigner. On the distribution of the roots of certain symmetric matrices. *Annals of Mathematics*, 67(2):325–327, 1958. DOI: 10.2307/1970008.

[46] D. P. Woodruff. Sketching as a tool for numerical linear algebra. *Foundations and Trends in Theoretical Computer Science*, 10(1–2):1–157, 2014. DOI: 10.1561/0400000060.

[47] Y. Zhou and R.-C. Li. Bounding the spectrum of large Hermitian matrices. *Linear Algebra and its Applications*, 435(3):480–493, 2011. DOI: 10.1016/j.laa.2010.06.034.

# Appendix A

# Spectral transformation

In section 2.3.1 we have mentioned a way in which we can apply a spectral transformation $\tau : [a, b] \to [-1, 1]$ to use our methods for computing spectral densities of matrices whose spectrum is contained in a different interval $[a, b]$. The following derivations prove the validity of this approach.

Let us call $\bar{\phi}$ the spectral density of $\bar{\boldsymbol{A}} = \tau(\boldsymbol{A})$. Then we can relate it to the spectral density $\phi$ of $\boldsymbol{A}$ through

$$
\begin{aligned}
\phi(\tau^{-1}(t)) &= \frac{1}{n}\sum_{i=1}^{n}\delta(\tau^{-1}(t) - \lambda_i) && \text{(definition (1.1))} \\
&= \frac{1}{n}\sum_{i=1}^{n}\delta(\tau^{-1}(t) - \tau^{-1}(\bar{\lambda}_i)) && \text{(transformed eigenvalues } \bar{\lambda}_i = \tau(\lambda_i)) \\
&= \frac{1}{n}\sum_{i=1}^{n}\delta\left(\frac{b-a}{2}(t - \bar{\lambda}_i)\right) && \text{(explicit form of } \tau^{-1}(s) = \frac{b-a}{2}s + \frac{b+a}{2}) \\
&= \frac{2}{b-a}\frac{1}{n}\sum_{i=1}^{n}\delta(t - \bar{\lambda}_i) && \text{(scaling property } \delta(cs) = \frac{1}{|c|}\delta(s), c \in \mathbb{R}) \\
&= \frac{2}{b-a}\bar{\phi}(t) && \text{(definition of } \bar{\phi})
\end{aligned}
$$

such that finally $\phi = \frac{2}{b-a}\bar{\phi} \circ \tau$. Finding an expression for $\phi_\sigma$ is a bit trickier. For this, we need to go back all the way to the definition of $\phi_\sigma$ (1.4) and evaluate the

expression

$$
\begin{aligned}
&\phi_\sigma(\tau^{-1}(t)) \\
&= (\phi * g_\sigma)(\tau^{-1}(t)) && \text{(definition of } \phi_\sigma\text{)} \\
&= \int_{-\infty}^{\infty} \phi(s) g_\sigma(\tau^{-1}(t) - s) ds && \text{(convolution)} \\
&= \frac{2}{b-a} \int_{-\infty}^{\infty} \bar{\phi}(\tau(s)) g_\sigma(\tau^{-1}(t) - s) ds && \text{(identity } \phi = \frac{2}{b-a}\bar{\phi} \circ \tau\text{)} \\
&= \frac{2}{b-a} \int_{-\infty}^{\infty} \bar{\phi}(\bar{s}) g_\sigma\left(\tau^{-1}(t) - \tau^{-1}(\bar{s})\right) \frac{b-a}{2} d\bar{s} && \text{(substitution } \bar{s} = \tau(s)\text{)} \\
&= \frac{2}{b-a} \int_{-\infty}^{\infty} \bar{\phi}(\bar{s}) g_\sigma\left(\frac{b-a}{2}(t - \bar{s})\right) \frac{b-a}{2} d\bar{s} && \text{(explicit form of } \tau^{-1}\text{)}
\end{aligned}
$$

By identifying

$$
\bar{g}_\sigma(s) = g_\sigma\left(\frac{b-a}{2} s\right) \frac{b-a}{2} \tag{A.1}
$$

we see that $\phi_\sigma = \frac{2}{b-a}(\bar{\phi} * \bar{g}_\sigma) \circ \tau$. Therefore, we can obtain $\phi_\sigma$ of $\mathbf{A}$ by running algorithm 2.3 with $\bar{g}_\sigma$ and $\bar{\mathbf{A}}$ on the transformed evaluation points $\{\tau(t_i)\}_{i=1}^{n_t}$, and rescale the result with $\frac{2}{b-a}$. Since all of the smoothing kernel $g_\sigma$ we consider in this thesis (Gaussian, Lorentzian) are of the form $g_\sigma(s) = \frac{1}{\sigma} f(\frac{s}{\sigma})$ for some function $f$ independent of $\sigma$, we only need to rescale the smoothing parameter $\sigma$ to

$$
\bar{\sigma} = \frac{2\sigma}{b-a} \tag{A.2}
$$

to determine $\bar{g}_\sigma = g_{\bar{\sigma}}$.

# Appendix B

# Numerical rank

The $\varepsilon$-numerical rank $r_{\varepsilon,\cdot}$ of the symmetric PSD matrix $g_\sigma(t\mathbf{I}_n - \mathbf{A})$ is used to motivate the introduction of a low-rank factorization in chapter 3 and also appears in the formulation and proof of theorem 3.9. (3.24) provides a tractable expression of $r_{\varepsilon,\cdot}$ for Gaussian $g_\sigma(t\mathbf{I}_n - \mathbf{A})$. We now show how this formula was determined.

For the spectral norm $\|\cdot\|_2$, we calculate

$$
\begin{aligned}
&r_{\varepsilon,2}(g_\sigma(t\mathbf{I}_n - \mathbf{A})) \\
&= \min\{1 \leqslant r \leqslant n : \sigma_{r+1}(t) \leqslant \varepsilon\} && \text{(definition (3.7))} \\
&= \min\left\{1 \leqslant r \leqslant n : \frac{1}{n\sqrt{2\pi\sigma^2}} e^{-\frac{(t-\lambda_{(r+1)})^2}{2\sigma^2}} \leqslant \varepsilon\right\} && \text{(expression (3.23))} \\
&= \min\left\{1 \leqslant r \leqslant n : |t - \lambda_{(r+1)}| \geqslant \sigma\sqrt{-2\log(\varepsilon n\sqrt{2\pi\sigma^2})}\right\} && \text{(rearrangement)} \\
&= \#\left\{1 \leqslant i \leqslant n : |t - \lambda_i| < \sigma\sqrt{-2\log(\varepsilon n\sqrt{2\pi\sigma^2})}\right\} && (|t - \lambda_{(1)}| \leqslant \cdots \leqslant |t - \lambda_{(n)}|)
\end{aligned}
$$

Here, we identify $C_{\varepsilon,2} = \sigma\sqrt{-2\log(\varepsilon n\sqrt{2\pi\sigma^2})}$ to end up with the expression (3.25).

For the nuclear and Frobenius norm we use the fact that

$$
\frac{1}{n}\sqrt{\sum_{i=r+1}^{n} \sigma_i(t)^2} \leqslant \frac{1}{\sqrt{n}} \sum_{i=r+1}^{n} \sigma_i(t) \leqslant \sigma_{r+1}(t) \tag{B.1}
$$

to apply the same reasoning as for the spectral norm above, though for slightly different constants $C_{\varepsilon,*}$ and $C_{\varepsilon,F}$.